GROUP 2 PROJECT PROGRESS REPORT

1) Title : Prediction of Cost of Living Across U.S. Regions Using a Data Mining Approach

2) Author(s)

1. **Sasank Sribhashyam**
   sasank.sribhashyam-1@ou.edu
2. **Venkat Tarun Adda**
   venkat.tarun.adda-1@ou.edu
3. **Hima Deepika Mannam**
   hima.deepika.mannam-1@ou.edu

3) Objectives and Contributions

Objectives

The primary objective of our project is to analyze and predict the cost of living across various U.S. regions using data mining techniques. Our system will provide a comprehensive set of functionalities that deliver insights into cost patterns, regional economic conditions, and future predictions. The system will support governments, businesses, and individuals in decision-making related to economic planning, relocation, and business strategy.

Functionalities

1. **Cost of Living Clustering:** Identify natural groupings of U.S. regions based on cost-of-living attributes (housing, healthcare, food, etc.) using K-Means Clustering. This helps users discern the cost structure of different regions.

2. **Outlier Detection in Cost of Living:** Highlight regions with significantly high or low living costs compared to economically or geographically similar counterparts to identify regions with unique economic policies or socio-economic conditions.

3. **Time-Series Cost Clustering:** Group regions based on how their cost of living changes over time, allowing users to track and compare regional economic trends and policies.

4. **Socioeconomic Vulnerability Classification:** Classify regions into categories of socioeconomic vulnerability (high, moderate, low) based on various cost of living factors like housing, food, income, and healthcare costs. This classification helps policymakers design targeted interventions.

5. **Targeted Policy Classification:** Provide models to classify regions for policy interventions (e.g., tax benefits or subsidies) based on their socioeconomic vulnerability and cost of living categories.

6. **Future Cost Prediction:** Utilize regression models to forecast future cost of living increases across regions, assisting governments and businesses in inflation adjustments and long-term planning.

7. **Policy Impact Simulation:** Predict how changes in local, state, or federal policies will impact the cost of living, enabling scenario analysis for policymakers.

8. **Association Rule Mining:** Discover patterns and associations between income levels and other living costs (housing, healthcare, etc.), providing insights into how different socioeconomic factors interplay.

9. **Interactive Dashboard:** Develop a user-friendly dashboard to allow stakeholders to explore and visualize cost-of-living patterns, predictions, and region classifications, supporting interactive and data-driven decision-making.

CONTRIBUTIONS:-

Our project's significance lies in its ability to provide a comprehensive and actionable analysis of the cost of living, which is a critical factor for governments, businesses, and individuals. While there are studies and tools that explore cost-of-living patterns, our approach is novel in several ways:

Integration of Multiple Algorithms for Multi-Purpose Analysis: Many existing studies focus on either clustering, classification, or regression independently. Our system integrates multiple algorithms—K-Means, Random Forest, Gradient Boosting, and Apriori—to offer clustering, classification, and predictive insights in a single platform.

Clustering for Time-Based Regional Trends: Traditional cost-of-living tools often present static snapshots of the data. Our project introduces time-series clustering, enabling the detection of dynamic trends over time, making it particularly valuable for analyzing the long-term effects of economic policies and regional developments.

Policy Simulation for Forecasting Impacts: Unlike existing systems, which primarily focus on presenting historical data, our project will allow users to simulate the potential impacts of policy changes. This provides a forward-looking, actionable insight for policymakers, particularly in managing inflation or assessing the effect of tax reforms.

Association Mining for Deeper Socioeconomic Insights: Few studies incorporate association rule mining to explore relationships between socioeconomic factors. Our project leverages the Apriori algorithm to uncover nuanced relationships between income levels and costs (e.g., "regions with high incomes tend to have higher healthcare costs"), enabling a more granular understanding of socioeconomic conditions.

Robust Dashboard for Accessibility: Current literature often lacks accessible visualization tools for non-technical users. Our interactive dashboard will bridge this gap by offering intuitive, data-rich visualizations, allowing a wider range of stakeholders (from policymakers to citizens) to explore and understand cost-of-living data effectively.

4) WORK COMPLETED

a) The functionalities that your application will provide to the users and the graphical user interface (GUI) of your system

a.1) GUI Diagram

The GUI consists of the following major sections:

Navigation Bar: Allows users to switch between the primary functionalities, including "Cost Clustering," "Vulnerability Classification," "Prediction Models," and "Policy Simulation."

