

**Aging Gracefully: Economic Participation and the Silver
Economy in India - Using LASI Data 2017-2018 wave**

By

Dr. ASHWIN J V (MBBS, DNB, DM, MBA, MDS)

DISSERTATION

Presented to the Swiss School of Business and Management, Geneva

In Partial Fulfilment

Of the Requirements

For the Degree

DOCTOR OF BUSINESS ADMINISTRATION

**SWISS SCHOOL OF BUSINESS AND MANAGEMENT,
GENEVA**

May 2026

Aging Gracefully: Economic Participation and the Silver Economy in India - Using LASI Data 2017-2018 wave

by

Dr ASHWIN J V



APPROVED BY

Apostolos
Dasilas

Dissertation chair

RECEIVED/APPROVED BY:

Renee Goldstein Osmic

Admissions Director

ACKNOWLEDGEMENT

First and foremost, I thank the divine presence that continues to guide me in ways I can't always explain. Through moments of doubt, fatigue, and confusion, there was always something — a quiet strength — that kept me going. I don't take that for granted.

This thesis would not have seen the light of day without the guidance and patience of my mentor, **Dr. Mirsada Cehic (Ph.D.)**. She believed in the work even when I wasn't so sure. Her timely nudges, honest feedback, and calm reassurance made all the difference. Thank you for standing beside me, not just academically, but with genuine support.

I also want to thank all my teachers — from my **MBA in Healthcare Management**, **Diploma in Business Analysis and Data Science**, and **MDS in Data Science**. Each course gave me something — clarity, discipline, or just the courage to keep learning. I still remember struggling with large datasets in the beginning; they were intimidating. But thanks to the patience of my mentors, I slowly found my way.

I'm also grateful to my fellow doctors and colleagues who helped me access and navigate the **LASI dataset**. Without your cooperation and shared knowledge, this project would've only been a dream.

And lastly, to my family — **my mother, father, and brother** — thank you for everything. For the late-night pep talks, the quiet understanding when I was absent-minded at dinner, and the constant reminders that I could do this. You never pushed, but you always stood by me. This is as much yours as it is mine.

With all my heart,

Dr. Ashwin J V

AGING GRACEFULLY: ECONOMIC PARTICIPATION AND THE SILVER ECONOMY IN INDIA — USING LASI DATA 2017–2018

Dr Ashwin J V

2025

Dissertation Chair: Dr Mirsada Cehic

Background: The older adult population is gradually on the rise. More emphasis is focused on economic participation and silver economy to improve the overall QoL. However, there is dearth of studies that capture the economic participation and factors that influence economic participation in older adults and ways how to improve the silver economy. This study aims to address this problem.

Methodology: This study is a secondary analysis of the wave 1 of the Longitudinal aging study of India. Descriptive analysis examined state-level variation in economic participation. Structured Equation model was used to assess associations between various clinical and socio-demographic characteristics and different levels of participation. In addition, machine learning classification and clustering models were also used to identify patterns and predictors related to economic participation.

Results: This study observed that nearly one-third of India's older adults remain economically active, largely within informal forms of work. Considerable variation was observed across states and regions. Economic participation was more common among men, individuals with lower educational attainment. Spatial analysis revealed clusters of higher economic engagement in predominantly developed states. The machine-learning models observed health, and financial security as one of the primaries contributes to economic participation. Cluster analysis highlighted that economic participation across India can be divided into three clusters and developed states has better silver economy prospects.

Conclusion: The study argues that economic participation in later life should not be interpreted uniformly as a sign of well-being. For many older adults, continued work reflects economic compulsion shaped by inadequate social protection and healthcare access. For others, it represents continuity, independence, and the maintenance of social roles. These findings underline the need for region-specific and health-sensitive policy responses.

TABLE OF CONTENTS

List of Tables	v
List of Figures	vi
CHAPTER I: INTRODUCTION	1
1.1 Introduction	1
1.2 Research Problem	1
1.3 Purpose of Research.....	1
1.4 Significance of the Study	2
1.5 Research Purpose and Questions	2
CHAPTER II: REVIEW OF LITERATURE	3
2.1 Theoretical Framework.....	3
2.2 Theory of Reasoned Action	5
2.3 Human Society Theory	9
2.4 Summary	11
CHAPTER III: METHODOLOGY	12
3.1 Overview of the Research Problem	12
3.2 Operationalization of Theoretical Constructs	12
3.3 Research Purpose and Questions	14
3.4 Research Design.....	15
3.5 Population and Sample	15
3.6 Participant Selection	15
3.7 Instrumentation	16
3.8 Data Collection Procedures.....	16
3.9 Data Analysis	16
3.9 Research Design Limitations	18
3.9 Conclusion	19
CHAPTER IV: RESULTS.....	20
4.1 Research Question One.....	20
4.2 Research Question Two	41
4.2 Summary of Findings.....	45
4.2 Conclusion	45
CHAPTER V: DISCUSSION.....	47
5.1 Discussion of Results	47
5.2 Discussion of Research Question One	52
5.2 Discussion of Research Question Two	54
CHAPTER VI: SUMMARY, IMPLICATIONS, AND RECOMMENDATIONS.....	57
6.1 Summary	57
6.2 Implications.....	57
6.3 Recommendations for Future Research	62

6.4 Conclusion	64
APPENDIX A PLAGARISM REPORT	65
APPENDIX B PYTHON NOTEBOOK CODES.....	68
REFERENCES	194

LIST OF TABLES

Table 1.1	Univariate analysis of the Socio-demographic data of the older adults: Age, Gender, religion and Caste	20
Table 1.2	Univariate analysis of the Socio-Demographic Profile of the older adults: Marital Status, Residence, and Living Arrangements	21
Table 1.3	Univariate analysis of the Socio-Demographic Profile of the older adults: Education, Occupation, working hours Socio-occupation status and Economic participation	22
Table 1.4	Univariate analysis of the Socio-Demographic Profile of the older adults: Economic, and Health Profile of the Study Population	23
Table 1.5	Univariate analysis of distribution of Healthy Aging and various Indicators	24
Table 2.1	Bivariate analysis of comparing the Socio-demographic data of the older adults: Age, Gender, Religion and Caste with Economic participation	25
Table 2.2	Bivariate analysis of comparison of socio-Demographic Profile of the older adults: Marital Status, Residence, and Living Arrangements with the Economic participation	26
Table 2.3	Bivariate analysis of education, occupation, working hours, social status, and economic participation	27
Table 2.4	Bivariate analysis of the Socio-Demographic Profile of the older adults: Economic, and Health Profile of the Study Population with Economic participation	28
Table 2.5	Bivariate analysis of distribution of Healthy Aging and various Indicators with Economic participation in older adults	29
Table 3	State-wise Distribution of older adults by Economic Participation Status	31
Table 4	Performance of different Machine learning models	37
Table 5	Top 10 factors influencing minimal/no economic participation among older adults (Findings from the XG Boost SHAPLEY model)	38
Table 6	Top 10 factors influencing moderate economic participation among older adults (Findings from the XG Boost SHAPLEY model)	39
Table 7	Top 10 factors influencing active economic participation among older adults (Findings from the XG Boost SHAPLEY model)	40
Table 8	Performance Comparison of Clustering Models for Grouping older adults Based on Participation Patterns	41
Table 9	DBSCAN-Derived Clusters of older adults in the LASI data	42

LIST OF FIGURES

Figure 1	Stacked bar plot of state-wise Distribution of older adults by Economic Participation Level	30
Figure 2	Average economic participation across various States and UT in the age group of Young-Old adults (60-74 years)	33
Figure 3	Average economic participation across various States and UT in the age group of Middle-Old adults (75 to 84 years)	34
Figure 4	Average economic participation across various States and UT in the age group of Old-Old adults (85 years and above)	35
Figure 5	State-wise Economic Participation Index and Spatial Association Patterns (LISA Labels)	36
Figure 6	ROC & AUC of the XG Boost model in supervised machine learning	37
Figure 7	Top 10 factors influencing minimal/no economic participation among older adults (Findings from the XG Boost- SHAPLEY model)	38
Figure 8	Top 10 factors influencing moderate economic participation among older adults (Findings from the XG Boost - SHAPLEY model)	39
Figure 9	Top 10 factors influencing active economic participation among older adults (Findings from the XG Boost - SHAPLEY model)	40
Figure 10	Mediating and Moderating Effects on Economic Participation Among the older adults: A Path Analysis	41
Figure 11	Spider Chart of Key Indicators by Cluster in older adults Participation	43
Figure 12	Regional Clusters of older adults Based on Economic Engagement and Health Vulnerability (DBSCAN)	44

CHAPTER I:

INTRODUCTION

1.1 Introduction

India is undergoing a demographic shift, for the first time in history a significant and growing share of its population is now over the age of 60.⁽¹⁾ According to the Longitudinal Aging Study in India (LASI, 2017–18), nearly 10% of India’s population—about 104 million people—are aged 60 and above. By 2050, this number is expected to reach 319 million, accounting for roughly 20% of the total population. This shift brings important changes to how we think about aging, health, and economic activity. It also challenges long-held ideas that older adults are passive dependents rather than active participants in the economy.^(1,2)

The idea of older people continuing to work, earn, or contribute economically is often left out of policy discussions. Much of the existing research focuses on caregiving roles, health decline, or the social vulnerabilities of this group. While these are critical issues, there is relatively little attention given to what *enables* or *prevents* older adults from staying economically active.⁽³⁾ Health conditions like chronic illness, access to health insurance, social networks, and awareness of government entitlements all play a part. Yet these factors are not often studied together when discussing economic participation.^(3,4)

1.2 Research Problem

Older adults are not a uniform group. Some continue to work out of necessity, while others do so by choice or habit. The LASI data shows that nearly 35% of older adults in India are still engaged in some form of work—mainly in informal sectors like agriculture or petty trade.⁽⁵⁾ But their ability to work and earn is often shaped by whether they can manage their health, get proper care, and afford medicine or treatment when needed. Chronic illnesses, which affect more than 60% of older adults, can directly reduce their ability to earn a stable income. At the same time, economic participation may also help improve their health—by giving them a sense of purpose, access to resources, or greater bargaining power within the household. This two-way relationship between health and economic activity deserves more attention than it currently receives.^(6,7)

1.3 Purpose of Research

From a healthcare and management perspective, this opens up a range of questions and opportunities. Many older adults who are still working may not visit hospitals regularly or may only do so in emergencies. They may rely on local clinics or traditional healers, not necessarily

because they prefer them, but because of costs or convenience.⁽⁸⁾ If health systems could recognize the financial and functional diversity within this group, they could offer more suitable services. This is particularly relevant for entrepreneurs, and private healthcare providers also have an important role to play in addressing these needs.^(9,10) As India's "silver economy" slowly takes shape, the health needs of active older adults are likely to grow. This includes not just curative care, but also preventive services, rehabilitation, and mental health support. Yet, few private models cater specifically to this segment.

1.4 Significance of the Study

There is a gap between what older adults need and what is currently offered.⁽¹¹⁾ Understanding what keeps them economically engaged can help design healthcare infrastructure that is both responsive and sustainable. This study tries to address the gap by aiming to explore these dynamics by using LASI data to understand how social factors relate to older adults' economic participation.

1.5 Research Purpose and Questions

There's little analysis of how this economic participation affects their healthcare access, affordability, and long-term health management. For healthcare providers and entrepreneurs, this presents a missed opportunity: financially active older adults represent a growing and under-served market segment. Understanding what enables or limits their participation can inform the design of healthcare services and business models—such as clinics, wellness centers, or insurance-linked care units—tailored to an aging population.

CHAPTER II:

REVIEW OF LITERATURE

2.1 Theoretical Framework

The term “older adults” or “older adults” is usually taken to mean people aged 60 years and above, based on the definition used by the World Health Organization (WHO).⁽¹²⁾ Across the world, populations are getting older at a pace that has not been seen before. A major reason for this is that people are having fewer children, while at the same time living longer than earlier generations. According to the United Nations World Population Prospects (2022), the share of people aged 65 years and above is expected to increase from about 10% in 2022 to close to 20% by 2050.⁽¹³⁾ In actual numbers, this means the global older adults population may grow from around 771 million to more than 1.6 billion. This rise is especially noticeable in middle- and high-income countries, and India is no exception to this pattern.⁽¹⁴⁾

In India, the aging of the population is becoming more visible with each passing decade. Data from the 2011 Census showed that there were about 104 million people aged 60 years and above, which was roughly 8.6% of the population.⁽¹⁵⁾ More recent projections by the Ministry of Statistics and Programme Implementation (MoSPI) suggest that this number could increase to nearly 194 million by 2031 and may reach around 319 million by 2050. At that point, older adults would make up more than one-fifth of the country’s total population.⁽¹⁶⁾ Another important point is that women form a larger share of the older adults population, largely because they tend to live longer than men. At the same time, the old-age dependency ratio is expected to rise from about 15% in 2021 to over 30% by 2050, indicating that fewer working-age adults will be available to support a growing older adults population.⁽¹⁷⁾

Alongside these demographic changes, broader social shifts are also affecting how aging is experienced in India. Urbanisation has increased over the years, and many younger people move away from their hometowns for jobs or education. Because of this, older parents are often left behind.⁽¹⁸⁾ At the same time, the joint family system, which traditionally provided support to older adults, is not as common as it used to be. This has led to a situation where a number of older adults are living either alone or with very limited support. In such circumstances, even minor health problems or financial difficulties can become serious concerns. These changes are not happening in isolation and are beginning to place visible strain on healthcare services and social support systems.⁽¹⁹⁾

Because of this, it becomes important to understand what older adults are actually doing in terms of work and income. In India, many people do not completely stop working after a certain age. Some continue out of need, others by choice.⁽⁵⁾ The term “silver economy,” which refers to economic activities associated with older populations, is important because it highlights the growing economic role of aging societies. However, in practice, this participation is quite heterogeneous.⁽²⁰⁾ Some older adults remain active in small jobs or businesses, while others depend on pensions or family. So there is no single pattern. Looking at these differences more closely can help in understanding what kind of support or policy is really needed.⁽²⁰⁾

At the same time, aging is not only about money or work. It is also about how people stay involved in life, how their roles change, economic participation changes and what becomes important to them as they grow older. Different theories try to explain this, but in reality, things are not always so clear-cut.⁽²¹⁾

Disengagement theory: The disengagement theory, given by Elaine Cumming and William Henry in 1961, suggests that people slowly withdraw from social and work roles as they age. Society also, in a way, expects less from them. But if we look around, this does not always happen exactly like that. Many older people remain quite active.⁽²²⁾

Continuity theory: Continuity theory, proposed later by Robert Atchley, takes a different view. It says people try to continue what they have been doing earlier in life. So instead of stepping away completely, they adjust. They may stay engaged in smaller or familiar roles, which helps them maintain a sense of routine and identity.⁽²³⁾

Socio-emotional selectivity theory: This theory looks more at relationships. It suggests that with age, people become more selective about where they invest their time and energy. They may not want a large network, but rather a few close and meaningful connections. This can also affect how much they stay involved in wider social or economic activities.⁽²⁴⁾

Life course perspective theory: The life course perspective is more about looking back over time. It suggests that old age does not develop suddenly. It is shaped by earlier life—education, type of work, health, family situation—all of these matter. So differences seen in older age often have roots much earlier.⁽²⁵⁾

Selective optimization theory: Selective optimization with compensation theory sounds complex, but the idea is quite simple. As people age, they adjust. They focus on what they can

do, try to maintain it, and find ways to deal with what they cannot do as easily anymore. This might mean changing the kind of work they do or reducing how much they take on.⁽²⁶⁾

Resource theory: Resource theory is probably the most straightforward. It says that people who have more resources—like money, good health, or social support—are in a better position to stay active and independent. Those who do not have these may face more difficulties. This difference becomes quite visible in older age.⁽²⁷⁾

All these theories mostly look at aging from the point of view of the individual—how people adjust, stay active, or sometimes step back. But aging is not happening in isolation. As more people live longer, it starts to show up in the economy as well, in terms of demand, work, and services. This is where the idea of the silver economy comes in.⁽²⁸⁾

Silver Economy: The idea of the silver economy has come up more often in recent years, mainly because populations are getting older in many countries. In simple terms, it is used to describe the kinds of economic activities that are linked to older adults.⁽²⁰⁾ This usually includes people above 50, although the exact definition is not always fixed. It is important here is that aging is not being looked at only as a burden, but also as something that changes demand in the economy.⁽²⁹⁾

It does not belong to just one sector. In fact, it spreads across several areas, sometimes in ways that are not very obvious. Healthcare is one of the more visible ones, especially with long-term illnesses becoming more common with age. But even outside healthcare, there are changes. Financial planning becomes more important, housing needs shift, and even daily mobility starts to matter more than before.⁽³⁰⁾

Some areas are easier to notice than others. Services related to treatment and care are often discussed. But things like transport, social activities, or even basic independence do not always get the same attention, even though they are important in everyday life.⁽³¹⁾ Technology is also slowly becoming part of this space, though not everyone uses it in the same way. In some cases, it helps a lot, like in remote consultations, but in other cases access itself is still a problem.⁽³²⁾

2.2 Theory of Reasoned Action

Silver economy and economic participation: When discussing economic participation in older age, it is important to note that this does not always refer to formal employment. The reality is that it can be quite diverse. Some individuals might keep working, but this is frequently in reduced roles or in informal arrangements.⁽³³⁾ There are also those who may not

be earning money directly, yet they remain engaged in everyday tasks, such as assisting with household duties, supporting family members in their work, or handling minor responsibilities. These things are easy to overlook because they are not counted as income, but they still have value in everyday life. So, it is not very useful to look at participation only in terms of jobs or earnings.^(33,34)

This also links to how quality of life is usually understood in later years. It is not only about how much money someone has. Health, feeling secure, and having some form of connection with others seem to matter just as much.⁽³⁴⁾ Even a small amount of financial independence can reduce dependence on others, which many older people prefer. At the same time, being involved in something—no matter how small—often helps maintain a routine. That, in itself, can make a difference.⁽²⁰⁾

The idea of active aging is often mentioned here, although it is not always very clearly defined in practice. In a general sense, it just points to the idea of staying involved instead of completely withdrawing. This does not have to mean working in a strict sense. It could be social, community-based, or even within the family. Not everyone is able to stay active in the same way, so it really depends on health and personal situation. It is more about having the possibility to remain engaged, rather than a fixed expectation.⁽³⁵⁾

Economic participation is only one part of a larger situation. It connects with health, social life, and the kind of support a person has. When older adults are able to stay involved—even in small ways—it often reflects in how they manage their daily lives. At the same time, these small contributions also have a wider effect, which is why they are often discussed along with the idea of the silver economy.^(36,37)

So, while the silver economy is often described in a very positive way—as something that can support growth and inclusion—the actual situation is more uneven.⁽²⁰⁾ It depends on how systems are organised and who is able to access them. If we look at other countries, some have adjusted faster than others. Japan is often discussed in this context, mostly because it has been dealing with an aging population for a longer time. There has been more focus there on adapting services and also keeping older adults involved in different ways, including work. Other countries like Sweden or Germany have also taken steps, though their approaches are not exactly the same.⁽³⁵⁾ Spending patterns also give some idea of what is happening. In many developed countries, healthcare spending related to older adults is expected to go up over time. Alongside that, services related to care are also expanding. In some cases, these sectors are

growing faster than the general economy, which shows how significant this shift is becoming.^(33,34)

In India, the situation is still developing but moving in a similar direction. The number of older adults is already large, and it is increasing. Because of this, demand for different kinds of services is also rising. This includes healthcare, but also areas like housing, financial services, and basic support systems.⁽³⁸⁾ Some estimates suggest that this is already becoming a sizeable market, though it is not evenly developed across regions. At the same time, there are clear limitations. A large part of the elderly population still depends on family support, and access to formal services is not always consistent. Even when services exist, they may not reach everyone equally.⁽³⁹⁾ This gap can be addressed with the help of telemedicine, but it works better in some areas than others. So while the potential is often highlighted, the ground reality is still mixed.⁽⁴⁰⁾

There have been efforts at the policy level as well. Programs like the National Programme for Health Care of the Elderly, the National Social Assistance Programme, and the Atal Pension Yojana are meant to address some of these gaps. But their impact is not uniform, and there are still areas where support is limited.⁽⁴¹⁾

A number of schemes have been introduced over time for older adults in India. Pradhan Mantri Vaya Vandana Yojana (PMVVY) is meant to provide a steady pension for people aged 60 years and above, mainly to ensure some regular income.⁽⁴²⁾ Then there are programmes like the Integrated Programme for Older Persons (IPOP), which are not limited to just financial support. They also include services related to basic healthcare and day-to-day well-being, although how much support actually reaches people can differ from place to place.⁽⁴³⁾

There are also more specific efforts, such as the Rashtriya Vayoshri Yojana (RVY). This scheme focuses on older adults from economically weaker sections and provides assistive devices. In practical terms, this can help with mobility and routine activities, though again, access is not always the same everywhere.^(44,45) So while these government schemes exist, their actual impact depends quite a bit on awareness and local implementation.⁽⁴⁶⁾

Apart from government initiatives, non-governmental organizations are also involved in different ways. Many of them work more closely with communities, especially where formal services are limited. Their work can include health-related support, social activities, or simply helping older individuals stay connected. It is not uniform across all regions, but in some cases, they play an important role where gaps exist.⁽⁴⁷⁾

There have also been attempts to involve older adults in some form of economic activity. Skill-based programmes are one example, though their reach is still uneven. Some people are able to take part, while others may not, depending on their situation—health, education, or even location can make a difference.^(47,48)

At the same time, awareness efforts have been carried out by both government and non-government groups. These are mainly aimed at changing how aging is viewed, and also at highlighting that older adults can still contribute in different ways. However, changes in social attitudes usually happen slowly, so the effect is gradual rather than immediate. Overall, these different efforts show that there is some movement in this area, even if it is not consistent everywhere.⁽⁴⁶⁻⁴⁸⁾ Even with these steps, there are still clear challenges. In some places, infrastructure is limited, and access to healthcare is not always easy. Social attitudes can also play a role, especially where aging is still associated more with dependency than participation. Because of this, the potential often discussed in relation to the silver economy is not fully seen on the ground.⁽⁴⁹⁾

When it comes to data, the Longitudinal Aging Study in India (LASI) is often referred to. It started around 2017 and its first round was completed in 2018, mainly to fill the gap in detailed information on aging in India. The sample size was quite large, covering more than 72,000 individuals aged 18 years and above along with their spouses, which makes it fairly representative. The data was collected through structured interviews, though, as with most large surveys, some variation is always possible.^(2, 49)

The study looks at different areas rather than focusing on just one. It includes health, access to healthcare, financial conditions, and also social aspects like family support and participation. It also looks at income, work, and retirement. So in that sense, it gives a broader picture of aging in India, even if it may not capture every small difference across regions.^(9, 11)

One distinct aspect of LASI is that the data it provides can also be compared with studies that have been conducted in other countries. This makes it easier to place India's aging population in a broader context, rather than looking at it in isolation. At the same time, the dataset itself is quite detailed and is expected to improve with future rounds, so it is likely to remain relevant for ongoing research and policy discussions.⁽⁴⁹⁾

The information from LASI covers different areas, but it does not always need to be seen in separate categories. Health-related data—like disease patterns or risk factors—can give some idea about what kind of services may be needed in the future. It also indirectly points to gaps,

such as where access to care is limited or where newer options like telemedicine are not being used as much. In a similar way, details about living arrangements and family support help in understanding how older adults are managing on a daily basis.^(49, 50) In some cases, strong family networks are present, while in others, support may be limited. This kind of variation suggests that the need for services like assisted living or community-based support is not the same everywhere.⁽⁵¹⁾

Economic information in LASI—such as income, savings, or spending—adds another layer to this. It gives a rough idea of what older adults can afford, although this can differ quite a bit across groups. So while some may have the capacity to spend on services like travel or leisure, others may still be focused on basic needs. There is also some indication of how people approach work and retirement.⁽²⁾ Not everyone follows a fixed pattern, and some continue working longer, while others stop earlier. These differences can matter when thinking about financial planning or even opportunities for re-engagement in work, though this area is still not fully explored.⁽⁵²⁾

Taken together, LASI can be useful for identifying patterns, but it does not answer everything on its own. It can point towards possible gaps or emerging needs, which may be helpful for both policy and business decisions.⁽²⁾ At the same time, there are still areas where more detailed understanding is needed, especially when looking at older adults from a market or business perspective.⁽³⁶⁾

While LASI gives information on income and expenditure, it does not fully explain why some older adults remain economically active and others do not, which could be explained once wave-2 of LASI data is published.⁽⁰⁾ Factors like type of work, health conditions, or personal preferences may all play a role, but these are not always captured in a straightforward way. Because of this, there is still scope for more focused research that looks at these aspects in detail, particularly if the aim is to understand how to design services or products that older adults would actually use.⁽⁵³⁾

2.3 Human Society Theory

At the same time, certain aspects remain insufficiently explored, especially the way multiple factors—such as health status, occupational history, education, and social support—interact to influence economic opportunities in later life. Age alone does not explain much. Gender, financial background, and even where a person lives can make a difference.⁽⁵⁴⁾ For instance, older women—especially those from more disadvantaged groups—may face different

challenges, but at the same time they may also represent a group whose needs are not fully understood yet.⁽⁵⁾ In a similar way, older adults in rural areas may not follow the same spending or consumption patterns seen in urban settings. Because of this, a uniform approach may not always work, and differences across groups need more attention.^(54, 55)

Another point is that most of the available discussion tends to focus more on demand—what older adults need or prefer. There is comparatively less work looking at the supply side. In other words, how prepared businesses or service providers actually are to respond to these needs is not always clear.^(30,31) This includes practical issues, such as how services are delivered, what kind of constraints providers face, and whether existing systems are able to adapt.⁽⁵⁶⁾ In some areas, especially healthcare, these gaps become more visible. There is also growing interest in whether technology can help improve service delivery, although its use is still uneven.⁽⁵⁷⁾

There is also a lack of studies that follow changes over time. Most information gives a snapshot, but does not show how economic participation or well-being evolves as people age. This makes it harder to understand longer-term patterns. If such data were available, it could help in identifying shifts in behaviour, preferences, or needs, which may not be visible in cross-sectional data.^(2, 52)

Another limitation is that research is often done within separate fields. Economic, health, and social aspects are usually studied on their own, even though in reality they are closely connected. Bringing these perspectives together could provide a more complete understanding of aging, especially in the context of the silver economy. It may also help in identifying practical ways to respond to emerging needs, rather than looking at them in isolation.^(35,39)

Because of these gaps, there is still a need for more focused work in this area. The present study attempts to look at economic participation among older adults in India using LASI Wave 1 data. The aim is not only to describe patterns, but also to understand what might be influencing them, and where possible gaps exist. In doing so, it hopes to contribute to a better understanding of how older adults engage with economic activities, and what this might mean for their well-being as well as for the broader context of the silver economy.

The aging population in India has been increasing for some time now, and it is starting to show more clearly in different ways. It is not only about how many people are getting older, but also about how their day-to-day situation is changing. For many older adults, health issues, financial

concerns, and dependence on others tend to overlap. Because of this, looking at only one aspect does not really explain much.

Health seems to be an important part of this. If a person is not in good health, it makes it more difficult to keep working or to remain active in general. At the same time, even when services are available, not every individual can access these services in the same manner. Therefore, there is a connection between health and economic activity, but this connection is not always very clear or straightforward. There is also the issue of how this increasing population of older adults is being supported.

2.4 Summary

As the number of older adults rises, there will likely be a greater demand for various services, particularly those related to care and basic support. However, in many instances, it is not very obvious if the current systems are capable of adapting to these changes.

In this situation, data sources such as LASI are frequently utilized because they compile different types of information all in one location. They give some idea about health, finances, and social conditions, although they may not capture everything in detail.

So, this study is mainly trying to look at how older adults are involved in economic activities and how this connects with their health and overall situation. It is also an attempt to see where things are not very clear yet, and where more understanding might be needed.

CHAPTER III: METHODOLOGY

3.1 Overview of the Research Problem

The older adult population in India has been gradually increasing over time. Due, to this rise in number, many older adults face problem related to health, money, and dependence on family members for support. These problems often occur together, making their situation more complicated than it appears when viewed from only one perspective.

Economic participation has been cornerstone in overall well-being of older adults. May factors influence the economic participation. With the growing number of older adults in India, there is increasing need to understand various factors that influence economic participation.

3.2 Operationalization of Theoretical Constructs

To capture the multidimensional aspects of aging and well-being, composite indices were constructed across five different domains which include physical health, mental well-being, functional ability, economic security, and healthcare satisfaction. Each domain was operationalized using a set of binary or categorical indicators selected from the survey dataset. *Physical health* included self-reported chronic conditions such as hypertension, diabetes, cancer, cardiovascular disease, joint disorders, and stroke, along with other morbidities (oral problems, thyroid disorders, sleep problems, etc.). *Mental well-being* encompassed psychosocial variables such as depression, loneliness, memory impairment, participation in religious gatherings, and life satisfaction measures. *Functional ability* was measured through impairments in Activities of Daily Living (ADL), Instrumental Activities of Daily Living (IADL), frailty, and sensory deprivation. *Economic security* comprised financial and food security variables such as household expenditure (MPCE), insurance coverage, covered by government schemes, skipping meals, and treatment discontinuation for monetary reasons. Lastly, *healthcare satisfaction* was constructed using variables on treatment respect, cleanliness, waiting time, and satisfaction with hospitalisation.

For each domain, a simple additive index was computed. For physical, mental health, functional ability, 0 was coded for presence/affirmative and 1 otherwise, Healthcare satisfaction and economic security was coded 1 was presence/affirmative and 0 otherwise. In the case of categorical variables, values were harmonised into an ordinal scale ranging from 0 (lowest satisfaction/poorest condition) to the highest score depending on the number of categories. The domain index for an individual i was then computed as:

$$D_i = \frac{\sum_{j=1}^k x_{ij}}{k} \quad (1)$$

where D_i denotes the index score of individual i for a given domain, x_{ij} represents the coded response to indicator j , and k is the total number of indicators in that domain. This yields a normalized score between 0 and 1, where higher values indicate better outcomes (after reverse-coding adverse indicators).

To ensure comparability across domains and avoid bias due to unequal scaling, all variables were standardised using *z-scores* prior to aggregation:

$$z_{ij} = \frac{x_{ij} - \mu_j}{\sigma_j} \quad (2)$$

where μ_j and σ_j are the mean and standard deviation of indicator j . For robustness, I also calculated Cronbach's alpha to assess internal consistency of each domain index:

$$\alpha = \frac{k}{k-1} \left(1 - \frac{\sum_{j=1}^k \sigma_{x_j}^2}{\sigma_T^2} \right) \quad (3)$$

where $\sigma_{x_j}^2$ is the variance of indicator j , and σ_T^2 is the variance of the total score. Domains with $\alpha > 0.7$ were considered to have acceptable reliability.

Finally, to create an **overall *Healthy aging index (HAI)*** denoted by W_i , the five domain indices were averaged:

$$W_i = \frac{1}{m} \sum_{d=1}^m D_{id} \quad (4)$$

where $m=4$ the number of domains (Physical health index, Mental well-being index, functional ability index and Economic security index) and D_{id} represents the domain-specific score for individual i . This composite indicator reflects a multidimensional profile of aging and allows subsequent regression and inequality analyses.

To capture the multidimensional aspect of older adults' engagement in economic activity, I constructed an ***Economic Participation Score (EPSi)*** by combining information from four

survey-based indicators: current occupation status, average work hours, job-seeking behaviour, and overall economic security. Each of these dimensions reflects a unique facet of economic involvement and security, and together they provide a composite measure of participation.

The score was calculated as:

$$EPS_i = O_i + H_i + J_i + ESI_i \quad (5)$$

where

- O_i = indicator of occupation status (1 if currently employed, 0 Unemployed/otherwise),
- H_i = indicator of work hours (0 if Working time is less than 48 hours, 1 if Working time is 48 to 52 hours, and 2 if working hours is more than 52 hours),
- J_i = indicator of job-seeking (1 if currently looking for a job/Unemployed, 0 otherwise), and
- ESI_i = continuous Economic Security Index score derived separately (described

The indicators were simply summed, yielding a total score with a theoretical maximum of 5. The decision to retain the raw additive structure was intentional, as it better preserved the intuitive interpretation of the components: higher scores directly indicated greater levels of economic engagement and security.

3.3 Research Purpose and Questions

Aim: The aim of this study is to identify and analyse the correlation between the economic participation of older adults in India, examining the relationships between demographic and health-related factors such as age, sex, education level, income level, and health status, using LASI data from 2017-2018.

Objective: By identifying specific correlations between economic participation and selected defined demographic and health factors, using LASI data for the period 2017-2018 in India, this research aims to generate concrete, evidence-based findings that will contribute to more effective and improved decisions in healthcare management.

Based on the analysis of LASI data and insights gathered from various primary and secondary sources, formulate practical recommendations key to healthcare management strategies. The purpose of these recommendations is to have a direct impact on the creation of the mentioned strategies, with the aim of improving the status of the older adults in India

Null hypothesis (H0): There is no significant relationship between any of the factors (age, sex, education level, income level, and health status) and economic participation among the older adults in India.

Alternate Hypothesis (H1): Significant relationship exists between selected socio-demographic and health factors (age, sex, education level, income level, and health status) and economic participation among the older adults in India.

3.4 Research Design

The study design was adopted as a secondary research. The LASI (2017–2018), a nationally representative survey covering multiple aspects of aging, was used as the sole data source. Since no primary data were collected, the study relies solely on secondary analysis of the existing LASI dataset. The analytical sample was reduced to respondents aged 60 and over. Relevant variables, key to identifying correlations, were extracted and operationalized in accordance with the defined research objectives. Permission to access and use the LASI data was previously obtained through a formal request process from the international institute for population sciences (IIPS).

The LASI study was approved by the Indian Council of Medical Research (ICMR) Ethics Committee as well as collaborating institutional review boards in India and internationally. Since the present research is based on secondary data analysis of anonymised, de-identified survey data, no additional ethical clearance was required. However, all analyses were conducted in strict adherence to the terms of data use. Formal permission to access and use the LASI dataset was obtained, and the confidentiality and privacy of participants were fully maintained.

3.5 Population and Sample

The total number of observations analysed in this study was 31,902, representing older adults aged 60 years and above across India. This large sample ensured adequate statistical power for domain-specific and subgroup analyses, while also enabling generalisable insights at the national level.

3.6 Participant Selection

This study is based on secondary data obtained from the Longitudinal Ageing Study in India (LASI), Wave 1. LASI employed a multistage stratified area probability cluster sampling design to obtain a nationally representative sample of individuals aged 45 years and above

across India. For the present study, only respondents aged 60 years and above were included in the analysis.

3.7 Instrumentation

This study is a secondary data analysis; therefore, no primary instruments or scales were administered by the researcher. The study used variables already available in the Longitudinal Ageing Study in India (LASI) Wave 1 dataset.

3.8 Data Collection Procedures

The data used in this study was obtained from the Longitudinal Ageing Study in India (LASI) Wave 1 dataset. In LASI, data collection was carried out by face-to-face interviews using structured questionnaires administered by trained field investigators. The survey collected information on demographic, health, social, and economic aspects of older adults across India. The secondary dataset was accessed and used for the present analysis in accordance with the objectives of the study.

3.9 Data Analysis

The data analysis was carried out using *SPSS version 27.0.0 (personal license)* and *Python version 3.3.9*. The dataset used was *LASI_W1_Dataset_All_State&UT.SPSS*. The dataset was first checked for duplicate entries. No correction of outliers or treatment of missing values was performed, as the analysis was designed to work with the dataset in its original form.

Data-Preprocessing: The variables required for the study were screened manually, and unnecessary variables were removed. A filter was applied to restrict the dataset to respondents aged 60 years and above. *To prepare the data for machine learning and cluster analysis, continuous variables were changed into categorical variables so that homogeneity could be maintained.*

Univariate and Bivariate Analysis: Univariate analysis was carried out in SPSS to summarize the distribution of all study variables. Frequencies and percentages were computed, and the findings are presented in tabular form. Composite indices were created only after assessing internal consistency, and only those indices with a Cronbach's alpha value ≥ 0.8 were retained for further analysis. Bivariate analysis was performed with Economic Participation Level (No/Minimal participation, Moderate participation, and Active participation) as the dependent variable. It was compared against socio-demographic variables, health profile, economic

profile, and the various indices. The results are presented in tabular format and with stacked bar plots.

Further, the Economic Participation Score was compared across states and union territories for three age groups: 60–74 years, 75–84 years, and 85 years and above. The results were displayed in tables, and maps of India were generated using Geographic Information Systems (GIS). States and union territories with economic participation levels below the national average were highlighted with red borders.

Mediation and Moderation Analysis: Mediation and moderation analyses were performed to identify the influence of demographic and health-related variables on economic participation.

- **Mediation analysis** was used to assess whether the effect of an independent variable on economic participation was transmitted through a third variable (the mediator).
- **Moderation analysis** was used to determine whether the relationship between an independent variable and economic participation changed depending on the level of another variable (the moderator).

The **dependent variable (target variable)** in both cases was *Economic Participation Level*. Interaction terms were used in moderation models. The results were summarized in tables and visualized using flow diagrams. A significance level of $p < 0.05$ was considered statistically significant. Values > 0.05 were considered *non-significant (NS)*.

Machine Learning and Deep Learning Models: Machine learning (ML) models were implemented in Python 3.3.9. The dataset was split into training and test sets in an 80:20 ratio. The target variable was *Economic Participation Level*, and all other variables were used as independent variables. The following ML algorithms were applied: Random Forest, XGBoost, and Decision Tree. To maintain model consistency, all variables were converted into categorical form. Deep learning (DL) models were also applied using the same data. DL analysis did not need the preselection of one specific target variable which is different from ML models because the models were able to find patterns in several outcomes. The evaluation of the models' performance was done through Receiver Operator Curve, Area Under the Curve (AUC), as well as Recall, Accuracy, and the generalisability ratio which is a comparison of test performance to train performance.

Cluster Analysis: Unsupervised clustering techniques were applied in Python 3.3.9 to explore natural groupings in the dataset. The algorithms included DBSCAN, K-Means, and

Agglomerative Clustering. To determine the optimal number of clusters, the Elbow method was applied and visualized using scree plots. The model that had the best silhouette score was utilized for the cluster analysis. Clustering is a method that does not require supervision, so there was no dependent variable that was defined. Spatial analysis was done using Local Indicators of Spatial Association which is called LISA to look at geographic patterns of economic participation among older adults people in various states of India. The ***Economic Participation Index (EPI)*** was mapped throughout the states and LISA was used to identify spatial clusters. The LISA analysis sorted the states into four distinct categories.

The first category is ***High-High***, which includes states that have high levels of economic participation and are surrounded by other states that also have high levels of participation. The second category is ***Low-Low***, which consists of states with low levels of economic participation that are surrounded by other low-performing states. The third category is ***High-Low***, where states with high participation are surrounded by states that have low levels of participation. The fourth category is ***Low-High***, which includes states with low participation that are surrounded by states with high levels of participation.

The results of the analysis were displayed in tables and were also visually represented on a map of India using Geographic Information Systems, which is referred to as GIS. This representation helped to emphasize the regional clustering of states with high and low levels of participation, which in turn allowed for the identification of spatial inequalities in economic participation among the older adults population.

3.9 Research Design Limitations

The present study is a secondary data analysis from the Longitudinal Ageing Study in India (LASI) Wave 1. Therefore, the researcher had limited control over the available variables and methods used for assessment. Also, the causal relationship could not be established with cross sectional data. Missing variables and recall bias could have affected the quality of the data to some extent.

Despite these limitations, LASI provides a large, nationally representative dataset that provides valuable insights into economic participation and silver economy.

3.10 Conclusion

This chapter outlined the research methodology adopted for this study. This chapter described the research design, source of data, participant selection process, instrumentation, and data collection procedures. The chapter also discussed the limitations associated with the research design.

CHAPTER IV: RESULTS

4.1 Research Question One

Table 1.1: Univariate analysis of the Socio-demographic data of the older adults: Age, Gender, religion and Caste

Variable	Sub variable	Count (n)	% of Total
Age	<i>60 to 74 yrs</i>	25,022	78.4%
	<i>75 to 84 yrs</i>	5,412	17.00%
	<i>≥85 yrs</i>	1,468	4.60%
Gender	<i>Male</i>	16,036	50.24%
	<i>Female</i>	15,866	49.76%
Religion	<i>None/Atheist</i>	65	0.20%
	<i>Hindu</i>	23,292	72.96%
	<i>Muslim</i>	3,731	11.68%
	<i>Christian</i>	3,194	10.00%
	<i>Sikh</i>	979	3.07%
	<i>Buddhist</i>	347	1.09%
	<i>Jain</i>	73	0.23%
	<i>Jewish</i>	2	0.01%
	<i>Parsi</i>	3	0.01%
	<i>Others</i>	215	0.67%
Caste	<i>Scheduled Caste</i>	5,157	16.17%
	<i>Scheduled Tribe</i>	5,334	16.48%
	<i>OBC</i>	12,137	38.04%
	<i>General</i>	8,224	25.81%

As shown in table 1.1, a majority of participants were from the young old age group, which accounted for 78.4 percent of the total. There were fewer older adults in the old-old age category. The distribution of gender among the participants was nearly equal. A large part of the participants identified as Hindus, accounting for 72.96 percent of the sample, while the other religious groups were smaller when compared. In terms of caste, the largest group was Other Backward Classes, which made up 38.04 percent, and this was followed by the General category. Scheduled Castes and Scheduled Tribes were present in similar proportions.

Table 1.2: Univariate analysis of the Socio-Demographic Profile of the older adults: Marital Status, Residence, and Living Arrangements

Variable	Sub variable	Count (n)	% of Total
Current Marital Status	<i>Currently married</i>	20,212	63.35%
	<i>Widowed</i>	10,845	33.98%
	<i>Divorced</i>	91	0.29%
	<i>Separated</i>	170	0.53%
	<i>Deserted</i>	102	0.32%
	<i>Live-in relationship</i>	170	0.53%
	<i>Never married</i>	312	0.98%
Residence	<i>Rural</i>	21,085	66.10%
	<i>Urban</i>	10,817	33.90%
Co-habitation & Living Arrangements	<i>Living alone</i>	1,630	5.11%
	<i>Living with spouse</i>	6,269	19.65%
	<i>With spouse & children</i>	13,697	42.94%
	<i>With children and others</i>	8,541	26.77%
	<i>With others only</i>	1,765	5.53%

As shown in Table 1.2, the majority of participants were married (63.35%), while widowed older adults formed a substantial proportion, and other marital categories were minimally represented. Most of the older adults resided in rural areas (66.1%), with fewer participants from urban settings. In terms of living arrangements, most participants lived with their spouse and children (42.94%), followed by those living with children and other relatives. A smaller proportion of participants lived only with their spouse, with other relatives, or alone.

