COMILLAS
UNIVERSIDAD PONTIFICIA

ICAI

MÁSTER EN INGENIERÍA INDUSTRIAL

TRABAJO FIN DE MÁSTER

ANALYSIS OF LOCAL INVESTMENTS IN CLIMATE CHANGE ADAPTATION IN NEW YORK STATE

Autor: María Granados Santos

Director: Dra. María del Mar Cledera Castro

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Analysis of local investments in climate change adaptation in New York State

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2023/2024 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

Fdo.: María Granados Santos Fecha: 08/ 07/ 2024

[illegible]

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

[illegible]

Fdo.: Dra. María del Mar Cledera Castro Fecha: 08/ 07/ 2024

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

MÁSTER EN INGENIERÍA INDUSTRIAL

TRABAJO FIN DE MÁSTER

ANALYSIS OF LOCAL INVESTMENTS IN CLIMATE CHANGE ADAPTATION IN NEW YORK STATE

Autor: María Granados Santos

Director: Dra. María del Mar Cledera Castro

Madrid

Agradecimientos

En primer lugar, quiero expresar mi sincero agradecimiento a mi tutora, Dra. María del Mar Cledera Castro, y a mi profesora Dra. Allison Coffey Reilly por su ayuda constante.

A mis amigos, tanto los de siempre como los que he conocido en estos dos años, gracias por las risas y el apoyo en todo momento.

De manera especial, a mi familia: padres y hermanos, gracias por ser ejemplo de esfuerzo y dedicación, y por nunca soltarme la mano.

ANÁLISIS DE LAS INVERSIONES LOCALES EN ADAPTACIÓN AL CAMBIO CLIMÁTICO EN EL ESTADO DE NUEVA YORK

Autor: Granados Santos, María.
Director: Cledera Castro, María del Mar.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

El presente trabajo analiza la adaptación al cambio climático que realizan los municipios del estado de Nueva York, enfocándose tanto en los costes de los proyectos, como en el análisis sociodemográfico y climatológico de las comunidades. Además, se incluyen propuesta de iniciativas para los municipios del estado.

Palabras clave: Cambio climático, Nueva York, Climate Smart Communities, Análisis sociodemográfico

1. Introducción

El cambio climático es un problema muy importante que afecta a la vida terrestre y su principal impacto es el aumento de temperaturas. En particular, Estados Unidos se calentó un 68% más rápido que el promedio mundial en los últimos años [1].

Dada la gran importancia que tiene el cambio climático, los diferentes estados que conforman el país están dedicando ingentes cantidades de recursos a amortiguar sus efectos. En especial, el estado de Nueva York está implementando medidas como la promoción de la limpieza ecológica, el aumento de la eficiencia energética y la mejora de reciclaje para abordar los efectos del cambio climático [2]. Además, existe un programa denominado Climate Smart Communities, en el que el gobierno otorga ayudas a los municipios para desarrollas proyectos de adaptación al cambio climático [3].

2. Definición del proyecto

Este proyecto analiza de forma exhaustiva las comunidades del estado de Nueva York (a excepción de las que conforman la ciudad de Nueva York) que están realizando proyectos de cambio climático, así como las características de dichos proyectos. Para ello, se emplean las respuestas de las encuestas publicadas por el gobierno del estado para las comunidades que forman parte del programa Climate Smart Communities en el año 2022.

Este estudio busca determinar si las comunidades intervienen más en proyectos de infraestructuras o los ambientales (proteger sistemas naturales y plantar árboles). La finalidad es comprobar que, aunque se hable de proyectos de cambio climático, estos no siempre implican simplemente plantar árboles.

Además del análisis mencionado, se pretender comprender las características demográficas y climatológicas de los municipios que forman parte del programa y de los que no, así como identificar quiénes están trabajando en la adaptación al cambio climático y quiénes no. También se analiza el coste de los proyectos en función de las variables demográficas y climatológicas, junto con las respuestas de las encuestas, donde se determinan los motivos del proyecto o la financiación de este mismo.

Finalmente, se ha realizado una propuesta de iniciativas para las comunidades, sugiriendo que se inspiren en los proyectos de comunidades con similitudes demográficas y climatológicas.

3. Descripción del modelo

Para realizar este proyecto, se ha empleado Python, un lenguaje de programación que destaca por su extensa biblioteca para el análisis, optimización y representación de datos y resultados. Además, se ha empleado diferentes herramientas estadísticas, como la correlación y los percentiles, para determinar si las comunidades invierten más en proyectos estructurales o proyectos ambientales.

Se han construido modelos probit y modelos random forest para analizar las variables más significativas en la predicción de los municipios que forman parte del programa y en la implicación de estos en la adaptación al cambio climático. Un modelo muy similar a estos, pero donde difiere el tipo de datos a analizar, es el modelo de regresión lineal de los mínimo cuadrados ordinarios, empleado para predecir los costes de los proyectos que las comunidades están llevando a cabo.

Finalmente, se ha utilizado el método clustering para agrupar las comunidades en función de las características más importantes, definidas en los modelos probit y random forest. Esto permite recomendar proyectos a comunidades que no están realizando proyectos y también a la que sí, para que puedan inspirarse y llevar a cabo más iniciativas.

4. Resultados

En primer lugar, se incluye en la Ilustración 1,la muestra de las comunidades que forman parte del programa, destacando la parte sureste del estado de Nueva York, en especial las comunidades Nassau y Suffolk, ya que tanto los propios condados como las ciudades y pueblos están intentando poner freno al cambio climático.

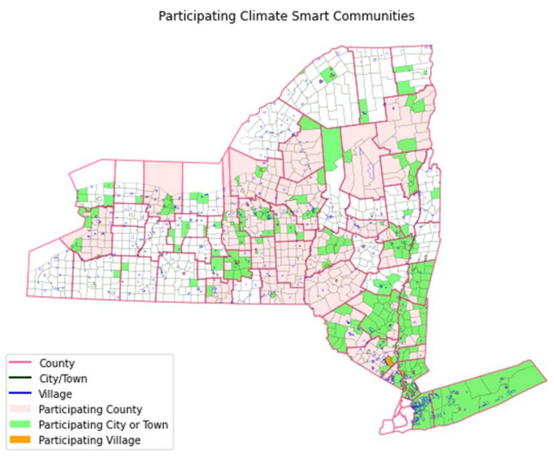

Participating Climate Smart Communities
County
City/Town
Village
Participating County
Participating City or Town
Participating Village

Ilustración 1. Participación de las comunidades en el programa Climate Smart Communities

Los proyectos estructurales y ambientales no están correlacionadas entre sí. Además, se ha comprobado que las comunidades invierten más en proyectos estructurales que en ambientales.

El modelo Random Forest es más preciso que el modelo Probit para el estudio de la participación y grado de implicación de las comunidades, destacando por predecir correctamente las comunidades que no están en el programa o que no realizan proyectos. La variable más relevante en el caso de la participación es el porcentaje de personas licenciadas; sin embargo, en el análisis de la realización de proyectos, el factor más significativo es el porcentaje de personas en situación de pobreza.

El análisis de los costes de los proyectos muestra que las variables más significativas en el modelo son los porcentajes de inversión financiada a nivel federal, estatal y local, lo cual es lógico dado que un coste elevado es difícil de afrontar sin ayuda. Además, se ha demostrado que las inundaciones, un fenómeno muy común en el estado de Nueva York, son la principal causa de la realización de estos proyectos.

Se ha obtenido un número alto de propuesta de iniciativas, no solo comparando los municipios a nivel global, sino para cada caso individual. Por ejemplo, en el análisis global, se puede observar que hay comunidades con más de 39 recomendaciones, aunque este número disminuye en los niveles de condado (3), “cities and towns” (21) y “villages” (22).

5. Conclusiones

Aunque el estado de Nueva York se ha comprometido a mejorar el cambio climático, es evidente que todavía falta mucho por hacer, al igual que en otras muchas zonas del mundo.

Este proyecto define, desarrolla y contrasta una metodología que permite recomendar proyectos en base a los que se han realizado en otras comunidades similares por sus características demográficas, económicas y climatológicas.

6. Referencias

[1] Univision, “EEUU se calentó un 68% más rápido que el resto del mundo en los últimos 50 años”, Univision. https://www.univision.com/noticias/medio-ambiente/informe-federal-cambio-climatico-evaluacion-nacional-del-clima

[2] “GreenNY: State Purchasing and Operations,” Office of General Services. https://ogs.ny.gov/greenny

[3] “Climate Smart Communities.” https://climatesmart.ny.gov/

ANALYSIS OF LOCAL INVESTMENTS IN CLIMATE CHANGE ADAPTATION IN NEW YORK STATE

Autor: Granados Santos, María.
Director: Cledera Castro, María del Mar.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

ABSTRACT

This study examines the climate change adaptation strategies implemented by municipalities in the state of New York, with a particular focus on the financial implications of these projects and the sociodemographic and climatological characteristics of the communities involved. Furthermore, proposals for initiatives for municipalities in the state are included.

Keywords: Climate change, New York, Climate Smart Communities, Sociodemographic analysis

1. Introduction

Climate change represents a significant challenge to life on Earth, with the most pronounced impact being the rise in temperatures. In particular, the United States has exhibited a warming trend of 68% faster than the global average in recent years [1].

Given the profound significance of climate change, the various states that comprise the country are allocating a considerable number of resources to mitigate its effects. In particular, the state of New York is implementing measures such as the promotion of green cleaning, the increase in energy efficiency, and the improvement of recycling in order to address the effects of climate change [2]. Additionally, a program known as Climate Smart Communities provides financial assistance to local municipalities to develop climate change adaptation strategies [3].

2. Project definition

This project employs a comprehensive analysis of the communities in New York State (except for those that comprise New York City) that are engaged in climate change projects, as well as the characteristics of those projects. This is accomplished through the utilization of survey responses published by the state government for communities that are part of the Climate Smart Communities program in 2022.

This study endeavors to ascertain whether communities are more involved in infrastructure projects or environmental projects (the protection of natural systems and the planting of trees). The objective is to ascertain whether, despite the prevalence of discussions about climate change projects, they do not always entail the simple planting of trees.

In addition to the aforementioned analysis, the aim is to comprehend the demographic and climatological characteristics of the municipalities that are part of the program and those that are not, as well as to identify those engaged in climate change adaptation and those that are not. The cost of the projects is also analyzed according to the demographic and climatological variables, together with the responses from the surveys, where the reasons for the project or the financing of the project are determined.

Ultimately, a series of recommendations for the communities have been put forth, suggesting that they draw inspiration from the projects of communities with similar demographic and climatological characteristics.

3. Model description

The Python programming language, renowned for its comprehensive array of tools for data analysis, optimization, and representation, was employed in the execution of this project. Furthermore, statistical tools such as correlation and percentiles have been employed to ascertain whether communities exhibit a greater propensity to invest in structural projects or environmental projects.

Probit and random forest models were constructed to analyze the most significant variables in the prediction of municipalities that are part of the program and their involvement in climate change adaptation. A similar model, differing only in the type of data to be analyzed, is the ordinary least squares linear regression model, which is used to predict the costs of the projects that the communities are carrying out.

Finally, the clustering method was employed to group the communities according to the most significant characteristics identified in the probit and random forest models. This methodology allows for the recommendation of projects to communities that are not currently undertaking any projects, as well as to those that are, thereby providing inspiration for the latter to initiate further initiatives.

4. Results

First, Illustration 1 includes a sample of the communities that are part of the program, highlighting the southeastern part of New York State, especially the communities of Nassau and Suffolk, as the counties themselves, as well as the cities and towns, are trying to curb climate change.

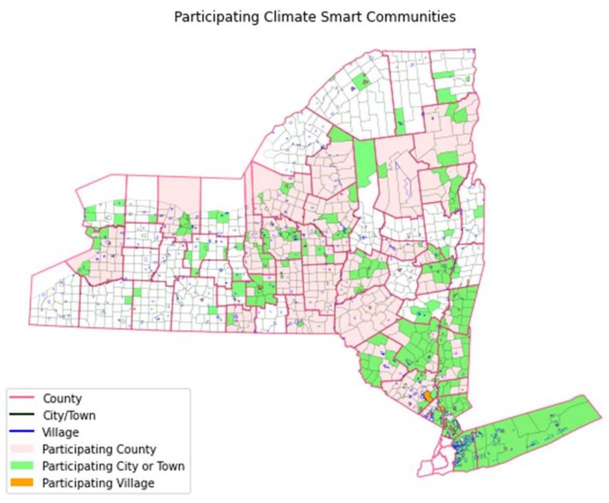

Participating Climate Smart Communities
County
City/Town
Village
Participating County
Participating City or Town
Participating Village

Illustration 1. Participating Climate Smart Communities

It can be demonstrated that there is no correlation between structural and environmental projects. Furthermore, empirical evidence indicates that communities tend to allocate a greater proportion of their resources towards structural projects than environmental projects.

The Random Forest model demonstrated greater accuracy than the Probit model in the study of participation and degree of involvement of communities. It correctly predicted those communities that were not included in the program or that did not carry out projects. The most relevant variable in the case of participation was the percentage of people licensed. However, in the analysis of project implementation, the most significant factor was the percentage of people living in poverty.

The analysis of project costs indicates that the most significant variables in the model are the percentages of investments financed at the federal, state, and local levels. This is logical given that a high cost is difficult to meet without assistance. Additionally, flooding, a prevalent phenomenon in New York State, has been identified as the primary cause for the implementation of these projects.

A high number of initiative proposals were obtained, not only by comparing the municipalities at the global level but also for each case. For instance, the global analysis reveals that certain communities have received more than 39 recommendations, although this number declines at the county (3), city and town (21), and village (22) levels.

5. Conclusions

Although New York State is dedicated to enhancing its efforts to combat climate change, it is evident that there is still much to be accomplished, as is the case in numerous other regions around the globe.

This project defines, develops, and contrasts a methodology for recommending projects based on the experiences of other similar communities with respect to their demographic, economic, and climatological characteristics.

6. Bibliography

[1] Univision, "EEUU se calentó un 68% más rápido que el resto del mundo en los últimos 50 años", Univision. https://www.univision.com/noticias/medio-ambiente/informe-federal-cambio-climatico-evaluacion-nacional-del-clima

[2] "GreenNY: State Purchasing and Operations," Office of General Services. https://ogs.ny.gov/greenny

[3] "Climate Smart Communities." https://climatesmart.ny.gov/

Table of Contents

Chapter 1. Introduction ***7***

1.1 State of the art 8

1.2 Motivation 9

1.3 Objectives 9

1.3.1 Sustainable Development Goals (SDG) *9*

1.4 Methodology 11

1.5 Files and Resources Used 12

Chapter 2. Contextualization ***16***

2.1 Algorithms 16

2.1.1 Socioeconomic Data *17*

2.1.2 Climate Smart Communities *19*

2.1.3 Survey *20*

2.1.4 Flood Risk *22*

2.1.5 Revenues *24*

2.1.6 Variables *25*

2.2 Geographic Distribution of Climate Smart Communities 30

Chapter 3. Structural or Environmental projects ***37***

3.1 Algorithm 38

3.2 Results 40

3.2.1 Structural projects *40*

3.2.2 Environmental projects *43*

3.3 Relation 46

Chapter 4. Degree of Community Participation and Involvement ***49***

4.1 Probit Model 49

4.2 Random Forest Model 51

4.3 Confusion matrix and accuracy 53

4.4 Participation model 53

4.4.1 Algorithm *54*

4.4.2 Results *56*

4.5 Involvement model 75
4.5.1 Algorithm 75
4.5.2 Results 77

Chapter 5. Economic Model 96
5.1 Algorithm 98
5.2 Results 100
5.2.1 Training Model 100
5.2.2 General Model 103

Chapter 6. Project Initiative 107
6.1 Algorithms 108
6.2 Results 110
6.2.1 Global 110
6.2.2 Counties 112
6.2.3 Cities and Towns 115
6.2.4 Villages 117

Chapter 7. Conclusions 120

Chapter 8. Bibliography 123

ANEXO I 126
Main Program 126
Functions 128
Demographic 128
Climate Smart Communities 133
Survey 134
MapNY 136
Flood Risk 137
Revenues 138
Structural or environmental projects 142
Visualization 144
Participation 152
Economic model 161
Proposal 164

List of Figures

Figure 1. Energy funding from Bipartisan Infrastructure Law and Inflation Reduction Act major funding theme, totaling $370 billion [9] 8
Figure 2. Municipal Geographic Levels Hierarchy. Source: Own elaboration 13
Figure 3. Map of New York State Counties[12] 14
Figure 4. Participating Climate Smart Communities 30
Figure 5. Participating Climate Smart Counties 31
Figure 6. Participating Climate Smart Cities and Towns 32
Figure 7. Direct and Indirect Participation Climate Smart Cities and Towns 33
Figure 8. Participating Climate Smart Villages 34
Figure 9. Direct and Indirect Participation Climate Smart Villages 35
Figure 10. Direct and Indirect Participation Climate Smart Villages (Second County) 36
Figure 11. Counties with structural projects 40
Figure 12. Counties with environmental projects 44
Figure 13. Counties with expensive projects 48
Figure 14. Random forest model [29] 52
Figure 15. Participating Communities: Probit Model Confusion Matrix 57
Figure 16. Participating Communities: Random Forest Model Confusion Matrix 59
Figure 17. Participating Counties: Probit Model Confusion Matrix 62
Figure 18. Participating Counties: Random Forest Model Confusion Matrix 64
Figure 19. Participating Cities and Towns: Probit Model Confusion Matrix 66
Figure 20. Participating Cities and Towns: Random Forest Model Confusion Matrix....... 68
Figure 21. Participating Villages: Probit Model Confusion Matrix 71
Figure 22. Participating Villages: Random Forest Model Confusion Matrix 73
Figure 23. Communities Conducting Projects: Probit Model Confusion Matrix 78
Figure 24. Communities Conducting Projects: Random Forest Model Confusion Matrix . 80
Figure 25. Counties Conducting Projects: Probit Model Confusion Matrix 83
Figure 26. Counties Conducting Projects: Random Forest Model Confusion Matrix 85
Figure 27. Cities and Towns Conducting Projects: Probit Model Confusion Matrix 87

Figure 28. Cities and Towns Conducting Projects: Random Forest Model Confusion Matrix 89
Figure 29. Villages Conducting Projects: Probit Model Confusion Matrix 92
Figure 30. Villages Conducting Projects: Random Forest Model Confusion Matrix 94
Figure 31. Residuals versus Fitted Values (Training Set) 101
Figure 32. Q-Q Plot of Residuals (Training Set) 101
Figure 33. Homoscedasticity (Training Set) 102
Figure 34. Residuals versus Fitted Values 104
Figure 35. Q-Q Plot of Residuals 104
Figure 36. Homoscedasticity 105
Figure 37. Coefficients and p-values for the General Model 106
Figure 38. Elbow method for all Communities 111
Figure 39. Elbow method for Counties 113
Figure 40. Elbow method for Cities and Towns 116
Figure 41. Elbow method for Villages 118

List of Tables

Table 1. Description of df_demographic variables 27
Table 2. Description of df_survey_yes variables 29
Table 3. Types of projects 37
Table 4. Top Counties by Structural Project Expenditure 42
Table 5. Counties with the Lowest Structural Project Expenditure 43
Table 6. Top Counties by Environmental Project Expenditure 45
Table 7. Counties with the Lowest Environmental Project Expenditure 46
Table 8. Top Counties by Project Expenditure 47
Table 9. Coefficients and p-values for the Global Participation Probit Model 58
Table 10. Importance of Variables for the Global Participation Random Forest Model 60
Table 11. Coefficients and p-values for the Counties Participation Probit Model 63
Table 12. Importance of Variables for the Counties Participation Random Forest Model . 65
Table 13. Coefficients and p-values for the Cities and Towns Participation Probit Model 67
Table 14. Importance of Variables for the Cities and Towns Participation Random Forest Model 69
Table 15. Coefficients and p-values for the Villages Participation Probit Model 72
Table 16. Importance of Variables for the Villages Participation Random Forest Model .. 74
Table 17. Coefficients and p-values for the Global Project Involvement Probit Model 79
Table 18. Importance of Variables for the Global Project Involvement Random Forest Model 81
Table 19. Coefficients and p-values for the Counties Project Involvement Probit Model.. 84
Table 20. Importance of Variables for the Counties Project Involvement Random Forest Model 86
Table 21. Coefficients and p-values for the Cities and Towns Project Involvement Probit Model 88
Table 22. Importance of Variables for the Cities and Towns Project Involvement Random Forest Model 90
Table 23. Coefficients and p-values for the Villages Project Involvement Probit Model .. 93

Table 24. Importance of Variables for the Villages Project Involvement Random Forest Model 95
Table 25. Key Variables for Global Initiative 111
Table 26. Recommendations for Community Initiatives 112
Table 27. Key Variables for County Initiative 113
Table 28. Recommendations for County Initiatives 115
Table 29. Key Variables for City and Town Initiative 115
Table 30. Recommendations for City and Town Initiatives 117
Table 31. Key Variables for Village Initiative 118
Table 32. Recommendations for Villages Initiatives 119

Chapter 1. INTRODUCTION

There is currently a high level of concern about climate change, which is generating a large number of alarming effects. The main impact is the rise in temperatures that has occurred in recent years. This has led to numerous other effects, including the increase in tropical cyclones and their intensities due to the warming of the ocean, the frequency of destructive storms and their intensity due to the evaporation of more humidity, the increase in the probability of uncontrolled fires, and the level of the ocean, among others [1].

Drought, food shortages, health problems, natural disasters, and the disappearance of species are the main consequences of those effects of climate change. The primary driver of climate deterioration is human activity. The routine operations of transportation, food and material production, energy generation, and energy consumption collectively result in the emission of a considerable quantity of greenhouse gases, which contribute to climate change. The Intergovernmental Panel on Climate Change (IPCC) report indicates that the United States has experienced a 68% faster warming trend than the global average over the past 50 years [2]. New York City is included in the list of the most polluting cities in the United States due to its high population density, extensive economic, commercial, and tourist activity, as well as infrastructure development [3].

However, the State of New York is implementing measures such as promoting green cleaning, increasing energy efficiency, and improving recycling to address the effects of climate change [4]. To adapt to climate change and reduce greenhouse gas emissions, the state government has established a program called Climate Smart Communities. This program assists municipal governments in developing sustainable and balanced action plans with the help of grants and free assistance[5].

1.1 STATE OF THE ART

The United States, the country with the most advanced economy in the world, is devoting a significant portion of its public resources to combating climate change and its consequences.

In 2023, the U.S. Environmental Protection Agency (EPA) allocated over ten billion dollars for this purpose, in addition to the work of over 15,000 state officials [6].

President Biden announced a budget allocation of $6 billion at the end of 2023 to empower communities nationwide to make investments to combat the effects of climate change [7].

On August 16, 2022, the IRA was enacted, which involves the endowment of nearly $400 billion between 2023 and 2030 for climate change mitigation projects [8].

Additionally, these resources have diverse destinations, including the promotion of clean energy, infrastructure construction, and the mitigation of effects on animal and marine life, as observed in Figure 1 [9].

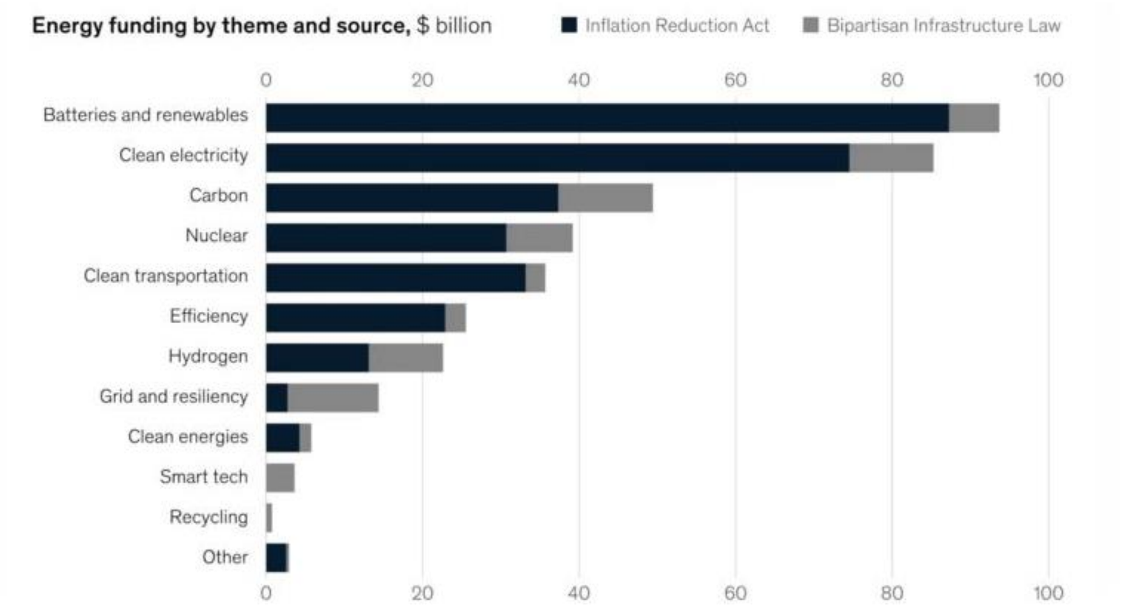


Figure 1. Energy funding from Bipartisan Infrastructure Law and Inflation Reduction Act major funding theme, totaling $370 billion [9]

1.2 MOTIVATION

The project was initiated due to the lack of structured and correlated information on climate change projects implemented by the municipalities of New York. The objective of the present study is to conduct an exhaustive analysis of the climate change projects implemented by municipalities in New York State, with a focus on their sociodemographic characteristics. This analysis aims to provide a more precise understanding of the projects and to inform recommendations regarding the most suitable types of projects for specific communities, based on their socio-demographic profiles and the experiences of other communities that have implemented similar projects. The objective is not merely to conduct this study for the State of New York, but to establish a methodology that can be applied to all communities in the United States.

1.3 OBJECTIVES

The objective of the project is to analyze local climate change adaptation investments in New York State. To achieve this, the following objectives have been established:

1. Analyze whether communities are implementing more structural or environmental projects (protecting natural systems and planting or replacing trees or vegetation).
2. Study the demographic (socio-economic) and flood risk characteristics of the municipal jurisdictions in the Climate Smart Communities program and determine their degree of participation.
3. Analysis of project costs based on demographics, flood risk, and per-capital revenues of the communities.
4. Proposing project initiatives to communities with similar demographic and climatological characteristics.

1.3.1 SUSTAINABLE DEVELOPMENT GOALS (SDG)

The Sustainable Development Goals (hereafter SDGs), 17 in total, were defined in 2015 by the United Nations to eradicate poverty, protect the planet, and ensure prosperity by 2030

[10]. To analyze the correct coordination of this project with the SDGs, it is necessary to first introduce the main data to be analyzed.

In 2022, the New York State government surveyed municipal governments to create a knowledge database of the main municipal actions being carried out to mitigate climate change. All these projects were classified according to the actions undertaken (e.g., modernization of municipal buildings, protection of natural systems, etc.) and the risk they were intended to mitigate.

Upon analysis of the aforementioned source material, it can be established that this work contributes to eight of the seventeen Sustainable Development Goals (SDGs). In particular, it would contribute to the following:

- SDG 3: Good health and well-being: "Ensure healthy lives and promote well-being for all at all ages." The proposed initiatives are conducive to the mitigation of climate change, which in turn would lead to a healthier and safer environmental context, enhanced air quality, and reduced incidence of diseases.
- SDG 6: Clean water and sanitation: "Ensure availability and sustainable management of water and sanitation for all." Sewers represent an essential component of the sanitation cycle, responsible for transporting water from buildings to wastewater treatment plants. At these facilities, the water is treated and returned to the environment in a purified state, free of pollutants. The study of sewer extensions, replacements, and creations will enhance access to clean water and facilitate the sanitation process.
- SDG7: Affordable and clean energy: "Ensure access to affordable, secure, sustainable, and modern energy." Municipal building or retrofit projects may involve constructing energy-efficient structures.
- SDG 9: Industry, innovation, and infrastructure: "Build resilient infrastructure, promote inclusive, and sustainable industrialization, and foster innovation." The analyzed projects mostly involve modernizing, reconstructing, or elevating

municipal buildings and critical infrastructure. Additionally, some projects may include relocating or demolishing municipal buildings or critical infrastructure.

- SDG 11: Sustainable cities and communities: "Make cities and human settlements inclusive, safe, resilient, and sustainable." To achieve this objective, the proposed projects are highly relevant, as they will benefit this purpose not only in the cities, but also in the counties, towns, and villages.
- SDG 13: Climate action: "Take urgent action to combat climate change and its impacts": The previously defined goals serve to highlight the main contribution to this objective since an analysis of climate change adaptation projects is carried out. This work mainly addresses this objective, since the purpose is to analyze the climate change adaptation projects that are being carried out in the communities of New York State.
- SDG 14: Life below water: "Conserve and sustainably use the oceans, seas and marine resources for sustainable development." Protecting natural systems is a category of project actions to be studied.
- SDG 15: Life on land: "Protect, restore and promote sustainable use of terrestrial ecosystems, sustainably manage forests, combat desertification, and halt and reverse land degradation and halt biodiversity loss." As previously stated, the analysis of projects for the protection of natural systems, including those about the planting or replacement of trees or vegetation, serves to reinforce the protection of terrestrial ecosystems.

1.4 METHODOLOGY

To execute the project, a methodology based on four principal steps will be employed:

1. Contextualization
2. Acquisition of data from municipalities in the State of New York
3. Typification of the projects
4. Proposal of projects

The initial step is devoted to the study of the survey, the responses provided, and an analysis of the respondents and their participation.

In the second step, all the information obtained in the previous section is collected together with the demographic, economic, geographic, and climatic data of the communities. During this process, data cleaning is carried out to eliminate duplicate, incoherent, or irrelevant values.

The next step consists of the typification of projects according to classifications such as building projects or eco-projects, as well as the study of the economic correlation between them. Subsequently, the requisite modeling will be conducted to ascertain which demographic and risk factors are influential in program participation as well as project implementation. Additionally, the cost of different projects will be analyzed in terms of per capita expenditures.

In the final step, project initiatives will be proposed to communities with demographic, economic, and climatological similarities.

1.5 FILES AND RESOURCES USED

In conducting this research, the responses to a survey published by the New York City government were utilized, in which members of the Climate Smart Communities program were asked whether they were undertaking or intending to undertake projects during the years 2017 and 2027, respectively. Furthermore, the report includes information about each project, including its cost, duration, and the primary and secondary climate threat reasons addressed by the project.

Concerning the data about the communities in question, the following information is provided: the name of the municipality, the county to which it belongs, and the geographic level of the community. Figure 2 depicts the scheme of how the state of New York is divided. First, there is the primary legal division of the state, which is the county. Each county has a specific function and set of powers [11].

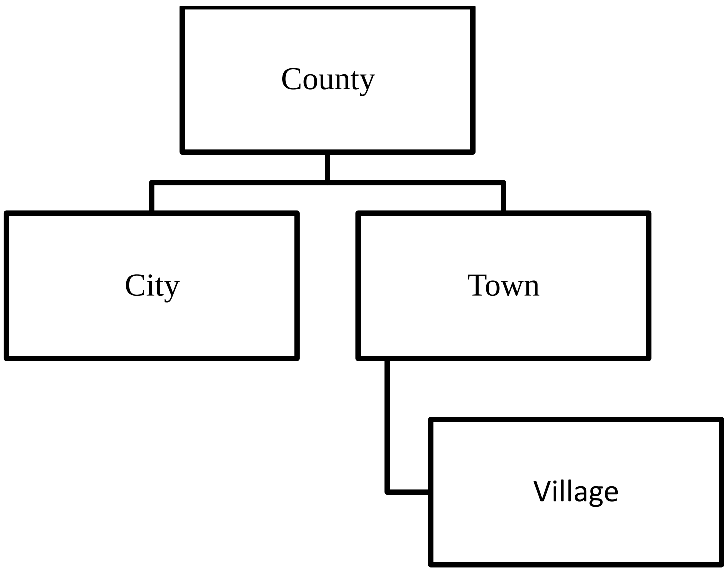


Figure 2. Municipal Geographic Levels Hierarchy. Source: Own elaboration

The state of New York is comprised of 62 counties, as illustrated in Figure 3. Of these, five constitute New York City: the Bronx, Kings, New York, Queens, and Richmond.

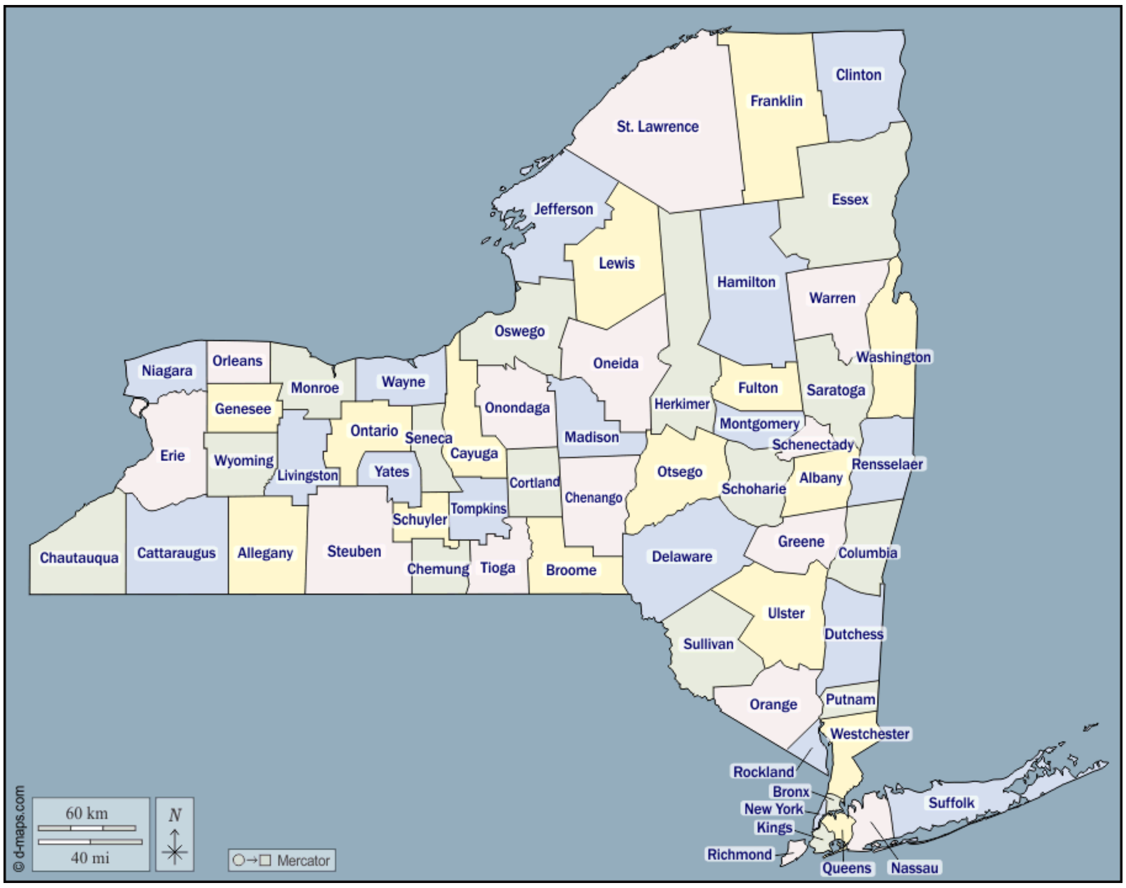


Figure 3. Map of New York State Counties[12]

The survey was made available to program members, but the results for New York City have been compiled in a separate file that is not accessible. The second geographic level comprises cities and towns, which may be self-governing and differ primarily in terms of size and population. Finally, there are the villages, which are smaller entities and may share government with the town to which they belong or have a government of their own.

To complete this project, a multitude of sources were consulted to obtain the following resources:

- List of communities (counties, cities, towns, and villages) that were able to participate in the survey [5].
- Demographic, social, and economic data for the different municipal jurisdictions, provided by the U.S. Census Bureau [13], [14].

- Geographic data for municipal areas provided by the Office of the New York State Comptroller [15].
- Flood risk data for New York State communities [15], [16], [17], [18].

The investigation and subsequent depiction of findings are conducted using the Python programming language, which is advantageous for its extensive array of libraries tailored for data analysis and distinguished by their efficacy in data interpretation and optimization.

Chapter 2. CONTEXTUALIZATION

This chapter presents the algorithms utilized for data processing and cleaning, providing a comprehensive account of the information employed to execute this project. Furthermore, to facilitate the interpretation of the algorithm responsible for the study of the communities that are part of the program, a graphical representation has been prepared to illustrate the results on the geographic map of the state of New York.

2.1 ALGORITHMS

The primary challenge in the processing and data cleansing is the disparate sources utilized to collect the information. Consequently, all algorithms incorporate a comparison to ascertain the accuracy of the community names, their classification, and the county and/or town to which they belong. This is achieved through the "New_York_State_Locality_Hierarc.xlsx" file, which collates the information that has undergone preliminary review. This is because, in numerous instances, the names are written in different formats. For example, the town "LaGrange" may be written in either of two ways: "La Grange" or "LaGrange". Consequently, a comparison has been conducted to ensure that the community's name is consistently written in the same manner, thereby facilitating the accurate collection of all relevant documentation. Furthermore, since data for the year 2022 is often incomplete, data from previous years have been utilized and adjusted to estimate values for the year 2022 [19].

The algorithms defined for data processing correspond to the resources used:

- Socioeconomic data, including population number, total income, percentage of graduates, and other relevant information.
- Participation data from the Climate Smart Communities program indicates which communities are members of this program.
- Survey responses, defining a new dataset to collect information from communities.

- Flood risk data, detailing the set of variables to be used for the study of climatological conditions.
- Finally, the dataset of revenues per capita for the communities has been included.

2.1.1 SOCIOECONOMIC DATA

To obtain demographic, social, and economic data from the various communities of New York, Algorithm 1 has been developed. The initial stage of this algorithm is to gather data and store it in a data frame, eliminating irrelevant variables. This is necessary as all files include a considerable amount of non-relevant data that must be excluded. Subsequently, the column of community names is modified. In addition to including the name, it includes the geographic level and the name of the state. For this purpose, the word "New York" is eliminated, and the name column is separated into two: Muni Name (the name of the community is stored) and Class (the type of community is stored).

It is important to note that, although the data frame includes the percentage of the white population, it has been adjusted to study the non-white population variable through the corresponding calculation. In addition, for the rented housing variable, which is not expressed as a percentage, we have chosen to show the corresponding percentage.

```
Algorithm 1: Demographic data
  Input: files
  n=len(files);
  df_aux =( );
  df_demographic=( );
  for $(i = 1, i \leq n, i{+}{+})$ do
      df_aux = Store data from files (i);
      df_demographic = df_demographic + df_aux;
  end
  Remove "New York" from df_demographic ("Muni Name");
  Split df_demographic ("Muni Name"") in df_demographic ("Muni Name") and df_demographic
    ("Class");
  Remove unnecessary columns from df_demographic;
  Negate df_demographic ("White_%");
  Convert df_demographic ("Rent") in percentage;
  Delete rows from df_demographic where population=0;
  Output: df_demographic
```

Algorithm 1. Demographic data

The algorithm has been applied and customized for the three geographic levels of New York State, which are as follows:

1. Villages
2. Cities and towns
3. Counties

2.1.1.1 Villages

The preliminary step is to eliminate rows that do not correspond to villages, as the dataset encompasses not only villages but also other communities with different geographic census tracts. This is necessary to ensure the accuracy and precision of the data analysis. The latter non-village communities have been excluded from the study because they do not align with the geographic scope of the Climate Smart Communities program, which is limited to villages.

A limitation of the data from villages is that it does not include the county to which it belongs, nor does it indicate whether the village belongs to a town. For this reason, the following steps have been included in the initial algorithm:

1. Reading file "New_York_State_Location": database with the names of the communities, towns, and counties to which they belong (if applicable).
2. Incorporate the pertinent data (including the column of villages and counties to which the various villages belong) into the database of the villages. It is important to note that, in addition to the County column, a second column titled "2nd County" has been included, as many villages are situated within two counties simultaneously. In this instance, the designation "County" would be the primary county in question.

2.1.1.2 Cities and towns

In the data set derived from the Census, the names of the communities include the county to which the cities and towns belong. Consequently, the algorithm has been customized to process and separate the names. The name column has been divided into three sections: "Muni Name", which corresponds to the name of the city or town; "County", which indicates the county to which it belongs; and "Class", which specifies whether the entity is a city or a town.

2.1.1.3 Counties

At this level, the only modification that has been necessary is to include a new column, which includes the name of the county in the "County" column. This alteration has been implemented to facilitate the coding of visual representations.

2.1.2 CLIMATE SMART COMMUNITIES

To ascertain the communities that have had access to the survey, an algorithm has been devised to determine who is included in the program at the time of the survey and who has had access. Algorithm 2 illustrates the process of creating a column in the data frame that includes a value of 1 for each community if the municipality is included in the program and a value of 0 otherwise. To this end, the file containing the names of the participating communities has been utilized. The issue is that the data set contains a single column that includes both the type of community and its name. Therefore, the initial step is to divide this column into two separate columns, one containing the name of the municipality and the other containing its classification. The challenge in analyzing the communities that are part of the program is that only the name of the community and its geographic level were included. Additionally, there are some communities in the state with the same geographic level that have the same name. Consequently, in such instances, it was necessary to ascertain which of the two communities was included in the program.

```
Algorithm 2: Climate Smart Communities Program
Input: df_demographic
df_smart_communities =( );
df_smart_communities = Store data from (List of Climate Smart Communities at time of Survey);
Split df_smart_communities ("Muni Name"") in df_smart_communities ("Muni Name") and
  df_smart_communities ("Class");
n=len (df_demographic);
m=len (df_smart_communities);
for ($i = 1, i \leq n$ ,i++ ) do
    for ($j = 1, j \leq m$ ,j++ ) do
        if (df_demographic ( i; "Muni Name","Class")== df_smart_communities ( j; "Muni
          Name","Class")) then
            df_demographic (i;"Participation")= 1;
        end
    end
end
for ($i = 1, i \leq n$ ,i++ ) do
    if df_demographic ("Participation" )!= 1 then
        df_demographic (i;"Participation" )= 0;
    end
end
Output: df_demographic
```

Algorithm 2. Climate Smart Communities Program

2.1.3 SURVEY

Algorithm 3 illustrates the methodology for obtaining survey responses. These responses include pertinent information such as the community undertaking the project, its characteristics, the nature of the project, the date of its commencement, and the associated cost. The responses are initially stored in a data frame and then copied into two separate data frames:

- One includes the responses of respondents who indicated that they are engaged in climate change adaptation projects, and the dates of these projects align with the dates of the survey questions, which were administered between 2017 and 2027. It should be noted that in some cases, respondents indicated that they were engaged in climate change adaptation projects, yet the projects were not underway at the time of the survey. Consequently, the responses have been modified in the annexed code to ensure that they do not appear in this data frame).