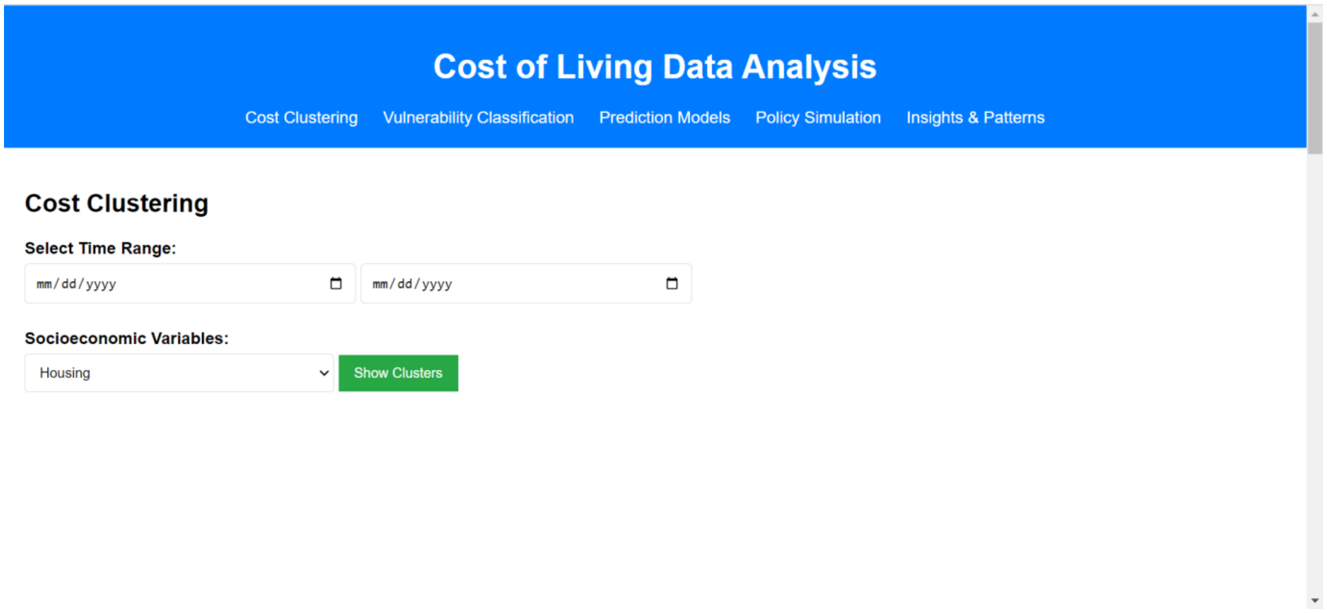
Clustering Dashboard: Displays clusters of regions based on cost-of-living attributes. Users can select time ranges and specific socioeconomic variables to visualize how different regions group together.

Classification Dashboard: Shows classifications of regions into categories of vulnerability. The user can explore factors such as healthcare, housing, and transportation costs.

Prediction Dashboard: A section that visualizes the results of regression models, allowing users to see forecasts for future cost-of-living trends across selected regions.

Policy Impact Simulation: Allows users to simulate the effects of potential policy changes on the cost of living. Users input tax or subsidy adjustments, and the system predicts the resulting cost changes.

Insights Section: Provides users with associations and discovered patterns between socioeconomic variables and living costs. The user can input specific parameters to generate association rules.



a.2) Functionality Status

Cost of Living Clustering:

Description: Users can view clusters of regions based on the cost of living. Filters allow selection by time period and specific cost factors.

Status: *In Progress*. The K-Means Clustering algorithm is implemented, and initial visualizations are being designed for the dashboard.

Outlier Detection in Cost of Living:

Description: Detect and highlight regions with extreme costs (either too high or too low) compared to others.

Status: *Not Started*. This feature will be implemented once the clustering model is fully functional.

Time-Series Cost Clustering:

Description: Group regions based on how their cost of living has changed over a defined period.

Status: *Not Started*. Time-series data preprocessing is underway.

Socioeconomic Vulnerability Classification:

Description: Classify regions into high, moderate, or low vulnerability categories based on cost-of-living data.

Status: *In Progress*. The Random Forest Classifier is being trained on the dataset, and early testing shows promising accuracy.

Targeted Policy Classification:

Description: Identify regions for targeted policy interventions based on cost and vulnerability levels.

Status: *Not Started*. The initial classification model will be extended to support this feature.

Future Cost Prediction:

Description: Predict future changes in living costs across regions using a regression model.

Status: *In Progress*. A Gradient Boosting Regression model has been developed, and testing is currently being conducted to assess its accuracy.

Policy Impact Simulation:

Description: Allows users to input policy changes (e.g., tax, subsidies) and view how these changes affect the cost of living.

Status: *Not Started*. This will be implemented once prediction models are optimized.

Association Rule Mining:

Description: Uncover patterns and relationships between socioeconomic factors and living expenses across regions.

Status: *Not Started*. Data preprocessing for the Apriori algorithm is underway.

b) The Datasets You Use

The dataset we are using remains unchanged from the one described in our project proposal: the **"cost_of_living_us.csv"**dataset sourced from Kaggle, containing 31,430 records and 15 attributes. The dataset covers various cost-of-living factors, including housing, food, transportation, healthcare, and taxes, across U.S. regions.

We have decided to continue using this dataset without any modifications for the following reasons:

1. **Data Coverage:** The dataset provides comprehensive coverage of cost-of-living variables across U.S. regions, which aligns well with our project's goals of clustering, classification, and prediction of cost variations.