Table 1.3: Univariate analysis of the Socio-Demographic Profile of the older adults: Education, Occupation, working hours Socio-occupation status and Economic participation

Variable	Sub variable	Count (n)	% of Total
Education	<i>Cannot read or write</i>	17,191	53.89%
	<i>Less than Primary school</i>	3,835	12.02%
	<i>Primary completed</i>	3,813	11.95%
	<i>Secondary school</i>	2,388	7.49%
	<i>Middle completed</i>	2,252	7.06%
	<i>Higher Secondary</i>	959	3.01%
	<i>Graduate degree completed</i>	823	2.58%
	<i>PG graduate completed</i>	259	0.81%
	<i>Professional course</i>	236	0.74%
	<i>Diploma and certificate</i>	146	0.46%
Socio-economic Status	<i>Poorest</i>	6580	20.6%
	<i>Poorer</i>	6573	20.6%
	<i>Middle</i>	6502	20.6%
	<i>Richer</i>	6259	19.6%
	<i>Richest</i>	5988	18.8%
Economic Participation	<i>No/minimal participation</i>	11,031	34.58%
	<i>Active participation</i>	10,540	33.04%
	<i>Moderate participation</i>	10,331	32.38%
Occupation	<i>Skilled agri. & fishery workers</i>	4,508	14.13%
	<i>Others</i>	1,357	4.25%
	<i>Elementary occupations</i>	1,077	3.38%
	<i>Workers not classified anywhere</i>	1,077	3.38%
	<i>Service workers</i>	913	2.86%
	<i>Craft & related trade workers</i>	287	0.90%
	<i>Professionals</i>	166	0.52%
	<i>Plant & machine operators</i>	117	0.37%
	<i>Clerks</i>	111	0.35%
	<i>Technicians</i>	68	0.21%
	<i>Legislators</i>	48	0.15%
Working Hours	<i>Less than 48 hours</i>	4,853	15.21%
	<i>48 to 52 hours</i>	4,367	13.69%
	<i>More than 52 hours</i>	4,089	12.82%

As shown in table 1.3, nearly half of the older adults participants were unable to read or write, while fewer than 5% had attained graduation or higher education. Socio-economic status was evenly distributed across the poorest, poorer, middle, richer, and richest categories. About one-third of the participants reported no or minimal economic participation. Most participants were engaged in the unorganized sector, particularly in agriculture, fishery, elementary occupations, and informal trades. The majority worked less than 48 hours per week, followed by those working 48–52 hours or during peak periods.

Table 1.4: Univariate analysis of the Socio-Demographic Profile of the older adults: Economic, and Health Profile of the Study Population

Variable	Sub variable	Count (n)	% of Total
Health Insurance Coverage	<i>No</i>	6,590	20.66%
	<i>Yes</i>	25,006	78.40%
No. of Hospital (OPD) Visit in Past 1 Yr	<i>Less than 3 hospital visits</i>	13072	41%
	<i>3-18 hospital visits</i>	8,215	25.8%
	<i>18-34 hospital visits</i>	454	1.4%
	<i>34 to 50 hospital visits</i>	16	1%
	<i>More than 50 hospital visits</i>	3	0%
Cost of Per OPD Visit	<i>Less than 1000INR</i>	6,055	19%
	<i>1000 to 10,000 INR</i>	9003	28.2%
	<i>More than 10,000 INR</i>	2425	7.6%
Sources from Reimbursement claimed	<i>No</i>	31,596	99.04%
	<i>Single Source</i>	298	0.93%
	<i>Multiple Sources</i>	8	0.03%
Covered with some kind of employment-based Insurance	<i>No</i>	225	0.71%
	<i>Yes</i>	22,589	70.81%
Benefits from Govt Scheme	<i>No</i>	11,993	37.60%
	<i>Single</i>	7,279	22.82%
	<i>Multiple</i>	12,630	39.58%
Awareness of Old Age Welfare Schemes	<i>Not Aware</i>	3,775	11.83%
	<i>Aware</i>	26,950	84.48%
Monetary Support Received for treatment	<i>No/minimal support</i>	29,465	92.37%
	<i>Received from single person</i>	1,558	4.88%
	<i>Received from multiple persons</i>	879	2.76%

As shown in Table 1.4, the majority of participants reported having health insurance coverage (78.40%), while a smaller proportion had no coverage. Most older adults had fewer outpatient visits, and the cost per visit was most commonly in the mid-range category. Reimbursement for healthcare expenses was almost absent, with the vast majority not receiving any. A considerable proportion were covered under employment-based insurance (70.81%). Most participants were aware of old-age welfare schemes, and a majority reported receiving benefits (62.40%). Monetary support for treatment was minimal for most participants (92.37%).

Table 1.5: Univariate analysis of distribution of Healthy Aging and various Indicators

Variable	Sub variable	Count (n)	% of Total
Physical health index (PHI)	<i>Good</i>	11	0.03%
	<i>Moderate</i>	1,478	4.63%
	<i>Poor</i>	30,341	95.14%
Mental Health index (MHI)	<i>Good</i>	1,001	3.14%
	<i>Average</i>	14,091	44.17%
	<i>Poor</i>	16,810	52.68%
Functional ability index (FAI)	<i>Good</i>	11,988	37.58%
	<i>Average</i>	12,216	38.30%
	<i>Poor</i>	7,615	23.87%
Economic security index (ESI)	<i>Good</i>	2,018	6.32%
	<i>Average</i>	10,744	33.67%
	<i>Poor</i>	19,140	59.99%
Healthy Aging Index (HAI)	<i>Good</i>	375	62.1%
	<i>Moderate</i>	11710	36.7%
	<i>Poor</i>	375	1.2%
Health care expense gap	<i>No gap</i>	1,473	4.62%
	<i>Average gap</i>	5,043	15.81%
	<i>Extreme gap</i>	331	1.04%
Healthcare satisfaction Index (HIS)	<i>Satisfied</i>	2,487	7.80%
	<i>Average</i>	415	1.30%
	<i>Not satisfied</i>	27,964	87.70%

As shown in Table 1.5, the majority of older adults had poor physical health (95.14%), and over half reported poor mental health (52.68%). In terms of functional ability, a larger proportion had good functionality (37.58%), while a smaller proportion had poor functionality. Most participants were found to have poor economic security (59.99%). With regard to healthy aging, the majority fell within the average category (62.7%), followed by those with high healthy aging, while fewer participants were classified as low. The healthcare expense gap was minimal for most participants. However, a large proportion of respondents were not satisfied with healthcare services (87.70%).

Table 2.1: Bivariate analysis of comparing the Socio-demographic data of the older adults: Age, Gender, Religion and Caste with Economic participation

Variables	Sub-variables	No/Minimal Participation n(%)	Moderate Participation n(%)	Active Participation n(%)
Age	<i>60 to 74 yrs</i>	8,607 (27.0%)	8,267 (25.9%)	8,148 (25.6%)
	<i>75 to 84 yrs</i>	1,878 (5.9%)	1,632 (5.1%)	1,902 (6.0%)
	<i>≥85 yrs</i>	546 (1.7%)	432 (1.4%)	490 (1.5%)
Gender	<i>Male</i>	3,509 (11.0%)	6,104 (19.1%)	6,423 (20.1%)
	<i>Female</i>	7,522 (23.6%)	4,227 (13.3%)	4,117 (12.9%)
Religion	<i>None/atheist</i>	21 (0.07%)	23 (0.07%)	21 (0.07%)
	<i>Hindu</i>	7,598 (23.8%)	7,869 (24.7%)	7,825 (24.5%)
	<i>Muslim</i>	1,617 (5.1%)	969 (3.0%)	1,145 (3.6%)
	<i>Christian</i>	1,091 (3.4%)	1,136 (3.6%)	967 (3.0%)
	<i>Sikh</i>	481 (1.5%)	149 (0.47%)	349 (1.1%)
	<i>Buddhist</i>	147 (0.46%)	81 (0.25%)	119 (0.37%)
	<i>Jain</i>	29 (0.09%)	16 (0.05%)	28 (0.09%)
	<i>Jewish</i>	0 (0.00%)	0 (0.00%)	2 (0.006%)
	<i>Parsi</i>	1 (0.003%)	1 (0.003%)	1 (0.003%)
	<i>Others</i>	46 (0.14%)	86 (0.27%)	83 (0.26%)
Caste	<i>Scheduled Caste</i>	1,576 (4.9%)	1,820 (5.7%)	1,761 (5.5%)
	<i>Scheduled Tribe</i>	1,806 (5.7%)	2,108 (6.6%)	1,420 (4.5%)
	<i>OBC</i>	3,911 (12.3%)	4,082 (12.8%)	4,144 (13.0%)
	<i>General</i>	3,293 (10.3%)	2,065 (6.5%)	2,866 (9.0%)

As shown in Table 2.1, economic participation declines with age. The young-old group (60–74 years) shows the highest involvement, with participation fairly evenly distributed across levels. Men have more active economic participation at 20.1%, while females demonstrate a higher rate of no or minimal participation at 23.6%. Hindus make up the largest share of participation at all levels, accounting for around 24% of the sample, while people from other religious groups are represented in smaller proportions. When looking at caste groups, older adults from the Other Backward Classes show relatively higher participation across categories, contributing roughly 12% to 13%.

Table 2.2: Bivariate analysis of comparison of socio-Demographic Profile of the older adults: Marital Status, Residence, and Living Arrangements with the Economic participation

Variables	Sub-variables	No/Minimal Participation n(%)	Moderate Participation n(%)	Active Participation n(%)
Current Marital Status	<i>Currently married</i>	5,930 (18.6%)	7,057 (22.1%)	7,225 (22.7%)
	<i>Widowed</i>	4,802 (15.1%)	3,002 (9.4%)	3,041 (9.5%)
	<i>Divorced</i>	33 (0.1%)	26 (0.1%)	32 (0.1%)
	<i>Separated</i>	47 (0.1%)	63 (0.2%)	60 (0.2%)
	<i>Deserted</i>	43 (0.1%)	32 (0.1%)	27 (0.1%)
	<i>Live in relationship</i>	64 (0.2%)	58 (0.2%)	48 (0.2%)
	<i>Never married</i>	112 (0.4%)	93 (0.3%)	107 (0.3%)
Residence	<i>Rural</i>	6,720 (21.1%)	7,570 (23.7%)	6,795 (21.3%)
	<i>Urban</i>	4,311 (13.5%)	2,761 (8.7%)	3,745 (11.7%)
Living Arrangements	<i>Living alone</i>	511 (1.6%)	559 (1.8%)	560 (1.8%)
	<i>Living with spouse</i>	1,752 (5.5%)	2,227 (7.0%)	2,290 (7.2%)
	<i>Living with spouse & children</i>	4,106 (12.9%)	4,763 (14.9%)	4,828 (15.1%)
	<i>Living with children and others</i>	3,902 (12.2%)	2,316 (7.3%)	2,323 (7.3%)
	<i>Living with others only</i>	760 (2.4%)	466 (1.5%)	539 (1.7%)

As shown in Table 2.2, married older adults show higher economic participation across all levels, while widowed older adults have relatively lower participation. Other marital groups form only a small part of the sample. Most participants are from rural areas, and they show greater involvement in economic activities compared to urban residents. Living arrangements also matter. Older adults living with their spouse and children are more actively engaged. Those living alone or only with others show much lower participation.

Table 2.3: Bivariate analysis of education, occupation, working hours, social status, and economic participation

Variables	Sub-variables	No/Minimal Participation n(%)	Moderate Participation n(%)	Active Participation n(%)
Education	<i>Cannot read or write</i>	6,643 (20.8%)	5,526 (17.3%)	5,022 (15.7%)
	<i>Less than Primary school</i>	1,284 (4.0%)	1,352 (4.2%)	1,199 (3.8%)
	<i>Primary completed</i>	1,313 (4.1%)	1,205 (3.8%)	1,295 (4.1%)
	<i>Secondary school</i>	670 (2.1%)	693 (2.2%)	889 (2.8%)
	<i>Middle completed</i>	650 (2.0%)	742 (2.3%)	996 (3.1%)
	<i>Higher Secondary</i>	226 (0.7%)	294 (0.9%)	439 (1.4%)
	<i>Graduate degree completed</i>	24 (0.08%)	49 (0.15%)	73 (0.23%)
	<i>PG graduate completed</i>	148 (0.46%)	301 (0.94%)	374 (1.2%)
	<i>Professional course</i>	45 (0.14%)	79 (0.25%)	135 (0.42%)
	<i>Diploma and certificate</i>	28 (0.09%)	90 (0.28%)	118 (0.37%)
Socio-economic Status	<i>Poorest</i>	2908 (9.26%)	2891 (9.21%)	781 (2.49%)
	<i>Poorer</i>	2066 (6.58%)	2073 (6.61%)	2434 (7.76%)
	<i>Middle</i>	2028 (6.46%)	1865 (5.94%)	2609 (8.31%)
	<i>Richer</i>	1984 (6.32%)	1817 (5.79%)	2458 (7.82%)
	<i>Richest</i>	2045 (6.51%)	1685 (5.37%)	2258 (7.19%)
Occupation	<i>Legislators</i>	1 (0.003%)	11 (0.03%)	36 (0.11%)
	<i>Professionals</i>	13 (0.04%)	70 (0.22%)	83 (0.26%)
	<i>Technicians</i>	5 (0.02%)	22 (0.07%)	41 (0.13%)
	<i>Clerks</i>	3 (0.01%)	45 (0.14%)	63 (0.20%)
	<i>Service workers</i>	63 (0.20%)	368 (1.2%)	482 (1.5%)
	<i>Skilled agricultural & fishery workers</i>	468 (1.5%)	2,145 (6.7%)	1,895 (5.9%)
	<i>Craft & related trade workers</i>	26 (0.08%)	99 (0.31%)	162 (0.51%)
	<i>Plant & machine operators</i>	11 (0.03%)	35 (0.11%)	71 (0.22%)
	<i>Elementary occupations</i>	116 (0.36%)	409 (1.3%)	552 (1.7%)
	<i>Workers not classified anywhere</i>	116 (0.36%)	409 (1.3%)	552 (1.7%)
	<i>Others</i>	90 (0.28%)	597 (1.9%)	670 (2.1%)
Working Hours	<i>Less than 48 hours</i>	321 (1.0%)	2,046 (6.4%)	2,486 (7.8%)
	<i>48 to 52 hours</i>	285 (0.9%)	1,871 (5.9%)	2,211 (6.9%)
	<i>More than 52 hours</i>	262 (0.8%)	1,724 (5.4%)	2,103 (6.6%)

As shown in Table 2.3, older adults with no or low education, were more likely to have minimal participation, while those with higher education showed greater active involvement. Economic participation was lowest among the poorest groups, with only 2.5% showing active participation, and increased among the middle groups, where active participation was highest (8.3%). In terms of occupation, most participants were engaged in agricultural work, which

formed the largest share across participation levels. Working hours also showed a clear pattern, with those working fewer than 48 hours reporting higher active participation (7.8%) compared to minimal participation.

Table 2.4: Bivariate analysis of the Socio-Demographic Profile of the older adults: Economic, and Health Profile of the Study Population with Economic participation

Variables	Sub-variables	No/Minimal Participation n(%)	Moderate Participation n(%)	Active Participation n(%)
Health Insurance Coverage	<i>No</i>	2,663 (8.3%)	3,594 (11.3%)	333 (1.0%)
	<i>Yes</i>	8,247 (25.8%)	6,727 (21.1%)	10,032 (31.5%)
No. of Hospital (OPD) Visit in Past 1 Yr	<i>Less than 3 visits</i>	4463 (13.9%)	4298 (13.47%)	4311 (13.51%)
	<i>3-18 visits</i>	2848 (8.93)	2619 (8.21%)	2748 (8.6%)
	<i>18 to 34 visits</i>	157 (0.49%)	129(0.40%)	168 (0.53%)
	<i>34 to 50 visits</i>	0 (0%)	6 (0.02%)	10 (0.03%)
	<i>>50 visits</i>	1 (0%)	1 (0%)	1 (0%)
Cost of Per OPD Visit	<i>Less than 1000INR</i>	1,947 (6.1%)	2,211 (6.9%)	1,897 (5.9%)
	<i>1000 to 10,000 INR</i>	1,966 (6.2%)	1,962 (6.2%)	1,770 (5.5%)
	<i>More than 10,000 INR</i>	1,963 (6.2%)	1,661 (5.2%)	2,106 (6.6%)
Reimbursement Status	<i>No</i>	10,909(34.2%)	10,237(32.1%)	10,450 (32.8%)
	<i>Single Source</i>	118 (0.4%)	93 (0.3%)	87 (0.3%)
	<i>Multiple Source</i>	4 (0.01%)	1 (0.003%)	3 (0.01%)
Covered with some kind of employment-based Insurance	<i>No</i>	22 (0.07%)	112 (0.4%)	91 (0.3%)
	<i>Yes</i>	1,929 (6.0%)	10,441 (32.7%)	22,589 (70.8%)
Benefits from Govt Scheme	<i>No</i>	4,367 (13.7%)	3,408 (10.7%)	4,218 (13.2%)
	<i>Single</i>	2,371 (7.4%)	2,580 (8.1%)	2,328 (7.3%)
	<i>Multiple</i>	4,293 (13.5%)	4,343 (13.6%)	3,994 (12.5%)
Awareness of Old Age Welfare Schemes	<i>Not Aware</i>	1,109 (3.5%)	1,297 (4.1%)	1,369 (4.3%)
	<i>Aware</i>	9,488 (29.7%)	8,819 (27.6%)	8,643 (27.1%)
Monetary Support Received for Treatment	<i>No/Minimal</i>	10,277(32.2%)	9,530 (29.9%)	9,658 (30.3%)
	<i>From Single Person</i>	495 (1.6%)	530 (1.7%)	533 (1.7%)
	<i>From Multiple Persons</i>	259 (0.8%)	271 (0.9%)	349 (1.1%)

As shown in Table 2.4, health insurance is strongly linked with economic participation, with a much higher share of active participants having coverage (31.5%) compared to very few among the uninsured. Employment-based insurance is also more common among those actively engaged (70.8%), suggesting its role in supporting continued work. Awareness of welfare schemes appears important, as active participation is higher among those who are aware

(27.1%) compared to those who are not (4.3%). Most participants had fewer healthcare visits, and very frequent visits were rare. Higher healthcare spending was more common among those who were actively participating. Across all groups, most older adults did not receive reimbursement and relied on little or no financial support for treatment.

Table 2.5: Bivariate analysis of distribution of Healthy Aging and various Indicators with Economic participation in older adults

Variables	Sub-variables	No/Minimal Participation n(%)	Moderate Participation n(%)	Active Participation n(%)
Physical health index (PHI)	<i>Good</i>	3 (0.0%)	2 (0.0%)	6 (0.0%)
	<i>Moderate</i>	623 (2.0%)	358 (1.1%)	497 (1.6%)
	<i>Poor</i>	10,363 (32.5%)	9,971(31.3%)	10,007 (31.4%)
Mental Health index (MHI)	<i>Good</i>	307 (1.0%)	326 (1.0%)	368 (1.2%)
	<i>Average</i>	4,769 (15.0%)	4,681(14.7%)	4,641 (14.6%)
	<i>Poor</i>	5,955 (18.7%)	5,324(16.7%)	5,531 (17.3%)
Functional ability index (FAI)	<i>Good</i>	4,601 (14.4%)	3,619 (11.3%)	3,768 (11.8%)
	<i>Average</i>	4,009 (12.6%)	3,971 (12.4%)	4,236 (13.3%)
	<i>Poor</i>	2,373 (7.4%)	2,741 (8.6%)	2,501 (7.8%)
Economic security index (ESI)	<i>Good</i>	1,853 (5.8%)	77 (0.2%)	88 (0.3%)
	<i>Average</i>	2,862 (9.0%)	7,443 (23.3%)	439 (1.4%)
	<i>Poor</i>	6,316 (19.8%)	2,811 (8.8%)	10,013 (31.4%)
Healthy Aging Index (HAI)	<i>Good</i>	6,231 (19.5%)	5,583 (17.5%)	8,003 (25.1%)
	<i>Average</i>	4,621 (14.5%)	4,604 (14.4%)	2,485 (7.8%)
	<i>Poor</i>	179 (0.6%)	144 (0.5%)	52 (0.2%)
Health care expense gap	<i>No gap</i>	529 (1.7%)	636 (2.0%)	308 (1.0%)
	<i>Average gap</i>	1,681 (5.3%)	2,252 (7.1%)	1,110 (3.5%)
	<i>Extreme gap</i>	118 (0.4%)	114 (0.4%)	99 (0.3%)
Healthcare satisfaction Index (HIS)	<i>Good</i>	757 (2.4%)	827 (2.6%)	903 (2.8%)
	<i>Average</i>	140 (0.4%)	121 (0.4%)	154 (0.5%)
	<i>Poor</i>	9,740 (30.5%)	9,202 (28.8%)	9,022 (28.3%)

As shown in Table 2.5, most older adults had poor physical health across all levels of participation (around 31–32%), showing that economic activity continues despite health limitations. Poor mental health was also common across all groups, with similar levels of participation even among those affected. Economic insecurity was strongly linked with

participation, with a higher proportion of active participants having poor economic security (31.4%). Older adults with better overall healthy aging were more likely to be actively engaged (25.1%). Healthcare satisfaction was low across all groups, with most participants reporting poor satisfaction (around 28–30%).

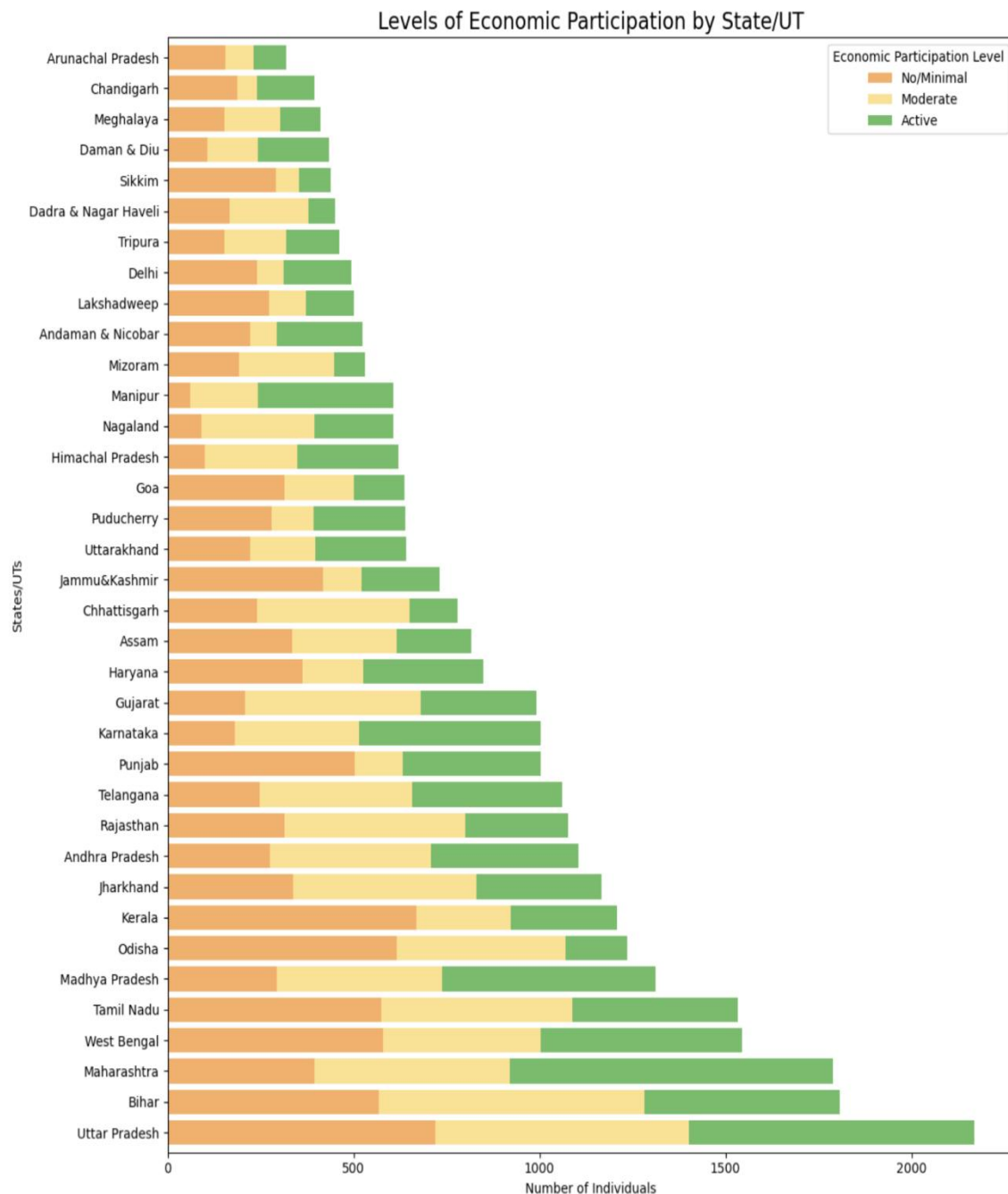


Figure 1: Stacked bar plot of state-wise Distribution of older adults by Economic Participation Level

Table 3: State-wise Distribution of older adults by Economic Participation Status

State/UT	No/Minimal Participation n (%)	Moderate Participation n (%)	Active Participation n (%)
Uttar Pradesh	720 (2.26%)	683 (2.14%)	766 (2.40%)
Bihar	569 (1.78%)	714 (2.24%)	525 (1.65%)
Maharashtra	394 (1.24%)	526 (1.65%)	870 (2.73%)
West Bengal	579 (1.81%)	424 (1.33%)	541 (1.70%)
Tamil Nadu	575 (1.80%)	513 (1.61%)	446 (1.40%)
Madhya Pradesh	293 (0.92%)	446 (1.40%)	574 (1.80%)
Odisha	616 (1.93%)	453 (1.42%)	168 (0.53%)
Kerala	669 (2.10%)	254 (0.80%)	286 (0.90%)
Jharkhand	337 (1.06%)	494 (1.55%)	337 (1.06%)
Andhra Pradesh	275 (0.86%)	434 (1.36%)	396 (1.24%)
Rajasthan	315 (0.99%)	485 (1.52%)	278 (0.87%)
Telangana	248 (0.78%)	410 (1.28%)	403 (1.26%)
Punjab	503 (1.58%)	129 (0.40%)	372 (1.17%)
Karnataka	180 (0.56%)	334 (1.05%)	490 (1.54%)
Gujarat	208 (0.65%)	473 (1.48%)	310 (0.97%)
Haryana	362 (1.13%)	164 (0.51%)	322 (1.01%)
Assam	336 (1.05%)	281 (0.88%)	199 (0.62%)
Chhattisgarh	240 (0.75%)	411 (1.29%)	129 (0.40%)
Jammu&Kashmir	418 (1.31%)	104 (0.33%)	209 (0.65%)
Uttarakhand	222 (0.70%)	176 (0.55%)	243 (0.76%)
Puducherry	279 (0.87%)	114 (0.36%)	247 (0.77%)
Goa	315 (0.99%)	185 (0.58%)	137 (0.43%)
Himachal Pradesh	101 (0.32%)	249 (0.78%)	271 (0.85%)
Nagaland	90 (0.28%)	306 (0.96%)	212 (0.66%)
Manipur	62 (0.19%)	181 (0.57%)	363 (1.14%)
Mizoram	193 (0.61%)	255 (0.80%)	83 (0.26%)
Andaman & Nicobar	222 (0.70%)	71 (0.22%)	230 (0.72%)
Lakshadweep	273 (0.86%)	99 (0.31%)	130 (0.41%)
Delhi	240 (0.75%)	71 (0.22%)	184 (0.58%)
Tripura	153 (0.48%)	165 (0.52%)	143 (0.45%)
Dadra & Nagar Haveli	168 (0.53%)	212 (0.66%)	71 (0.22%)
Sikkim	291 (0.91%)	63 (0.20%)	84 (0.26%)
Daman & Diu	108 (0.34%)	134 (0.42%)	192 (0.60%)
Meghalaya	154 (0.48%)	148 (0.46%)	110 (0.34%)
Chandigarh	188 (0.59%)	52 (0.16%)	154 (0.48%)
Arunachal Pradesh	155 (0.49%)	77 (0.24%)	86 (0.27%)

As shown in Table 3 and Figure 1, economic participation among older adults varies widely across states. Higher participation is seen in states like Uttar Pradesh, Bihar, and Maharashtra, while lower participation is observed in states such as Odisha, Chhattisgarh, and Punjab. Southern states, including Kerala and Tamil Nadu, show more older adults with minimal participation. In contrast, smaller states and union territories show relatively higher active participation.

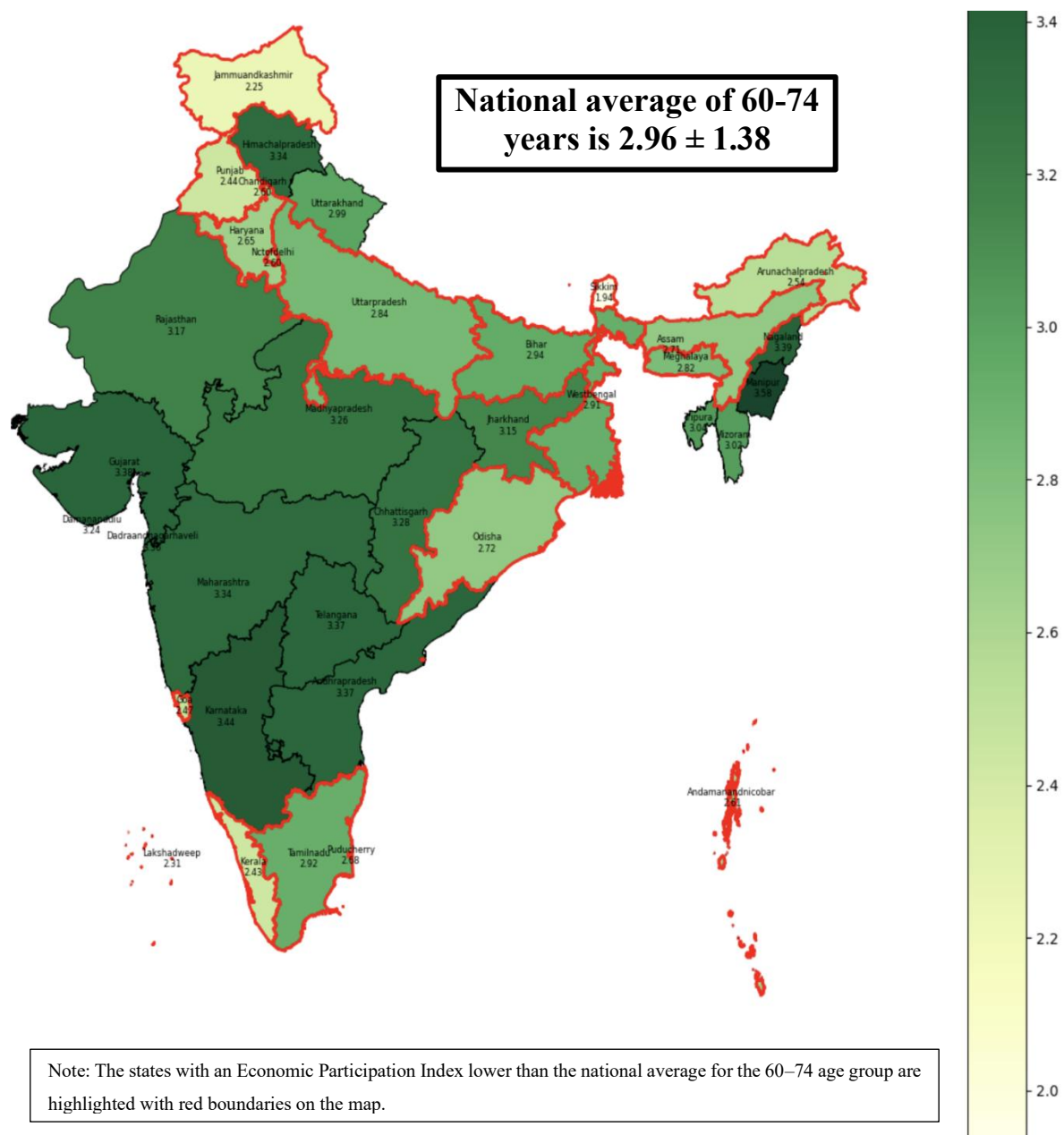
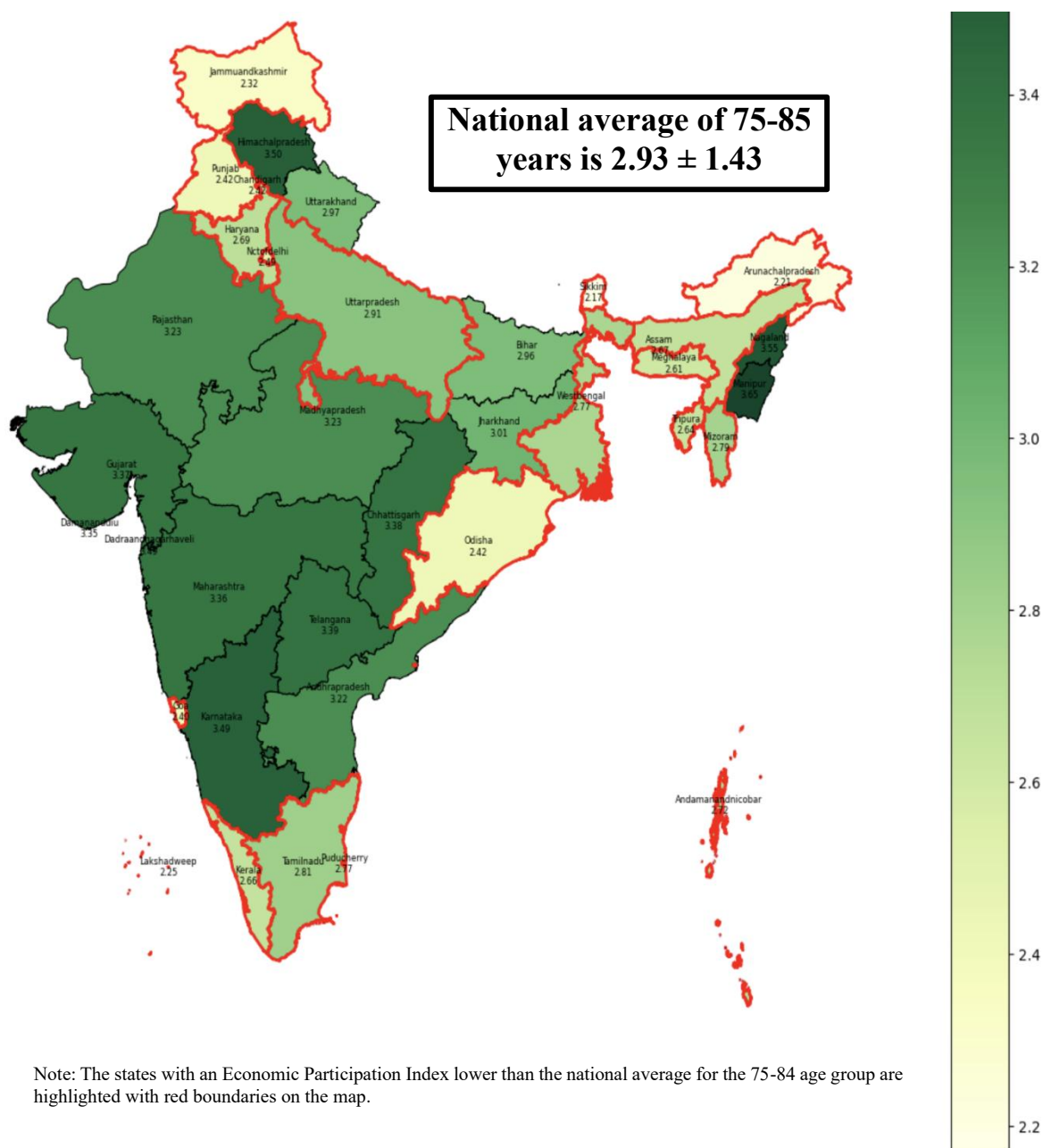


Figure 2: Average economic participation across various States and UT in the age group of Young-Old adults (60-74 years)

As shown in Figure 2, the national average economic participation score among the young-old is moderate (2.96). Higher participation scores are seen in states like Manipur, Karnataka, and Gujarat. In contrast, lower scores are observed in regions such as Sikkim, Jammu & Kashmir, and Kerala. Several states fall close to the national average, while others remain below it, showing uneven levels of engagement across the country.



Note: The states with an Economic Participation Index lower than the national average for the 75-84 age group are highlighted with red boundaries on the map.

Figure 3: Average economic participation across various States and UT in the age group of Middle-Old adults (75 to 84 years)

As shown in Figure 3, the national average economic participation score for older adults aged 75–84 years is 2.93. Higher participation is seen in states like Manipur, Nagaland, and Himachal Pradesh. In contrast, lower scores are observed in Kerala and Delhi, despite being more developed states. Economic participation was distributed unequally among different states and UTs as well.

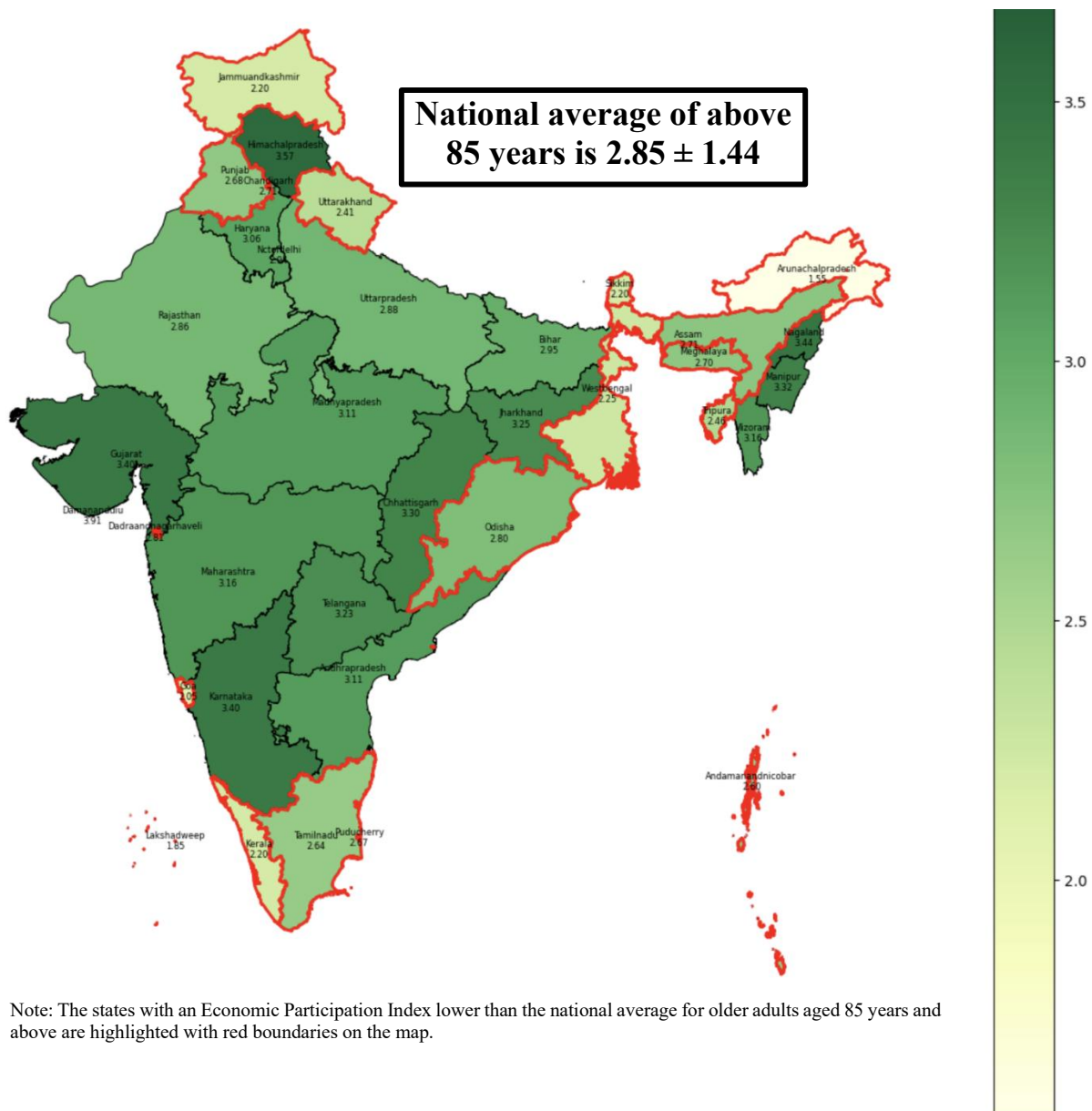


Figure 4: Average economic participation across various States and UT in the age group of Old-Old adults (85 years and above)

Figure 4 show that economic participation among older adults aged 85 years and above varies widely across states and union territories. Higher participation is seen in states like Himachal Pradesh, Gujarat, Karnataka, and some north-eastern regions such as Nagaland and Manipur. In contrast, lower participation is observed in states like Kerala, West Bengal, and Jammu & Kashmir, along with a few smaller regions. Economic participation in the oldest group appears to less influenced by geographic patterns.

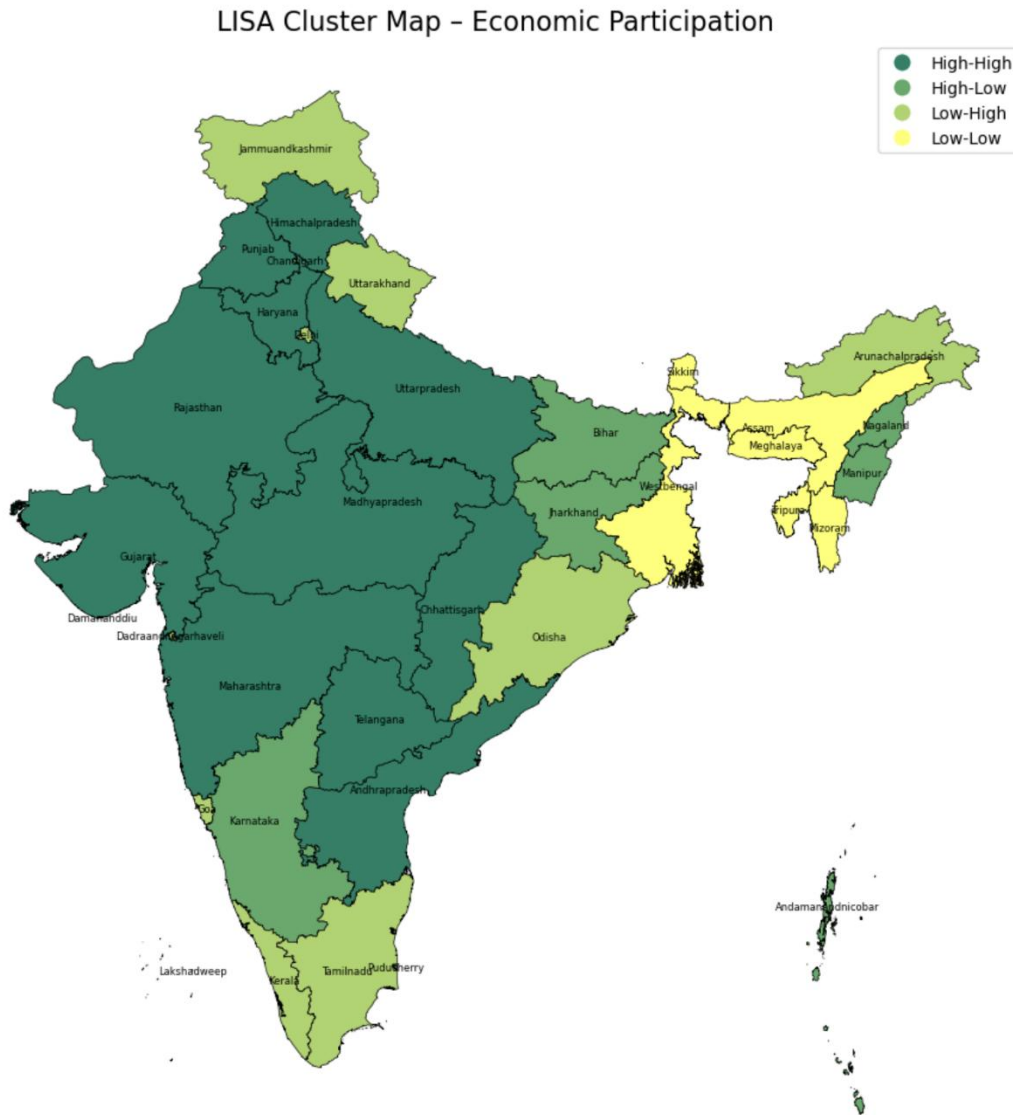


Figure 5: State-wise Economic Participation Index and Spatial Association Patterns (LISA Labels)

Many states such as Andhra Pradesh, Madhya Pradesh, Telangana, Gujarat, Maharashtra, Haryana, and Punjab fall in the *High–High* group. This shows strong regional clustering of high participation. States like Karnataka, Nagaland, and Jharkhand fall in the *High–Low* group. They perform well despite being surrounded by lower-performing states. States such as Kerala, Tamil Nadu, Odisha, and Jammu & Kashmir fall in the *Low–High* group. States such as Assam, Tripura, Sikkim, Lakshadweep, and Mizoram fall into the *Low–Low* cluster, indicating that both these states and their neighbouring regions experience similarly low levels of participation.