- The other data frame contains the responses to the surveys indicating that the respondents are not engaged in climate change adaptation projects or that they are undertaking such projects, but not on the specified date.

Subsequently, a new binary variable was included in both data frames, representing a value of 1 if the municipality responded to the survey and a value of 0 otherwise. The same process was repeated to integrate a variable that represents whether the municipality has answered in the affirmative to the survey and is implementing the planned projects on the established date. If the answer is in the affirmative, the variable is set to a value of 1, and if it is in the negative, it is set to a value of 0.

```
Algorithm 3: Survey
Input: df_demographic
df_survey =( );
df_survey_yes =( );
df_survey_no =( );
df_survey = Store data from (2023-0702 FOIL Data from survey used in Climate Change report);
df_survey_yes = df_survey (“YesNo to HSA”) == “Yes“;
df_survey_no=df_survey \ { df_survey (“YesNo to HSA”) == “Yes“ };
n=len (df_demographic);
m=len (df_survey);
for ($i = 1, i \leq n$ ,i++ ) do
    for ($j = 1, j \leq m$ ,j++ ) do
        if (df_demographic ( i; “Muni Name”,“Class”, “County” )== df_survey ( j; “Muni
          Name”,“Class”, “County” )) then
            df_demographic ( i; “Answer_survey” )= 1;
        end
    end
end
for ($i = 1, i \leq n$ ,i++ ) do
    if df_demographic (“Answer_survey” )!= 1 then
        df_demographic ( i; “Answer_survey” )= 0;
    end
end
p=len (df_survey_yes);
for ($i = 1, i \leq n$ ,i++ ) do
    for ($k = 1, k \leq p$ ,k++ ) do
        if (df_demographic ( i; “Muni Name”,“Class”, “County” )== df_survey_yes ( k; “Muni
          Name”,“Class”, “County” )) then
            df_demographic ( i; “Answer_yes_survey” )= 1;
        end
    end
end
for ($i = 1, i \leq n$ ,i++ ) do
    if df_demographic (“Answer_yes_survey” )!= 1 then
        df_demographic ( i; “Answer_yes_survey” )= 0;
    end
end
Remove unnecessary columns from df_survey_yes;
Output: df_demographic,df_survey,df_survey_yes,df_survey_no
```

Algorithm 3. Survey

2.1.4 FLOOD RISK

To store the flood risk data that each municipality has, Algorithm 4 has been designed. The algorithm receives the demographic data frame and the so-called geo_FIPS, which is another data frame containing the FIPS (Federal Information Processing Standards) code of each community, together with its name, its classification, and the corresponding county and/or town.

The initial step is to extract the flood risk data from the disparate files. The analysis of the flood risk conditions was conducted using three variables (included in the weather conditions files) that represent the percentage of properties according to the risk of exposure to floods (being multiplied_percent1 minimum risk and multiplied_percent10 extreme risk):

- Extreme flooding: multiplied_percent10
- High flooding: multiplied_percent8-10
- Significant flooding: multiplied_percent5-10

Subsequently, the data are linked to the geoFIPS data frame via the FIPS code, given that the flood risk file solely comprises the FIPS code and the flood variables. Subsequently, the risk variables were merged with the demographic characteristics data frame via the municipality name, along with the classification and the county to which it belongs.

```
Algorithm 4: Flood risk
Input: df_demographic, gdf_FIPS
df_flood_risk =( );
df_flood_risk= Store data from (MuniFloodRisk);
n=len (df_flood_risk);
for ($i = 1, i \leq n$ ,i++ ) do
    df_flood_risk (i; "Extreme_Flooding"= df_flood_risk (i; 'multiplied_percent10");
    df_flood_risk (i; "High_Flooding") = df_flood_risk (i; 'multiplied_percent8") + df_flood_risk (i;
      "multiplied_percent9") + df_flood_risk (i; 'multiplied_percent10");
    df_flood_risk (i; "Significant_Flooding") = df_flood_risk (i; 'multiplied_percent5") +
      df_flood_risk (i; "multiplied_percent6") + df_flood_risk (i; 'multiplied_percent7") +
      df_flood_risk (i; "multiplied_percent8") + df_flood_risk (i; 'multiplied_percent9") +
      df_flood_risk (i; "multiplied_percent10");
end
m=len (gdf_FIPS);
for ($j = 1, j \leq m$ ,j++ ) do
    for ($i = 1, i \leq n$ ,i++ ) do
        if (gdf_FIPS ( i; "FIPS_CODE" )== df_flood_risk ( j; "FIPS_CODE" )) then
            gdf_FIPS ( i; "Extreme_Flooding" ) = df_flood_risk ( i; "Extreme_Flooding" );
            gdf_FIPS ( i; "High_Flooding" ) = df_flood_risk ( i; "High_Flooding" );
            gdf_FIPS ( i; "Significant_Flooding" ) = df_flood_risk ( i; "Significant_Flooding" );
        end
    end
end
z=len (df_demographic);
for ($k = 1, k \leq z$ ,k++ ) do
    for ($j = 1, j \leq m$ ,j++ ) do
        if (df_demographic ( k; "Muni Name","Class", "County" )== (gdf_FIPS ( j;
          "FIPS_CODE" ) then
            df_demographic ( k; "Extreme_Flooding" ) = df_flood_risk ( j; "Extreme_Flooding" );
            df_demographic ( k; "High_Flooding" ) = df_flood_risk ( j; "High_Flooding" );
            df_demographic( k; "Significant_Flooding" ) = df_flood_risk ( j; "Significant_Flooding"
              );
        end
    end
end
Output: df_demographic
```

Algorithm 4. Flood risk

2.1.5 REVENUES

The final data processing algorithm is designed to incorporate the total per capita revenue data for the communities, as illustrated in Algorithm 5. First, the file is read to extract the data, which is included in the main data frame (demographic). Nevertheless, the objective is to study the per capita revenues, which is why the monetary amount was divided by the population of each community.

```
Algorithm 5: Revenues
Input: df_demographic
df_revenues =( );
df_revenues = Store data from (Total Revenues per capita);
n=len (df_demographic);
m=len (df_revenues);
for ($i = 1, i \leq n$ ,i++ ) do
    for ($j = 1, j \leq m$ ,j++ ) do
        if (df_demographic ( i; “Muni Name”,“Class”, “County” )== df_revenues ( j; “Muni
          Name”,“Class”, “County” )) then
            df_demographic ( i; “Total_Revenues” )= df_revenues ( i; “Total_Revenues” ) ;
            df_demographic ( i; “Revenues” )= df_demographic ( i; “Total_Revenues” ) /
              df_demographic ( i; “Population” );
        end
    end
end
Output: df_demographic
```

Algorithm 5. Revenues

2.1.6 VARIABLES

This section defines the variables to be studied. Firstly, Table 1 contains information regarding the data frame df_demographic. This data frame primarily comprises demographic, socio-economic, and flood risk data for the communities, as well as information on their participation in the program and the implementation of projects.

Variable	**Type**	**Definition**
Muni Name	Text	Name of the community
Class	Text	Geographic level (county, city, town, or village)
County	Text	County to which the community belongs

Variable	**Type**	**Definition**
2nd County	Text	Second county to which the municipality belongs, if applicable
Town	Text	Town to which the community belongs, if applicable
GEO_ID	Numeric	Geographic identifier
Population	Numeric	Total population
Male_%	Numeric	Percentage of male population
Black_%	Numeric	Percentage of African American population
Asian_%	Numeric	Percentage of Asian population
Hispanic_%	Numeric	Percentage of Hispanic or Latino population
Nonwhite_%	Numeric	Percentage of non-white population
Employment_%	Numeric	Percentage of the population aged 16 and over employed
Degree_%	Numeric	Percentage of the population aged 25 and over with a university degree or higher
Median_income	Numeric	Median household income
Poverty_%	Numeric	Percentage of the population in poverty
English_%	Numeric	Percentage of the population over the age of five that speaks only English
Rent_%	Numeric	Percentage of rented homes

Variable	**Type**	**Definition**
Disability_%	Numeric	Percentage of the population with a disability
Revenues	Numeric	Revenues per capita
Extreme_Flooding	Numeric	Percentage of households with extreme flood risk
High_Flooding	Numeric	Percentage of households with high flood risk
Significant_Flooding	Numeric	Percentage of households with significant flood risk
Participation	Binary	Indicates if a municipality participates (1) or does not participate (0) in the Climate Smart Communities Program
Answer_survey	Binary	Indicates whether the community has responded to the survey: 1 indicates response, 0 indicates no response
Answer_yes_survey	Binary	Represents whether the municipality is undertaking projects within the specified timeframe, that is, between 2017 and 2027: 1 indicates yes, 0 indicates no

Table 1. Description of df_demographic variables

The following table, Table 2, presents the survey response variables from the df_survey_yes data frame.

Variable	**Type**	**Definition**
Muni Name	Text	Name of the community
Class	Text	Geographic level (county, city, town, or village)

Variable	**Type**	**Definition**
County	Text	County to which the community belongs
HSA ID	Numeric	Project type identifier
HSA name	Text	Project type
Project Name	Text	Project name
StartDate	Numeric	Start year
EndDate	Numeric	End year
Cost	Numeric	Project cost
Pct Cost CC	Numeric	Incremental percentage cost of a new measure or solution compared to applying a less suitable solution
Pct Federal	Numeric	Percentage of funding from federal sources
Pct State	Numeric	Percentage of funding from state sources
Pct Local	Numeric	Percentage of funding from local sources
drought_1	Binary	Indicates if the primary hazard being addressed is drought (1) or another hazard (0)
erosion_1	Binary	Indicates if the primary hazard being addressed is erosion (1) or another hazard (0)
extreme heat_1	Binary	Indicates if the primary hazard being addressed is extreme heat (1) or another hazard (0)

Variable	**Type**	**Definition**
extreme weather_1	Binary	Indicates if the primary hazard being addressed is extreme weather (1) or another hazard (0)
flooding (not related to sea-level rise)_1	Binary	Indicates if the primary hazard being addressed is flooding (not related to sea-level rise) (1) or another hazard (0)
sea-level rise_1	Binary	Indicates if the primary hazard being addressed is sea-level rise (1) or another hazard (0)
drought_2	Binary	Indicates if the secondary hazard being addressed is drought (1) or another hazard (0)
erosion_2	Binary	Indicates if the secondary hazard being addressed is erosion (1) or another hazard (0)
extreme heat_2	Binary	Indicates if the secondary hazard being addressed is extreme heat (1) or another hazard (0)
extreme weather_2	Binary	Indicates if the secondary hazard being addressed is extreme weather (1) or another hazard (0)
flooding (not related to sea-level rise)_2	Binary	Indicates if the secondary hazard being addressed is flooding (not related to sea-level rise) (1) or another hazard (0)
sea-level rise_2	Binary	Indicates if the secondary hazard being addressed is sea-level rise (1) or another hazard (0)

Table 2. Description of df_survey_yes variables

2.2 GEOGRAPHIC DISTRIBUTION OF CLIMATE SMART COMMUNITIES

To initiate this project, the initial step was to create Figure 4, which depicts the communities that were members of Climate Smart Communities at the time the survey was conducted. This revealed that 353 communities out of 1581 communities, or 22.33% of communities, participated in the program. While this may appear to be a low number, it has been increasing over time and currently stands at 412 communities [5].

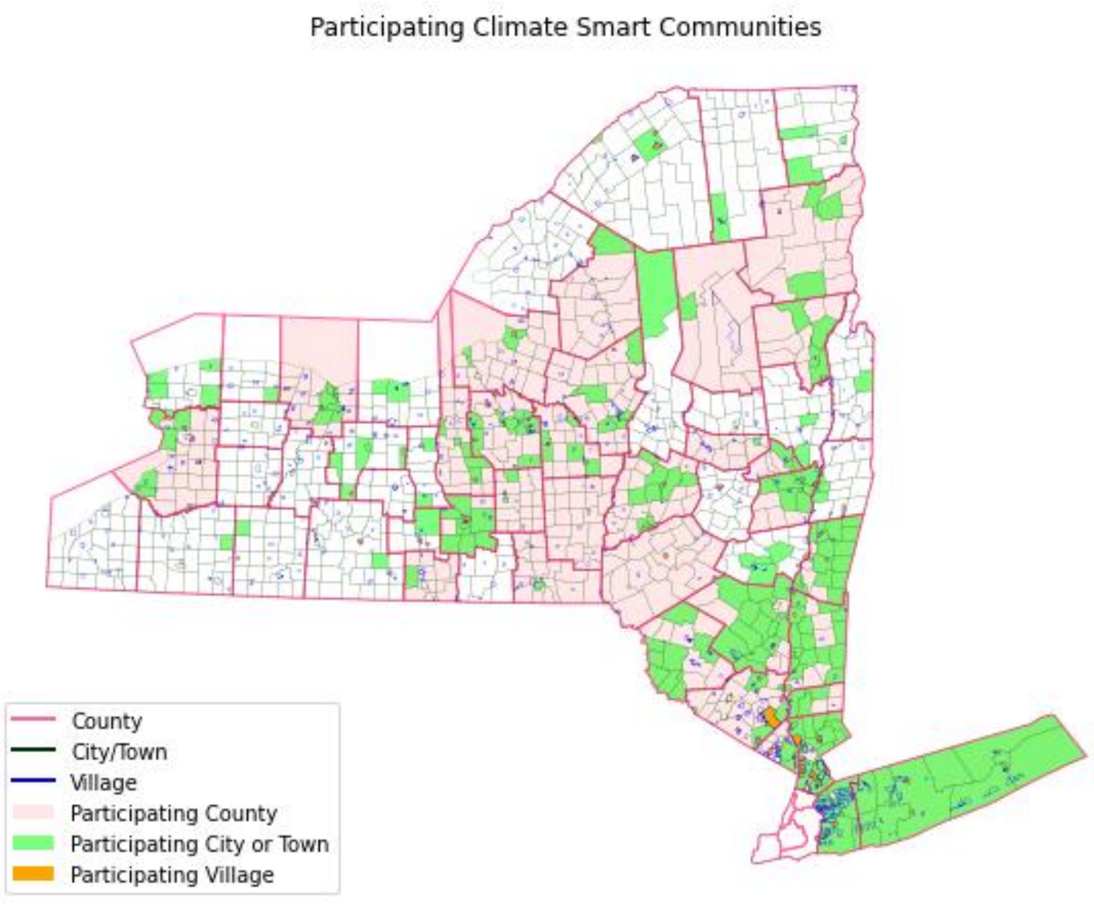

Participating Climate Smart Communities
County
City/Town
Village
Participating County
Participating City or Town
Participating Village

Figure 4. Participating Climate Smart Communities

Due to the overlapping of the different levels of communities, graphs have been constructed for each community class. Figure 5 illustrates the counties that are participating in the program, demonstrating that over half of the counties are included in the program.

Furthermore, it can be inferred that most of the participating counties are located in the southeast region of the state (54.39%).

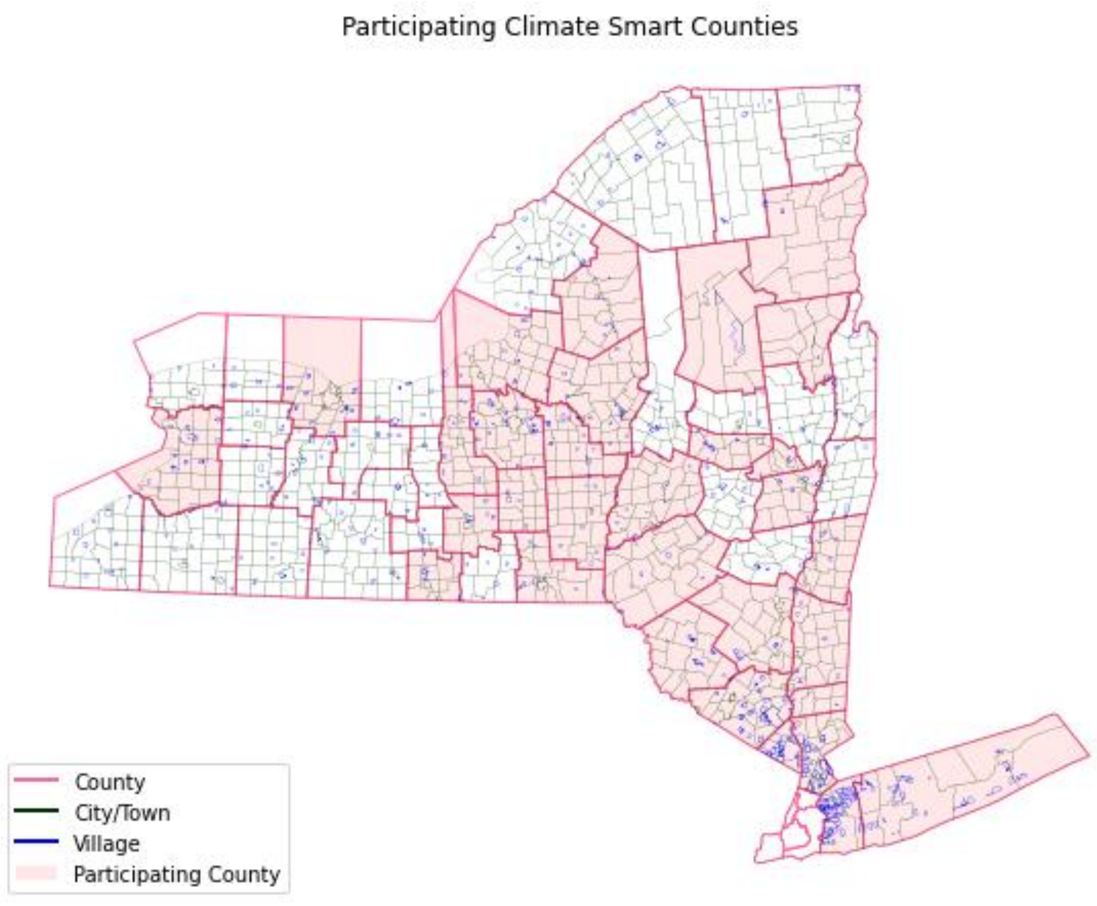


Figure 5. Participating Climate Smart Counties

As illustrated in Figure 6, the program encompasses 210 of the 782 cities and towns, representing 21.17% of the total.

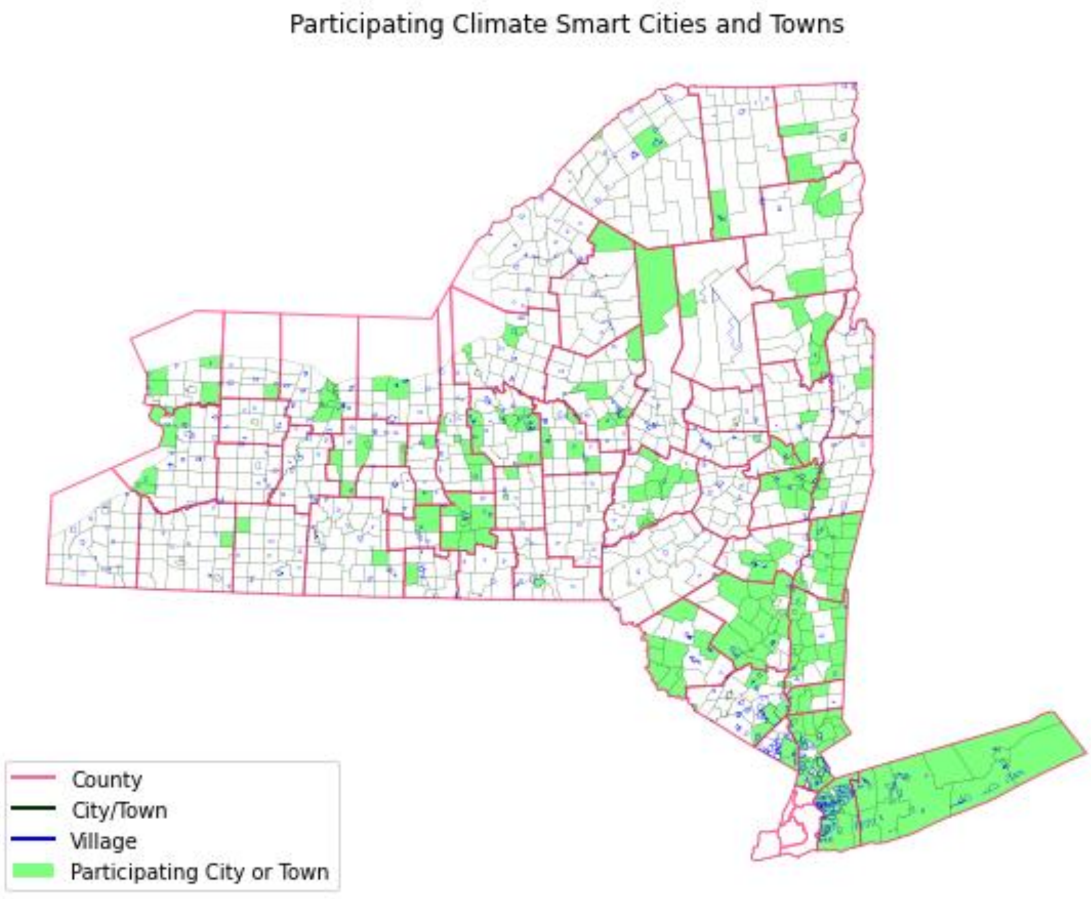

Participating Climate Smart Cities and Towns
County
City/Town
Village
Participating City or Town

Figure 6. Participating Climate Smart Cities and Towns

Additionally, the communities belonging to this level that are part of the program can be studied indirectly, as illustrated in Figure 7. This is because these communities are not participants per se, but rather the counties to which they belong. Consequently, the counties themselves have been considered as a unit of analysis, given that the projects they undertake are for all the cities, towns, villages, and hamlets that comprise them.

It is notable that all municipalities in Nassau and Suffolk, along with their respective counties, are engaged in the program, collectively striving to mitigate the effects of climate change. Additionally, most municipalities in other counties, namely Columbo, Ulster, and Putnam, are also engaged in the program.

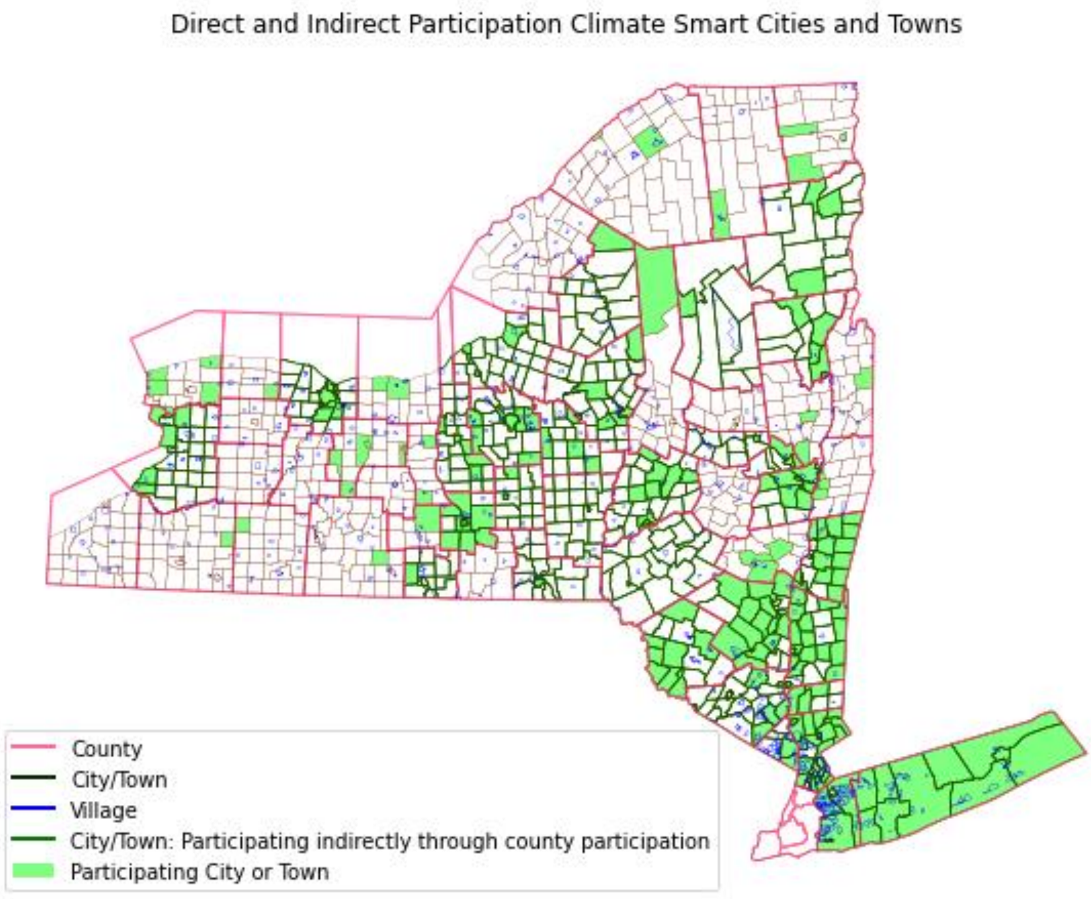

Direct and Indirect Participation Climate Smart Cities and Towns
County
City/Town
Village
City/Town: Participating indirectly through county participation
Participating City or Town

Figure 7. Direct and Indirect Participation Climate Smart Cities and Towns

Concerning the villages, the participation rate is lower, with 112 out of 420 individuals (21.05%) participating, as illustrated in Figure 8.

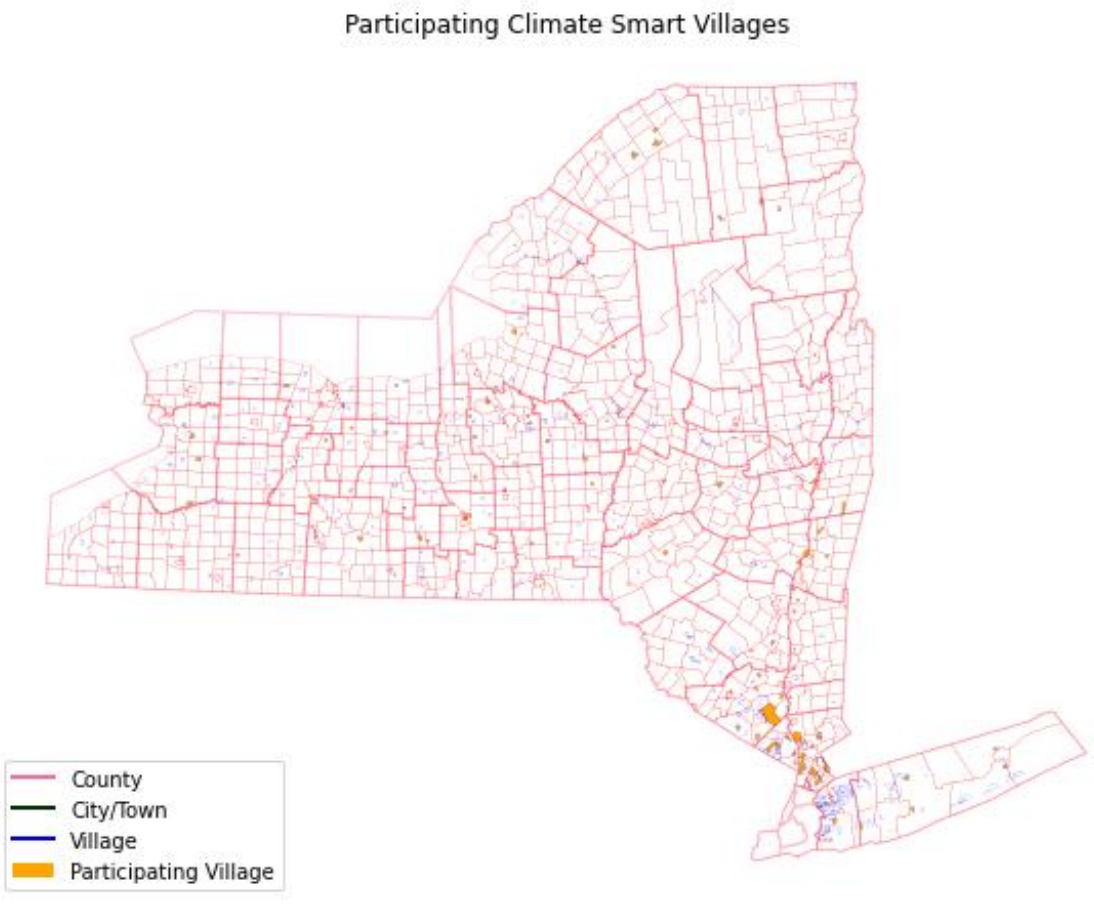


Figure 8. Participating Climate Smart Villages

As with the case of cities and towns, a similar study was conducted to ascertain the villages that indirectly participate in the program. Figure 9 illustrates that the number of villages increases significantly because of this approach.

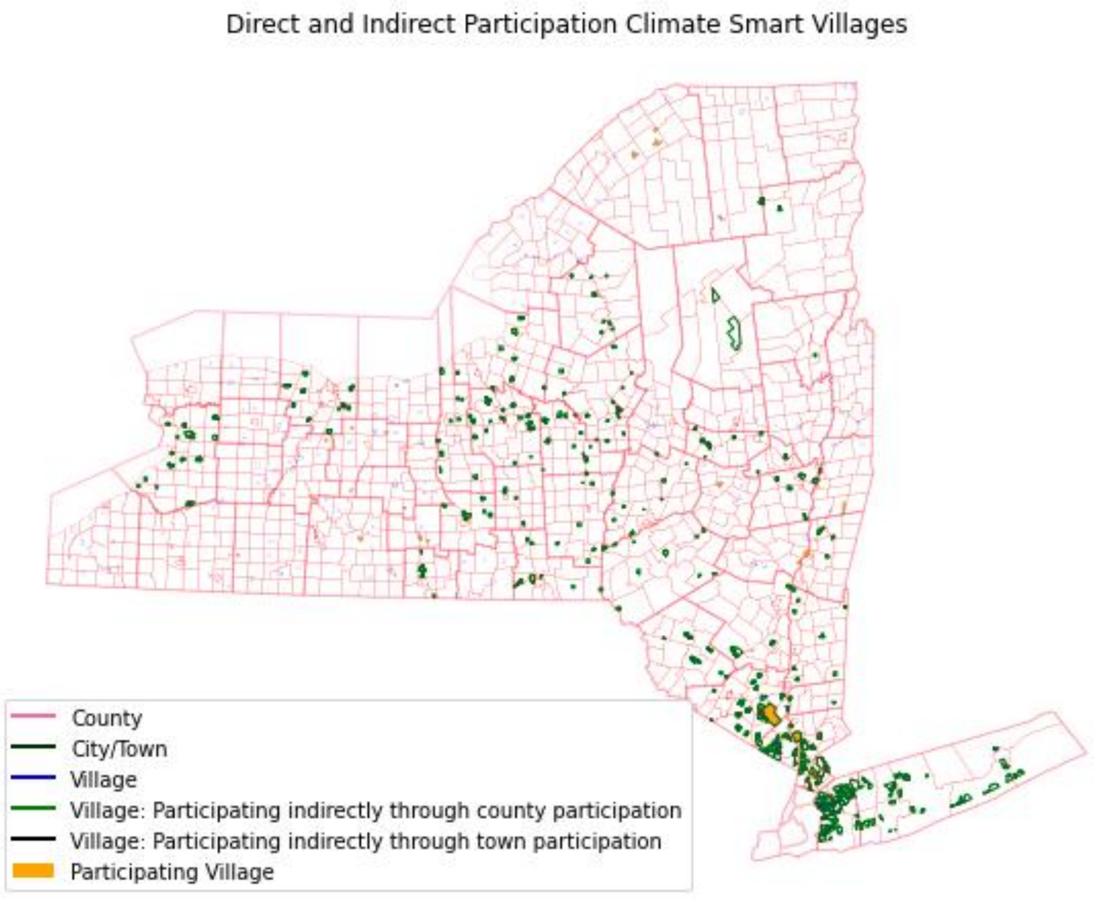

Direct and Indirect Participation Climate Smart Villages
County
City/Town
Village
Village: Participating indirectly through county participation
Village: Participating indirectly through town participation
Participating Village

Figure 9. Direct and Indirect Participation Climate Smart Villages

Finally, Figure 10 illustrates the villages included in the main county, as well as those belonging to the second county included in the program. Consequently, the increase is insignificant, given that only three villages are affected.

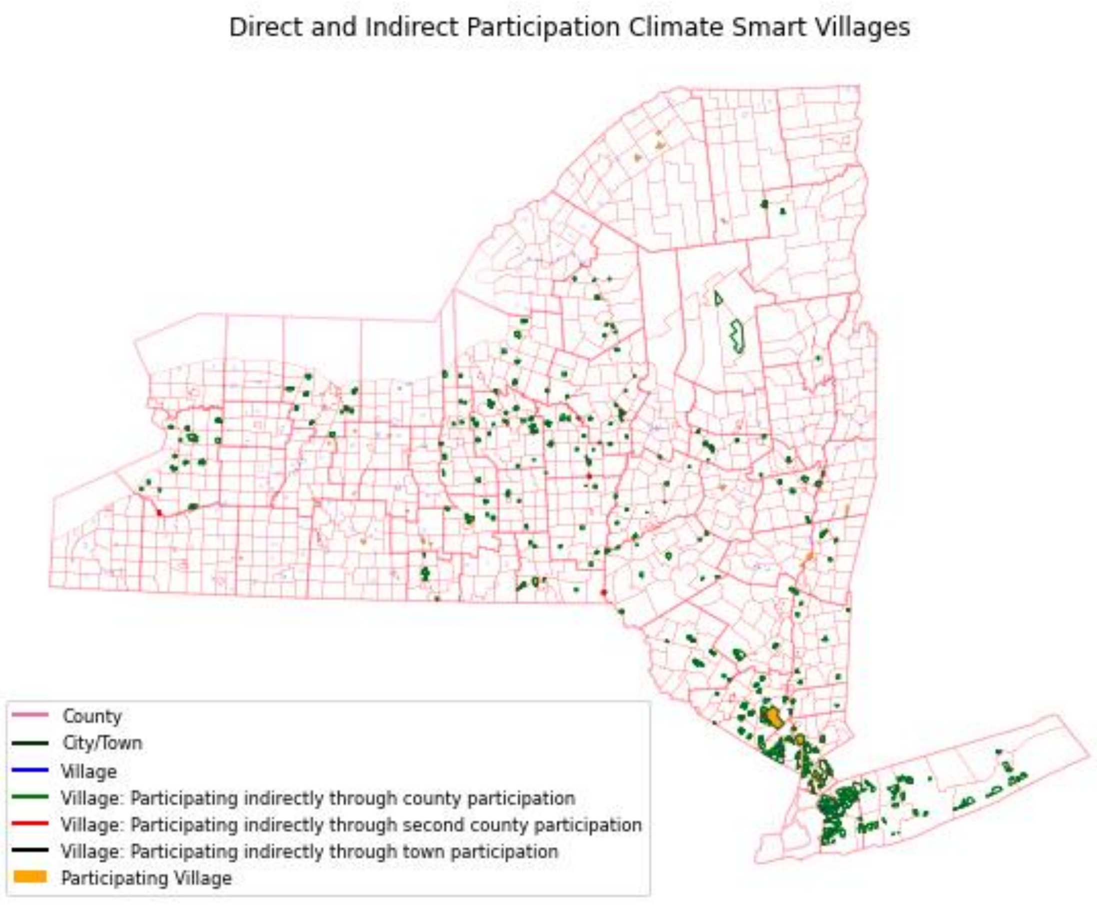

Direct and Indirect Participation Climate Smart Villages
County
City/Town
Village
Village: Participating indirectly through county participation
Village: Participating indirectly through second county participation
Village: Participating indirectly through town participation
Participating Village

Figure 10. Direct and Indirect Participation Climate Smart Villages (Second County)

Chapter 3. STRUCTURAL OR ENVIRONMENTAL PROJECTS

To ascertain which counties, allocate a greater proportion of their financial resources to structural projects or those designated as environmental projects, the variable designated as HSA ID has been employed. This variable provides insight into the specific type of project being undertaken, as illustrated in Table 3.

HSA ID	HSA
1	Retrofit, raise or rebuild municipal buildings
2	Relocate or demolish municipal buildings or other critical infrastructure
3	Rebuild or retrofit critical infrastructure other than buildings
4	Address increased pavement deterioration on roads
5	Enlarge, replace, or create culverts
6	Replace, build, or raise bridges
7	Build or make significant improvements to protective structures
8	Protect natural systems
9	Plant or replace trees or vegetation
10	Additional

Table 3. Types of projects

The value of this variable has been used to group projects into two categories:

- structural projects (identified as 1 to 7)
- environmental projects (categories 8 and 9)

Projects with identification 10, have not been studied since they do not fit into any of the other categories, and, additionally, their number is much lower than that of the other projects.

Once the projects have been classified according to the above scheme, they have been grouped by county, rather than by the level of municipal jurisdiction, to facilitate the analysis and to ensure the drawing of accurate conclusions.

3.1 ALGORITHM

In order to ascertain which counties are investing a greater proportion of their resources in structural projects and environmental projects, Algorithm 6 has been developed. The initial stage of the process entails aggregating the financial resources allocated to construction initiatives at the village, city, town, and county levels, subsequently categorizing them according to the county to which they pertain. The same procedure is then employed for the categorization of environmental projects. Subsequently, the classification is conducted through the application of percentiles, to organize the municipalities into three distinct groups: those that have allocated the greatest proportion of expenditure, those that have allocated an intermediate amount, and those that have allocated the least (not including the communities that are not undertaking any projects). Once the counties with the highest and lowest expenditures on each category have been identified, a correlation analysis is conducted on the subset of counties.

```
Algorithm 6: Structural or environmental projects
Input: df_demographic, df_survey_yes
df_total_cost = ();
df_structural_cost = ();
df_environmental_cost = ();
df_total_cost = Sum cost by County for all projects in df_survey_yes;
df_structural_cost = Sum cost by County for projects with $HSA\ ID \leq 7$ in df_survey_yes ;
Rename column Cost to Structural_cost in df_structural_cost;
df_environmental_cost = Sum cost by County for projects with $HSA\ ID == (8 | 9)$ in df_survey_yes;
Rename column Cost to ost_environmental in df_environmental_cost;
min_structural = 33.33 percentile of Structural_cost;
max_structural = 66.6 percentile of Structural_cost;
df_structural_cost_min = Copy of df_structural_cost;
df_structural_cost_max = Copy of df_structural_cost;
df_structural_cost_med = Copy of df_structural_cost;
n=len (df_structural_cost);
for ($i = 1, i \leq n, i{+}{+}$ ) do
    a = df_structural_cost[i, "Structural_cost"];
    if $a \leq min\_structural$ then
        Remove row i from df_structural_cost_max and df_structural_cost_med;
    end
    else if $a \geq max\_structural$ then
        Remove row i from df_structural_cost_min and df_structural_cost_med;
    end
    else
        Remove row i from df_structural_cost_min and df_structural_cost_max;
    end
end
Sort df_structural_cost_min, df_structural_cost_max, df_structural_cost_med by Structural_cost in descending
  order;
min_environmental = 33.33 percentile of environmental_cost;
max_environmental = 66.6 percentile of environmental_cost;
df_environmental_cost_min = Copy of df_environmental_cost;
df_environmental_cost_max = Copy of df_environmental_cost;
df_environmental_cost_med = Copy of df_environmental_cost;
m=len (df_environmental_cost);
for ($j = 1, j \leq m, j{+}{+}$ ) do
    a = df_environmental_cost[j,"environmental_cost"];
    if $a \leq min\_environmental$ then
        Remove row j from df_environmental_cost_med and df_environmental_cost_max;
    end
    else if $a \geq max\_environmental$ then
        Remove row j from df_environmental_cost_min and df_environmental_cost_med;
    end
    else
        Remove row j from df_environmental_cost_min and df_environmental_cost_max;
    end
end
Sort df_environmental_cost_min, df_environmental_cost_max, df_environmental_cost_med by
  environmental_cost in descending order;
df_cost_g = Merge df_structural_cost and df_environmental_cost on County, fill NaN with 0;
df_cost_g = df_cost_g.set_index("County");
correlation = Correlation between environmental_cost and Structural_cost;
Output: df_total_cost, df_structural_cost_max, df_environmental_cost_max, df_structural_cost_med,
        df_environmental_cost_med, df_structural_cost_min, df_environmental_cost_min, df_structural_cost,
        df_environmental_cost, correlation
```

Algorithm 6. Structural or environmental projects

3.2 RESULTS

3.2.1 STRUCTURAL PROJECTS

Figure 11 shows the distribution of structural projects in the different counties.

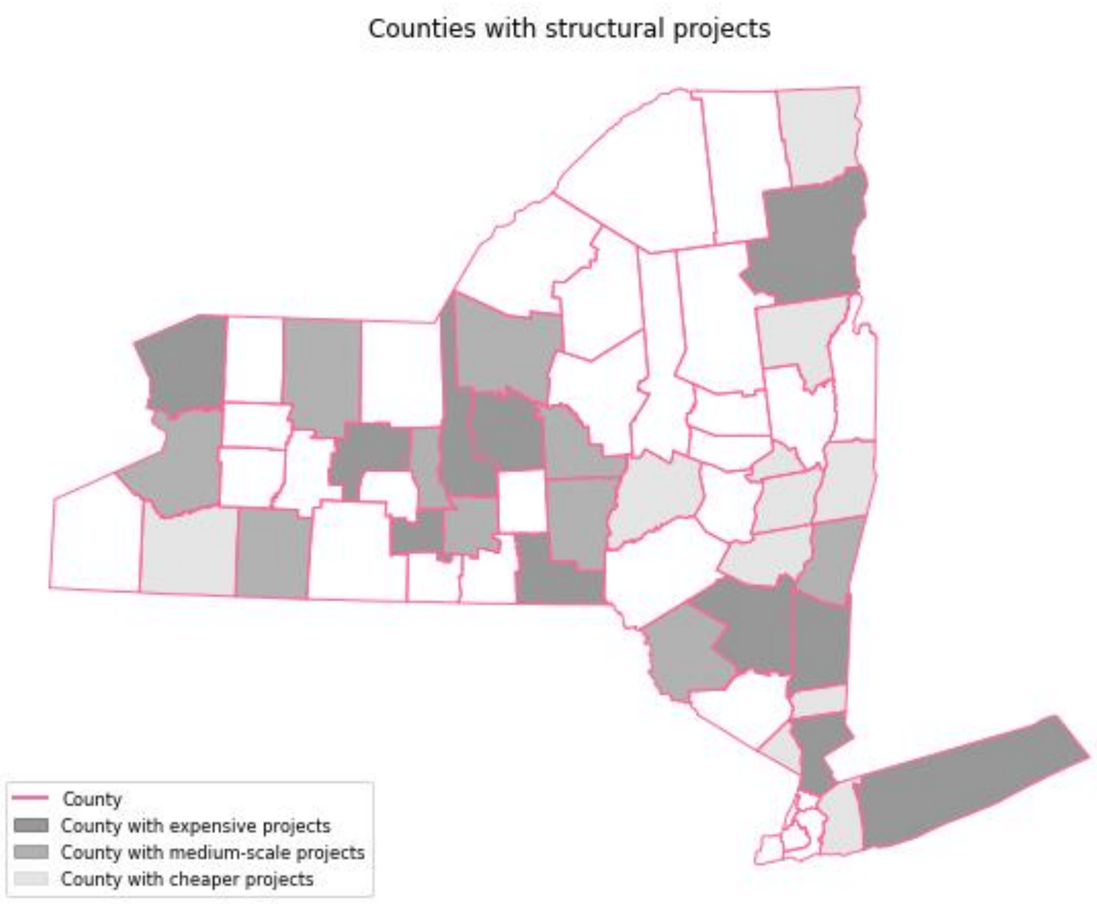

Counties with structural projects
County
County with expensive projects
County with medium-scale projects
County with cheaper projects

Figure 11. Counties with structural projects

As can be seen in Table 4, Broome is the county that spent the most money on structural projects. The main reason why it is the county that has spent the most on structural projects is because of Binghamton because it is the only city in the county that has done projects. The most expensive project and why it stands out in the table is because of the Binghamton and Johnson City Wastewater Treatment Plant Rehabilitation project, which was undertaken between 2017 and 2021 and cost $275,000,000. This plant was flooded in September 2011 by Tropical Storm Lee [20]. Other projects undertaken by the same town include the relocation or new construction of the town's fire station and improving the energy efficiency of the town hall building through the installation of a green roof.