2. **Attribute Relevance:** The attributes in the dataset (e.g., housing cost, transportation cost, family size, healthcare cost) are directly relevant to the data mining tasks (clustering, classification, regression) and provide sufficient granularity for our analysis.
3. **Dataset Size and Structure:** The dataset contains a manageable number of records (31,430) and attributes (15), making it suitable for our computational resources. Its structure supports the requirements for implementing machine learning algorithms like K-Means Clustering, Random Forest Classification, and Gradient Boosting Regression.
4. **Consistency with Project Objectives:** The dataset's attributes are closely tied to our objectives of predicting the cost of living, classifying regions by socioeconomic vulnerability, and identifying patterns in living expenses. Therefore, no changes were necessary as the dataset already meets the data quality and completeness requirements for our tasks.

c) The Data Mining Tasks and Their Purposes

Our project incorporates several key data mining tasks to achieve the goals of predicting, analyzing, and classifying cost-of-living patterns across U.S. regions.

1. Clustering: K-Means Clustering

Purpose: The purpose of clustering in our project is to identify natural groupings of U.S. regions based on their cost-of-living characteristics. This task helps segment regions with similar living expenses, allowing users to explore which areas share similar economic conditions and which differ drastically. The clusters also help highlight trends in geographic and economic similarities.

Implementation Details:

1. **Algorithm:** K-Means Clustering
2. **Input Features:** Housing cost, food cost, healthcare cost, transportation cost, etc.
3. **Output:** Clusters of U.S. regions grouped into categories such as "low-cost," "moderate-cost," and "high-cost" regions.
4. **Why K-Means:** K-Means is chosen for its simplicity and scalability, especially with our dataset's size (31,430 records). It effectively handles multiple continuous features, providing interpretable cluster assignments that help users understand regional cost variations.

Application: The clusters will be displayed in the dashboard, enabling users to select specific regions and understand how their costs compare with others in the same or different clusters. Policymakers can use this insight to target regions for economic interventions, while businesses may use it to strategize location-based operations or marketing.

2. Classification: Random Forest Classifier

Purpose: Classification is used to categorize U.S. regions based on socioeconomic vulnerability. This task provides an assessment of which areas are most vulnerable to economic hardships based on their cost-of-living data. It also helps users identify regions that may require targeted policy interventions such as subsidies or tax reliefs.

Implementation Details:

1. **Algorithm:** Random Forest Classifier
2. **Input Features:** Housing cost, healthcare cost, food cost, income levels, family size, etc.
3. **Output:** Classifications of regions into categories such as "high vulnerability," "moderate vulnerability," and "low vulnerability."
4. **Why Random Forest:** Random Forest is a powerful ensemble method that handles high-dimensional data well, providing robustness against overfitting. It's ideal for handling mixed types of features (continuous and categorical) and can deal with missing or imbalanced data, which is present in our dataset.

Application: This classification will help policymakers and governmental agencies prioritize regions for economic assistance. Users of the dashboard will be able to explore which regions are classified as highly vulnerable and the key factors contributing to their vulnerability.

3. Regression: Gradient Boosting Regression (GBR)

Purpose: The regression task aims to predict future cost-of-living increases across U.S. regions. This is critical for planning, as governments, businesses, and individuals need to anticipate economic changes that could impact budgets, pricing, and relocation decisions.

Implementation Details:

Algorithm: Gradient Boosting Regression (GBR)

Input Features: Historical cost data (e.g., housing, food, healthcare, taxes), economic indicators, and income levels.

Output: Predicted cost of living (total cost) for each region.

Why GBR: GBR is chosen for its ability to handle complex, non-linear relationships in the data. It builds a series of weak learners (decision trees) and refines predictions iteratively, providing better predictive accuracy than simpler regression methods. This is crucial for producing reliable cost-of-living forecasts.

Application: Users can forecast how living costs will evolve in different regions and adjust their economic strategies accordingly. For example, businesses may use these predictions to plan price adjustments, while individuals can evaluate the financial implications of moving to a particular region.

4. Association Rule Mining: Apriori Algorithm

Purpose: The association rule mining task is designed to uncover patterns and relationships between different cost-of-living factors, such as how income levels correlate with housing or healthcare costs. This provides deeper insights into how various socioeconomic variables interact across U.S. regions.

Implementation Details:

Algorithm: Apriori Algorithm

Input Features: Income levels, housing costs, healthcare costs, family size, transportation costs, etc.

Output: Association rules such as "Regions with high income levels tend to have high housing costs," with associated support, confidence, and lift metrics.

Why Apriori: Apriori is well-suited for mining frequent itemsets and generating association rules, which makes it ideal for identifying correlations in transactional or multidimensional datasets. It is computationally efficient and interpretable, which allows us to present actionable insights.

Application: The discovered associations will help policymakers and researchers understand the underlying factors that drive living costs in different regions. Businesses can use these patterns to tailor their strategies based on income levels and associated living costs in particular areas.

5. Anomaly Detection: Outlier Detection in Cost of Living

Purpose: Anomaly detection focuses on identifying regions with significantly higher or lower living costs compared to similar regions. These outliers can highlight regions with unique economic or social conditions and can be a target for further investigation.