Table 4: Performance of different Machine learning models

Different Models	Overall Accuracy of the train model	Overall accuracy of Test model	Generalization ratio
Decision tree Classifier	0.79	0.90	1.1392
Random forest Classifier model	0.8599	0.7894	0.9178
XG Boost Classifier model	0.8603	0.8121	0.9439 (best model)
Deep neural network Classification model	0.82	0.78	0.9512

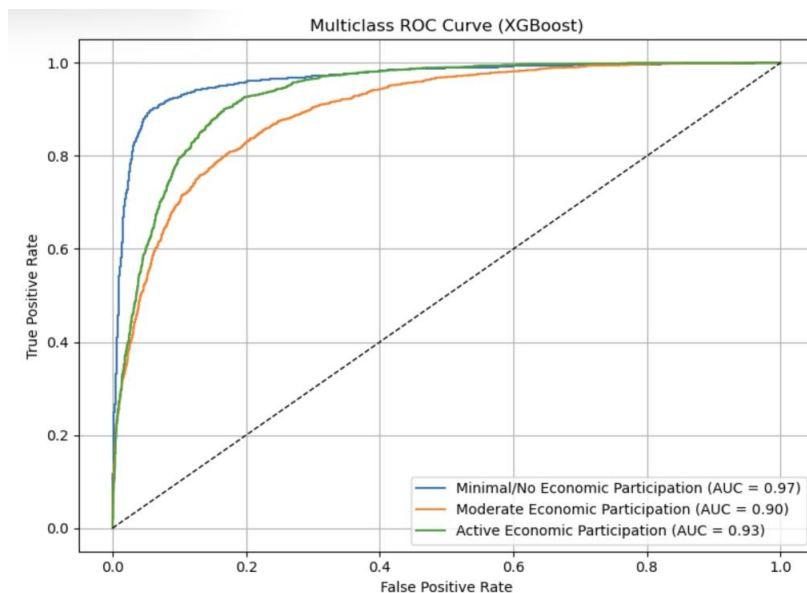
**Figure 6: ROC & AUC of the XG Boost model in supervised machine learning.**

Table 4 shows that the XGBoost model has the most consistent performance when looking at all the models that were considered. It achieves strong accuracy on both the training dataset and the test dataset, which suggests that it generalizes better than the Decision Tree, Random Forest, and Deep Neural Network models. This finding is also supported by Figure 6, which indicates that the XGBoost model performs well across all three categories. The AUC values are particularly high in situations with minimal or no economic participation. This indicates that the model can distinguish between different levels of economic engagement.

Table 5: Top 10 factors influencing minimal/no economic participation among older adults (Findings from the XG Boost- SHAPLEY model)

Feature	SHAP (%)	Mean SHAP value	Direction
Working hours	32.93%	-0.340590	Negative
Employment	13.93%	0.147193	Positive
Gender	13.46%	-0.059361	Negative
Health insurance Coverage	7.27%	-0.076484	Negative
Socio-economic status	5.43%	-0.044731	Negative
Healthy aging Index	4.92%	-0.083042	Negative
Healthcare expense Gap	4.00%	-0.021896	Negative
Education	2.35%	-0.009353	Negative
Religion	1.68%	-0.003042	Negative
Marital status	1.41%	0.003372	Positive

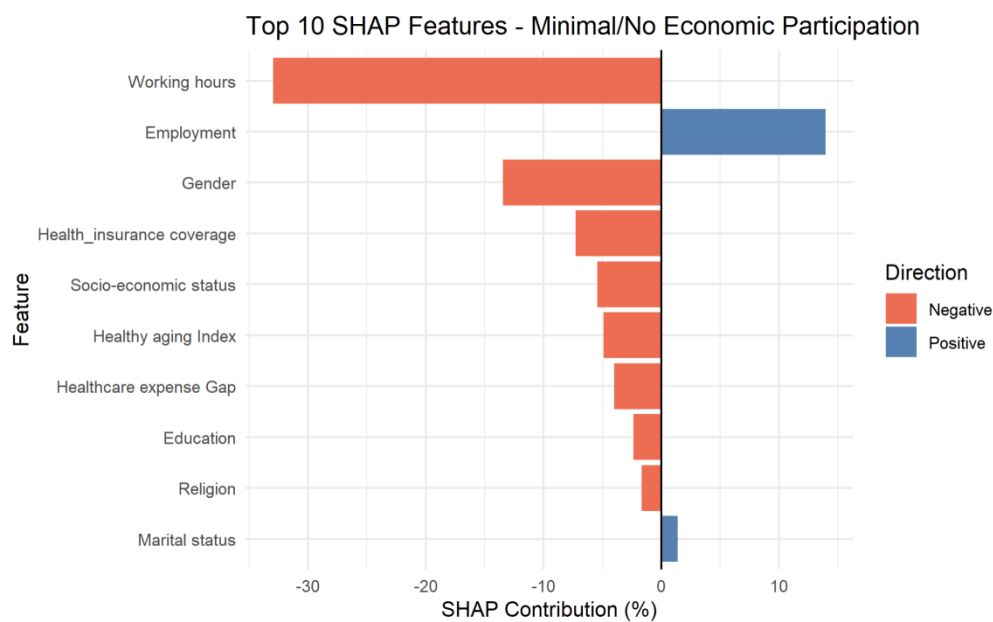


Figure 7: Top 10 factors influencing minimal/no economic participation among older adults (Findings from the XG Boost- SHAPLEY model)

Table 5 and Figure 7 show the main factors linked to minimal or no economic participation among older adults based on SHAP analysis. Working hours appear to have the strongest influence, followed by employment status and gender. Other factors, such as health insurance, socio-economic status, and healthy aging, tend to show a negative influence on minimal/no economic participation.

Table 6: Top 10 factors influencing moderate economic participation among older adults (Findings from the XG Boost - SHAPLEY model)

Feature	SHAP (%)	Mean SHAP Value	Direction
Working hours	16.34%	0.050501	Positive
Health insurance coverage	14.56%	-0.041688	Negative
Gender	11.84%	-0.104645	Negative
Socio-economic status	9.13%	-0.005538	Negative
Healthy aging index	7.66%	0.000781	Positive
Health expenditure gap	7.11%	-0.010358	Negative
Occupation	5.55%	-0.069291	Negative
Functional index	3.47%	-0.005892	Negative
Caste	3.08%	-0.008669	Negative
Domicile	2.96%	-0.004403	Negative

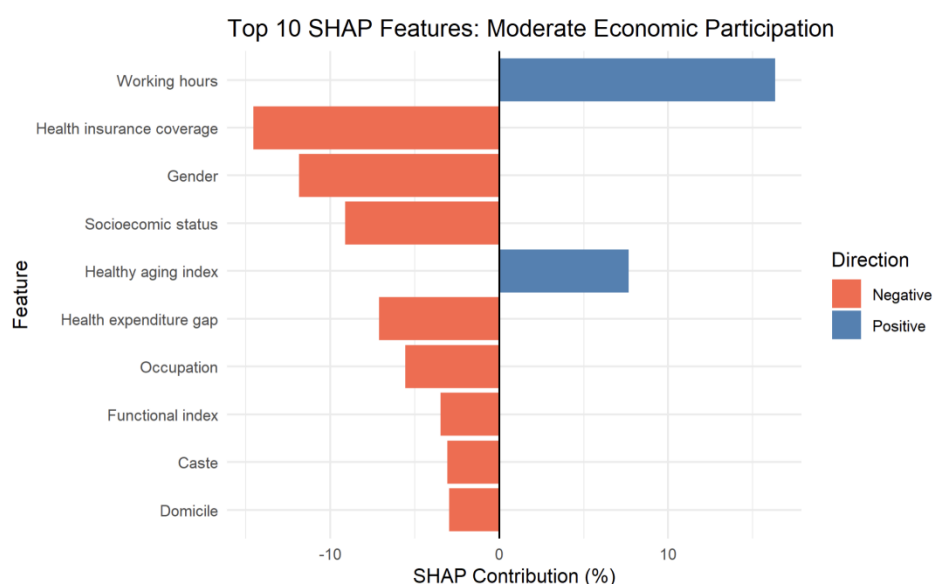


Figure 8: Top 10 factors influencing moderate economic participation among older adults (Findings from the XG Boost - SHAPLEY model)

Table 6 and Figure 8 present the main factors linked to moderate economic participation. Working hours again have the strongest positive influence, indicating that increased engagement in work is associated with higher levels of participation. On the other hand, health insurance, gender, and socio-economic status demonstrate negative effects in this category. This suggests that limited resources and existing disparities reduce participation. Functional ability and place of residence also show negative contributions.

Table 7: Top 10 factors influencing active economic participation among older adults (Findings from the XG Boost - SHAPLEY model)

Feature	SHAP(%)	Mean SHAP Value	Direction
Health insurance coverage	18.11%	-0.212225	Negative
Working hours	15.28%	-0.093121	Negative
Gender	10.87%	-0.092836	Negative
Healthy aging index	9.30%	-0.016279	Negative
Health care expenditure gap	8.56%	-0.040943	Negative
Socio-economic status	6.76%	-0.019848	Negative
Occupation	5.83%	-0.057776	Negative
Functional index	3.71%	-0.010369	Negative
Treatment satisfaction	3.36%	-0.004937	Negative
Education	2.79%	-0.008574	Negative

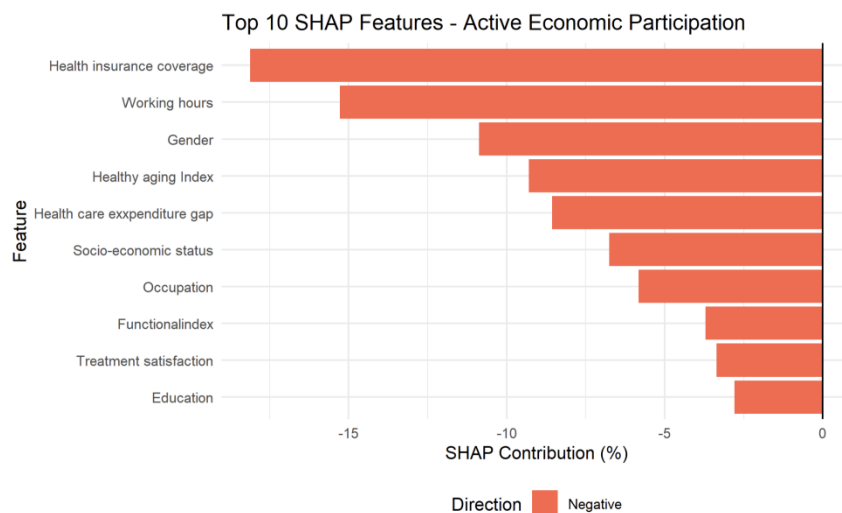
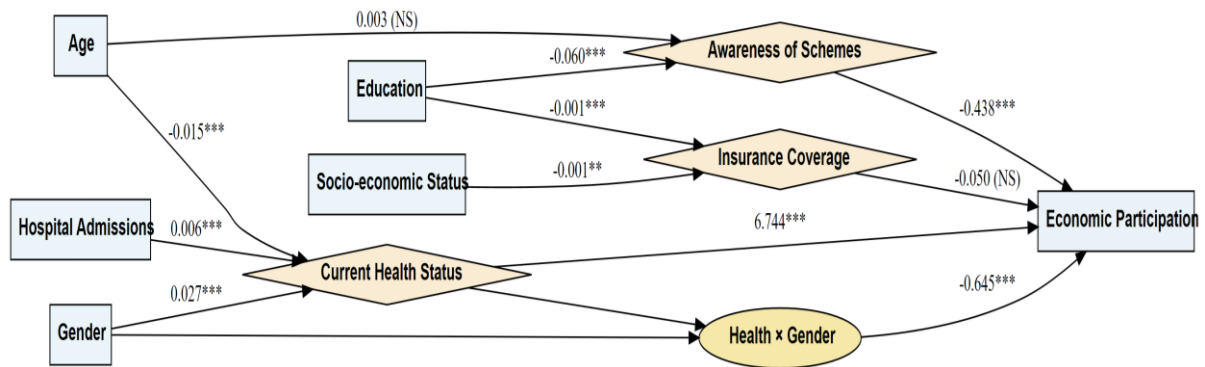


Figure 9: Top 10 factors influencing active economic participation among older adults (Findings from the XG Boost - SHAPLEY model)

Table 7 and Figure 9 show the main factors linked to active economic participation among older adults. Health insurance has the strongest influence, followed by working hours and gender, all showing negative effects. Other factors such as healthy aging, healthcare expenditure gap, socio-economic status, and occupation also show negative influence. Functional ability, treatment satisfaction, and education follow a similar pattern.



Note: In the above figure, boxes represent observed independent variables, diamonds show later constructs or mediators and ellipse indicate interaction/moderator terms. : A mediator is a variable that explains the pathway or mechanism through which an independent variable influences a dependent variable. A moderator, on the other hand, is a variable that affects the strength or direction of the relationship between an independent and a dependent variable. An "NS" indicates a non-significant effect, with its absence implying significance ($p > 0.05$). * p value < 0.005 , ** p value < 0.01 and *** p value < 0.001 .

Figure 10: Mediating and Moderating Effects on Economic Participation Among the older adults: A Path Analysis

Figure 10 shows how different factors influence economic participation among older adults. Current health status is shaped by age, gender, and hospital admissions. Economic participation is mainly driven by health. Better health is linked with higher participation. Awareness of schemes shows a negative effect. Insurance does not show a clear impact. The interaction between health and gender reduces participation and is significant. The current health status stands out as the key factor influencing economic participation.

4.2 Research Question Two

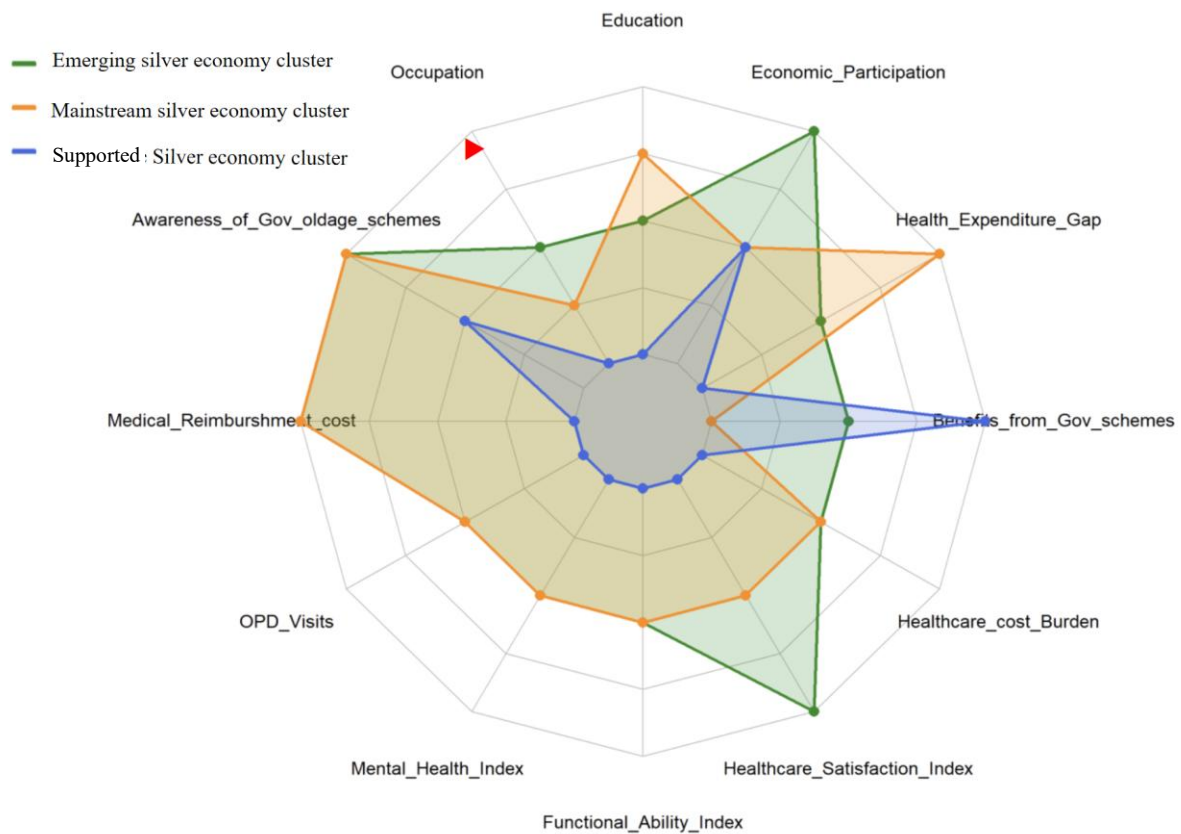
Table 8: Performance Comparison of Clustering Models for Grouping older adults Based on Participation Patterns

Model name	Ideal number of Clusters	Silhouette score	Performance of model
KNN cluster analysis	3	0.09	Poor clustering
Agglomerative cluster	2	0.55	Average clustering
DBScan	3	0.86	Good clustering

As shown in Table 8, DBSCAN performs best, with the highest silhouette score (0.86) and clear clustering. KNN performs poorly, while Agglomerative Clustering shows moderate results.

Table 9: DBSCAN-Derived Clusters of older adults in the LASI data

Variable	Emerging silver economy cluster	Mainstream Silver economy cluster	Supported Silver economy cluster
Age Group	Mostly Middle-old	Mostly Middle-old	Mostly Middle-old
Gender	Mostly Male	Mostly Males	Mostly Males
Religion	Mostly Hindu	Mostly Hindu	Hindu
Caste	Mostly General	Mostly General	General
Education Level	Most received some kind of education	Most received some kind of education	Mostly Illiterate
Residence	Mostly Rural	Mostly Rural	Mostly Rural
Marital Status	Mostly married	Mostly married	Mostly married
Living Arrangements	Mostly Living with others	Mostly Living with others	Mostly Living with others
Socio-economic status	Mostly rich to middle socio-economic status	Mostly Middle socio-economic status	Mostly Low socio-economic status
Work Hours	Mostly moderate work	Mostly Moderate work	Mostly Moderate work
Health care utilization	Mostly Very high	Mostly Very high	Mostly Moderate
Occupation	Mostly Unorganized sector work	Mostly Unorganized sector work	Mostly Farmers & workers of fisheries
Awareness of Old age scheme	Mostly High	Mostly High	Mostly Moderate
Health Insurance Coverage	Mostly Covered	Mostly Covered	Mostly Covered
Insurance covering work related injuries	Mostly Not covered	Mostly Not covered	Mostly Not covered
Medical Reimbursement	Mostly Covered	Mostly Covered	Mostly Uncovered
Treatment Support Received	Mostly Moderate support	Mostly Moderate support	Mostly Moderate support
Healthcare admission	Mostly Frequent	Mostly Frequent	Mostly Frequent
Physical Health Status	Mostly Poor	Mostly Poor	Mostly Poor
Mental Wellbeing Status	Mostly Moderate	Mostly Moderate	Mostly Poor
Functional Ability Status	Mostly Moderate	Mostly Moderate	Mostly Poor
Healthcare Satisfaction	Mostly satisfied	Not satisfied	Mostly Not satisfied
OPD Visit	Mostly Less than 2 visits / year	Less than 2 visits / year	Mostly Less than 2 visits/year
Reimbursement Status	Most did not receive reimbursement	Most did not receive reimbursement	Most did not receive reimbursement
Govt Scheme Benefit Status	Mostly did not receive benefits	Mostly did not receive benefits	At least benefits from one govt benefits schemes
Health Expenditure Gap	Mostly Low gap	Mostly No gap	Mostly High gap
Healthy aging Index (HAI)	Mostly Unhealthy aging	Mostly Unhealthy aging	Mostly Unhealthy aging
Economic Participation	Mostly Active	Mostly Moderate	Mostly Moderate



Note: The spider plot visualizes only those features that show variation across clusters

Figure 11: Spider Chart of Key Indicators by Cluster in older adults Participation

As shown in Table 9 and Figure 11, there were three distinct groups within the silver economy. On the other hand, the Supported Silver Economy Cluster is the most vulnerable group, with poorer health and limited economic resources, relying more on government support. For this group, continued work is mainly due to necessity instead of choice. The Emerging Silver Economy Cluster consists of older adults who keep working despite having health problems. Their participation seems to be influenced by a mix of necessity, established work routines, and the societal importance attached to staying active. The Mainstream Silver Economy Cluster includes a middle group that is more stable and has moderate levels of involvement while also having more even socio-economic and health situations.

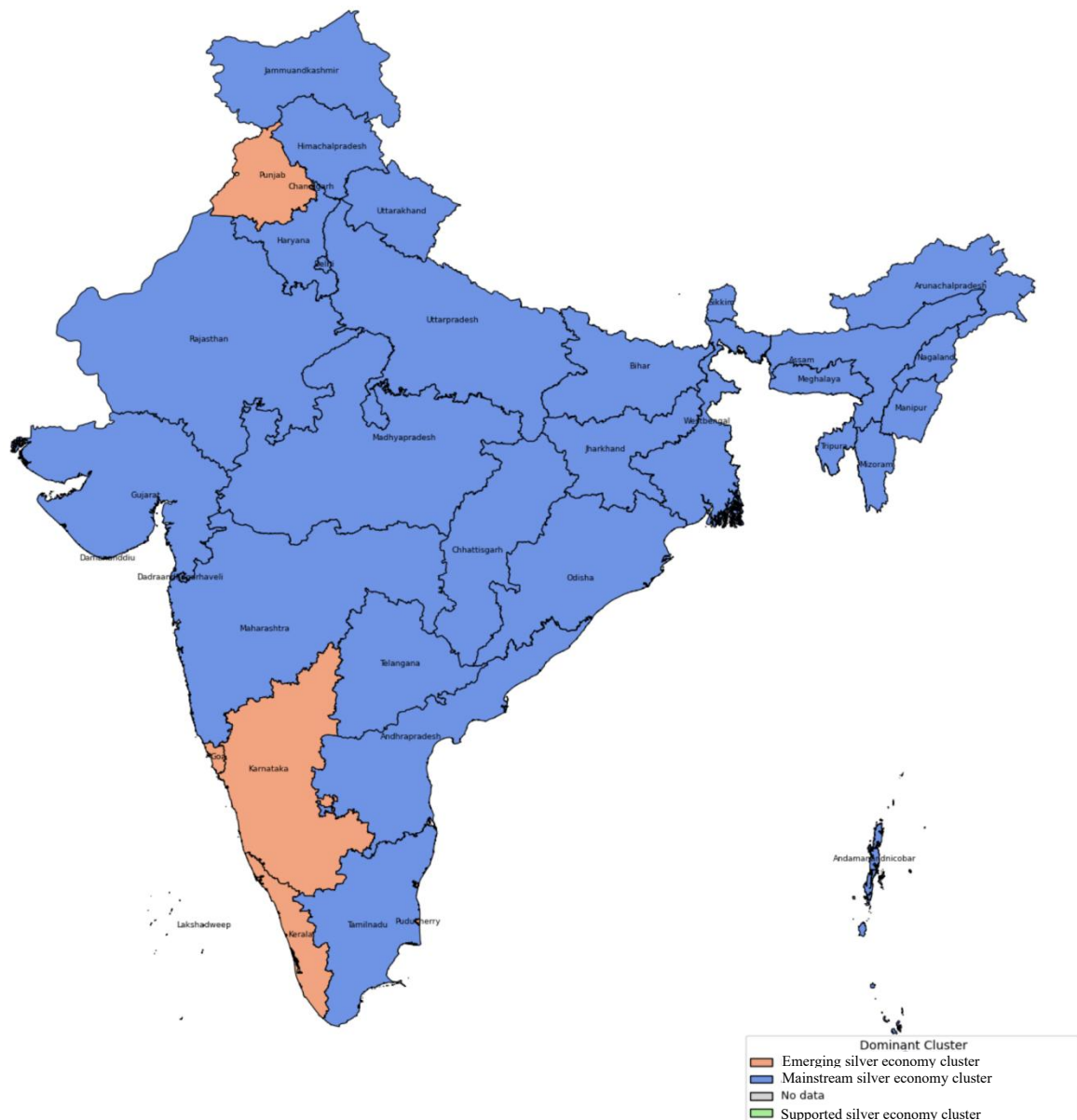


Figure 12: Regional Clusters of older adults Based on Economic Engagement and Health Vulnerability (DBSCAN)

As illustrated in Figure 12, two broad spatial patterns can be observed across states. Kerala, Karnataka, Goa, Punjab, and Puducherry are grouped within the “Emerging Silver Economy Cluster,” indicating relatively higher levels of continued engagement among older adults in these regions. Older adults in these regions remain active but have more health-related concerns. Most other states fall into the “mainstream silver economy cluster”. They show balanced economic participation with relatively better functional health. No state falls into the most vulnerable cluster.

4.2 Summary of Findings

- Older adults in India show a moderate level of economic participation. However, this tends to decline as age increases. Many continue to work even in later life, mostly because they need financial support rather than because they choose to. There is a noticeable gender gap. Men are more likely to remain active in the workforce, while women—especially widows—participate less.
- Economic need plays a major role. Older adults from poorer backgrounds are more likely to keep working. In contrast, those who are better off often withdraw from work. This suggests that continued participation is largely driven by necessity.
- A large share of older adults work in agriculture and informal sectors. They also face clear challenges in accessing skilled or age-appropriate jobs. Participation rates are much higher in rural areas compared to urban areas, where engagement is lower.
- Health problems are common in this group. Many older adults deal with serious physical and mental health issues but continue to work, showing a struggle between survival and well-being. Health is the most important factor influencing participation, as better health is linked to higher engagement levels.
- Insurance and welfare programs do not provide enough protection. While many are aware of these programs, they still face high out-of-pocket costs, and the benefits do not always ensure financial security.
- There are differences among states. States like Gujarat, Maharashtra, and Andhra Pradesh have higher participation rates, while Kerala and Punjab have lower rates, highlighting regional disparities.
- The clustering of data identifies three groups within the silver economy: Emerging, Mainstream, and Supported. Each group has different levels of participation and vulnerability. The Emerging group continues to engage despite health issues, the Supported group depends more on assistance, and the Mainstream group falls in the middle. Punjab, Karnataka, Goa, and Kerala mainly belong to the Emerging group.

4.2 Conclusion

In conclusion, these findings indicate that the silver economy in India has significant inequalities, where continued economic participation is more often due to need and

vulnerability rather than security and personal choice. Economic participation is different across various states in India. Emerging silver economy is most prominently restricted to developed and urbanized states of India.

CHAPTER V:

DISCUSSION

5.1 Discussion of Results

This secondary analysis of LASI Wave 1 shows that there is a decline related to age in economic participation but it is still significant in later life and is influenced by gender, socio-economic status, health, and regional context.

A closer look at these patterns indicates that socio-demographic factors affect economic participation in several ways. In this study, participation decreased as age increased. This finding is consistent with data from the National Sample Survey Office and the Periodic Labour Force Survey that report a continuous decline in labour force participation among older adults as they get older.^(58, 59) Similar patterns have been observed in studies conducted in India where participation tends to drop more sharply after older adults reach the age of 70 years and this is often associated with physiological changes and functional limitations. This trend is also supported by research from other countries.⁽⁵⁾ As people age, there is often a link to decreased physical ability, more chronic health issues, and a gradual withdrawal from economic roles.^(60, 61)

Men have reported higher levels of economic participation while women have reported lower levels or even none more frequently. This pattern is in line with findings from studies conducted in India by the International Institute for Population Sciences and the Institute for Social and Economic Change, which documented that older women have lower labour force participation compared to men.⁽⁴⁹⁾ These studies show that differences based on gender in employment continue into later stages of life. Similar findings have emerged in research focused on aging in India, where women's participation is limited because of their indian traditional lack of engagement in formal employment and their greater roles in unpaid household work and caregiving tasks.^(4,5)

A difference in economic participation was also noted between rural and urban regions, with older adults living in rural areas demonstrating higher levels of engagement compared to those living in urban areas. Similar patterns have been observed in Indian studies by the National Council of Applied Economic Research and the Institute of Economic Growth, which note higher labour force participation among the rural older adults.^(63, 64) This difference has also been observed in aging research in India across age groups. Evidence from various low- and middle-income regions indicates a comparable pattern, where older adults in rural locations

often continue to be economically active for an extended period.^(63, 64, 65) This rural–urban difference may be explained by several factors. In rural settings, older adults are more likely to be engaged in informal, self-employed, or family-based work, particularly in agriculture, where participation is flexible and does not require formal retirement. In contrast, urban employment is more likely to be formal and age-restricted, with fewer opportunities for continued work after retirement.^(4, 5)

The breakdown of participants by religion and caste in this research generally mirrors the overall demographic makeup in India, with Hindu and OBC groups constituting a more significant portion. Similar trends have been noted in studies in India regarding aging and participation in labor.⁽⁶³⁾ Additionally, current literature indicates that affiliations related to caste and community may impact access to job opportunities, especially in informal labour markets.^(63,66) At the same time, existing literature suggests that caste and community affiliations can influence access to employment opportunities, particularly within informal labour markets.⁽⁶⁶⁾

Existing studies suggest that caste and community backgrounds have an influence on access to employment and this is particularly evident in informal labour markets.⁽⁶⁷⁾ Evidence comes from institutions like the Institute of Economic Growth and the National Council of Applied Economic Research which indicates that Scheduled Castes and Scheduled Tribes are often more engaged in economic activity and this engagement is frequently in informal and low-paying work.^(62,67) Groups that are relatively better-off demonstrate lower levels of participation in the labour market and this is likely a result of their greater financial security.⁽⁶⁸⁾ This pattern can be understood when considering how social and economic disadvantages accumulate throughout a person's life. Individuals who come from SC/ST backgrounds often find themselves in informal jobs that are physically demanding and they tend to stay in these jobs for their entire lives without any real option for retirement so they keep working even as they get older. On the other hand, wealthier groups usually obtain formal employment that provides pensions or savings which enables them to retire at an earlier age.^(62,67,69) Various factors like age, health, location, and financial needs impact everyone and this somewhat blurs the distinctions related to caste in old age but the inequalities do not go away.⁽⁶⁷⁾

In this study, marital status and living arrangements were linked to differences in economic participation. Married older adults and those living with their spouse and children showed higher engagement, which is consistent with Indian studies highlighting the role of family

support in later-life work. Widowed older adults and those living alone showed lower participation. Similar patterns can be observed in the literature, and a connection exists between decreased household support and social isolation along with withdrawal from economic activities.^(3,68,69) Nevertheless, some studies originating from low-income environments indicate that older adults who are widowed may have higher participation levels due to financial necessity. This situation can occur when the death of a spouse leads to a loss of income, which requires the individual to keep working in order to meet basic survival needs.^(3,4) This may be explained by the availability of emotional, physical, and financial support within households, which facilitates continued participation in economic activities.⁽⁶⁹⁾

In this study, older adults with lower education had no/minimal economic participation. This is consistent with evidence linking human capital to employment in later life.⁽⁷⁰⁾ Indian studies report that those with little or no schooling are likely to be engaged in manual and low-paying work. Similar patterns were seen in studies from settings that are low and middle-income.^(71,72) The lower economic participation among less educated older adults in this study may be related to social and family factors. In many lower-income households, children start earning early and support the family. Because of this, older adults may stop working sooner. Some of the work they still do—like helping on farms or in small shops—is not always treated as a proper job. Often, it isn't reported. This may partly explain the lower participation seen in this study.^(71,72)

In this study, people from middle socio-economic groups seemed to participate more in economic activities. At the same time, those who are financially better off often reduce their involvement in work. This is in line with the observations from earlier studies, both from India and other countries.^(27,30) The higher participation seen in the middle socioeconomic group may not be due to a single factor. People in this group may still need to earn, but at the same time, they are more able to continue working compared to poorer groups.⁽⁷¹⁾ Those from poorer backgrounds are often involved in physically demanding work, which may not be sustainable at older ages. On the other hand, those who are better off may choose to reduce or stop working because of financial security. Differences in health, type of work, and access to support may also play a role in this pattern.^(8,27,30)

However, some studies, including those by the Institute of Economic Growth and the National Council of Applied Economic Research, have reported higher participation among poorer groups.^(71,72,73) There is some evidence that suggests older adults who come from poorer backgrounds continue to work in informal sectors and help in family based work. While those

in middle-income groups might reduce their working hours if they have some form of support like pensions or assistance from family members.^(71,72) The differences that are observed in this situation when compared to other studies could be explained by various reasons such as variations in the sample used, the methods for measuring socio-economic status, or the types of job opportunities that are available in different regions.⁽⁷³⁾

This study also highlighted that majority of older adults worked in agriculture and informal work. This is not particularly surprising because similar trends have been documented in other studies conducted in India, including those by the Institute of Economic Growth and the National Council of Applied Economic Research.^(71,72,73) Informal work often provides older adults with the chance to stay active, even though the working conditions may not be ideal. A comparable situation can be observed in many low- and middle-income countries where formal retirement systems are either inadequate or do not cover all older adults.^(72,73)

Another point worth noting is that a good number of older adults continue to work despite having poor physical or mental health. This has been observed in other studies as well, especially in countries like India. Health problems do not always mean that people stop working. In many cases, they simply cannot afford to.^(8,63) This is quite different from what is seen in high-income countries, where poor health is more closely linked to retirement.⁽⁷⁴⁾ There, stronger pension systems and social security make it possible for people to step away from work when needed. Differences in social protection, income security, & the availability of informal work likely explain why these patterns are not the same everywhere.^(35,74)

Functional ability seems to matter a great deal when it comes to whether older adults stay economically active. Those who are physically and mentally more capable tend to keep working for longer. This fits with broader aging research, including frameworks from the World Health Organization, which emphasise functional ability as the central to healthy aging.⁽³⁵⁾ Similar findings have been reported in Indian studies, where older adults who are more mobile and independent are more likely to remain engaged in work.^(75,76) From a biological point of view, this is not surprising. The ability to work depends on how well different body systems function together. Muscle strength, joint flexibility, and coordination all help with movement and reduce tiredness while working. Heart and lung function also matter, especially when it comes to sustaining effort over time.⁽⁷⁷⁾ Along with this, cognitive abilities like attention and decision-making help people stay focused and handle their work more effectively. When these

capacities are relatively well maintained, managing everyday activities, including work, tends to be easier.^(77,78)

Another important observation from this study is that, older adults who had moderate and active economic participation have been covered with health insurance, but many of these no/minimal economic participating older adults were not covered by health insurance. The relationship could be bi-directional. It is also possible that some older adults continue to work in order to keep their insurance benefits, particularly when those benefits are tied to their employment. This observation has been made in various low- and middle-income areas where insurance coverage often does not include all types of care, particularly for routine appointments or long-term conditions.^(79,80)

Many of the older adults live with chronic illnesses, leading to multiple regular hospital visits and spend around 1000-10,000INR per hospital visit, and covered by at-least one government schemes or work based insurance schemes irrespective of the economic participation. Similar observations were observed in several Indian studies.^(49,81) Despite an increase in public spending on health, reaching about 1.9% of GDP, the gap is yet to be fulfilled completely.⁽⁸²⁾ Schemes like Ayushman Bharat have helped, especially for hospital care, it provides only 5 lakhs per family/year for health care needs for hospitalised individuals.⁽⁸³⁾ But most everyday healthcare happens outside hospitals. OPD visits, medicines, and small but frequent expenses are often not covered. Because of this situation, working to prevent one from falling into debt for health care expenses may not always be a choice, but can sometimes be a necessity to handle ongoing medical costs.⁽⁸⁴⁾ However, in the no/minimal economic participating group, Barriers like processes for enrollment, the requirements for documentation, and the limited outreach efforts can create difficulties for older adults, when they try to access the available schemes.⁽⁸⁵⁾ Furthermore, some individuals may not see insurance as being useful, particularly if their immediate needs are more focused on outpatient care, claim disputes and medications, which do not always have sufficient coverage.⁽⁸⁶⁾ Over a period of time, this leads to lower levels of coverage among those who are already in more vulnerable situations and makes them overtly reliant on the government schemes which can be observed in this study.^(81,86)

One of the observations from our study was that healthcare satisfaction was low across all groups, and this did not differ by level of economic participation. This suggests that the dissatisfaction could be due to the gaps in the healthcare system rather than to individual factors. Similar observations have been reported from India and other low- and middle-income

countries, where people frequently describe long waiting times, limited options for care, inadequate infrastructure, and poor communication with doctors.^(87,88) Among older adults, having multiple chronic conditions may further worsen their experience of care provided by the health care facilities.⁽⁸⁹⁾ At the same time, a few studies report better satisfaction, especially among those with insurance or those using private facilities. This variation may be because satisfaction is a subjective measure that is influenced by expectations and past experiences, leading to “satisfaction paradox”—the phenomenon of reporting higher satisfaction despite providing suboptimal care.⁽⁹⁰⁾

5.2 Discussion of Research Question One

This study looked at how economic participation was different in various regions. Higher levels of engagement were noted among older adults in states like Gujarat, Maharashtra, Andhra Pradesh, and Haryana. On the other hand, lower levels of engagement were found in states such as Assam, Tripura, and Sikkim. The remaining states lay between these extremes. A closer look at spatial patterns indicated a clustering effect. The Local Indicators of Spatial Association suggested that neighbouring states often display similar levels of participation. This highlights a broader regional influence, rather than isolated patterns. Several other factors may explain this. Adjacent states commonly share labour markets, migration trends, immigrant workers, urbanization and socio-economic conditions. These shared features can shape opportunities for continued work in later life. Earlier work by the Institute of Economic Growth and the National Council of Applied Economic Research has reported similar observations.⁽⁹¹⁾ Regions with more immigrant workers, faster urbanization, a large informal sector, and limited social protection usually show higher participation rates. This is probably because people continue to work due to economic need. On the other hand, areas that have stronger welfare systems tend to report lower levels of engagement. Differences in demographic transition may also contribute.^(91,92) Where aging is relatively recent, older adults may remain in the workforce due to financial necessity. In more advanced aging settings, longer survival is often accompanied by greater morbidity, which can restrict participation.⁽⁹³⁾ At the same time, the evidence is not entirely consistent. Certain studies highlight factors that are at the individual level. These factors include health status, support from family, and financial need. These individual-level factors can have a direct effect on participation and in some instances, they might be more significant than the context of the region.^(92,94)

The results from the machine learning models highlights economic participation among older adults seems to change depending on the level of engagement. At the lower end, a mix of factors appears to help people move out of inactivity. These include working longer hours, being employed, and to some extent gender and health. Having insurance and a relatively better socio-economic position also seem to matter. At the same time, not everyone moves beyond this stage. Factors like marital status and the type of work people are in may keep them within this lower level rather than pushing them further. When we look at moderate participation, the picture shifts a bit. Working hours still matter, and better health seems to support continued involvement. But some barriers do not go away. Gender differences remain visible, and socio-economic position and place of residence still shape how much people can actually participate. So even if older adults are willing or able, the surrounding conditions limit what is possible. At higher levels of participation, the pattern becomes less clear. There are no strong positive drivers that stand out. Instead, several factors show negative links—such as education, health-related variables, insurance, and socio-economic status. This suggests that sustaining high levels of work in later life may not be easy. There may be limits coming from the type of work available or from broader structural conditions.

The SEM results point in a similar direction, though they highlight health more clearly. Current health status seems to sit at the centre, shaped by age, gender, and hospital visits. In general, better health is linked with higher participation. But access to support does not always increase work. In some cases, knowing about welfare schemes or having insurance may reduce the need to keep working. Gender also changes how these relationships play out. Even with similar health, not everyone participates in the same way, which suggests that deeper inequalities are still present.

Some of these findings from machine learning and SEM analysis were similar to what earlier Indian studies and studies from other countries.^(4,7,95,96) These studies also point to the importance of work-related factors and health, while showing that gender and socio-economic differences continue to matter. At the same time, there are some differences. In other settings, better health and resources sometimes support higher participation, but here they seem to limit it at higher levels.^(4,7,11,97)

One way to understand this is to look at the type of work available. Much of the work older adults can access is informal and physically demanding.^(10,11) It may help them stay active to some extent, but it does not always allow sustained or higher levels of engagement. Economic

need also plays a role.^(11,12) Doing some work may be necessary to avoid inactivity, but those with more financial security may not need to continue at higher levels. Health and functional ability help people stay involved, but they may not be enough on their own. Family support, social roles, and expectations also influence whether and how long people continue working.^(7,10,11,19) There are also studies that show a different pattern. In some contexts, education, better health, and insurance are linked with higher participation. This is often seen where job opportunities are more suitable for older adults and where active aging is more strongly supported.^(98,99) So the difference may come down to the broader setting—both in terms of economic development and the nature of the labour market.⁽⁸⁴⁾ It may also be related to the methods used, since machine learning approaches can pick up more complex relationships than traditional models.⁽¹⁰⁰⁾

5.2 Discussion of Research Question Two

The cluster analysis shows three broad groupings. One group includes older adults who continue working even with health issues—this is described as the emerging cluster. Another group shows more stable and moderate participation, referred to as the mainstream cluster. The third group includes those who are more vulnerable and depend more on support. At the state level, Kerala, Karnataka, Goa, Punjab, and Puducherry fall into the emerging category. Most other states fall into the mainstream group. No state appears in the supported category. There also seems to be a regional pattern, with southern states showing higher participation, but also more health-related concerns among older adults.