Similar to Broome County, Dutchess County is also undertaking significant projects. The City of Poughkeepsie is investing over $90,000,000 in projects to improve its drainage infrastructure, aiming to prevent flooding and structural damage. This is followed by the County itself, which has undertaken a $20,000,000 drinking water infrastructure improvement project. A relevant fact is that the third most expensive project is being carried out by the Village of Millerton, which has spent more than $10,000,000 on projects aimed at sewage drainage. This is significant because the villages are very small communities and Millerton has invested half of what its county has invested.

Next on the list would be Cayuga County which, like Poughkeepsie, has done $80,000,000 worth of sewer improvement projects and wastewater separation upgrades in the Town of Auburn. On the water issue, it has also allocated $10,000 to relocate its public safety building out of areas of potential flooding in the event of a failure of a dam they have upstream. The rest of the amount, which is more than $26 million, comes from projects undertaken by the Aurora and Cayuga, which is relevant because they are small villages.

County	Structural cost
Broome	$ 286,600,000.00
Dutchess	$ 143,374,000.00
Cayuga	$ 115,010,000.00
Suffolk	$ 113,599,788.00
Westchester	$ 93,367,228.00
Essex	$ 71,250,000.00

County	Structural cost
Schuyler	$ 38,640,000.00
Ontario	$ 38,620,000.00
Niagara	$ 35,000,000.00
Onondaga	$ 35,000,000.00
Ulster	$ 34,485,852.00

Table 4. Top Counties by Structural Project Expenditure

As Table 5 shows, the two counties that have allocated the least to infrastructure projects are Warren and Rensselaer. In the former, the village of Chester allocated $72,000 to install a backup generator and $50,000 to increase the size of sewers. The second lowest spending county is Rensselaer, of which East Nassau Village stands out with a project to apply chip seals to road surfaces throughout the village to improve their durability and strength ($134,000). Like the previous county, it also spent $14,400 on improving the sewer system and another project it will do is replace a bridge for $10,000. All these projects in this village were done primarily to mitigate the effects of flooding.

County	Structural cost
Warren	$ 122,000.00
Rensselaer	$ 158,400.00
Rockland	$ 1,200,000.00
Clinton	$ 1,545,000.00
Otsego	$ 1,615,000.00

County	**Structural cost**
Schenectady	$ 2,885,690.00
Albany	$ 3,400,000.00
Putnam	$ 3,633,000.00
Nassau	$ 3,760,000.00
Cattaraugus	$ 7,862,000.00
Greene	$ 10,870,000.00

Table 5. Counties with the Lowest Structural Project Expenditure

3.2.2 ENVIRONMENTAL PROJECTS

As was done for structural projects, the distribution of environmental projects is shown in Figure 12. The darkest counties are those that invest the most in these projects, while the lightest counties are those that invest the least, without considering those that do not invest.

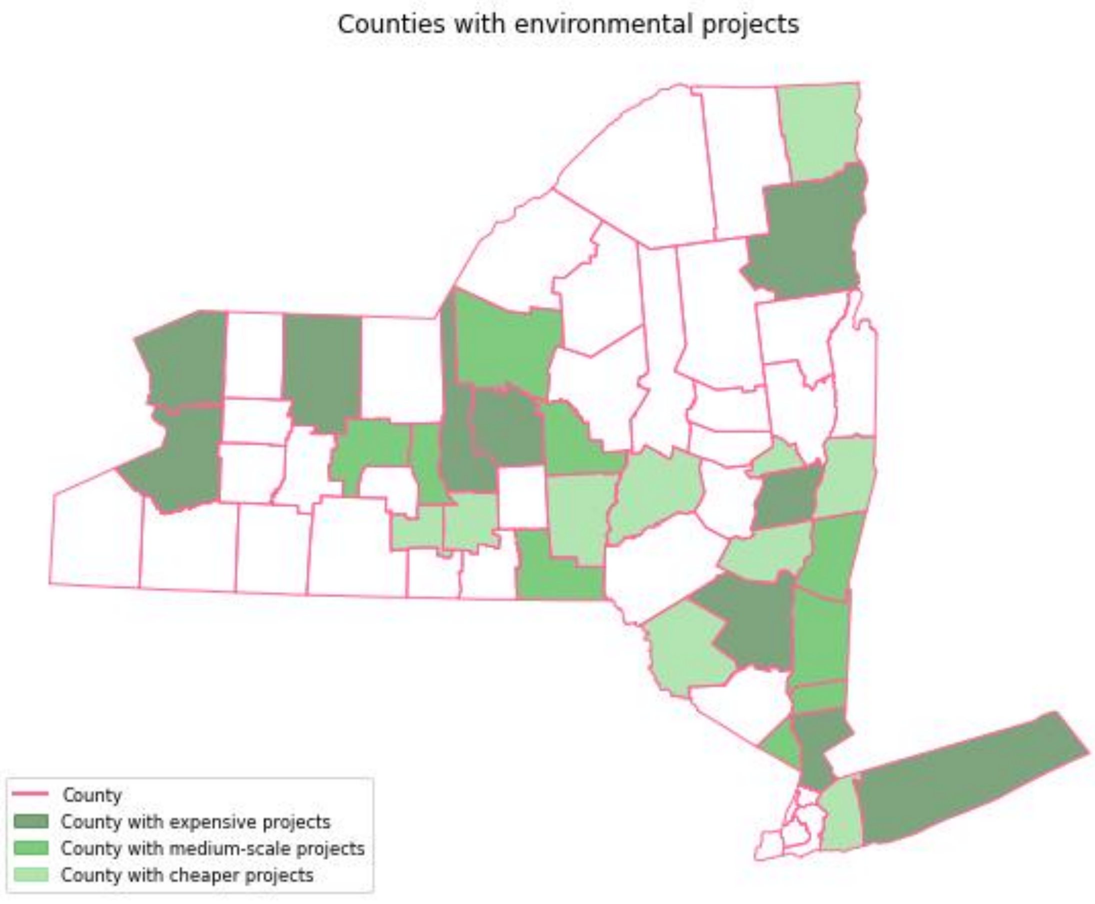

Counties with environmental projects
County
County with expensive projects
County with medium-scale projects
County with cheaper projects

Figure 12. Counties with environmental projects

Suffolk is the county that has spent the most money on environmental projects, as shown in Table 6, the county itself has spent about $15 million on shoreline restoration to protect the ecosystem. The town of Brookhaven, which is part of Suffolk, has spent more than $8,000,000 to promote natural resource conservation and reforestation by planting trees and shrubs in streams and lakes [21]. It has also invested in promoting plant education among the community [22]. The Village of Greenport has invested more than $10,000 in tree planting and maintenance.

The City of Syracuse, part of Onondaga County, committed $8,000,000 to the Onondaga Creek Flood Storage Study and Construction Project, which uses open space (natural areas and existing parks) or the creek to store water and prevent flooding [23]. It also allocated $5,000,000 for tree planting.

County	**Environmental cost**
Suffolk	$ 24,709,038.00
Onondaga	$ 13,000,000.00
Erie	$ 6,575,000.00
Monroe	$ 6,057,800.00
Albany	$ 5,435,000.00
Niagara	$ 3,300,000.00
Ulster	$ 2,470,000.00
Westchester	$ 1,835,000.00
Cayuga	$ 1,147,000.00
Essex	$ 1,005,000.00

Table 6. Top Counties by Environmental Project Expenditure

It is noteworthy that the Town of Dannemora in Clinton County has initiated at least one environmental project, namely the spraying of stems with herbicides to prevent the proliferation of invasive species. This project has not incurred any costs, as evidenced in Table 7. Another municipality that merits mention for its minimal expenditure is the Village of Montour Falls, situated in Schuyler County. It allocates a mere $2.5 million to the Arbor Program, which is devoted to the planting and upkeep of trees [24].

County	**Environmental cost**
Clinton	$ 0

County	Environmental cost
Schuyler	$ 2,500.00
Greene	$ 10,000.00
Rensselaer	$ 10,060.00
Schenectady	$ 20,000.00
Otsego	$ 42,500.00
Chenango	$ 49,757.00
Tompkins	$ 51,000.00
Nassau	$ 55,000.00
Sullivan	$ 100,000.00

Table 7. Counties with the Lowest Environmental Project Expenditure

3.3 RELATION

Once the counties that allocate the most money to structural projects and environmental projects have been studied, it has been considered interesting to determine the Pearson correlation between the different costs of the communities. The value of this correlation is 0.19, indicating a weak but statistically significant relationship. While the difference between the number of counties that carry out structural and environmental projects is relatively small, the amount of money allocated to structural projects ($1,243,573,238.00) is much higher than in environmental projects ($70,453,656.00).

As was done in the previous section, it is necessary to highlight the significant efforts of Suffolk County and its municipal jurisdictions in climate change adaptation. Suffolk County ranks third in spending on climate change adaptation projects, as shown in Table 2, with 17 projects underway.

Broome County spends the most on its major project, while Dutchess County spends a similar amount on 18 projects, ranking second.

County	**Cost**
Broome	$ 289,620,000.00
Dutchess	$ 144,230,500.00
Suffolk	$ 139,768,826.00
Cayuga	$ 116,169,000.00
Westchester	$ 95,502,228.00
Essex	$ 72,255,000.00
Onondaga	$ 48,000,000.00
Monroe	$ 39,747,920.00
Ontario	$ 38,786,500.00
Schuyler	$ 38,667,500.00
Niagara	$ 38,300,000.00
Ulster	$ 37,360,852.00
Seneca	$ 31,920,000.00

Table 8. Top Counties by Project Expenditure

To conclude the analysis, Figure 13 illustrates the counties that invest the greatest proportion of resources in structural and environmental projects. The counties of Cayuga, Niagara, Onondaga, Suffolk, Ulster, Westchester, and Essex are identified as the counties that allocate the greatest proportion of resources to each type of project.

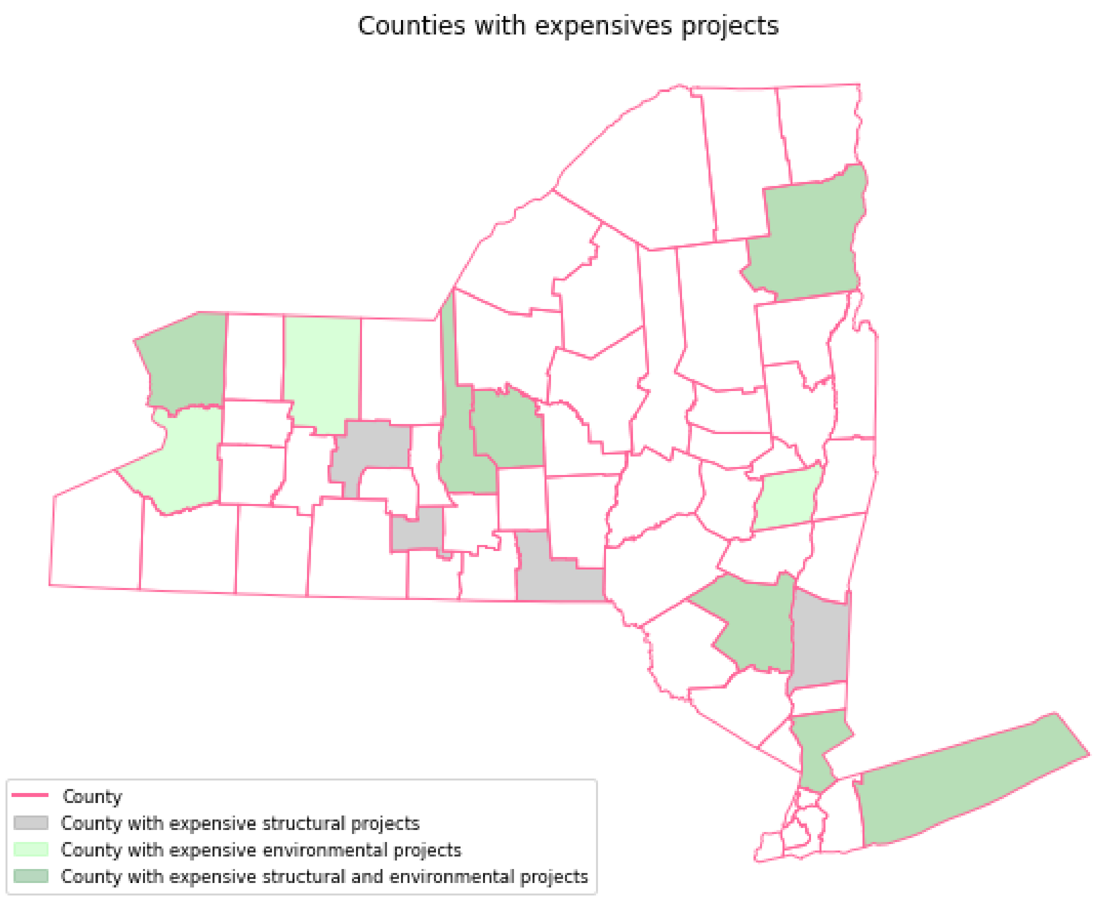

Counties with expensives projects
County
County with expensive structural projects
County with expensive environmental projects
County with expensive structural and environmental projects

Figure 13. Counties with expensive projects

Chapter 4. DEGREE OF COMMUNITY PARTICIPATION AND INVOLVEMENT

The objective of this chapter is to identify the most significant variables that determine whether a community is part of the Climate Smart Communities program and to quantify the degree of its participation. To achieve this objective, two statistical models have been employed: the probit model and the random forest model. These have been applied at both the global level and the level of each municipal jurisdiction, which includes counties, cities, towns, and villages.

4.1 PROBIT MODEL

The probit model is a type of regression model that is employed to forecast the values of a binary variable (referred to as the dependent variable) in accordance with a set of variables designated as independent variables. The technique is not merely employed to forecast values; it is also utilized to discern which variables are the most pivotal, that is, which variables prompt the dependent variable to transition from 0 to 1 [25], [26].

The probit model is predicated on the following equation:

$$P(\,X) = \Phi(X\beta)$$

- X: Vector of independent variables
- β: Vector of model coefficients, estimated by the maximum likelihood method
- Y: represents the dependent variable
- $P(\,X)$: denotes the conditional probability that the dependent variable (Y) assumes the value of 1 when X, the set of independent variables, is given
- Φ: denoted by the probit function, represents the cumulative distribution function of a standard normal distribution

A more detailed explanation of the probit model is provided below:

$$P(X) = \Phi(\beta_0 + \sum_{i=1}^{n} \beta_i X_i)$$

$$\Phi(z) = \int_{-\infty}^{z} \frac{1}{\sqrt{2\pi}} e^{\frac{-t^2}{2}} dt$$

- Y: represents the dependent variable
- X: denotes the set of independent variables
- $P(X)$: is the conditional probability that Y assumes the value of 1 in the presence of X
- β_0: represents the constant or intercept of the model
- $\beta_1, \beta_2, \ldots \beta_n$: the coefficients of the model, β_i indicate the relative importance of X_i in determining the probability that Y equals 1
- $X_1, X_2, \ldots, X_n$: the vector of independent variables
- Φ: is a cumulative distribution function that transforms a value of z, which is drawn from a standard normal distribution, into a value between 0 and 1

The probability is defined on a scale between 0 and 1, to determine whether or not individuals will participate in the program and whether they will undertake projects. To achieve this, the variable to be predicted (Y*) must assume a value of 1 or 0.

The classification in question is determined through the following equation:

$$Y^* = \left\{ \begin{matrix} 1 & si \;\; Y > 0.5 \\ 0 & si \;\; Y \leq 0.5 \end{matrix} \right\}$$

The study of the importance of the independent variables is conducted through the application of statistical significance. To this end, once the model has been constructed, the p-values and coefficients for each variable are obtained from it. A variable is deemed to be statistically significant if the p-value is less than 0.05, as this indicates that the variable has a notable impact on the dependent variable [27]. Additionally, the sign of the coefficient indicates whether the independent variable exerts a positive or negative influence on the dependent variable's prediction, while its numerical value represents the variation in the probability of Y for each unit increase in the independent variable under study.

4.2 RANDOM FOREST MODEL

The random forest model is an algorithm utilized in machine learning to address classification or regression issues. It is founded upon the training of numerous decision trees, wherein a series of queries are posed in each tree, and a final determination is reached based on the responses. The model generates a substantial number of decision trees through a random subset of data (bagging method) and subsequently integrates all the trees to yield an outcome [28], as shown in Figure 14.

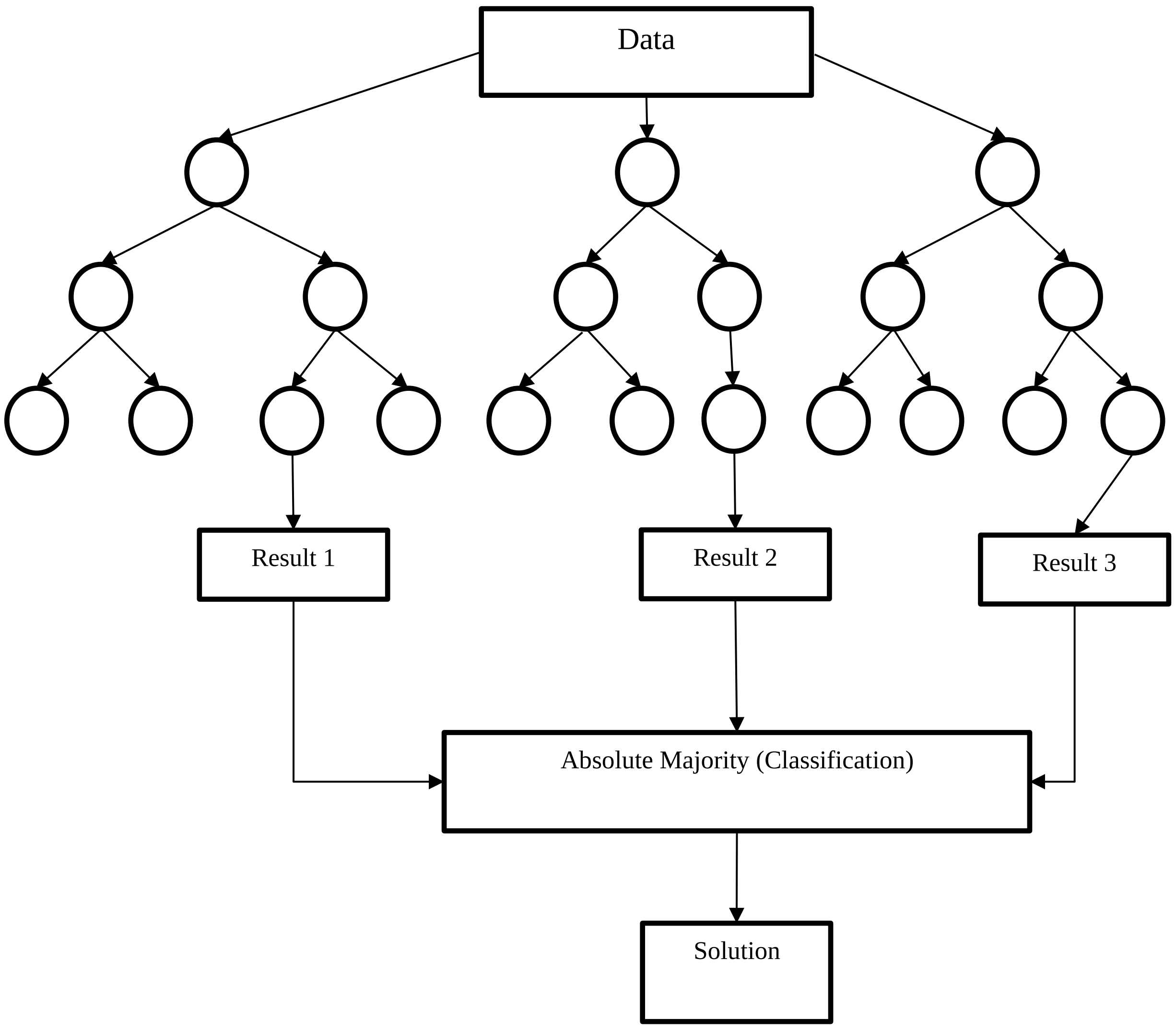


Figure 14. Random forest model [29]

As in the previous model, the most significant independent variables can be identified to predict the value correctly, although it is not possible to define whether they affect to prediction of positive or negative values. To obtain a complete analysis, two methods have been used: the Gini importance method, also called mean decreasing impurity (MDI), and the permutation importance method, also known as mean decreasing accuracy (MDA). In both methods, the difference in the absolute value of the precision of the full model and a specific model (different in each case) is calculated, and this difference is studied. The greater the difference, the more important the variable is in the model [30].

In the first method, MDI, the specific model employed is the model comprising all variables except the one under analysis. In the second method, MDA, the specific method is obtained by permuting the values of the variable under analysis.

4.3 CONFUSION MATRIX AND ACCURACY

It is crucial to ascertain which variables are the most significant in each scenario. Additionally, to ascertain the optimal model, the confusion matrix and the accuracy of each model have been employed. To this end, the model is trained with 80% of the data and tested with the remaining 20%, which are used to obtain the predictive values. It is important to note that to facilitate a meaningful comparison between both models, a seed value has been set. This value serves to partition the data in a deterministic manner, ensuring that each iteration of the experiment selects the same test and training data. This issue is considered satisfactory since the model has been run without a seed on numerous occasions, and it has been estimated that the accuracy of the model is barely affected by plus or minus one percent. Subsequently, the predicted values are compared against the actual values, thus obtaining the so-called confusion matrix. Furthermore, the accuracy can be calculated as the percentage of units correctly predicted concerning the total number of predictions made.

4.4 PARTICIPATION MODEL

The objective of this section is to demonstrate how the algorithm has been designed to predict whether a community belongs to the Climate Smart Communities program, as well as to identify the most relevant variables.

In this instance, the independent variables are demographic, social, economic, and flood risk data. The dependent variable represents membership in the program, with a value of 1 assigned to those participating and 0 to those not participating.

4.4.1 ALGORITHM

To apply this model, Algorithm 7 has been designed to process the data set of the communities, to store in a data frame those rows that do not contain numerical values in the columns of per capita revenues and flood risk, due to the lack of complete data on these. Subsequently, the data are divided into two groups. The dependent variable, Y, determines whether a community is included in the program. The independent variables, X, includes all other variables, except variables about the survey and non-numerical variables, such as the name of the municipality or county to which the community belongs. The proportion of ones and zeros in Y is analyzed to ensure that at least 30% of the minority class is represented, which is a necessary condition for the model to be valid. X is normalized to facilitate the study of the results and highly correlated variables are eliminated to avoid redundancies. Subsequently, the data are divided into two distinct sets: 80% for training the model and 20% for testing. The probit and random forest models are then fitted with the training data, and their accuracy and confusion matrix are evaluated with the test data. Ultimately, the model is re-run with the complete data set to ascertain the most influential variables for each model.

```
Algorithm 7: Participation Model
Input: df_demographic,corr
df_study=();
q=0;
n=len (df_demographic);
for ($i = 1, i \leq n$ ,i++ ) do
    if df_demographic[i, "Total Revenues"] != NaN & df_demographic[i, "Extreme_Flooding"] != NaN then
        df_study [q] = df_demographic[i] ;
        q= q + 1;
    end
end
Y = df_study["Participation"];
X = df_study ["Population", "Male_ % ",...];
m=len (Y);
Sum=0;
for ($j = 1, j \leq m$ ,j++ ) do
    if Y[j] == 1 then
        Sum = Sum + 1;
    end
end
Proportion = Sum / m ;
if ($Proportion < 0.3$) or ($Proportion > 0.7$) then
    Balance(X, Y)
end
Scale(X);
Matrix = correlation(X);
c = len(columns(X));
for ($z = 1, z \leq c$ ,z++ ) do
    for ($k = 1, k \leq c$ ,k++ ) do
        if $Matrix[z, k] > corr$ then
            Delete k from X;
        end
    end
end
Add column "const" to X;
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2);
Probit_Model_Train = Probit_Model(Y_train, X_train);
Probit_Predicted_Y = Probit_Model_Train(X_test);
Probit_Matrix = Confusion_Matrix(Y_test, Probit_Predicted_Y);
Probit_Model_Results = Probit_Model(X);
Random_Forest_Model_Train = Random_Forest_Model(Y_train, X_train);
Random_Forest_Predicted_Y = Random_Forest_Model_Train(X_test);
Random_Forest_Matrix = Confusion_Matrix(Y_test, Random_Forest_Predicted_Y);
Random_Forest_Model_Results = Random_Forest_Model(X);
Calculate variable importance using Random_Forest_Model_Results via Gini impurity reduction;
Calculate variable importance using Random_Forest_Model_Results via permutation method;
Output: Probit_Model_Results, Random_Forest_Model_Results
```

Algorithm 7. Participation Model

The participation model has been implemented using this algorithm at various levels, with modifications made to the algorithm's input. This includes the introduction of a new variable that defines the geographic level at which the participation model is to be performed.

4.4.2 RESULTS

The designed algorithm requires an input variable, which serves as the limit, to ascertain whether the variables are highly correlated and to eliminate one of them. Given the considerable volume of data, it is estimated that variables are highly related if the correlation coefficient exceeds 0.75. This threshold is relatively high, reflecting the need for precise model analysis.

4.4.2.1 Global

In the implementation of the various models at the global level, covering all communities, significant adjustments were made to eliminate highly correlated variables to avoid multicollinearity problems. For example, the variable representing the non-white population was excluded due to its high correlation with the percentage of the black population, as well as the median income, which shows a significant correlation with the graduate variable. Furthermore, the variable representing the percent of high flood level was excluded due to its high correlation with the percentage of extreme flood levels.

4.4.2.1.1 Probit Model

When applying the probit model at the global level, covering all communities, and using the training and test data sets, the model has been able to correctly predict 266 of the 342 communities analyzed, reaching an accuracy of 77.78%. As shown in Figure 15, it is notable that the model is more effective in predicting the non-participation of communities than in predicting their active participation.

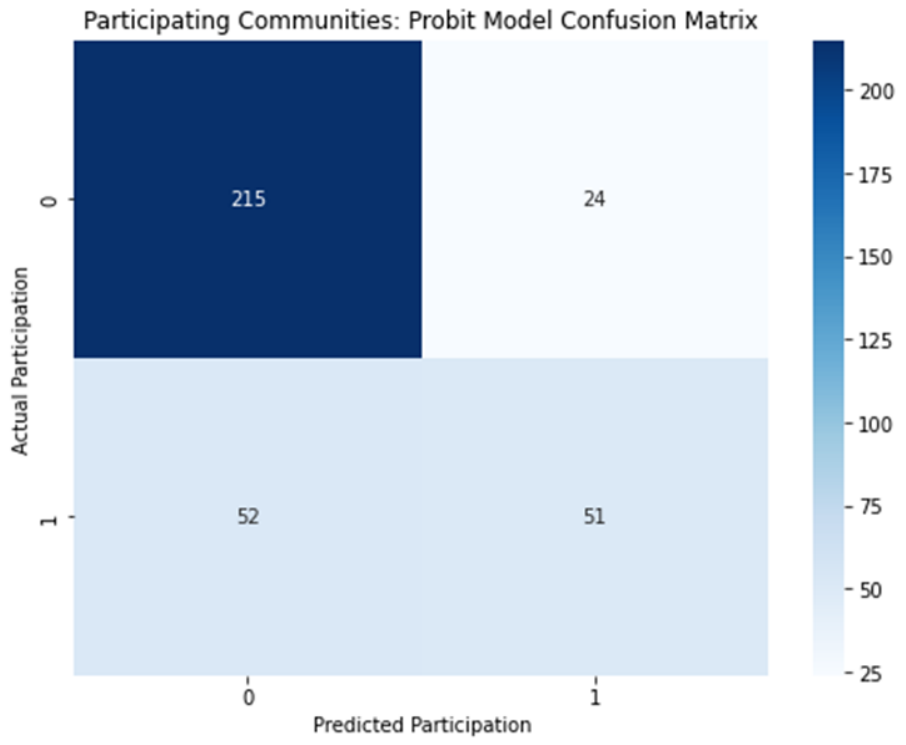


Figure 15. Participating Communities: Probit Model Confusion Matrix

Table 9 presents the variables along with their coefficients and p-values. It is observed that increases in the percentage of people with higher education, linked to higher median income, as well as increases in total population, rental housing units, people with disabilities, the black population (highly correlated with the non-white population), and the Hispanic population increase the probability of participation in the program.

This observation is consistent with the hypothesis that municipalities with higher incomes and better educational attainment are likely to have more awareness and resources to address the challenges of climate change. Furthermore, a higher population density may result in a greater concern for mitigating the risk of climate change, given the potential for greater pollution.

The significance of the disabled population may be attributed to the necessity for communities to guarantee accessibility and security from natural disasters, which profoundly impact this demographic.

Nevertheless, it is noteworthy that the Asian population exhibits a propensity to refrain from participation in the program.

Probit model		
variable	**coefficient**	**p-value**
Degree_%	0.727018	1.43E-36
Population	0.861297	2.95E-10
Rent_%	0.248736	9.16E-08
Disability_%	0.163509	1.46E-03
Black_%	0.144973	1.86E-03
Asian_%	-0.154291	4.95E-03
Hispanic_%	0.16954	5.09E-03
English_%	0.139858	7.03E-02
Revenues	-0.101154	1.97E-01
Employment_%	0.032898	4.57E-01
Poverty_%	0.038367	4.73E-01
Extreme_Flooding	0.026691	6.49E-01
Significant_Flooding	-0.015917	7.49E-01
Male_%	-0.002675	9.48E-01

Table 9. Coefficients and p-values for the Global Participation Probit Model

4.4.2.1.2 Random Forest Model

Figure 16 illustrates the predicted values for the test data set using the random forest model, which exhibits an 81.29% accuracy rate in predicting outcomes. This highlights the model is capacity to accurately predict the communities that are not part of the program.

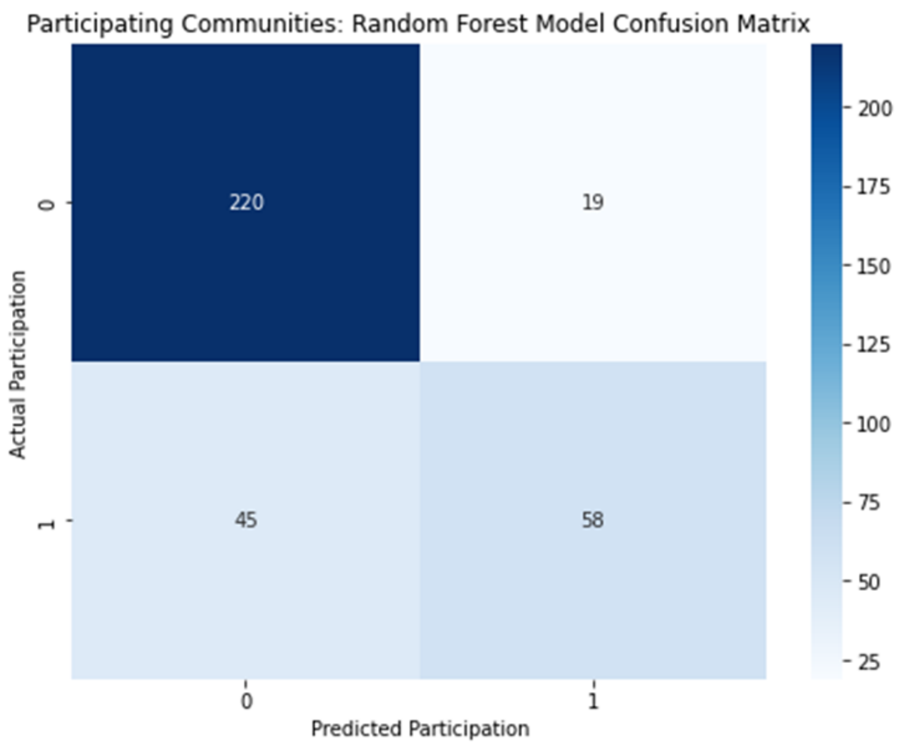


Figure 16. Participating Communities: Random Forest Model Confusion Matrix

As in the preceding model, the most significant variables have been identified, through the Gini and permutation method, as illustrated in Table 10. The most relevant predictors are the percentage of the population with a college degree, which is highly correlated with median income, and the global population. This finding is consistent with the observations mentioned above regarding the optimal economic and educational levels, as well as population density. The significance of the variable representing revenues is noteworthy, as communities with higher incomes are better positioned to allocate greater resources to such initiatives. Conversely, communities with lower revenues may prioritize allocating their budget to more pressing needs, thereby limiting their capacity to participate.

Furthermore, the variables representing the English-speaking, Hispanic, and Black (highly correlated with non-white) populations are also significant.

Gini method		**Permutation method**	
variable	**importance**	**variable**	**importance**
Degree_%	0.1247389	Degree_%	0.0939696
Population	0.1092269	Population	0.0603584
Revenues	0.0890791	Revenues	0.0420236
English_%	0.0838669	English_%	0.0410529
Hispanic_%	0.0762654	Hispanic_%	0.0283419
Black_%	0.0731771	Black_%	0.0221096
Asian_%	0.0665549	Rent_%	0.0181653
Extreme_Flooding	0.0624541	Extreme_Flooding	0.0141979
Rent_%	0.0616331	Asian_%	0.0129114
Significant_Flooding	0.0533445	Male_%	0.0126032
Male_%	0.0515098	Significant_Flooding	0.008736
Employment_%	0.0510126	Poverty_%	0.0084741
Disability_%	0.0503583	Disability_%	0.0059704
Poverty_%	0.0467785	Employment_%	0.0046453

Table 10. Importance of Variables for the Global Participation Random Forest Model

4.4.2.1.3 Comparison

The random forest model is more appropriate for predicting values, as it demonstrates an accuracy approximately 3.5% higher than the probit model.

Concerning the variables, both models concur that the most significant are the percentage of the population with university degrees and the total population. Notably, factors such as the male population, poverty levels, and flood risk conditions are less relevant in both models.

However, there are discrepancies in the valuation of certain variables. The probit model assigns greater importance to rented housing and disabled individuals, whereas the random forest model places greater emphasis on revenues and the English-speaking population.

4.4.2.2 Counties

The undertaking of a participation study at the county level is inherently challenging due to the limited number of communities at this level, which stands at 57 in total. Due to the high correlation of certain variables, the following have been eliminated:

- The non-white population exhibited a correlation of 0.93 with the black population.
- It is also the case that highly educated people are highly correlated with the Asian population.
- English-only speakers exhibited a negative correlation of 0.88 with Hispanics.
- The special needs population demonstrated a correlation of -0.79 with the income variable.
- As in the global model, the percentage of the population at high flood risk (highly correlated with extreme flood risk) was excluded (0.8).

4.4.2.2.1 Probit Model

Figure 17 shows that the probit model is more likely to predict negative real values, i.e. the model would be characterized by predicting counties that are not part of the program, with an accuracy of 75%.

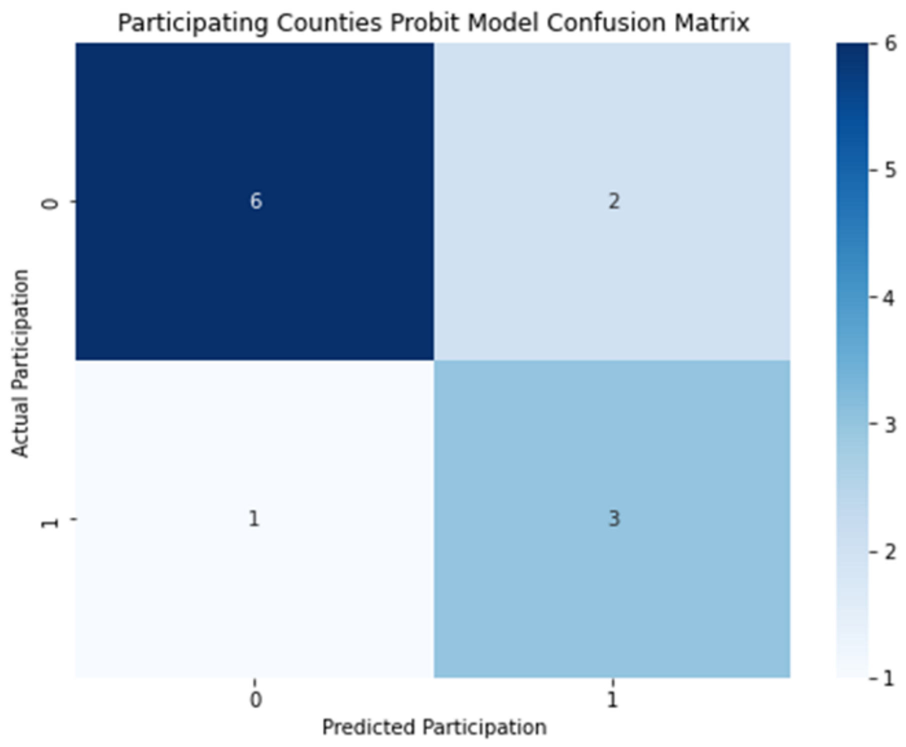


Figure 17. Participating Counties: Probit Model Confusion Matrix

As a consequence of the aforementioned limitation, an examination of Table 11 reveals that there is no deterministic variable that influences the behavior, either positively or negatively, to predict whether a county is a member or not.

Probit model		
variable	**coefficient**	**p-value**
Revenues	0.551759	0.050871
Male_%	-0.616569	0.091152
Extreme_Flooding	0.418008	0.238159
Hispanic_%	0.598179	0.377862

Probit model		
variable	**coefficient**	**p-value**
Asian_%	0.632894	0.436668
Employment_%	-0.316101	0.475188
Significant_Flooding	-0.118216	0.697607
Median_income	0.20908	0.817357
Black_%	0.147267	0.839432
Population	0.212157	0.858403
Rent_%	-0.045704	0.905041
Poverty_%	0.003094	0.993921

Table 11. Coefficients and p-values for the Counties Participation Probit Model

4.4.2.2.2 Random Forest Model

Figure 18 depicts the confusion matrix for the random forest model of participation at the county level. The model's accuracy is 66.67%.

It can be observed that the model correctly predicts four counties that are not part of the program and four that are. However, the model also incorrectly predicts four counties that are members.

This issue is partially attributable to the limited quantity of data, as there are only 57 counties and 80% of them have been utilized to train the model, thereby constraining the model's capacity.

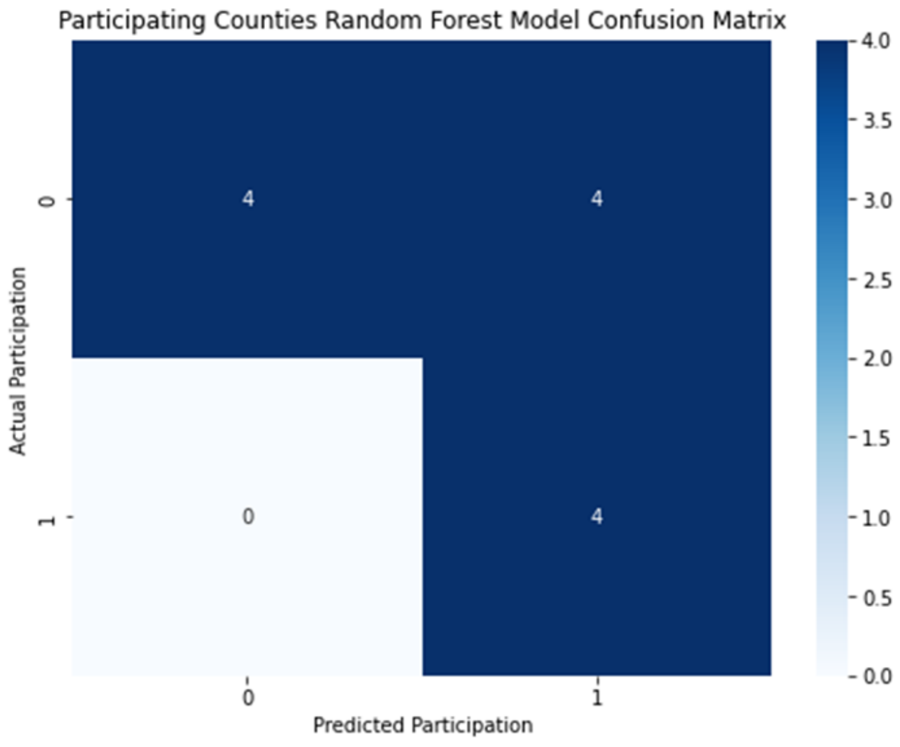


Figure 18. Participating Counties: Random Forest Model Confusion Matrix

Table 12 illustrates the order of importance of the variables in this model. It shows that the Asian population variable (highly correlated with the population with higher education) and the Hispanic population are of great significance for both methods, as is the percentage of rental units. It is reasonable to posit that the Asian education variable is a consequence of the heightened awareness of climate change among this population. However, there is a discrepancy in the relevance of the extreme flood risk variables, with the latter being the most important in the permutation study.

It is noteworthy that, according to the permutation method, certain variables, including global population, black (and non-white) population, median income, significant flood risk, and revenues, do not contribute to the prediction of the dependent variable.

Gini method		Permutation method	
variable	**importance**	**variable**	**importance**
Asian_%	0.13506455	Extreme_Flooding	0.01953748
Hispanic_%	0.11916632	Hispanic_%	0.01594896
Rent_%	0.10455928	Asian_%	0.01076555
Black_%	0.08957675	Rent_%	0.00956938
Revenues	0.08067259	Male_%	0.00438596
Population	0.07679919	Poverty_%	0.00398724
Extreme_Flooding	0.072616	Employment_%	0.00079745
Significant_Flooding	0.06992401	Population	0
Employment_%	0.06816671	Black_%	0
Male_%	0.06714054	Median_income	0
Poverty_%	0.06255213	Significant_Flooding	0
Median_income	0.05376194	Revenues	0

Table 12. Importance of Variables for the Counties Participation Random Forest Model

4.4.2.2.3 Comparison

In conclusion, the probit model is a superior fit to the data, with an accuracy difference of 9%. Although this model fits better, it was impossible to determine the variables' importance due to the low number of counties. This was to be expected, as the models are most effective when applied to large volumes of data.