Implementation Details:

Algorithm: Z-Score Method or Isolation Forest

Input Features: Cost variables (housing, healthcare, food, etc.).

Output: Regions that are statistical outliers based on cost comparisons.

Why Anomaly Detection: Regions with unusual cost structures could either indicate economic resilience or stress. Anomaly detection helps us flag these regions for further analysis, which is important for targeting economic policies or investigating root causes.

Application: This functionality allows users to explore regions that are outliers in terms of cost structure. Outliers may represent regions with unique policies, rapid economic changes, or anomalous socioeconomic conditions, providing key insights for government and economic researchers.

d) The Data Mining Algorithms and How They Work

1. Clustering: K-Means Clustering

Algorithm Overview: K-Means Clustering is an iterative, unsupervised learning algorithm used to partition a dataset into *K* distinct clusters based on feature similarity.

How It Solves Our Task: K-Means Clustering is effective for grouping U.S. regions based on their cost-of-living attributes. The algorithm's ability to handle multiple continuous variables (e.g., housing costs, healthcare costs) allows it to create clusters that reflect real-world economic patterns. By partitioning regions into "low-cost," "moderate-cost," and "high-cost" clusters, we provide users with an intuitive view of regional differences in living expenses.

Why K-Means Was Chosen:

Scalability: K-Means is computationally efficient and can handle large datasets, making it ideal for our dataset with over 31,000 records.

Interpretability: The algorithm produces clear and distinct clusters, which can be easily visualized on the dashboard, making it user-friendly for policymakers and businesses.

2. Classification: Random Forest Classifier

Algorithm Overview: Random Forest is an ensemble learning method that constructs multiple decision trees during training and aggregates their outputs to improve accuracy and reduce overfitting.

How It Solves Our Task: Random Forest is well-suited for classifying U.S. regions into categories of socioeconomic vulnerability (e.g., high, moderate, or low vulnerability). By considering multiple decision trees, the algorithm captures complex interactions between cost-of-living factors (housing, healthcare, income, etc.), providing accurate and robust classifications.

Why Random Forest Was Chosen:

High Accuracy: Random Forest is known for producing highly accurate results because it reduces overfitting through the ensemble of trees.

Feature Importance: The algorithm provides feature importance scores, which help in understanding which factors most strongly influence the classification of vulnerability.

3. Regression: Gradient Boosting Regression (GBR)

Algorithm Overview: Gradient Boosting Regression (GBR) is a machine learning technique that builds models sequentially. Each new model focuses on correcting the errors made by the previous model.

How It Solves Our Task: GBR is used to predict future cost-of-living increases across U.S. regions. Given the complex, non-linear relationships between cost variables (housing, healthcare, taxes, etc.) and the total cost of living, GBR is well-suited for this task. It builds multiple decision trees in a sequential manner, progressively reducing prediction errors.

Why GBR Was Chosen:

High Predictive Power: GBR is known for its ability to handle non-linear relationships and provide accurate predictions, making it ideal for forecasting cost of living based on diverse economic factors.

Flexibility: The algorithm can be fine-tuned by adjusting the number of trees, learning rate, and tree depth to optimize performance for our dataset.

4. Association Rule Mining: Apriori Algorithm

Algorithm Overview: The Apriori Algorithm is used for mining frequent itemsets and discovering association rules in a dataset.

How It Solves Our Task: Apriori is used to uncover relationships between different cost-of-living factors (e.g., "regions with high income levels tend to have high housing costs"). These patterns provide valuable insights into how socioeconomic conditions are interlinked across regions.

Why Apriori Was Chosen:

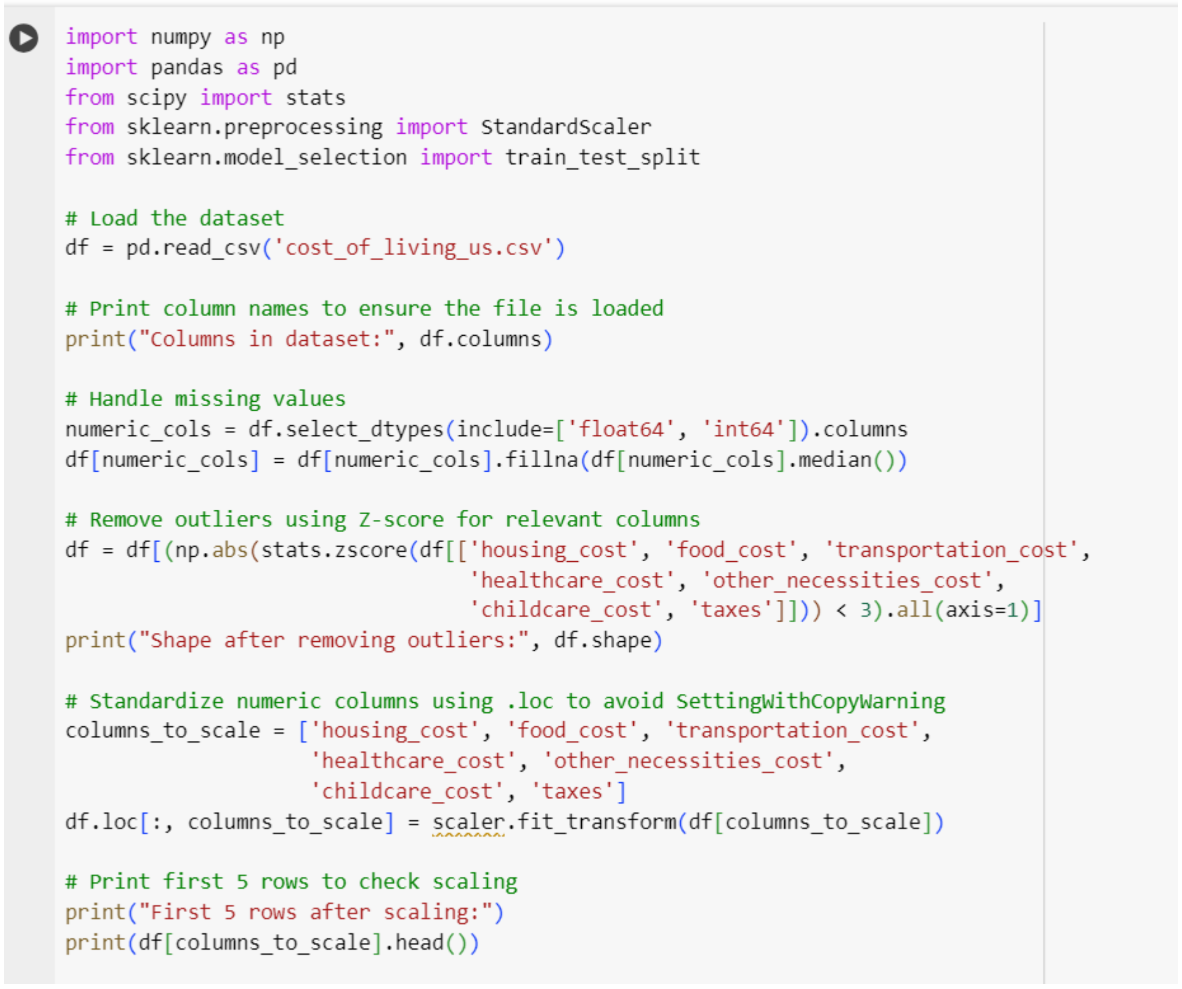
Interpretability: The rules generated by Apriori are easy to understand, making it a good fit for exploring relationships between cost variables in our dataset.

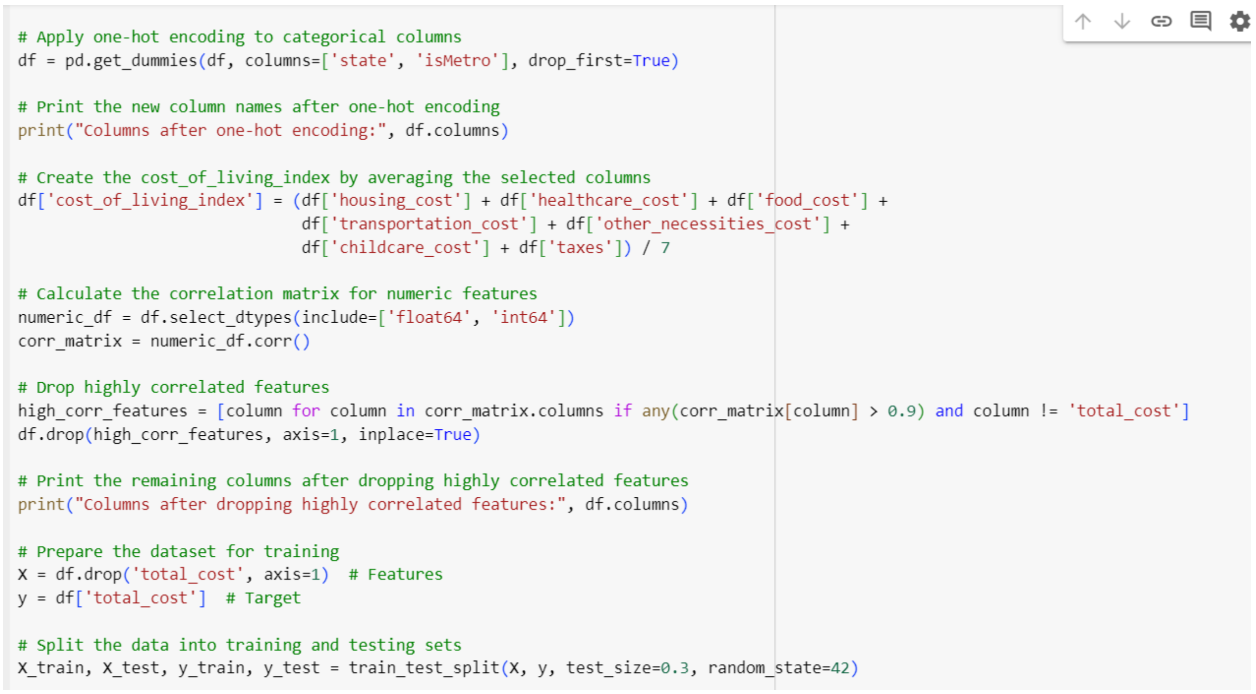
Efficiency for Large Datasets: Apriori's pruning mechanism ensures that it can efficiently handle large datasets like ours by only focusing on itemsets that meet the minimum support threshold.