Findings from the Institute of Economic Growth and the National Council of Applied Economic Research also point in a similar direction, where older adults continue to stay engaged in work even when their health is not very strong.^(101,102) The “mainstream” cluster, in comparison, shows a more balanced situation. Here, participation is moderate, and people seem to manage with relatively stable functional ability. This kind of pattern has been noted in other low- and middle-income settings as well, where many older adults remain somewhat active without being fully dependent or completely disengaged.⁽¹⁰³⁾

Looking at the regional pattern, the grouping of southern states under the “emerging” category suggests a somewhat mixed situation. Higher levels of participation exist, but they are seen alongside greater health concerns. Earlier work, including that from the Institute of Economic Growth and the National Council of Applied Economic Research, has described something similar in southern India—people tend to live longer, but also deal with more chronic

conditions, and yet they do not fully withdraw from economic activity.^(11,102) At the same time, this does not look the same across all states. In Kerala and Punjab, migration seems to play a role. With family members working elsewhere and sending money back, some financial pressure may be reduced.⁽¹⁰³⁾ Even so, older adults may still stay involved in work, often in informal roles or activities that are socially embedded.⁽¹⁰⁴⁾ In Karnataka, the situation appears more mixed. There is a growing formal sector, but a large rural and informal base still exists. So while some people step back, others continue working.⁽¹⁰⁵⁾ In smaller regions like Goa and Puducherry, higher human development and service-oriented economies may ease the need to work for some. But opportunities linked to tourism or informal activities still allow others to remain engaged.⁽¹⁰⁶⁾

There are also studies that show a different trend, especially in more developed settings. In such contexts, stronger pension systems, wider social protection, and better overall health are often linked to lower participation in later life.^(4,11) Here, continuing to work may be more of a choice than a necessity. The differences across findings may come from the fact that multiple situations exist at the same time. Within the same region, some older adults are able to reduce or stop working because they have enough support, whether through pensions, savings, or remittances.⁽⁴¹⁾ Others, however, continue working because they need to, or because suitable opportunities are still available. Local labour markets, migration patterns, and access to support all seem to shape these outcomes in different ways.^(11,31,33)

This study has some strengths. It uses data from across India, so different states and groups can be compared. The sample size is also large, with more than 30,000 older adults. This helps in showing general patterns. Another point is that more than one method has been used. The study does not rely on a single approach. It looks at basic characteristics, regional patterns, clustering, and also uses machine learning. This gives a broader view of how different factors are linked.

There are also some limitations. The data is cross-sectional, so it only shows one point in time. It cannot explain changes or cause and effect. The age distribution is uneven, which may affect some results. Many variables, such as health and participation, are self-reported. This means there may be some reporting error. Machine learning shows patterns, but it does not explain them clearly. The study also does not go deep into the type of work older adults do. It is not always clear if they are working by choice or because they need to. Informal work is complex and not fully captured here. Some state-level differences may be due to things not included in

the data, like migration or local job conditions. Also, in large states, differences within regions may not be fully visible. Southern states, along with Uttar Pradesh, Bihar, and Punjab, were more strongly represented in the sample compared to other states.

Overall, participation in later life does not depend on one factor. It is shaped by many things together. Health is one part, but not the only one. Financial situation and available work also matter. Many older adults still have to spend their own money on health. Insurance does not always cover everything. Because of this, some people continue working even when they may not want to.

Looking at all the results together, participation seems to reflect both ability and need. Some people work because they are able to. Others work because they have no option. So, the silver economy cannot be understood only by looking at who is working. It also needs to consider inequality, gaps in support, and the conditions people are living in.

Chapter VI:

SUMMARY, IMPLICATIONS, AND RECOMMENDATIONS

6.1 Summary

The findings from this study show that different factors are closely linked when we look at older adults in India. Aging, health, work, and economic condition all come together in shaping participation. Many older people are still working even when their health is not very good. In many cases, this does not seem to be a matter of choice. It is more about managing daily expenses. Limited insurance support and high out-of-pocket costs make this situation harder, especially for poorer groups, women, widows, and those living in rural areas.

There also seems to be a gap between having insurance and actually being protected. Just because someone is covered does not mean their expenses are fully taken care of. People often still pay a lot from their own pocket. This creates pressure over time and may lead them to continue working even when they are not in a good position to do so.

Because of this, the recommendations focus on a few basic areas. One is reducing financial stress. Another is improving how healthcare works in practice, especially for long-term needs. There is also a need to think about older adults as a diverse group, since their situations are not the same. Policies need to reflect this.

6.2 Implications

1. Public Health and Policy Implications

- The results suggest that work in later life is often linked to financial need. Some older adults keep working despite poor health. This is not always voluntary. It is often because they still have expenses to meet, and support systems are not enough.
- This means income support becomes important. Pension systems need to be stronger and reach more people, especially those outside formal jobs. Support for women and widows also needs more attention. In some cases, linking financial help with health conditions may be useful, particularly for people with long-term illness. Without this, people may continue working even when it affects their health.
- Another issue is the gap between coverage and actual benefit. Even with insurance, people often do not get full reimbursement. Out-of-pocket spending remains high. This keeps financial pressure in place. Improving how claims are handled and expanding coverage for regular care could help reduce this burden.

- There are also differences between rural and urban areas. Work opportunities and support systems are not the same everywhere. Because of this, policies should not be the same for all regions. Local conditions, such as migration and the type of work available, need to be considered.

2. Healthcare System Implications

- The results point to a basic issue. What older adults need and what the system provides do not always match. Many are still working even when their health is not good. This suggests that care is either not reaching them on time or is not enough.
- Most of the care is still short-term. People go, get treated, and return. But many older adults are dealing with long-term conditions. They need regular care, not just one-time treatment. Things like follow-up, rehabilitation, and preventive care are still limited. Functional ability also matters here. If people can manage daily activities, they are more likely to keep going, both in life and in work.
- Insurance is another area where things do not fully work as expected. On paper, coverage exists. In reality, people still pay from their own pocket. The process is often slow or confusing. Because of this, some people delay care or avoid it. Making things simpler could help. Faster claims, better coordination, and easier access to cashless services may reduce some of this burden.
- In rural and smaller towns, the problem is different but related. Services are not always nearby. Travel, cost, and time become barriers. At the same time, many older adults in these areas continue to work. So, bringing care closer—through mobile units or local outreach—may make more sense than expecting people to travel.

3. Socio-Economic and Structural Implications

- The differences across groups are quite visible. Women, especially widows, tend to participate less. This is not just about choice. It often reflects fewer resources and less support. On the other side, poorer groups and rural populations show higher participation. But this does not mean better conditions. In many cases, it means they have no option but to keep working.
- So participation, by itself, does not explain much. It needs to be seen along with inequality. If not, there is a risk of misunderstanding the situation. Policies that only

aim to increase participation may miss the bigger issue. Support systems matter here—family, community, and local networks all play a role.

4. Service Delivery and Market Implications

- Older adults are not one single group. Their situations are quite different. Some are still working but have health problems. They need regular and affordable care. Others are somewhat stable and may benefit from preventive services. Then there are those who are more vulnerable and need stronger public support.
- Right now, services do not always reflect these differences. The approach is still quite general. There is room to make it more flexible. A mix of preventive care, treatment, and support services would work better.
- Insurance is still an issue across all this. Many people are technically covered, but using that coverage is not always easy. The process can be slow or unclear. Because of this, people still spend their own money. Simplifying these steps may help improve access and reduce pressure.

5. Implications for older adults-Focused Service Innovation

- The findings point to some clear gaps. Many older adults are still working even when their health is not good. This suggests that support systems are not fully meeting their needs. There seems to be a demand for services that help people manage both health and daily functioning.
- Care at present often feels fragmented. What may work better is a more combined approach. For example, managing long-term conditions along with physiotherapy, mobility support, and regular check-ups in one place. For those who have difficulty moving around, home-based care or community-linked services could make a difference. Things like teleconsultation or basic diagnostics at home may help reduce the need to travel.
- Cost is another issue that keeps coming up. Even when services exist, affordability becomes a problem. Models that reduce uncertainty—like bundled services or simple

payment plans—may help people continue care without interruption. Insurance-linked pathways could also be made easier to use.

- There is also a need to think about those who are still working. Fixed timings and rigid systems do not always suit them. Flexible consultation hours or targeted support for common issues like joint pain or fatigue may be more practical.

6. Proposed Integrated older adults Care Model

- The study shows that older adults are not all in the same situation. Some are working, some are partly active, and some are not working at all. Their health and financial conditions also differ. Because of this, a single model may not work for everyone. What is needed is something that can adjust to these differences.
- One group includes those who are still working but not in the best health. They need regular and affordable care. Managing chronic illness, getting help for pain or mobility, and having flexible access to care becomes important. The focus here is on helping them continue without their condition getting worse.
- Another group is more stable. They are somewhat active and managing well for now. In their case, prevention becomes important. Regular check-ups, basic fitness, and early care can help avoid future problems. Community-level support may also be useful here.
- Then there is a group that is more vulnerable. They may not be working and may have more serious health or financial issues. For them, care needs to be simple, low-cost, and easy to access. Home-based services, basic rehabilitation, and support through public systems are important. Reducing out-of-pocket costs is especially critical for this group.
- Across all these groups, a few things remain common. Costs should not force people to keep working when they are unwell. Care should be continuous, not one-time. And access should be easier, especially through local services, outreach, or digital options.
- This kind of approach is not about creating something completely new, but about organising services in a way that fits real-life situations better.

7. Elderly Re-employment and Productive Aging:

- Many older adults continue working even after retirement age. In many cases, this happens because of financial problems and increasing healthcare expenses. Therefore, proper re-employment support for older adults is important.
- India has introduced some programs for elderly re-employment. One example is the SACRED portal started by the Ministry of Social Justice and Empowerment. The program helps older adults get work opportunities according to their earlier job experience and skills.
- However, many people are still not aware of these services. Information about such programs can be shared through primary health centres, local self-government institutions, senior citizen associations, and community groups so that more older adults can benefit from them.
- Job opportunities for older adults should be planned according to their health condition and physical ability. Jobs with less physical strain may be better for many older adults. Some older adults may prefer lighter work such as part-time jobs, office work, teaching, helping others learn, or community-based activities instead of physically demanding work.
- Older adults may also face difficulty while using computers, smartphones, and other newer systems used at workplaces today. They may require basic support and training to use them comfortably. Simple training programs may help them learn these skills more easily. Short training sessions may help improve their confidence and increase job opportunities.
- Re-employment should not be considered only for earning money. For some older adults, working after retirement may help them stay in contact with people and remain involved in day-to-day activities. Some may also feel satisfied when they continue doing useful work.
- Government departments and private companies can also provide support for employment among older adults. Older adults with extensive experience may be involved in guidance, training, supervision, or similar supportive roles. Work settings may also be adjusted according to the needs of older workers.

- At the same time, re-employment should not replace pension and social security benefits. Older adults should be able to choose whether they want to work, instead of working only because of financial difficulty.

Overall, the findings suggest that work in later life is linked to money issues, health problems, and general living conditions. Many older adults are still working even when their health is not good. This does not always look like a choice. In many cases, it feels more like they have to keep working because they still need the income.

- Because of this, the focus may need to shift a little. Income support becomes important here. If basic needs are covered, the pressure to keep working may reduce. Healthcare is another part of the issue. It needs to be easier to access and not just one-time treatment. Many older adults deal with ongoing health problems, so regular care matters.
- It is also important to remember that not everyone is in the same situation. Some people may have support, others may not. So, one single solution may not work for all. Taken together, this points to a more practical way of looking at things. Real-life conditions matter. Reducing financial pressure matters. And supporting people to manage their health as they grow older also matters.

6.3 Recommendations for Future Research

Future studies can look at how economic participation changes over a longer period of time. Since the present study used data from one time period only, it is difficult to understand how aging, health problems, retirement, and financial conditions change together over the years. Future waves of LASI data may help in understanding whether older adults stop working because of poor health, family responsibilities, or financial difficulties.

More studies involving direct interaction with older adults may also be useful. Statistics alone may not show the actual life situation of older adults. Studies based on speaking directly with older adults may help in understanding why some continue to work even at older ages. The reasons may be different for each person. Some older adults may continue working because of family needs or money problems. Some may also continue working because they do not like staying idle at home.

Certain groups among older adults may need more attention in future studies. Widowed older women may face more financial and social difficulties. Some may not have pension income,

savings, or property in their own name. As a result, their experiences related to work, healthcare access, and daily living conditions may differ from those of other older adults. Older adults who live alone or belong to rural and economically weaker backgrounds may also experience different challenges that are not fully visible in large survey data.

The nature of informal work among older adults also needs further study. In India, many older adults still work in farming, daily labour, small shops, and home-based work. There is not enough information about how safe these jobs are for older people, how much income they provide, or how regular the work is. Future studies may help understand whether older adults continue working mainly because of financial need or whether the work really improves their economic condition.

Medical expenses are another important issue. Even when older adults have health insurance, many still spend their own money for medicines, tests, travel, and regular treatment. Future studies can examine whether increasing medical expenses are one of the reasons why many older adults continue working after retirement age. Regional differences may also need more study. The situation may differ between states and even between villages, towns, and cities in the same state.

The life of older adults may be different from one place to another. In some areas, there may be better job opportunities, hospitals, and local support. In some families, younger members move to other places for work, and older adults may be left alone at home. Future studies can also look at the support available for elderly people. Some older adults may continue working because of family needs or money problems. Some may also continue working because they do not like staying idle at home. Certain groups among older adults may need more attention in future studies. Widowed older women experience significant financial and social problems. Some may not have pension income, savings, or property in their own name.

Mental health is another topic that needs more study in older adults. Problems such as depression, anxiety, loneliness, memory issues, and social isolation may affect both the ability to work and the quality of life of older adults. Many older adults may not openly talk about these problems, and therefore, they may remain unnoticed for a long time. Studying mental health along with economic participation would help in understanding the day-to-day life and well-being of older adults more clearly.

Future studies may also examine whether government schemes and welfare programs are actually helping older adults in daily life. There are different pension schemes, healthcare

services, and employment support programs for older adults. However, all elderly people may not be able to use these services in the same way. Future studies can look at the problems faced while accessing these services and the areas where existing programs are still lacking.

6.4 Conclusion

- Many older adults in India are still working. For most, this is not really a choice. It is mainly because they need the money. This is seen more among men, people in rural areas, and those from weaker economic backgrounds.
- The type of work also matters. A large share is in informal or agricultural work. These jobs are often not very safe or suitable for older age.
- Health is another concern. Many older adults report poor health, but still continue to work. This suggests a kind of trade-off. They keep working even when it affects their well-being.
- Support systems do exist, but they do not always help in a strong way. Insurance and welfare schemes are present, but coverage is uneven and gaps remain. Differences across regions are also quite visible.
- There also seems to be a difference in why people work. Some continue because they have no option. Others, though fewer, are able to work under more stable conditions.
- One pattern stands out. Those who are more vulnerable—because of poor health or lower income—are often the ones still working. At the same time, they have the least support available to them.

Overall, work in later life seems to reflect these larger issues. It is shaped by gaps in health support, limited social security, and a lack of suitable work options for older adults.

APPENDIX A PLAGIARISM REPORT



1 India is undergoing a demographic shift, for the first time in history a significant and growing share of its population is now over the age of 60.(1) According to the Longitudinal Ageing Study in India (LASI, 2017? 18), nearly 10% of India?s population?about 104 million people ?are aged 60 and above. By 2050, this number is expected to reach 319 million, accounting for roughly 20% of the total population. This shift brings important changes to how we think about aging, health, and economic activity. It also challenges long-held ideas that older adults are passive dependents rather than active participants in the economy. (1,2) The idea of older people continuing to work, earn, or contribute economically is often left out of policy discussions. ... (only first 800 chars shown)



Analysis complete. Our feedback is listed below in printable form. Some of the items have been truncated or removed to provide better print compatibility.

Plagiarism Detection

Original Work

Originality: 100%



Your writing demonstrates **no signs of plagiarism**. Great work!

More info on our originality scoring process (<http://www.PaperRater.com/page/plagiarism-detection>) .

Word Choice

Usage of Bad Phrases

Bad Phrase Score: 2.31 (lower is better)

The Bad Phrase Score is based on the quality and quantity of trite or inappropriate words, phrases, egregious misspellings, and cliches found in your paper. You did equal or better than **67%** of the people in your grade.



Good work - your writing demonstrates that you have a strong understanding of which phrases to use, which places you above average.

Style

Usage of Transitional Phrases

Transitional Words Score: 65

This score is based on quality of transitional phrases used within your paper. You did equal or better than **50%** of the people in your grade.



Your usage of transitional phrases may be within an acceptable range, but I know you can do better. Please read the information below and apply the advice to your writing.

One sign of an excellent writer is the use of transitional phrases (e.g. therefore, consequently, furthermore). Transitional words and phrases contribute to the **cohesiveness** (http://www.PaperRater.com/vocab_builder/show/cohesiveness) of a text and allow the sentences to flow smoothly. Without transitional phrases, a text will often seem disorganized and will most likely be difficult to understand. When these special words are used, they provide organization within a text and lead to greater understanding and enjoyment on the part of the reader.

The following transitional phrases were found in your document:

again, specifically, and, since, usually, especially, at the same time, instead, yet, in other words, comparatively, in the same way, even so, even though, however, in fact, for instance, therefore, generally, particularly, in a similar, rather, although, in contrast, nevertheless

Consider using additional transitions where appropriate:

- consequently (http://paperrater.com/vocab_builder/show/consequently)
- moreover (http://paperrater.com/vocab_builder/show/moreover)
- notwithstanding (http://paperrater.com/vocab_builder/show/notwithstanding)
- accordingly (http://paperrater.com/vocab_builder/show/accordingly)
- conversely (http://paperrater.com/vocab_builder/show/conversely)
- ordinarily (http://paperrater.com/vocab_builder/show/ordinarily)

Transitional phrases may be used in various places in a text:

- between paragraphs
- between sentences
- between sentence parts
- within sentence parts

Consider this example:

She is one of the most kind persons I have ever known. **Moreover**, she is generous, patient and possesses a magnificent sense of humor.

The word '**moreover**' contributes to greater unity or cohesion between sentences and allows the text to flow more smoothly.

Style

Sentence Length Info

Total Sentences: 461

Avg. Length: 16.7 words

Short Sentences (< 17 words): 247 (54%)

Long Sentences (> 35 words): 9 (2%)

Sentence Variation: 7.6 words (std deviation)



Great work! Your average sentence length is within an acceptable range.

Line chart of the length of each sentence (first 50 sentences). A jagged chart indicates variation.



Helpful Resources:

- Effective Use of Sentence Length (<http://blog.paperrater.com/2015/05/effective-use-of-sentence-length.html>)

Vocabulary Words

Usage of Academic Vocabulary

Vocabulary Score: 209.97

This score is based on the quantity and quality of scholarly vocab words found in the text. You did equal or better than **71%** of the people in your grade.



Vocabulary Word Count: 294

Percentage of Vocab Words: 5.29%

Vocab Words in this Paper (top 20):

longitudinal, curative, populations, implementation, heterogeneous, disengagement, socio, inclusion, initiatives, governmental, insufficiently, rationale, constituting, attributed, frameworks, contexts, demographic, significant, economic, participants



Great job! Your knowledge of complex words is clearly very strong. Take a look at our **Vocab Builder** (http://www.PaperRater.com/vocab_builder/index) to extend it even more.

Tips

Whether you are writing for a school assignment or professionally, it is imperative that you have a vocabulary that will provide for clear communication of your ideas and thoughts. You need to know the type and level of your audience and adjust your vocabulary accordingly. It is worthwhile to constantly work at improving your knowledge of words. To help with this task, please consider using our **Vocabulary Builder** (http://www.PaperRater.com/vocab_builder/index) to improve your comprehension and usage of words.

APPENDIX B

PYTHON LASI NOTEBOOK CODES – WATERFALL PATTERN

The following sections extract and organize the notebook code sequentially in a waterfall-style structure for direct inclusion in a dissertation.

Step 1

```
import pyreadstat

# Path to your file
file_path = r"C:\Users\ASUS\OneDrive\Desktop\DBA_DATA.sav"

# Read the .sav file
df, meta = pyreadstat.read_sav(file_path)

# Display the first few rows
print(df.head())
```

Step 2

```
# Get column names
column_names = df.columns.tolist()

# Print column names
for name in column_names:
    print(name)
```

Step 3

```
# Get column names
column_names = df.columns.tolist()

# Print column names
for name in column_names:
    print(name)
```

Step 4

```
def classify_categorical_variables(df, categorical_vars):
    var_types = []

    for col in categorical_vars:
        try:
            unique_vals = sorted(df[col].dropna().unique())
            n_unique = len(unique_vals)

            if all(isinstance(val, (int, float)) for val in unique_vals):
```

```

        if n_unique == 2:
            var_type = 'binary'
        elif 2 < n_unique <= 5:
            var_type = 'multinomial'
        else:
            var_type = 'numeric_categorical'
    else:
        var_type = 'text_or_mixed'

    var_types.append((col, unique_vals, var_type))
except Exception as e:
    var_types.append((col, ['error'], f'error: {str(e)}'))

return var_types

# Run classification
cat_var_info = classify_categorical_variables(df, [
    'oral_problems', 'ever_heart_attack', 'fluctuation_in_price_expected', 'multiple_cancer',
    'thyroid_disorder', 'ever_diagnosed_hypertension', 'religious_gatherings', 'sleep_problems',
    'mental_impairment', 'ever_diagnosed_chronic_joint_condition', 'ever_diagnosed_stroke',
    'memory_at_present', 'ever_diagnosed_chronic_heart_disease', 'feeling_lonely',
    'ever_diagnosed_diabetes', 'ever_diagnosed_with_cld', 'ever_diagnosed_high_cholesterol',
    'ever_diagnosed_cancer', 'fall_in_2years', 'ever_reduce_or_skip_meals', 'having_friends',
    'skin_disease', 'gastrointestinal_problems', 'ear_problems', 'other_chronic_conditions',
    'major_injury_2years', 'pass_urine_in_activities', 'pain_problems',
    'covered_with_some_health_insurance', 'current_employment_status',
    'physical_mental_impairment', 'urogenital_condition', 'women_health_problems',
    'vision_problems', 'depression', 'ever_diagnosed_neurological_psychiatry_problem'
])

# Print the results
for col, uniq_vals, col_type in cat_var_info:
    print(f'{col}: {col_type}, Unique values: {uniq_vals}')

```

Step 5

```

type_a = [
    'ever_heart_attack', 'multiple_cancer', 'ever_diagnosed_hypertension',
    'ever_diagnosed_chronic_joint_condition', 'ever_diagnosed_stroke',
    'ever_diagnosed_chronic_heart_disease', 'ever_diagnosed_diabetes',
    'ever_diagnosed_with_cld', 'ever_diagnosed_high_cholesterol',
    'ever_diagnosed_cancer', 'fall_in_2years', 'ever_reduce_or_skip_meals',
    'having_friends', 'ear_problems', 'major_injury_2years',
    'pass_urine_in_activities', 'pain_problems', 'covered_with_some_health_insurance',
    'current_employment_status', 'physical_mental_impairment', 'vision_problems',

```

```

    'depression', 'ever_diagnosed_neurological_psychiatry_problem'
]

```

Step 6

```

for col in type_a:
    print(f'--- {col} ---')
    print(df[col].unique())
    print()

```

Step 7

```

type_b = [
    'thyroid_disorder', 'mental_impairment', 'skin_disease',
    'gastrointestinal_problems', 'other_chronic_conditions'
]

```

Step 8

```

# Convert 1/2 → 'No'/'Yes' (1 = unhealthy → 'Yes', 2 = healthy → 'No')
for col in type_a:
    df[col] = df[col].replace({1: 'Yes', 2: 'No'})

# Convert 1/0 → 'Yes'/'No' (1 = Yes, 0 = No)
for col in type_b:
    df[col] = df[col].replace({1: 'Yes', 0: 'No'})

```

Step 9

```

# Check post-conversion unique values
for col in type_a + type_b:
    print(f'{col}: {sorted(df[col].dropna().unique())}')

```

Step 10

```

# List of binary variables now homogenized
binary_categorical_vars = [
    'ever_heart_attack', 'multiple_cancer', 'ever_diagnosed_hypertension',
    'ever_diagnosed_chronic_joint_condition', 'ever_diagnosed_stroke',
    'ever_diagnosed_chronic_heart_disease', 'ever_diagnosed_diabetes',
    'ever_diagnosed_with_cld', 'ever_diagnosed_high_cholesterol',
    'ever_diagnosed_cancer', 'fall_in_2years', 'ever_reduce_or_skip_meals',
    'having_friends', 'ear_problems', 'major_injury_2years',
    'pass_urine_in_activities', 'pain_problems', 'covered_with_some_health_insurance',

```

```

'current_employment_status', 'physical_mental_impairment', 'vision_problems',
'depression', 'ever_diagnosed_neurological_psychiatry_problem',
'thyroid_disorder', 'mental_impairment', 'skin_disease',
'gastrointestinal_problems', 'other_chornic_conditions'
]

```

```

# Convert to categorical type
for col in binary_categorical_vars:
    df[col] = df[col].astype('category')

```

Step 11

```

# Check data types
df[binary_categorical_vars].dtypes

```

Step 12

```

# List of categorical variables you've finalized
categorical_columns = [
    'ever_heart_attack', 'multiple_cancer', 'ever_diagnosed_hypertension',
    'ever_diagnosed_chronic_joint_condition', 'ever_diagnosed_stroke',
    'ever_diagnosed_chronic_heart_disease', 'ever_diagnosed_diabetes',
    'ever_diagnosed_with_cld', 'ever_diagnosed_high_cholesterol',
    'ever_diagnosed_cancer', 'fall_in_2years', 'ever_reduce_or_skip_meals',
    'having_friends', 'ear_problems', 'major_injury_2years',
    'pass_urine_in_activities', 'pain_problems', 'covered_with_some_health_insurance',
    'current_employment_status', 'physical_mental_impairment', 'vision_problems',
    'depression', 'ever_diagnosed_neurological_psychiatry_problem',
    'thyroid_disorder', 'mental_impairment', 'skin_disease',
    'gastrointestinal_problems', 'other_chornic_conditions'
]

# Display number of missing values for each
missing_vals = df[categorical_columns].isnull().sum()
missing_vals = missing_vals[missing_vals > 0] # Only show columns with missing values

print("Categorical variables with missing values:\n")
print(missing_vals.sort_values(ascending=False))

```

Step 13

```

print(df['urogenital_condition'].value_counts(dropna=False))

```

Step 14

```

import pandas as pd
import numpy as np

# Example: Assume df is your existing DataFrame
# Replace with your actual DataFrame loading method
# df = pd.read_csv("your_file.csv")

# Clean the column
def clean_urogenital_condition(value):
    if pd.isna(value) or value in ['.', '.d']:
        return np.nan
    elif value == '5':
        return 'No'
    else:
        return 'Yes'

df['urogenital_condition'] = df['urogenital_condition'].apply(clean_urogenital_condition)

```

Step 15

```
print(df['urogenital_condition'].unique())
```

Step 16

```

import numpy as np

# Function to clean the column
def clean_women_health_problems(value):
    if value in ['', '.']:
        return np.nan
    elif value == '10':
        return 'No'
    else:
        return 'Yes'

# Apply the function to the column
df['women_health_problems'] =
df['women_health_problems'].apply(clean_women_health_problems)

```

Step 17

```
print(df['women_health_problems'].value_counts(dropna=False))
```

Step 18

```

import numpy as np

# Function to clean oral_problems
def clean_oral_problems(value):
    if pd.isna(value) or value in ['', '.']:
        return np.nan
    elif value == '9':
        return 'No'
    else:
        return 'Yes'

# Apply the function to the column
df['oral_problems'] = df['oral_problems'].apply(clean_oral_problems)

```

Step 19

```

print(df['oral_problems'].value_counts(dropna=False))

```

Step 20

```

# For behaviour_issues:
# If there's any non-null value → 'No' (Unhealthy), else NaN
df['behaviour_issues'] = df['behaviour_issues'].apply(
    lambda x: 'No' if pd.notna(x) else np.nan
)

# For feeling_lonely:
# 1 (Rarely lonely)
# 2–5 (Feels lonely)
df['feeling_lonely'] = df['feeling_lonely'].apply(
    lambda x: 'No' if x == 1 else ('Yes' if x in [2, 3, 4, 5] else np.nan)
)

# Check the output
print(df['feeling_lonely'].value_counts(dropna=False))

```

Step 21

```

df1 = df.copy()

```

Step 22

```
domains = {
    'functional_ability': ['ADL', 'IADL', 'Sensory_deprivation', 'days_bed_ridden', 'frail01'],
    'social_engagement': ['having_friends', 'religious_gatherings'],
    'economic_security': [
        'fluctuation_in_price_expected', 'current_employment_status',
        'money_spend_in_past_1year', 'covered_with_some_health_insurance',
        'ever_reduce_or_skip_meals'
    ]
}
```

```
# Check each column
for domain, columns in domains.items():
    print(f"\n=== Domain: {domain} ===")
    for col in columns:
        if col in df.columns:
            unique_vals = df[col].dropna().unique()
            num_unique = len(unique_vals)
            dtype = df[col].dtype

            # Simple heuristic
            if dtype == 'object' or num_unique <= 10:
                classification = 'Categorical'
            else:
                classification = 'Continuous'

            print(f"{col}: {classification} (dtype={dtype}, unique={num_unique})")
        else:
            print(f"{col}: Not found in DataFrame")
```

Step 23

```
df[['depression', 'feeling_lonely', 'mental_impairment', 'behaviour_issues']].info()
```

Step 24

```
# Define your domains again (excluding 'money_spend_in_past_1year')
columns_to_check = {
    'functional_ability': ['ADL', 'IADL', 'Sensory_deprivation', 'days_bed_ridden', 'frail01'],
    'social_engagement': ['having_friends', 'religious_gatherings'],
    'economic_security': [
        'fluctuation_in_price_expected', 'current_employment_status',
        'covered_with_some_health_insurance', 'ever_reduce_or_skip_meals'
    ]
}
```

```
# Loop to print unique values
for domain, columns in columns_to_check.items():
    print(f"\n=== Domain: {domain} ===")
    for col in columns:
        if col in df.columns:
            print(f"{col}: {sorted(df[col].dropna().unique().tolist())}")
        else:
            print(f"{col}: Not found in DataFrame")
```

Step 25

```
print(df['religious_gatherings'].unique())
print(df['religious_gatherings'].dtype)
```

Step 26

```
# First, convert column to numeric safely, coercing '.' and other non-numeric to NaN
df['religious_gatherings'] = pd.to_numeric(df['religious_gatherings'], errors='coerce')
```

```
# Now apply binarization
```

```
def binarize_religious_gathering(val):
    if pd.isna(val):
        return np.nan
    elif val in [1, 2, 3, 4]:
        return 'Yes'
    elif val in [5, 6]:
        return 'No'
    else:
        return np.nan
```

```
df['religious_gatherings'] = df['religious_gatherings'].apply(binarize_religious_gathering)
```

Step 27

```
print(df['religious_gatherings'].unique())
```

Step 28

```
print(df['religious_gatherings'].value_counts(dropna=False))
```

Step 29

```
# ADL full independence = 12
df['ADL_impaired'] = df['ADL'].apply(
    lambda x: 'No' if x >= 12 else 'Yes' if not pd.isna(x) else np.nan
)
```

Step 30

```
# IADL full independence = 14
df['IADL_impaired'] = df['IADL'].apply(
    lambda x: 'No' if x >= 14 else 'Yes' if not pd.isna(x) else np.nan
)
```

Step 31

```
print(df[['ADL', 'ADL_impaired']].dropna().head())
print(df[['IADL', 'IADL_impaired']].dropna().head())
```

Step 32

```
df['frail01_binary'] = df['frail01'].apply(
    lambda x: 'No' if x >= 24 else 'Yes' if not pd.isna(x) else np.nan
)
print(df['frail01_binary'].value_counts(dropna=False))
```

Step 33

```
# Binary: 1 = Sensory Deprivation (Impaired), 0 = Normal
df['Sensory_deprivation_binary'] = df['Sensory_deprivation'].apply(
    lambda x: 'No' if x >= 17 else 'Yes' if not pd.isna(x) else np.nan
)
print(df['Sensory_deprivation_binary'].value_counts(dropna=False))
```

Step 34

```
def binarize_bedridden_days(x):
    if pd.isna(x):
        return np.nan
    elif x == 0:
        return 'No' # Not bedridden
    else:
        return 'Yes' # Bedridden at least 1 day

df['days_bed_ridden_binary'] = df['days_bed_ridden'].apply(binarize_bedridden_days)
print(df['days_bed_ridden_binary'].value_counts(dropna=False))
```

Step 35

```

columns_by_domain = {
    'physical_health': [
        'ever_diagnosed_hypertension','ever_diagnosed_diabetes','physical_mental_impairment',
        'mental_impairment','ever_diagnosed_cancer','ever_diagnosed_with_cld',
        'ever_diagnosed_chronic_heart_disease','ever_diagnosed_chronic_joint_condition',
        'ever_diagnosed_stroke','ever_diagnosed_neurological_psychiatry_problem',
        'ever_diagnosed_high_cholesterol','ever_heart_attack','urogenital_condition',
        'ear_problems','oral_problems','vision_problems','major_injury_2years',
        'other_chornic_conditions','fall_in_2years','skin_disease','gastrointestinal_problems',
        'thyroid_disorder','women_health_problems','pass_urine_in_activities',
        'pain_problems','multiple_cancer', 'sleep_problems'
    ],
    'mental_health': [
        'depression', 'feeling_lonely', 'mental_impairment', 'behaviour_issues'
    ],
    'functional_ability': ['ADL_impaired', 'IADL_impaired', 'Sensory_deprivation_binary',
    'days_bed_ridden_binary','frail01_binary'],
    'social_engagement': ['having_friends', 'religious_gatherings'],
    'economic_security': [
        'fluctuation_in_price_expected','current_employment_status',
        'money_spend_in_past_1year', 'covered_with_some_health_insurance',
        'ever_reduce_or_skip_meals'
    ]
}

```

```

# Flatten all selected columns

```

```

selected_columns = list({col for domain in columns_by_domain.values() for col in domain})

```

```

# Subset the DataFrame

```

```

selected_df = df[selected_columns].copy()

```

```

# Detect types

```

```

categorical = []

```

```

continuous = []

```

```

for col in selected_columns:

```

```

    unique_vals = df[col].dropna().unique()

```

```

    if df[col].dtype == 'object':

```

```

        categorical.append(col)

```

```

    elif df[col].nunique() <= 5: # Heuristic: few unique values => categorical

```

```

        categorical.append(col)

```

```

    else:

```

```

        continuous.append(col)

```

```

# Print the results

```

```
print("Categorical Variables:")
print(categorical)
```

```
print("\nContinuous Variables:")
print(continuous)
```

Step 36

```
df[['depression', 'feeling_lonely', 'mental_impairment', 'behaviour_issues']].info()
```

Step 37

```
print(f'Unique value count: {df['money_spend_in_past_1year'].nunique(dropna=False)}')
```

Step 38

```
df['mpce_poor'] = df['mpce_amount_in_rs'].apply(
    lambda x: 'No' if x >= 1500 else 'Yes' if not pd.isna(x) else np.nan
)
```

Step 39

```
# Check mean and standard deviation (ignoring NaNs)
mean_value = df['money_to_be_spend_in_next_1year'].mean()
std_dev = df['money_to_be_spend_in_next_1year'].std()

print(f'Mean: {mean_value:.2f}')
print(f'Standard Deviation: {std_dev:.2f}')
```

Step 40

```
# 1 = good (spends ≥ 5000), 0 = low
df['money_spent_lastyear'] = df['money_spend_in_past_1year'].apply(
    lambda x: 'High' if x >= 5000 else 'Low' if not pd.isna(x) else np.nan
)
```

```
# Categorize based on mean
df['spending_category'] = df['money_to_be_spend_in_next_1year'].apply(
    lambda x: 'High' if x > 21209.48 else ('Low' if pd.notna(x) else np.nan)
)
```

Step 41

```
# List of variables to check
variables_to_check = ['current_employment_status', 'immunization', 'living_arrangements']

# Print unique values for each
for var in variables_to_check:
    print(f"=== {var} ===")
    print(df[var].unique())
    print()
```

Step 42

```
def binarize_living_arrangements(val):
    if pd.isna(val):
        return np.nan
    elif str(val).startswith('1'):
        return 'Yes' # Living alone → vulnerable
    elif str(val).startswith(('2', '3', '4', '5')):
        return 'No' # Any form of co-living → socially supported
    else:
        return np.nan

df['living_alone'] = df['living_arrangements'].apply(binarize_living_arrangements)
print(df['living_alone'].value_counts(dropna=False))
```

Step 43

```
def binarize_immunization(val):
    val_str = str(val).strip()
    if val_str in ['.', '.d', '.e', '']:
        return np.nan
    elif val_str in ['7', '7 None']:
        return 'No' # No immunization
    else:
        return 'Yes' # Any other combo means at least one immunization

df['immunization_binary'] = df['immunization'].apply(binarize_immunization)
print(df['immunization_binary'].value_counts(dropna=False))
```

Step 44

```
# Show unique values in the column
print(df['fluctuation_in_price_expected'].unique())
```

Step 45

```
df['raise_in_price'] = df['fluctuation_in_price_expected'].apply(
    lambda x: 'Yes' if x == 1 else ('No' if x in [2, 3] else np.nan)
)
```

Step 46

```
df['sleep_problems'] = df['sleep_problems'].apply(
    lambda x: 'No' if x in [1, 2] else ('Yes' if x in [3, 4] else np.nan)
)
```

Step 47

```
df[['depression', 'feeling_lonely', 'mental_impairment', 'behaviour_issues']].info()
```

Step 48

```
df3 = df.copy()
```

Step 49

```
columns_by_domain = {
    'physical_health': [
        'ever_diagnosed_hypertension', 'ever_diagnosed_diabetes',
        'physical_mental_impairment',
        'ever_diagnosed_cancer', 'ever_diagnosed_with_cld',
        'ever_diagnosed_chronic_heart_disease', 'ever_diagnosed_chronic_joint_condition',
        'ever_diagnosed_stroke', 'ever_diagnosed_neurological_psychiatry_problem',
        'ever_diagnosed_high_cholesterol', 'ever_heart_attack', 'urogenital_condition',
        'ear_problems', 'oral_problems', 'vision_problems', 'major_injury_2years',
        'other_chornic_conditions', 'fall_in_2years', 'skin_disease', 'gastrointestinal_problems',
        'thyroid_disorder', 'women_health_problems', 'pass_urine_in_activities',
        'pain_problems', 'multiple_cancer', 'sleep_problems',
        'immunization_binary' # if available and clean
    ],
    'mental_health': [
        'depression', 'feeling_lonely', 'mental_impairment', 'behaviour_issues'
    ],
    'functional_ability': [
        'ADL_impaired', 'IADL_impaired', 'Sensory_deprivation_binary',
        'days_bed_ridden_binary', 'frail01_binary'
    ],
    'social_engagement': [
        'having_friends',
        'religious_gatherings',
        'living_alone'
    ]
}
```

```

],
'economic_security': [
    'raise_in_price', # Include, but interpret carefully
    'mpce_poor',
    'covered_with_some_health_insurance',
    'spending_category',
    'ever_reduce_or_skip_meals',
    'money_spent_lastyear',
    'current_employment_status'
]
}

```

Step 50

```

from collections import Counter

# Flatten the domain dictionary to get all column names
all_columns = sum(columns_by_domain.values(), [])

# Check value counts for each
for col in all_columns:
    if col in df3.columns:
        print(f'--- {col} ---')
        print(df3[col].value_counts(dropna=False))
        print()

```

Step 51

```

# Flatten all domain variables into one list
all_vars = []
for domain, cols in columns_by_domain.items():
    all_vars.extend(cols)

# Check if each variable is binary (only 0, 1, or NaN)
non_binary_vars = {}

for var in all_vars:
    if var in df.columns:
        unique_vals = df[var].dropna().unique()
        binary_set = set([0, 1])
        if not set(unique_vals).issubset(binary_set):
            non_binary_vars[var] = unique_vals
    else:
        print(f"⚠ Column not found in DataFrame: {var}")

```

```
# Output
if non_binary_vars:
    print("❌ These variables are NOT strictly binary (0/1):\n")
    for var, vals in non_binary_vars.items():
        print(f'{var}: {sorted(vals)}')
else:
    print("✅ All listed variables are binary (0/1) categorical.")
```

Step 52

```
df['immunization_binary'] = df['immunization_binary'].replace({
    'Yes': 'High',
    'No': 'Low'
})

df['religious_gatherings'] = df['religious_gatherings'].replace({
    'Yes': 'High',
    'No': 'Low'
})

df['current_employment_status'] = df['current_employment_status'].astype(str).replace({
    'Yes': 'High',
    'No': 'Low'
})
```

Step 53

```
df['current_employment_status'] = df['current_employment_status'].astype(str).replace({
    'Yes': 'High',
    'No': 'Low'
}).astype('category')
```

Step 54

```
# Define variables and binary mapping function
to_binary_vars = [
    'treated_poorly_in_stores_hotels',
    'people_act_they_are_smart',
    'people_act_afraid',
    'threatened_harassed'
]

# Create a dictionary of new columns
```

```

binary_columns = {
    var + '_binary': df[var].apply(
        lambda x: 'Yes' if x in [1, 2, 3, 4] else ('No' if x in [5, 6] else pd.NA)
    )
    for var in to_binary_vars if var in df.columns
}

```

```

# Combine all new columns in one go using pd.concat
df = pd.concat([df, pd.DataFrame(binary_columns)], axis=1)

```

```

# Optional: Defragment the DataFrame
df = df.copy()

```

Step 55

```

# Variables to reframe
subjective_positive_vars = [
    'life_close_to_ideal',
    'life_conditions_excellent',
    'satisfied_with_life',
    'got_important_things_in_life',
    'live_life_again', 'treated_with_respect'
]

# Apply the mapping: 1–4 => "Yes" (i.e., now negatively worded), 5–7 => "No"
for var in subjective_positive_vars:
    if var in df.columns:
        df[var + '_reframed'] = df[var].apply(
            lambda x: 'Yes' if x in [1, 2, 3, 4] else ('No' if x in [5, 6, 7] else pd.NA)
        )
    else:
        print(f"⚠ Column not found: {var}")

```

Step 56

```

# Define the variables and the new mapping rule
inner_peace_vars = [
    'feeling_inner_peace',
    'spiritually_touched',
    'thankful_for_things_in_life',
    'selflessly_caring_for_others'
]

# Apply recoding: 4/5 → "Yes", 1–3 → "No"

```

```

for var in inner_peace_vars:
    if var in df.columns:
        df[var + '_mapped'] = df[var].apply(
            lambda x: 'Yes' if x in [4, 5] else ('No' if x in [1, 2, 3] else pd.NA)
        )
    else:
        print(f" ⚠ Column not found: {var}")

```

Step 57

```

economic_security_vars = [
    'raise_in_price', 'mpce_poor', 'covered_with_some_health_insurance',
    'spending_category', 'ever_reduce_or_skip_meals',
    'money_spent_lastyear', 'current_employment_status',
    'financial_aid_for_treatment', 'total_value_of_financial_help',
    'treatment_discontinued_monetary_reasons'
]

# Display unique values for each column
for var in economic_security_vars:
    if var in df.columns:
        print(f"\n--- Unique values for '{var}' ---")
        print(df[var].dropna().unique())
    else:
        print(f"\n ⚠ Column '{var}' not found in DataFrame.")