4.4.2.3 Cities and Towns

At the city and town levels, there is a high positive correlation between the characteristics of the black population and the non-white population. This is also the case for variables indicating high and extreme flood risk. The English-speaking variable has been excluded from the analysis due to its negative correlation with the Hispanic population variable.

4.4.2.3.1 Probit Model

When performing the probit model of cities and towns, the model once again predicts better the municipalities that are not part of the program, as can be seen in Figure 19. The accuracy of the model is 82.95%.

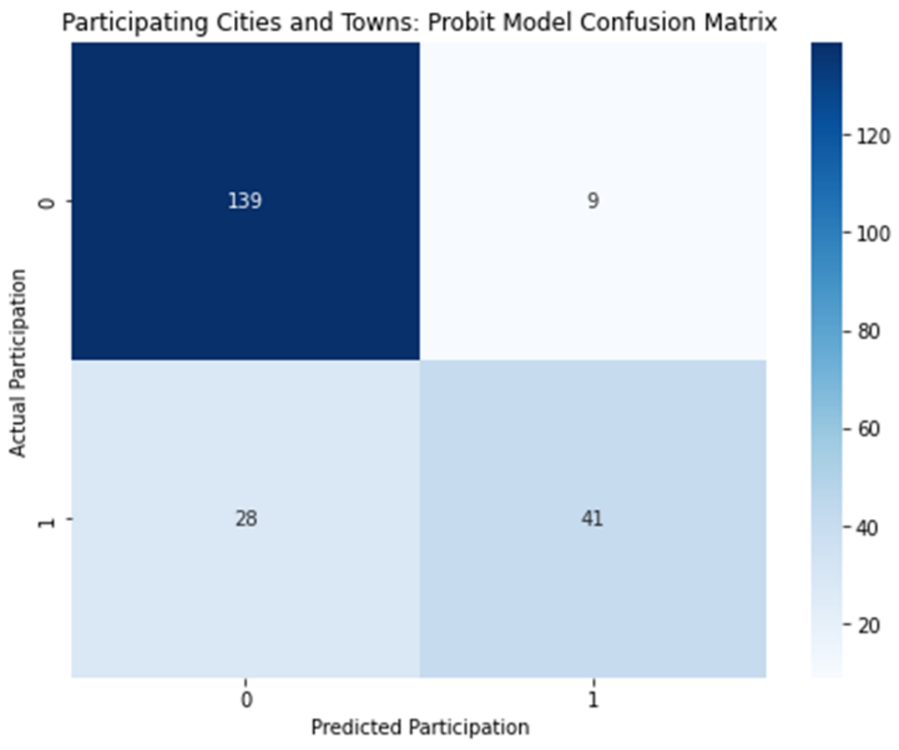


Figure 19. Participating Cities and Towns: Probit Model Confusion Matrix

The most crucial variables in the probit model, as illustrated in Table 13, encompass the proportion of individuals with higher education, as well as the disabled population, the Hispanic population (negatively correlated with the English-speaking population), revenues, and the black population (representing the non-white population). As previously stated, these variables are consistently correlated with an increased likelihood of community participation in the program.

Probit model		
variable	**coefficient**	**p-value**
Degree_%	0.847178	8.68E-24
Disability_%	0.157125	1.09E-02
Hispanic_%	0.146871	1.99E-02
Revenues	0.106753	2.05E-02
Black_%	0.144231	3.00E-02
Poverty_%	0.117577	7.00E-02
Population	0.149697	1.07E-01
Rent_%	0.07169	2.81E-01
Median_income	-0.064816	4.79E-01
Employment_%	0.033449	5.76E-01
Male_%	-0.018244	7.36E-01
Asian_%	-0.01783	7.88E-01
Extreme_Flooding	0.013129	8.29E-01
Significant_Flooding	-0.012182	8.46E-01

Table 13. Coefficients and p-values for the Cities and Towns Participation Probit Model

4.4.2.3.2 Random Forest Model

The method demonstrates an 86.18% accuracy rate in forecasting actual test values. As illustrated in Figure 20, there is a greater prevalence of accurate negative predictions than positive ones. Consequently, the model exhibits enhanced efficacy in anticipating communities not participating in the Climate Smart Communities program.

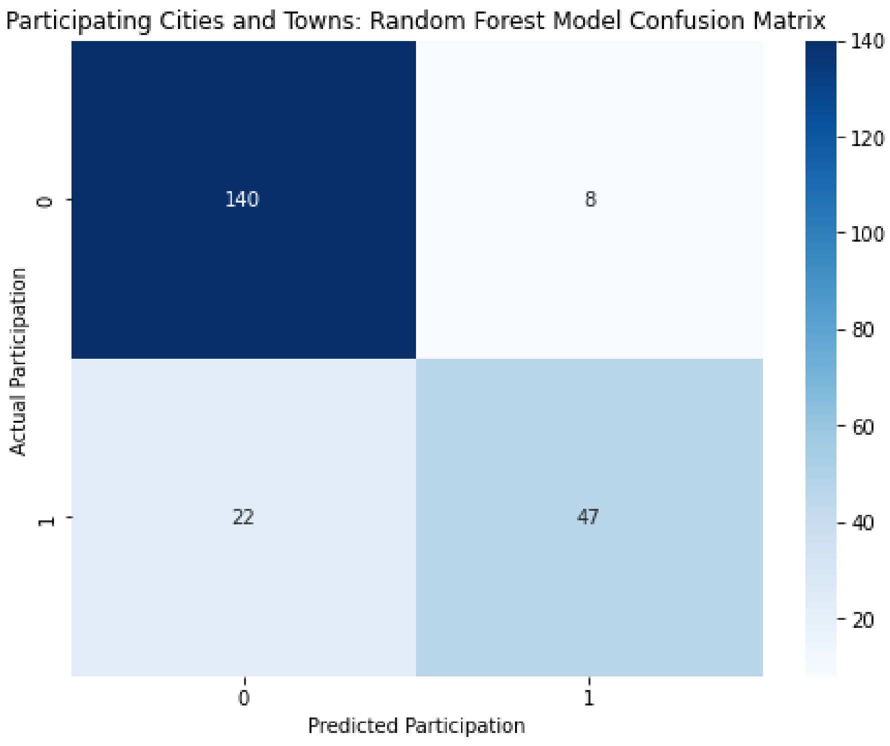


Figure 20. Participating Cities and Towns: Random Forest Model Confusion Matrix

As illustrated in Table 14, the most crucial elements of this model continue to be individuals with higher education. The two methods differ in the relative importance of the total population and revenues variables. As illustrated by the Gini method, these variables occupy the second and fourth positions, respectively. In contrast, in the permutation method, they occupy the third and second positions, respectively.

Gini method		**Permutation method**	
variable	**importance**	**variable**	**importance**
Degree_%	0.15639433	Degree_%	0.09918469
Population	0.09844177	Revenues	0.02991847
Asian_%	0.08426672	Population	0.02952854
Revenues	0.08125669	Black_%	0.02231478
Hispanic_%	0.0807266	Hispanic_%	0.02153492
Black_%	0.07694871	Male_%	0.01570365
Median_income	0.07062417	Median_income	0.0150833
Rent_%	0.05531612	Asian_%	0.01409075
Extreme_Flooding	0.05386875	Extreme_Flooding	0.01102446
Male_%	0.0535595	Rent_%	0.00983694
Significant_Flooding	0.04957897	Poverty_%	0.00896845
Employment_%	0.04789613	Disability_%	0.00840128
Disability_%	0.04582729	Significant_Flooding	0.00825948

Table 14. Importance of Variables for the Cities and Towns Participation Random Forest Model

4.4.2.3.3 Comparison

In conclusion, the random forest model once more demonstrates superior accuracy to the probit model, with a difference of approximately 4%. In both cases, the variable with the greatest deterministic influence is the percentage of the population with a high level of education. Other variables of greater relevance include the total population, the percentage of Hispanics, and revenues. The models differ in the magnitude of the variables for the disabled population, the Asian population, and the Black population. It is noteworthy that at the city and town level, there is no discernible correlation between income and the proportion of the population that has completed higher education.

4.4.2.4 Villages

The analysis yielded the following results: the percentage of the population with higher education and the income variable are highly correlated, as are the black and non-white population variables. As in previous cases, the variable indicating the percentage of households at high risk of flooding has been excluded from the study, as the extreme risk variable provides the same information.

4.4.2.4.1 Probit Model

The probit model was tested on 118 villages, with 89 correctly predicted, achieving an accuracy of 75.42%, Figure 21.

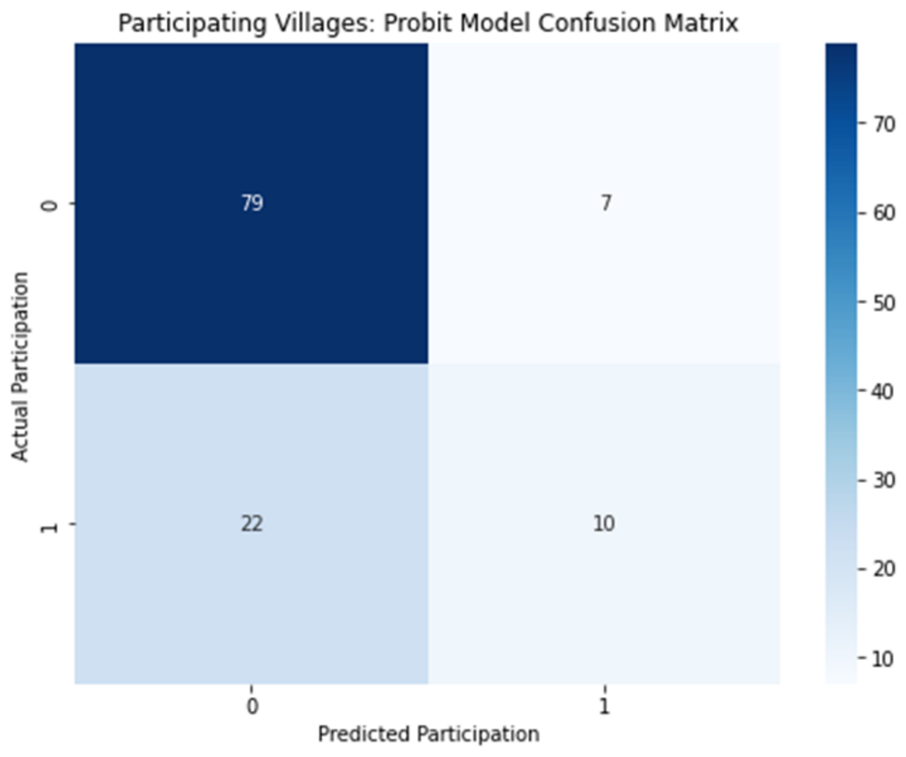


Figure 21. Participating Villages: Probit Model Confusion Matrix

At this level of analysis, the most significant variables include the percentage of graduates (which also represents income) and rental housing. Furthermore, the probability of a village not participating in the program increases in the presence of factors such as a higher proportion of the Asian population, as illustrated in Table 15.

Probit model		
variable	**coefficient**	**p-value**
Degree_%	0.700288	3.02E-11
Rent_%	0.433755	1.14E-07
Asian_%	-0.21258	2.83E-02
Disability_%	0.170679	6.47E-02

Probit model		
variable	**coefficient**	**p-value**
Hispanic_%	0.157917	1.05E-01
Black_%	0.107915	1.73E-01
Population	0.098547	1.98E-01
Significant_Flooding	0.108837	2.04E-01
Extreme_Flooding	-0.13767	2.69E-01
Poverty_%	-0.101343	2.85E-01
Revenues	-0.149142	4.85E-01
English_%	0.083866	5.06E-01
Male_%	0.027557	6.84E-01
Employment_%	-2.86E-04	9.97E-01

Table 15. Coefficients and p-values for the Villages Participation Probit Model

4.4.2.4.2 Random Forest Model

The model accuracy is 80.51%, which is noteworthy given its ability to accurately predict negative outcomes during testing. This is demonstrated in Figure 22. The model has correctly identified 81 villages that do not belong to the program, while it has incorrectly predicted that 18 villages, that are not part of the program, are members.

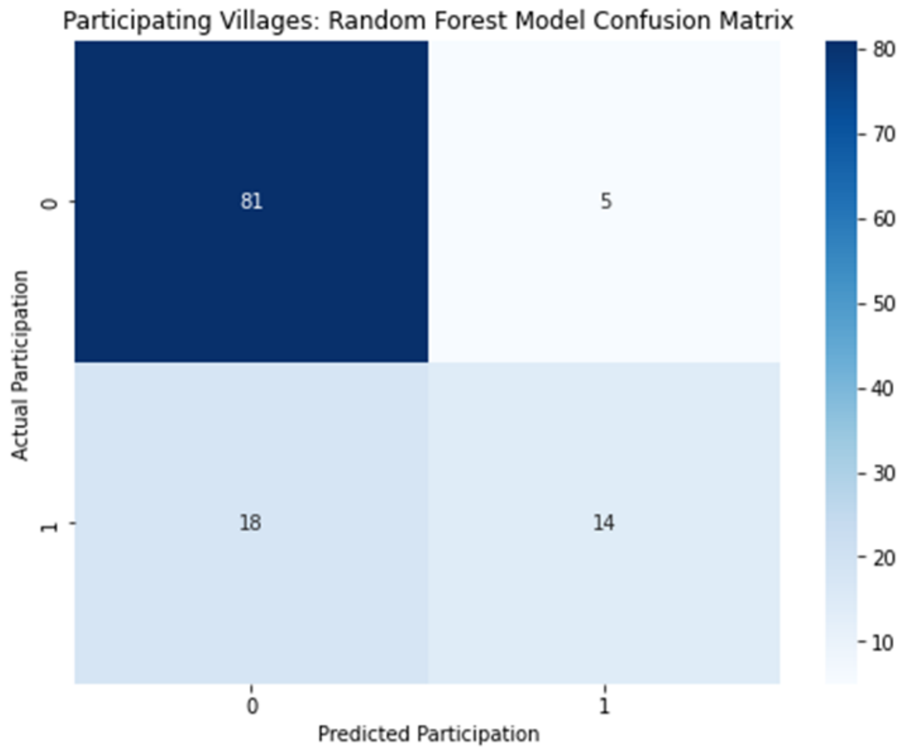


Figure 22. Participating Villages: Random Forest Model Confusion Matrix

The most crucial element in this model is the population. The relative importance of revenues, Hispanic population, rented units, and degree varies according to the method employed (Gini or permutation), as illustrated in Table 16.

Gini method		**Permutation method**	
variable	**importance**	**variable**	**importance**
Population	0.1032344	Population	0.0589007
Revenues	0.08622756	Degree_%	0.033483
Hispanic_%	0.08337593	Revenues	0.0332675
Rent_%	0.08054898	Hispanic_%	0.0229208

Gini method		**Permutation method**	
variable	**importance**	**variable**	**importance**
Degree_%	0.07880698	Rent_%	0.0185558
English_%	0.07665827	English_%	0.01455
Significant_Flooding	0.07140993	Black_%	0.0101311
Extreme_Flooding	0.06904588	Male_%	0.0100772
Black_%	0.06236813	Disability_%	0.0070954
Employment_%	0.05874271	Significant_Flooding	0.0069158
Asian_%	0.05826504	Employment_%	0.0065565
Male_%	0.05784797	Poverty_%	0.0060715
Poverty_%	0.05780571	Extreme_Flooding	0.0058559
Disability_%	0.05566251	Asian_%	0.004383

Table 16. Importance of Variables for the Villages Participation Random Forest Model

4.4.2.4.3 Comparison

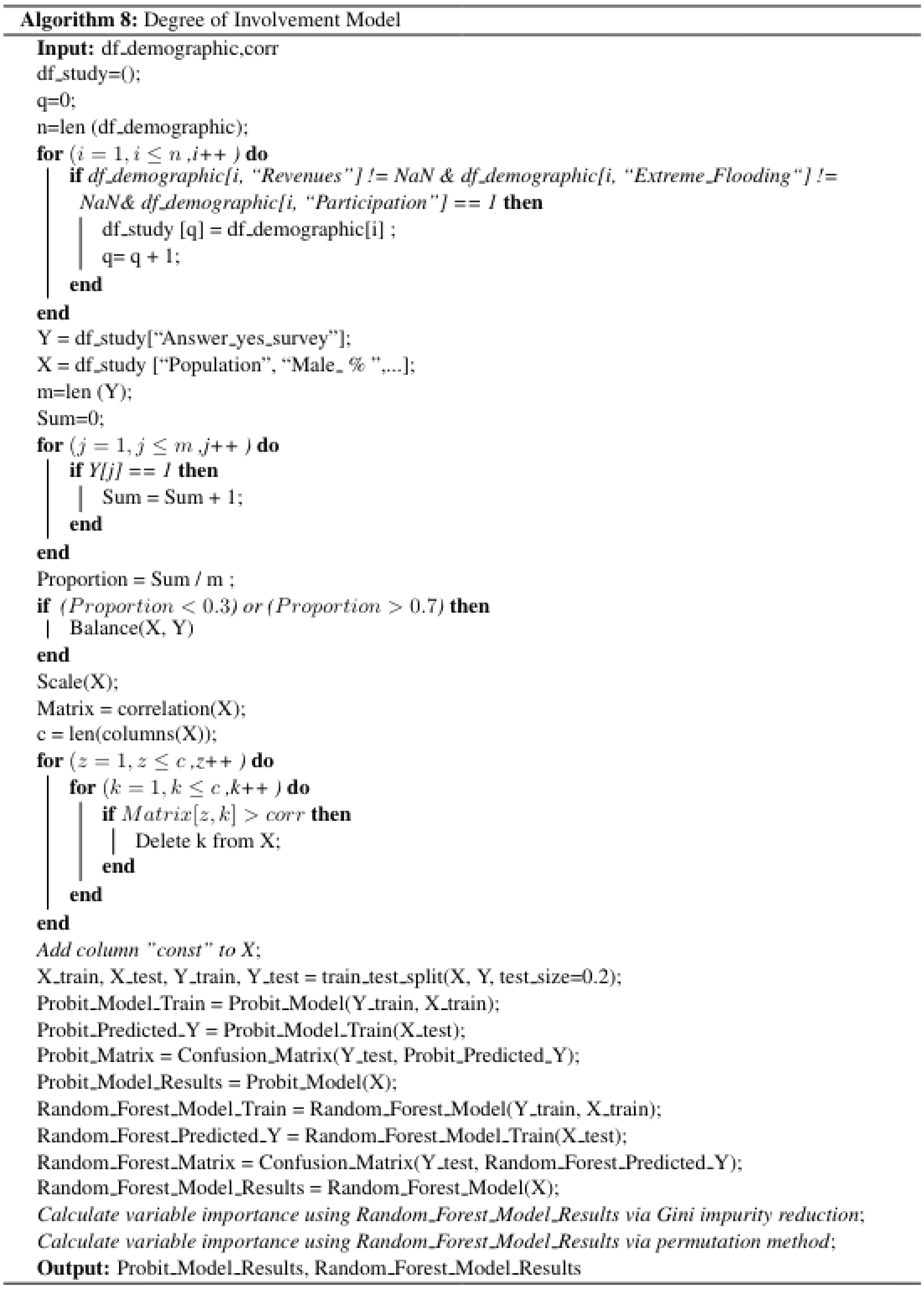
The random model demonstrated a 5% greater degree of accuracy than the probit model. The most significant variables in both models are graduates and rented housing. However, they are distinguished by the difference in the importance they assign to the following variables. The probit model places greater emphasis on the Asian population versus the total and Hispanic population, graduates (income), and revenues. In contrast, the random forest model places greater emphasis on these variables.

4.5 INVOLVEMENT MODEL

To ascertain the extent of involvement of the municipalities that are part of the program, an investigation was conducted to determine whether they were engaged in the implementation of projects or merely acting in a membership capacity, with no tangible contributions. In order to achieve this objective, the same statistical models that were previously employed to ascertain the degree of involvement of the communities have been utilized. However, the dependent variable has been modified to become a binary variable that collects information on whether the municipalities are carrying out projects (coded as 1) or not (coded as 0). The independent variables remain unchanged, comprising demographic, socioeconomic, and flood risk variables.

4.5.1 ALGORITHM

As this model is highly analogous to the preceding one, the algorithm is slightly modified, becoming Algorithm 8. This entails the removal of the rows of municipalities that are not part of the program and the alteration of the value of Y, which becomes "answer_yes_survey".



```
Algorithm 8: Degree of Involvement Model
Input: df_demographic,corr
df_study=();
q=0;
n=len (df_demographic);
for ($i = 1, i \leq n$ ,i++ ) do
    if df_demographic[i, "Revenues"] != NaN & df_demographic[i, "Extreme_Flooding"] !=
      NaN& df_demographic[i, "Participation"] == 1 then
        df_study [q] = df_demographic[i] ;
        q= q + 1;
    end
end
Y = df_study["Answer_yes_survey"];
X = df_study ["Population", "Male_ % ",...];
m=len (Y);
Sum=0;
for ($j = 1, j \leq m$ ,j++ ) do
    if Y[j] == 1 then
        Sum = Sum + 1;
    end
end
Proportion = Sum / m ;
if ($Proportion < 0.3$) or ($Proportion > 0.7$) then
    Balance(X, Y)
end
Scale(X);
Matrix = correlation(X);
c = len(columns(X));
for ($z = 1, z \leq c$ ,z++ ) do
    for ($k = 1, k \leq c$ ,k++ ) do
        if $Matrix[z, k] > corr$ then
            Delete k from X;
        end
    end
end
Add column "const" to X;
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2);
Probit_Model_Train = Probit_Model(Y_train, X_train);
Probit_Predicted_Y = Probit_Model_Train(X_test);
Probit_Matrix = Confusion_Matrix(Y_test, Probit_Predicted_Y);
Probit_Model_Results = Probit_Model(X);
Random_Forest_Model_Train = Random_Forest_Model(Y_train, X_train);
Random_Forest_Predicted_Y = Random_Forest_Model_Train(X_test);
Random_Forest_Matrix = Confusion_Matrix(Y_test, Random_Forest_Predicted_Y);
Random_Forest_Model_Results = Random_Forest_Model(X);
Calculate variable importance using Random_Forest_Model_Results via Gini impurity reduction;
Calculate variable importance using Random_Forest_Model_Results via permutation method;
Output: Probit_Model_Results, Random_Forest_Model_Results
```

Algorithm 8. Degree of Involvement Model

4.5.2 RESULTS

The correlation coefficient (0.6) defined for this model is lower than that of the participation model. This is due to the smaller number of data points, which resulted in the correlation being set at a value that yielded no significant results. Consequently, the model was deemed to be inefficient and unsuitable for this study.

4.5.2.1 Global

To facilitate the analysis, the following variables have been excluded from consideration:

- The non-white population and income variables, as well as the significant and high probability of flooding variables, were eliminated due to their positive correlation with the black population, graduates, and the extreme probability of flooding variables
- The English-only population and individuals with special needs were excluded from the analysis due to their high negative correlation with the Hispanic population (which is to be expected) and graduates (given the possibility that they may require assistance in obtaining their college degree).

4.5.2.1.1 Probit Model

The probit model of the level of participation of the municipalities exhibits an accuracy of 66.23%, with a notable capacity to accurately predict the non-implementation of projects, as illustrated in Figure 23.

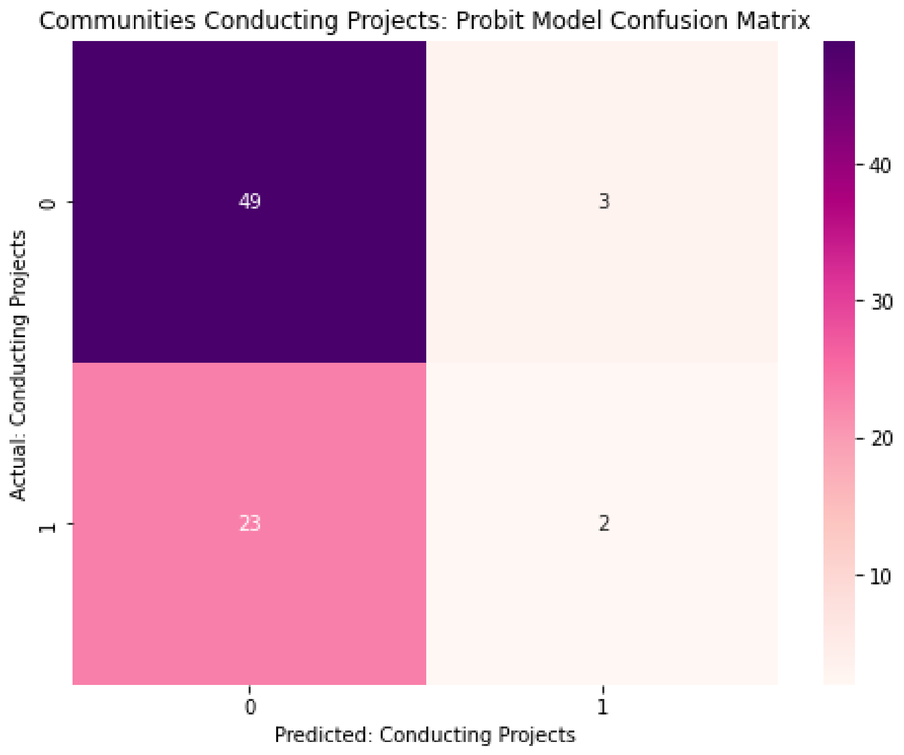


Figure 23. Communities Conducting Projects: Probit Model Confusion Matrix

Although the accuracy is high, it has not been possible to describe the relationship between the independent variables and the probability that the dependent variable increases. This is because the p-value for all of them is less than 0.005, as illustrated in Table 17.

Probit model		
variable	**coefficient**	**p-value**
Hispanic_%	-0.147117	1.15E-01
Revenues	-0.134934	1.68E-01
Population	0.094653	1.87E-01
Employment_%	-0.085533	3.16E-01

Probit model		
variable	**coefficient**	**p-value**
Rent_%	0.096217	3.22E-01
Poverty_%	0.083256	4.20E-01
Black_%	0.065019	4.90E-01
Extreme_Flooding	0.043596	5.28E-01
Degree_%	0.05966	5.54E-01
Asian_%	0.028138	7.60E-01
Male_%	0.023171	7.65E-01

Table 17. Coefficients and p-values for the Global Project Involvement Probit Model

4.5.2.1.2 Random Forest Model

Figure 24 depicts the confusion matrix for the test data set, which indicates an accuracy rate of 77.62%. As in the previous model, the correct prediction of the communities that do not carry out projects is highlighted.

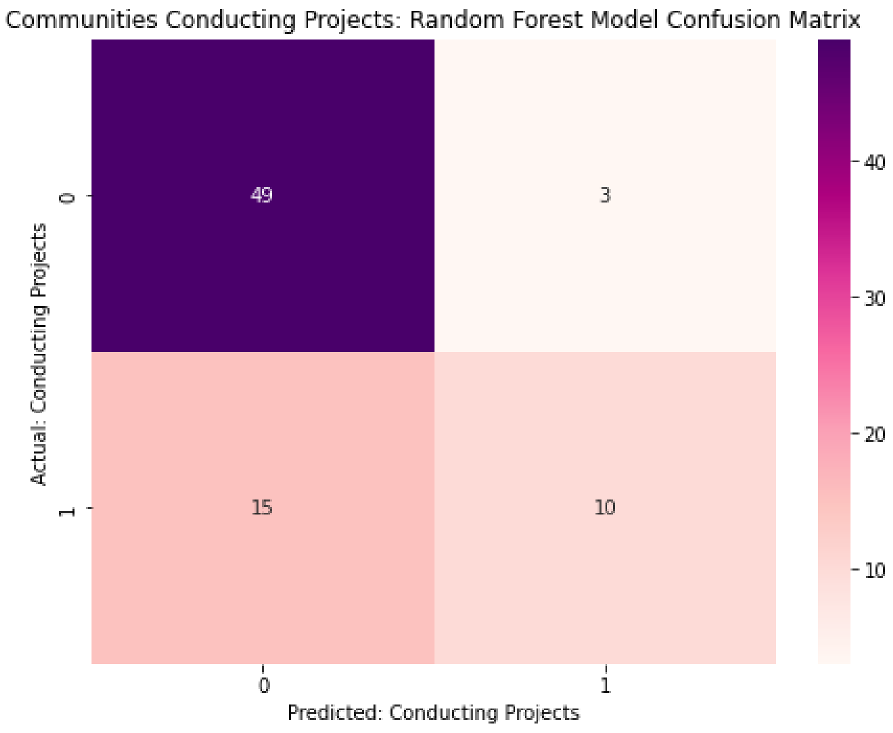


Figure 24. Communities Conducting Projects: Random Forest Model Confusion Matrix

The Gini method identifies three key variables: poverty, the black population (highly correlated with the non-white population), and the percentage of employed people. In contrast, the permutation method reveals that poverty is the variable most likely to influence behavior, followed by variables that diverge significantly from those in the Gini model, such as extreme flooding characteristics and rented housing units. These are illustrated in Table 18.

Gini method		**Permutation method**	
variable	**importance**	**variable**	**importance**
Poverty_%	0.112957635	Poverty_%	0.059214942
Black_%	0.097543829	Extreme_Flooding	0.04433095
Employment_%	0.096879044	Rent_%	0.031256384

Gini method		Permutation method	
variable	**importance**	**variable**	**importance**
Population	0.095664782	Revenues	0.031168831
Rent_%	0.093137912	Employment_%	0.029651248
Revenues	0.090900169	Hispanic_%	0.019028163
Degree_%	0.088348322	Black_%	0.017423026
Extreme_Flooding	0.086505031	Population	0.014854808
Hispanic_%	0.083468829	Asian_%	0.014358675
Asian_%	0.0782954	Male_%	0.011790457
Male_%	0.076299045	Degree_%	0.010010215

Table 18. Importance of Variables for the Global Project Involvement Random Forest Model

4.5.2.1.3 Comparison

The discrepancy in accuracy between the two models is notable, with the random forest model exhibiting a higher degree of precision. In light of these findings, the random forest model may be a more suitable choice for this particular context. In this study, the most significant variable for the random forest model is the number of individuals residing in poverty. Nevertheless, it is not feasible to ascertain whether this variable exerts a beneficial or detrimental influence, given that this information cannot be gleaned from any of the random forest model's methodologies.

4.5.2.2 Counties

In order to facilitate the analysis, the set of independent variables has been considerably simplified, as they demonstrated a correlation of more than 60% in absolute value. For example, the percentage of individuals who speak only English is inversely correlated with the percentage of the Asian population. Consequently, the former variable was excluded from further consideration. Similarly, the variable for disability (correlated with population) has been discarded.

In this case, the following variables have been discarded due to their correlation with other variables:

- Graduates due to its correlation with the Asian population
- Black and non-white population, employed persons, and median income due to its correlation with the total population
- Extreme risk of flooding due to its correlation with the high risk of flooding

4.5.2.2.1 Probit Model

The Probit model, as demonstrated in Figure 25, is only able to predict with an accuracy of 85.71% whether a municipality at that level carries out projects.

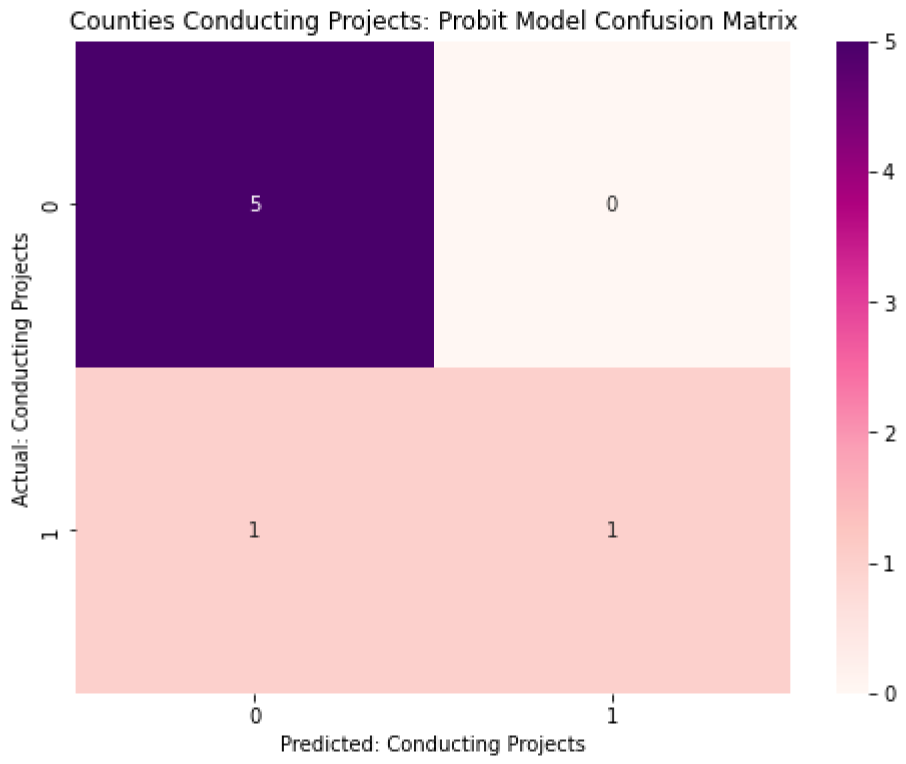


Figure 25. Counties Conducting Projects: Probit Model Confusion Matrix

As in the county participation model, no variable predicts whether or not projects are being undertaken, as demonstrated in Table 19.

Probit model		
variable	**coefficient**	**p-value**
Male_%	9.70E-01	0.072455
Poverty_%	-7.81E-01	0.081301
Rent_%	8.78E-01	0.104696
Population	7.89E-01	0.145613
Significant_Flooding	-6.63E-01	0.505002

Probit model		
variable	**coefficient**	**p-value**
Revenues	-4.04E-01	0.551694
Hispanic_%	-2.33E-01	0.565249
Asian_%	-0.221348	0.65823
Extreme_Flooding	0.116004	0.882748

Table 19. Coefficients and p-values for the Counties Project Involvement Probit Model

4.5.2.2.2 Random Forest Model

The random forest model exhibits an accuracy of 71.43%. It is noteworthy that the model incorrectly predicts a county that is carrying out projects as if it were not, as illustrated in Figure 26.

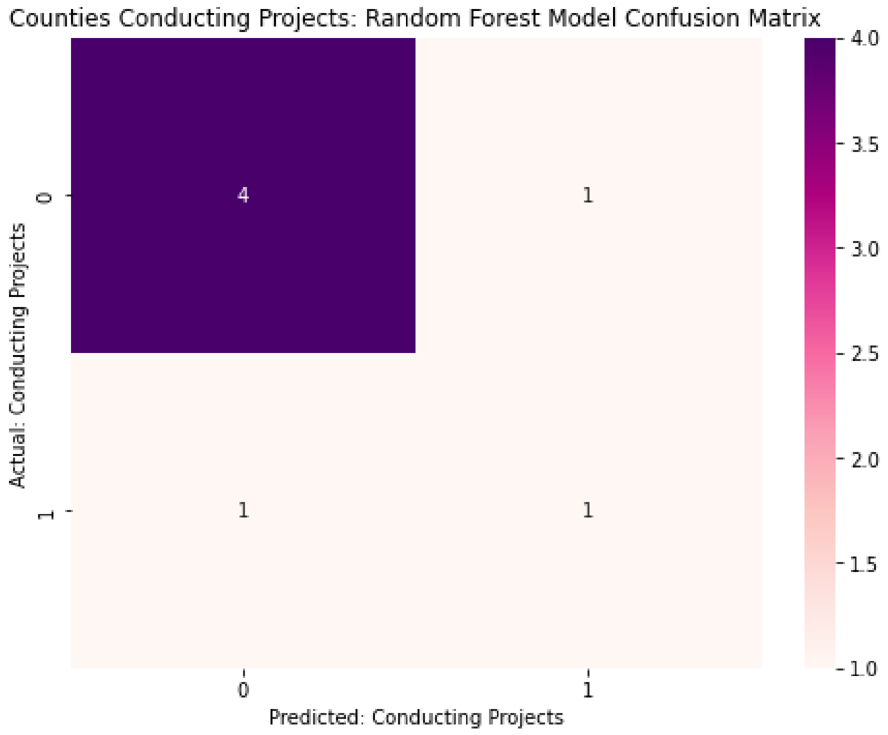


Figure 26. Counties Conducting Projects: Random Forest Model Confusion Matrix

It is noteworthy that, in this instance, the most significant variables are those of poverty and the Asian population (Table 20). This indicates that variables representing graduates and those who only speak English are also significant, given their high correlation with the Asian population. In the former case, the correlation is negative, while in the latter it is positive. The permutation method indicates that the population, rented housing, and extreme flood risk variables do not provide any information.

Gini method		**Permutation method**	
variable	**importance**	**variable**	**importance**
Asian_%	0.18327154	Poverty_%	0.05333333
Poverty_%	0.16269764	Asian_%	0.0447619

Gini method		**Permutation method**	
variable	**importance**	**variable**	**importance**
Significant_Flooding	0.15289833	Hispanic_%	0.0247619
Hispanic_%	0.12183201	Male_%	0.01904762
Male_%	0.09284449	Significant_Flooding	0.01238095
Population	0.0848876	Extreme_Flooding	0.00857143
Extreme_Flooding	0.0810714	Population	0
Rent_%	0.07245889	Rent_%	0
Revenues	0.0480381	Revenues	0

Table 20. Importance of Variables for the Counties Project Involvement Random Forest Model

4.5.2.2.3 Comparison

The findings indicate that the Probit model is more accurate (15% more precise), although it provides less information on the dependence of the independent variables on the dependent variable. The Random Forest model identified two key variables: poverty and the Asian population (comprising graduates and non-English speakers).

4.5.2.3 Cities and Towns

Due to their high correlation, the following variables were excluded:

- Non-white population and rental housing units (positively correlated with the black population)
- Median income (positively correlated with graduates)
- English speakers (negatively correlated with the Hispanic population)
- Disabled population (negatively correlated with graduates)

- High flooding (positively correlated with extreme flooding)

4.5.2.3.1 Probit Model

The model can correctly predict 30 cities or towns that are not actively participating in the program, with an accuracy of 63.83%, as highlighted in the confusion matrix in Figure 27.

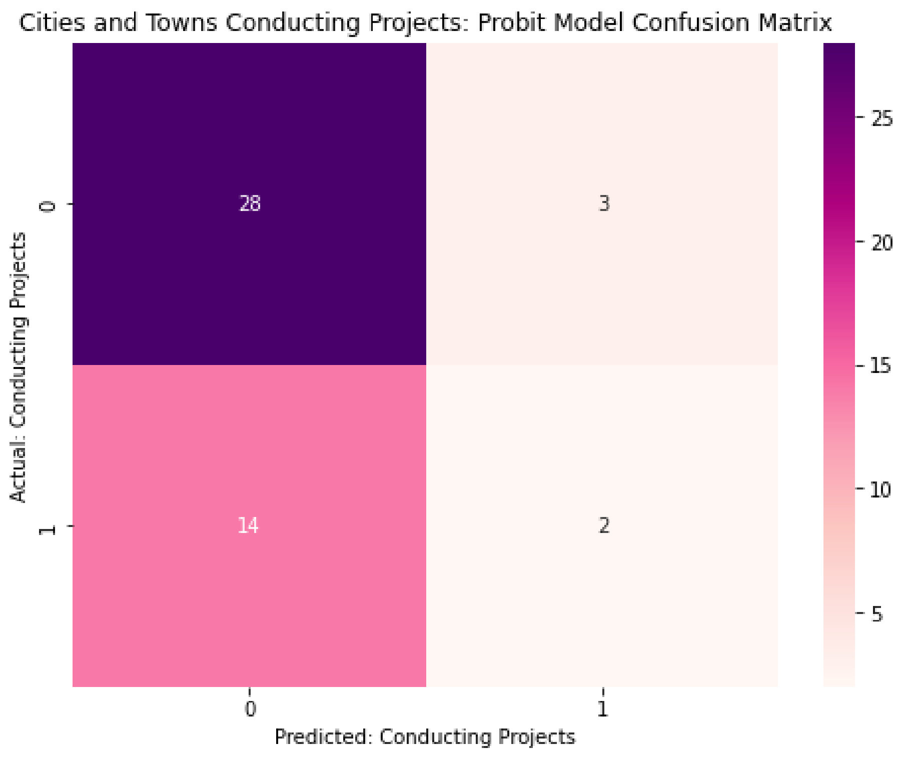


Figure 27. Cities and Towns Conducting Projects: Probit Model Confusion Matrix

In the Probit model, it is evident that the sole significant variable is the percentage of homes at significant risk of flooding, as illustrated in Table 21. This is logical, as the greater the risk of flooding, the greater the number of climate change mitigation projects that would have to be carried out.

Probit model		
variable	**coefficient**	**p-value**
Significant_Flooding	0.267845	2.55E-02
Black_%	0.249078	7.14E-02
Revenues	-0.311812	1.08E-01
Hispanic_%	-0.201024	1.44E-01
Asian_%	-0.212567	1.48E-01
Poverty_%	0.172347	2.06E-01
Population	0.129337	2.31E-01
Employment_%	-0.126813	3.34E-01
Extreme_Flooding	-0.11103	3.64E-01
Degree_%	0.113091	4.22E-01
Male_%	-0.071097	5.55E-01

Table 21. Coefficients and p-values for the Cities and Towns Project Involvement Probit Model

4.5.2.3.2 Random Forest Model

The Random Forest model correctly predicts the status of five cities and towns that are engaged in projects and 29 that are not, in contrast to the erroneous predictions for 13 communities, as illustrated in Figure 28. The model demonstrates an accuracy rate of 72.34%.