e) Data Preprocessing Activities

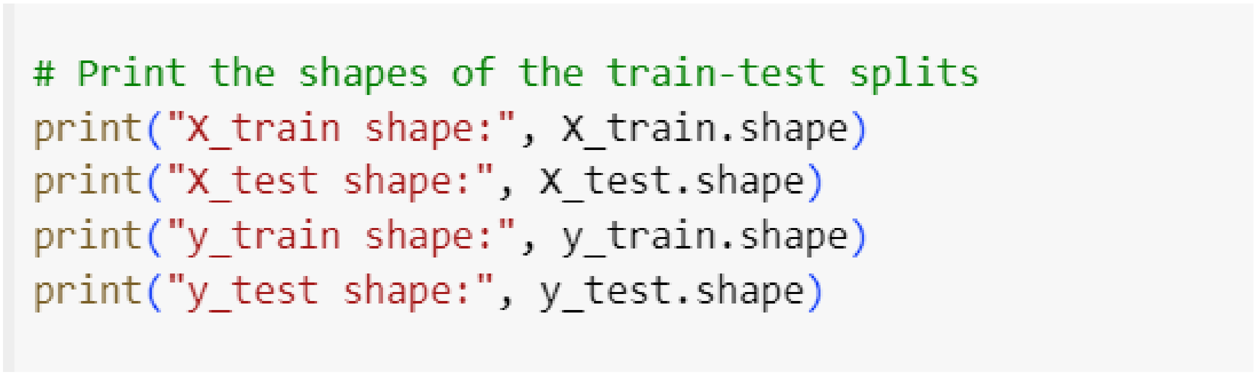
For the U.S. cost of living dataset, several preprocessing steps were applied to prepare the data for modeling. These steps include:

1. **Handling Missing Values**: Numeric columns with missing values were filled using the median to prevent gaps that could affect model training.
2. **Removing Outliers**: Outliers in key cost columns like housing_cost, food_cost, and taxes were removed using Z-scores, ensuring extreme values don't distort the analysis.
3. **Standardizing Numeric Features**: The cost columns were standardized to give them equal weight in the model, which is important for algorithms sensitive to feature scales.
4. **One-Hot Encoding Categorical Features**: Categorical columns, such as state and isMetro, were one-hot encoded to convert them into numerical form for machine learning algorithms.
5. **Creating a Cost of Living Index**: A custom cost_of_living_index was created by averaging costs like housing, healthcare, and taxes to provide a single metric summarizing the overall cost of living.
6. **Removing Highly Correlated Features**: Features with correlations above 0.9 were removed to avoid redundancy and multicollinearity.
7. **Splitting the Data**: Finally, the dataset was split into training and testing sets (70-30 split) to ensure reliable model evaluation on unseen data.


```python
import numpy as np
import pandas as pd
from scipy import stats
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split

# Load the dataset
df = pd.read_csv('cost_of_living_us.csv')

# Print column names to ensure the file is loaded
print("Columns in dataset:", df.columns)

# Handle missing values
numeric_cols = df.select_dtypes(include=['float64', 'int64']).columns
df[numeric_cols] = df[numeric_cols].fillna(df[numeric_cols].median())

# Remove outliers using Z-score for relevant columns
df = df[(np.abs(stats.zscore(df[['housing_cost', 'food_cost', 'transportation_cost',
                                 'healthcare_cost', 'other_necessities_cost',
                                 'childcare_cost', 'taxes']])) < 3).all(axis=1)]
print("Shape after removing outliers:", df.shape)

# Standardize numeric columns using .loc to avoid SettingWithCopyWarning
columns_to_scale = ['housing_cost', 'food_cost', 'transportation_cost',
                    'healthcare_cost', 'other_necessities_cost',
                    'childcare_cost', 'taxes']
df.loc[:, columns_to_scale] = scaler.fit_transform(df[columns_to_scale])

# Print first 5 rows to check scaling
print("First 5 rows after scaling:")
print(df[columns_to_scale].head())
```