```

Step 58

```

# Step 1: Compute the median (ignoring NaNs)
median_help = df['total_value_of_financial_help'].median()

# Step 2: Convert into High / Low
df['total_value_of_financial_help_cat'] = df['total_value_of_financial_help'].apply(
    lambda x: 'High' if pd.notna(x) and x >= median_help else ('Low' if pd.notna(x) else
pd.NA)
)

# Optional: print median value
print(f'Median total_value_of_financial_help: {median_help}')

```

Step 59

```
print("Unique values in 'total_value_of_financial_help_cat':")
print(df['total_value_of_financial_help_cat'].dropna().unique())
```

Step 60

```
df['treatment_discontinued_monetary_reasons'] =
df['treatment_discontinued_monetary_reasons'].apply(
    lambda x: 1 if x == 0 else (0 if x == 1 else pd.NA)
)
```

Step 61

```
import pandas as pd
import numpy as np
```

```
# Step 1: Define variables by domain
```

```
healthcare_domains = {
    'access': [
        'consult_healthcare_providers', 'visit_to_healthcare_facility',
        'distance_of_hospital', 'mode_of_transport_to_hospital',
        'covered_with_some_health_insurance', 'last_HI_premium_paid'
    ],
    'utilization': [
        'numer_opd_visits_1year', 'number_of_hospital_admission_lastyear',
        'current_health', 'treatment_discontinued_monetary_reasons'
    ],
    'affordability': [
        'money_spent_on_medicine', 'financial_aid_for_treatment',
        'cost_opd_visit', 'money_spend_in_past_1year',
        'total_value_of_financial_help'
    ],
    'satisfaction': [
        'treated_with_respect', 'satisfaction_with_treatment_recieved',
        'QoC_cleanliness_health_facility', 'QoC_clear_explanation',
        'QoC_waiting_time', 'QoC_respect',
        'QoC_priv_talk_with_doctor', 'QoC_provider_of_your_choice'
    ]
}
```

```
# Step: Loop through each variable and classify
for domain, cols in healthcare_domains.items():
    print(f"\n--- Domain: {domain.upper()} ---")
    for col in cols:
        if col in df.columns:
            unique_vals = df[col].dropna().unique()
```

```

n_unique = len(unique_vals)
dtype = df[col].dtype

if dtype == 'object' or n_unique <= 10:
    var_type = 'Categorical'
else:
    var_type = 'Continuous'

print(f'{col:35} | Type: {var_type:12} | Unique values: {n_unique}')
else:
    print(f'{col:35} | ❌ Column not found")

```

Step 62

```

# List of satisfaction-related variables to recode
satisfaction_vars = [
    'treated_with_respect',
    'satisfaction_with_treatment_recieved',
    'QoC_cleanliness_health_facility',
    'QoC_clear_explanation',
    'QoC_waiting_time',
    'QoC_respect',
    'QoC_priv_talk_with_doctor',
    'QoC_provider_of_your_choice',
    'satisfied_with_treatment_hospitalization' # typo retained as per your column
]

# Recode each into High/Low after type casting
for col in satisfaction_vars:
    if col in df.columns:
        df[col + '_cat'] = df[col].apply(
            lambda x: 'High' if pd.to_numeric(x, errors='coerce') in [1, 2]
            else ('Low' if pd.to_numeric(x, errors='coerce') in [3,4, 5, 6]
            else pd.NA)
        )
    else:
        print(f"❌ Column not found: {col}")

```

Step 63

```

for col in satisfaction_vars:
    cat_col = col + '_cat'
    if cat_col in df.columns:
        print(f"\n{cat_col}:")

```

```
print(df[cat_col].value_counts(dropna=False))
```

Step 64

```
# Variables to check
health_access_vars = [
    'consult_healthcare_providers',
    'visit_to_healthcare_facility',
    'numer_opd_visits_1year',
    'number_of_hospital_admission_lastyear',
    'current_health',
    'time_reach_opd',
    'dist_from_hospital',
    'travel_time_hospital'
]

# Print unique values and frequencies
for var in health_access_vars:
    if var in df.columns:
        print(f"\n--- {var} ---")
        print(df[var].value_counts(dropna=False).sort_index())
    else:
        print(f"\n ❌ Column not found: {var}")
```

Step 65

```
# Step 1: Convert values to string and classify based on presence of "1"
df['consult_with_allopathy'] = df['consult_healthcare_providers'].astype(str).apply(
    lambda x: 'High' if '1' in x else 'Low'
)

df['consult_with_allopathy'].value_counts(dropna=False)
```

Step 66

```
continuous_vars = [
    'time_reach_opd',
    'dist_from_hospital',
    'travel_time_hospital',
    'numer_opd_visits_1year',
    'number_of_hospital_admission_lastyear'
]

for var in continuous_vars:
```

```

if var in df.columns:
    median_val = df[var].median(skipna=True)
    if var in ['time_reach_opd', 'dist_from_hospital', 'travel_time_hospital']:
        # Invert logic: lower value is better (High access)
        df[var + '_bin'] = df[var].apply(
            lambda x: 'High' if pd.notna(x) and x <= median_val else ('Low' if pd.notna(x) else
pd.NA)
        )
    else:
        # Utilization: higher value is better
        df[var + '_bin'] = df[var].apply(
            lambda x: 'High' if pd.notna(x) and x > median_val else ('Low' if pd.notna(x) else
pd.NA)
        )
    else:
        print(f"❌ Column not found: {var}")
for var in continuous_vars:
    bin_col = var + '_bin'
    if bin_col in df.columns:
        print(f"\n{bin_col} distribution:")
        print(df[bin_col].value_counts(dropna=False))

```

Step 67

```

df['current_health_bin'] = df['current_health'].apply(
    lambda x: 'High' if x in [1.0, 2.0, 3.0] else ('Low' if x in [4.0, 5.0] else pd.NA)
)

```

Step 68

```

# 1. Reverse map for string values
reverse_map = {'No': 1, 'Yes': 0, 'High': 1, 'Low': 0}

```

Step 69

```

# Step 1: Rename the column
df.rename(columns={'memory_at_present': 'memory_impairment'}, inplace=True)

# Step 2: Recode values: 5 and 4 → 0 (Yes), all others → 1 (No)
df['memory_impairment'] = df['memory_impairment'].apply(lambda x: 0 if x in [4, 5] else 1
if pd.notnull(x) else np.nan)

```

Step 70

```
df['felt_ill_treated'] = df['felt_ill_treated'].apply(lambda x: 0 if pd.isna(x) else 1)
```

Step 71

```
df['frustrated_with_life'] = df['life_close_to_ideal'].apply(lambda x: 1 if x in [5, 6, 7] else (0 if x in [1, 2, 3, 4] else pd.NA))
```

Step 72

```
import pandas as pd
from pandas.api.types import is_numeric_dtype

# Step 1: Clean reverse mapping
reverse_map = {'Yes': 0, 'No': 1, 'High': 0, 'Low': 1}
# Step 2: All domain variables grouped
columns_by_domain = {
    'physical_health': [
        'ever_diagnosed_hypertension', 'ever_diagnosed_diabetes',
        'ever_diagnosed_cancer', 'ever_diagnosed_with_cld',
        'ever_diagnosed_chronic_heart_disease', 'ever_diagnosed_chronic_joint_condition',
        'ever_diagnosed_stroke',
        'ever_diagnosed_high_cholesterol', 'ever_heart_attack', 'urogenital_condition',
        'oral_problems', 'major_injury_2years',
        'other_chronic_conditions', 'fall_in_2years', 'skin_disease', 'gastrointestinal_problems',
        'thyroid_disorder', 'women_health_problems',
        'pain_problems', 'multiple_cancer', 'sleep_problems',
        'immunization_binary'
    ],
    'mental_well_being': [
        'depression', 'feeling_lonely', 'mental_impairment', 'behaviour_issues',
        'memory_impairment', 'felt_ill_treated', 'frustrated_with_life', 'having_friends',
        'religious_gatherings', 'living_alone',
        'feeling_inner_peace', 'spiritually_touched', 'thankful_for_things_in_life', 'selflessly_caring_for_others',
        'treated_with_respect', 'life_close_to_ideal', 'life_conditions_excellent', 'satisfied_with_life', 'got_important_things_in_life', 'live_life_again'
    ],
    'functional_ability': [
        'ADL_impaired', 'IADL_impaired', 'Sensory_deprivation_binary',
        'frail01_binary', 'ear_problems', 'vision_problems'
    ],
    'economic_security': [
        'raise_in_price', 'mpce_poor', 'covered_with_some_health_insurance',
        'spending_category', 'ever_reduce_or_skip_meals',
        'money_spent_lastyear', 'current_employment_status', 'financial_aid_for_treatment',
```

```

        'total_value_of_financial_help', 'treatment_discontinued_monetary_reasons'
    ],
    'healthcare_satisfaction_vars': [
        'treated_with_respect_cat',
        'satisfaction_with_treatment_recieved_cat',
        'QoC_cleanliness_health_facility_cat',
        'QoC_clear_explanation_cat',
        'QoC_waiting_time_cat',
        'QoC_respect_cat',
        'QoC_priv_talk_with_doctor_cat',
        'QoC_provider_of_your_choice_cat',
        'satisfied_with_treatment_hospitalization_cat' # typo retained as per your column
    ],

}

# Step 2: Recode all domain variables using reverse_map
for domain, cols in columns_by_domain.items():
    for col in cols:
        if col in df.columns:
            df[col] = df[col].replace(reverse_map)

            # Force all to numeric (coerce non-convertibles to NaN)
            df[col] = pd.to_numeric(df[col], errors='coerce')

            # Optional: Cap all values outside {0,1}
            df[col] = df[col].where(df[col].isin([0, 1]), other=pd.NA)

# Step 3: Compute domain indices (proportion of 1s)
domain_indices = []
index_series = {}

for domain, cols in columns_by_domain.items():
    valid = df[cols].notnull().sum(axis=1)
    healthy = df[cols].sum(axis=1)
    index = healthy / valid
    index_name = f'{domain}_index'
    index_series[index_name] = index
    domain_indices.append(index_name)

# Step 4: Assign domain indices to DataFrame
df = pd.concat([df, pd.DataFrame(index_series)], axis=1)

# Step 5: Compute overall HAI score (0 to 1)
valid_indices = df[domain_indices].notnull().sum(axis=1)

```

```
sum_indices = df[domain_indices].sum(axis=1)
df['HAI_score'] = sum_indices / valid_indices
```

Step 73

```
df[domain_indices + ['HAI_score']].describe()
```

Step 74

```
'HAI_score' in df.columns
```

Step 75

```
df['HAI_score'] = df['HAI_score'].round(2)
```

```
def categorize_hai(score):
    if pd.isna(score):
        return 'Missing'
    elif score >= 0.75:
        return 'Healthy'
    elif score >= 0.5:
        return 'Moderate'
    else:
        return 'Severe'
```

```
df['HAI_category'] = df['HAI_score'].apply(categorize_hai)
```

Step 76

```
for domain, cols in columns_by_domain.items():
    domain_valid = df[cols].notnull().sum(axis=1)
    domain_healthy = df[cols].sum(axis=1)
    df[f'{domain}_index'] = domain_healthy / domain_valid
```

Step 77

```
df['HAI_score'].unique()
```

Step 78

```
# Step 3: Apply reverse mapping and convert to numeric
for domain, cols in columns_by_domain.items():
    for col in cols:
        if col in df.columns:
```

```
df[col] = df[col].replace(reverse_map).infer_objects()
df[col] = pd.to_numeric(df[col], errors='coerce')
```

Step 4: Compute domain-wise index and store in dictionary

```
domain_indices = []
```

```
index_series = {}
```

```
for domain, cols in columns_by_domain.items():
```

```
    valid = df[cols].notnull().sum(axis=1)    # Valid (non-NaN) responses
```

```
    healthy = df[cols].sum(axis=1)           # Sum of '1's (healthy items)
```

```
    index = healthy / valid                  # Proportion of healthy responses
```

```
    index_name = f'{domain}_index'
```

```
    df[index_name] = index                  # Store directly in df
```

```
    domain_indices.append(index_name)
```

Step 5: Calculate HAI score

```
valid_indices = df[domain_indices].notnull().sum(axis=1)
```

```
sum_indices = df[domain_indices].sum(axis=1)
```

```
df['HAI_score'] = sum_indices / valid_indices
```

Step 79

```
for domain_index in domain_indices:
```

```
    print(f"\n ♦ {domain_index} summary:")
```

```
    print(df[domain_index].describe())
```

Step 80

```
for domain, cols in columns_by_domain.items():
```

```
    print(f"\n ■ DOMAIN: {domain.upper()}")
```

```
    for col in cols:
```

```
        if col in df.columns:
```

```
            print(f"\n--- {col} ---")
```

```
            print(df[col].value_counts(dropna=False))
```

```
        else:
```

```
            print(f" ⚠ Column not found: {col}")
```

Step 81

```
# List all columns that end with '_index'
```

```
domain_index_columns = [col for col in df3.columns if col.endswith('_index')]
```

```
print("Domain index columns:\n")
```

```
for col in domain_index_columns:
```

```
    print(col)
```

Step 82

```
# Check for duplicate column names
duplicates = df3.columns[df3.columns.duplicated()].tolist()
print("Duplicate columns:\n", duplicates)
```

Step 83

```
df3 = df3.loc[:, ~df3.columns.duplicated()]
```

Step 84

```
from collections import Counter

col_counter = Counter(df3.columns)
new_columns = []
col_counts = {}

for col in df3.columns:
    if col_counter[col] > 1:
        count = col_counter.get(col, 0)
        new_col = f"{col}_{count}" if count > 0 else col
        new_columns.append(new_col)
        col_counts[col] = count + 1
    else:
        new_columns.append(col)

df3.columns = new_columns
```

Step 85

```
df3 = df3.copy()
```

Step 86

```
df3 = df3[df3['Age'] >= 60]
```

Step 87

```
df3.shape[0]
```

Step 88

```
pip install geopandas matplotlib pandas
```

Step 89

```

import geopandas as gpd
import pandas as pd
import matplotlib.pyplot as plt

# Load the GeoJSON file
gdf_states =
gpd.read_file(r"C:\Users\ASUS\Downloads\gadm41_IND_1.json\gadm41_IND_1.json")

# View available columns (usually includes 'NAME_1' for state names)
print(gdf_states.columns)
print(gdf_states[['NAME_1']].drop_duplicates())

```

Step 90

```

hai_summary = df3.groupby('stateid')['HAI_score'].agg(
    Average_HAI='mean',
    Median_HAI='median',
    Min_HAI='min',
    Max_HAI='max'
).reset_index()

# Optional: round for neatness
hai_summary = hai_summary.round(3)

print(hai_summary)

```

Step 91

```

state_id_to_name = {
    1: "Jammu and Kashmir",
    2: "Himachal Pradesh",
    3: "Punjab",
    4: "Chandigarh",
    5: "Uttarakhand",
    6: "Haryana",
    7: "Delhi",
    8: "Rajasthan",
    9: "Uttar Pradesh",
    10: "Bihar",
    11: "Sikkim",
    12: "Arunachal Pradesh",
    13: "Nagaland",
    14: "Manipur",
    15: "Mizoram",

```

```

16: "Tripura",
17: "Meghalaya",
18: "Assam",
19: "West Bengal",
20: "Jharkhand",
21: "Odisha",
22: "Chhatisgarh",
23: "Madhya Pradesh",
24: "Gujarat",
25: "Daman and Diu",
26: "Dadra and Nagar Haveli",
27: "Maharashtra",
28: "Andhra Pradesh",
29: "Karnataka",
30: "Goa",
31: "Lakshadweep",
32: "Kerala",
33: "Tamil Nadu",
34: "Puducherry",
35: "Andaman and Nicobar",
36: "Telangana"
}

```

Step 92

```
df3['state_name'] = df3['stateid'].map(state_id_to_name)
```

Step 93

```

hai_summary = df3.groupby('stateid')['HAI_score'].agg(
    Average_HAI='mean',
    Median_HAI='median',
    Min_HAI='min',
    Max_HAI='max'
).reset_index()

```

```
# Optional: round for neatness
```

```
hai_summary = hai_summary.round(3)
```

```
print(hai_summary)
```

Step 94

```

import geopandas as gpd
import pandas as pd

```

```

import matplotlib.pyplot as plt

# Load the GeoJSON file
gdf_states =
gpd.read_file(r"C:\Users\ASUS\Downloads\gadm41_IND_1.json\gadm41_IND_1.json")

# View available columns (usually includes 'NAME_1' for state names)
print(gdf_states.columns)
print(gdf_states[['NAME_1']].drop_duplicates())

```

Step 95

```

import geopandas as gpd
import pandas as pd
import matplotlib.pyplot as plt

# Step 1: Load the GeoJSON
gdf =
gpd.read_file(r"C:\Users\ASUS\Downloads\gadm41_IND_1.json\gadm41_IND_1.json")

# Step 2: Clean state names
gdf['state_clean'] = gdf['NAME_1'].str.replace(" ", "").str.title()

# Fix mismatches
gdf['state_clean'] = gdf['state_clean'].replace({
    'NcTofdelhi': 'Delhi',
    'Nctofdelhi': 'Delhi',
    'Chhatisgarh': 'Chhattisgarh'
})

# Remove duplicates by dissolving
gdf = gdf.dissolve(by='state_clean', as_index=False)

# Step 3: State ID to name mapping
state_id_to_name = {
    1: "Jammu and Kashmir", 2: "Himachal Pradesh", 3: "Punjab", 4: "Chandigarh",
    5: "Uttarakhand", 6: "Haryana", 7: "Delhi", 8: "Rajasthan", 9: "Uttar Pradesh",
    10: "Bihar", 11: "Sikkim", 12: "Arunachal Pradesh", 13: "Nagaland",
    14: "Manipur", 15: "Mizoram", 16: "Tripura", 17: "Meghalaya", 18: "Assam",
    19: "West Bengal", 20: "Jharkhand", 21: "Odisha", 22: "Chhattisgarh",
    23: "Madhya Pradesh", 24: "Gujarat", 25: "Daman and Diu", 26: "Dadra and Nagar
Haveli",
    27: "Maharashtra", 28: "Andhra Pradesh", 29: "Karnataka", 30: "Goa",
    31: "Lakshadweep", 32: "Kerala", 33: "Tamil Nadu", 34: "Puducherry",

```

```

    35: "Andaman and Nicobar", 36: "Telangana"
}

# Step 4: Prepare df3
df3['state_name'] = df3['stateid'].map(state_id_to_name)
df3['state_clean'] = df3['state_name'].str.replace(" ", "").str.title()

# Define age groups
age_groups = {
    '60-74': df3[(df3['Age'] >= 60) & (df3['Age'] <= 74)],
    '75-84': df3[(df3['Age'] >= 75) & (df3['Age'] <= 84)],
    '85+': df3[df3['Age'] >= 85]
}

# Step 5: Generate map for each age group
for group_label, group_df in age_groups.items():
    hai_by_state = group_df.groupby('state_clean')['HAI_score'].median().reset_index()
    hai_by_state['HAI_score'] = hai_by_state['HAI_score'].clip(0, 1)

merged = gdf.merge(hai_by_state, on='state_clean', how='left')

fig, ax = plt.subplots(figsize=(13, 13))
merged.plot(
    column='HAI_score',
    cmap='summer',
    linewidth=0.5,
    ax=ax,
    edgecolor='black',
    legend=True,
    missing_kwds={"color": "lightgrey", "label": "No data"}
)

# Labels
for idx, row in merged.iterrows():
    point = row['geometry'].representative_point()
    label = f"{row['state_clean']}\n{row['HAI_score']:.2f}" if pd.notnull(row['HAI_score'])
    else f"{row['state_clean']}\nN/A"
    ax.text(point.x, point.y, label, fontsize=6.5, ha='center', va='center', color='black')

ax.set_title(f"Healthy Aging Index (HAI) by State – Age {group_label}", fontsize=16)
ax.axis('off')
plt.tight_layout()
plt.show()

```

Step 96

```
import geopandas as gpd
import pandas as pd
import matplotlib.pyplot as plt

# Step 1: Load the GeoJSON
gdf =
gpd.read_file(r"C:\Users\ASUS\Downloads\gadm41_IND_1.json\gadm41_IND_1.json")

# Step 2: Clean state names
gdf['state_clean'] = gdf['NAME_1'].str.replace(" ", "").str.title()

# Step 2.5: Fix known mismatches
gdf['state_clean'] = gdf['state_clean'].replace({
    'NcTofdelhi': 'Delhi',
    'Nctofdelhi': 'Delhi',
    'Chhatisgarh': 'Chhattisgarh'
})

# Step 2.6: Remove duplicate geometries by dissolving
gdf = gdf.dissolve(by='state_clean', as_index=False)

# Step 3: Prepare HAI data
state_id_to_name = {
    1: "Jammu and Kashmir", 2: "Himachal Pradesh", 3: "Punjab", 4: "Chandigarh",
    5: "Uttarakhand", 6: "Haryana", 7: "Delhi", 8: "Rajasthan", 9: "Uttar Pradesh",
    10: "Bihar", 11: "Sikkim", 12: "Arunachal Pradesh", 13: "Nagaland",
    14: "Manipur", 15: "Mizoram", 16: "Tripura", 17: "Meghalaya", 18: "Assam",
    19: "West Bengal", 20: "Jharkhand", 21: "Odisha", 22: "Chhattisgarh",
    23: "Madhya Pradesh", 24: "Gujarat", 25: "Daman and Diu", 26: "Dadra and Nagar
Haveli",
    27: "Maharashtra", 28: "Andhra Pradesh", 29: "Karnataka", 30: "Goa",
    31: "Lakshadweep", 32: "Kerala", 33: "Tamil Nadu", 34: "Puducherry",
    35: "Andaman and Nicobar", 36: "Telangana"
}

# Map state ID to name in df3
df3['state_name'] = df3['stateid'].map(state_id_to_name)
df3['state_clean'] = df3['state_name'].str.replace(" ", "").str.title()

# Step 4: Aggregate HAI (use median)
hai_by_state = df3.groupby('state_clean')['HAI_score'].median().reset_index()
hai_by_state['HAI_score'] = hai_by_state['HAI_score'].clip(lower=0, upper=1) # Ensure
range 0–1
```

```

# Step 5: Merge with GeoDataFrame
merged = gdf.merge(hai_by_state, on='state_clean', how='left')

# Step 6: Plot the map
fig, ax = plt.subplots(figsize=(13, 13))
merged.plot(
    column='HAI_score',
    cmap='summer', # Light, readable palette
    linewidth=0.5,
    ax=ax,
    edgecolor='black',
    legend=True,
    missing_kwds={"color": "lightgrey", "label": "No data"}
)

# Step 7: Add labels with state name and HAI score
for idx, row in merged.iterrows():
    point = row['geometry'].representative_point()
    x, y = point.x, point.y

    state_label = row['state_clean']
    hai_val = f"{row['HAI_score']:.2f}" if pd.notnull(row['HAI_score']) else "N/A"
    label = f"{state_label}\n{hai_val}"

    ax.text(x, y, label, fontsize=6.5, ha='center', va='center', color='black')

# Final touches
ax.set_title("Healthy Aging Index (HAI) by Indian State", fontsize=16)
ax.axis('off')
plt.tight_layout()
plt.show()

```

Step 97

```

import geopandas as gpd
import pandas as pd
import matplotlib.pyplot as plt
from scipy.stats import zscore

# Step 1: Load the GeoJSON
gdf =
gpd.read_file(r"C:\Users\ASUS\Downloads\gadm41_IND_1.json\gadm41_IND_1.json")

# Step 2: Clean state names

```

```

gdf['state_clean'] = gdf['NAME_1'].str.replace(" ", "").str.title()

# Step 2.5: Fix known mismatches
gdf['state_clean'] = gdf['state_clean'].replace({
    'NcTofdelhi': 'Delhi',
    'Nctofdelhi': 'Delhi',
    'Chhatisgarh': 'Chhattisgarh'
})

# Step 2.6: Remove duplicate geometries by dissolving
gdf = gdf.dissolve(by='state_clean', as_index=False)

# Step 3: Prepare HAI data
state_id_to_name = {
    1: "Jammu and Kashmir", 2: "Himachal Pradesh", 3: "Punjab", 4: "Chandigarh",
    5: "Uttarakhand", 6: "Haryana", 7: "Delhi", 8: "Rajasthan", 9: "Uttar Pradesh",
    10: "Bihar", 11: "Sikkim", 12: "Arunachal Pradesh", 13: "Nagaland",
    14: "Manipur", 15: "Mizoram", 16: "Tripura", 17: "Meghalaya", 18: "Assam",
    19: "West Bengal", 20: "Jharkhand", 21: "Odisha", 22: "Chhattisgarh",
    23: "Madhya Pradesh", 24: "Gujarat", 25: "Daman and Diu", 26: "Dadra and Nagar
Haveli",
    27: "Maharashtra", 28: "Andhra Pradesh", 29: "Karnataka", 30: "Goa",
    31: "Lakshadweep", 32: "Kerala", 33: "Tamil Nadu", 34: "Puducherry",
    35: "Andaman and Nicobar", 36: "Telangana"
}

# Map state ID to name
df3['state_name'] = df3['stateid'].map(state_id_to_name)
df3['state_clean'] = df3['state_name'].str.replace(" ", "").str.title()

# Step 4: Aggregate mean HAI
hai_by_state = df3.groupby('state_clean')['HAI_score'].mean().reset_index()
hai_by_state['HAI_score'] = hai_by_state['HAI_score'].clip(lower=0, upper=1)

# Step 4.5: Z-score standardization of mean HAI scores
hai_by_state['HAI_zscore'] = zscore(hai_by_state['HAI_score'], nan_policy='omit')

# Step 5: Merge with GeoDataFrame
merged = gdf.merge(hai_by_state, on='state_clean', how='left')

# Step 6: Plot the standardized score map
fig, ax = plt.subplots(figsize=(13, 13))
merged.plot(
    column='HAI_zscore',
    cmap='summer',

```

```

linewidth=0.5,
ax=ax,
edgecolor='black',
legend=True,
missing_kwds={"color": "lightgrey", "label": "No data"}
)

# Step 7: Add state name + z-score
for idx, row in merged.iterrows():
    point = row['geometry'].representative_point()
    x, y = point.x, point.y

    state_label = row['state_clean']
    hai_val = f'{row["HAI_zscore"]:.2f}' if pd.notnull(row['HAI_zscore']) else "N/A"
    label = f'{state_label}\n{hai_val}'

    ax.text(x, y, label, fontsize=6.5, ha='center', va='center', color='black')

# Final touches
ax.set_title("Standardized Healthy Aging Index (Z-Score) by Indian State", fontsize=16)
ax.axis('off')
plt.tight_layout()
plt.show()

```

Step 98

```

import geopandas as gpd
import pandas as pd
import matplotlib.pyplot as plt

# --- Step 1: Load and clean shapefile ---
gdf =
gpd.read_file(r"C:\Users\ASUS\Downloads\gadm41_IND_1.json\gadm41_IND_1.json")
gdf['state_clean'] = gdf['NAME_1'].str.replace(" ", "").str.title()

# Fix known mismatches
gdf['state_clean'] = gdf['state_clean'].replace({
    'NcTofdelhi': 'Delhi',
    'Nctofdelhi': 'Delhi',
    'Chhatisgarh': 'Chhattisgarh'
})

# Remove duplicate geometries
gdf = gdf.dissolve(by='state_clean', as_index=False)

```

```

# --- Step 2: Add state name info to df3 ---
state_id_to_name = {
    1: "Jammu and Kashmir", 2: "Himachal Pradesh", 3: "Punjab", 4: "Chandigarh",
    5: "Uttarakhand", 6: "Haryana", 7: "Delhi", 8: "Rajasthan", 9: "Uttar Pradesh",
    10: "Bihar", 11: "Sikkim", 12: "Arunachal Pradesh", 13: "Nagaland",
    14: "Manipur", 15: "Mizoram", 16: "Tripura", 17: "Meghalaya", 18: "Assam",
    19: "West Bengal", 20: "Jharkhand", 21: "Odisha", 22: "Chhattisgarh",
    23: "Madhya Pradesh", 24: "Gujarat", 25: "Daman and Diu", 26: "Dadra and Nagar
Haveli",
    27: "Maharashtra", 28: "Andhra Pradesh", 29: "Karnataka", 30: "Goa",
    31: "Lakshadweep", 32: "Kerala", 33: "Tamil Nadu", 34: "Puducherry",
    35: "Andaman and Nicobar", 36: "Telangana"
}

df3['state_name'] = df3['stateid'].map(state_id_to_name)
df3['state_clean'] = df3['state_name'].str.replace(" ", "").str.title()

# --- Step 3: List of domain index columns ---
domain_indices = [
    'physical_health_index',
    'mental_well_being_index',
    'functional_ability_index',
    'economic_security_index',
    'healthcare_satisfaction_vars_index',
]

# --- Step 4: Loop through each domain and plot maps ---
for domain_col in domain_indices:
    # 4a. Group by state and get median (or mean, if preferred)
    domain_by_state = df3.groupby('state_clean')[domain_col].mean().reset_index()

    # 4b. Clip values to [0, 1]
    domain_by_state[domain_col] = domain_by_state[domain_col].clip(0, 1)

    # 4c. Merge with GeoDataFrame
    merged = gdf.merge(domain_by_state, on='state_clean', how='left')

    # 4d. Plot the map
    fig, ax = plt.subplots(figsize=(13, 13))
    merged.plot(
        column=domain_col,
        cmap='YlGn', # Yellow-Green scheme
        linewidth=0.5,
        ax=ax,
        edgecolor='black',

```

```

        legend=True,
        missing_kwds={"color": "lightgrey", "label": "No data"}
    )

# 4e. Add state labels
for idx, row in merged.iterrows():
    point = row['geometry'].representative_point()
    val = f'{row[domain_col]:.2f}' if pd.notnull(row[domain_col]) else "N/A"
    label = f'{row[state_clean]}\n{val}'
    ax.text(point.x, point.y, label, fontsize=6.5, ha='center', va='center', color='black')

# 4f. Title and aesthetics
ax.set_title(f'{domain_col.replace('_', ' ').title()} (Statewise)', fontsize=16)
ax.axis('off')
plt.tight_layout()
plt.show()

```

Step 99

```

import geopandas as gpd
import pandas as pd
import matplotlib.pyplot as plt

# --- Step 1: Load and clean shapefile ---
gdf =
gpd.read_file(r"C:\Users\ASUS\Downloads\gadm41_IND_1.json\gadm41_IND_1.json")
gdf[state_clean] = gdf[NAME_1].str.replace(" ", "").str.title()

# Fix known mismatches
gdf[state_clean] = gdf[state_clean].replace({
    'NcTofdelhi': 'Delhi',
    'Nctofdelhi': 'Delhi',
    'Chhatisgarh': 'Chhattisgarh'
})

# Dissolve duplicates
gdf = gdf.dissolve(by='state_clean', as_index=False)

# --- Step 2: Map state name info to df3 ---
state_id_to_name = {
    1: "Jammu and Kashmir", 2: "Himachal Pradesh", 3: "Punjab", 4: "Chandigarh",
    5: "Uttarakhand", 6: "Haryana", 7: "Delhi", 8: "Rajasthan", 9: "Uttar Pradesh",
    10: "Bihar", 11: "Sikkim", 12: "Arunachal Pradesh", 13: "Nagaland",
    14: "Manipur", 15: "Mizoram", 16: "Tripura", 17: "Meghalaya", 18: "Assam",
    19: "West Bengal", 20: "Jharkhand", 21: "Odisha", 22: "Chhattisgarh",

```

```

23: "Madhya Pradesh", 24: "Gujarat", 25: "Daman and Diu", 26: "Dadra and Nagar
Haveli",
27: "Maharashtra", 28: "Andhra Pradesh", 29: "Karnataka", 30: "Goa",
31: "Lakshadweep", 32: "Kerala", 33: "Tamil Nadu", 34: "Puducherry",
35: "Andaman and Nicobar", 36: "Telangana"
}
df3['state_name'] = df3['stateid'].map(state_id_to_name)
df3['state_clean'] = df3['state_name'].str.replace(" ", "").str.title()

# --- Step 3: Define domains ---
domain_indices = [
    'physical_health_index',
    'mental_well_being_index',
    'functional_ability_index',
    'economic_security_index',
    'healthcare_satisfaction_vars_index',
]
# Drop old domain columns if they exist in gdf (to avoid duplicates)
gdf_clean = gdf.drop(columns=[col for col in gdf.columns if col in domain_indices],
errors='ignore')
merged = gdf_clean.merge(domain_by_state, on='state_clean', how='left')

# --- Step 4: Define age groups ---
age_groups = {
    '60-74': df3[(df3['Age'] >= 60) & (df3['Age'] <= 74)],
    '75-84': df3[(df3['Age'] >= 75) & (df3['Age'] <= 84)],
    '85+': df3[df3['Age'] >= 85]
}

for age_label, age_df in age_groups.items():
    for domain_col in domain_indices:
        domain_by_state = age_df.groupby('state_clean')[domain_col].mean().reset_index()
        domain_by_state[domain_col] = domain_by_state[domain_col].clip(0, 1)

        merged = gdf.merge(domain_by_state, on='state_clean', how='left')

# ✅ Safely convert to numeric
try:
    merged[domain_col] = pd.to_numeric(merged[domain_col], errors='coerce')
except Exception as e:
    print(f'Skipping {domain_col} for age group {age_label}: {e}')
    continue

fig, ax = plt.subplots(figsize=(13, 13))
merged.plot(

```

```

        column=domain_col,
        cmap='YlGn',
        linewidth=0.5,
        ax=ax,
        edgecolor='black',
        legend=True,
        missing_kwds={"color": "lightgrey", "label": "No data"}
    )

    for idx, row in merged.iterrows():
        point = row['geometry'].representative_point()
        val = f'{row[domain_col]:.2f}' if pd.notnull(row[domain_col]) else "N/A"
        label = f'{row[state_clean]}\n{val}'
        ax.text(point.x, point.y, label, fontsize=6.5, ha='center', va='center', color='black')

    ax.set_title(f'{domain_col.replace('_', ' ').title()} – Age {age_label}', fontsize=16)
    ax.axis('off')
    plt.tight_layout()
    plt.show()

```

Step 100

```
df3.head(5)
```

Step 101

```
print(df3['current_marital_status'].value_counts(dropna=False))
```

Step 102

```
import numpy as np
```

```
# Step 1: Start with Female if menopause year is known
```

```
df3['Gender'] = df3['gender'].apply(lambda x: 'Female' if pd.notna(x) and str(x).isdigit() else
np.nan)
```

```
# Step 2: Set Male if occupation is known and gender not already assigned
```

```
df3.loc[(df3['occupation'].notna()) & (df3['Gender'].isna()), 'Gender'] = 'Male'
```

```
# Step 3: For remaining unassigned, randomly assign 25% as Male, rest as Female
```

```
remaining_indices = df3[df3['Gender'].isna()].index
```

```
n_males_to_add = int(0.25 * len(remaining_indices))
```

```
np.random.seed(42)
```

```
male_indices = np.random.choice(remaining_indices, size=n_males_to_add, replace=False)
```

```
# Assign Male to selected randoms
```

```
df3.loc[male_indices, 'Gender'] = 'Male'
```

```
# Assign Female to the rest
df3.loc[df3['Gender'].isna(), 'Gender'] = 'Female'
```

```
# Final value counts
print(df3['Gender'].value_counts(dropna=False))
```

Step 103

```
for col in df3.columns:
    print(col)
```

Step 104

```
# List of columns to remove
columns_to_remove = [
    'gender', 'current_employment_status', 'Sensory_deprivation', 'having_friends',
    'ever_diagnosed_hypertension', 'ever_diagnosed_diabetes', 'ever_diagnosed_cancer',
    'ever_diagnosed_with_cld', 'ever_diagnosed_chronic_heart_disease',
    'ever_diagnosed_chronic_joint_condition', 'ever_diagnosed_stroke',
    'ever_diagnosed_neurological_psychiatry_problem', 'ever_diagnosed_high_cholesterol',
    'ever_heart_attack', 'urogenital_condition', 'Chornic_urogenital_conditions',
    'ear_problems', 'oral_problems', 'ease_chewing_food', 'vision_problems',
    'major_injury_2years', 'fall_in_2years', 'fracture', 'natural_disaster',
    'man_made_disaster', 'chest_pain', 'immunization', 'women_health_problems',
    'sleep_problems', 'pain_problems', 'pass_urine_in_activities', 'troublesome_problems',
    'any_form_of_physical_mental_impairment', 'overall_score_orientation', 'hypertension',
    'which_health_professional_diagnosed_htn_first',
    'first_health_professional_to_diagnose_diabetes',
    'first_heatlhcare_professional_to_diagnose_cancer', 'multiple_cancer',
    'first_health_professional_to_diagnose_cld',
    'first_health_professional_to_diagnoise_heart_disease',
    'first_health_professional_to_diagnose_stroke', 'memory_impairment',
    'frist_professional_to_diagnose_ra',
    'first_health_professional_neurology_psychiatry_problem',
    'first_health_professional_diagnose_htn', 'thyroid_disorder', 'gastrointestinal_problems',
    'skin_disease', 'other_chornic_conditions', 'visionproblem_one_two', 'sleep',
    'Pain_in_30days', 'thyroid', 'git_problems', 'skin_problmes', 'other_chronic_illness',
    'hearing_problems', 'doctor_visit_women_health', 'physical_mental_impairment',
    'mental_impairment', 'depression', 'Memory4', 'executive_function', 'attention',
    'Learning', 'Praxis_calculation', 'behaviour_issues', 'frail01', 'ADL', 'IADL',
    'hc220', 'ADL_impaired', 'IADL_impaired', 'frail01_binary', 'Sensory_deprivation_binary',
    'days_bed_ridden_binary', 'mpce_poor', 'immunization_binary', 'Sex', 'Sex_0', 'Sex_12',
    'Sex_11', 'Sex_9'
]
```

```
# Create df4 by dropping specified columns from df3
df4 = df3.drop(columns=columns_to_remove, errors='ignore')
```

Step 105

```
num_removed = df3.shape[1] - df4.shape[1]
print(f'Number of variables removed: {num_removed}')
```

Step 106

```
df4.info()
```

Step 107

```
for col in df4.columns:
    print(col)
```

Step 108

```
columns_to_remove = [
    'QoC_waiting_time',
    'QoC_respect',
    'QoC_clear_explanation',
    'QoC_priv_talk_with_doctor',
    'QoC_provider_of_your_choice',
    'QoC_cleanliness_health_facility',
    'satisfaction_with_treatment_recieved',
    'treated_poorly_in_stores_hotels_binary',
    'people_act_they_are_smart_binary',
    'people_act_afraid_binary',
    'threatened_harassed_binary',
    'life_close_to_ideal_reframed',
    'life_conditions_excellent_reframed',
    'satisfied_with_life_reframed',
    'got_important_things_in_life_reframed',
    'live_life_again_reframed',
    'treated_with_respect_reframed',
    'feeling_inner_peace_mapped',
    'spiritually_touched_mapped',
    'thankful_for_things_in_life_mapped',
    'selflessly_caring_for_others_mapped',
    'total_value_of_financial_help_cat',
    'treated_with_respect_cat',
    'satisfaction_with_treatment_recieved_cat',
    'QoC_cleanliness_health_facility_cat',
    'QoC_clear_explanation_cat',
```

```

'QoC_waiting_time_cat',
'QoC_respect_cat',
'QoC_priv_talk_with_doctor_cat',
'QoC_provider_of_your_choice_cat',
'satisfied_with_treatment_hospitalization_cat',
'consult_healthcare_providers_clean',
'consult_with_allopathy',
'mode_transport_access',
'time_reach_opd_bin',
'dist_from_hospital_bin',
'travel_time_hospital_bin',
'numer_opd_visits_1year_bin',
'number_of_hospital_admission_lastyear_bin',
'current_health_bin', 'immediate_recall', 'current_health', 'time_spent_on_tv',
'likely_to_work_samejob_next5yrs',
'felt_ill_treated',
'physical_abuse',
'verbal_abuse', 'economic_exploitation', 'emotional_exploitation', 'neglect',
'feeling_safe', 'feeling_lonely', 'violent_crime_victim', 'mpce_quintile'
]

```

```
df4 = df4.drop(columns=columns_to_remove, errors='ignore')
```

Step 109

```

df4.drop(columns=[
    'exercise_moderate_atleast',
    'time_spent_on_exercise_mod_atleast',
    'yoga_meditation',
    'time_spent_on_yoga',
    'religious_gatherings',
    'preferred_religion', 'ever_reduce_or_skip_meals',
    'eat_enough_food_of_choice', 'hungry_but_didnt_eat', 'not_eat_for_whole_day', 'days_bed_ridden',
    'fluctuation_in_price_expected'
], inplace=True, errors='ignore')

```

Step 110

```

df4.drop(columns=[
    'feeling_safe_when_walking_down_on_street_locality',
    'life_close_to_ideal',
    'life_conditions_excellent',
    'satisfied_with_life',
    'got_important_things_in_life',
    'live_life_again',

```

```

    'feeling_inner_peace',
    'spiritually_touched',
    'thankful_for_things_in_life',
    'selflessly_caring_for_others'
], inplace=True, errors='ignore')

```

Step 111

```

cols_to_remove = [
    'visit_subcentre', 'visit_phc', 'visit_chc', 'visit_dhc', 'visit_tertiary',
    'visit_ayush', 'visit_pvthospital', 'visit_pvt_clinic', 'visit_ngo', 'visit_pvt_ayush',
    'visit_camp', 'visit_mobile_unit', 'visit_pharmacy', 'visit_homevist', 'visit_other', 'visit_none'
]

df4 = df4.drop(columns=cols_to_remove, errors='ignore')

```

Step 112

```

# Step 1: Drop specified columns
cols_to_remove = [
    'caretaker', 'op_care', 'person_accompanying_during_OPDvisit',
    'people_accompanying_healthcare_activities', 'relative_accompanying_opd'
]

df4 = df4.drop(columns=cols_to_remove, errors='ignore')

```

Step 113

```

# Columns to remove
cols_to_remove = [
    'treated_poorly_in_stores_hotels',
    'people_act_they_are_smart',
    'people_act_afraid',
    'threatened_harassed',
    'feeling_safe_when_walking_down_on_street_locality'
]

# Drop columns
df4 = df4.drop(columns=cols_to_remove, errors='ignore')

```

Step 114

```
# List of columns to remove
cols_to_remove = [
    'aware_annapurna_scheme', 'aware_any_other_state_scheme', 'aware_other_scheme',
    'problem_as', 'problem_ss', 'problem_others',
    'mode_of_transport_to_hospital', 'travel_time_hospital',
    'mode_transport_healthcare', 'mode_of_transport', 'time_reach_opd'
]

# Drop them from df4
df4 = df4.drop(columns=cols_to_remove, errors='ignore')
```

Step 115

```
columns_to_remove = [
    # ❌ Health Service Complaints (unless part of satisfaction index)
    'poor_service_from_paraclinical_staff',
    'treated_with_respect', # Optional: keep if used in satisfaction index
    'treated_poorly_in_stores_hotels',
    'people_act_they_are_smart',
    'people_act_afraid',
    'threatened_harassed',

    # ❌ Visit-Specific Operational Details
    'opd_healthcare_provider_visited',
    'facility_visited_opcare',
    'mode_of_transport_hospital',
    'mode_of_transport',
    'expenditure_of_accompanying_relative_opd',
    'relative_accompanying_opd',
    'time_reach_opd', 'reimbursement_received_none'
]

df4.drop(columns=columns_to_remove, errors='ignore', inplace=True)
```

Step 116

```
df4.info()
```

Step 117

```
for col in df4.columns:
    print(col)
```

Step 118

```
cols = [
    'reimbursement_recieved_govt_employee',
    'reimbursement_recieved_pvt_employee',
    'reimbursement_recieved_medical_insurance_company',
    'reimbursement_recieved_other_agency',
]
```

```
for col in cols:
    if col in df4.columns:
        print(f"\nUnique values in '{col}':")
        print(df4[col].unique())
    else:
        print(f"\nColumn '{col}' not found in df4.")
```

Step 119

Define the reimbursement source columns

```
reimb_cols = [
    'reimbursement_recieved_govt_employee',
    'reimbursement_recieved_pvt_employee',
    'reimbursement_recieved_medical_insurance_company',
    'reimbursement_recieved_other_agency',
]
```