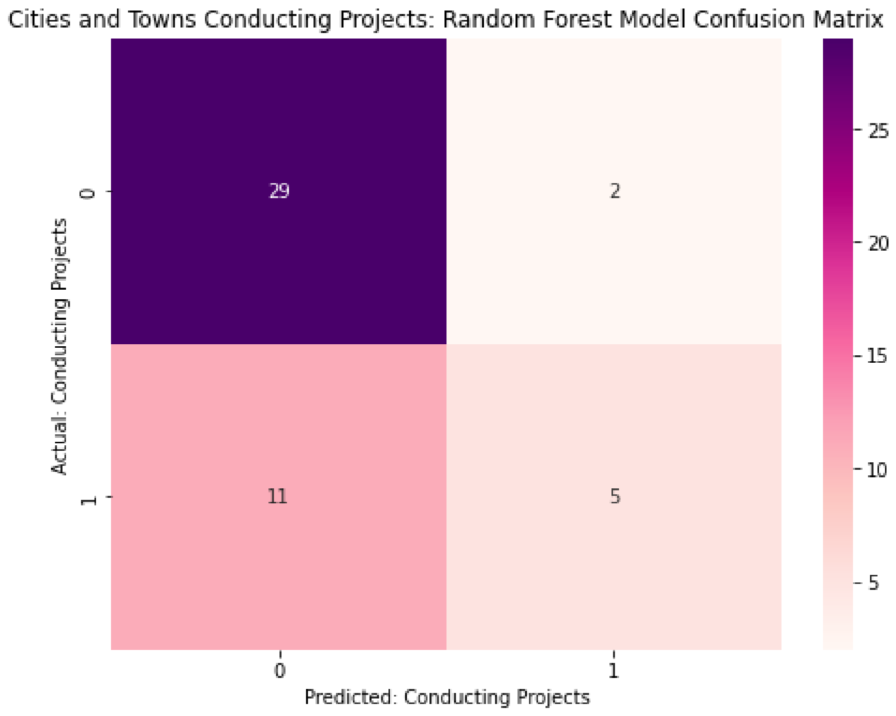


Figure 28. Cities and Towns Conducting Projects: Random Forest Model Confusion Matrix

The variable representing the number of individuals at risk of poverty is the most relevant in the Random Forest model. As illustrated in Table 22, other significant variables, including the proportion of the population in employment and the black (and non-white) population, exhibit differing levels of relevance according to the method employed. Additionally, the variables of revenue and extreme flood risk warrant mention, albeit to a lesser extent.

Gini method		**Permutation method**	
variable	**importance**	**variable**	**importance**
Poverty_%	0.1225148	Poverty_%	0.0798283
Black_%	0.1183494	Employment_%	0.0404864
Employment_%	0.1040651	Black_%	0.0246066

Gini method		**Permutation method**	
variable	**importance**	**variable**	**importance**
Revenues	0.0909807	Revenues	0.0226037
Extreme_Flooding	0.0892901	Extreme_Flooding	0.0158798
Population	0.0881894	Degree_%	0.0151645
Degree_%	0.0871953	Significant_Flooding	0.0130186
Significant_Flooding	0.083247	Male_%	0.0075823
Male_%	0.077474	Population	0.0072961
Hispanic_%	0.0694959	Asian_%	0.0052933
Asian_%	0.0691983	Hispanic_%	0.0025751

Table 22. Importance of Variables for the Cities and Towns Project Involvement Random Forest Model

4.5.2.3.3 Comparison

To predict the towns or cities that do not actively participate in the program with an accuracy of approximately 73%, the Random Forest model would be employed, given that the Probit model has a 10% lower accuracy. Upon examination of the most significant variables for both models, it becomes evident that the risk of flooding is a prominent factor. This is particularly evident in the Probit model, where it is identified as a crucial element, and in the Random Forest model, where it is identified as a highly significant factor. In addition, the Random Forest method identified three further relevant characteristics: the proportion of the population in employment, the proportion of the population who are Black, and the number of individuals at risk of poverty.

4.5.2.4 Villages

To conclude this study on the degree of involvement of the different municipalities, Probit and Random Forest models were employed for the villages. In this analysis, the Hispanic variable was excluded due to its high correlation with the population variable. The same is true for the non-white population and English speakers. Additionally, the factors of median income, disabled persons, and graduates are highly correlated. The former and the latter have a positive relationship with each other, while the latter has a negative relationship with the latter; therefore, the first two have been removed. The risk conditions have been represented solely with the extreme flood risk variable, as all of the variables are highly correlated.

4.5.2.4.1 Probit Model

Figure 29 illustrates that the Probit model correctly identifies 15 villages, of which 14 do not engage in project implementation and 1 does engage in project implementation. In contrast, the model incorrectly identifies 9 villages. The model exhibits a noteworthy capacity to predict communities that do not engage in project implementation, achieving an accuracy level of 62.5%.

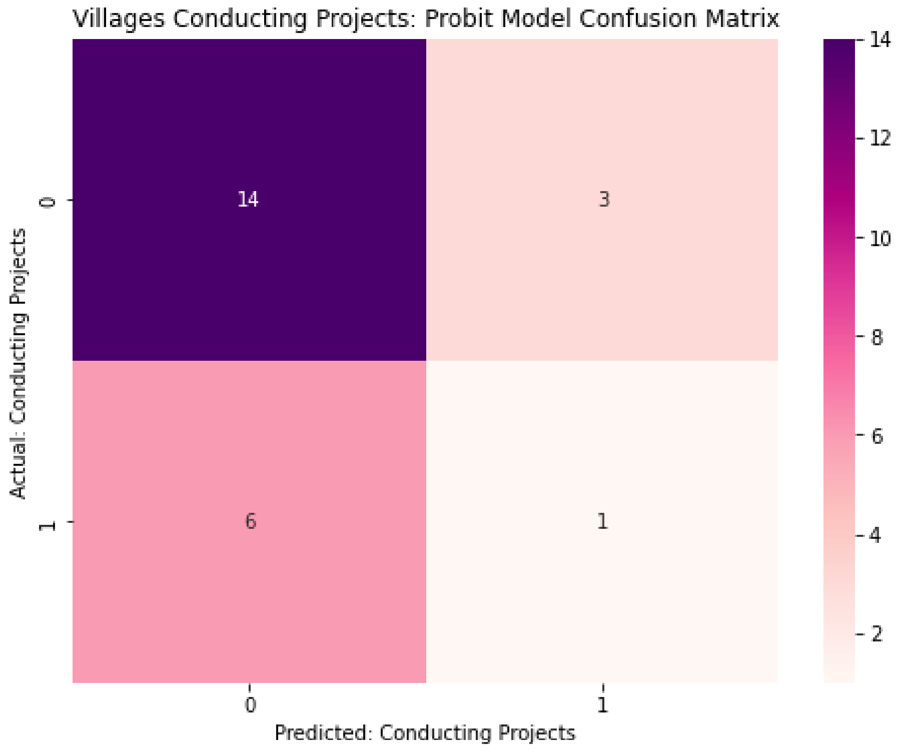


Figure 29. Villages Conducting Projects: Probit Model Confusion Matrix

It is not possible to identify any significant variables, as the p-value for all characteristics is greater than 0.05, as shown in Table 23.

Probit model		
variable	**coefficient**	**p-value**
Black_%	-0.430185	0.077565
Employment_%	-0.27708	0.08066
Population	-0.467779	0.120164
Asian_%	0.221594	0.162876
Male_%	0.175265	0.218958

Probit model		
variable	**coefficient**	**p-value**
Extreme_Flooding	0.119914	0.398415
Poverty_%	-0.163478	0.415972
Revenues	-0.049304	0.703223
Rent_%	0.035873	0.84576
Degree_%	-0.035098	0.858694
Revenues	-0.009602	0.939945

Table 23. Coefficients and p-values for the Villages Project Involvement Probit Model

4.5.2.4.2 Random Forest Model

The model in question is characterized by an incorrect prediction of five villages, where it has been erroneously assumed that they are not undertaking projects when, in fact, they are. Conversely, one village has been incorrectly identified as carrying out projects, even though it is not. The remaining municipalities have been correctly identified. A total of 16 municipalities were identified as not carrying out projects, while two were identified as carrying them out. This yielded an accuracy of 75%, as illustrated in Figure 30.

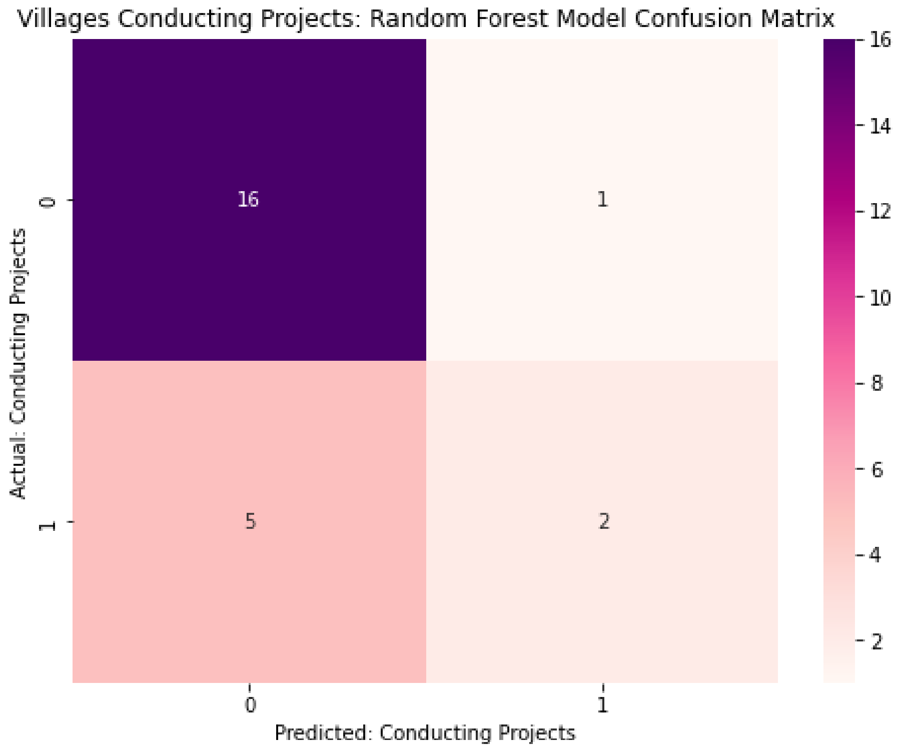


Figure 30. Villages Conducting Projects: Random Forest Model Confusion Matrix

The Gini and permutation methods concur that the most significant variable for correctly predicting project communities is revenues, as evidenced by Table 24. The Gini method identifies the percentage of the population (representing the Hispanic, non-Black, and English-speaking population) as the most relevant factor, as well as employed persons. In contrast, the permutation method highlights the percentage of the Asian population and the Black population.

Gini method		**Permutation method**	
variable	**importance**	**variable**	**importance**
Revenues	0.1189072	Revenues	0.0448276
Population	0.1146927	Asian_%	0.0387931

Gini method		Permutation method	
variable	**importance**	**variable**	**importance**
Employment_%	0.1037665	Black_%	0.0347701
Rent_%	0.1034724	Extreme_Flooding	0.0227011
Black_%	0.1027019	Employment_%	0.0221264
Asian_%	0.1013963	Rent_%	0.0195402
Poverty_%	0.1003703	Male_%	0.0149425
Male_%	0.0935786	Population	0.0140805
Extreme_Flooding	0.0894803	Poverty_%	0.0031609
Degree_%	0.0716339	Degree_%	0

Table 24. Importance of Variables for the Villages Project Involvement Random Forest Model

4.5.2.4.3 Comparison

It is finally worth noting that, at the village level, the Random Forest model outperforms the Probit model by approximately 13%. Furthermore, the variables identified can only be derived from the Random Forest model, with revenues being the most significant variable.

Chapter 5. ECONOMIC MODEL

This chapter examines the significance of demographic and climatological variables in the model for forecasting the total cost of projects implemented by municipalities. As the variable to be predicted is continuous and not binary, the probit model is not applicable. For this reason, the project costs are studied using the Ordinary Least Squares (OLS) regression model.

The dependent variable is the sum of the costs of the different projects by communities. The independent variables are the demographic and climatological variables of the communities where the projects are implemented, in addition to other project-related variables. These include the percentage of the incremental cost of a new solution compared to implementing a less adequate solution, the percentage of costs funded from federal, state, and local sources, and the primary and secondary causes of the risks being addressed. These are non-sea-level rise flooding, sea-level rise, drought, erosion, extreme heat, or extreme weather.

The model equation is as follows [31]:

$$Y = \beta_0 + \sum_{i=1}^{n} \beta_i X_i + \in$$

- Y: dependent variable
- β_0: constant or intercept of the model
- β_i: coefficient of independent variable i, indicating the change in the dependent variable as X_i increases by one unit.
- X_i: independent variable i
- $\in$: random error with expectation 0 and variance σ^2

The objective of calculating the coefficients is to minimize the sum of squared differences between the observed values of the dependent variables and the values predicted by the model. This model is considered to be efficient if it can meet the assumptions typically associated with any linear regression model [32]:

1. Linearity: The independent variables exhibit a linear relationship with the dependent variable.
2. Normality: The residuals, or differences between observed and predicted values, are distributed according to a normal distribution.
3. Homoscedasticity: refers to the homogeneity of the variance of the errors or the residuals of the model.
4. Independence: The residuals are independent of one another.

To ascertain whether the model fulfills the four conditions set out above and the results can be considered valid, several different graphs and methods are employed:

1. Linearity: refers to the plot of residuals versus predicted values. If the plot obtained adheres to a linear pattern (either straight or curved), the linearity assumption is not met, as the points should be scattered [32].
2. Normality: is assessed through a residual plot versus normal distribution (Q-Q plot). This condition is met when the points are situated close to the line of the normal distribution, which is oblique [32].
3. Homoscedasticity: a homoscedasticity plot or a square root plot of standardized residuals. If the plotted points exhibit a discernible pattern, the underlying assumption would be invalidated. Conversely, if the points are randomly scattered, the model can be applied with confidence [32].
4. Independence: determines the autocorrelation between the residuals, will be employed to assess the condition of independence. If the obtained value is approximately 2, the condition of independence is satisfied. Nevertheless, if the value falls between 0 and 2 or between 2 and 4, the condition is not met [33].

5.1 ALGORITHM

The algorithmic design for this economic model, detailed in Algorithm 9, is presented below. The initial step is to integrate the data frames containing the demographic data with those containing the survey responses. Subsequently, the project costs are classified according to the communities. The remaining variables are also grouped by first summing the variables and then calculating the average. It is crucial to highlight that the average has been calculated proportionally. This approach entails calculating the federal, state, and local costs of each project, aggregating these values, and subsequently deriving the percentages.

Once the initial data set for the model is available, the data is divided into two distinct categories: X, which represents the independent variables, and Y, which represents the dependent variable. The data are then standardized, and highly correlated variables are removed from X. In addition, the logarithm has been applied to Y due to the large variability of the values. The OLS model is constructed using the training and test data, as previously described. Subsequently, the four conditions are subjected to testing using the previously defined graphs and statistical methods. The coefficient of determination (R2), the quadratic error (MSE), and the mean absolute error (MAE) are employed to assess the precision of the economic model. Once the results have been analyzed, the model is rebuilt for the entire data set, and the conditions of the linear regression model are once again verified. Finally, the most pertinent independent variables for the model are identified.

```
Algorithm 9: Project Cost
  Input: df_survey_yes, df_demographic
  df_final_aux =Copy of df_survey_yes;
  df_final_aux = Merge(df_final_aux, df_demographic, on=['Muni Name', 'County', 'Class'];
  Delete rows from df_final_aux where columns= NaN;
  n=len(df_final_aux);
  for ($i = 1, i \leq n$ ,i++ ) do
      df_final_aux (i; "Fed_Funded") = df_final_aux (i; "Cost") * df_final_aux (i; "Pct Federal") / 100;
      df_final_aux (i; "State_Funded") = df_final_aux (i; "Cost") * df_final_aux (i; "Pct State") / 100;
      df_final_aux (i; "Local_Funded") = df_final_aux (i; "Cost") * df_final_aux (i; "Pct Local") / 100;
      df_final_aux (i; "Cost_CC") = df_final_aux (i; "Cost") * df_final_aux (i; "Pct Cost CC") / 100;
  end
  df_grouped = Sum and mean data by ['Muni Name', 'County', 'Class'] for all projects in
    df_final_aux;
  df_grouped ("Pct Federal") = (df_grouped ("Fed_Funded") / df_grouped ("Cost")) * 100;
  df_grouped ("Pct State") = (df_grouped ("State_Funded") / df_grouped ("Cost")) * 100;
  df_grouped ("Pct Local") = (df_grouped ("Local_Funded") / df_grouped ("Cost")) * 100;
  df_grouped ("Pct Cost CC") = (df_grouped ("Cost_CC") / df_grouped ("Cost")) * 100;
  Delete columns=['Fed_Funded', 'State_Funded', 'Local_Funded', 'Cost_CC'] of df_grouped;
  df_grouped ("LogCost") = log(df_grouped ("Cost"));
  Y = df_grouped ("LogCost");
  X = Copy of df_grouped;
  Delete columns=['Fed_Funded', 'State_Funded', 'Local_Funded', 'Cost_CC'] of X;
  Scale(X);
  Matrix = correlation(X);
  c = len(columns(X));
  for ($z = 1, z \leq c$ ,z++ ) do
      for ($k = 1, k \leq c$ ,k++ ) do
          if $Matrix[z, k] > corr$ then
              Delete k from X;
          end
      end
  end
  Add column "const" to X;
  X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2);
  Model_Train = OLS_Model(Y_train, X_train);
  Calculate residuals of Model_Train;
  Predicted_Y = Model_Train(X_test);
  Plot: residuals vs predicted values;
  Plot: residuals vs normal distribution (Q-Q plot);
  Plot: homoscedasticity or standardized residuals square root plot;
  Durbin-Watson statistic;
  MSE_test = Calculate mean_squared_error(Y_test, Predicted_Y);
  $R^2$_test = Calculate$R^2$_score(Y_test, Predicted_Y);
  MAE_test = Calculate mean_absolute_error(Y_test,Predicted_Y);
  Model_Results = OLS_Model(Y, X);
  Calculate residuals of Model_Results;
  Plot: residuals vs predicted values;
  Plot: residuals vs normal distribution (Q-Q plot);
  Plot: homoscedasticity or standardized residuals square root plot;
  Durbin-Watson statistic;
  Output: Model_Results
```

Algorithm 9. Project cost

5.2 RESULTS

5.2.1 TRAINING MODEL

First, as was done in the other models, highly correlated variables (correlation greater than 0.6) have been eliminated. In this instance, the variables that have been eliminated are listed below, accompanied by the correlated variable and the type of relationship:

- The non-white population is positively correlated with the black population.
- Individuals with high levels of education (positively correlated with the Asian population).
- Populations experiencing poverty and the disabled population (negatively correlated with income).
- English-speaking populations (negatively correlated with the Asian population).
- The proportion of areas at high risk of flooding and the proportion of areas at extreme risk of flooding (positively correlated).

As illustrated in Figure 31, the discrepancy between the observed values and those predicted by the model does not exhibit a linear pattern. The residuals are randomly distributed, indicating that the first assumption of the linear regression model is met: linearity, the relationship between the independent variables, and the dependent variable.

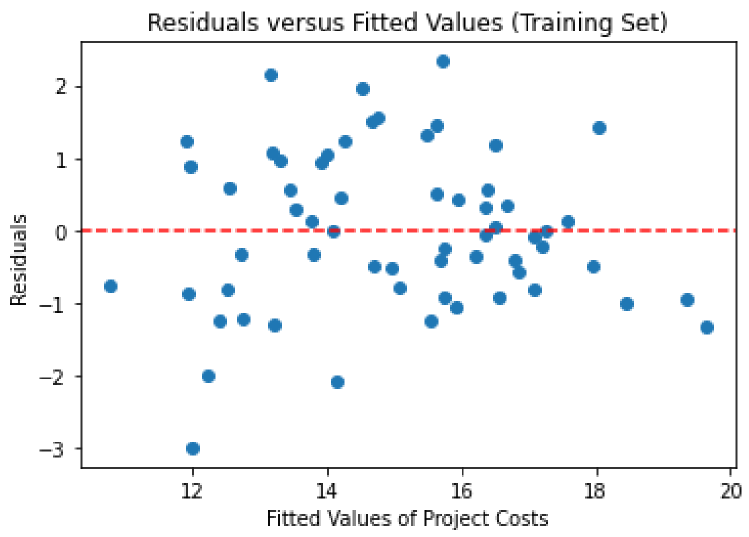


Figure 31. Residuals versus Fitted Values (Training Set)

Figure 32 verifies the assumption of normality, as evidenced by the large number of points that follow a normal distribution, being close to the diagonal. It can therefore be stated that the residuals are normally distributed.

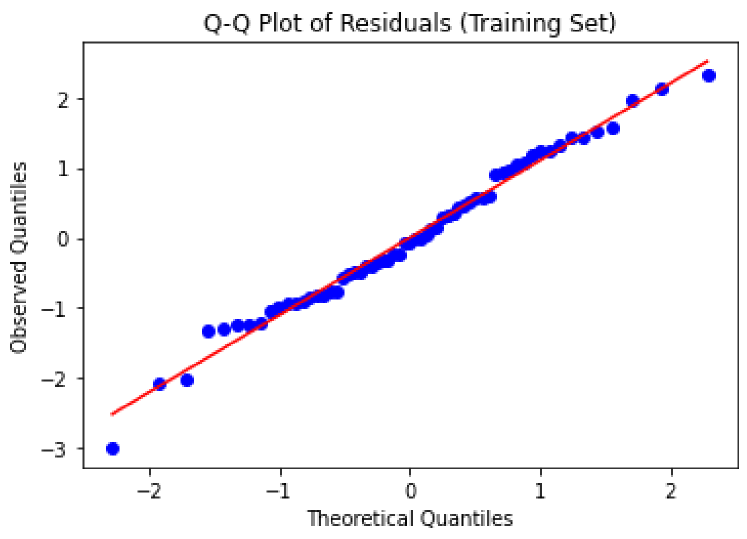


Figure 32. Q-Q Plot of Residuals (Training Set)

The graph in Figure 33 demonstrates that the dispersion of the residuals does not exhibit a discernible pattern, such as a curve. This indicates that the homoscedasticity condition is met, or that the variance of the residuals is homogeneous.

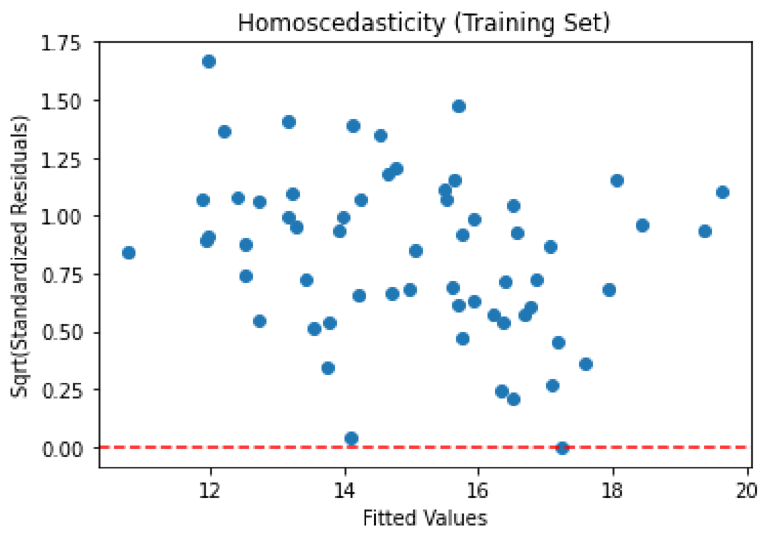


Figure 33. Homoscedasticity (Training Set)

To complete the verification of the conditions of the linear regression methods, the value obtained from the Durbin-Watson method is 1.926, which is very close to 2. This indicates that the condition of independence is fulfilled, and therefore the residuals are independent of each other.

The mean square error (MSE) is employed as a metric for gauging the mean of the squared errors [34]. The model presents an error of 13.95, which is a relatively high figure. This indicates that there is considerable variability in project costs, suggesting a lack of information to make a more precise estimation. Such discrepancies could be justified by the considerable differences in project costs across communities. While some communities allocate hundreds of dollars, others allocate millions.

The value of the coefficient of determination (R^2) is employed to assess the overall accuracy of the model [35]. When applied to this model, the coefficient of determination is 0.255, which indicates that 25.5% of the variability in the data is explained by the model. This value is below the ideal value of 1, which would indicate that the model is unable to perfectly predict all values.

The mean absolute error (MAE) is a measure of the distance between the predicted values and the actual values in absolute terms [36]. The obtained value is 2.38, a relatively low figure that indicates that the model predictions are like the actual values. This suggests that the accuracy is adequate in terms of the average deviation.

Finally, to summarize the accuracy of the model on the training and test data, it can be stated that the model has an acceptable accuracy according to the mean absolute error. Nevertheless, the mean square error and the coefficient of determination are relatively low, indicating that the model does not fit perfectly due to the high variability of the data. It is important to consider this aspect when studying the significant variables, as while the relevant variables have been identified, there is a possibility that some significant variables may have been omitted or that some irrelevant variables may have been included.

5.2.2 GENERAL MODEL

Once the model has been constructed, the conditions necessary to apply the OLS model are re-examined, specifically the four assumptions of the model. Figure 34 demonstrates that no discernible pattern exists, thereby indicating that the linearity assumption is met. This implies that there is a linear relationship between the independent variables and the dependent variable.

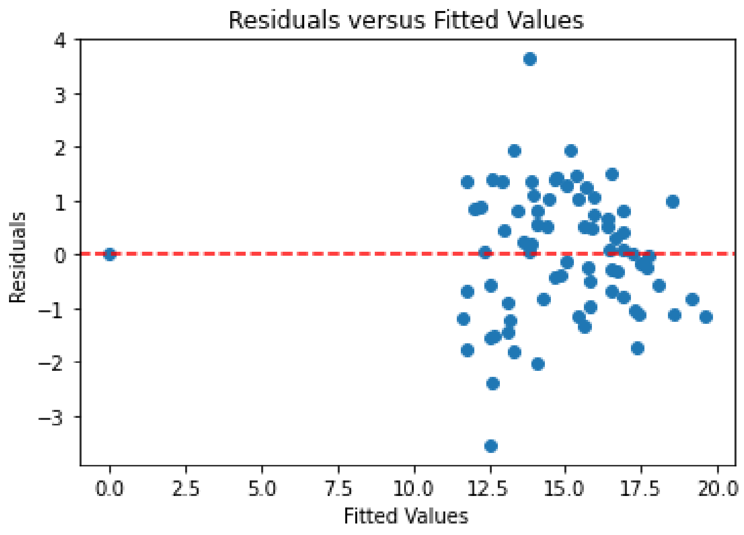


Figure 34. Residuals versus Fitted Values

As was the case in the training and test set model, the majority of the points are aligned with the red line, which represents the normal distribution, as can be seen in Figure 35.

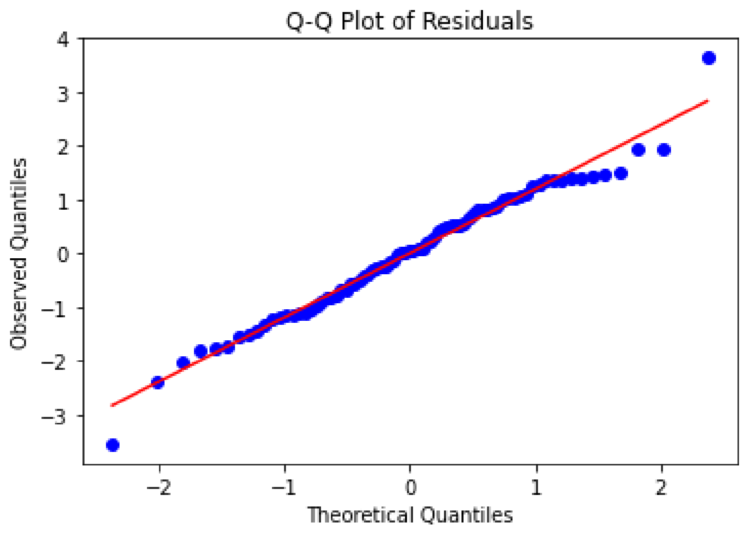


Figure 35. Q-Q Plot of Residuals

The homoscedasticity plot, illustrated in Figure 36, does not reveal a discernible pattern, indicating that the variance of the residuals is homogeneous. This implies that the homoscedasticity condition is satisfied.

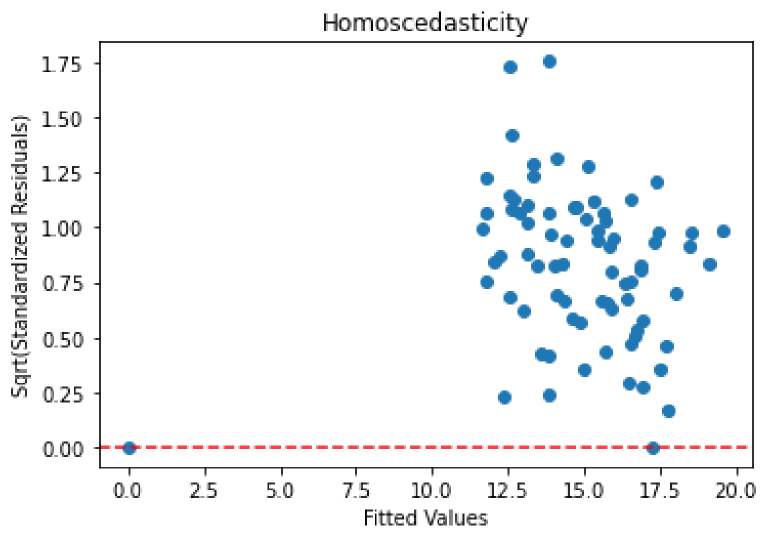


Figure 36. Homoscedasticity

The Durbin-Watson statistic method yielded a value of 2.29, which is very close to 2. This indicates that the model meets the conditions of the linear regression method, thereby confirming the adequacy of the OLS model for the economic data set of the projects.

Finally, the most significant variables of the model have been identified in Table 24, which demonstrates that a higher percentage of federally financed investment is associated with an increase in project cost. A similar phenomenon occurs with the percentages of state and local investment. This is logical, given that if the cost of a project is exceedingly high, without aid it would be impossible to carry it out.

It is notable that the percentage of rental housing also exerts a positive influence on the cost of the projects. Finally, it is noteworthy that the majority of the projects were primarily the result of flooding, which is not surprising given the significant impact this has on the economy.

variable	**coefficient**	**p-values**
Pct Federal	3.518873	6.65E-10

variable	**coefficient**	**p-values**
Pct State	3.9043608	2.22E-09
Pct Local	4.0323441	2.34E-09
Rent_%	1.3898433	0.0002617
flooding (not related to sea-level rise)_1	1.1575897	0.0014834

Figure 37. Coefficients and p-values for the General Model

Chapter 6. PROJECT INITIATIVE

In order to conclude this project, it has been determined that it would be beneficial to present potential recommendations for future initiatives, with the intention of not merely examining the significance of demographic, social, economic, and flood risk data and the projects that are being conducted to mitigate the natural disasters caused by climate change. Instead, these recommendations are intended to serve as a tool that can assist other communities.

To achieve this objective, the unsupervised classification algorithm known as k-means has been employed. This algorithm partitions the data set into several groups or clusters (k) according to the similarities of the data points. The algorithmic procedure is as follows [37]:

1. The number of clusters (k) is defined as follows: The number of clusters (k) into which the data set is to be classified is established.
2. Initialization of the centroids: For each cluster, a point, called a centroid, is randomly established as the central or average point.
3. Iteration: the steps of data assignment and centroid definition are repeated until the variation of the centroids is not significantly modified or the maximum number of iterations is reached.
4. Iteración: se repiten los pasos de asignación de datos y definición de centroides hasta que la variación de los centroides no se modifique significativamente o se alcance el número máximo de iteraciones.
5. The optimal number of clusters can be determined through the elbow method. To achieve this, a maximum number of clusters is first defined, which is typically ten. Subsequently, the k-means algorithm is executed, and for each cluster, the sum of the squared errors (previously defined in the previous chapter) is calculated as the sum of the squared distances from each point of the group to the centroid of the nearest cluster [38].
6. Finally, the results are graphed, with the horizontal axis representing the number of clusters and the vertical axis representing the sum of the squared errors. The point at

which a slow decrease is observed, known as the elbow, represents the optimal number of clusters.

6.1 ALGORITHMS

The model comprises two algorithms, designated as Algorithm 10 and Algorithm 11. It should be noted that, as previously stated, this model has been designed for the global data set and then for each geographical level. The algorithms for the general case are presented below.

Algorithm 10 has been defined to apply the elbow method and identify the optimal number of clusters. To achieve this objective, the data frame designated as df_demographic, which encompasses all municipalities, and a list designated as variables, which includes the most significant variables derived from the model of participation and degree of involvement, are introduced into the algorithm.

The data frame of df_demographic is copied into df_demographic_proposal, and the rows where no numbers appear are eliminated, indicating that there is no information for that municipality. Subsequently, the information on whether or not the municipalities are carrying out projects is stored in a list, thereby eliminating this column from the data frame.

The data is standardized, and the model is initiated with the cluster number set to 2. The model is then adjusted for that number, and the sum of squared errors is calculated and stored in a vector. This process is repeated iteratively until the requisite number of clusters, ten in this case, is reached. Finally, the values of the sum of mean squared errors are plotted to define the optimal number of clusters. The output of this algorithm is a standardized data frame of the municipalities, along with a list indicating whether each municipality is engaged in project implementation.

```
Algorithm 10: Elbow
Input: df_demographic, variables
df_demographic_proposal = Copy of df_demographic [variables];
Delete rows from df_demographic_proposal where columns= NaN;
df_demographic_proposal = Copy of df_demographic [variables];
answer_yes_survey_list= df_demographic_proposal ["Answer_yes_survey"];
Delete from df_demographic_proposal column ='Answer_yes_survey';
Scale(df_demographic_proposal);
SSE = [ ];
i=0;
k_end=11;
for ($k = 2, k \leq k\_end, k{+}{+}$ ) do
    k_model = Model KMeans(k,df_demographic_proposal);
    SSE [i] =Calculate Sum of Squared Error of k_model;
    i=i+1;
end
Plot: k vs SSE;
Output: df_demographic_proposal, answer_yes_survey_list
```

Algorithm 10. Elbow method

Once the number of condos has been established, Algorithm 11 is run, which has as input variables the output variables of the previous algorithm, in addition to the variable of the optimal number of clusters established in the first algorithm.

The model is then adjusted to accommodate the input data set, after which the number of clusters to which each community belongs is predicted. Once this step is complete, a column indicating whether the community contributes to climate change projects is added to the data frame. Subsequently, the data frame is traversed to identify the participating communities within the same cluster. The Euclidean distance is employed to determine the order of recommendation. That is, the communities that are the closest in terms of their characteristics are considered to be more similar, and thus, they are the first to be fixed. Finally, a new data frame is generated, in which the community name is included, along with its geographical level and the county to which it belongs. The following columns include the municipalities in which the aforementioned communities should be set.

```
Algorithm 11: Clustering
  Input: df_demographic_proposal, answer_yes_survey_list,optimal_clusters
  k_model = Model KMeans(optimal_clusters,df_demographic_proposal);
  df_demographic_proposal ["Cluster"] = Predict cluster from k_model;
  df_demographic_proposal ["Answer_yes_survey"] = answer_yes_survey;
  engaged_communities = df_demographic_proposal[df_demographic_proposal['Answer_yes_survey']
    == 1];
  proposal = [ ];
  recommendation = [ ];
  n = len(df_demographic_proposal);
  for ($i = 1, i \leq n$ ,i++ ) do
      df_demographic_proposal_cluster = df_demographic_proposal[i];
      cluster = a ["Cluster"];
      engaged_communities_cluster = engaged_communities[engaged_communities["Cluster"] ==
        cluster];
      Calculate the Euclidean distances between df_demographic_proposal[i] and
        engaged_communities_cluster;
      Sorting engaged_communities_cluster by proximity;
      Create a dictionary recommendation including not_engaged_community_cluster and
        engaged_communities_cluster;
      proposal [i] = recommendation;
  end
  Output: proposal
```

Algorithm 11. Clustering

6.2 RESULTS

6.2.1 GLOBAL

For the variables presented in Table 25, the optimal number of clusters for a global analysis of the communities has been determined to be four, as this is where the elbow is located in the elbow method graph, as illustrated in Figure 38. Subsequently, the reduction in the sum of squared errors becomes less pronounced.

Variables
Poverty_%
Employment_%

Variables
Extreme_Flooding
Black_%
Rent_%

Table 25. Key Variables for Global Initiative

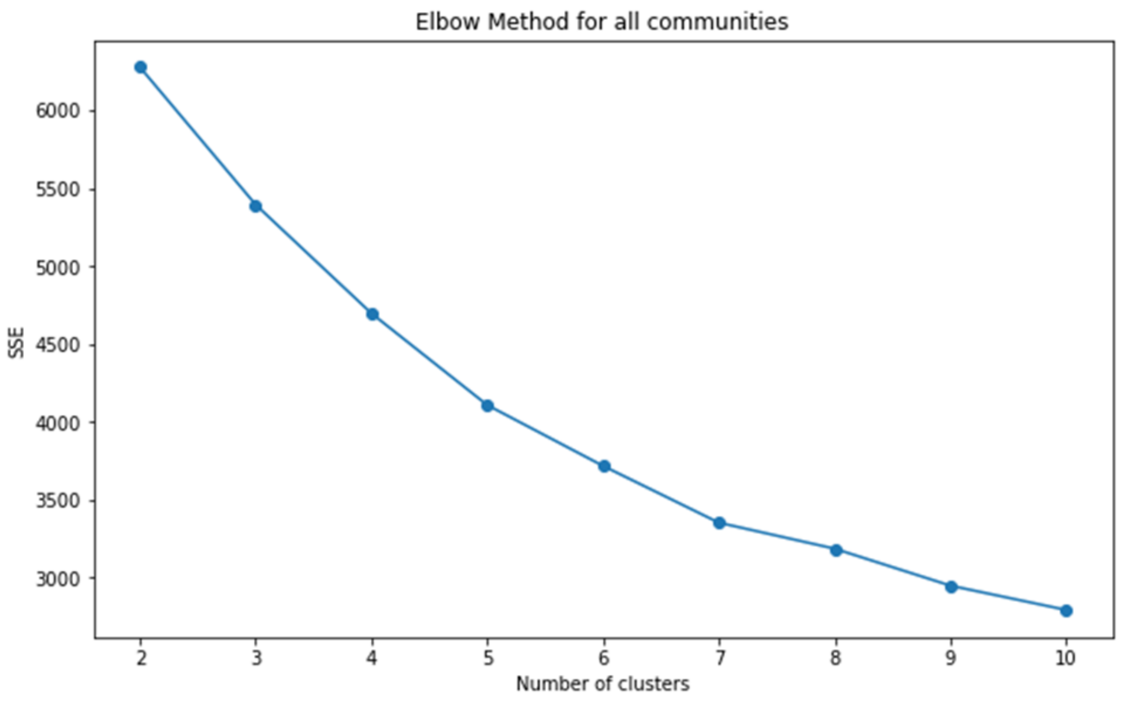


Figure 38. Elbow method for all Communities

Due to the considerable number of municipalities, a selection of communities obtained from the algorithm has been included in Table 26, with a reduced number of communities to be based on. In certain instances, up to 39 recommendations have been generated for specific municipalities. Upon examination of the summarized results, it can be concluded that the necessity for communities to be situated at the same geographic level to carry out projects is not a prerequisite. For instance, although the initial recommendation for the village of Adams would be to undertake projects analogous to those in Springville, it would also be advantageous to consider a town.

Community	**Recommendation1**	**Recommendation2**	**Recommendation3**
Adams, Town, Jefferson	New Lebanon, Town, Columbia	Cherry Valley, Village, Otsego	Ghent, Town, Columbia
Adams, Village, Jefferson	Springville, Village, Erie	Amenia, Town, Dutchess	Oneida, City, Madison
Addison, Town, Steuben	Dryden, Town, Tompkins	Sullivan, County, Sullivan	Seneca Falls, Town, Seneca
Albany, City, Albany	Poughkeepsie, City, Dutchess	Syracuse, City, Onondaga	Rochester, City, Monroe
Albany, County, Albany	Schenectady, County, Schenectady	Erie, County, Erie	Catskill, Village, Greene

Table 26. Recommendations for Community Initiatives

6.2.2 COUNTIES

To facilitate the analysis of the counties, the most pertinent variables have been included, in addition to two other variables that, although relevant, are not of the same level of importance. As illustrated in Table 27, the variables employed in this analysis differ from those utilized in the global method. This is due to the fact that the analysis is more specific to the counties.

Variables
Poverty_%

Variables
Asian_%
Hispanic_%
Male_%
Significant_Flooding

Table 27. Key Variables for County Initiative

The Figure 39 illustrates that the elbow is at cluster number 4, as evidenced by the significant reduction in mean square error (SSE) from 2 to 4. Consequently, the reduction of this error is less pronounced. This point indicates that four clusters represents the optimal number, as the addition of further clusters does not result in a significant improvement in error minimization.

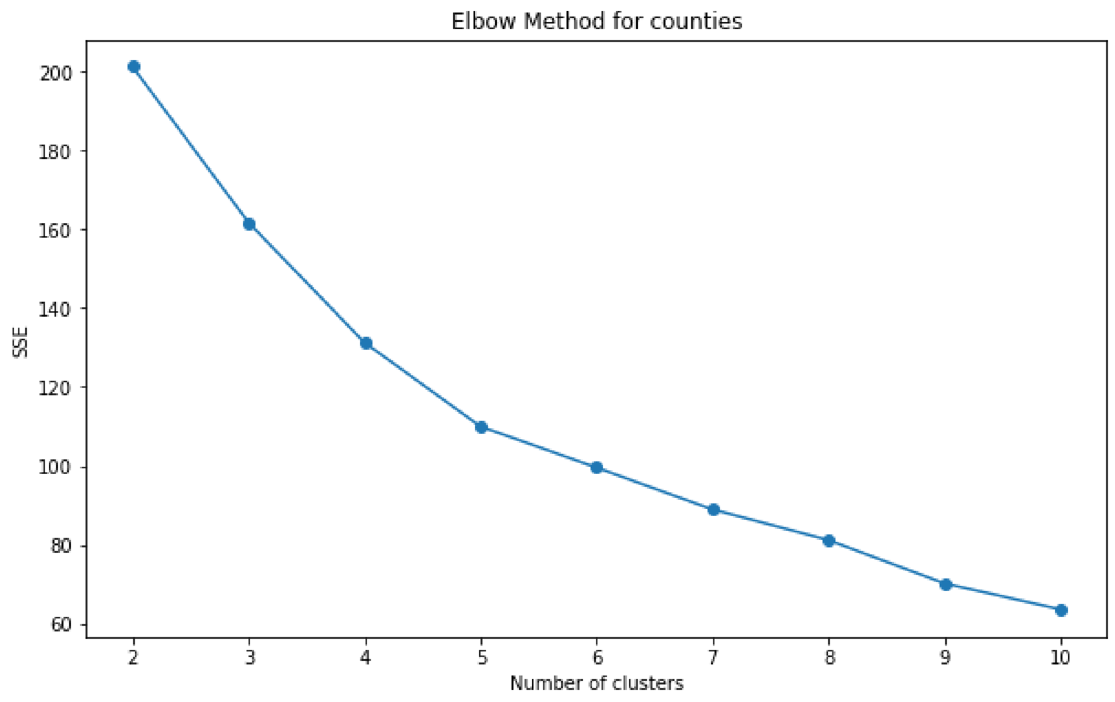


Figure 39. Elbow method for Counties

Table 28 presents the recommendations for each county, with a maximum of three recommendations per county. However, this is a considerably high number, given that there are only six counties undertaking projects. Moreover, the recommendations are based on similar characteristics among the counties. A total of 18 counties have been identified as having no recommendations. For example, Hamilton, Lewis, and Sullivan have no recommendations. The first two counties are not engaged in project implementation, and thus it would be prudent to consider communities such as cities, towns, or villages, given that there are no counties with comparable characteristics. In the case of Sullivan, it is not a concern, as they are indeed undertaking projects. Nevertheless, as previously stated, they may wish to consider the global recommendations, which, although less accurate, are preferable to the absence of any recommendations.