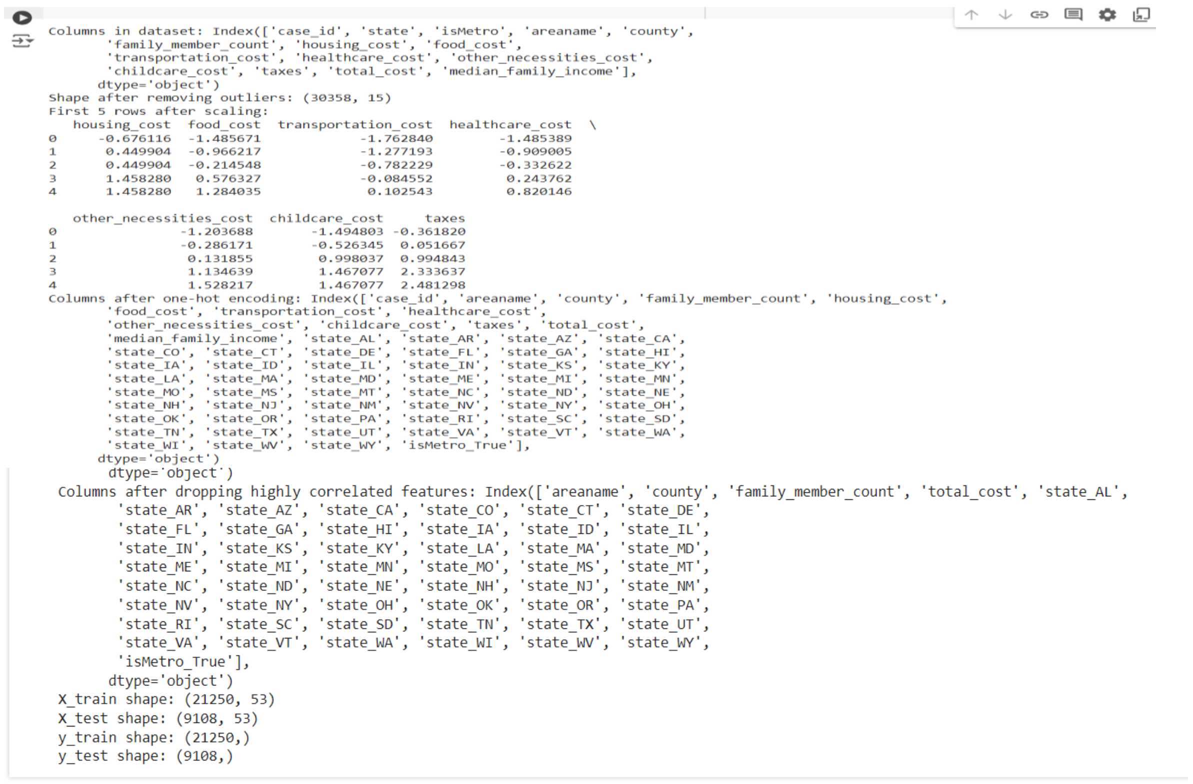


```python
# Apply one-hot encoding to categorical columns
df = pd.get_dummies(df, columns=['state', 'isMetro'], drop_first=True)

# Print the new column names after one-hot encoding
print("Columns after one-hot encoding:", df.columns)

# Create the cost_of_living_index by averaging the selected columns
df['cost_of_living_index'] = (df['housing_cost'] + df['healthcare_cost'] + df['food_cost'] +
                              df['transportation_cost'] + df['other_necessities_cost'] +
                              df['childcare_cost'] + df['taxes']) / 7

# Calculate the correlation matrix for numeric features
numeric_df = df.select_dtypes(include=['float64', 'int64'])
corr_matrix = numeric_df.corr()

# Drop highly correlated features
high_corr_features = [column for column in corr_matrix.columns if any(corr_matrix[column] > 0.9) and column != 'total_cost']
df.drop(high_corr_features, axis=1, inplace=True)

# Print the remaining columns after dropping highly correlated features
print("Columns after dropping highly correlated features:", df.columns)

# Prepare the dataset for training
X = df.drop('total_cost', axis=1)  # Features
y = df['total_cost']  # Target

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```



```python
# Print the shapes of the train-test splits
print("X_train shape:", X_train.shape)
print("X_test shape:", X_test.shape)
print("y_train shape:", y_train.shape)
print("y_test shape:", y_test.shape)
```

Output



```
Columns in dataset: Index(['case_id', 'state', 'isMetro', 'areaname', 'county',
       'family_member_count', 'housing_cost', 'food_cost',
       'transportation_cost', 'healthcare_cost', 'other_necessities_cost',
       'childcare_cost', 'taxes', 'total_cost', 'median_family_income'],
      dtype='object')
Shape after removing outliers: (30358, 15)
First 5 rows after scaling:
   housing_cost  food_cost  transportation_cost  healthcare_cost  \
0     -0.676116  -1.485671            -1.762840        -1.485389
1      0.449904  -0.966217            -1.277193        -0.909005
2      0.449904  -0.214548            -0.782229        -0.332622
3      1.458280   0.576327            -0.084552         0.243762
4      1.458280   1.284035             0.102543         0.820146