Fill NaN with 0 (safe since 0 = No)

```
df4[reimb_cols] = df4[reimb_cols].fillna(0)
```

Count how many sources are marked as 1 for each row

```
df4['reimb_count'] = df4[reimb_cols].sum(axis=1)
```

Categorize based on count

```
def categorize_reimbursement(count):
```

```
    if count == 0:
```

```
        return 'No Reimbursement'
```

```
    elif count == 1:
```

```
        return 'Single Source'
```

```
    else:
```

```
        return 'Multiple Sources'
```

```
df4['reimbursement_status'] = df4['reimb_count'].apply(categorize_reimbursement)
```

Drop the helper column

```
df4.drop('reimb_count', axis=1, inplace=True)
```

Step 120

```
df4.drop([
    'reimbursement_recieved_govt_employee',
    'reimbursement_recieved_pvt_employee',
    'reimbursement_recieved_medical_insurance_company',
    'reimbursement_recieved_other_agency',
], axis=1, inplace=True)
```

Step 121

```
for col in df4.columns:
    print(col)
```

Step 122

```
# List of columns to drop
cols_to_drop = [
    'problem_nps',
    'problem_wps',
    'problem_as',
    'problem_ss',
    'protblem_others' # Check spelling in your dataset
]

# Drop columns safely (ignore if column not found)
df4 = df4.drop(columns=[col for col in cols_to_drop if col in df4.columns])
```

Step 123

```
cols_to_drop = [
    'poor_service_from_paraclinical_staff',
    'treated_with_respect',
    'treated_poorly_in_stores_hotels',
    'people_act_they_are_smart',
    'people_act_afraid',
    'threatened_harassed'
]

df4.drop(columns=cols_to_drop, inplace=True, errors='ignore')
```

Step 124

```
cols_to_drop = [
    'Reasons_for_not_covered_byprovident_fund',
    'reasons_not_covered_health_insurance',
    'reason_not_covered_medical_reimburshment',
    'reaons_not_covered_with_insurance_work',
    'reasons_for_covered_workers_employed_insurnce','frustrated_with_life'
]
```

```
df4.drop(columns=cols_to_drop, inplace=True, errors='ignore')
```

Step 125

```
cols_to_drop = [
    'HI_covers_surgery',
    'HI_covers_tests',
    'HI_covers_docotors_visit',
    'HI_covers_meds',
    'HI_covers_dental_care',
    'HI_covers_in_home_care',
    'HI_covers_hospital_charges',
    'HI_covers_others',
    'reason_not_having_HI',
    'covered_health_insurance_exptLIC',
    'types_health_insurance_covered', 'stateid'
]
```

```
df4.drop(columns=cols_to_drop, inplace=True, errors='ignore')
```

Step 126

Define the list again

```
cols_to_check = [
    'treatment_source_income',
    'source_home_income',
    'treatment_source_saving',
    'treatatment_source_loans',
    'treatatment_source_friendsrelative',
    'treatatment_source_sellingproperty',
    'treatatment_source_insurance',
    'treatatment_source_reimburshment',
    'treatatment_source_other'
]
```

Calculate total treatment support amount

```
df4['total_treatment_support_amount'] = df4[cols_to_check].sum(axis=1, skipna=True)
```

```

# Count non-zero and non-null values
df4['treatment_source_count'] = df4[cols_to_check].gt(0).sum(axis=1)

# Categorize into No source, One source, Multiple sources
def group_sources(n):
    if n == 0:
        return 'No source'
    elif n == 1:
        return 'One source'
    else:
        return 'Multiple sources'

df4['treatment_source_category'] = df4['treatment_source_count'].apply(group_sources)

```

Step 127

```
df4.drop(columns=cols_to_check + ['treatment_source_count'], inplace=True)
```

Step 128

```

df4.drop(columns=[
    'last_HI_premium_paid',
    'treatment_discontinued_monetary_reasons',
    'living_alone'
], inplace=True)

```

Step 129

```

benefit_cols = [
    'benefited_anapurnascheme',
    'benefited_wps',
    'benefited_NPS',
    'benefited_state_scheme',
    'benefited_other_scheme'
]

# Step 1: Create a new column that counts how many benefit sources are received (i.e., values == 1.0)
df4['num_benefits_received'] = df4[benefit_cols].apply(lambda row:
row.fillna(0).astype(int).sum(), axis=1)

# Step 2: Map the count into a categorical label
def categorize_benefits(n):
    if n == 0:

```

```

        return 'No benefits'
    elif n == 1:
        return 'One benefit'
    else:
        return 'Multiple benefits'

df4['benefit_status_various_gov_scheme'] =
df4['num_benefits_received'].apply(categorize_benefits)

# Step 3: Drop the individual benefit columns if no longer needed
df4.drop(columns=benefit_cols + ['num_benefits_received'], inplace=True)

```

Step 130

```

# Set the threshold (80% nulls)
threshold = 0.8

# Calculate the fraction of missing values per column
null_fraction = df4.isnull().mean()

# Filter columns with ≥ 80% nulls
high_null_columns = null_fraction[null_fraction >= threshold]

# Display the results
print("Columns with ≥ 80% missing values:\n")
print(high_null_columns.sort_values(ascending=False))

```

Step 131

```

# List of columns to drop due to high missingness or low relevance
columns_to_drop = [
    'total_value_of_financial_help',
    'health_problem_limitin_work',
    'expected_work_duration',
    'lump_sum_amount_recieved',
    'satisfied_with_treatment_hospitalization',
    'illinnes_last_hospitalization',
    'distance_of_hospital',
    'dist_from_hospital',
    'reason_for_last_hospitalization',
    'health_condition_after_hospitalization',
    'number_night_spent_lastyr',
    'facility_of_last_hospitalization',
    'was_working_at_time_hospitalization',
    'financial_aid_for_daily_expense',

```

```

    'financial_aid_for_treatment',
    'alcohol',
    'smoking',
    'max_amount_insurance_coverage'
]

# Drop columns from df4
df4.drop(columns=columns_to_drop, inplace=True)

```

Step 132

```
df4.info()
```

Step 133

```

# List of columns to drop nulls from
columns_to_check = [
    "covered_with_workrelated_ps", "covered_provident_fund",
    "covered_with_injury_insurance_work", "visit_to_healthcare_facility", "reason_op_care",
    "healthcare_provider_op_care", "main_purpose_opd_care",
    "purpose_opd_visit", "raise_in_price", "treatment_source_category",
    "reason_for_last_visit_hospital",
    "consult_healthcare_providers", "numer_opd_visits_1year", "number_opd_visits",
    "outcome_opd_visit",
    'recommended_by_healthcare_provider_to_visit_hospital', 'average_work_hours',
    'Not_employed', 'use_benefits', 'reason_for_no_health_insurance'
]

# Drop columns from df4
df4.drop(columns=columns_to_drop, inplace=True, errors='ignore')

```

Step 134

```

# Columns to remove
columns_to_drop = [
    'old_age',
    'healthcare_consultation',
    'reason_for_last_visit_hospital',
    'reason_op_care',
    'recommended_by_healthcare_provider_to_visit_hospital',
    'consumption_of_med_without_consultation',
    'visit_to_healthcare_facility'
]

# Drop the columns

```

```
df4.drop(columns=columns_to_drop, inplace=True)
```

Step 135

```
df4.loc[:, 'anticipated_vs_actual_health_expense_gap'] = (
    df4['money_to_be_spend_in_next_1year'] - df4['money_spend_in_past_1year']
)
def categorize_expense_gap(row):
    anticipated = row['money_to_be_spend_in_next_1year']
    actual = row['money_spend_in_past_1year']

    if pd.isnull(anticipated) or pd.isnull(actual) or actual == 0:
        return 'Unknown'

    gap = anticipated - actual
    percentage_diff = gap / actual

    if abs(percentage_diff) <= 0.10:
        return 'Accurate'
    elif gap < 0:
        return 'Underestimated'
    else:
        return 'Overestimated'

df4.loc[:, 'expense_gap_category'] = df4.apply(categorize_expense_gap, axis=1)
```

Step 136

```
df4.drop(columns=['money_to_be_spend_in_next_1year', 'money_spend_in_past_1year'],
inplace=True)
```

Step 137

```
def classify_stress(row):
    if row['raise_in_price'] == 1:
        if row['expense_gap_category'] == 'Underestimated':
            return 'High Stress'
        elif row['expense_gap_category'] == 'Accurate':
            return 'Moderate Stress'
        else: # Overestimated
            return 'Low Stress'
    else:
        if row['expense_gap_category'] == 'Underestimated':
            return 'Moderate Stress'
```

```

elif row['expense_gap_category'] == 'Accurate':
    return 'Low Stress'
else: # Overestimated
    return 'Very Low Stress'

```

```
df4['healthcare_financial_stress_level'] = df4.apply(classify_stress, axis=1)
```

Step 138

```

print("Raise in Price:")
print(df4['raise_in_price'].value_counts(dropna=False))

print("\nExpense Gap Category:")
print(df4['expense_gap_category'].value_counts(dropna=False))

print("\nHealthcare Financial Stress Level:")
print(df4['healthcare_financial_stress_level'].value_counts(dropna=False))

```

Step 139

```

# Replace 'df' with the name of your DataFrame if it's different
columns_to_inspect = [
    "main_purpose_opd_care",
    "visit_to_healthcare_facility",
    "treatment_source_category",
    "total_treatment_support_amount",
    "reason_op_care",
    "purpose_opd_visit",
    "level_healthcare_visited",
    "healthcare_provider_op_care"
]

# Print unique values for each column (excluding NaNs)
for col in columns_to_inspect:
    if col in df.columns:
        print(f"Unique values in '{col}':")
        print(df[col].dropna().unique())
        print("\n" + "-"*50 + "\n")

```

Step 140

```

# Define mapping
healthcare_level_map = {
    1: 'Primary care',
    2: 'Primary care',
    3: 'Primary care',
    4: 'Secondary care',
    5: 'Secondary care',
    6: 'Secondary care',
    7: 'Private care',
    8: 'Private care',
    9: 'Private care',
    10: 'Private care',
    11: 'Other care',
    12: 'Other care',
    13: 'Other care',
    14: 'Other care',
    15: 'Other care'
}

# Create new grouped column
df4['healthcare_level_grouped'] = df4['level_healthcare_visited'].map(healthcare_level_map)

# Optional: check frequency counts
print(df4['healthcare_level_grouped'].value_counts(dropna=False))

```

Step 141

```

cols_to_drop = [
    'level_healthcare_visited',
    'healthcare_provider_op_care',
    'reason_op_care',
    'purpose_opd_visit',
    'visit_to_healthcare_facility', 'outcome_opd_visit', 'raise_in_price', 'spending_category',
    'money_spent_lastyear'
]

df4.drop(columns=cols_to_drop, inplace=True, errors='ignore')

```

Step 142

```

# Drop the two columns
df4.drop(columns=["main_reason_for_not_seeking_doctor", "money_spent_on_medicine"],
inplace=True)

```

Step 143

```
df4.info()
```

Step 144

```
spss_path = r"C:\Users\ASUS\OneDrive\Desktop\PHD Business  
admin\Analysis\Mastersheet\dataset_for_analysis.sav"  
pyreadstat.write_sav(df4, spss_path)
```

Step 145

```
excel_path = r"C:\Users\ASUS\OneDrive\Desktop\PHD Business  
admin\Analysis\Mastersheet\dataset_for_analysis.xlsx"  
df4.to_excel(excel_path, index=False)
```

Step 146

```
import pandas as pd  
  
# Use raw string (r"") to handle backslashes or double backslashes  
file_path = r"C:\Users\ASUS\OneDrive\Desktop\PHD Business  
admin\Analysis\Mastersheet\dataset_for_analysis.xlsx"  
  
# Load the Excel file  
df = pd.read_excel(file_path)
```

Step 147

```
df.head()
```

Step 148

```
for col in df.columns:  
    print(col)
```

Step 149

```
df['economic_score'] = (  
    df['occupation'].notna().astype(int) +          # 1 = working  
    (df['average_work_hours'] > 0).astype(int) +    # 1 = working hours  
    df['currently_looking_for_job'].fillna(0).astype(int) + # 1 = seeking job  
    df['economic_security_index'].fillna(0)          # Continuous score  
)  
  
# Normalize if needed or check distribution first  
threshold = 3 # Can tune this based on histogram
```

```
df['Economic_Participation'] = (df['economic_score'] >= threshold).astype(int)
```

Step 150

```
import matplotlib.pyplot as plt
```

```
df['economic_score'].hist(bins=20)
plt.title("Distribution of Economic Score")
plt.xlabel("Score")
plt.ylabel("Count")
plt.grid(True)
plt.show()
```

Step 151

```
df['Economic_Participation'].hist(bins=20)
plt.title("Distribution of Economic participation")
plt.xlabel("Score")
plt.ylabel("Count")
plt.grid(True)
plt.show()
```

Step 152

```
# Unique values in economic_score (sorted for clarity)
print("Unique values in 'economic_score':")
print(sorted(df['economic_score'].dropna().unique()))

# Frequency counts
print("\nValue counts in 'economic_score':")
print(df['economic_score'].value_counts().sort_index())

# Unique values in Economic_Participation
print("\nUnique values in 'Economic_Participation':")
print(df['Economic_Participation'].unique())

# Frequency counts
print("\nValue counts in 'Economic_Participation':")
print(df['Economic_Participation'].value_counts())
```

Step 153

```
df['economic_score'].hist(bins=40)
```

Step 154

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix, roc_auc_score,
roc_curve
import matplotlib.pyplot as plt
import seaborn as sns
```

Step 155

```
df['Gender'] = df['Gender'].map({'Male': 1, 'Female': 0})
```

Step 156

```
stress_order = ['Very Low Stress', 'Low Stress', 'Moderate Stress', 'High Stress']
df['healthcare_financial_stress_level'] = pd.Categorical(
    df['healthcare_financial_stress_level'],
    categories=stress_order,
    ordered=True
)
df['healthcare_financial_stress_cat'] = df['healthcare_financial_stress_level'].cat.codes
```

Step 157

```
# Show unique values and their corresponding original categories
print("Unique codes:", df['healthcare_financial_stress_cat'].unique())
print("\nCode-to-label mapping:")
print(dict(enumerate(df['healthcare_financial_stress_level'].astype('category').cat.categories))
)
```

Step 158

```
ml_features_selected = [
    #Demographic Factors
    'Age', 'Gender', 'education', 'residence', 'religion', 'caste_category', 'current_marital_status',
    'living_arrangements',

    #Economic Status / Support
```

```

'mpce_amount_in_rs',
'use_benefits',
'covered_provident_fund',
'economic_security_index',

# 🏠 Health Access & Status
'physical_health_index',
'mental_well_being_index',
'functional_ability_index',
'HAI_score',
'healthcare_satisfaction_vars_index',

# 🗣️ Health Service Utilization
'number_of_hospital_admission_lastyear',
'numer_opd_visits_1year',
'cost_opd_visit',

# 📖 Knowledge & Scheme Awareness
'aware_of_govt_oldagebenefits',
'aware_welfareparents_senior_citizen',

# ⚠️ Financial Risk & Stress
'anticipated_vs_actual_health_expense_gap',
'healthcare_financial_stress_cat'
]

```

Step 159

```

# Convert Age to categorical bins
df['Age_group'] = pd.cut(df['Age'], bins=[59, 74, 84, df['Age'].max()], labels=[0, 1, 2])

# Convert other variables to categorical codes
categorical_cols = ['Gender', 'education', 'residence', 'religion',
                    'caste_category', 'current_marital_status', 'living_arrangements']

for col in categorical_cols:
    df[col + '_cat'] = df[col].astype('category').cat.codes

```

Step 160

```

cols = ['Age_group', 'Gender', 'education', 'residence', 'religion', 'caste_category',
'current_marital_status', 'living_arrangements']

# Display unique values for each column

```

```

for col in cols:
    print(f'--- {col} ---')
    print(df[col].dropna().unique())
    print()

```

Step 161

```
df.info()
```

Step 162

```

columns_to_check = [
    'mpce_amount_in_rs', 'occupation', 'average_work_hours', 'currently_looking_for_job',
    'Not_employed', 'use_benefits', 'aware_of_govt_oldagebenefits',
    'aware_welfareparents_senior_citizen', 'covered_with_workrelated_ps',
    'covered_provident_fund', 'covered_with_some_health_insurance',
    'covered_with_injury_insurance_work', 'covered_medical_reimbursment',
    'covered_with_workers_employment_insurance',
    'number_of_hospital_admission_lastyear',
    'reason_for_no_health_insurance'
]

```

```

for col in columns_to_check:
    print(f'--- {col} ---')
    print(df[col].dropna().unique())
    print()

```

Step 163

```

import pandas as pd
import numpy as np

```

```

# 1. Categorize `mpce_amount_in_rs` into income levels (Low, Medium, High)
df['mpce_category'] = pd.qcut(df['mpce_amount_in_rs'], q=3, labels=['Low', 'Medium', 'High'])

```

```

# 2. Categorize `average_work_hours` into workload categories
def categorize_work_hours(hours):
    if hours == 0:
        return 'Unemployed'
    elif hours <= 20:
        return 'Part-time'
    elif hours <= 40:
        return 'Full-time'
    elif hours <= 60:

```

```

        return 'Overtime'
    else:
        return 'Extreme'

df['work_hours_category'] = df['average_work_hours'].apply(categorize_work_hours)

# 3. Categorize `number_of_hospital_admission_lastyear` into bins
def categorize_hospital_visits(n):
    if n == 0:
        return 'None'
    elif n == 1:
        return 'Rare'
    elif 2 <= n <= 4:
        return 'Occasional'
    elif 5 <= n <= 10:
        return 'Frequent'
    else:
        return 'Very Frequent'

df['hospital_admission_category'] =
df['number_of_hospital_admission_lastyear'].apply(categorize_hospital_visits)

# View the new categories
print(df[['mpce_category', 'work_hours_category', 'hospital_admission_category']].head())

```

Step 164

```

# Marking the following columns as categorical and renaming with '_categorical' suffix

categorical_columns = [
    'occupation',
    'currently_looking_for_job',
    'Not_employed',
    'use_benefits',
    'aware_of_govt_oldagebenefits',
    'aware_welfareparents_senior_citizen',
    'covered_with_workrelated_ps',
    'covered_provident_fund',
    'covered_with_some_health_insurance',
    'covered_with_injury_insurance_work',
    'covered_medical_reimbursment',
    'covered_with_workers_employment_insurance',
    'reason_for_no_health_insurance'
]

```

```
# Convert to categorical and rename
for col in categorical_columns:
    new_col = f'{col}_categorical'
    df[new_col] = pd.Categorical(df[col])
```

Step 165

```
df.info()
```

Step 166

```
columns_to_check = [
    'numer_opd_visits_1year',
    'main_purpose_opd_care',
    'cost_opd_visit',
    'consult_healthcare_providers',
    'number_opd_visits',
    'physical_health_index',
    'mental_well_being_index',
    'functional_ability_index',
    'economic_security_index',
    'healthcare_satisfaction_vars_index',
    'HAI_score',
    'HAI_category',
    'state_name',
    'state_clean',
    'Gender',
    'reimbursment_status',
    'total_treatment_support_amount',
    'treatment_source_category',
    'benefit_status_various_gov_scheme',
    'anticipated_vs_actual_health_expense_gap',
    'expense_gap_category',
    'healthcare_financial_stress_level',
    'healthcare_level_grouped',
    'economic_score'
]

# Print unique values for each column
for col in columns_to_check:
    print(f'--- Unique values in column: {col} ---')
    print(df[col].unique())
    print("\n")
```

Step 167

```

# Define bins manually to avoid duplicate edge errors
bins = [-1, 0, 20000, float('inf')] # -1 ensures 0 is included in the first bin
labels = [0, 1, 2] # 0 = None, 1 = Low/Moderate, 2 = High

# Apply manual binning
df['total_treatment_support_category'] = pd.cut(
    df['total_treatment_support_amount'],
    bins=bins,
    labels=labels
)

# Check output
print(df['total_treatment_support_category'].value_counts(dropna=False))

```

Step 168

```

# Check if column exists
if 'HAI_category' in df.columns:
    print(df['HAI_category'].unique())
else:
    print("Column 'HAI_category' not found in the DataFrame.")
# Define mapping
hai_mapping = {'Healthy': 0, 'Moderate': 1, 'Severe': 2}

# Apply mapping to the column
df['HAI_category'] = df['HAI_category'].map(hai_mapping)

```

Step 169

```

# Replace -1 with a valid value that will map to 'Rare' during binning
df['numer_opd_visits_1year_cleaned'] = df['numer_opd_visits_1year'].replace(-1, 2)

# Define bins and labels
bins = [-2, 0, 3, 6, 12, 24, float('inf')] # -2 ensures that 0 is properly included
labels = ['None', 'Rare', 'Occasional', 'Frequent', 'Very Frequent', 'Chronic Utilizer']

# Create categorical column
df['numer_opd_visits_1year_categorical'] = pd.cut(df['numer_opd_visits_1year_cleaned'],
    bins=bins, labels=labels)

# Assign numeric codes to categories
label_to_code = {
    'None': 0,
    'Rare': 1,
    'Occasional': 2,

```

```

    'Frequent': 3,
    'Very Frequent': 4,
    'Chronic Utilizer': 5
}

df['numer_opd_visits_1year_cat_code'] =
df['numer_opd_visits_1year_categorical'].map(label_to_code)

# Display value counts
print(df[['numer_opd_visits_1year', 'numer_opd_visits_1year_categorical',
'nnumer_opd_visits_1year_cat_code']].head())

```

Step 170

```

# Create cost category using quantile-based binning
df['cost_opd_visit_categorical'] = pd.qcut(df['cost_opd_visit'], q=3, labels=['0','1','2'])
#0 Lowcost, '1 Medium Cost', '2 High Cost'
# Optional: Display value counts
print(df['cost_opd_visit_categorical'].value_counts(dropna=False))
print(df['numer_opd_visits_1year_cat_code'].value_counts(dropna=False))

```

Step 171

```

# Define bin edges and labels
bins = [-0.01, 0.4, 0.7, 1.0]
labels = ['Low', 'Moderate', 'High']

# Columns to convert
index_columns = [
    'physical_health_index',
    'mental_well_being_index',
    'functional_ability_index',
    'economic_security_index',
    'healthcare_satisfaction_vars_index'
]

# Create category columns and then map to numeric codes
label_to_code = {'Low': 0, 'Moderate': 1, 'High': 2}

for col in index_columns:
    category_col = f'{col}_category'
    code_col = f'{col}_cat_code'

    # Create category labels
    df[category_col] = pd.cut(df[col], bins=bins, labels=labels)

```

```
# Map labels to numeric codes
df[code_col] = df[category_col].map(label_to_code)
```

Step 172

```
# Step 1: Bin into 3 quantile-based cost groups
df['cost_opd_visit_category'] = pd.qcut(df['cost_opd_visit'], q=3, labels=['Low', 'Medium', 'High'])
```

```
# Step 2: Map to numeric values
cost_mapping = {'Low': 0, 'Medium': 1, 'High': 2}
df['cost_opd_visit_category'] = df['cost_opd_visit_category'].map(cost_mapping)
```

```
# Step 3: Confirm result
print(df['cost_opd_visit_category'].value_counts(dropna=False))
```

Step 173

```
df.drop(columns=['consult_healthcare_providers'], inplace=True)
```

Step 174

```
df.drop(columns=['number_opd_visits'], inplace=True)
```

Step 175

```
# Define mapping
reimb_map = {
    'No Reimbursement': 0,
    'Single Source': 1,
    'Multiple Sources': 2
}

# Apply mapping
df['reimbursment_status_category'] = df['reimbursment_status'].map(reimb_map)

# Optional: check result
print(df[['reimbursment_status', 'reimbursment_status_category']].head())
```

Step 176

```
# Define mapping
benefit_mapping = {
```

```

    'No benefits': 0,
    'One benefit': 1,
    'Multiple benefits': 2
}

# Create new numeric categorical column
df['benefit_status_various_gov_scheme_category'] =
df['benefit_status_various_gov_scheme'].map(benefit_mapping)

# Confirm result
print(df['benefit_status_various_gov_scheme_category'].value_counts(dropna=False))

```

Step 177

```

# Step 1: Drop NaNs temporarily to compute quantiles
gap_series = df['anticipated_vs_actual_health_expense_gap'].dropna()

# Step 2: Create quantile-based bins
try:
    df['anticipated_vs_actual_health_expense_gap_category'] = pd.qcut(
        df['anticipated_vs_actual_health_expense_gap'],
        q=3,
        labels=[0, 1, 2]
    )
except ValueError:
    # Fallback to manual bins if qcut fails due to repeated values
    bins = [-float('inf'), 0, 10000, float('inf')] # Example thresholds
    df['anticipated_vs_actual_health_expense_gap_category'] = pd.cut(
        df['anticipated_vs_actual_health_expense_gap'],
        bins=bins,
        labels=[0, 1, 2]
    )

# Step 3: Confirm result
print(df['anticipated_vs_actual_health_expense_gap_category'].value_counts(dropna=False))

```

Step 178

```

expense_gap_map = {
    'Accurate': 0,
    'Overestimated': 1,
    'Underestimated': 2,
    'Unknown': np.nan # or assign 3 if you want to track unknowns
}

```

```
df['expense_gap_category_numeric'] = df['expense_gap_category'].map(expense_gap_map)
```

Step 179

```
stress_level_map = {  
    'Very Low Stress': 0,  
    'Low Stress': 1,  
    'Moderate Stress': 2,  
    'High Stress': 3  
}
```

```
df['healthcare_financial_stress_level_numeric'] =  
df['healthcare_financial_stress_level'].map(stress_level_map)
```

Step 180

You can drop or keep 'Other care' as 3 or NaN based on preference

```
healthcare_level_map = {  
    'Primary care': 0,  
    'Secondary care': 1,  
    'Private care': 2,  
    'Other care': np.nan # optional: or assign 3  
}
```

```
df['healthcare_level_grouped_numeric'] =  
df['healthcare_level_grouped'].map(healthcare_level_map)
```

Step 181

```
df['economic_score_category'] = pd.qcut(  
    df['economic_score'],  
    q=3,  
    labels=[0, 1, 2]  
)
```

Step 182

Mapping text labels to numeric codes

```
benefit_map = {  
    'No benefits': 0,  
    'One benefit': 1,  
    'Multiple benefits': 2  
}
```

```
# Apply the mapping to a new column
df['benefit_status_various_gov_scheme'] =
df['benefit_status_various_gov_scheme'].map(benefit_map)
```

Step 183

```
# Define mapping of categorical columns and their code counterparts
code_pairs = [
    ('physical_health_index_category', 'physical_health_index_cat_code'),
    ('mental_well_being_index_category', 'mental_well_being_index_cat_code'),
    ('functional_ability_index_category', 'functional_ability_index_cat_code'),
    ('economic_security_index_category', 'economic_security_index_cat_code'),
    ('healthcare_satisfaction_vars_index_category',
'healthcare_satisfaction_vars_index_cat_code'),
    ('numer_opd_visits_1year_categorical', 'numer_opd_visits_1year_cat_code'),
    ('cost_opd_visit_categorical', 'cost_opd_visit_category'),
]

# Drop the categorical columns if their code version exists
for cat_col, code_col in code_pairs:
    if cat_col in df.columns and code_col in df.columns:
        df.drop(columns=[cat_col], inplace=True)
```

Step 184

```
df.info()
```

Step 185

```
# List of all relevant columns from your message
columns = [
    "Age_group", "Gender_cat", "education_cat", "residence_cat", "religion_cat",
    "caste_category_cat",
    "current_marital_status_cat", "living_arrangements_cat", "mpce_category",
    "work_hours_category",
    "hospital_admission_category", "occupation_categorical",
    "currently_looking_for_job_categorical",
    "Not_employed_categorical", "use_benefits_categorical",
    "aware_of_govt_oldagebenefits_categorical",
    "aware_welfareparents_senior_citizen_categorical",
    "covered_with_workrelated_ps_categorical",
    "covered_provident_fund_categorical",
    "covered_with_some_health_insurance_categorical",
    "covered_with_injury_insurance_work_categorical",
    "covered_medical_reimbursment_categorical",
```

```

    "covered_with_workers_employment_insurance_categorical",
    "reason_for_no_health_insurance_categorical",
    "total_treatment_support_category", "numer_opd_visits_1year_cleaned",
    "numer_opd_visits_1year_cat_code",
    "physical_health_index_cat_code", "mental_well_being_index_cat_code",
    "functional_ability_index_cat_code",
    "economic_security_index_cat_code", "healthcare_satisfaction_vars_index_cat_code",
    "cost_opd_visit_category",
    "reimbursement_status_category", "benefit_status_various_gov_scheme_category",
    "anticipated_vs_actual_health_expense_gap_category", "expense_gap_category_numeric",
    "healthcare_financial_stress_level_numeric", "healthcare_level_grouped_numeric",
    "economic_score_category"
]

```

```

# Dictionary to store results
unique_values = {}

```

```

# Loop through columns and get unique values
for col in columns:
    if col in df.columns:
        unique_values[col] = df[col].unique()

```

```

# Optional: display nicely
for col, values in unique_values.items():
    print(f'{col} ({len(values)} unique): {values}\n')

```

Step 186

```

# Convert 'mpce_category' to numerical codes
# Mapping: 'Low' = 0, 'Medium' = 1, 'High' = 2
df['mpce_category'] = df['mpce_category'].cat.codes

# Convert 'work_hours_category' to numerical codes
# Mapping: 'Extreme' = 0, 'Full-time' = 1, 'Overtime' = 2, 'Part-time' = 3, 'Unemployed' = 4
df['work_hours_category'] = df['work_hours_category'].astype('category').cat.codes

# Convert 'hospital_admission_category' to numerical codes
# Mapping: 'None' = 0, 'Very Frequent' = 1, 'Rare' = 2, 'Occasional' = 3, 'Frequent' = 4
df['hospital_admission_category'] =
df['hospital_admission_category'].astype('category').cat.codes

```

Step 187

```
# Save DataFrame to CSV
df.to_csv(r"C:\Users\ASUS\OneDrive\Desktop\PHD Business
admin\Analysis\Mastersheet\lasi_cleaned.csv", index=False)
# Save DataFrame to Excel
df.to_excel(r"C:\Users\ASUS\OneDrive\Desktop\PHD Business
admin\Analysis\Mastersheet\lasi_cleaned.xlsx", index=False)
```

Step 188

```
# Check data types of all columns
print(df.dtypes)
```

Step 189

```
# Filter non-numeric columns
non_numeric = df.select_dtypes(exclude=["float64"])
print(non_numeric.columns)
```

Step 190

```
# List of columns to convert
columns_to_float = [
    'Age_group', 'Gender_cat', 'education_cat', 'residence_cat', 'religion_cat',
    'caste_category_cat',
    'current_marital_status_cat', 'living_arrangements_cat', 'mpce_category',
    'work_hours_category',
    'hospital_admission_category', 'occupation_categorical',
    'currently_looking_for_job_categorical',
    'Not_employed_categorical', 'use_benefits_categorical',
    'aware_of_govt_oldagebenefits_categorical',
    'aware_welfareparents_senior_citizen_categorical',
    'covered_with_workrelated_ps_categorical',
    'covered_provident_fund_categorical', 'covered_with_some_health_insurance_categorical',
    'covered_with_injury_insurance_work_categorical',
    'covered_medical_reimbursement_categorical',
    'covered_with_workers_employment_insurance_categorical',
    'reason_for_no_health_insurance_categorical',
    'total_treatment_support_category', 'numer_opd_visits_1year_cat_code',
    'physical_health_index_cat_code',
    'mental_well_being_index_cat_code', 'functional_ability_index_cat_code',
    'economic_security_index_cat_code',
    'healthcare_satisfaction_vars_index_cat_code', 'cost_opd_visit_category',
    'reimbursement_status_category',
    'benefit_status_various_gov_scheme_category',
```

```
'anticipated_vs_actual_health_expense_gap_category',
  'healthcare_financial_stress_level_numeric', 'economic_score_category', 'HAI_category'
]
```

```
# Convert selected columns to float
df[columns_to_float] = df[columns_to_float].astype(float)
```

Step 191

```
# Define the desired column list (as seen in your dtype output)
selected_columns = [
  'Age_group', 'Gender_cat', 'education_cat', 'residence_cat', 'religion_cat',
  'caste_category_cat', 'current_marital_status_cat', 'living_arrangements_cat',
  'mpce_category', 'work_hours_category', 'hospital_admission_category',
  'occupation_categorical',
  'currently_looking_for_job_categorical', 'use_benefits_categorical',
  'aware_of_govt_oldagebenefits_categorical',
  'aware_welfareparents_senior_citizen_categorical',
  'covered_with_workrelated_ps_categorical', 'covered_provident_fund_categorical',
  'covered_with_some_health_insurance_categorical',
  'covered_with_injury_insurance_work_categorical',
  'covered_medical_reimbursement_categorical',
  'covered_with_workers_employment_insurance_categorical',
  'reason_for_no_health_insurance_categorical', 'total_treatment_support_category',
  'numer_opd_visits_1year_cleaned', 'numer_opd_visits_1year_cat_code',
  'physical_health_index_cat_code', 'mental_well_being_index_cat_code',
  'functional_ability_index_cat_code',
  'healthcare_satisfaction_vars_index_cat_code', 'cost_opd_visit_category',
  'reimbursement_status_category', 'benefit_status_various_gov_scheme_category',
  'anticipated_vs_actual_health_expense_gap_category', 'expense_gap_category_numeric',
  'healthcare_financial_stress_level_numeric', 'healthcare_level_grouped_numeric',
  'economic_score_category', 'HAI_category'
]
```

```
# Create df1 with only the selected columns
df1 = df[selected_columns].copy()
```

Step 192

```
df1.info()
```

Step 193

```
# Check number of missing values in each column
df1.isnull().sum().sort_values(ascending=False)
```

Step 194

```
from sklearn.impute import KNNImputer
import numpy as np

# Step 1: Convert all non-numeric columns to numeric (if possible)
# Note: KNNImputer works only on numeric data
df1_numeric = df1.select_dtypes(include=['number']) # numeric-only subset

# Step 2: Initialize KNN Imputer
imputer = KNNImputer(n_neighbors=5)

# Step 3: Apply imputer to all numeric columns
df1_imputed = imputer.fit_transform(df1_numeric)

# Step 4: Reassign imputed values back to df1
df1[df1_numeric.columns] = df1_imputed
```

Step 195

```
df1.info()
```

Step 196

```
# Drop less useful or redundant columns
df1 = df1.drop([
    'currently_looking_for_job_categorical',
    'covered_with_workrelated_ps_categorical', 'covered_provident_fund_categorical'
], axis=1)

# Rename the long column name to something simpler
df1 = df1.rename(columns={
    'anticipated_vs_actual_health_expense_gap_category': 'health_exp_gap_cat'
})
```

Step 197

```
df1 = df1.drop(columns=['use_benefits_categorical'])
```

Step 198

```
df1 =  
df1.drop(['reason_for_no_health_insurance_categorical','healthcare_level_grouped_numeric'  
], axis=1)
```

Step 199

```
X = df1.drop(columns=['economic_score_category'])  
y = df1['economic_score_category']
```

Step 200

```
from sklearn.model_selection import train_test_split  
  
X_train, X_test, y_train, y_test = train_test_split(  
    X, y, test_size=0.2, random_state=42, stratify=y  
)
```

Step 201

```
from sklearn.tree import DecisionTreeClassifier  
from sklearn.metrics import classification_report  
  
# Gini Tree  
clf_gini = DecisionTreeClassifier(criterion='gini', random_state=42)  
clf_gini.fit(X_train, y_train)  
  
# Entropy Tree  
clf_entropy = DecisionTreeClassifier(criterion='entropy', random_state=42)  
clf_entropy.fit(X_train, y_train)
```

Step 202

```
import pandas as pd  
  
gini_importances = pd.Series(clf_gini.feature_importances_, index=X.columns)  
top10_gini = gini_importances.sort_values(ascending=False).head(11)  
  
print("Top 10 Features by Gini Importance:")  
print(top10_gini)
```

Step 203

```

from sklearn.tree import plot_tree
import matplotlib.pyplot as plt

plt.figure(figsize=(14, 6))
plt.title("Decision Tree (Gini)")
plot_tree(clf_gini, feature_names=X.columns, class_names=True, max_depth=3, filled=True)
plt.show()

plt.figure(figsize=(14, 6))
plt.title("Decision Tree (Entropy)")
plot_tree(clf_entropy, feature_names=X.columns, class_names=True, max_depth=3,
filled=True)
plt.show()

```

Step 204

```

from sklearn.metrics import accuracy_score

y_pred_gini = clf_gini.predict(X_test)
y_pred_entropy = clf_entropy.predict(X_test)

print("Gini Tree Classification Report:")
print(classification_report(y_test, y_pred_gini))

print("Entropy Tree Classification Report:")
print(classification_report(y_test, y_pred_entropy))

print("Gini Accuracy:", accuracy_score(y_test, y_pred_gini))
print("Entropy Accuracy:", accuracy_score(y_test, y_pred_entropy))

```

Step 205

```

train_acc_gini = clf_gini.score(X_train, y_train)
train_acc_entropy = clf_entropy.score(X_train, y_train)

print("Gini Train Accuracy:", train_acc_gini)
print("Entropy Train Accuracy:", train_acc_entropy)

```

Step 206

```

from sklearn.model_selection import GridSearchCV

param_grid = {
    'max_depth': [3, 5, 10, None],

```

```

        'min_samples_split': [2, 10, 20],
        'min_samples_leaf': [1, 5, 10],
        'criterion': ['gini', 'entropy']
    }

    grid = GridSearchCV(
        DecisionTreeClassifier(random_state=42),
        param_grid,
        cv=5,
        scoring='f1_macro',
        n_jobs=-1
    )

    grid.fit(X_train, y_train)

    print("Best Parameters:", grid.best_params_)
    print("Best CV Score:", grid.best_score_)

    # Evaluate the best model on the test set
    from sklearn.metrics import classification_report
    y_pred_best = grid.predict(X_test)
    print(classification_report(y_test, y_pred_best))

```

Step 207

```

train_acc = grid.best_estimator_.score(X_train, y_train)
test_acc = grid.best_estimator_.score(X_test, y_test)

print("Train Accuracy:", train_acc)
print("Test Accuracy:", test_acc)

```

Step 208

```

import matplotlib.pyplot as plt
import pandas as pd

importances = pd.Series(grid.best_estimator_.feature_importances_, index=X.columns)
top10 = importances.sort_values(ascending=False).head(10)

top10.plot(kind='barh', title='Top 10 Features by Importance', figsize=(8,6))
plt.xlabel("Gini Importance")
plt.gca().invert_yaxis()
plt.tight_layout()
plt.show()

```

Step 209

```
import pandas as pd

# Get feature importances and sort them
importances = pd.Series(grid.best_estimator_.feature_importances_, index=X.columns)
top10 = importances.sort_values(ascending=False).head(10)

# Convert to DataFrame for tabular display
top10_df = top10.reset_index()
top10_df.columns = ['Feature', 'Importance']

# Optional: Format as percentage
top10_df['Importance (%)'] = (top10_df['Importance'] * 100).round(2)

# Display the table
print(top10_df)
```

Step 210

```
import pandas as pd
import numpy as np

import shap

# Example: for XGBoost or RandomForest
# Replace `your_model` with rf_grid.best_estimator_ or xgb_random.best_estimator_ or
# grid.best_estimator_ for DecisionTree

explainer = shap.TreeExplainer(grid.best_estimator_)
shap_values = explainer.shap_values(X_test, check_additivity=False) # For multiclass:
shape = (n_samples, n_features, n_classes)

# Extract SHAP values for class 0 (you can loop over 0, 1, 2)
class_index = 0 # choose 0, 1, or 2

# Get SHAP values for that class: shape (n_samples, n_features)
shap_class = shap_values[:, :, class_index]

# Create DataFrame
shap_df = pd.DataFrame(shap_class, columns=X_test.columns)

# Compute summary
mean_shap = shap_df.mean()
```

```

mean_abs_shap = shap_df.abs().mean()

summary_df = pd.DataFrame({
    'Feature': mean_shap.index,
    'Mean SHAP Value': mean_shap.values,
    'Mean(|SHAP Value|)': mean_abs_shap.values
})

# Add direction and percentage contribution
summary_df['Direction'] = summary_df['Mean SHAP Value'].apply(lambda x: 'Positive' if x >
0 else 'Negative')
summary_df['SHAP (%)'] = (summary_df['Mean(|SHAP Value|)'] /
summary_df['Mean(|SHAP Value|)'].sum() * 100).round(2)

# Top 10 features
top10 = summary_df.sort_values(by='Mean(|SHAP Value|)',
ascending=False).head(10).reset_index(drop=True)

print(top10)

```

Step 211

```

import pandas as pd
import numpy as np

# shap_values should be of shape (n_samples, n_features, n_classes)
assert shap_values.shape[2] == 3 # Ensure it's a 3-class model

# Store all summaries
all_class_summaries = []

# Loop over class indices (0, 1, 2)
for class_index in range(shap_values.shape[2]):
    # Get SHAP values for that class
    shap_class = shap_values[:, :, class_index]

    # Create DataFrame
    shap_df = pd.DataFrame(shap_class, columns=X_test.columns)

    # Compute mean SHAP and mean absolute SHAP
    mean_shap = shap_df.mean()
    mean_abs_shap = shap_df.abs().mean()

    # Summary table
    summary_df = pd.DataFrame({

```

```

        'Feature': mean_shap.index,
        'Mean SHAP Value': mean_shap.values,
        'Mean(|SHAP Value|)': mean_abs_shap.values
    })

    # Add direction and percentage contribution
    summary_df['Direction'] = summary_df['Mean SHAP Value'].apply(lambda x: 'Positive' if
x > 0 else 'Negative')
    summary_df['SHAP (%)'] = (summary_df['Mean(|SHAP Value|)'] /
summary_df['Mean(|SHAP Value|)'].sum() * 100).round(2)

    # Sort and keep top 10
    top10 = summary_df.sort_values(by='Mean(|SHAP Value|)',
ascending=False).head(10).reset_index(drop=True)

    print(f"\n ♦ Top 10 SHAP Features for Class {class_index}:")
    print(top10)

    # Optionally store all
    all_class_summaries.append(top10)

```

Step 212

```
print(np.shape(shap_values))
```

Step 213

```

from sklearn.metrics import roc_curve, auc
from sklearn.preprocessing import label_binarize
from sklearn.multiclass import OneVsRestClassifier
from sklearn.metrics import RocCurveDisplay
import matplotlib.pyplot as plt
import numpy as np

```

Step 214

```

import matplotlib.pyplot as plt
from sklearn.metrics import roc_curve, auc
from sklearn.preprocessing import label_binarize
from sklearn.multiclass import OneVsRestClassifier
import numpy as np

```

```

# Step 1: Binarize the output (important for multiclass ROC)
classes = [0, 1, 2]

```

```

y_test_bin = label_binarize(y_test, classes=classes)

# Step 2: Get predicted probabilities from best model
y_score = grid.predict_proba(X_test)

# Step 3: Compute ROC and AUC for each class
fpr = dict()
tpr = dict()
roc_auc = dict()

for i in range(len(classes)):
    fpr[i], tpr[i], _ = roc_curve(y_test_bin[:, i], y_score[:, i])
    roc_auc[i] = auc(fpr[i], tpr[i])

# Step 4: Plot ROC curves
plt.figure(figsize=(8, 6))
colors = ['blue', 'green', 'red']
for i, color in zip(classes, colors):
    plt.plot(fpr[i], tpr[i], color=color, lw=2,
             label=f'Class {i} (AUC = {roc_auc[i]:.2f})')

plt.plot([0, 1], [0, 1], 'k--', lw=2)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate', fontsize=12)
plt.ylabel('True Positive Rate', fontsize=12)
plt.title('ROC Curve for Multiclass Classification', fontsize=14)
plt.legend(loc="lower right")
plt.grid(True)
plt.tight_layout()
plt.show()

```

Step 215

```

from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV

# Parameter grid for Random Forest
rf_param_grid = {
    'n_estimators': [100, 200],
    'max_depth': [None, 10, 20],
    'min_samples_split': [2, 10],
    'min_samples_leaf': [1, 5],
    'criterion': ['gini', 'entropy']
}

```

```

# GridSearchCV
rf_grid = GridSearchCV(
    RandomForestClassifier(random_state=42),
    rf_param_grid,
    cv=5,
    scoring='f1_macro',
    n_jobs=-1,
    verbose=1
)

rf_grid.fit(X_train, y_train)

# Best parameters and score
print("Best Parameters:", rf_grid.best_params_)
print("Best CV Score:", rf_grid.best_score_)

```

Step 216

```

from sklearn.metrics import classification_report

y_pred_rf = rf_grid.predict(X_test)
print(classification_report(y_test, y_pred_rf))

```

Step 217

```

from sklearn.preprocessing import label_binarize
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt
import numpy as np

classes = sorted(y.unique())
y_test_bin = label_binarize(y_test, classes=classes)
y_score_rf = rf_grid.predict_proba(X_test)

# ROC per class
fpr, tpr, roc_auc = {}, {}, {}
for i in range(len(classes)):
    fpr[i], tpr[i], _ = roc_curve(y_test_bin[:, i], y_score_rf[:, i])
    roc_auc[i] = auc(fpr[i], tpr[i])

# Plot
plt.figure(figsize=(8, 6))
colors = ['blue', 'green', 'red']
for i, color in zip(classes, colors):

```

```

plt.plot(fpr[i], tpr[i], color=color, lw=2,
         label=f'Class {i} (AUC = {roc_auc[i]:.2f})')

plt.plot([0, 1], [0, 1], 'k--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Random Forest ROC Curve (Multiclass)')
plt.legend(loc='lower right')
plt.grid(True)
plt.tight_layout()
plt.show()

```

Step 218

```

from sklearn.metrics import accuracy_score, f1_score

# Predict on training and test sets
y_train_pred = rf_grid.predict(X_train)
y_test_pred = rf_grid.predict(X_test)

# Accuracy
train_acc = accuracy_score(y_train, y_train_pred)
test_acc = accuracy_score(y_test, y_test_pred)

# F1-score (macro average for multiclass)
train_f1 = f1_score(y_train, y_train_pred, average='macro')
test_f1 = f1_score(y_test, y_test_pred, average='macro')

# Print results
print(f'Train Accuracy: {train_acc:.4f}')
print(f'Test Accuracy: {test_acc:.4f}')
print(f'Train F1 Score: {train_f1:.4f}')
print(f'Test F1 Score: {test_f1:.4f}')

```

Step 219

```

# Tighter grid to reduce overfitting
rf_param_grid = {
    'n_estimators': [100], # Keep moderate
    'max_depth': [5, 10, 15],
    'min_samples_split': [5, 10],
    'min_samples_leaf': [5, 10],
    'criterion': ['gini'] # You can also add 'entropy' if you want to compare}

```

Step 220

```

from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV

rf_grid = GridSearchCV(
    estimator=RandomForestClassifier(random_state=42),
    param_grid=rf_param_grid,
    cv=5,
    scoring='f1_macro',
    n_jobs=-1,
    verbose=1
)

rf_grid.fit(X_train, y_train)

```

Step 221

```

from sklearn.metrics import accuracy_score, f1_score

# Predict on both train and test sets
y_train_pred = rf_grid.predict(X_train)
y_test_pred = rf_grid.predict(X_test)

# Compute accuracy and F1 scores
train_acc = accuracy_score(y_train, y_train_pred)
test_acc = accuracy_score(y_test, y_test_pred)

train_f1 = f1_score(y_train, y_train_pred, average='macro')
test_f1 = f1_score(y_test, y_test_pred, average='macro')

# Print the results
print(f"Train Accuracy: {train_acc:.4f}")
print(f"Test Accuracy: {test_acc:.4f}")
print(f"Train F1 Score: {train_f1:.4f}")
print(f"Test F1 Score: {test_f1:.4f}")

```

Step 222

```

from sklearn.metrics import classification_report

y_pred_rf = rf_grid.predict(X_test)
print(classification_report(y_test, y_pred_rf))

```

Step 223

```

import pandas as pd
import matplotlib.pyplot as plt

# Extract feature importances
importances = pd.Series(rf_grid.best_estimator_.feature_importances_, index=X.columns)

# Top 10 features
top10 = importances.sort_values(ascending=False).head(10)

# Display as table
top10_df = top10.reset_index()
top10_df.columns = ['Feature', 'Importance']
top10_df['Importance (%)'] = (top10_df['Importance'] * 100).round(2)

print(top10_df)

```

Step 224

```

def get_shap_summary_df(shap_array, class_index, feature_names, top_n=10):
    """
    Extract top N SHAP values for a specific class from a 3D SHAP array.