Dutchess County is undertaking projects, yet it also has recommendations to consider in other counties. This is a positive aspect, as it indicates that even though the county is currently engaged in projects, it has the potential to undertake a greater number of projects.

Community	**Recommendation1**	**Recommendation2**	**Recommendation3**
Albany, County, Albany	Schenectady, County, Schenectady	Erie, County, Erie	Columbia, County, Columbia
Dutchess, County, Dutchess	Suffolk, County, Suffolk		
Hamilton, County, Hamilton			
Herkimer, County, Herkimer			

Community	**Recommendation1**	**Recommendation2**	**Recommendation3**
Jefferson, County, Jefferson	Sullivan, County, Sullivan		
Lewis, County, Lewis			
Sullivan, County, Sullivan			
Fulton, County, Fulton	Columbia, County, Columbia	Erie, County, Erie	Schenectady, County, Schenectady

Table 28. Recommendations for County Initiatives

6.2.3 CITIES AND TOWNS

The number of variables employed in the formulation of recommendations for cities and towns is the same as that used in the case of counties. Table 29 presents the variables that will be utilized in this instance.

Variables
Black_%
Poverty_%
Significant_Flooding
Employment_%

Table 29. Key Variables for City and Town Initiative

Although it may be challenging to discern the elbow in the graph depicted in Figure 40, it has been determined that the optimal number of clusters is four. This is evidenced by the significant decrease in mean squared errors (SSE) between three and four clusters. The reduction in mean squared errors (SSE) between 4 and 5 clusters is less pronounced and continues to diminish with the addition of further clusters. Consequently, it can be posited that the optimal number of clusters is four.

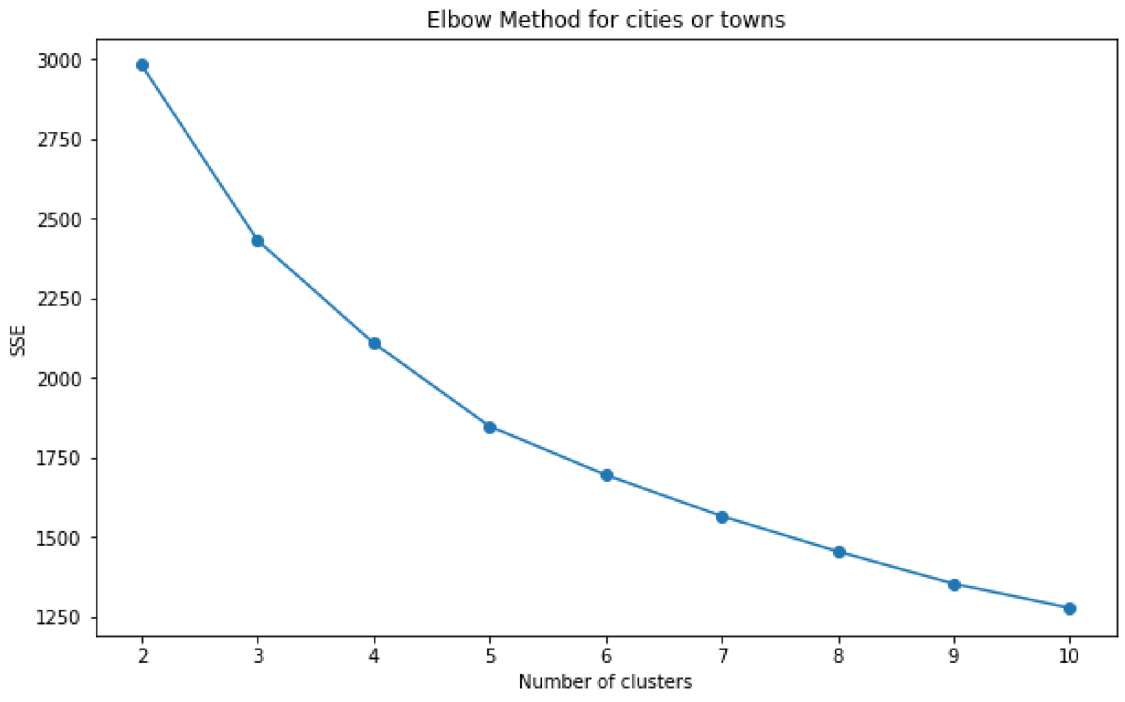


Figure 40. Elbow method for Cities and Towns

Table 30 presents a demonstration of the results obtained for recommendations for municipalities that have been classified as cities or towns. The number of recommendations has been modified in comparison to counties, as it has been determined that up to 20 recommendations can be achieved. Moreover, no municipality has been identified without a recommendation, which is a remarkably high and beneficial number. This offers a diverse array of potential avenues for cities and towns, in contrast to the observations made in counties.

Community	Recommendation1	Recommendation2	Recommendation3
Albany, City, Albany	Poughkeepsie, City, Dutchess	Syracuse, City, Onondaga	Rochester, City, Monroe
Berne, Town, Albany	Ancram, Town, Columbia	Clinton, Town, Dutchess	Ghent, Town, Columbia
Bethlehem, Town, Albany	Somers, Town, Westchester	Ghent, Town, Columbia	Canandaigua, Town, Ontario
Coeymans, Town, Albany	Putnam Valley, Town, Putnam	New Lebanon, Town, Columbia	Hillsdale, Town, Columbia
Cohoes, City, Albany	Amenia, Town, Dutchess	Naples, Town, Ontario	Dryden, Town, Tompkins

Table 30. Recommendations for City and Town Initiatives

6.2.4 VILLAGES

Finally, the last significant variables are included in this case for the analysis of the villas, as shown in Table 31.

Variables
Revenues
Population
Asian_%
Employment_%

Variables
Black_%

Table 31. Key Variables for Village Initiative

Once the list of variables has been included, the elbow method is then performed. As in the previous cases, the Figure 41 indicates that the optimal number of clusters is four, as evidenced by the significant change in the slope of the curve at that point.

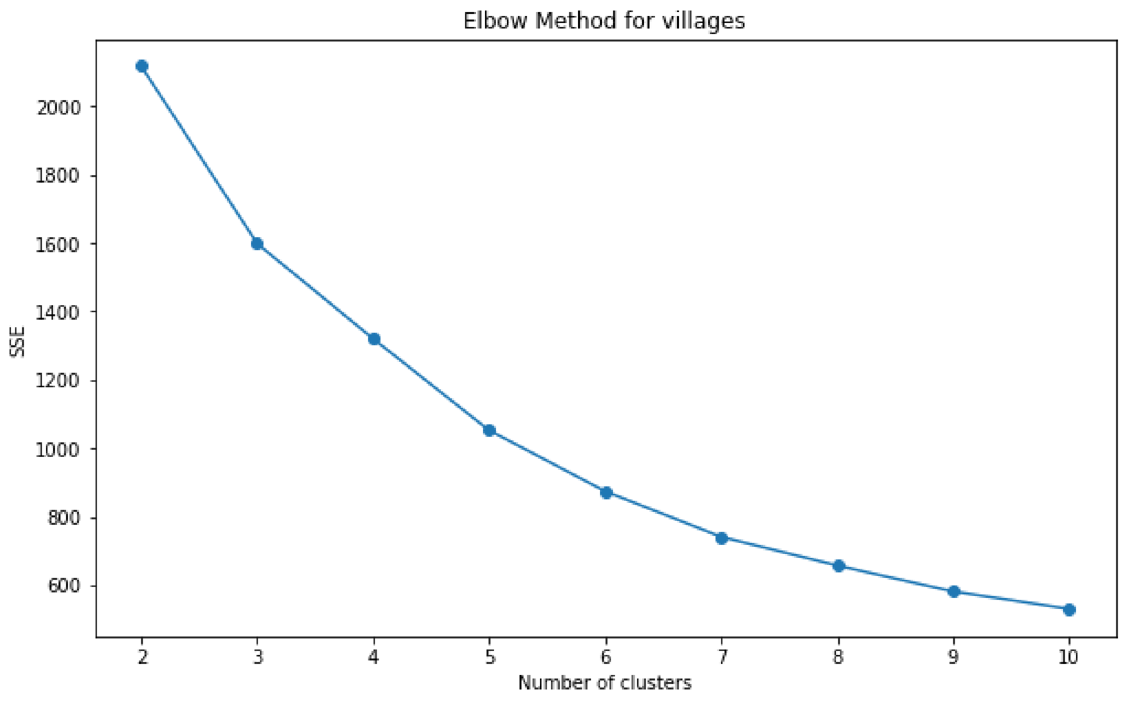


Figure 41. Elbow method for Villages

The maximum number of recommendations between villas is 21, which is a very high and positive number, as it offers a greater number of possibilities for projects. Table 32 presents a selection of illustrative proposals. For Adams Village, some recommendations align with those of the global case, yet differ in the order of recommendations, which is logical given the discrepancies in crucial variables. Furthermore, some villages may appear as recommended in the global case, but not in the specific village case.

Community	**Recommendation1**	**Recommendation 2**	**Recommendation3**
Adams, Village, Jefferson	Pulaski, Village, Oswego	Cuba, Village, Allegany	Rhinebeck, Village, Dutchess
Addison, Village, Steuben	Montour Falls, Village, Schuyler	Millerton, Village, Dutchess	Rhinebeck, Village, Dutchess
Afton, Village, Chenango	Nelsonville, Village, Putnam	Aurora, Village, Cayuga	East Nassau, Village, Rensselaer

Table 32. Recommendations for Villages Initiatives

Chapter 7. CONCLUSIONS

It can be observed that only approximately 22.33% of communities in New York State are part of the Climate Smart Communities program. This is a relatively low number when compared to the current times, where there is a high level of concern regarding climate change. It is noteworthy that the participation of municipalities, both directly and indirectly, is primarily concentrated in the southeastern region of the state, with Nassau and Suffolk counties representing a significant proportion of this concentration, as observed in the study of counties, cities, and towns.

It is a common assumption that climate change projects focus on tree planting and the protection of natural systems (environmental projects). The analysis of structural projects and environmental projects has revealed that the majority of projects are focused on infrastructure development. Suffolk County has been identified as having the greatest number of projects, both in the area of infrastructure and environmental protection.

The demographic and climatological characteristics were analyzed, and several predictive modes were studied. It was concluded that the random forest statistical model is the most appropriate for all cases, except for the counties. However, it is not reliable for counties, since the number of data to be analyzed from the counties is much lower. Moreover, the models demonstrate exceptional predictive capabilities concerning non-participating and non-project communities, respectively.

In the field of community participation, the most significant variable is the proportion of individuals with higher education, which is typically highly (and positively) correlated with income. This is logical, as individuals with higher levels of education are more aware of climate change and possess greater knowledge to undertake projects. Similarly, those with higher incomes are more likely to be able to undertake projects. Climate variables have proven to be of limited significance in determining community participation.

In examining the involvement of communities, it has not been possible to identify a relevant variable that appears in all cases. However, if one were to highlight one variable, it would be the prevalence of poverty. The relevance of the variables differs with respect to the type of municipality under study.

The majority of projects are financed to a significant extent by contributions from federal, state, and local sources. This is a logical conclusion, as the greater the financial support, the greater the amount of money that can be invested in the projects. Moreover, it has been demonstrated that the majority of projects are initiated due to the primary risk of flooding. Consequently, a greater number of projects in this area would require further analysis and implementation. Nevertheless, it is essential to acknowledge that, despite the model's high degree of accuracy, caution should be exercised when interpreting the relative importance of the factors analyzed due to the considerable variability in project costs.

The proposal of communities is highly pertinent, given that in the global case, a maximum of 39 recommendations have been identified. While it would be more appropriate to make recommendations at the municipal level, this is not always feasible. As demonstrated by the case of counties, where some communities have no suggestions, this can lead to complications. This is why the global analysis is also useful, not only to provide recommendations to municipalities that have been unable to obtain them at the local level but also to increase the number of proposed improvements in the case of cities and towns. In certain instances, the recommendations vary, resulting in an increased number of projects.

However, a challenge arises when attempting to provide recommendations for communities with similar characteristics (demographic and flood risk) but differing geographical conditions. For instance, some projects focus on rivers, which is not a relevant consideration for communities lacking such a feature.

As a prospective line of inquiry, it would be advantageous to incorporate more comprehensive geographic data, such as the presence of rivers and reservoirs, soil types, and topographic features. These enhancements would facilitate a substantial expansion and enhancement of this proposed initiative. Furthermore, the methodology could be employed in any area to be studied, even outside the United States.

Chapter 8. BIBLIOGRAPHY

[1] "Causes and Effects of Climate Change | United Nations." Accessed: Mar. 18, 2024. [Online]. Available: https://www.un.org/en/climatechange/science/causes-effects-climate-change
[2] Univision, "EEUU se calentó un 68% más rápido que el resto del mundo en los últimos 50 años," Univision. Accessed: Mar. 18, 2024. [Online]. Available: https://www.univision.com/noticias/medio-ambiente/informe-federal-cambio-climatico-evaluacion-nacional-del-clima
[3] C. N. Y. Team, "Report places New York City in top 10 most polluted US cities - CBS New York." [Online]. Available: https://www.cbsnews.com/newyork/news/new-york-city-most-polluted-united-states-cities/
[4] "GreenNY: State Purchasing and Operations," Office of General Services. Accessed: Mar. 18, 2024. [Online]. Available: https://ogs.ny.gov/greenny
[5] "Climate Smart Communities." Accessed: Feb. 09, 2022. [Online]. Available: https://climatesmart.ny.gov/
[6] O. US EPA, "EPA's Budget and Spending." Accessed: Apr. 08, 2024. [Online]. Available: https://www.epa.gov/planandbudget/budget
[7] R. EFEverde, "EEUU destina más de 6 mil millones a resiliencia de comunidades por la crisis climática," EFEverde. Accessed: Apr. 08, 2024. [Online]. Available: https://efeverde.com/eeuu-destina-mas-de-6-mil-millones-a-resiliencia-de-comunidades-por-la-crisis-climatica/
[8] "What's in the Inflation Reduction Act (IRA) of 2022 | McKinsey." Accessed: Apr. 08, 2024. [Online]. Available: https://www.mckinsey.com/industries/public-sector/our-insights/the-inflation-reduction-act-heres-whats-in-it
[9] "US metals demand set to soar on Biden's big bets on clean economy - The Oregon Group - Investment Insights." Accessed: Apr. 21, 2024. [Online]. Available: https://theoregongroup.com/energy-transition/climate-legislation/us-metals-demand-set-to-soar-on-bidens-big-bets-on-clean-economy/
[10] "THE 17 GOALS | Sustainable Development." Accessed: Apr. 22, 2024. [Online]. Available: https://sdgs.un.org/goals
[11] "Geographies | Census Survey Explorer." Accessed: Jun. 26, 2024. [Online]. Available: https://www.census.gov/data/data-tools/survey-explorer/geo.html
[12] "Nueva York Mapa gratuito, mapa mudo gratuito, mapa en blanco gratuito, plantilla de mapa contornos, condados, nombres, color." Accessed: May 04, 2024. [Online]. Available: https://d-maps.com/carte.php?num_car=21298&lang=es
[13] "Census Bureau Data." Accessed: Mar. 19, 2024. [Online]. Available: https://data.census.gov/
[14] "Financial Data for Local Governments." Accessed: Apr. 20, 2024. [Online]. Available: https://wwe1.osc.state.ny.us/localgov/findata/financial-data-for-local-governments.cfm
[15] "Civil Boundaries," gis. [Online]. Available: https://gis.ny.gov/civil-boundaries

[16] "NOAA Shoreline Website." Accessed: Apr. 05, 2024. [Online]. Available: https://shoreline.noaa.gov/data/datasheets/medres.html
[17] "First Street," First Street. Accessed: Apr. 05, 2024. [Online]. Available: https://firststreet.org/
[18] "Climate at a Glance | County Mapping | National Centers for Environmental Information (NCEI)." Accessed: Apr. 05, 2024. [Online]. Available: https://www.ncei.noaa.gov/access/monitoring/climate-at-a-glance/county/mapping
[19] "New York State Locality Hierarchy with Websites | State of New York." Accessed: Feb. 13, 2024. [Online]. Available: https://data.ny.gov/Government-Finance/New-York-State-Locality-Hierarchy-with-Websites/55k6-h6qq/data_preview
[20] "Binghamton - Johnson City Joint Sewage Treatment Plant Rehabilitation Project | City of Binghamton New York." Accessed: May 04, 2024. [Online]. Available: https://www.binghamton-ny.gov/government/departments/joint-sewage-project
[21] "Trees For Tribs - NYSDEC." Accessed: May 04, 2024. [Online]. Available: https://dec.ny.gov/nature/forests-trees/saratoga-tree-nursery/trees-for-tribs
[22] "Trees for America at arborday.org." Accessed: May 04, 2024. [Online]. Available: https://www.arborday.org/programs/trees4america.cfm
[23] "Arsenal Park Hydraulic Study." Accessed: May 04, 2024. [Online]. Available: https://www.syr.gov/Departments/Engineering/Initiatives/Arsenal-Park
[24] "Tree Planting Nonprofit | Arbor Day Foundation." Accessed: May 04, 2024. [Online]. Available: https://www.arborday.org/
[25] G. Westreicher, "Modelo Probit," Economipedia. Accessed: May 21, 2024. [Online]. Available: https://economipedia.com/definiciones/modelo-probit.html
[26] por E. HD, "Modelo Probit," Economía HD. Accessed: May 21, 2024. [Online]. Available: https://economiahd.com/2023/08/03/modelo-probit/
[27] J. Abbadia, "Cómo determinar la significación estadística: Guía práctica," Blog Mind the Graph. Accessed: May 21, 2024. [Online]. Available: https://mindthegraph.com/blog/es/how-to-determine-statistical-significance/
[28] Daniel, "Random Forest: Bosque aleatorio. Definición y funcionamiento," Formación en ciencia de datos | DataScientest.com. Accessed: May 21, 2024. [Online]. Available: https://datascientest.com/es/random-forest-bosque-aleatorio-definicion-y-funcionamiento
[29] M. G. Santos, "Técnicas de procesamiento del lenguaje natural para analizar artículos sobre Machine Learning y la gestión de redes de agua".
[30] "¿Qué es un bosque aleatorio? | IBM." Accessed: May 28, 2024. [Online]. Available: https://www.ibm.com/mx-es/topics/random-forest
[31] "Regresión lineal de los mínimos cuadrados (OLS)," XLSTAT, Your data analysis solution. Accessed: May 30, 2024. [Online]. Available: https://www.xlstat.com/es/soluciones/funciones/regresion-lineal-de-los-minimos-cuadrados-ols
[32] "RPubs - Document." Accessed: May 30, 2024. [Online]. Available: https://rpubs.com/dsulmont/740953
[33] "Durbin Watson Test: What It Is in Statistics, With Examples," Investopedia. Accessed: May 30, 2024. [Online]. Available: https://www.investopedia.com/terms/d/durbin-watson-statistic.asp

[34] "Aprendizaje automático: Una introducción al error cuadrático medio y las líneas de regresión.," freeCodeCamp.org. Accessed: May 31, 2024. [Online]. Available: https://www.freecodecamp.org/espanol/news/aprendizaje-automatico-una-introduccion-al-error-cuadratico-medio-y-las-lineas-de-regresion/

[35] "IBM Cognos Analytics 11.1.x." Accessed: May 31, 2024. [Online]. Available: https://www.ibm.com/docs/es/cognos-analytics/11.1.0?topic=terms-r2

[36] "Error medio absoluto cuantificacion de la precision del pronostico con el error medio absoluto una guia completa," FasterCapital. Accessed: May 31, 2024. [Online]. Available: https://fastercapital.com/es/contenido/Error-medio-absoluto--cuantificacion-de-la-precision-del-pronostico-con-el-error-medio-absoluto--una-guia-completa.html

[37] "kmeans." Accessed: Jun. 17, 2024. [Online]. Available: https://www.unioviedo.es/compnum/laboratorios_py/kmeans/kmeans.html

[38] L. Ramírez, "Algoritmo k-means: ¿Qué es y cómo funciona?," *Thinking for Innovation*, Jan. 2023, Accessed: Jun. 17, 2024. [Online]. Available: https://www.iebschool.com/blog/algoritmo-k-means-que-es-y-como-funciona-big-data/

ANEXO I

The main program is included below, along with the functions used in this project.

MAIN PROGRAM

```
files=[]
for i in range(24):
 files.append(str(i+1)+".csv")
from demographic import demographic
df_demographic=demographic(files)
from climate_smart_communities import participation
df_demographic=participation(df_demographic)
from survey import survey
(df_demographic, df_survey_all, df_survey_yes, df_survey_no) =
survey(df_demographic)
from mapNY import geo_df
(gdf_final,gdf_cities_towns,gdf_counties,gdf_villages,gdf_FIPS)=geo_df()
from flood_risk import flood_risk
df_demographic=flood_risk(df_demographic,gdf_FIPS)
from revenues import revenues
df_demographic,df_revenues=revenues(df_demographic)
df_demographic = df_demographic.drop('FIPS_CODE', axis=1)
df_demographic = df_demographic.drop(df_demographic[(df_demographic['County'] ==
'bronx') |  (df_demographic['County'] == 'kings') |  (df_demographic['County'] ==
'new york') | (df_demographic['County'] == 'queens') | (df_demographic['County']
== 'richmond')].index)
from structural_or_environmental_projects import env_struc
(df_cost_g,df_structural_cost_max,df_environmental_cost_max,df_structural_cost_me
d,df_environmental_cost_med,df_structural_cost_min,df_environmental_cost_min,df_s
tructural_cost,df_environmental_cost,correlation_total)=env_struc(df_demographic,
df_survey_yes)
from visualization import visualization
visualization(df_demographic,df_survey_all,df_survey_yes,gdf_final,gdf_counties,d
f_structural_cost_max,df_environmental_cost_max,df_structural_cost_med,df_environ
mental_cost_med,df_structural_cost_min,df_environmental_cost_min)
corr_participation=0.75
corr_yes=0.6
corr=0.6
from participation import
total_participation,class_participation,total_yes,class_yes
participation_probit_results_df,participation_probit_accuracy,participation_gini_
variables_df,participation_perm_variables_df,participation_random_forest_accuracy
,participation_df_no_study=total_participation(df_demographic,
corr_participation)
```

```
yes_probit_results_df,yes_probit_accuracy,yes_gini_variables_df,yes_perm_variable
s_df,yes_random_forest_accuracy,yes_df_no_study=total_yes(df_demographic,
corr_yes)
class_='county'
participation_county_probit_results_df, participation_county_probit_accuracy,
participation_county_gini_variables_df, participation_county_perm_variables_df,
participation_county_random_forest_accuracy, participation_county_df_no_study
=class_participation(df_demographic, corr_participation,class_)
yes_county_probit_results_df, yes_county_probit_accuracy,
yes_county_gini_variables_df, yes_county_perm_variables_df,
yes_county_random_forest_accuracy, yes_county_df_no_study
=class_yes(df_demographic, corr_yes,class_)
class_='city'
participation_city_town_probit_results_df,
participation_city_town_probit_accuracy,
participation_city_town_gini_variables_df,
participation_city_town_perm_variables_df,
participation_city_town_random_forest_accuracy,
participation_city_town_df_no_study=class_participation(df_demographic,
corr_participation,class_)
yes_city_town_probit_results_df, yes_city_town_probit_accuracy,
yes_city_town_gini_variables_df, yes_city_town_perm_variables_df,
yes_city_town_random_forest_accuracy,
yes_city_town_df_no_study=class_yes(df_demographic, corr_yes,class_)
class_='village'
participation_village_probit_results_df, participation_village_probit_accuracy,
participation_village_gini_variables_df, participation_village_perm_variables_df,
participation_village_random_forest_accuracy,
participation_village_df_no_study=class_participation(df_demographic,
corr_participation, class_)
yes_village_probit_results_df, yes_village_probit_accuracy,
yes_village_gini_variables_df, yes_village_perm_variables_df,
yes_village_random_forest_accuracy,
yes_village_df_no_study=class_yes(df_demographic, corr_yes, class_)
from economic_model import OLS
col,importance=OLS(df_demographic,df_survey_yes,corr)
from proposal_redaccion import prepare_clustering,evaluate_clustering
class_='all'
global_variables=['Poverty_%','Employment_%','Extreme_Flooding','Black_%','Rent_%
','Answer_yes_survey']
scaled_df_demographic_proposal_global,df_demographic_proposal_global,df_answer_ye
s_proposal_global,semilla_global=prepare_clustering(df_demographic,global_variabl
es,class_)
optimal_clusters_global=4
df_proposal_global=evaluate_clustering(df_demographic,scaled_df_demographic_propo
sal_global,df_demographic_proposal_global,df_answer_yes_proposal_global,optimal_c
lusters_global,semilla_global)
class_='county'
variables_county=['Poverty_%','Asian_%','Hispanic_%','Male_%','Significant_Floodi
ng','Answer_yes_survey']
scaled_df_demographic_proposal_county,df_demographic_proposal_county,df_answer_ye
s_proposal_county,semilla_county=prepare_clustering(df_demographic,variables_coun
ty,class_)
```

```
optimal_clusters_county=4
df_proposal_county=evaluate_clustering(df_demographic,scaled_df_demographic_propo
sal_county, df_demographic_proposal_county, df_answer_yes_proposal_county,
optimal_clusters_county, semilla_county)
class_='city'
variables_city_town=['Poverty_%','Employment_%','Black_%','Significant_Flooding',
'Answer_yes_survey']
scaled_df_demographic_proposal_city_town,df_demographic_proposal_city_town,df_ans
wer_yes_proposal_city_town,semilla_city_town=prepare_clustering(df_demographic,va
riables_city_town,class_)
optimal_clusters_city_town=4
df_proposal_city_town=evaluate_clustering(df_demographic,scaled_df_demographic_pr
oposal_city_town, df_demographic_proposal_city_town,
df_answer_yes_proposal_city_town, optimal_clusters_city_town, semilla_city_town)
class_='village'
variables_village=['Population','Asian_%','Revenues','Employment_%','Black_%','An
swer_yes_survey']
scaled_df_demographic_proposal_village,df_demographic_proposal_village,df_answer_
yes_proposal_village,semilla_village=prepare_clustering(df_demographic,variables_
village,class_)
optimal_clusters_village=4
df_proposal_village=evaluate_clustering(df_demographic,scaled_df_demographic_prop
osal_village, df_demographic_proposal_village, df_answer_yes_proposal_village,
optimal_clusters_village, semilla_village)
```

FUNCTIONS

DEMOGRAPHIC

```
import pandas as pd
def demographic(archivos):
    df_city_town= pd.DataFrame()
    df1=pd.DataFrame()
    for i in archivos[:8]:
        df_aux=df_city_town.copy()
        df1=pd.read_csv(i)
        df_city_town=pd.concat([df_aux,df1], axis=1)
        df_city_town=df_city_town.loc[:,~df_city_town.columns.duplicated()]
    df_city_town['NAME'] = df_city_town['NAME'].astype(str)
    df_city_town[['NAME', 'County']] = df_city_town['NAME'].str.split(',', n=1,
expand=True)
    df_city_town['County'] = df_city_town['County'].astype(str)
    df_city_town['County'] =
df_city_town['County'].str.split(',').str[0].str.strip()

df_city_town['County']=df_city_town['County'].str.rsplit(n=1).str[0].str.strip()
    last_word = df_city_town['NAME'].str.split().str[-1]
    df_city_town = pd.concat([df_city_town[['NAME']], last_word.rename('Class'),
df_city_town.drop('NAME', axis=1)], axis=1)
```

```
    df_city_town['NAME'] = df_city_town['NAME'].apply(lambda x: '
'.join(x.split()[:-1]))

columns=['NAME','Class','County','GEO_ID','DP05_0001E','DP05_0002PE','DP05_0038PE
','DP05_0044PE','DP05_0073PE','DP05_0037PE',

'DP03_0004PE','S1501_C02_015E','S1901_C01_012E','S1701_C03_001E','S1601_C02_002E'
,'S2501_C05_001E','S2501_C01_001E','S1810_C03_001E']
    df_city_town=df_city_town[columns]
    df_city_town = df_city_town.applymap(lambda x: x.lower() if isinstance(x,
str) else x)
    df_city_town['NAME'] = df_city_town['NAME'].str.replace('.', '', regex=False)
    df_city_town['County'] = df_city_town['County'].str.replace('.', '',
regex=False)
    NAME=['mount vernon','mcdonough','au sable','north east','de
kalb','fallsburg','deruyter','de witt','lagrange','st armand']
    County=['westchester','chenango','clinton','dutchess','st
lawrence','sullivan','madison','onondaga','dutchess','essex']
    Class=[
'city','town','town','town','town','town','town','town','town','town']
    Data=['mt vernon','mc
donough','ausable','northeast','dekalb','fallsburgh','de ruyter','dewitt','la
grange','st. armand']
    for m,c,cl,d in zip(NAME,County,Class,Data):
        fila = df_city_town[(df_city_town['NAME'] == m) & (df_city_town['County']
== c)& (df_city_town['Class'] == cl)]
        df_city_town.loc[fila.index, 'NAME'] = d
    Muni_Name=[     'clare','scarsdale','inlet']
    County=['st lawrence','westchester','hamilton']
    Class=['town','town','town']
    Data=[41250,250000,92857]
    ipc_2022_2021=0.065/0.07
    Data=[x*ipc_2022_2021 for x in Data]
    for m,c,cl,d in zip(Muni_Name,County,Class,Data):
        fila = df_city_town[(df_city_town['NAME'] == m) & (df_city_town['County']
== c)& (df_city_town['Class'] == cl)]
        df_city_town.loc[fila.index, 'S1901_C01_012E'] = d
    df_city_town= df_city_town[df_city_town['Class'].isin(['city','town'])]
    df_city_town = df_city_town[~((df_city_town['NAME'] == 'geneva') &
(df_city_town['Class'] == 'city') & (df_city_town['County'] == 'seneca'))]
    df_city_town = df_city_town[~((df_city_town['NAME'] == 'palm tree') &
(df_city_town['Class'] == 'town') & (df_city_town['County'] == 'orange'))]
    df_county= pd.DataFrame()
    df2=pd.DataFrame()
    for i in archivos[8:16]:
        df_aux=df_county.copy()
        df2=pd.read_csv(i)
        df_county=pd.concat([df_aux,df2], axis=1)
        df_county=df_county.loc[:,~df_county.columns.duplicated()]
    df_county['NAME']=df_county['NAME'].str.split(',',n=1).str[0]
    df_county['NAME'] = df_county['NAME'].astype(str)
    last_word = df_county['NAME'].str.split().str[-1]
    df_county = pd.concat([df_county[['NAME']], last_word.rename('Class'),
df_county.drop('NAME', axis=1)], axis=1)
    df_county['NAME'] = df_county['NAME'].apply(lambda x: ' '.join(x.split()[:-
1]))
    df_county['County']=df_county['NAME']

columns=['NAME','Class','County','GEO_ID','DP05_0001E','DP05_0002PE','DP05_0038PE
','DP05_0044PE','DP05_0073PE','DP05_0037PE',

'DP03_0004PE','S1501_C02_015E','S1901_C01_012E','S1701_C03_001E','S1601_C02_002E'
,'S2501_C05_001E','S2501_C01_001E','S1810_C03_001E']
    df_county=df_county[columns]
    df_county = df_county.applymap(lambda x: x.lower() if isinstance(x, str) else
x)
    df_county['NAME'] = df_county['NAME'].str.replace('.', '', regex=False)
    df_county['County'] = df_county['County'].str.replace('.', '', regex=False)
    df_county= df_county[df_county['Class'].isin(['county'])]
    df_county_city_town=pd.merge(df_city_town,df_county,how='outer')
    df_county_city_town=df_county_city_town.rename(columns={'NAME':'Muni Name'})
    df_county_city_town = df_county_city_town.applymap(lambda x: x.lower() if
isinstance(x, str) else x)
    df_county_city_town['Split_Texto'] =
df_county_city_town['GEO_ID'].str.split('us')
    df_county_city_town['GEO_ID'] = df_county_city_town['Split_Texto'].str.get(1)
    df_county_city_town= df_county_city_town.drop('Split_Texto', axis=1)
    df_county_city_town=
df_county_city_town[df_county_city_town['Class'].isin(['town',
'city','county','village'])]

columns=['DP05_0001E','DP05_0002PE','DP05_0038PE','DP05_0044PE','DP05_0073PE','DP
05_0037PE',

'DP03_0004PE','S1501_C02_015E','S1901_C01_012E','S1701_C03_001E','S1601_C02_002E'
,'S2501_C05_001E','S2501_C01_001E','S1810_C03_001E']
    for columna in columns:
        df_county_city_town[columna] =
pd.to_numeric(df_county_city_town[columna], errors='coerce')
    df_county_city_town['DP05_0037PE'] = 100 - df_county_city_town['DP05_0037PE']
    df_county_city_town['S2501_C05_001PE'] =
(df_county_city_town['S2501_C05_001E']/df_county_city_town['S2501_C01_001E'])*100
    df_county = df_county_city_town[df_county_city_town['Class'] == 'county']
    df_village= pd.DataFrame()
    df3=pd.DataFrame()
    for i in archivos[16:24]:
        df_aux=df_village.copy()
        df3=pd.read_csv(i,low_memory=False)
        df_village=pd.concat([df_aux,df3], axis=1)
        df_village=df_village.loc[:,~df_village.columns.duplicated()]
    df_village['NAME']=df_village['NAME'].str.split(',',n=1).str[0]
    df_village['NAME'] = df_village['NAME'].astype(str)
    last_word = df_village['NAME'].str.split().str[-1]
    df_village = pd.concat([df_village[['NAME']], last_word.rename('Class'),
df_village.drop('NAME', axis=1)], axis=1)
```

```
    df_village['NAME'] = df_village['NAME'].apply(lambda x: ' '.join(x.split()[:-
1]))

columns=['NAME','Class','GEO_ID','DP05_0001E','DP05_0002PE','DP05_0038PE','DP05_0
044PE','DP05_0073PE','DP05_0037PE',

'DP03_0004PE','S1501_C02_015E','S1901_C01_012E','S1701_C03_001E','S1601_C02_002E'
,'S2501_C05_001E','S2501_C01_001E','S1810_C03_001E']
    df_village=df_village[columns]
    df_village=df_village.rename(columns={'NAME':'Muni Name'})
    df_village = df_village.applymap(lambda x: x.lower() if isinstance(x, str)
else x)
    df_village['Split_Texto'] = df_village['GEO_ID'].str.split('us')
    df_village['GEO_ID'] = df_village['Split_Texto'].str.get(1)
    df_village= df_village.drop('Split_Texto', axis=1)
    df_village= df_village[df_village['Class'].isin(['village'])]
    df_municode = pd.read_excel('Municode.xlsx')
    df_municode=df_municode.rename(columns={'Geo ID': 'GEO_ID'})
    df_municode = df_municode.applymap(lambda x: x.lower() if isinstance(x, str)
else x)
    columns=['Muni Code','Muni Name','Class','County']
    df_municode=df_municode[columns]
    df_municode= df_municode[df_municode['Class'].isin(['village'])]
    df_municode=df_municode.drop_duplicates(subset=['Muni Name'])
    df_municode= df_municode.dropna()
    df_village= pd.merge(df_village, df_municode, on=['Muni Name', 'Class'],
how='outer')
    df_village = df_village.drop_duplicates(subset=['Muni Name'], keep='first')
    df_village= df_village.dropna()
    columns=['Muni
Name','Class','County','GEO_ID','DP05_0001E','DP05_0002PE','DP05_0038PE','DP05_00
44PE','DP05_0073PE','DP05_0037PE',

'DP03_0004PE','S1501_C02_015E','S1901_C01_012E','S1701_C03_001E','S1601_C02_002E'
,'S2501_C05_001E','S2501_C01_001E','S1810_C03_001E']
    df_village=df_village[columns]

columns=['DP05_0001E','DP05_0002PE','DP05_0038PE','DP05_0044PE','DP05_0073PE','DP
05_0037PE',

'DP03_0004PE','S1501_C02_015E','S1901_C01_012E','S1701_C03_001E','S1601_C02_002E'
,'S2501_C05_001E','S2501_C01_001E','S1810_C03_001E']
    for columna in columns:
        df_village[columna] = pd.to_numeric(df_village[columna], errors='coerce')
    df_village['DP05_0037PE'] = 100 - df_village['DP05_0037PE']
    df_village['S2501_C05_001PE'] =(
df_village['S2501_C05_001E']/df_village['S2501_C01_001E'])*100
    Muni_Name=["brookville", "east hills", "flower hill", "head of the harbor",
"hewlett bay park", "hewlett harbor", "hewlett neck", "huntington bay", "laurel
hollow", "mill neck", "munsey park", "ocean beach", "old field", "old westbury",
"oyster bay cove", "pelham manor", "plandome", "plandome heights", "quogue",
"roslyn estates", "saltaire", "sands point", "scarsdale", "tuxedo park", "west
hampton dunes", "woodsburgh"]
```

```
    County = ["nassau", "nassau", "nassau", "suffolk", "nassau", "nassau",
"nassau", "suffolk", "nassau", "nassau", "nassau", "suffolk", "suffolk",
"nassau", "nassau", "westchester", "nassau", "nassau", "suffolk", "nassau",
"suffolk", "nassau", "westchester", "orange", "suffolk", "nassau"]
    Data=[250000,250000,250000,250000,250000,250000,250000,
          250000,250000,250000,250000,72273,250000,250000,250000,
          250000,250000,250000,13403,250000,
          46528,250000,250000,250000,250000,250000]
    ipc_2022_2021=0.065/0.07
    Data=[x*ipc_2022_2021 for x in Data]
    for m,c,d in zip(Muni_Name,County,Data):
        fila = df_village[(df_village['Muni Name'] == m) & (df_village['County']
== c)]
        df_village.loc[fila.index, 'S1901_C01_012E'] = d
    Muni_Name=['saranac lake','attica']
    County=['franklin','wyoming']
    Class=[ 'village','village']
    Data=['essex','genesee','genesse']
    for m,c,cl,d in zip(Muni_Name,County,Class,Data):
        fila = df_village[(df_village['Muni Name'] == m) & (df_village['County']
== c)& (df_village['Class'] == cl)]
        df_village.loc[fila.index, 'County'] = d
    # Actualizar directamente el valor en la posición encontrada
    Muni_Name=['mcgraw','deruyter','manorhaven','bloomingburg','leroy','st.
johnsville']
    County=['cortland','madison','nassau','sullivan','genesee','montgomery']
    Data=['mc graw','de ruyter','manor haven','bloomingburgh','le roy','st
johnsville']
    for m,c,d in zip(Muni_Name,County,Data):
        fila = df_village[(df_village['Muni Name'] == m) & (df_village['County']
== c)]
        df_village.loc[fila.index, 'Muni Name'] = d
    fila = (df_village['Muni Name'] == 'dolgeville').idxmax()
    df_village.at[fila, 'County'] = 'fulton'
    df_village['County'] = df_village['County'].str.replace('.', '', regex=False)

df_geo=pd.read_csv('New_York_State_Locality_Hierarchy_with_Websites_20240327.csv'
)
    df_geo = df_geo.applymap(lambda x: x.lower() if isinstance(x, str) else x)
    df_geo.drop(df_geo[df_geo['Type Code'] != 4].index, inplace=True)
    df_village['Town'] = df_village.apply(lambda row: df_geo[(df_geo['Village
Name'] == row['Muni Name']) & (df_geo['County Name'] == row['County'])]['Town
Name'].values[0] if not df_geo[(df_geo['Village Name'] == row['Muni Name']) &
(df_geo['County Name'] == row['County'])].empty else None, axis=1)
    df_village['2nd County'] = df_village.apply(lambda row:
df_geo[(df_geo['Village Name'] == row['Muni Name'])]['2nd County'].values[0] if
not df_geo[(df_geo['Village Name'] == row['Muni Name'])].empty else None, axis=1)
    df_demographic=pd.merge(df_county_city_town,df_village,how='outer')
    columns=['Muni Name','Class','County', '2nd County',
'Town','GEO_ID','DP05_0001E','DP05_0002PE','DP05_0038PE','DP05_0044PE','DP05_0073
PE','DP05_0037PE',
'DP03_0004PE','S1501_C02_015E','S1901_C01_012E','S1701_C03_001E','S1601_C02_002E'
,'S2501_C05_001PE','S1810_C03_001E']
    df_demographic = df_demographic[columns]
    df_demographic =
df_demographic.drop(df_demographic[df_demographic['DP05_0001E'] == 0].index)
    df_demographic = df_demographic.drop(df_demographic[(df_demographic['County']
== 'bronx') |  (df_demographic['County'] == 'kings') |  (df_demographic['County']
== 'new york') | (df_demographic['County'] == 'queens') |
(df_demographic['County'] == 'richmond')].index)
    new_columns=['Muni Name','Class','County', '2nd County',
'Town','GEO_ID','Population','Male_%','Black_%','Asian_%','Hispanic_%','Nonwhite_
%',

'Employment_%','Degree_%','Median_income','Poverty_%','English_%','Rent_%','Disab
ility_%']
    df_demographic.columns=new_columns
    return(df_demographic)
```

CLIMATE SMART COMMUNITIES

```
import pandas as pd
def participation(df_demographic):
    df_smart_communities = pd.read_excel('2023-0702 FOIL List of Climate Smart
Communities at time of Survey.xlsx')
    df_smart_communities= df_smart_communities.applymap(lambda x: x.lower() if
isinstance(x, str) else x)
    df_smart_communities.rename(columns={'Climate Smart Communities as of
2/9/2022': 'Muni Name'}, inplace=True)
    df_smart_communities[['Class', 'Muni Name']] = df_smart_communities['Muni
Name'].str.split(' of ', expand=True)
    df_smart_communities=df_smart_communities[['Muni Name', 'Class']]
    df_smart_communities = df_smart_communities.applymap(lambda x: x.lower() if
isinstance(x, str) else x)
    df_smart_communities=
df_smart_communities[df_smart_communities['Class'].isin(['town',
'city','county','village'])]
    NAME=['north east','fallsburg']
    Data=['northeast','fallsburgh']
    for m,d in zip(NAME,Data):
        fila = df_smart_communities[(df_smart_communities['Muni Name'] == m) ]
        df_smart_communities.loc[fila.index, 'Muni Name'] = d
    df_demographic['Participation'] = df_demographic[['Muni Name',
'Class']].apply(lambda x: 1 if (x['Muni Name'], x['Class']) in
zip(df_smart_communities['Muni Name'], df_smart_communities['Class']) else 0,
axis=1)
    valor1 = 'clinton'
    valor2 = 'clinton'
    valor3='town'
    indices_to_drop = df_demographic[(df_demographic['Muni Name'] == valor1) &
(df_demographic['County'] == valor2)&(df_demographic['Class'] == valor3)].index
```

```
    df_demographic.loc[indices_to_drop, 'Participation'] = 0
    valor1 = 'lewis'
    valor2 = 'lewis'
    valor3='town'
    indices_to_drop = df_demographic[(df_demographic['Muni Name'] == valor1) &
(df_demographic['County'] == valor2)&(df_demographic['Class'] == valor3)].index
    df_demographic.loc[indices_to_drop, 'Participation'] = 0
    valor1 = 'brighton'
    valor2 = 'franklin'
    valor3='town'
    indices_to_drop = df_demographic[(df_demographic['Muni Name'] == valor1) &
(df_demographic['County'] == valor2)&(df_demographic['Class'] == valor3)].index
    df_demographic.loc[indices_to_drop, 'Participation'] = 0
    valor1 = 'chester'
    valor2 = 'orange'
    valor3='town'
    indices_to_drop = df_demographic[(df_demographic['Muni Name'] == valor1) &
(df_demographic['County'] == valor2)&(df_demographic['Class'] == valor3)].index
    df_demographic.loc[indices_to_drop, 'Participation'] = 0
    df_demographic.reset_index(drop=True, inplace=True)
    df_demographic=df_demographic.reset_index(drop=True)
    return (df_demographic)
```

SURVEY

```
import pandas as pd
def survey(df_demographic):
    df_survey_all = pd.read_excel('2023-0702 FOIL Data from survey used in
Climate Change report.xlsx')
    df_survey_all= df_survey_all.applymap(lambda x: x.lower() if isinstance(x,
str) else x)
    df_survey_all = df_survey_all[df_survey_all['Muni code'].notna()]
    df_survey_all = df_survey_all.dropna(axis=1, how='all')
    df_survey_all = df_survey_all.drop('2020 Population', axis=1)
    df_survey_aux= pd.get_dummies(df_survey_all['PrimeCCHaz'],dtype=int)
    df_survey_all = pd.concat([df_survey_all, df_survey_aux], axis=1)
    df_survey_all = df_survey_all.drop(columns='PrimeCCHaz')
    columns= ['drought', 'erosion', 'extreme heat', 'extreme weather', 'flooding
(not related to sea-level rise)','sea-level rise']
    new_columns = [i + '_1' for i in columns]
    df_survey_all.rename(columns=dict(zip(columns, new_columns)), inplace=True)
    columns_to_process = ['OtherCCHaz1', 'OtherCCHaz2', 'OtherCCHaz3',
'OtherCCHaz4', 'OtherCCHaz5', 'OtherCCHaz6']
    df_survey_aux = pd.get_dummies(df_survey_all[columns_to_process], dtype=int)
    df_survey_all = pd.concat([df_survey_all, df_survey_aux], axis=1)
    df_survey_all = df_survey_all.drop(columns=columns_to_process)
    columns=['OtherCCHaz1_flooding (not related to sea-level
rise)','OtherCCHaz2_sea-level rise', 'OtherCCHaz3_extreme
heat','OtherCCHaz4_extreme weather', 'OtherCCHaz5_erosion','OtherCCHaz6_drought']
    columns_=['flooding (not related to sea-level rise)','sea-level
rise','extreme heat','extreme weather','erosion','drought']
```

```
    new_columns = [i + '_2' for i in columns_]
    df_survey_all.rename(columns=dict(zip(columns, new_columns)), inplace=True)
    df_survey_all['Muni Name'] = df_survey_all['Muni Name'].astype(str)
    df_survey_all[['Muni Name', 'County']] = df_survey_all['Muni
Name'].str.rsplit(n=1, expand=True)
    df_survey_all['County'] = df_survey_all['County'].str.strip()
    df_survey_all['County'] = df_survey_all['County'].apply(lambda x:
x.replace('(', '').replace(')', ''))
    df_survey_all['Muni Name'] = df_survey_all['Muni Name'].str.replace(r'.*?of
', '', regex=True)
    NAME=['north east','fallsburg']
    Data=['northeast','fallsburgh']
    for m,d in zip(NAME,Data):
        fila = df_survey_all[(df_survey_all['Muni Name'] == m) ]
        df_survey_all.loc[fila.index, 'Muni Name'] = d
    NAME=['saranac lake']
    Data=['essex']
    for m,d in zip(NAME,Data):
        fila = df_survey_all[(df_survey_all['Muni Name'] == m) ]
        df_survey_all.loc[fila.index, 'County'] = d
    df_survey=df_survey_all.copy()
    df_survey_no = df_survey_all.copy()
    df_survey_no =
df_survey[((df_survey['StartDate'].astype(str).str.lower().str.contains('beyond
cy 2027')) &

(df_survey['EndDate'].astype(str).str.lower().str.contains('beyond cy 2027'))) |

((df_survey['StartDate'].astype(str).str.lower().str.contains('before cy 2017'))
&

(df_survey['EndDate'].astype(str).str.lower().str.contains('before cy 2017'))) |

(df_survey['StartDate'].astype(str).str.lower().str.contains('2027')) |
                          (df_survey['YesNo to HSA'] != 'yes') |
                          (df_survey['YesNo to HSA'].isna())]
    df_survey_yes = df_survey.drop(df_survey_no.index)
    df_survey_yes['StartDate'] = pd.to_numeric(df_survey_yes['StartDate'],
errors='coerce')
    df_survey_yes['EndDate'] = pd.to_numeric(df_survey_yes['EndDate'],
errors='coerce')
    df_survey_yes['Cost']= df_survey_yes['Cost'].apply(round)
    df_survey_yes['HSA ID']=df_survey_yes['HSA ID'].astype(int)
    df_demographic['Answer_survey'] = df_demographic[['Muni Name',
'Class','County']].apply(lambda x: 1 if (x['Muni Name'], x['Class'], x['County'])
in zip(df_survey['Muni Name'], df_survey['Class'], df_survey['County']) else 0,
axis=1)
    df_demographic['Answer_yes_survey'] = df_demographic[['Muni Name',
'Class','County']].apply(lambda x: 1 if (x['Muni Name'], x['Class'], x['County'])
in zip(df_survey_yes['Muni Name'], df_survey_yes['Class'],
df_survey_yes['County']) else 0, axis=1)
    return (df_demographic,df_survey_all,df_survey_yes,df_survey_no)
```

MAPNY

```
import geopandas as gpd
import pandas as pd
def geo_df():
    gdf_cities_towns= gpd.read_file("Cities_Towns.shp")
    gdf_cities_towns=gdf_cities_towns.applymap(lambda x: x.lower() if 
isinstance(x, str) else x)
    new_columns={'NAME':'Muni Name','MUNI_TYPE':'Class','COUNTY':'County'}
    gdf_cities_towns=gdf_cities_towns.rename(columns=new_columns)
    select_columns=['Muni 
Name','Class','County','Shape_Leng','Shape_Area','geometry','FIPS_CODE']
    gdf_cities_towns=gdf_cities_towns[select_columns]
    NAME=['mount vernon','mcdonough','north east','st 
armand','leroy','deruyter','de witt','de kalb','fallsburg']

County=['westchester','chenango','dutchess','essex','genesee','madison','onondaga
','st lawrence','sullivan']
    Class=[ 'city','town','town','town','town','town','town','town','town']
    Data=['mt vernon','mc donough','northeast','st. armand','le roy','de 
ruyter','dewitt','dekalb','fallsburgh']
    for m,c,cl,d in zip(NAME,County,Class,Data):
        fila = gdf_cities_towns[(gdf_cities_towns['Muni Name'] == m) & 
(gdf_cities_towns['County'] == c)& (gdf_cities_towns['Class'] == cl)]
        gdf_cities_towns.loc[fila.index, 'Muni Name'] = d
    gdf_counties = gpd.read_file("Counties.shp")
    gdf_counties['County']=gdf_counties['NAME']
    gdf_counties['Class']='county'
    gdf_counties=gdf_counties.applymap(lambda x: x.lower() if isinstance(x, str) 
else x)
    new_columns={'NAME':'Muni Name'}
    gdf_counties=gdf_counties.rename(columns=new_columns)
    gdf_counties=gdf_counties[select_columns]
    gdf_villages = gpd.read_file("villages.shp")
    gdf_villages=gdf_villages.applymap(lambda x: x.lower() if isinstance(x, str) 
else x)
    gdf_villages['Class']='village'
    new_columns={'NAME':'Muni Name','COUNTY':'County'}
    gdf_villages=gdf_villages.rename(columns=new_columns)
    fila = (gdf_villages['Muni Name'] == 'dolgeville').idxmax()
    gdf_villages.at[fila, 'County'] = 'fulton'
    fila = (gdf_villages['Muni Name'] == 'earlville').idxmax()
    gdf_villages.at[fila, 'County'] = 'chenango'
    fila = (gdf_villages['Muni Name'] == 'adams').idxmax()
    gdf_villages.at[fila, 'County'] = 'jefferson'
    gdf_villages['Muni Name'] = gdf_villages['Muni Name'].str.replace('.', '', 
regex=False)
    gdf_villages['County'] = gdf_villages['County'].str.replace('.', '', 
regex=False)
    Muni_Name=['mcgraw','deruyter','manorhaven','bloomingburg','leroy']
    County=['cortland','madison','nassau','sullivan','genesee']
    Data=['mc graw','de ruyter','manor haven','bloomingburgh','le roy']
    for m,c,d in zip(Muni_Name,County,Data):
        fila = gdf_villages[(gdf_villages['Muni Name'] == m) & 
(gdf_villages['County'] == c)]
        gdf_villages.loc[fila.index, 'Muni Name'] = d
    gdf_FIPS = pd.concat([gdf_cities_towns,gdf_counties, gdf_villages], 
ignore_index=True)
    gdf_FIPS['County'] = gdf_FIPS['County'].str.split(',').str.get(0)
    gdf_final=gdf_FIPS.copy()
    select_columns=['Muni 
Name','Class','County','Shape_Leng','Shape_Area','geometry']
    gdf_final=gdf_final[select_columns]
    select_columns=['Muni Name','Class','County','FIPS_CODE']
    gdf_FIPS=gdf_FIPS[select_columns]
    return (gdf_final,gdf_cities_towns,gdf_counties,gdf_villages,gdf_FIPS)
```

FLOOD RISK

```
import pandas as pd
def flood_risk(df_demographic,gdf_FIPS):
    df_flood_risk_city_town=pd.read_csv("CityTownFloodRisk.csv")
    
df_flood_risk_city_town=df_flood_risk_city_town.rename(columns={'FIPSMuni':'FIPS_
CODE'})
    df_flood_risk_county=pd.read_csv("CountyFloodRisk.csv")
    
df_flood_risk_county=df_flood_risk_county.rename(columns={'countyFIPS':'FIPS_CODE
'})
    df_flood_risk_village=pd.read_csv("VillageFloodRisk.csv")
    
df_flood_risk_village=df_flood_risk_village.rename(columns={'VillageCode':'FIPS_C
ODE'})
    df_flood_risk = pd.concat([df_flood_risk_city_town, df_flood_risk_county, 
df_flood_risk_village], ignore_index=True)
    df_flood_risk['FIPS_CODE'] = df_flood_risk['FIPS_CODE'].astype(str)
    df_flood_risk['Extreme_Flooding']=df_flood_risk['multiplied_percent10']
    
df_flood_risk['High_Flooding']=df_flood_risk['multiplied_percent8']+df_flood_risk
['multiplied_percent9']+df_flood_risk['multiplied_percent10']
    df_flood_risk['Significant_Flooding'] = df_flood_risk['multiplied_percent5'] 
+ df_flood_risk['multiplied_percent6'] + df_flood_risk['multiplied_percent7'] + 
df_flood_risk['multiplied_percent8'] + df_flood_risk['multiplied_percent9'] + 
df_flood_risk['multiplied_percent10']
    columns_to_drop = [f'multiplied_percent{i}' for i in range(1, 11)]
    df_flood_risk.drop(columns=columns_to_drop, inplace=True)
    df_aux=pd.merge(gdf_FIPS,df_flood_risk,on='FIPS_CODE',how='left')
    df_demographic = pd.merge(df_demographic, df_aux, on=['Muni 
Name','County','Class'], how='left')
    filtro = (df_demographic['Muni Name'] == "st. lawrence") 
&(df_demographic['County'] == "st. lawrence") & (df_demographic['Class'] == 
'county')
    df_demographic.loc[filtro, 'Extreme_Flooding'] = 2.958333333
    df_demographic.loc[filtro, 'High_Flooding'] = 2.458333333 + 5.796333333 +
2.958333333
    df_demographic.loc[filtro, 'Significant_Flooding'] = 1.482333333 +
2.795666667 + 1.813 + 2.458333333 + 5.796333333 + 2.958333333
    filtro = (df_demographic['Muni Name'] == "earlville")
&(df_demographic['County'] == "chenango") & (df_demographic['Class'] ==
'village')
    df_demographic.loc[filtro, 'Extreme_Flooding'] = 1.849861013
    df_demographic.loc[filtro, 'High_Flooding'] =
4.456451854+5.928679818+1.849861013
    df_demographic.loc[filtro, 'Significant_Flooding']
=3.001051352+1.994071177+4.456451854+5.928679818+1.849861013
    return df_demographic
```

REVENUES

```
import pandas as pd
def revenues (df_demographic):
    ipc_21=1.06
    ipc_20=1.14
    ipc_19=1.15
    ipc_18=1.18
    ipc_17=1.2
    ipc_16=1.23
    ipc_15=1.25
    df_revenues_city=pd.read_excel("levelone22_cities.xlsx", header=4)
    df_revenues_city['Class']='city'
    df_revenues_city_21=pd.read_excel("levelone21_cities.xlsx", header=4)
    df_revenues_city_21['Class']='city'
    df_revenues_city_21['Total Revenues and Other Sources']=
df_revenues_city_21['Total Revenues and Other Sources']*ipc_21
    df_revenues_city_20=pd.read_excel("levelone20_cities.xlsx", header=4)
    df_revenues_city_20['Class']='city'
    df_revenues_city_20['Total Revenues and Other Sources']=
df_revenues_city_20['Total Revenues and Other Sources']*ipc_20
    df_revenues_city_19=pd.read_excel("levelone19_cities.xlsx", header=4)
    df_revenues_city_19['Class']='city'
    df_revenues_city_19['Total Revenues and Other Sources']=
df_revenues_city_19['Total Revenues and Other Sources']*ipc_19
    df_revenues_city_18=pd.read_excel("levelone18_cities.xlsx", header=4)
    df_revenues_city_18['Class']='city'
    df_revenues_city_18['Total Revenues and Other Sources']=
df_revenues_city_18['Total Revenues and Other Sources']*ipc_18
    df_revenues_city_17=pd.read_excel("levelone17_cities.xlsx", header=4)
    df_revenues_city_17['Class']='city'
    df_revenues_city_17['Total Revenues and Other Sources']=
df_revenues_city_17['Total Revenues and Other Sources']*ipc_17
    df_revenues_city_16=pd.read_excel("levelone16_cities.xlsx", header=4)
    df_revenues_city_16['Class']='city'
    df_revenues_city_16['Total Revenues and Other Sources']=
df_revenues_city_16['Total Revenues and Other Sources']*ipc_16
    df_revenues_city_15=pd.read_excel("levelone15_cities.xlsx", header=4)
    df_revenues_city_15['Class']='city'
    df_revenues_city_15['Total Revenues and Other Sources']=
df_revenues_city_15['Total Revenues and Other Sources']*ipc_15
    ind = df_revenues_city[df_revenues_city['Total Revenues and Other
Sources'].isna()].index
    df_revenues_city.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_city_21.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_city[df_revenues_city['Total Revenues and Other
Sources'].isna()].index
    df_revenues_city.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_city_20.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_city[df_revenues_city['Total Revenues and Other
Sources'].isna()].index
    df_revenues_city.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_city_19.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_city[df_revenues_city['Total Revenues and Other
Sources'].isna()].index
    df_revenues_city.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_city_18.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_city[df_revenues_city['Total Revenues and Other
Sources'].isna()].index
    df_revenues_city.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_city_17.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_city[df_revenues_city['Total Revenues and Other
Sources'].isna()].index
    df_revenues_city.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_city_16.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_city[df_revenues_city['Total Revenues and Other
Sources'].isna()].index
    df_revenues_city.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_city_15.loc[ind, 'Total Revenues and Other Sources'].values
    df_revenues_town=pd.read_excel("levelone22_towns.xlsx", header=4)
    df_revenues_town['Class']='town'
    df_revenues_town_21=pd.read_excel("levelone21_towns.xlsx", header=4)
    df_revenues_town_21['Class']='town'
    df_revenues_town_21['Total Revenues and Other Sources']=
df_revenues_town_21['Total Revenues and Other Sources']*ipc_21
    df_revenues_town_20=pd.read_excel("levelone20_towns.xlsx", header=4)
    df_revenues_town_20['Class']='town'
    df_revenues_town_20['Total Revenues and Other Sources']=
df_revenues_town_20['Total Revenues and Other Sources']*ipc_20
    df_revenues_town_19=pd.read_excel("levelone19_towns.xlsx", header=4)
    df_revenues_town_19['Class']='town'
    df_revenues_town_19['Total Revenues and Other Sources']=
df_revenues_town_19['Total Revenues and Other Sources']*ipc_19
    df_revenues_town_18=pd.read_excel("levelone18_towns.xlsx", header=4)
    df_revenues_town_18['Class']='town'
    df_revenues_town_18['Total Revenues and Other Sources']=
df_revenues_town_18['Total Revenues and Other Sources']*ipc_18
    df_revenues_town_17=pd.read_excel("levelone17_towns.xlsx", header=4)
    df_revenues_town_17['Class']='town'
    df_revenues_town_17['Total Revenues and Other Sources']=
df_revenues_town_17['Total Revenues and Other Sources']*ipc_17
```

```
    df_revenues_town_16=pd.read_excel("levelone16_towns.xlsx", header=4)
    df_revenues_town_16['Class']='town'
    df_revenues_town_16['Total Revenues and Other Sources']=
df_revenues_town_16['Total Revenues and Other Sources']*ipc_16
    df_revenues_town_15=pd.read_excel("levelone15_towns.xlsx", header=4)
    df_revenues_town_15['Class']='town'
    df_revenues_town_15['Total Revenues and Other Sources']=
df_revenues_town_15['Total Revenues and Other Sources']*ipc_15
    ind = df_revenues_town[df_revenues_town['Total Revenues and Other
Sources'].isna()].index
    df_revenues_town.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_town_21.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_town[df_revenues_town['Total Revenues and Other
Sources'].isna()].index
    df_revenues_town.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_town_20.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_town[df_revenues_town['Total Revenues and Other
Sources'].isna()].index
    df_revenues_town.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_town_19.loc[ind, 'Total Revenues and Other Sources'].values
    df_revenues_town.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_town_17.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_town[df_revenues_town['Total Revenues and Other
Sources'].isna()].index
    df_revenues_town.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_town_16.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_town[df_revenues_town['Total Revenues and Other
Sources'].isna()].index
    df_revenues_town.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_town_15.loc[ind, 'Total Revenues and Other Sources'].values
    df_revenues_county=pd.read_excel("levelone22_counties.xlsx", header=4)
    df_revenues_county['Class']='county'
    df_revenues_villas=pd.read_excel("levelone22_villas.xlsx", header=4)
    df_revenues_villas['Class']='village'
    df_revenues_villas_21=pd.read_excel("levelone21_villas.xlsx", header=4)
    df_revenues_villas_21['Class']='village'
    df_revenues_villas_21['Total Revenues and Other Sources']=
df_revenues_villas_21['Total Revenues and Other Sources']*ipc_21
    df_revenues_villas_20=pd.read_excel("levelone20_villas.xlsx", header=4)
    df_revenues_villas_20['Class']='village'
    df_revenues_villas_20['Total Revenues and Other Sources']=
df_revenues_villas_20['Total Revenues and Other Sources']*ipc_20
    df_revenues_villas_19=pd.read_excel("levelone19_villas.xlsx", header=4)
    df_revenues_villas_19['Class']='village'
    df_revenues_villas_19['Total Revenues and Other Sources']=
df_revenues_villas_19['Total Revenues and Other Sources']*ipc_19
    df_revenues_villas_18=pd.read_excel("levelone18_villas.xlsx", header=4)
    df_revenues_villas_18['Class']='village'
    df_revenues_villas_18['Total Revenues and Other Sources']=
df_revenues_villas_18['Total Revenues and Other Sources']*ipc_18
    df_revenues_villas_17=pd.read_excel("levelone17_villas.xlsx", header=4)
    df_revenues_villas_17['Class']='village'
    df_revenues_villas_17['Total Revenues and Other Sources']=
df_revenues_villas_17['Total Revenues and Other Sources']*ipc_17
    df_revenues_villas_16=pd.read_excel("levelone16_villas.xlsx", header=4)
    df_revenues_villas_16['Class']='village'
    df_revenues_villas_16['Total Revenues and Other Sources']=
df_revenues_villas_16['Total Revenues and Other Sources']*ipc_16
    df_revenues_villas_15=pd.read_excel("levelone15_villas.xlsx", header=4)
    df_revenues_villas_15['Class']='village'
    df_revenues_villas_15['Total Revenues and Other Sources']=
df_revenues_villas_15['Total Revenues and Other Sources']*ipc_15
    ind = df_revenues_villas[df_revenues_villas['Total Revenues and Other
Sources'].isna()].index
    df_revenues_villas.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_villas_21.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_villas[df_revenues_villas['Total Revenues and Other
Sources'].isna()].index
    df_revenues_villas.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_villas_20.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_villas[df_revenues_villas['Total Revenues and Other
Sources'].isna()].index
    df_revenues_villas.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_villas_19.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_villas[df_revenues_villas['Total Revenues and Other
Sources'].isna()].index
    df_revenues_villas.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_villas_18.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_villas[df_revenues_villas['Total Revenues and Other
Sources'].isna()].index
    df_revenues_villas.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_villas_17.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_villas[df_revenues_villas['Total Revenues and Other
Sources'].isna()].index
    df_revenues_villas.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_villas_16.loc[ind, 'Total Revenues and Other Sources'].values
    ind = df_revenues_villas[df_revenues_villas['Total Revenues and Other
Sources'].isna()].index
    df_revenues_villas.loc[ind, 'Total Revenues and Other Sources'] =
df_revenues_villas_15.loc[ind, 'Total Revenues and Other Sources'].values

df_revenues=pd.concat([df_revenues_city,df_revenues_town,df_revenues_county,df_re
venues_villas])
    columns=['Entity Name','County','Class','Total Revenues and Other Sources']
    df_revenues=df_revenues[columns]
    df_revenues=df_revenues.rename(columns={'Entity Name':'Muni Name'})
    df_revenues=df_revenues.rename(columns={'Total Revenues and Other
Sources':'Total Revenues'})
    df_revenues[['Muni Name','County']] = df_revenues[['Muni
Name','County']].astype(str)
    df_revenues['Muni Name'] = df_revenues['Muni Name'].apply(lambda x: '
'.join(x.split()[2:]))
    df_revenues = df_revenues.applymap(lambda x: x.lower() if isinstance(x, str)
else x)
```

```
    df_revenues['Muni Name'] = df_revenues['Muni Name'].str.replace('.', '',
regex=False)
    df_revenues['County'] = df_revenues['County'].str.replace('.', '',
regex=False)
    df_demographic = pd.merge(df_demographic, df_revenues, on=['Muni
Name','County','Class'], how='left')
    NAME=['summerhill','st. armand','saranac lake','northeast','mt vernon','mc
graw','manor haven','fallsburgh','dolgeville','dekalb','bloomingburgh','attica']

County=['cayuga','essex','essex','dutchess','westchester','cortland','nassau','su
llivan','fulton','st lawrence','sullivan','genesee']
    Class=[
'town','town','village','town','city','village','village','town','village','town'
,'village','village']

Data=[1130927.24,1881057.08,12384422,3714677,156530123.5875,1369057.16,5675159,25
839141.18,2778034.3,1903166.21,849054,3498583.27]
    for m, c, cl, d in zip(NAME, County, Class, Data):
        df_demographic.loc[(df_demographic['Muni Name'] == m) &
(df_demographic['County'] == c) & (df_demographic['Class'] == cl), 'Total
Revenues'] = d
    df_demographic['Revenues']=df_demographic['Total
Revenues']/df_demographic['Population']
    df_demographic=df_demographic.drop('Total Revenues',axis=1)
    return(df_demographic,df_revenues)
```

STRUCTURAL OR ENVIRONMENTAL PROJECTS

```
import pandas as pd
import numpy as np
def env_struc(df_demographic,df_survey_yes):
    df_total_cost = df_survey_yes.groupby('County')['Cost'].sum().reset_index()
    df_structural_cost = df_survey_yes[df_survey_yes['HSA
ID']<8].groupby('County')['Cost'].sum().reset_index()
    df_structural_cost =df_structural_cost
.rename(columns={'Cost':'structural_cost'})
    df_environmental_cost = df_survey_yes[(df_survey_yes['HSA ID']>7) &
(df_survey_yes['HSA ID']<10) ].groupby('County')['Cost'].sum().reset_index()
    df_environmental_cost =df_environmental_cost
.rename(columns={'Cost':'environmental_cost'})
    min_structural=np.percentile(df_structural_cost['structural_cost'],33.33)
    max_structural=np.percentile(df_structural_cost['structural_cost'],66.6)
    df_structural_cost_min=df_structural_cost.copy()
    df_structural_cost_max=df_structural_cost.copy()
    df_structural_cost_med=df_structural_cost.copy()
    for i in df_structural_cost.index:
        a=df_structural_cost.loc[i,'structural_cost']
        if a<=min_structural:
            df_structural_cost_max=df_structural_cost_max.drop(i)
            df_structural_cost_med=df_structural_cost_med.drop(i)
        elif a>=max_structural:
             df_structural_cost_min=df_structural_cost_min.drop([i])
             df_structural_cost_med=df_structural_cost_med.drop(i)
        else:
            df_structural_cost_min=df_structural_cost_min.drop([i])
            df_structural_cost_max=df_structural_cost_max.drop([i])

df_structural_cost_min=df_structural_cost_min.sort_values('structural_cost',ascen
ding=False)

df_structural_cost_max=df_structural_cost_max.sort_values('structural_cost',ascen
ding=False)

df_structural_cost_med=df_structural_cost_med.sort_values('structural_cost',ascen
ding=False)

min_environmental=np.percentile(df_environmental_cost['environmental_cost'],33.33
)

max_environmental=np.percentile(df_environmental_cost['environmental_cost'],66.6)
    df_environmental_cost_min=df_environmental_cost.copy()
    df_environmental_cost_max=df_environmental_cost.copy()
    df_environmental_cost_med=df_environmental_cost.copy()
    for i in df_environmental_cost.index:
        a=df_environmental_cost.loc[i,'environmental_cost']
        if a<=min_environmental:
            df_environmental_cost_med=df_environmental_cost_med.drop([i])
            df_environmental_cost_max=df_environmental_cost_max.drop([i])
        elif a>=max_environmental:
            df_environmental_cost_min=df_environmental_cost_min.drop(i)
            df_environmental_cost_med=df_environmental_cost_med.drop([i])
        else:
            df_environmental_cost_min=df_environmental_cost_min.drop(i)
            df_environmental_cost_max=df_environmental_cost_max.drop([i])

df_environmental_cost_min=df_environmental_cost_min.sort_values('environmental_co
st',ascending=False)

df_environmental_cost_max=df_environmental_cost_max.sort_values('environmental_co
st',ascending=False)

df_environmental_cost_med=df_environmental_cost_med.sort_values('environmental_co
st',ascending=False)
    df_cost_g=pd.merge(df_structural_cost,df_environmental_cost,on='County',
how='outer')
    df_cost_g = df_cost_g.fillna(0)
    df_cost_g=df_cost_g.set_index('County')

correlation=df_cost_g['environmental_cost'].corr(df_cost_g['structural_cost'])
    print(correlation)
    correlation_total=df_cost_g.corr()
    print(correlation_total)

return(df_total_cost,df_structural_cost_max,df_environmental_cost_max,df_structur
al_cost_med,df_environmental_cost_med,df_structural_cost_min,df_environmental_cos
t_min,df_structural_cost,df_environmental_cost,correlation_total)
```

VISUALIZATION

```
import matplotlib.pyplot as plt
from matplotlib.lines import Line2D
from matplotlib.patches import Patch
def
visualization(df_demographic,df_survey_all,df_survey_yes,gdf_final,gdf_counties,d
f_structural_cost_max,df_environmental_cost_max,df_structural_cost_med,df_environ
mental_cost_med,df_structural_cost_min,df_environmental_cost_min):

df_demographic_county=df_demographic.loc[df_demographic['Class']=='county'].copy(
)

df_demographic_city_town=df_demographic.loc[(df_demographic['Class']=='city')|(df
_demographic['Class']=='town')].copy()

df_demographic_village=df_demographic.loc[df_demographic['Class']=='village'].cop
y()
    df_demographic_city_town['part_county'] =
df_demographic_city_town['County'].map(df_demographic_county.set_index('County')[
'Participation'])
    df_demographic_village['part_county'] =
df_demographic_village['County'].map(df_demographic_county.set_index('County')['P
articipation'])
    df_demographic_village['part_2nd_county'] = df_demographic_village['2nd
County'].map(df_demographic_county.set_index('County')['Participation'])
    df_demographic_village['town_county'] = df_demographic_village['Town'] + '_'
+ df_demographic_village['County']
    df_demographic_city_town_aux=df_demographic_city_town.copy()
    df_demographic_city_town_aux['town_county'] =
df_demographic_city_town_aux['Muni Name'] + '_' +
df_demographic_city_town['County']
    df_demographic_city_town_aux.drop_duplicates(subset='town_county',
inplace=True)
    df_demographic_village['part_town'] =
df_demographic_village['town_county'].map(df_demographic_city_town_aux.set_index(
'town_county')['Participation'])
    df_demographic_village.drop(columns=['town_county'], inplace=True)
    df_demographic_city_town['county_survey'] =
df_demographic_city_town['County'].map(df_demographic_county.set_index('County')[
'Answer_survey'])
    df_demographic_village['county_survey'] =
df_demographic_village['County'].map(df_demographic_county.set_index('County')['A
nswer_survey'])
    df_demographic_village['2ndcounty_survey'] = df_demographic_village['2nd
County'].map(df_demographic_county.set_index('County')['Answer_survey'])
    df_demographic_village['town_county'] = df_demographic_village['Town'] + '_'
+ df_demographic_village['County']
```

```
    df_demographic_city_town_aux=df_demographic_city_town.copy()
    df_demographic_city_town_aux['town_county'] =
df_demographic_city_town_aux['Muni Name'] + '_' +
df_demographic_city_town['County']
    df_demographic_city_town_aux.drop_duplicates(subset='town_county',
inplace=True)
    df_demographic_village['town_survey'] =
df_demographic_village['town_county'].map(df_demographic_city_town_aux.set_index(
'town_county')['Answer_survey'])
    df_demographic_village.drop(columns=['town_county'], inplace=True)
    df_demographic_city_town['county_yes_survey'] =
df_demographic_city_town['County'].map(df_demographic_county.set_index('County')[
'Answer_yes_survey'])
    df_demographic_village['county_yes_survey'] =
df_demographic_village['County'].map(df_demographic_county.set_index('County')['A
nswer_yes_survey'])
    df_demographic_village['2ndcounty_yes_survey'] = df_demographic_village['2nd
County'].map(df_demographic_county.set_index('County')['Answer_yes_survey'])
    df_demographic_village['town_county'] = df_demographic_village['Town'] + '_'
+ df_demographic_village['County']
    df_demographic_city_town_aux=df_demographic_city_town.copy()
    df_demographic_city_town_aux['town_county'] =
df_demographic_city_town_aux['Muni Name'] + '_' +
df_demographic_city_town['County']
    df_demographic_city_town_aux.drop_duplicates(subset='town_county',
inplace=True)
    df_demographic_village['town_yes_survey'] =
df_demographic_village['town_county'].map(df_demographic_city_town_aux.set_index(
'town_county')['Answer_yes_survey'])
    df_demographic_village.drop(columns=['town_county'], inplace=True)
    #SMART_COMMUNITY
    fig, ax = plt.subplots(figsize=(10, 8))
    gdf_filtered = gdf_final.merge(df_demographic[df_demographic['Participation']
== 1], on=['Muni Name', 'County','Class'])
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='county'].plot(ax=ax, color=(1, 0.7,
0.7, 0.3))
        gdf_filtered[gdf_filtered['Class'].isin(['city', 'town'])].plot(ax=ax,
color=(0, 1, 0, 0.5))
        gdf_filtered[gdf_filtered['Class']=='village'].plot(ax=ax,
color='orange')
    gdf_final[gdf_final['Class'] == 'village'].plot(ax=ax, color='none',
edgecolor='blue', linewidth=0.2, linestyle='-', zorder=3,label='Village')
    gdf_final[gdf_final['Class'] == 'county'].plot(ax=ax, color='none',
edgecolor='#FF6699', linewidth=1, linestyle='-',label='County')
    gdf_final[gdf_final['Class'].isin(['city', 'town'])].plot(ax=ax,
color='none', edgecolor='#003300', linewidth=0.1, linestyle='-',
zorder=2,label='City/Town')
    legend_elements = [
        Line2D([0], [0], color='#FF6699', lw=2, label='County'),
        Line2D([0], [0], color='#003300', lw=2, label='City/Town'),
        Line2D([0], [0], color='blue', lw=2, label='Village'),
        Patch(facecolor=(1, 0.7, 0.7, 0.3), edgecolor='none',
label='Participating County'),
        Patch(facecolor=(0, 1, 0, 0.5), edgecolor='none', label='Participating
City or Town'),
        Patch(facecolor='orange', edgecolor='none', label='Participating
Village')
    ]
    ax.legend(handles=legend_elements, loc='lower left')
    plt.axis('off')
    plt.title("Participating Climate Smart Communities")
    plt.show()
    #SMART_COMMUNITY COUNTY
    fig, ax = plt.subplots(figsize=(10, 8))
    gdf_filtered = gdf_final.merge(df_demographic[df_demographic['Participation']
== 1], on=['Muni Name', 'County','Class'])
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='county'].plot(ax=ax, color=(1, 0.7,
0.7, 0.3))
        # gdf_filtered[gdf_filtered['Class'].isin(['city', 'town'])].plot(ax=ax,
color=(0, 1, 0, 0.5))
        # gdf_filtered[gdf_filtered['Class']=='village'].plot(ax=ax,
color='orange')
    gdf_final[gdf_final['Class'] == 'village'].plot(ax=ax, color='none',
edgecolor='blue', linewidth=0.2, linestyle='-', zorder=3,label='Village')
    gdf_final[gdf_final['Class'] == 'county'].plot(ax=ax, color='none',
edgecolor='#FF6699', linewidth=1, linestyle='-',label='County')
    gdf_final[gdf_final['Class'].isin(['city', 'town'])].plot(ax=ax,
color='none', edgecolor='#003300', linewidth=0.1, linestyle='-',
zorder=2,label='City/Town')
    # gdf_final[gdf_final['Class'] == 'village'].plot(ax=ax, color='none',
edgecolor='blue', linewidth=0.08, linestyle='-', zorder=3,label='Village')
    legend_elements = [
        Line2D([0], [0], color='#FF6699', lw=2, label='County'),
        Line2D([0], [0], color='#003300', lw=2, label='City/Town'),
        Line2D([0], [0], color='blue', lw=2, label='Village'),
        Patch(facecolor=(1, 0.7, 0.7, 0.3), edgecolor='none',
label='Participating County'),
        # Patch(facecolor=(0, 1, 0, 0.5), edgecolor='none', label='Participating
City or Town'),
        # Patch(facecolor='orange', edgecolor='none', label='Participating
Village')
    ]
    ax.legend(handles=legend_elements, loc='lower left')
    plt.axis('off')
    plt.title("Participating Climate Smart Counties")
    plt.show()
    #SMART_COMMUNITY CITY/TOWN
    fig, ax = plt.subplots(figsize=(10, 8))
    gdf_filtered =
gdf_final.merge(df_demographic_city_town[df_demographic_city_town['Participation'
] == 1], on=['Muni Name', 'County','Class'])
    if not gdf_filtered.empty:
        # gdf_filtered[gdf_filtered['Class']=='county'].plot(ax=ax, color=(1,
0.7, 0.7, 0.3))
        gdf_filtered[gdf_filtered['Class'].isin(['city', 'town'])].plot(ax=ax,
color=(0, 1, 0, 0.5))
        # gdf_filtered[gdf_filtered['Class']=='village'].plot(ax=ax,
color='orange')
    gdf_final[gdf_final['Class'] == 'village'].plot(ax=ax, color='none',
edgecolor='blue', linewidth=0.2, linestyle='-', zorder=3,label='Village')
    gdf_final[gdf_final['Class'] == 'county'].plot(ax=ax, color='none',
edgecolor='#FF6699', linewidth=1, linestyle='-',label='County')
    gdf_final[gdf_final['Class'].isin(['city', 'town'])].plot(ax=ax,
color='none', edgecolor='#003300', linewidth=0.1, linestyle='-',
zorder=2,label='City/Town')
    legend_elements = [
        Line2D([0], [0], color='#FF6699', lw=2, label='County'),
        Line2D([0], [0], color='#003300', lw=2, label='City/Town'),
        Line2D([0], [0], color='blue', lw=2, label='Village'),
        # Patch(facecolor=(1, 0.7, 0.7, 0.3), edgecolor='none',
label='Participating County'),
        Patch(facecolor=(0, 1, 0, 0.5), edgecolor='none', label='Participating
City or Town'),
        # Patch(facecolor='orange', edgecolor='none', label='Participating
Village')
    ]
    ax.legend(handles=legend_elements, loc='lower left')
    plt.axis('off')
    plt.title("Participating Climate Smart Cities and Towns")
    plt.show()
    fig, ax = plt.subplots(figsize=(10, 8))
    gdf_final[gdf_final['Class'].isin(['city', 'town'])].plot(ax=ax,
color='none', edgecolor='red', linewidth=0.1, linestyle='-', zorder=2)
    gdf_final[gdf_final['Class'] == 'county'].plot(ax=ax, color='none',
edgecolor='pink', linewidth=1, linestyle='-', zorder=1)
    gdf_final[gdf_final['Class'] == 'village'].plot(ax=ax, color='none',
edgecolor='blue', linewidth=0.08, linestyle='-', zorder=3)
    gdf_filtered =
gdf_final.merge(df_demographic_city_town[df_demographic_city_town['Participation'
] == 1], on=['Muni Name', 'County','Class'])
    if not gdf_filtered.empty:
        # gdf_filtered[gdf_filtered['Class']=='county'].plot(ax=ax, color=(1,
0.7, 0.7, 0.3))
        gdf_filtered[gdf_filtered['Class'].isin(['city', 'town'])].plot(ax=ax,
color=(0, 1, 0, 0.5))
        # gdf_filtered[gdf_filtered['Class']=='village'].plot(ax=ax,
color='orange')
    gdf_filtered =
gdf_final.merge(df_demographic_city_town[df_demographic_city_town['part_county']
== 1], on=['Muni Name', 'County','Class'])
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class'].isin(['city', 'town'])].plot(ax=ax,
color='none', edgecolor='green', linewidth=0.8, linestyle='-',
zorder=1,label='City/Town County_Smart')
```

```
    gdf_final[gdf_final['Class'] == 'village'].plot(ax=ax, color='none',
edgecolor='blue', linewidth=0.2, linestyle='-', zorder=3,label='Village')
    gdf_final[gdf_final['Class'] == 'county'].plot(ax=ax, color='none',
edgecolor='#FF6699', linewidth=1, linestyle='-',label='County')
    gdf_final[gdf_final['Class'].isin(['city', 'town'])].plot(ax=ax,
color='none', edgecolor='#003300', linewidth=0.1, linestyle='-',
zorder=2,label='City/Town')
    legend_elements = [
        Line2D([0], [0], color='#FF6699', lw=2, label='County'),
        Line2D([0], [0], color='#003300', lw=2, label='City/Town'),
        Line2D([0], [0], color='blue', lw=2, label='Village'),
        Line2D([0], [0], color='green', lw=2, label='City/Town: Participating
indirectly through county participation'),
        # Patch(facecolor=(1, 0.7, 0.7, 0.3), edgecolor='none',
label='Participating County'),
        Patch(facecolor=(0, 1, 0, 0.5), edgecolor='none', label='Participating
City or Town'),
        # Patch(facecolor='orange', edgecolor='none', label='Participating
Village')
    ]
    ax.legend(handles=legend_elements, loc='lower left')
    plt.axis('off')
    plt.title("Direct and Indirect Participation Climate Smart Cities and Towns")
    plt.show()
    #SMART_COMMUNITY VILLAGE
    fig, ax = plt.subplots(figsize=(10, 8))
    gdf_final[gdf_final['Class'].isin(['city', 'town'])].plot(ax=ax,
color='none', edgecolor='red', linewidth=0.1, linestyle='-', zorder=2)
    gdf_final[gdf_final['Class'] == 'county'].plot(ax=ax, color='none',
edgecolor='pink', linewidth=1, linestyle='-', zorder=1)
    gdf_final[gdf_final['Class'] == 'village'].plot(ax=ax, color='none',
edgecolor='blue', linewidth=0.08, linestyle='-', zorder=3)
    gdf_filtered =
gdf_final.merge(df_demographic_village[df_demographic_village['Participation'] ==
1], on=['Muni Name', 'County','Class'])
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='village'].plot(ax=ax,
color='orange')
    legend_elements = [
        Line2D([0], [0], color='#FF6699', lw=2, label='County'),
        Line2D([0], [0], color='#003300', lw=2, label='City/Town'),
        Line2D([0], [0], color='blue', lw=2, label='Village'),
        # Patch(facecolor=(1, 0.7, 0.7, 0.3), edgecolor='none',
label='Participating County'),
        #Patch(facecolor=(0, 1, 0, 0.5), edgecolor='none', label='Participating
City or Town'),
        Patch(facecolor='orange', edgecolor='none', label='Participating
Village')
    ]
    ax.legend(handles=legend_elements, loc='lower left')
    plt.axis('off')
    plt.title("Participating Climate Smart Villages")
    plt.show()
    fig, ax = plt.subplots(figsize=(10, 8))
    gdf_final[gdf_final['Class'].isin(['city', 'town'])].plot(ax=ax,
color='none', edgecolor='red', linewidth=0.1, linestyle='-', zorder=2)
    gdf_final[gdf_final['Class'] == 'county'].plot(ax=ax, color='none',
edgecolor='pink', linewidth=1, linestyle='-', zorder=1)
    gdf_final[gdf_final['Class'] == 'village'].plot(ax=ax, color='none',
edgecolor='blue', linewidth=0.08, linestyle='-', zorder=3)
    gdf_filtered =
gdf_final.merge(df_demographic_village[df_demographic_village['Participation'] ==
1], on=['Muni Name', 'County','Class'])
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='village'].plot(ax=ax,
color='orange')
    gdf_filtered =
gdf_final.merge(df_demographic_village[df_demographic_village['part_county'] ==
1], on=['Muni Name', 'County','Class'])
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='village'].plot(ax=ax, color='none',
edgecolor='green', linewidth=1, linestyle='-', zorder=1)

    gdf_filtered = gdf_final[gdf_final['Class'] ==
'town'].merge(df_demographic_village[df_demographic_village['town_survey'] == 1],
on=['Muni Name', 'County','Class'])
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='village'].plot(ax=ax, color='none',
edgecolor='black', linewidth=4, linestyle='-', zorder=2)
    legend_elements = [
        Line2D([0], [0], color='#FF6699', lw=2, label='County'),
        Line2D([0], [0], color='#003300', lw=2, label='City/Town'),
        Line2D([0], [0], color='blue', lw=2, label='Village'),
        Line2D([0], [0], color='green', lw=2, label='Village: Participating
indirectly through county participation'),
        Line2D([0], [0], color='black', lw=2, label='Village: Participating
indirectly through town participation'),
        # Patch(facecolor=(1, 0.7, 0.7, 0.3), edgecolor='none',
label='Participating County'),
        # Patch(facecolor=(0, 1, 0, 0.5), edgecolor='none', label='Participating
City or Town'),
        Patch(facecolor='orange', edgecolor='none', label='Participating
Village')
    ]
    ax.legend(handles=legend_elements, loc='lower left')
    plt.axis('off')
    plt.title("Direct and Indirect Participation Climate Smart Villages")
    plt.show()

    #Only 2nd county
    fig, ax = plt.subplots(figsize=(10, 8))
    gdf_final[gdf_final['Class'].isin(['city', 'town'])].plot(ax=ax,
color='none', edgecolor='red', linewidth=0.1, linestyle='-', zorder=2)
    gdf_final[gdf_final['Class'] == 'county'].plot(ax=ax, color='none',
edgecolor='pink', linewidth=1, linestyle='-', zorder=1)
```

```
    gdf_final[gdf_final['Class'] == 'village'].plot(ax=ax, color='none',
edgecolor='blue', linewidth=0.08, linestyle='-', zorder=3)
    #Rellena la villa de color naranja si pertenece al programa
    gdf_filtered =
gdf_final.merge(df_demographic_village[df_demographic_village['Participation'] ==
1], on=['Muni Name', 'County','Class'])
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='village'].plot(ax=ax,
color='orange')
    gdf_filtered =
gdf_final.merge(df_demographic_village[df_demographic_village['part_county'] ==
1], on=['Muni Name', 'County','Class'])
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='village'].plot(ax=ax, color='none',
edgecolor='green', linewidth=1, linestyle='-', zorder=1)
    gdf_filtered =
gdf_final.merge(df_demographic_village[df_demographic_village['part_2nd_county']
== 1], on=['Muni Name', 'County','Class'])
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='village'].plot(ax=ax, color='none',
edgecolor='red', linewidth=1, linestyle='-', zorder=1)
    gdf_filtered = gdf_final[gdf_final['Class'] ==
'town'].merge(df_demographic_village[df_demographic_village['town_survey'] == 1],
on=['Muni Name', 'County','Class'])
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='village'].plot(ax=ax, color='none',
edgecolor='black', linewidth=4, linestyle='-', zorder=2)
    legend_elements = [
        Line2D([0], [0], color='#FF6699', lw=2, label='County'),
        Line2D([0], [0], color='#003300', lw=2, label='City/Town'),
        Line2D([0], [0], color='blue', lw=2, label='Village'),
        Line2D([0], [0], color='green', lw=2, label='Village: Participating
indirectly through county participation'),
        Line2D([0], [0], color='red', lw=2, label='Village: Participating
indirectly through second county participation'),
        Line2D([0], [0], color='black', lw=2, label='Village: Participating
indirectly through town participation'),
        # Patch(facecolor=(1, 0.7, 0.7, 0.3), edgecolor='none',
label='Participating County'),
        # Patch(facecolor=(0, 1, 0, 0.5), edgecolor='none', label='Participating
City or Town'),
        Patch(facecolor='orange', edgecolor='none', label='Participating
Village')
    ]
    plt.rc('legend', fontsize='small')
    ax.legend(handles=legend_elements, loc='lower left')
    plt.axis('off')
    plt.title("Direct and Indirect Participation Climate Smart Villages")
    plt.show()

    #Eco_Structura projects

    #Structural projects
    fig, ax = plt.subplots(figsize=(10, 8))
    gdf_filtered = gdf_counties.merge(df_structural_cost_max)
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='county'].plot(ax=ax,
color='#333333', alpha=0.5)
    gdf_filtered = gdf_counties.merge(df_structural_cost_med)
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='county'].plot(ax=ax,
color='#666666', alpha=0.5)
    gdf_filtered = gdf_counties.merge(df_structural_cost_min)
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='county'].plot(ax=ax,
color='#CCCCCC', alpha=0.5)
    gdf_final[gdf_final['Class'] == 'county'].plot(ax=ax, color='none',
edgecolor='#FF6699', linewidth=1, linestyle='-',label='County')
    legend_elements = [
        Line2D([0], [0], color='#FF6699', lw=2, label='County'),
        Patch(color='#333333', alpha=0.5, label='County with expensive
projects'),
        Patch(color='#666666', alpha=0.5, label='County with medium-scale
projects'),
        Patch(color='#CCCCCC', alpha=0.5, label='County with cheaper projects'),
    ]
    ax.legend(handles=legend_elements, loc='lower left')
    plt.axis('off')
    plt.title("Counties with structural projects")
    plt.show()
    #Ecoprojects
    fig, ax = plt.subplots(figsize=(10, 8))
    gdf_filtered = gdf_counties.merge(df_environmental_cost_max)
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='county'].plot(ax=ax,
color='#004d00', alpha=0.5)
    gdf_filtered = gdf_counties.merge(df_environmental_cost_med)
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='county'].plot(ax=ax,
color='#009900', alpha=0.5)
    gdf_filtered = gdf_counties.merge(df_environmental_cost_min)
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='county'].plot(ax=ax,
color='#66CC66', alpha=0.5)
    gdf_final[gdf_final['Class'] == 'county'].plot(ax=ax, color='none',
edgecolor='#FF6699', linewidth=1, linestyle='-',label='County')
    legend_elements = [
        Line2D([0], [0], color='#FF6699', lw=2, label='County'),
        Patch(color='#004d00', alpha=0.5, label='County with expensive
projects'),
        Patch(color='#009900', alpha=0.5, label='County with medium-scale
projects'),
        Patch(color='#66CC66', alpha=0.5, label='County with cheaper projects'),
    ]
    ax.legend(handles=legend_elements, loc='lower left')
    plt.axis('off')
    plt.title("Counties with environmental projects")
    plt.show()
    #Structural and environmental projects
    fig, ax = plt.subplots(figsize=(10, 8))
    gdf_filtered = gdf_counties.merge(df_structural_cost_max)
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='county'].plot(ax=ax,
color='#666666', alpha=0.3)
    gdf_filtered = gdf_counties.merge(df_environmental_cost_max)
    if not gdf_filtered.empty:
        gdf_filtered[gdf_filtered['Class']=='county'].plot(ax=ax,
color='#80FF80', alpha=0.3)
    gdf_final[gdf_final['Class'] == 'county'].plot(ax=ax, color='none',
edgecolor='#FF6699', linewidth=1, linestyle='-',label='County')
    legend_elements = [
        Line2D([0], [0], color='#FF6699', lw=2, label='County'),
        Patch(color='#666666', alpha=0.3, label='County with expensive structural
projects'),
        Patch(color='#80FF80', alpha=0.3, label='County with expensive
environmental projects'),
        Patch(color='#73B380', alpha=0.5, label='County with expensive structural
and environmental projects'),

    ]
    ax.legend(handles=legend_elements, loc='lower left')
    plt.axis('off')
    plt.title("Counties with expensive projects")
    plt.show()
    return()
```