    Parameters:
    - shap_array: numpy array of shape (n_samples, n_features, n_classes)
    - class_index: index of the class (e.g., 0, 1, 2)
    - feature_names: list of feature names (columns of X_test)
    - top_n: number of top features to return

    Returns:
    - DataFrame with SHAP summary for the specified class
    """
    # Extract SHAP values for this class
    shap_class = shap_array[:, :, class_index] # shape: (n_samples, n_features)

    # Create SHAP DataFrame
    shap_df = pd.DataFrame(shap_class, columns=feature_names)

    # Compute summary
    mean_shap = shap_df.mean()
    mean_abs_shap = shap_df.abs().mean()

    summary_df = pd.DataFrame({
        'Feature': mean_shap.index,
        'Mean SHAP Value': mean_shap.values,
        'Mean(|SHAP Value|)': mean_abs_shap.values
    })

```

```
}}
```

```
summary_df['Direction'] = summary_df['Mean SHAP Value'].apply(lambda x: 'Positive' if
x > 0 else 'Negative')
summary_df['SHAP (%)'] = (summary_df['Mean(|SHAP Value|)'] /
summary_df['Mean(|SHAP Value|)'].sum() * 100).round(2)

return summary_df.sort_values(by='Mean(|SHAP Value|)',
ascending=False).head(top_n).reset_index(drop=True)
```

Step 225

```
shap_summary_class_0 = get_shap_summary_df(shap_values, 0, X_test.columns)
shap_summary_class_1 = get_shap_summary_df(shap_values, 1, X_test.columns)
shap_summary_class_2 = get_shap_summary_df(shap_values, 2, X_test.columns)

print("\n ♦ Class 0 SHAP Summary\n", shap_summary_class_0)
print("\n ♦ Class 1 SHAP Summary\n", shap_summary_class_1)
print("\n ♦ Class 2 SHAP Summary\n", shap_summary_class_2)
```

Step 226

```
top10.plot(kind='barh', figsize=(8, 6), title='Top 10 Features by Importance')
plt.xlabel("Feature Importance (Gini-based)")
plt.gca().invert_yaxis()
plt.tight_layout()
plt.show()
```

Step 227

```
from sklearn.preprocessing import label_binarize
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt
import numpy as np

# Binarize the test labels (One-vs-Rest approach)
classes = sorted(y.unique())
y_test_bin = label_binarize(y_test, classes=classes)

# Get predicted probabilities for each class
y_score = rf_grid.predict_proba(X_test)

# Compute ROC curve and AUC for each class
fpr = dict()
```

```

tpr = dict()
roc_auc = dict()

for i in range(len(classes)):
    fpr[i], tpr[i], _ = roc_curve(y_test_bin[:, i], y_score[:, i])
    roc_auc[i] = auc(fpr[i], tpr[i])

# Plot the ROC curves
plt.figure(figsize=(8, 6))
colors = ['blue', 'green', 'red']
for i, color in zip(classes, colors):
    plt.plot(fpr[i], tpr[i], color=color, lw=2,
             label=f'Class {i} (AUC = {roc_auc[i]:.2f})')

# Diagonal reference line
plt.plot([0, 1], [0, 1], 'k--', lw=1)

plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate', fontsize=12)
plt.ylabel('True Positive Rate', fontsize=12)
plt.title('Multiclass ROC Curve (Random Forest)', fontsize=14)
plt.legend(loc='lower right')
plt.grid(True)
plt.tight_layout()
plt.show()

```

Step 228

```

import xgboost as xgb
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score, f1_score, classification_report

# Define XGBoost classifier
xgb_clf = xgb.XGBClassifier(
    objective='multi:softprob',
    num_class=len(y.unique()),
    eval_metric='mlogloss',
    use_label_encoder=False,
    random_state=42
)

# Parameter grid
xgb_param_grid = {
    'max_depth': [3, 5, 10],

```

```

    'learning_rate': [0.1, 0.3],
    'n_estimators': [100, 200],
    'subsample': [0.8, 1.0],
    'colsample_bytree': [0.8, 1.0]
}

# Grid Search
xgb_grid = GridSearchCV(
    xgb_clf,
    xgb_param_grid,
    scoring='f1_macro',
    cv=5,
    n_jobs=-1,
    verbose=1
)

# Fit model
xgb_grid.fit(X_train, y_train)

# Best parameters and scores
print("Best Parameters:", xgb_grid.best_params_)
print("Best CV Score:", xgb_grid.best_score_)

# Evaluate on test set
y_pred_xgb = xgb_grid.predict(X_test)
print(classification_report(y_test, y_pred_xgb))

# Accuracy and F1
train_acc = accuracy_score(y_train, xgb_grid.predict(X_train))
test_acc = accuracy_score(y_test, y_pred_xgb)
train_f1 = f1_score(y_train, xgb_grid.predict(X_train), average='macro')
test_f1 = f1_score(y_test, y_pred_xgb, average='macro')

print(f"Train Accuracy: {train_acc:.4f}")
print(f"Test Accuracy: {test_acc:.4f}")
print(f"Train F1 Score: {train_f1:.4f}")
print(f"Test F1 Score: {test_f1:.4f}")

```

Step 229

```

from sklearn.preprocessing import label_binarize
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt
import numpy as np

```

```

# Binarize target for ROC
classes = sorted(y.unique())
y_test_bin = label_binarize(y_test, classes=classes)
y_score_xgb = xgb_grid.predict_proba(X_test)

# Compute ROC curve & AUC for each class
fpr, tpr, roc_auc = {}, {}, {}
for i in range(len(classes)):
    fpr[i], tpr[i], _ = roc_curve(y_test_bin[:, i], y_score_xgb[:, i])
    roc_auc[i] = auc(fpr[i], tpr[i])

# Plot
plt.figure(figsize=(8, 6))
colors = ['blue', 'green', 'red']
for i, color in zip(classes, colors):
    plt.plot(fpr[i], tpr[i], lw=2, color=color,
             label=f'Class {i} (AUC = {roc_auc[i]:.2f})')

plt.plot([0, 1], [0, 1], 'k--', lw=1)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Multiclass ROC Curve (XGBoost)')
plt.legend(loc='lower right')
plt.grid(True)
plt.tight_layout()
plt.show()

```

Step 230

```

from sklearn.model_selection import RandomizedSearchCV
from scipy.stats import uniform, randint

xgb_random = RandomizedSearchCV(
    estimator=xgb_clf,
    param_distributions=xgb_param_grid,
    n_iter=30, # Try 30 random combinations
    scoring='f1_macro',
    cv=5,
    verbose=1,
    random_state=42,
    n_jobs=-1
)

xgb_random.fit(X_train, y_train)

```

Step 231

```
from sklearn.metrics import classification_report

print("🔍 Classification Report (XGBoost RandomizedSearchCV):")
print(classification_report(y_test, y_pred_xgb))
```

Step 232

```
from sklearn.metrics import classification_report

# Predict using the best model from RandomizedSearchCV
y_pred_xgb = xgb_random.predict(X_test)

# Print classification report
print("🔍 Classification Report (XGBoost RandomizedSearchCV):")
print(classification_report(y_test, y_pred_xgb))
```

Step 233

```
from sklearn.metrics import accuracy_score, f1_score

# Predictions
y_train_pred = xgb_random.predict(X_train)
y_test_pred = y_pred_xgb # already predicted

# Accuracy
train_acc = accuracy_score(y_train, y_train_pred)
test_acc = accuracy_score(y_test, y_test_pred)

# F1 Score (macro)
train_f1 = f1_score(y_train, y_train_pred, average='macro')
test_f1 = f1_score(y_test, y_test_pred, average='macro')

# Report
print("\n📊 Model Performance:")
print(f"Train Accuracy: {train_acc:.4f}")
print(f"Test Accuracy: {test_acc:.4f}")
print(f"Train F1 Score: {train_f1:.4f}")
print(f"Test F1 Score: {test_f1:.4f}")
```

```
# Simple diagnostic
if abs(train_acc - test_acc) > 0.10:
    print("\n ⚠ Model is likely overfitting (high gap between train and test performance).")
elif test_acc < 0.60:
    print("\n ⚠ Model may be underfitting (low test accuracy).")
else:
    print("\n ✅ Model is generalizing well (no major signs of overfitting or underfitting).")
```

Step 234

```
from sklearn.preprocessing import label_binarize
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt
import numpy as np

# Binarize the test labels (for ROC calculation)
classes = sorted(y.unique())
y_test_bin = label_binarize(y_test, classes=classes)

# Predict probabilities
y_score = xgb_random.predict_proba(X_test)

# Compute ROC curve and AUC for each class
fpr, tpr, roc_auc = {}, {}, {}
for i in range(len(classes)):
    fpr[i], tpr[i], _ = roc_curve(y_test_bin[:, i], y_score[:, i])
    roc_auc[i] = auc(fpr[i], tpr[i])

# Plot all ROC curves
plt.figure(figsize=(8, 6))
colors = ['blue', 'green', 'red']
for i, color in zip(classes, colors):
    plt.plot(fpr[i], tpr[i], color=color, lw=2,
             label=f'Class {i} (AUC = {roc_auc[i]:.2f})')

plt.plot([0, 1], [0, 1], 'k--', lw=1)
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Multiclass ROC Curve (XGBoost)')
plt.legend(loc='lower right')
plt.grid(True)
plt.tight_layout()
plt.show()
```

Step 235

```
xgb_param_grid = {
    'max_depth': [3, 5, 7],
    'learning_rate': [0.01, 0.05, 0.1],
    'n_estimators': [100, 200],
    'subsample': [0.8, 1.0],
    'colsample_bytree': [0.8, 1.0],
    'min_child_weight': [1, 5, 10],
    'gamma': [0, 1, 5]
}
```

Step 236

```
xgb_random = RandomizedSearchCV(
    estimator=xgb_clf,
    param_distributions=xgb_param_grid,
    n_iter=30, # Number of random combinations to try
    scoring='f1_macro',
    cv=5,
    verbose=1,
    random_state=42,
    n_jobs=-1
)

xgb_random.fit(X_train, y_train)
```

Step 237

```
# Predictions
y_pred_xgb = xgb_random.predict(X_test)
y_train_pred_xgb = xgb_random.predict(X_train)

# Accuracy and F1 Score
train_acc = accuracy_score(y_train, y_train_pred_xgb)
test_acc = accuracy_score(y_test, y_pred_xgb)
train_f1 = f1_score(y_train, y_train_pred_xgb, average='macro')
test_f1 = f1_score(y_test, y_pred_xgb, average='macro')

print("\n 🏆 Model Performance:")
print(f"Train Accuracy: {train_acc:.4f}")
print(f"Test Accuracy: {test_acc:.4f}")
print(f"Train F1 Score: {train_f1:.4f}")
print(f"Test F1 Score: {test_f1:.4f}")

gap = train_f1 - test_f1
```

```

if gap > 0.1:
    print(" ⚠️ Model is likely **overfitting**.")
elif gap < -0.05:
    print(" ⚠️ Model is likely **underfitting**.")
else:
    print(" ✅ Model shows good generalization.")

# Classification Report
print("\n 📊 Classification Report (Test):")
print(classification_report(y_test, y_pred_xgb))

```

Step 238

```

from sklearn.preprocessing import label_binarize
from sklearn.metrics import roc_auc_score

# Binarize labels
y_test_bin = label_binarize(y_test, classes=np.unique(y))
y_score = xgb_random.predict_proba(X_test)
n_classes = y_test_bin.shape[1]

# Plot ROC curve
fpr = dict()
tpr = dict()
roc_auc = dict()

for i in range(n_classes):
    fpr[i], tpr[i], _ = roc_curve(y_test_bin[:, i], y_score[:, i])
    roc_auc[i] = auc(fpr[i], tpr[i])

# Plot
plt.figure(figsize=(8,6))
for i in range(n_classes):
    plt.plot(fpr[i], tpr[i], label=f"Class {i} (AUC = {roc_auc[i]:.2f})")
plt.plot([0, 1], [0, 1], 'k--')
plt.title("Multiclass ROC Curve (XGBoost)")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()

```

Step 239

```
import pandas as pd
import numpy as np

# Extract feature importances
importances = pd.Series(xgb_random.best_estimator_.feature_importances_,
                        index=X.columns)

# Top 10 features
top10 = importances.sort_values(ascending=False).head(10).reset_index()
top10.columns = ['Feature', 'Importance']
top10['Importance (%)'] = (top10['Importance'] * 100).round(2)

# Display top 10
print("📈 Top 10 Features by XGBoost Importance:")
print(top10)
```

Step 240

```
import shap

# Use SHAP's TreeExplainer
explainer = shap.TreeExplainer(xgb_random.best_estimator_)
shap_values = explainer.shap_values(X_test)

# Create a reusable function
def get_shap_summary_df(shap_input, class_index, feature_names, top_n=10):
    shap_class = shap_input[:, :, class_index]
    shap_df = pd.DataFrame(shap_class, columns=feature_names)

    mean_shap = shap_df.mean()
    mean_abs_shap = shap_df.abs().mean()

    summary_df = pd.DataFrame({
        'Feature': mean_shap.index,
        'Mean SHAP Value': mean_shap.values,
        'Mean(|SHAP Value|)': mean_abs_shap.values
    })
    summary_df['Direction'] = summary_df['Mean SHAP Value'].apply(lambda x: 'Positive' if
x > 0 else 'Negative')
    summary_df['SHAP (%)'] = (summary_df['Mean(|SHAP Value|)'] /
summary_df['Mean(|SHAP Value|)'].sum() * 100).round(2)

    return summary_df.sort_values(by='Mean(|SHAP Value|)',
```

```

ascending=False).head(top_n).reset_index(drop=True)

# Loop over all three classes
for i in range(3):
    print(f"\n🔍 Top 10 SHAP Features for Class {i}:")
    display(get_shap_summary_df(shap_values, i, X_test.columns))

```

Step 241

```

pip install semopy

```

Step 242

```

df1.info()

```

Step 243

```

from semopy import Model
import pandas as pd

# SEM model description (as before)
model_desc = """
# Latent variables
Health_Status =~ physical_health_index_cat_code + mental_well_being_index_cat_code +
functional_ability_index_cat_code
Insurance_Coverage =~ covered_with_some_health_insurance_categorical +
covered_with_injury_insurance_work_categorical +
covered_with_workers_employment_insurance_categorical
Awareness =~ aware_of_govt_oldagebenefits_categorical +
aware_welfareparents_senior_citizen_categorical

# Structural paths
economic_score_category ~ Health_Status + Insurance_Coverage + Awareness +
education_cat + Gender_cat + Age_group + mpce_category
Health_Status ~ Age_group + Gender_cat + hospital_admission_category +
health_exp_gap_cat
"""

# Create and fit the SEM model
model = Model(model_desc)
model.fit(df1)

# Show path coefficients
print("🔍 Path Coefficients and Loadings:")
estimates = model.inspect()

```

```
print(estimates[['lval', 'op', 'rval', 'Estimate', 'p-value']]))
```

Step 244

```
from semopy import Model
from semopy import calc_stats #  This is the correct import in 2.3.11

# Define your model as before
model_desc = """
# Latent variables
Health_Status =~ physical_health_index_cat_code + mental_well_being_index_cat_code +
functional_ability_index_cat_code
Insurance_Coverage =~ covered_with_some_health_insurance_categorical +
covered_with_injury_insurance_work_categorical +
covered_with_workers_employment_insurance_categorical
Awareness =~ aware_of_govt_oldagebenefits_categorical +
aware_welfareparents_senior_citizen_categorical

# Structural paths
economic_score_category ~ Health_Status + Insurance_Coverage + Awareness +
education_cat + Gender_cat + Age_group + mpce_category
Health_Status ~ Age_group + Gender_cat + hospital_admission_category +
health_exp_gap_cat
"""

# Fit the model
model = Model(model_desc)
model.fit(df1)

# Get parameter estimates
estimates = model.inspect()
print( Path Coefficients:")
print(estimates[['lval', 'op', 'rval', 'Estimate', 'p-value']]))
```

Step 245

```
from semopy import calc_stats

# Calculate fit statistics
stats = calc_stats(model)

# Show all available fit statistics
print("\n  Model Fit Statistics:")
```

```
print(stats) # Don't subset columns
```

Step 246

```
model_desc = """
# Latent variables
Health_Status =~ physical_health_index_cat_code + mental_well_being_index_cat_code +
functional_ability_index_cat_code
Insurance_Coverage =~ covered_with_some_health_insurance_categorical +
covered_with_injury_insurance_work_categorical +
covered_with_workers_employment_insurance_categorical
Awareness =~ aware_of_govt_oldagebenefits_categorical +
aware_welfareparents_senior_citizen_categorical

# Direct structural paths
economic_score_category ~ Health_Status + Insurance_Coverage + Awareness

# Mediated paths
Health_Status ~ Age_group + Gender_cat + hospital_admission_category
Insurance_Coverage ~ education_cat + mpce_category
Awareness ~ education_cat + Age_group
"""
```

Step 247

```
from semopy import Model
model = Model(model_desc)
model.fit(df1)

estimates = model.inspect()
print(estimates[['lval', 'op', 'rval', 'Estimate', 'p-value']])
```

Step 248

```
from semopy import calc_stats
fit_stats = calc_stats(model)
print(fit_stats)
```

Step 249

```
!pip install networkx
```

Step 250

```

import networkx as nx
import matplotlib.pyplot as plt
import pandas as pd

# Get path coefficients from model
results = model.inspect()

# Filter to structural paths only (op == '~')
paths = results[results['op'] == '~']

# Create a directed graph
G = nx.DiGraph()

for idx, row in estimates.iterrows():
    source = row['lval']
    target = row['rval']
    estimate = row['Estimate']

    try:
        p_val = float(row['p-value'])
    except ValueError:
        p_val = 0 # or np.nan or continue

    style = 'dotted' if p_val > 0.05 else 'solid'
    color = 'green' if p_val > 0.05 else 'black'

    G.add_edge(source, target, weight=estimate, style=style, color=color)

# Draw the graph
pos = nx.spring_layout(G, seed=42) # positions for nodes

plt.figure(figsize=(12, 8))
edges = G.edges(data=True)

# Draw nodes
nx.draw_networkx_nodes(G, pos, node_size=1500, node_color='skyblue')

# Draw labels
nx.draw_networkx_labels(G, pos, font_size=10)

# Draw edges with custom styles
for (u, v, d) in edges:
    nx.draw_networkx_edges(
        G, pos,

```

```

        edgelist=[(u, v)],
        width=2,
        alpha=0.7,
        style=d['style'],
        edge_color=d['color']
    )
    # Add weight (estimate) labels on edges
    mid_x = (pos[u][0] + pos[v][0]) / 2
    mid_y = (pos[u][1] + pos[v][1]) / 2
    plt.text(mid_x, mid_y, f"{d['weight']}", fontsize=9, color=d['color'])

plt.title("🔍 SEM Mediation Path Diagram (without Graphviz)", fontsize=14)
plt.axis('off')
plt.tight_layout()
plt.show()

```

Step 251

```

from semopy import Model
import pandas as pd

# Step 1: Define SEM model with mediation and moderation structure
model_desc = """
# Latent variables
Health_Status =~ physical_health_index_cat_code + mental_well_being_index_cat_code +
functional_ability_index_cat_code
Insurance_Coverage =~ covered_with_some_health_insurance_categorical +
covered_with_injury_insurance_work_categorical +
covered_with_workers_employment_insurance_categorical
Awareness =~ aware_of_govt_oldagebenefits_categorical +
aware_welfareparents_senior_citizen_categorical

# Direct structural paths
economic_score_category ~ Health_Status + Insurance_Coverage + Awareness +
Health_x_Gender

# Mediated paths
Health_Status ~ Age_group + Gender_cat + hospital_admission_category
Insurance_Coverage ~ education_cat + mpce_category
Awareness ~ education_cat + Age_group

# Moderator path (interaction term)
Health_x_Gender =~ physical_health_index_cat_code # Use Health as proxy
"""

```

```

# Step 2: Create interaction term in data (for moderation)
df1['Health_x_Gender'] = df1['physical_health_index_cat_code'] * df1['Gender_cat']

# Step 3: Fit model
model = Model(model_desc)
model.fit(df1)

# Step 4: View path coefficients
print("\n 📊 Path Coefficients:")
estimates = model.inspect()
print(estimates[['lval', 'op', 'rval', 'Estimate', 'p-value']])

from semopy import calc_stats
fit_stats = calc_stats(model)
print(fit_stats)
# Step 6: OPTIONAL - Export if needed
# estimates.to_csv("sem_estimates.csv", index=False)
# effects.to_csv("sem_effects.csv", index=False)

```

Step 252

```

import geopandas as gpd
import pandas as pd
import matplotlib.pyplot as plt

# Load the GeoJSON file
gdf_states =
gpd.read_file(r"C:\Users\ASUS\Downloads\gadm41_IND_1.json\gadm41_IND_1.json")

# View available columns (usually includes 'NAME_1' for state names)
print(gdf_states.columns)
print(gdf_states[['NAME_1']].drop_duplicates())

```

Step 253

```
df1['state_clean'] = df1['state_clean'].values
```

Step 254

```
df1.info()
```

Step 255

```
import geopandas as gpd
import pandas as pd
import matplotlib.pyplot as plt
```

Step 256

```
import geopandas as gpd

# Correct path to the actual file
geojson_path = r"C:\Users\ASUS\Downloads\gadm41_IND_1.json\gadm41_IND_1.json" #
Check if this file exists

gdf = gpd.read_file(geojson_path)
```

Step 257

```
# Rename the state column for consistency
gdf = gdf.rename(columns={'NAME_1': 'state_clean'})

# Then clean the state names
gdf['state_clean'] = gdf['state_clean'].str.strip().str.title()
```

Step 258

```
print(df1['state_clean'].unique())
```

Step 259

```
import geopandas as gpd
import matplotlib.pyplot as plt

# Clean state names in both dataframes to ensure consistency
df1['state_clean'] = df1['state_clean'].str.strip().str.title()
gdf['state_clean'] = gdf['state_clean'].str.strip().str.title()
# Fix mismatches
gdf['state_clean'] = gdf['state_clean'].replace({
    'NcTofdelhi': 'Delhi',
    'Nctofdelhi': 'Delhi',
    'Chhatisgarh': 'Chhattisgarh'
})
# Set labels for better map titles
age_labels = {
    0: 'Age Group 60–74',
    1: 'Age Group 75–84',
```

```

    2: 'Age Group 85+'
}

# Create the figure with 3 side-by-side maps
fig, axes = plt.subplots(1, 3, figsize=(24, 8), constrained_layout=True)

for idx, age_code in enumerate([0, 1, 2]):
    # Filter for the current age group
    df_age = df1[df1['Age_group'] == age_code]

    # Calculate average economic score by state
    state_scores =
df_age.groupby('state_clean')['economic_score_category'].mean().reset_index()
    state_scores.columns = ['state_clean', 'avg_economic_score']

    # Merge with the geodataframe
    gdf_age = gdf.merge(state_scores, on='state_clean', how='left')

    # Plot the map
    gdf_age.plot(
        column='avg_economic_score',
        cmap='YlGnBu',
        linewidth=0.5,
        edgecolor='0.7',
        legend=True,
        ax=axes[idx]
    )

    # Add titles and clean visuals
    axes[idx].set_title(f'{age_labels[age_code]}', fontsize=14)
    axes[idx].axis('off')

# Add a super title across all subplots
plt.suptitle("Spatial Variation in Economic Participation by Age Group", fontsize=18)
plt.show()

```

Step 260

```
from matplotlib.colors import LinearSegmentedColormap
```

Step 261

```

import geopandas as gpd
import matplotlib.pyplot as plt
import pandas as pd

```

```

# Clean state names
df1['state_clean'] = df1['state_clean'].str.strip().str.title()
gdf['state_clean'] = gdf['state_clean'].str.strip().str.title()

# Fix mismatches
gdf['state_clean'] = gdf['state_clean'].replace({
    'Nctofdelhi': 'Delhi',
    'NcTofdelhi': 'Delhi',
    'Chhatisgarh': 'Chhattisgarh'
})

# Dissolve to avoid label duplicates
gdf = gdf.dissolve(by='state_clean', as_index=False)

# Define age group labels
age_labels = {
    0: '60-74',
    1: '75-84',
    2: '85+'
}

# Loop through each age group and plot
for age_code, group_label in age_labels.items():
    # Filter data
    df_age = df1[df1['Age_group'] == age_code]

    # Compute mean economic score per state
    state_scores =
df_age.groupby('state_clean')['economic_score_category'].mean().reset_index()
    state_scores.columns = ['state_clean', 'economic_score_category']

# Merge with GeoDataFrame
merged = gdf.merge(state_scores, on='state_clean', how='left')

# Create single map per group
fig, ax = plt.subplots(figsize=(13, 13))
merged.plot(
    column='economic_score_category',
    cmap='summer',
    linewidth=0.5,
    ax=ax,
    edgecolor='black',
    legend=True,
    missing_kwds={"color": "lightgrey", "label": "No data"}
)

```

```

# Add state labels
for _, row in merged.iterrows():
    point = row['geometry'].representative_point()
    label = (
        f'{row["state_clean"]}\n{row["economic_score_category"]:.2f}'
        if pd.notnull(row["economic_score_category"])
        else f'{row["state_clean"]}\nN/A'
    )
    ax.text(
        point.x, point.y, label,
        fontsize=6.5, ha='center', va='center', color='black'
    )

ax.set_title(f'Economic Participation Score by State – Age {group_label}', fontsize=16)
ax.axis('off')
plt.tight_layout()
plt.show()

```

Step 262

```
custom_cmap = LinearSegmentedColormap.from_list("red_orange_green", colors)
```

Step 263

```

import pandas as pd
import geopandas as gpd
import matplotlib.pyplot as plt

# Clean state names
df1['state_clean'] = df1['state_clean'].str.strip().str.title()
gdf['state_clean'] = gdf['state_clean'].str.strip().str.title()

# Fix common mismatches
gdf['state_clean'] = gdf['state_clean'].replace({
    'Nctofdelhi': 'Delhi',
    'NcTofdelhi': 'Delhi',
    'Chhatisgarh': 'Chhattisgarh'
})

# Dissolve geometries to ensure unique state geometries
gdf = gdf.dissolve(by='state_clean', as_index=False)

# Define age group labels
age_labels = {
    0: '60–74',

```

```

1: '75–84',
2: '85+'
}

# Loop through each age group and plot
for age_code, group_label in age_labels.items():
    # Filter the data for the specific age group
    df_age = df1[df1['Age_group'] == age_code]

    # Compute mean economic score per state
    state_scores =
df_age.groupby('state_clean')['economic_score_category'].mean().reset_index()

# Merge with GeoDataFrame
merged = gdf.merge(state_scores, on='state_clean', how='left')

# Create single map per group
fig, ax = plt.subplots(figsize=(13, 13))
merged.plot(
    column='economic_score_category',
    cmap='summer',
    linewidth=0.5,
    ax=ax,
    edgecolor='black',
    legend=True,
    missing_kwds={"color": "lightgrey", "label": "No data"}
)

# Add labels for each state
for _, row in merged.iterrows():
    point = row['geometry'].representative_point()
    score = row['economic_score_category']
    label = f'{row["state_clean"]}\n{score:.2f}' if pd.notnull(score) else
f'{row["state_clean"]}\nN/A'
    ax.text(
        point.x, point.y, label,
        fontsize=6.5, ha='center', va='center', color='black'
    )

# Title and layout adjustments
ax.set_title(f'Economic Participation Score by State – Age {group_label}', fontsize=16)
ax.axis('off')
plt.tight_layout()

# Save the figure

```

```

plt.savefig(
    f'C:/Users/ASUS/OneDrive/Desktop/PHD Business
admin/Analysis/economic_participation_age_{group_label.replace('-', '-')}.png',
    dpi=300,
    bbox_inches='tight'
)
plt.close()
plt.show()

```

Step 264

```

summary_table = []

# Loop through age groups
for age_code, group_label in age_labels.items():
    # Filter data for the age group
    df_age = df1[df1['Age_group'] == age_code]

    # Compute state-wise mean economic score
    state_scores =
df_age.groupby('state_clean')['economic_score_category'].mean().reset_index()
    state_scores.columns = ['state_clean', 'economic_score_category']

    # Add age group label
    state_scores['age_group'] = group_label

    # Append to summary list
    summary_table.append(state_scores)

# Combine all into one DataFrame
final_summary_df = pd.concat(summary_table, ignore_index=True)

# Reorder columns
final_summary_df = final_summary_df[['state_clean', 'age_group',
'economic_score_category']]

# Round scores for clarity
final_summary_df['economic_score_category'] =
final_summary_df['economic_score_category'].round(2)

# Show result
print(final_summary_df)

# Save to CSV
final_summary_df.to_csv(
    r"C:/Users/ASUS/OneDrive/Desktop/PHD Business

```

```
admin/Analysis/economic_participation_summary_table.csv",
    index=False
)
```

Step 265

```
# Dictionary to store below-mean states for each age group
below_mean_states = {}

# Loop over age groups
for age_code, group_label in age_labels.items():
    # Filter data for current age group
    df_age = df1[df1['Age_group'] == age_code]

    # Compute national mean
    national_mean = df_age['economic_score_category'].mean()

    # Compute state-wise mean
    state_scores =
df_age.groupby('state_clean')['economic_score_category'].mean().reset_index()

    # Flag states below national mean
    below_avg = state_scores[state_scores['economic_score_category'] < national_mean]

    # Store in dictionary
    below_mean_states[group_label] = below_avg.sort_values(by='economic_score_category')

    # Print summary
    print(f"\n ▼ States below national average for Age Group {group_label} (Mean =
{national_mean:.2f}):")
    print(below_avg[['state_clean', 'economic_score_category']].to_string(index=False))

# Optional: Save to Excel or CSV
# all_below_df = pd.concat(below_mean_states, names=['Age Group']).reset_index()
# all_below_df.to_csv("below_mean_states_by_age.csv", index=False)
```

Step 266

```
import geopandas as gpd
import matplotlib.pyplot as plt
import pandas as pd
from matplotlib.colors import LinearSegmentedColormap

# Custom color palette: light red → orange → yellow → green
```

```

colors = ["#ffcccc", "#ff9900", "#ffff66", "#ccff66", "#66cc66"]
custom_cmap = LinearSegmentedColormap.from_list("red_orange_green", colors)

# Clean state names
df1['state_clean'] = df1['state_clean'].str.strip().str.title()
gdf['state_clean'] = gdf['state_clean'].str.strip().str.title()

# Fix mismatches
gdf['state_clean'] = gdf['state_clean'].replace({
    'Nctofdelhi': 'Delhi',
    'NcTofdelhi': 'Delhi',
    'Chhatisgarh': 'Chhattisgarh'
})

# Dissolve geometries to avoid label overlap
gdf = gdf.dissolve(by='state_clean', as_index=False)

# Age group labels
age_labels = {
    0: '60–74',
    1: '75–84',
    2: '85+'
}

# Loop through age groups
for age_code, group_label in age_labels.items():
    # Filter data for current age group
    df_age = df1[df1['Age_group'] == age_code]

    # Calculate state-wise and national mean
    state_scores =
df_age.groupby('state_clean')['economic_score_category'].mean().reset_index()
    national_mean = df_age['economic_score_category'].mean()
    state_scores.columns = ['state_clean', 'economic_score_category']

    # Merge with geodata
    merged = gdf.merge(state_scores, on='state_clean', how='left')

    # Flag below-average states
    merged['below_mean'] = merged['economic_score_category'] < national_mean

# Start plot
fig, ax = plt.subplots(figsize=(13, 13))

# Main choropleth

```

```

merged.plot(
    column='economic_score_category',
    cmap= 'summer',
    linewidth=0.5,
    ax=ax,
    edgecolor='black',
    legend=True,
    missing_kwds={"color": "lightgrey", "label": "No data"}
)

# Highlight below-average states with red border
merged[merged['below_mean']].plot(
    ax=ax,
    facecolor='none',
    edgecolor='red',
    linewidth=2
)

# Add state labels and optional ⚠ warning
for _, row in merged.iterrows():
    point = row['geometry'].representative_point()
    econ_score = row['economic_score_category']
    label = (
        f'{row["state_clean"]}\n{econ_score:.2f}'
        if pd.notnull(econ_score)
        else f'{row["state_clean"]}\nN/A'
    )

    ax.text(
        point.x, point.y, label,
        fontsize=6.5, ha='center', va='center', color='black'
    )

    if row['below_mean'] and pd.notnull(econ_score):
        ax.text(
            point.x, point.y + 0.8, # small offset above the label
            '⚠ Below Avg',
            fontsize=6, color='red', ha='center'
        )

# Title and formatting
ax.set_title(f"Economic Participation Score by State – Age {group_label}", fontsize=16)
ax.axis('off')
plt.tight_layout()

```

```

# Save the figure
plt.savefig(
    f'C:/Users/ASUS/OneDrive/Desktop/PHD Business
admin/Analysis/economic_participation_age_{group_label}.png',
    dpi=300,
    bbox_inches='tight'
)

plt.show()

```

Step 267

```
pip install geopandas libpysal esda matplotlib
```

Step 268

```

# Prepare state-level average economic score
state_scores = df1.groupby('state_clean')['economic_score_category'].mean().reset_index()
gdf['state_clean'] = gdf['state_clean'].str.strip().str.title()
gdf = gdf.dissolve(by='state_clean', as_index=False)
gdf = gdf.merge(state_scores, on='state_clean', how='left')

```

Step 269

```

from libpysal.weights import Queen

# Create spatial weight matrix based on shared borders
w = Queen.from_dataframe(gdf)
w.transform = 'r' # Row-standardize weights

```

Step 270

```

from esda.moran import Moran

# Global Moran's I
y = gdf['economic_score_category'].fillna(0) # fill NA if needed
moran = Moran(y, w)

print(f'Global Moran's I: {moran.I:.4f}')
print(f'p-value: {moran.p_sim:.4f}')

```

Step 271

```
from libpysal.weights import Queen
from esda.moran import Moran_Local

# Spatial weights
w = Queen.from_dataframe(merged)
w.transform = 'r'

# Economic score (fill NA with 0 or use .dropna() if preferred)
y = merged['economic_score_category'].fillna(0)

# Local Moran's I
lisa = Moran_Local(y, w)

# Add results to merged GeoDataFrame, not gdf
merged['lisa_cluster'] = lisa.q
merged['lisa_significance'] = lisa.p_sim < 0.05

# Map cluster labels
cluster_labels = {
    1: "High-High",
    2: "Low-High",
    3: "Low-Low",
    4: "High-Low"
}
merged['lisa_label'] = merged['lisa_cluster'].map(cluster_labels)
```

Step 272

```
fig, ax = plt.subplots(1, 1, figsize=(12, 10))

merged.plot(
    column='lisa_label',
    categorical=True,
    legend=True,
    cmap='summer',
    linewidth=0.5,
    edgecolor='black',
    ax=ax
)

ax.set_title("LISA Cluster Map – Economic Participation", fontsize=16)
ax.axis('off')
plt.tight_layout()
```

```
plt.show()
```

Step 273

```
fig, ax = plt.subplots(1, 1, figsize=(12, 10))
```

```
# Plot LISA clusters
```

```
merged.plot(  
    column='lisa_label',  
    categorical=True,  
    legend=True,  
    cmap='summer',  
    linewidth=0.5,  
    edgecolor='black',  
    ax=ax  
)
```

```
# Add state name labels
```

```
for _, row in merged.iterrows():  
    point = row['geometry'].representative_point() # Use representative point (more accurate  
    than centroid)  
    state_name = row['state_clean']  
    ax.text(  
        point.x, point.y,  
        state_name,  
        fontsize=6, ha='center', va='center',  
        color='black'  
    )
```

```
ax.set_title("LISA Cluster Map – Economic Participation", fontsize=16)
```

```
ax.axis('off')
```

```
plt.tight_layout()
```

```
plt.show()
```

```
#  Save the figure
```

```
output_path = r"C:\Users\ASUS\OneDrive\Desktop\PHD Business  
admin\Analysis\LISA_Cluster_Economic_Participation.png"  
plt.savefig(output_path, dpi=300, bbox_inches='tight')
```

Step 274

```
# Create a table showing state, economic score, and LISA cluster label
```

```
lisa_table = merged[['state_clean', 'economic_score_category', 'lisa_label']].copy()
```

```
# Optional: sort the table for clarity
```

```
lisa_table = lisa_table.sort_values(by='lisa_label')
```

```
# Display in notebook  
print(lisa_table.head(40)) # Show top 10 rows
```

Step 275

```
from sklearn.preprocessing import StandardScaler  
from sklearn.cluster import KMeans  
from sklearn.decomposition import PCA  
  
# Select relevant features  
features = [  
    'Gender_cat', 'work_hours_category', 'covered_with_some_health_insurance_categorical',  
    'mpce_category', 'occupation_categorical', 'HAI_category',  
    'health_exp_gap_cat', 'healthcare_satisfaction_vars_index_cat_code',  
    'covered_medical_reimbursement_categorical', 'functional_ability_index_cat_code'  
]  
  
X = df1[features]  
  
# Standardize  
scaler = StandardScaler()  
X_scaled = scaler.fit_transform(X)
```

Step 276

```
# Choose optimal k using the elbow method (optional)  
from sklearn.metrics import silhouette_score  
  
scores = []  
for k in range(2, 10):  
    kmeans = KMeans(n_clusters=k, random_state=42)  
    labels = kmeans.fit_predict(X_scaled)  
    score = silhouette_score(X_scaled, labels)  
    scores.append((k, score))  
  
# Fit final model  
kmeans = KMeans(n_clusters=3, random_state=42)  
df1['cluster'] = kmeans.fit_predict(X_scaled)
```

Step 277

```
cluster_summary = df1.groupby('cluster')[features + ['economic_score_category']].mean()
print(cluster_summary)
```

Step 278

```
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

df1['pca1'] = X_pca[:, 0]
df1['pca2'] = X_pca[:, 1]

plt.figure(figsize=(10, 6))
for cluster in df1['cluster'].unique():
    subset = df1[df1['cluster'] == cluster]
    plt.scatter(subset['pca1'], subset['pca2'], label=f'Cluster {cluster}', alpha=0.6)
plt.legend()
plt.title("Cluster Visualization using PCA")
plt.xlabel("PCA 1")
plt.ylabel("PCA 2")
plt.grid(True)
plt.tight_layout()
plt.show()
```

Step 279

```
cluster_names = {
    0: "Healthy & Economically Stable",
    1: "ulnerable & Disengaged",
    2: "Unhealthy but Economically Engaged",
}
df1['cluster_label'] = df1['cluster'].map(cluster_names)
```

Step 280

```
cluster_summary = df1.groupby('cluster_label')[features +
['economic_score_category']].mean()
print(cluster_summary)
```

Step 281

```
from sklearn.metrics import silhouette_score, calinski_harabasz_score, davies_bouldin_score

# X is your input data used for clustering
# labels are cluster assignments from KMeans
```

```

labels = kmeans.labels_

print("Silhouette Score:", silhouette_score(X, labels).round(3))
print("Calinski-Harabasz Index:", calinski_harabasz_score(X, labels).round(3))
print("Davies-Bouldin Index:", davies_bouldin_score(X, labels).round(3))

```

Step 282

```
df1.info()
```

Step 283

```
df1['state_clean'].unique()
```

Step 284

```

# Get unique states and sort them
unique_states = sorted(df1['state_clean'].unique())

# Create a mapping: state -> number
state_mapping = {state: float(i) for i, state in enumerate(unique_states)}

# Show the mapping
for state, num in state_mapping.items():
    print(f'{state}: {num}')

```

Step 285

```
df1['state_clean'] = df1['state_clean'].map(state_mapping)
```

Step 286

```
df1.dtypes
```

Step 287

```

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from sklearn.ensemble import RandomForestClassifier

```

```
from sklearn.metrics import classification_report, confusion_matrix
import shap
import warnings
warnings.filterwarnings('ignore')
```

Step 288

```
pip install kneed
```

Step 289

```
from kneed import KneeLocator

# Reuse previously computed values
k_range = range(2, 11)
# Assuming you already have the `inertia` list from earlier

# Detect the elbow point
knee = KneeLocator(k_range, inertia, curve="convex", direction="decreasing")
optimal_k = knee.knee

print(f"Ideal number of clusters (elbow method): {optimal_k}")
```

Step 290

```
# Apply KMeans with optimal_k = 6
optimal_k = 6
kmeans = KMeans(n_clusters=optimal_k, random_state=42, n_init=10)
df['cluster'] = kmeans.fit_predict(X_scaled)

# Compute silhouette score
from sklearn.metrics import silhouette_score
final_silhouette = silhouette_score(X_scaled, df['cluster'])
print(f"Final Silhouette Score for k={optimal_k}: {final_silhouette:.3f}")
```

Step 291

```
from sklearn.decomposition import PCA
import seaborn as sns
import matplotlib.pyplot as plt

# Reduce dimensions for plotting
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)
```

```

df['pca1'] = X_pca[:, 0]
df['pca2'] = X_pca[:, 1]

# Scatter plot of clusters
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='pca1', y='pca2', hue='cluster', palette='tab10')
plt.title(f'KMeans Clustering (k={optimal_k}) with PCA')
plt.xlabel('PCA 1')
plt.ylabel('PCA 2')
plt.legend(title='Cluster')
plt.grid(True)
plt.show()

```

Step 292

```

best_k = np.argmax(silhouette_scores) + 2 # since range starts from 2
best_score = silhouette_scores[best_k - 2]
print(f"Best silhouette k: {best_k} with score {best_score:.3f}")

```

Step 293

```

# Try reducing dimensions before clustering
pca = PCA(n_components=0.90) # retain 90% variance
X_pca_reduced = pca.fit_transform(X_scaled)

# Run KMeans again
kmeans_pca = KMeans(n_clusters=6, random_state=42, n_init=10)
df['cluster_pca'] = kmeans_pca.fit_predict(X_pca_reduced)
print(f'Silhouette (PCA-reduced) k=6: {silhouette_score(X_pca_reduced,
df['cluster_pca']):.3f}')

```

Step 294

```

from sklearn.cluster import DBSCAN
from sklearn.metrics import silhouette_score

# Apply DBSCAN
dbscan = DBSCAN(eps=1.5, min_samples=10) # Adjust eps/min_samples as needed
labels = dbscan.fit_predict(X_pca_reduced)

# Assign cluster labels to df1
df1['dbscan_cluster'] = labels

# Filter out noise (-1) for silhouette score calculation

```

```

mask = labels != -1
if mask.sum() > 0:
    dbscan_score = silhouette_score(X_pca_reduced[mask], labels[mask])
    print(f"DBSCAN Silhouette Score (excluding noise): {dbscan_score:.3f}")
else:
    print("DBSCAN found no core samples.")

# Report number of clusters and noise points
n_clusters_dbscan = len(set(labels)) - (1 if -1 in labels else 0)
n_noise = list(labels).count(-1)
print(f"DBSCAN Clusters found: {n_clusters_dbscan}")
print(f"Noise points (unclustered): {n_noise}")

# Filter out noise points
clustered = df1[df1['dbscan_cluster'] != -1]

# Cross-tab between cluster and economic participation category
cluster_summary = pd.crosstab(
    clustered['dbscan_cluster'],
    clustered['economic_score_category'],
    normalize='index'
) * 100

# Add cluster sizes
cluster_counts = clustered['dbscan_cluster'].value_counts().sort_index()
cluster_summary.insert(0, 'Count', cluster_counts)

# Display results
print(cluster_summary.round(2))

```

Step 295

```

from sklearn.decomposition import PCA

# Retain 90% variance with PCA (suitable for clustering)
pca = PCA(n_components=0.90)
X_pca_reduced = pca.fit_transform(X_scaled)

print(f"PCA reduced shape: {X_pca_reduced.shape}")

```

Step 296

```

from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score

```

```

for k in range(2, 8):
    kmeans = KMeans(n_clusters=k, random_state=42, n_init=10)
    labels = kmeans.fit_predict(X_pca_reduced)
    score = silhouette_score(X_pca_reduced, labels)
    print(f'KMeans: k={k}, silhouette score = {score:.3f}')

```

Step 297

```

# Apply final KMeans
final_k = 3 # replace with your best k
kmeans = KMeans(n_clusters=final_k, random_state=42, n_init=10)
df['kmeans_cluster'] = kmeans.fit_predict(X_pca_reduced)

```

Step 298

```

from sklearn.cluster import AgglomerativeClustering

for k in range(2, 9):
    agg = AgglomerativeClustering(n_clusters=k)
    labels = agg.fit_predict(X_pca_reduced)
    score = silhouette_score(X_pca_reduced, labels)
    print(f'Agglomerative: k={k}, silhouette score = {score:.3f}')

```

Step 299

```

# Apply final Agglomerative
final_k = 2 # replace accordingly
agglo = AgglomerativeClustering(n_clusters=final_k)
df['agglo_cluster'] = agglo.fit_predict(X_pca_reduced)

```

Step 300

```

# KMeans results
print("\nKMeans Clusters Summary:")
print(pd.crosstab(df['kmeans_cluster'], df['economic_score_category'],
normalize='index').round(2) * 100)

# Agglomerative results
print("\nAgglomerative Clusters Summary:")
print(pd.crosstab(df['agglo_cluster'], df['economic_score_category'],
normalize='index').round(2) * 100)

```

Step 301

Deep learning

Step 302

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from tensorflow.keras.utils import to_categorical

# Target and features
X = df.drop(columns=['economic_score_category', 'cluster', 'dbscan_cluster',
                    'kmeans_cluster', 'agglo_cluster',
                    'pca1', 'pca2', 'cluster_label'], errors='ignore')
y = df['economic_score_category'].astype(int)

# Standardize X
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# One-hot encode y for classification
y_cat = to_categorical(y)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_cat, test_size=0.2,
                                                    random_state=42, stratify=y)
```

Step 303

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout

model = Sequential([
    Dense(128, activation='relu', input_shape=(X_train.shape[1],)),
    Dropout(0.3),
    Dense(64, activation='relu'),
    Dropout(0.3),
    Dense(3, activation='softmax') # 3 output classes
])
```

Step 304

```
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

history = model.fit(X_train, y_train, epochs=50, batch_size=64, validation_split=0.2,
                    verbose=1)
```

Step 305

```
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

```
history = model.fit(X_train, y_train, epochs=50, batch_size=64, validation_split=0.2,  
verbose=1)
```

Step 306

```
loss, accuracy = model.evaluate(X_test, y_test)  
print(f"Test Accuracy: {accuracy:.3f}")
```

Step 307

```
import matplotlib.pyplot as plt  
  
plt.plot(history.history['accuracy'], label='Train Acc')  
plt.plot(history.history['val_accuracy'], label='Val Acc')  
plt.xlabel('Epochs')  
plt.ylabel('Accuracy')  
plt.title('Model Training Accuracy')  
plt.legend()  
plt.grid(True)  
plt.show()
```

Step 308

```
import numpy as np  
import seaborn as sns  
import matplotlib.pyplot as plt  
from sklearn.metrics import classification_report, confusion_matrix  
  
# Step 1: Predict using the trained DNN model  
y_pred = model.predict(X_test) # predicted probabilities  
y_pred_classes = np.argmax(y_pred, axis=1) # convert to class labels  
  
# Step 2: Convert y_test (one-hot) to label format  
y_true = np.argmax(y_test, axis=1)  
  
# Step 3: Print classification report  
print("Classification Report:\n")  
print(classification_report(y_true, y_pred_classes, target_names=[  
    "Minimal Engagement", "Moderate Engagement", "High Engagement"  
]))
```

```
# Step 4: Confusion Matrix
cm = confusion_matrix(y_true, y_pred_classes)

# Plotting the confusion matrix
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=[
    "Minimal", "Moderate", "High"], yticklabels=[
    "Minimal", "Moderate", "High"])

plt.xlabel("Predicted Category")
plt.ylabel("True Category")
plt.title("Confusion Matrix: Economic Engagement Prediction")
plt.tight_layout()
plt.show()
```

Step 309

```
loss, accuracy = model.evaluate(X_train, y_train)
print(f"Train Accuracy: {accuracy:.3f}")
```

Step 310

```
# Define your variable groups
health_vars = [
    'HAI_category',
    'functional_ability_index_cat_code',
    'mental_well_being_index_cat_code'
]

insurance_vars = [
    'covered_with_some_health_insurance_categorical',
    'covered_with_injury_insurance_work_categorical',
    'covered_medical_reimbursement_categorical',
    'covered_with_workers_employment_insurance_categorical'
]

awareness_vars = [
    'aware_of_govt_oldagebenefits_categorical',
    'aware_welfareparents_senior_citizen_categorical'
]

economic_vars = [
    'mpce_category',
    'health_exp_gap_cat',
    'healthcare_financial_stress_level_numeric'
```

```
]
```

```
demographic_vars = [  
    'Age_group',  
    'Gender_cat',  
    'education_cat',  
    'residence_cat',  
    'religion_cat',  
    'caste_category_cat'  
]
```

```
# Combine all selected features
```

```
selected_vars = health_vars + insurance_vars + awareness_vars + economic_vars +  
demographic_vars
```

Step 311

```
# Group by economic_score_category and compute mean
```

```
grouped_summary = df[selected_vars +  
['economic_score_category']].groupby('economic_score_category').mean().round(3)
```

```
# Display
```

```
print(grouped_summary)
```

Step 312

```
from sklearn.preprocessing import OneHotEncoder
```

```
# One-hot encode 'state_clean'
```

```
state_encoded = pd.get_dummies(df['state_clean'], prefix='state')
```

```
# Drop unused columns and join encoded state
```

```
X = df.drop(columns=['economic_score_category', 'cluster', 'dbscan_cluster',  
'kmeans_cluster',  
                    'agglo_cluster', 'pca1', 'pca2', 'cluster_label', 'state_clean'], errors='ignore')
```

```
X = pd.concat([X, state_encoded], axis=1)
```

```
# Target
```

```
y = df['economic_score_category'].astype(int)
```

Step 313

```

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from tensorflow.keras.utils import to_categorical

# Scale features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# One-hot encode target
y_cat = to_categorical(y)

# Split dataset
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_cat, test_size=0.2,
random_state=42, stratify=y)

```

Step 314

```

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout

model = Sequential([
    Dense(256, activation='relu', input_shape=(X_train.shape[1],)),
    Dropout(0.3),
    Dense(128, activation='relu'),
    Dropout(0.3),
    Dense(64, activation='relu'),
    Dense(3, activation='softmax') # 3 classes for economic score
])

```

Step 315

```

model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

history = model.fit(X_train, y_train, epochs=50, batch_size=64, validation_split=0.2,
verbose=1)

```

Step 316

```

from sklearn.metrics import classification_report, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# Predict class probabilities

```

```

y_pred_prob = model.predict(X_test)

# Convert to class labels
y_pred = np.argmax(y_pred_prob, axis=1)
y_true = np.argmax(y_test, axis=1)

# Classification report
print(classification_report(y_true, y_pred, target_names=['Not Active', 'Part-Time', 'Fully Active']))

# Confusion matrix
cm = confusion_matrix(y_true, y_pred)

# Plot confusion matrix
plt.figure(figsize=(6, 5))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=['Not Active', 'Part-Time', 'Fully Active'],
            yticklabels=['Not Active', 'Part-Time', 'Fully Active'])
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.tight_layout()
plt.show()

```

Step 317

```

# Plot training history
plt.figure(figsize=(12, 5))

# Accuracy
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Train Acc')
plt.plot(history.history['val_accuracy'], label='Val Acc')
plt.title('Model Accuracy Over Epochs')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()

# Loss
plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Val Loss')
plt.title('Model Loss Over Epochs')
plt.xlabel('Epoch')
plt.ylabel('Loss')

```

```
plt.legend()
```

```
plt.tight_layout()  
plt.show()
```

Step 318

```
# Predict on the full dataset if needed (or use X_test and df_test subset)  
full_preds = model.predict(X_scaled)  
pred_labels = np.argmax(full_preds, axis=1)
```

```
# Add predictions to your original DataFrame  
df['dnn_predicted_category'] = pred_labels
```

```
# Save as CSV  
df.to_csv('dnn_predictions_with_states.csv', index=False)
```

```
# Optional: Group summary by state  
state_group_summary = df.groupby(['state_clean',  
'dnn_predicted_category']).size().unstack().fillna(0)
```

```
print(state_group_summary.head())
```

```
# Save summary too  
state_group_summary.to_csv("dnn_predicted_categories_by_state.csv")
```

Step 319

```
import matplotlib.pyplot as plt
```

```
# Rename columns for clarity  
state_group_summary.columns = ['Not Active', 'Part-Time', 'Fully Active']
```

```
# Plot as stacked bar chart  
state_group_summary.plot(kind='bar', stacked=True, figsize=(12, 6), colormap='viridis')
```

```
plt.title('Predicted Economic Participation by State (DNN Output)')  
plt.xlabel('State (state_clean code)')  
plt.ylabel('Number of Individuals')  
plt.legend(title='Predicted Category')  
plt.tight_layout()  
plt.grid(axis='y')  
plt.show()
```

Step 320

```
# Convert to percentage share per state
state_percent = state_group_summary.div(state_group_summary.sum(axis=1), axis=0) * 100
state_percent = state_percent.round(1)

# Plot normalized stacked bar chart
state_percent.plot(kind='bar', stacked=True, figsize=(12, 6), colormap='coolwarm')

plt.title('Percentage Distribution of Predicted Economic Categories by State')
plt.xlabel('State (state_clean code)')
plt.ylabel('Percentage')
plt.legend(title='Predicted Category')
plt.tight_layout()
plt.grid(axis='y')
plt.show()
```

Step 321

```
import pandas as pd

# Define your mapping from state_clean codes to state names
state_mapping = {
    0.0: 'Andaman & Nicobar', 1.0: 'Andhra Pradesh', 2.0: 'Arunachal Pradesh', 3.0: 'Assam',
    4.0: 'Bihar', 5.0: 'Chandigarh', 6.0: 'Chhattisgarh', 7.0: 'Dadra & Nagar Haveli',
    8.0: 'Daman & Diu', 9.0: 'Delhi', 10.0: 'Goa', 11.0: 'Gujarat', 12.0: 'Haryana',
    13.0: 'Himachal Pradesh', 14.0: 'Jammu & Kashmir', 15.0: 'Jharkhand', 16.0: 'Karnataka',
    17.0: 'Kerala', 18.0: 'Lakshadweep', 19.0: 'Madhya Pradesh', 20.0: 'Maharashtra',
    21.0: 'Manipur', 22.0: 'Meghalaya', 23.0: 'Mizoram', 24.0: 'Nagaland', 25.0: 'Odisha',
    26.0: 'Puducherry', 27.0: 'Punjab', 28.0: 'Rajasthan', 29.0: 'Sikkim', 30.0: 'Tamil Nadu',
    31.0: 'Telangana', 32.0: 'Tripura', 33.0: 'Uttarakhand', 34.0: 'Uttar Pradesh', 35.0: 'West
    Bengal'
}

# Map state_clean to state names
df['state_name'] = df['state_clean'].map(state_mapping)

# Group by state and prediction
state_pred_summary = df.groupby(['state_name',
    'dnn_predicted_category']).size().unstack(fill_value=0)

# Rename prediction categories for readability
state_pred_summary.columns = ['Not Active', 'Part-Time', 'Fully Active']

# Optional: Sort by total elderly population in each state
```

```

state_pred_summary['Total'] = state_pred_summary.sum(axis=1)
state_pred_summary = state_pred_summary.sort_values(by='Total', ascending=False)

# Display
print(state_pred_summary.drop(columns='Total').head(40)) # Display top 10 states

# Optional: Export to CSV
state_pred_summary.to_csv("dnn_predicted_economic_categories_by_state.csv")

```

Step 322

```

import geopandas as gpd
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap

# Load GeoDataFrame and clean state names
gdf =
gpd.read_file(r"C:\Users\ASUS\Downloads\gadm41_IND_1.json\gadm41_IND_1.json")
gdf['state_clean'] = gdf['NAME_1'].str.replace(" ", "").str.title()
gdf['state_clean'] = gdf['state_clean'].replace({
    'NcTofdelhi': 'Delhi',
    'Nctofdelhi': 'Delhi',
    'Chhatisgarh': 'Chhattisgarh'
})
gdf = gdf.dissolve(by='state_clean', as_index=False)

# Prepare your DNN results df
df['state_clean'] = df['state_name'].str.replace(" ", "").str.title()

# Aggregate dominant predicted category per state
dominant_pred = df.groupby('state_clean')['dnn_predicted_category'] \
    .agg(lambda x: x.value_counts().idxmax()).reset_index()

label_map = {0: 'Minimal engagement', 1: 'Moderate engagement', 2: 'High engagement'}
dominant_pred['category_label'] = dominant_pred['dnn_predicted_category'].map(label_map)

# Merge geo and prediction data
merged = gdf.merge(dominant_pred, on='state_clean', how='left')

# Manual fixes for Jammu & Kashmir (yellow)
jk_mask = merged['state_clean'].str.lower() == 'jammuandkashmir'
if jk_mask.sum() == 0:
    jk_geo = gdf[gdf['state_clean'].str.lower() == 'jammuandkashmir'].copy()
    if not jk_geo.empty:

```

```

    jk_geo['dnn_predicted_category'] = 0
    jk_geo['category_label'] = 'Minimal engagement'
    merged = pd.concat([merged, jk_geo], ignore_index=True)
    print("✅ Jammu & Kashmir added manually.")
else:
    merged.loc[jk_mask & merged['category_label'].isna(), 'dnn_predicted_category'] = 0
    merged.loc[jk_mask & merged['category_label'].isna(), 'category_label'] = 'Minimal
engagement'
    print("✅ Jammu & Kashmir category fixed manually.")

# --- Manually add UTs with their dominant categories ---

# Prepare a DataFrame of these UTs with their categories
manual_ut_data = pd.DataFrame({
    'state_clean': ['Dadraandnagarhaveli', 'Damananddiu', 'Andamanandnicobar'],
    'dnn_predicted_category': [1, 2, 2], # Moderate, High, High
    'category_label': ['Moderate engagement', 'High engagement', 'High engagement']
})

# Extract these rows from gdf to get geometries
ut_geo = gdf[gdf['state_clean'].isin(manual_ut_data['state_clean'])].copy()

# Merge manual categories into geometries
ut_geo = ut_geo.merge(manual_ut_data, on='state_clean', how='left')

# Drop old rows from merged (if present)
merged = merged[~merged['state_clean'].isin(manual_ut_data['state_clean'])]

# Append manually fixed UT rows
merged = pd.concat([merged, ut_geo], ignore_index=True)

# Map category labels to codes for coloring
category_code = {'Minimal engagement': 0, 'Moderate engagement': 1, 'High engagement': 2}
merged['category_code'] = merged['category_label'].map(category_code)

# Define colormap: 0-yellow, 1-light green, 2-dark green
cmap = ListedColormap(['#FFD700', '#90EE90', '#006400'])

# Plot
fig, ax = plt.subplots(figsize=(16, 14))
merged.plot(
    column='category_code',
    cmap=cmap,
    linewidth=0.5,

```

```

    ax=ax,
    edgecolor='black',
    legend=True,
    categorical=True,
    missing_kwds={"color": "lightgrey", "label": "No data"}
)

# Add labels
for idx, row in merged.iterrows():
    point = row['geometry'].representative_point()
    label = f'{row["state_clean"]}\n{row["category_label"] if pd.notnull(row["category_label"])
else "N/A"}'
    ax.text(point.x, point.y, label, fontsize=6.5, ha='center', va='center', color='black')

ax.set_title("Dominant Elderly Economic Participation Category by State (DNN Model)",
    fontsize=16)
ax.axis('off')
plt.tight_layout()
plt.show()

```

Step 323

```

import os

# Define output path and filename
output_dir = r"C:\Users\ASUS\OneDrive\Desktop\PHD Business admin\Analysis\DBA result
images"
os.makedirs(output_dir, exist_ok=True)
output_file = os.path.join(output_dir, "Elderly_Economic_Participation_by_State.png")

# Plot with legend labels
fig, ax = plt.subplots(figsize=(16, 14))
merged.plot(
    column='category_code',
    cmap=cmap,
    linewidth=0.5,
    ax=ax,
    edgecolor='black',
    categorical=True,
    missing_kwds={"color": "lightgrey", "label": "No data"},
    legend=False # disable automatic legend, we create manual below
)

# Add labels to map
for idx, row in merged.iterrows():

```

```

    point = row['geometry'].representative_point()
    label = f'{row["state_clean"]}\n{row["category_label"] if pd.notnull(row["category_label"])
else "N/A"}'
    ax.text(point.x, point.y, label, fontsize=6.5, ha='center', va='center', color='black')

ax.set_title("Dominant Elderly Economic Participation Category by State (DNN Model)",
fontsize=16)
ax.axis('off')

# Create custom legend with full labels
import matplotlib.patches as mpatches

legend_handles = [
    mpatches.Patch(color='#FFD700', label='Minimal engagement'),
    mpatches.Patch(color='#90EE90', label='Moderate engagement'),
    mpatches.Patch(color='#006400', label='High engagement'),
    mpatches.Patch(color='lightgrey', label='No data')
]

ax.legend(handles=legend_handles, loc='lower left', fontsize=12, frameon=False)

plt.tight_layout()

# Save figure
plt.savefig(output_file, dpi=300)
print(f"✅ Map saved to: {output_file}")

plt.show()

```

REFERENCES

- 1) Dwivedi I, Dr. Aditya Raj. Rethinking Population Aging in India: A Narrative Synthesis of Emerging Challenges. *Int J Res Soc Sci Humanit.* 2026;7(3):12–21.
- 2) Perianayagam A, Bloom D, Lee J, Parasuraman S, Sekher TV, Mohanty SK, Chattopadhyay A, Govil D, Pedgaonkar S, Gupta S, Agarwal A, Posture A, Weerman A, Pramanik S. Cohort Profile: The Longitudinal Aging Study in India (LASI). *Int J Epidemiol.* 2022;51(4):e167–e176.
- 3) Lai J, Eales J, Lariviere M, Boger J, Fast J. Barriers and facilitators of older workers' abilities to obtain and maintain employment: a scoping review. *J Aging Res.* 2025;2025:5609589.
- 4) Rahayuwati L, Arsyad SS, Ekawati R, Dawam M, Rahmadewi R, Nurhayati S, Fahmi I, Fauziah SR. Factors that influence the high number of elderly people working in the informal sector. *J Multidiscip Healthc.* 2024;17:1827–1837.
- 5) Chattopadhyay A, Khan J, Bloom DE, Sinha D, Nayak I, Gupta S, Lee J, Perianayagam A. Insights into labor force participation among older adults: Evidence from the Longitudinal Aging Study in India. *J Popul Aging.* 2022;15(1):39–59.
- 6) Singh PK, Singh L, Dubey R, Singh S, Mehrotra R. Socioeconomic determinants of chronic health diseases among older Indian adults: a nationally representative cross-sectional multilevel study. *BMJ Open.* 2019;9(9):e028426.
- 7) Singh PK, Singh L, Dubey R, Singh S, Mehrotra R. Socioeconomic determinants of chronic health diseases among older Indian adults: a nationally representative cross-sectional multilevel study. *BMJ Open.* 2019;9(9):e028426.
- 8) Chowdhury P, Mohanty I, Singh A, Niyonsenga T. Informal sector employment and the health outcomes of older workers in India. *PLoS One.* 2023;18(2):e0266576.
- 9) Chowdhury P, Goli S. Informal employment and high burden of out-of-pocket healthcare payments among older workers: evidence from the Longitudinal Aging Study in India. *Health Policy Plan.* 2025;40(2):123–139.

- 10) Sahoo H, Govil D, James KS, Prasad RD. Health issues, healthcare utilization, and health care expenditure among elderly in India: thematic review of literature. *Aging Health Res.* 2021;1.
- 11) Lal D, Virk A, Goel A, Goel S, Lal KK, Garg S, Bharti B. Healthcare access and utilisation trends among the elderly in India: Evidence from the LASI Wave-1 survey. *Dialogues Health.* 2025;7:100232.
- 12) World Health Organization. *Proposed working definition of an older person in Africa for the WHO African Region.* Geneva: WHO; 2002.
- 13) United Nations Department of Economic and Social Affairs Population Division. *World population prospects 2024: summary of results.* New York: United Nations; 2024.
- 14) Samuel Tillery, Mark Msrnik. Executive summary. In: *Global aging 2023.* New York: S&P Global; 2023 Jan.
- 15) National Statistical Office. *Elderly in India 2016.* New Delhi: Ministry of Statistics and Programme Implementation; 2016. p. 1–2.
- 16) Technical Group on Population Projections. *Population Projections for India and States: 2011–2036.* Report No. M3. New Delhi: Ministry of Health and Family Welfare; 2020 Jul.
- 17) Press Information Bureau. Elderly in India. New Delhi: *Ministry of Statistics and Programme Implementation*; 2025 Oct 14.
- 18) Social and Policy Research Foundation. India's shifting demographics and the challenge of elderly care. New Delhi: *SPRF*; 2025 Sep 10.
- 19) R Vaidyanathan. Declining joint family and emerging crisis in old age security. *International Journal of Indian Culture and Business Management.* 2007;1(1–2).
- 20) Organisation for Economic Co-operation and Development. *The silver economy: aging and economic opportunities.* Paris: OECD Publishing; 2018.
- 21) Ng R, Indran N. Role-based framing of older adults linked to decreased ageism over 210 years: evidence from a 600-million-word historical corpus. *Gerontologist.* 2022;62(4):589-97.

- 22) W Andrew Achenbaum, Vern L Bengtson. Re-engaging the disengagement theory of aging: on the history and assessment of theory development in gerontology. *Gerontologist*. 1994;34(6):756–63.
- 23) Robert C Atchley. A continuity theory of normal aging. *Gerontologist*. 1989;29(2):183–90.
- 24) Laura L Carstensen. Socioemotional selectivity theory: the role of perceived endings in human motivation. *Gerontologist*. 2021;61(8):1188–96.
- 25) Elaine Wethington. An overview of the life course perspective: implications for health and nutrition. *J Nutr Educ Behav*. 2005;37(3):115–20.
- 26) M Moeller, William H O'Brien. Integrating acceptance and commitment therapy with selective optimization with compensation to predict wellbeing. *Clin Gerontol*. 2024:1–13.
- 27) Eero Lahelma, Pekka Martikainen, Mikko Laaksonen, Anu Aittomäki. Pathways between socioeconomic determinants of health. *J Epidemiol Community Health*. 2004;58(4):327–32.
- 28) Zainab Ismail, Wan Ibrahim Wan Ahmad, Salasiah Hanin Hamjah, I Komang Astina. The impact of population aging: a review. *Iran J Pub. Health*. 2021;50(12):2451–60.
- 29) Klimczuk A. Supporting the development of gerontechnology as part of silver economy building. *Ad Alta J Interdiscip Res*. 2012;2(2):52–6.
- 30) Jan van Hoof, Jacek K Kazak, Joanna M Perek-Białas, Sebastiaan T M Peek. The challenges of urban aging: making cities age-friendly in Europe. *Int J Environ Res Public Health*. 2018;15(11):2473.
- 31) Maryam Dehi Aroogh, Fatemeh Mohammadi Shahboulaghi. Social participation of older adults: a concept analysis. *Int J Community Based Nurs Midwifery*. 2020;8(1):55–72.
- 32) Dong Lin, Jin Cui. Transport and mobility needs for an aging society from a policy perspective: review and implications. *Int J Environ Res Public Health*. 2021;18(22):11802.
- 33) Reddy AB, Reddy PH. Labour force participation of the elderly in India: a study based on NSS data. *Indian J Labour Econ*. 2016;59(4):631–652.

- 34) Chowdhury P, Goli S. Informal employment and high burden of out-of-pocket healthcare payments among older workers: evidence from the Longitudinal Aging Study in India. *Health Policy Plan.* 2025;40(2):123-39.
- 35) World Health Organization. *Active aging: a policy framework.* Geneva: WHO; 2002. (Active aging series). WHO/NMH/NPH/02.8.
- 36) V Rogelj, D Bogataj. Social infrastructure of silver economy: literature review and research agenda. *IFAC Pap Online.* 2018;52(13):2680–5.
- 37) M Cristea, G G Noja, P Ștefea, A L Sala. The impact of population aging and public health support on EU labor markets. *Int J Environ Res Public Health.* 2020;17(4):1439.
- 38) India State-Level Disease Burden Initiative Mental Disorders Collaborators. The burden of mental disorders across the states of India: the Global Burden of Disease Study 1990–2017. *Lancet Psychiatry.* 2020 Feb;7(2):148–161.
- 39) Gadapani Pathak B, Mazumder S, Bin Nisar Y, Bhatt A, Singh M, Madhur T, Sandøy IF. Determinants and experiences of care-seeking for childhood pneumonia in a rural Indian setting: a mixed-methods study. *Health Expect.* 2025 Apr;28(2):e70263.
- 40) Kalaivanan RC, Rahul P, Manjunatha N, Kumar CN, Sivakumar PT, Math SB. Telemedicine in geriatric psychiatry: relevance in India. *Indian J Psychol Med.* 2021;43(5 Suppl):S121–S127. doi:10.1177/02537176211033007.
- 41) Sahoo H, Govil D, James KS, Prasad RD. Health issues, healthcare utilization, and health care expenditure among elderly in India: a thematic review of literature. *Aging Health Res.* 2021;1:100011.
- 42) Life Insurance Corporation of India. *Pradhan Mantri Vaya Vandana Yojana (PMVVY).* Mumbai: LIC of India; 2023.
- 43) Ministry of Social Justice and Empowerment, Government of India. *Integrated Programme for Older Persons (IPOP).* New Delhi: Government of India; 2023.
- 44) Ministry of Social Justice and Empowerment, Government of India. *Rashtriya Vayoshri Yojana (RVY).* New Delhi: Government of India; 2023

- 45) Ministry of Health and Family Welfare, Government of India. *National Programme for Health Care of the Elderly (NPHCE): operational guidelines*. New Delhi: Government of India; 2011.
- 46) A K Goswami, S Ramadass, M Kalaivani, B Nongkynrih, Sunil Kant, S K Gupta. Awareness and utilization of social welfare schemes by elderly persons residing in an urban resettlement colony of Delhi. *J Family Med Prim Care*. 2019;8(3):960–5.
- 47) M Sawhney. The role of non-governmental organizations for the welfare of the elderly: the case of HelpAge India. *J Aging Soc Policy*. 2003;15(2–3):179–91.
- 48) Y Yang. The role of NGOs in enabling elderly activity and care in the community: a case study of Silver Wings in South Korea. *J Cross Cult Gerontol*. 2018;33(2):217–28.
- 49) IIPS (International Institute for Population Sciences), National Programme for Health Care of Elderly (NPHCE), MoHFW, HSPH (Harvard T. H. Chan School of Public Health) and USC (University of Southern California). *Longitudinal Aging Study in India (LASI), Wave 1 2017–18. India National Report*. Mumbai, India: International Institute for Population Sciences, 2020.
- 50) Deepak Lal, A Virk, A Goel, S Goel, K K Lal, S Garg, B Bharti. Healthcare access and utilisation trends among the elderly in India: evidence from the LASI wave-1 survey. *Dialogues Health*. 2025;7:100232.
- 51) L Wang, L Yang, X Di, X Dai. Family support, multidimensional health, and living satisfaction among the elderly: a case from Shaanxi Province, China. *Int J Environ Res Public Health*. 2020;17(22):8434.
- 52) David E Bloom, T V Sekher, Jinkook Lee. Longitudinal aging study in India (LASI): new data resources for addressing aging in India. *Nat Aging*. 2021;1:1070–2.
- 53) M Kumar, N Kumari, S Chanda, et al. Multimorbidity combinations and their association with functional disabilities among Indian older adults: evidence from longitudinal aging study in India (LASI). *BMJ Open*. 2023;13:e062554.
- 54) C Q Zhang, P K Chung, R Zhang, B Schüz. Socioeconomic inequalities in older adults' health: the roles of neighborhood and individual-level psychosocial and behavioral resources. *Front. Public Health*. 2019;7:318.

- 55) C Bolster-Foucault, I Vedel, G Busa, G Hacker, N Sourial, A Quesnel-Vallée. Social inequity in aging in place among older adults in Organisation for Economic Cooperation and Development countries: a mixed studies systematic review. *Age Aging*. 2024;53(8):66.
- 56) Rechel B, Doyle Y, Grundy E, McKee M. How can health systems respond to population aging? *Health Syst Transit*. 2009;11(3):1–152.
- 57) A M Seifert, C Batsis, A Smith. Digital inequalities in older adults' use of eHealth technologies: a systematic review. *J Med Internet Res*. 2020;22(11):e20334.
- 58) National Sample Survey Office (NSSO). *Employment and Unemployment Situation in India*. NSS 68th Round. New Delhi: Ministry of Statistics and Programme Implementation; 2014.
- 59) Ministry of Statistics and Programme Implementation (MoSPI). *Periodic Labour Force Survey (PLFS) Annual Report 2022–23*. New Delhi: Government of India; 2023.
- 60) Börsch-Supan A. Labor market effects of population aging. *Labour*. 2003;17 Suppl 1:5-44.
- 61) Coile CC, Milligan KS, Wise DA. Introduction and summary to "Social security programs and retirement around the world: working longer". In: Coile CC, Milligan KS, Wise DA, editors. *Social Security Programs and Retirement around the World: Working Longer*. Chicago: University of Chicago Press; 2019. p. 1-32. (NBER Working Paper No. 24584; 2018).
- 62) Desai S, Vanneman R. India Human Development Survey-II (IHDS-II), 2011-12. New Delhi: *National Council of Applied Economic Research*; 2015. (ICPSR36151-v6).
- 63) Alam M, Mazumdar S. Labour force participation among older adults in India: evidence from LASI Wave 1. *N-IUSSP*. 2023 Jan 8.
- 64) Ministry of Statistics and Programme Implementation. Periodic Labour Force Survey (PLFS) Annual Report July 2022–June 2023. New Delhi: *National Statistical Office, Government of India*; 2024.
- 65) Gorman M, Jones S, Turner J. Older people, mobility and transport in low- and middle-income countries: a review of the research. London: *HelpAge International*; 2019.

- 66) International Labour Organization. *Women and men in the informal economy: a statistical picture*. 3rd ed. Geneva: ILO; 2018.
- 67) Deshpande A, Sharma S. Disadvantage and discrimination in self-employment: caste gaps in earnings in Indian small businesses. *Small Bus Econ*. 2015;46(2):325–46.
- 68) Berkman LF, Sekher TV, Capistrant BD, Zheng Y, Shih RA, Kim JK. Social networks, family, and caregiving among older adults in India. In: Smith JP, Majmundar M, editors. *Aging in Asia: Findings from New and Emerging Data Initiatives*. Washington (DC): National Academies Press; 2012. p. 261–292.
- 69) Lloyd-Sherlock P, Agrawal S, Minicuci N. Widowhood, socio-economic status, health and wellbeing in low and middle-income countries. *J Dev Stud*. 2015;51(10):1374–1388.
- 70) Becker GS. *Human capital: A theoretical and empirical analysis, with special reference to education*. 3rd ed. Chicago: University of Chicago Press; 1993.
- 71) Visaria A, Dommaraju P. Productive aging in India. *Soc Sci Med*. 2019;229:14–21.
- 72) Brigola AG, Alexandre TDS, Inouye K, Yassuda MS, Pavarini SCI, Mioshi E. Limited formal education is strongly associated with lower cognitive status, functional disability & frailty status in older adults. *Dement Neuropsychol*. 2019;13(2):216–224.
- 73) Dhar A. Workforce participation among the elderly in India: struggling for economic security. *Indian J Labour Econ*. 2014;57(3):395–410.
- 74) David M. Cutler, Steven J. Meara, Seth Richards-Shubik. Health and the retirement decision in the United States. *J Health Econ*. 2013;32(2):315–327.
- 75) Arokiasamy P, Bloom DE, Lee J, Feeney K, Ozolins M. Longitudinal Aging Study in India: vision, design, implementation, and preliminary findings. *Aging Clin Exp Res*. 2017;29(6):1245–1248.
- 76) Pengpid S, Peltzer K. Work participation and mental and physical health outcomes among a national community-dwelling sample of older adult men and women in India. *J Hum Behav Soc Environ*. 2023;33(3):407–18.

- 77) Ilmarinen J, Tuomi K, Klockars M. Changes in the work ability of active employees over an 11-year period. *Scand J Work Environ Health*. 1997;23 Suppl 1:49-57.
- 78) Somnath Chatterji, Jürgen Rehm, John Beard, et al. Health, functioning, and disability in older adults—present status and future implications. *Lancet*. 2015;385(9967):563–575. y in older adults—present status and future implications. *Lancet*. 2015;385(9967):563–575.
- 79) World Health Organization. *Tracking universal health coverage: 2023 global monitoring report*. Geneva: WHO; 2023.
- 80) World Bank. *Out-of-pocket health expenditure and financial protection in LMICs*. Washington DC: World Bank; 2022.
- 81) National Health Authority. *National Health Accounts Estimates for India (2019–20)*. New Delhi: MoHFW; 2022.
- 82) Ministry of Health and Family Welfare. *National Health Accounts / Economic Survey data on health expenditure*. Government of India; 2023.
- 83) Ayushman Bharat Pradhan Mantri Jan Arogya Yojana. *Operational guidelines and coverage details*. National Health Authority; 2021.
- 84) AS, Karan A, et al. *Impact of publicly financed health insurance schemes in India: a systematic review*. PLoS One. 2017;12(2):e0170996.
- 85) Rao KD, Peters DH, Bandeen-Roche K. Towards patient-centered health services in India—a scale to measure patient perceptions of quality. *Health Affairs*. 2006;25(6):w414–w424.
- 86) Peltzer K. Patient experiences and health system responsiveness in South Africa. *BMC Health Services Research*. 2009;9:117.
- 87) Arokiasamy P, Uttamacharya Bloom, D. E., Canning, D., & Fink, G. (2010). *Implications of population aging for economic growth*. *Oxford Review of Economic Policy*, 26(4), 583–612.
- 88) National Health Authority. *Ayushman Bharat PM-JAY: Policy brief and implementation challenges*. New Delhi; 2021.

- 89) Prinja S, Chauhan U, Jain K, Biritwum R, Yawson A, Wu F, Guo Y, Maximova T, Espinoza B, Rodríguez A, Afshar S, Peltzer K, Kowal P. The impact of multimorbidity on adult physical and mental health in low- and middle-income countries: what does the study on global aging and adult health (SAGE) reveal? *BMC Medicine*. 2015;13:178.
- 90) Sitzia J, Wood N. Patient satisfaction: a review of issues and concepts. *Social Science & Medicine*. 1997;45(12):1829–1843.
- 91) Desai Sonalde, Dubey Amaresh, Joshi Brij Lal, Sen Mitali, Shariff Abusaleh, Vanneman Reeve. Human development in India: challenges for a society in transition. New Delhi: *Oxford University Press*; 2010.
- 92) Rajan S Irudaya, Mishra U S, Sarma P Sankara. India's elderly: burden or challenge? New Delhi: *Sage Publications*; 1999.
- 93) Das Aabhas Kumar, Barua Anupam. Regional inequalities, economic growth and liberalisation: a study of the Indian economy. *Journal of Development Studies*. 1996;32(3):364–390.
- 94) Alam Moneer, Mazumdar Sarmila. Aging, economic participation and health in India. New Delhi: *Academic Foundation*; 2011.
- 95) Coile CC, Milligan K, Wise DA. Health shocks and retirement decisions: evidence from international comparisons. *Health Econ*. 2016;25(S2):39–55.
- 96) McGarry K. Health and retirement: do changes in health affect retirement expectations? *J Hum Resour*. 2004;39(3):624–648.
- 97) Banks J, Blundell R, Bozio A, Emmerson C. Disability, health and retirement in the United Kingdom. *J Eur Econ Assoc*. 2011;9(2):241–266.
- 98) Wahrendorf M, Akinwale B, Landy R, Matthews K, Blane D. Who in Europe works beyond the state pension age and under which conditions? Results from SHARE. *J Popul Aging*. 2017;10(3):269–285.
- 99) Zaninotto P, Falaschetti E, Sacker A. Age trajectories of quality of life among older adults: results from the English Longitudinal Study of Aging. *Qual Life Res*. 2009;18(10):1301–1309.

- 100) Obermeyer Z, Emanuel EJ. Predicting the future—big data, machine learning, and clinical medicine. *N Engl J Med*. 2016;375(13):1216–1219.
- 101) Dey S, Nambiar D, Lakshmi JK, Sheikh K, Reddy KS. Health of the elderly in India: challenges of access and affordability. In: Smith JP, Majmundar M, editors. *Aging in Asia: Findings from New and Emerging Data Initiatives*. Washington (DC): National Academies Press; 2012.
- 102) Institute of Economic Growth (IEG), National Council of Applied Economic Research (NCAER). *India Human Development Survey (IHDS)*. New Delhi: IEG and NCAER; 2015.
- 103) Kuhn R, Everett B, Silvey R. The effects of children’s migration on elderly kin’s health: a counterfactual approach. *Demography*. 2011;48(1):183–209.
- 104) Gruber J, Wise DA. *Social security programs and retirement around the world: the relationship to youth employment*. Chicago: University of Chicago Press; 2010.
- 105) Coile CC, Milligan K, Wise DA. Health shocks and retirement decisions: evidence from international comparisons. *Health Econ*. 2016;25(S2):39–55.
- 106) Mehrotra S, Parida JK, Sinha S, Gandhi A. Explaining employment trends in the Indian economy: 1993–94 to 2011–12. *Econ Polit Wkly*. 2014;49(32):49–54.