PARTICIPATION

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import StandardScaler
from imblearn.over_sampling import SMOTE
from sklearn.ensemble import RandomForestClassifier
from sklearn.inspection import permutation_importance
import numpy as np
import hashlib
import statsmodels.api as sm
from statsmodels.discrete.discrete_model import Probit

def total_participation(df_demographic,corr):
    df_final_aux=df_demographic.copy()
    nan= df_final_aux['Revenues'].isnull() |
df_final_aux['Extreme_Flooding'].isnull()
    df_no_study = df_final_aux[nan]
    df_final_aux = df_final_aux.dropna(subset=['Revenues', 'Extreme_Flooding'])
    Y = df_final_aux["Participation"]
    X= df_final_aux.drop(['Muni Name','Class','County','Town','2nd
County','GEO_ID','Participation','Answer_survey','Answer_yes_survey'], axis=1)
    total_ones = np.sum(Y==1)
    total_zeros = np.sum(Y==0)
    relation=total_ones/(total_zeros+total_ones)
    if(relation<0.3 or relation>0.7):
        data_str = str(X) + str(Y)
        hash_object = hashlib.md5(data_str.encode())
        semilla = int(hash_object.hexdigest(), 16) % 4294967295
        smote = SMOTE(sampling_strategy=0.4, random_state=semilla)
        X,Y = smote.fit_resample(X, Y)
        print("SMOTE")
    columns_orig=X.columns
    index_orig=X.index
    scaler = StandardScaler()
    X = scaler.fit_transform(X)
    X=pd.DataFrame(X, columns=columns_orig,index=index_orig)
    high_corr = X.corr().abs() >= corr
    removed_vars = set()
    for col in range(len(high_corr.columns)):
        for index in range(col):
            if high_corr.iloc[col, index]:
                var1 = high_corr.columns[col]
                var2 = high_corr.columns[index]
                if var1 in removed_vars or var2 in removed_vars:
                    continue
                correlation = X[[var1, var2]].corr().iloc[0, 1]
                X= X.drop(columns=[var1])
                removed_vars.add(var1)
                print(f"The variables {var1} and {var2} are highly correlated
with a correlation of {correlation:.2f}. Variable {var1} was eliminated.")
    X_probit=X.copy()
    X_random_forest=X.copy()
    X_probit = sm.add_constant(X_probit)
    num_datos = len(X)
    num_datos_str = str(num_datos)
    hash_object = hashlib.md5(num_datos_str.encode())
    random_seed = int(hash_object.hexdigest(), 16) % 4294967295
    X_train_probit, X_test_probit, Y_train, Y_test = train_test_split(X_probit,
Y, test_size=0.2, random_state=random_seed)
    X_train_random_forest, X_test_random_forest, Y_train, Y_test =
train_test_split(X_random_forest, Y, test_size=0.2, random_state=random_seed)
    probit_model_train = sm.Probit(Y_train, X_train_probit)
    probit_model_train_results = probit_model_train.fit()
    probit_predicted_Y = probit_model_train_results.predict(X_test_probit)
    probit_predicted_Y= (probit_predicted_Y > 0.5).astype(int)
    probit_accuracy = (probit_predicted_Y == Y_test).mean()
    random_forest_model_train = RandomForestClassifier(random_state=random_seed)
    random_forest_model_train_results =
random_forest_model_train.fit(X_train_random_forest,Y_train)
    random_forest_predicted_Y =
random_forest_model_train_results.predict(X_test_random_forest)
```

```
    random_forest_predicted_Y= (random_forest_predicted_Y > 0.5).astype(int)
    random_forest_accuracy = (random_forest_predicted_Y == Y_test).mean()
    probit_matrix = confusion_matrix(Y_test, probit_predicted_Y)
    plt.figure(figsize=(8, 6))
    sns.heatmap(probit_matrix, annot=True, fmt="d", cmap="Blues", cbar=True)
    plt.xlabel('Predicted Participation')
    plt.ylabel('Actual Participation')
    plt.title('Participating Communities: Probit Model Confusion Matrix')
    plt.show()
    random_forest_matrix = confusion_matrix(Y_test, random_forest_predicted_Y)
    plt.figure(figsize=(8, 6))
    sns.heatmap(random_forest_matrix, annot=True, fmt="d", cmap="Blues", 
cbar=True)
    plt.xlabel('Predicted Participation')
    plt.ylabel('Actual Participation')
    plt.title('Participating Communities: Random Forest Model Confusion Matrix')
    plt.show()
    probit_model = Probit(Y, X_probit.astype(float))
    probit_model_results = probit_model.fit()
    print(probit_model_results.summary())
    probit_model_coefs = probit_model_results.params
    probit_model_p_values = probit_model_results.pvalues
    probit_results_df = pd.DataFrame({'Coefficient': probit_model_coefs, 'p 
values': probit_model_p_values})
    probit_results_df = probit_results_df.sort_values(by='p values')
    print(probit_results_df)
    random_forest_model = RandomForestClassifier(random_state=random_seed)
    random_forest_model_results= random_forest_model.fit(X_random_forest,Y)
    n_repeats = int(hash_object.hexdigest(), 16) % 100 + 1
    variables = random_forest_model_results.feature_importances_
    gini_variables_df = pd.DataFrame({'Variable': X_random_forest.columns, 
'Importance': variables})
    gini_variables_df = gini_variables_df.sort_values(by='Importance', 
ascending=False)
    variables = permutation_importance(random_forest_model_results, 
X_random_forest, Y, n_repeats=n_repeats, random_state=random_seed)
    perm_variables = variables.importances_mean
    perm_variables_df = pd.DataFrame({'Variable': X_random_forest.columns, 
'Importance': perm_variables})
    perm_variables_df = perm_variables_df.sort_values(by='Importance', 
ascending=False)
    print("Gini\n", gini_variables_df.to_string(index=False))
    print("Permu\n", perm_variables_df.to_string(index=False))

return(probit_results_df,probit_accuracy,gini_variables_df,perm_variables_df,rand
om_forest_accuracy,df_no_study)

def class_participation(df_demographic,corr,class_):
    if class_=='city':
        df_final_aux = df_demographic[(df_demographic['Class']=='city') | 
(df_demographic['Class']=='town')].copy()
    else:
         df_final_aux = df_demographic[(df_demographic['Class']==class_)].copy()
    nan = df_final_aux['Revenues'].isnull() |
df_final_aux['Extreme_Flooding'].isnull()
    df_no_study = df_final_aux[nan]
    df_final_aux = df_final_aux.dropna(subset=['Revenues', 'Extreme_Flooding'])
    Y = df_final_aux["Participation"]
    X= df_final_aux.drop(['Muni Name','Class','County','Town','2nd
County','GEO_ID','Participation','Answer_survey','Answer_yes_survey'], axis=1)
    total_ones = np.sum(Y==1)
    total_zeros = np.sum(Y==0)
    relation=total_ones/(total_zeros+total_ones)
    if(relation<0.3 or relation>0.7):
        data_str = str(X) + str(Y)
        hash_object = hashlib.md5(data_str.encode())
        semilla = int(hash_object.hexdigest(), 16) % 4294967295
        smote = SMOTE(sampling_strategy=0.4, random_state=semilla)
        X,Y = smote.fit_resample(X, Y)
        print("SMOTE")
    columns_orig=X.columns
    index_orig=X.index
    scaler = StandardScaler()
    X = scaler.fit_transform(X)
    X=pd.DataFrame(X, columns=columns_orig,index=index_orig)
    high_corr = X.corr().abs() >= corr
    removed_vars = set()
    for col in range(len(high_corr.columns)):
        for index in range(col):
            if high_corr.iloc[col, index]:
                var1 = high_corr.columns[col]
                var2 = high_corr.columns[index]
                if var1 in removed_vars or var2 in removed_vars:
                    continue
                correlation = X[[var1, var2]].corr().iloc[0, 1]
                X= X.drop(columns=[var1])
                removed_vars.add(var1)
                print(f"The variables {var1} and {var2} are highly correlated
with a correlation of {correlation:.2f}. Variable {var1} was eliminated.")
    X_probit=X.copy()
    X_random_forest=X.copy()
    X_probit = sm.add_constant(X_probit)
    num_datos = len(X)
    num_datos_str = str(num_datos)
    hash_object = hashlib.md5(num_datos_str.encode())
    random_seed = int(hash_object.hexdigest(), 16) % 4294967295
    X_train_probit, X_test_probit, Y_train, Y_test = train_test_split(X_probit,
Y, test_size=0.2, random_state=random_seed)
    X_train_random_forest, X_test_random_forest, Y_train, Y_test =
train_test_split(X_random_forest, Y, test_size=0.2, random_state=random_seed)
    probit_model_train = sm.Probit(Y_train, X_train_probit)
    probit_model_train_results = probit_model_train.fit()
    probit_predicted_Y = probit_model_train_results.predict(X_test_probit)
    probit_predicted_Y= (probit_predicted_Y > 0.5).astype(int)
    probit_accuracy = (probit_predicted_Y == Y_test).mean()
    random_forest_model_train = RandomForestClassifier(random_state=random_seed)
    random_forest_model_train_results = 
random_forest_model_train.fit(X_train_random_forest,Y_train)
    random_forest_predicted_Y = 
random_forest_model_train_results.predict(X_test_random_forest)
    random_forest_predicted_Y= (random_forest_predicted_Y > 0.5).astype(int)
    random_forest_accuracy = (random_forest_predicted_Y == Y_test).mean()
    probit_matrix = confusion_matrix(Y_test, probit_predicted_Y)
    plt.figure(figsize=(8, 6))
    sns.heatmap(probit_matrix, annot=True, fmt="d", cmap="Blues", cbar=True)
    plt.xlabel('Predicted Participation')
    plt.ylabel('Actual Participation')
    if(class_=='county'):
        plt.title('Participating Counties: Probit Model Confusion Matrix')
    elif(class_=='village'):
        plt.title('Participating Villages: Probit Model Confusion Matrix')
    else:
        plt.title('Participating Cities and Towns: Probit Model Confusion 
Matrix')
    plt.show()
    random_forest_matrix = confusion_matrix(Y_test, random_forest_predicted_Y)
    plt.figure(figsize=(8, 6))
    sns.heatmap(random_forest_matrix, annot=True, fmt="d", cmap="Blues", 
cbar=True)
    plt.xlabel('Predicted Participation')
    plt.ylabel('Actual Participation')
    if(class_=='county'):
        plt.title('Participating Counties: Random Forest Model Confusion Matrix')
    elif(class_=='village'):
        plt.title('Participating Villages: Random Forest Model Confusion Matrix')
    else:
        plt.title('Participating Cities and Towns: Random Forest Model Confusion 
Matrix')
    plt.show()
    probit_model = Probit(Y, X_probit.astype(float))
    probit_model_results = probit_model.fit()
    print(probit_model_results.summary())
    probit_model_coefs = probit_model_results.params
    probit_model_p_values = probit_model_results.pvalues
    probit_results_df = pd.DataFrame({'Coefficient': probit_model_coefs, 'p 
values': probit_model_p_values})
    probit_results_df = probit_results_df.sort_values(by='p values')
    print(probit_results_df)
    random_forest_model = RandomForestClassifier(random_state=random_seed)
    random_forest_model_results= random_forest_model.fit(X_random_forest,Y)
    variables = random_forest_model_results.feature_importances_
    gini_variables_df = pd.DataFrame({'Variable': X_random_forest.columns, 
'Importance': variables})
    gini_variables_df = gini_variables_df.sort_values(by='Importance', 
ascending=False)
    n_repeats = int(hash_object.hexdigest(), 16) % 100 + 1
    variables = permutation_importance(random_forest_model_results, 
X_random_forest, Y, n_repeats=n_repeats, random_state=random_seed)
    perm_variables = variables.importances_mean
    perm_variables_df = pd.DataFrame({'Variable': X_random_forest.columns,
'Importance': perm_variables})
    perm_variables_df = perm_variables_df.sort_values(by='Importance',
ascending=False)
    print("Gini",gini_variables_df)
    print("Permu",perm_variables_df)

return(probit_results_df,probit_accuracy,gini_variables_df,perm_variables_df,rand
om_forest_accuracy,df_no_study)


def total_yes(df_demographic,corr):
    df_final_aux = df_demographic[df_demographic['Participation'] == 1].copy()
    nan = df_final_aux['Revenues'].isnull() |
df_final_aux['Extreme_Flooding'].isnull()
    df_no_study = df_final_aux[nan]
    df_final_aux = df_final_aux.dropna(subset=['Revenues', 'Extreme_Flooding'])
    Y = df_final_aux["Answer_yes_survey"]
    X= df_final_aux.drop(['Muni Name','Class','County','Town','2nd
County','GEO_ID','Participation','Answer_survey','Answer_yes_survey'], axis=1)
    total_ones = np.sum(Y==1)
    total_zeros = np.sum(Y==0)
    relation=total_ones/(total_zeros+total_ones)
    if(relation<0.3 or relation>0.7):
        data_str = str(X) + str(Y)
        hash_object = hashlib.md5(data_str.encode())
        semilla = int(hash_object.hexdigest(), 16) % 4294967295
        smote = SMOTE(sampling_strategy=0.4, random_state=semilla)
        X,Y = smote.fit_resample(X, Y)
        print("SMOTE")
    columns_orig=X.columns
    index_orig=X.index
    scaler = StandardScaler()
    X = scaler.fit_transform(X)
    X=pd.DataFrame(X, columns=columns_orig,index=index_orig)
    high_corr = X.corr().abs() >= corr
    removed_vars = set()
    for col in range(len(high_corr.columns)):
        for index in range(col):
            if high_corr.iloc[col, index]:
                var1 = high_corr.columns[col]
                var2 = high_corr.columns[index]
                if var1 in removed_vars or var2 in removed_vars:
                    continue
                correlation = X[[var1, var2]].corr().iloc[0, 1]
                X= X.drop(columns=[var1])
                removed_vars.add(var1)
                print(f"The variables {var1} and {var2} are highly correlated
with a correlation of {correlation:.2f}. Variable {var1} was eliminated.")
    X_probit=X.copy()
    X_random_forest=X.copy()
    X_probit = sm.add_constant(X_probit)
    num_datos = len(X)
    num_datos_str = str(num_datos)
    hash_object = hashlib.md5(num_datos_str.encode())
    random_seed = int(hash_object.hexdigest(), 16) % 4294967295
    X_train_probit, X_test_probit, Y_train, Y_test = train_test_split(X_probit,
Y, test_size=0.2, random_state=random_seed)
    X_train_random_forest, X_test_random_forest, Y_train, Y_test =
train_test_split(X_random_forest, Y, test_size=0.2, random_state=random_seed)
    probit_model_train = sm.Probit(Y_train, X_train_probit)
    probit_model_train_results = probit_model_train.fit()
    probit_predicted_Y = probit_model_train_results.predict(X_test_probit)
    probit_predicted_Y= (probit_predicted_Y > 0.5).astype(int)
    probit_accuracy = (probit_predicted_Y == Y_test).mean()
    random_forest_model_train = RandomForestClassifier(random_state=random_seed)
    random_forest_model_train_results =
random_forest_model_train.fit(X_train_random_forest,Y_train)
    random_forest_predicted_Y =
random_forest_model_train_results.predict(X_test_random_forest)
    random_forest_predicted_Y= (random_forest_predicted_Y > 0.5).astype(int)
    random_forest_accuracy = (random_forest_predicted_Y == Y_test).mean()
    probit_matrix = confusion_matrix(Y_test, probit_predicted_Y)
    plt.figure(figsize=(8, 6))
    sns.heatmap(probit_matrix, annot=True, fmt="d", cmap="RdPu", cbar=True)
    plt.xlabel('Predicted: Conducting Projects')
    plt.ylabel('Actual: Conducting Projects')
    plt.title('Communities Conducting Projects: Probit Model Confusion Matrix')
    plt.show()
    random_forest_matrix = confusion_matrix(Y_test, random_forest_predicted_Y)
    plt.figure(figsize=(8, 6))
    sns.heatmap(random_forest_matrix, annot=True, fmt="d", cmap="RdPu",
cbar=True)
    plt.xlabel('Predicted: Conducting Projects')
    plt.ylabel('Actual: Conducting Projects')
    plt.title('Communities Conducting Projects: Random Forest Model Confusion
Matrix')
    plt.show()
    probit_model = Probit(Y, X_probit.astype(float))
    probit_model_results = probit_model.fit()
    print(probit_model_results.summary())
    probit_model_coefs = probit_model_results.params
    probit_model_p_values = probit_model_results.pvalues
    probit_results_df = pd.DataFrame({'Coefficient': probit_model_coefs, 'p
values': probit_model_p_values})
    probit_results_df = probit_results_df.sort_values(by='p values')
    print(probit_results_df)
    random_forest_model = RandomForestClassifier(random_state=random_seed)
    random_forest_model_results= random_forest_model.fit(X_random_forest,Y)
    variables = random_forest_model_results.feature_importances_
    gini_variables_df = pd.DataFrame({'Variable': X_random_forest.columns,
'Importance': variables})
    gini_variables_df = gini_variables_df.sort_values(by='Importance',
ascending=False)
    n_repeats = int(hash_object.hexdigest(), 16) % 100 + 1
    variables = permutation_importance(random_forest_model_results,
X_random_forest, Y, n_repeats=n_repeats, random_state=random_seed)
    perm_variables = variables.importances_mean
    perm_variables_df = pd.DataFrame({'Variable': X_random_forest.columns,
'Importance': perm_variables})
    perm_variables_df = perm_variables_df.sort_values(by='Importance',
ascending=False)
    print("Gini",gini_variables_df)
    print("Permu",perm_variables_df)

return(probit_results_df,probit_accuracy,gini_variables_df,perm_variables_df,rand
om_forest_accuracy,df_no_study)

def class_yes(df_demographic,corr,class_):
    if class_=='city':
        df_final_aux = df_demographic[((df_demographic['Class']=='city') |
(df_demographic['Class']=='town'))& (df_demographic['Participation']==1)].copy()
    else:
         df_final_aux = df_demographic[(df_demographic['Class']==class_)&
(df_demographic['Participation']==1)].copy()
    nan = df_final_aux['Revenues'].isnull() |
df_final_aux['Extreme_Flooding'].isnull()
    df_no_study = df_final_aux[nan]
    df_final_aux = df_final_aux.dropna(subset=['Revenues', 'Extreme_Flooding'])
    Y = df_final_aux["Answer_yes_survey"]
    X= df_final_aux.drop(['Muni Name','Class','County','Town','2nd
County','GEO_ID','Participation','Answer_survey','Answer_yes_survey'], axis=1)
    total_ones = np.sum(Y==1)
    total_zeros = np.sum(Y==0)
    relation=total_ones/(total_zeros+total_ones)
    if(relation<0.3 or relation>0.7):
        data_str = str(X) + str(Y)
        hash_object = hashlib.md5(data_str.encode())
        semilla = int(hash_object.hexdigest(), 16) % 4294967295
        smote = SMOTE(sampling_strategy=0.4, random_state=semilla)
        X,Y = smote.fit_resample(X, Y)
        print("SMOTE")
    columns_orig=X.columns
    index_orig=X.index
    scaler = StandardScaler()
    X = scaler.fit_transform(X)
    X=pd.DataFrame(X, columns=columns_orig,index=index_orig)
    high_corr = X.corr().abs() >= corr
    removed_vars = set()
    for col in range(len(high_corr.columns)):
        for index in range(col):
            if high_corr.iloc[col, index]:
                var1 = high_corr.columns[col]
                var2 = high_corr.columns[index]
                if var1 in removed_vars or var2 in removed_vars:
                    continue
                correlation = X[[var1, var2]].corr().iloc[0, 1]
                X= X.drop(columns=[var1])
                removed_vars.add(var1)
                print(f"The variables {var1} and {var2} are highly correlated
with a correlation of {correlation:.2f}. Variable {var1} was eliminated.")
    X_probit=X.copy()
    X_random_forest=X.copy()
    X_probit = sm.add_constant(X_probit)
    num_datos = len(X)
    num_datos_str = str(num_datos)
    hash_object = hashlib.md5(num_datos_str.encode())
    random_seed = int(hash_object.hexdigest(), 16) % 4294967295
    X_train_probit, X_test_probit, Y_train, Y_test = train_test_split(X_probit,
Y, test_size=0.2, random_state=random_seed)
    X_train_random_forest, X_test_random_forest, Y_train, Y_test =
train_test_split(X_random_forest, Y, test_size=0.2, random_state=random_seed)
    probit_model_train = sm.Probit(Y_train, X_train_probit)
    probit_model_train_results = probit_model_train.fit()
    probit_predicted_Y = probit_model_train_results.predict(X_test_probit)
    probit_predicted_Y= (probit_predicted_Y > 0.5).astype(int)
    probit_accuracy = (probit_predicted_Y == Y_test).mean()
    random_forest_model_train = RandomForestClassifier(random_state=random_seed)
    random_forest_model_train_results =
random_forest_model_train.fit(X_train_random_forest,Y_train)
    random_forest_predicted_Y =
random_forest_model_train_results.predict(X_test_random_forest)
    random_forest_predicted_Y= (random_forest_predicted_Y > 0.5).astype(int)
    random_forest_accuracy = (random_forest_predicted_Y == Y_test).mean()
    probit_matrix = confusion_matrix(Y_test, probit_predicted_Y)
    plt.figure(figsize=(8, 6))
    sns.heatmap(probit_matrix, annot=True, fmt="d", cmap="RdPu", cbar=True)
    plt.xlabel('Predicted: Conducting Projects')
    plt.ylabel('Actual: Conducting Projects')
    if(class_=='county'):
        plt.title('Counties Conducting Projects: Probit Model Confusion Matrix')
    elif(class_=='village'):
        plt.title('Villages Conducting Projects: Probit Model Confusion Matrix')
    else:
        plt.title('Cities and Towns Conducting Projects: Probit Model Confusion
Matrix')
    plt.show()
    plt.show()
    random_forest_matrix = confusion_matrix(Y_test, random_forest_predicted_Y)
    plt.figure(figsize=(8, 6))
    sns.heatmap(random_forest_matrix, annot=True, fmt="d", cmap="RdPu",
cbar=True)
    plt.xlabel('Predicted: Conducting Projects')
    plt.ylabel('Actual: Conducting Projects')
    if(class_=='county'):
        plt.title('Counties Conducting Projects: Random Forest Model Confusion
Matrix')
    elif(class_=='village'):
        plt.title('Villages Conducting Projects: Random Forest Model Confusion
Matrix')
    else:
        plt.title('Cities and Towns Conducting Projects: Random Forest Model 
Confusion Matrix')
    plt.show()
    probit_model = Probit(Y, X_probit.astype(float))
    probit_model_results = probit_model.fit()
    print(probit_model_results.summary())
    probit_model_coefs = probit_model_results.params
    probit_model_p_values = probit_model_results.pvalues
    probit_results_df = pd.DataFrame({'Coefficient': probit_model_coefs, 'p 
values': probit_model_p_values})
    probit_results_df = probit_results_df.sort_values(by='p values')
    print(probit_results_df)
    random_forest_model = RandomForestClassifier(random_state=random_seed)
    random_forest_model_results= random_forest_model.fit(X_random_forest,Y)
    variables = random_forest_model_results.feature_importances_
    gini_variables_df = pd.DataFrame({'Variable': X_random_forest.columns, 
'Importance': variables})
    gini_variables_df = gini_variables_df.sort_values(by='Importance', 
ascending=False)
    variables = permutation_importance(random_forest_model_results, 
X_random_forest, Y, n_repeats=30, random_state=42)
    perm_variables = variables.importances_mean
    perm_variables_df = pd.DataFrame({'Variable': X_random_forest.columns, 
'Importance': perm_variables})
    perm_variables_df = perm_variables_df.sort_values(by='Importance', 
ascending=False)
    print("Gini",gini_variables_df)
    print("Permu",perm_variables_df)

return(probit_results_df,probit_accuracy,gini_variables_df,perm_variables_df,rand
om_forest_accuracy,df_no_study)
```

ECONOMIC MODEL

```
import pandas as pd
from sklearn.preprocessing import StandardScaler
import statsmodels.api as sm
import hashlib
from sklearn.metrics import mean_squared_error,r2_score,mean_absolute_error
import scipy.stats as stats
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from statsmodels.stats.stattools import durbin_watson
def OLS(df_demographic,df_survey_yes,corr):
    df_final_aux = df_survey_yes.copy()
    df_final_aux = pd.merge(df_final_aux, df_demographic, on=['Muni Name', 
'County', 'Class'], how='left')
    df_final_aux = df_final_aux.fillna(0)
    df_final_aux['Fed_Funded'] = df_final_aux['Cost'] * df_final_aux['Pct 
Federal'] / 100
```

```
    df_final_aux['State_Funded'] = df_final_aux['Cost'] * df_final_aux['Pct 
State'] / 100
    df_final_aux['Local_Funded'] = df_final_aux['Cost'] * df_final_aux['Pct 
Local'] / 100
    df_final_aux['Cost_CC'] = df_final_aux['Cost'] * df_final_aux['Pct Cost CC'] 
/ 100
    df_grouped = df_final_aux.groupby(['Muni Name', 'County', 'Class']).agg({
            'Cost': 'sum',
            'Population': 'mean',
            'Fed_Funded': 'sum',
            'State_Funded': 'sum',
            'Local_Funded': 'sum',
            'Cost_CC': 'sum',
            'Male_%': 'mean',
            'Black_%': 'mean',
            'Asian_%': 'mean',
            'Hispanic_%': 'mean',
            'Nonwhite_%': 'mean',
            'Employment_%': 'mean',
            'Degree_%': 'mean',
            'Median_income': 'mean',
            'Poverty_%': 'mean',
            'English_%': 'mean',
            'Rent_%': 'mean',
            'Disability_%': 'mean',
            'Revenues': 'mean',
            'Extreme_Flooding': 'mean',
            'High_Flooding': 'mean',
            'Significant_Flooding': 'mean',
            'drought_1': 'mean',
            'erosion_1': 'mean',
            'extreme heat_1': 'mean',
            'extreme weather_1': 'mean',
            'flooding (not related to sea-level rise)_1': 'sum',
            'sea-level rise_1': 'sum',
            'drought_2': 'sum',
            'erosion_2': 'sum',
            'extreme heat_2': 'sum',
            'extreme weather_2': 'sum',
            'flooding (not related to sea-level rise)_2': 'sum',
            'sea-level rise_2': 'sum'
        }).reset_index()
    df_grouped['Pct Federal'] = (df_grouped['Fed_Funded'] / df_grouped['Cost']) * 
100
    df_grouped['Pct State'] = (df_grouped['State_Funded'] / df_grouped['Cost']) * 
100
    df_grouped['Pct Local'] = (df_grouped['Local_Funded'] / df_grouped['Cost']) * 
100
    df_grouped['Pct Cost CC'] = (df_grouped['Cost_CC'] / df_grouped['Cost']) * 
100
    df_grouped = df_grouped.drop(columns=['Fed_Funded', 'State_Funded', 
'Local_Funded', 'Cost_CC'])
    df_aggregated = df_grouped.copy()
    df_aggregated = df_aggregated.fillna(0)
    df_aggregated['LogCost'] = np.log1p(df_aggregated['Cost'])
    Y = df_aggregated["LogCost"]
    X = df_aggregated.drop(['Muni Name', 'County', 'Class', 'Cost','LogCost'],
axis=1)
    columns_orig = X.columns
    index_orig = X.index
    scaler = StandardScaler()
    X = scaler.fit_transform(X)
    X = pd.DataFrame(X, columns=columns_orig, index=index_orig)
    high_corr = X.corr().abs() >= corr
    removed_vars = set()
    for col in range(len(high_corr.columns)):
        for index in range(col):
            if high_corr.iloc[col, index]:
                var1 = high_corr.columns[col]
                var2 = high_corr.columns[index]
                if var1 in removed_vars or var2 in removed_vars:
                    continue
                correlation = X[[var1, var2]].corr().iloc[0, 1]
                X = X.drop(columns=[var1])
                removed_vars.add(var1)
                print(f"The variables {var1} and {var2} are highly correlated
with a correlation of {correlation:.2f}. Variable {var1} was eliminated.")
    X = sm.add_constant(X)
    num_datos = len(X)
    num_datos_str = str(num_datos)
    hash_object = hashlib.md5(num_datos_str.encode())
    random_seed = int(hash_object.hexdigest(), 16) % 4294967295
    X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2,
random_state=random_seed)
    model_train = sm.OLS(Y_train, X_train)
    results_train = model_train.fit()
    residuals_train = results_train.resid
    y_pred = results_train.fittedvalues
    plt.scatter(y_pred, residuals_train)
    plt.axhline(y=0, color='r', linestyle='--')
    plt.title('Residuals versus Fitted Values (Training Set)' )
    plt.xlabel('Fitted Values of Project Costs')
    plt.ylabel('Residuals')
    plt.show()
    stats.probplot(residuals_train, dist="norm", plot=plt)
    plt.title('Q-Q Plot of Residuals (Training Set)')
    plt.xlabel('Theoretical Quantiles')
    plt.ylabel('Observed Quantiles')
    plt.show()
    standardized_residuals_train = residuals_train / np.std(residuals_train)
    plt.scatter(y_pred, np.sqrt(np.abs(standardized_residuals_train)))
    plt.axhline(y=0, color='r', linestyle='--')
    plt.title('Homoscedasticity (Training Set)')
    plt.xlabel('Fitted Values')
    plt.ylabel('Sqrt(Standardized Residuals)')
    plt.show()
    dw_stat = durbin_watson(residuals_train)
    print('Durbin-Watson statistic:', dw_stat)
    Y_test_pred = results_train.predict(X_test)
    mse_ols_test = mean_squared_error(Y_test, Y_test_pred)
    print("Mean Squared Error (MSE) - Test (OLS Model):", mse_ols_test)
    r2_ols_test = r2_score(Y_test, Y_test_pred)
    print("Coefficient of Determination (R^2) - Test (OLS Model):", r2_ols_test)
    mae_ols_test = mean_absolute_error(Y_test, Y_test_pred)
    print("Mean Absolute Error (MAE) - Test (OLS Model):", mae_ols_test)
    model = sm.OLS(Y, X)
    results = model.fit()
    residuals = results.resid
    y_pred = results.fittedvalues
    plt.scatter(y_pred, residuals)
    plt.axhline(y=0, color='r', linestyle='--')
    plt.title('Residuals versus Fitted Values')
    plt.xlabel('Fitted Values')
    plt.ylabel('Residuals')
    plt.show()
    stats.probplot(residuals, dist="norm", plot=plt)
    plt.title('Q-Q Plot of Residuals')
    plt.xlabel('Theoretical Quantiles')
    plt.ylabel('Observed Quantiles')
    plt.show()
    standardized_residuals= residuals / np.std(residuals)
    plt.scatter(y_pred, np.sqrt(np.abs(standardized_residuals)))
    plt.axhline(y=0, color='r', linestyle='--')
    plt.title('Homoscedasticity')
    plt.xlabel('Fitted Values')
    plt.ylabel('Sqrt(Standardized Residuals)')
    plt.show()
    dw_stat = durbin_watson(residuals)
    print('Durbin-Watson statistic:', dw_stat)
    results_df = pd.DataFrame({'Coefficient': results.params, 'p values':
results.pvalues})
    results_df = results_df.sort_values(by='p values')
    print(results_df)
    return(columns_orig,results_df)
```

PROPOSAL

```
import pandas as pd
import hashlib
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
import numpy as np
from scipy.spatial.distance import cdist
from sklearn.metrics import silhouette_score
import warnings
warnings.filterwarnings("ignore", message="KMeans is known to have a memory leak
on Windows with MKL")
```

```
def prepare_clustering(df_demographic,variables,class_):
    if class_=='city':
        df_demographic_proposal =
df_demographic[(df_demographic['Class']=='city') |
(df_demographic['Class']=='town')].copy()
    elif class_=='all':
        df_demographic_proposal=df_demographic.copy()
    else:
         df_demographic_proposal =
df_demographic[(df_demographic['Class']==class_)].copy()
    df_demographic_proposal = df_demographic_proposal[variables]
    df_demographic_proposal = df_demographic_proposal.dropna()
    answer_yes_survey_list =
df_demographic_proposal['Answer_yes_survey'].tolist()
    df_demographic_proposal.drop(columns=['Answer_yes_survey'], inplace=True)
    scaler = StandardScaler()
    scaled_df_demographic_proposal =
scaler.fit_transform(df_demographic_proposal)
    df_demographic_proposal_str = df_demographic_proposal.to_string()
    hash_object = hashlib.md5(df_demographic_proposal_str.encode())
    semilla = int(hash_object.hexdigest(), 16) % 4294967295
    sse = []
    silhouette_scores = []
    range_clusters = range(2, 11)
    for k in range_clusters:
        kmeans = KMeans(n_clusters=k, random_state=semilla, n_init=10)
        kmeans.fit(scaled_df_demographic_proposal)
        sse.append(kmeans.inertia_)
        silhouette_avg = silhouette_score(scaled_df_demographic_proposal,
kmeans.labels_)
        silhouette_scores.append(silhouette_avg)
    if class_ == 'city':
        community_type = 'cities or towns'
    elif class_ == 'county':
        community_type = 'counties'
    elif class_ == 'village':
        community_type = 'villages'
    else:
        community_type = 'all communities'
    plt.figure(figsize=(10, 6))
    plt.plot(range_clusters, sse, marker='o')
    plt.xlabel('Number of clusters')
    plt.ylabel('SSE')
    plt.title(f'Elbow Method for {community_type}')
    plt.show()

return(scaled_df_demographic_proposal,df_demographic_proposal,answer_yes_survey_l
ist,semilla)

def evaluate_clustering(df_demographic,scaled_df_demographic_proposal,
df_demographic_proposal,answer_yes_survey_list, optimal_clusters, semilla):
    kmeans = KMeans(n_clusters=optimal_clusters, random_state=semilla, n_init=10)
    df_demographic_proposal['Cluster'] =
kmeans.fit_predict(scaled_df_demographic_proposal)
    df_demographic_proposal['Answer_yes_survey'] = answer_yes_survey_list
    engaged_communities =
df_demographic_proposal[df_demographic_proposal['Answer_yes_survey'] == 1]
    proposal = []
    for i, a in df_demographic_proposal.iterrows():
        cluster = a['Cluster']
        engaged_communities_cluster =
engaged_communities[engaged_communities['Cluster'] == cluster]
        if not a.empty:
            distances = cdist([a.drop(['Cluster', 'Answer_yes_survey'])],
engaged_communities_cluster.drop(columns=['Cluster', 'Answer_yes_survey']),
metric='euclidean')
            distances = distances.flatten()
            valid_indices = np.where(distances > 0)[0]
            sorted_indices = valid_indices[np.argsort(distances[valid_indices])]
            closest_communities =
engaged_communities_cluster.iloc[sorted_indices].index.tolist()
            recommendation = {'Community': a.name}
            for idx, engaged_community in enumerate(closest_communities):
                recommendation[f'Recommendation{idx + 1}'] = engaged_community
            proposal.append(recommendation)
    df_proposal = pd.DataFrame(proposal)
    columns_recommendation=df_proposal.columns
    for index, row in df_proposal.iterrows():
        for col in columns_recommendation:
            id_value = row[col]
            if id_value in df_demographic.index:
                matching_row = df_demographic.loc[id_value]
                details = f"{matching_row['Muni Name']}, {matching_row['Class']},
{matching_row['County']}"
                df_proposal.at[index, col] = details
    df_proposal = df_proposal.apply(lambda x: x.str.title() if x.dtype ==
"object" else x)
    df_proposal = df_proposal.replace(np.nan, '', regex=True)
    return df_proposal
```