   other_necessities_cost  childcare_cost     taxes
0               -1.203688       -1.494803 -0.361820
1               -0.286171       -0.526345  0.051667
2                0.131855        0.998037  0.994843
3                1.134639        1.467077  2.333637
4                1.528217        1.467077  2.481298
Columns after one-hot encoding: Index(['case_id', 'areaname', 'county', 'family_member_count', 'housing_cost',
       'food_cost', 'transportation_cost', 'healthcare_cost',
       'other_necessities_cost', 'childcare_cost', 'taxes', 'total_cost',
       'median_family_income', 'state_AL', 'state_AR', 'state_AZ', 'state_CA',
       'state_CO', 'state_CT', 'state_DE', 'state_FL', 'state_GA', 'state_HI',
       'state_IA', 'state_ID', 'state_IL', 'state_IN', 'state_KS', 'state_KY',
       'state_LA', 'state_MA', 'state_MD', 'state_ME', 'state_MI', 'state_MN',
       'state_MO', 'state_MS', 'state_MT', 'state_NC', 'state_ND', 'state_NE',
       'state_NH', 'state_NJ', 'state_NM', 'state_NV', 'state_NY', 'state_OH',
       'state_OK', 'state_OR', 'state_PA', 'state_RI', 'state_SC', 'state_SD',
       'state_TN', 'state_TX', 'state_UT', 'state_VA', 'state_VT', 'state_WA',
       'state_WI', 'state_WV', 'state_WY', 'isMetro_True'],
      dtype='object')
       dtype='object')
 Columns after dropping highly correlated features: Index(['areaname', 'county', 'family_member_count', 'total_cost', 'state_AL',
        'state_AR', 'state_AZ', 'state_CA', 'state_CO', 'state_CT', 'state_DE',
        'state_FL', 'state_GA', 'state_HI', 'state_IA', 'state_ID', 'state_IL',
        'state_IN', 'state_KS', 'state_KY', 'state_LA', 'state_MA', 'state_MD',
        'state_ME', 'state_MI', 'state_MN', 'state_MO', 'state_MS', 'state_MT',
        'state_NC', 'state_ND', 'state_NE', 'state_NH', 'state_NJ', 'state_NM',
        'state_NV', 'state_NY', 'state_OH', 'state_OK', 'state_OR', 'state_PA',
        'state_RI', 'state_SC', 'state_SD', 'state_TN', 'state_TX', 'state_UT',
        'state_VA', 'state_VT', 'state_WA', 'state_WI', 'state_WV', 'state_WY',
        'isMetro_True'],
       dtype='object')
 X_train shape: (21250, 53)
 X_test shape: (9108, 53)
 y_train shape: (21250,)
 y_test shape: (9108,)
```

f) Implementation Details of the Application

Programming Languages and Software Systems

Programming Languages:

Python:

Python is the primary programming language used for implementing the data mining algorithms, data preprocessing, and building the graphical user interface (GUI). Python is chosen due to its rich ecosystem of libraries for machine learning, data analysis, and visualization, making it ideal for our application.

Why Python?

1. Extensive libraries for data mining tasks (e.g., scikit-learn, pandas, NumPy)
2. Easy-to-read syntax, which simplifies collaboration across team member
3. Excellent support for open-source tools and integration with machine learning frameworks

Software Libraries and Tools:

Scikit-learn:

This open-source library is used for implementing machine learning algorithms such as K-Means Clustering, Random Forest, Gradient Boosting, and Apriori. It provides high-level implementations of these algorithms with efficient optimization.

Functionalities:

1. Algorithm implementations (clustering, classification, regression)
2. Cross-validation and performance metrics (accuracy, RMSE, etc.)
3. Preprocessing utilities (scaling, imputation, encoding)

Source:
Scikit-learn is an open-source library available at https://scikit-learn.org.

pandas and NumPy: These libraries are used for data manipulation and numerical computation, handling tasks such as data cleaning, normalization, and feature engineering.

Functionalities:

1. Dataframes for structured data manipulation (pandas)
2. Array-based computations and mathematical operations (NumPy)

Source:
Both pandas and NumPy are open-source and available via https://pandas.pydata.org and https://numpy.org.

Plotly Dash: Plotly Dash is used for building the interactive dashboard that will display clustering, classification, and regression results. Dash is a Python framework for building web applications with a strong focus on data visualization.

Functionalities:

1. Interactive and real-time visualizations (e.g., scatter plots, heatmaps)
2. GUI development with user input components (dropdowns, sliders)
3. Supports seamless integration of machine learning models for visualization

Source:
Dash is an open-source library available at https://plotly.com/dash/.

Matplotlib and Seaborn: These visualization libraries are used for creating static data visualizations for exploratory data analysis (EDA) and generating plots like histograms, box plots, and correlation heatmaps.

Functionalities:

1. Creation of 2D plots for data distribution analysis
2. Heatmaps for feature correlation visualization

Source:
Both are open-source and can be installed via Python package managers like pip (https://matplotlib.org and https://seaborn.pydata.org).

2. Open-Source Availability

All the programming languages and software libraries used in this project are open-source, which allows for cost-effective development and community support. This also ensures that the project can be easily shared with others, adhering to the principles of transparency and reproducibility.

3. Graphical User Interface (GUI) and Visualization Methods

The GUI will be built using **Plotly Dash**, which offers an interactive web-based interface that users can access to interact with the data mining results and visualize cost-of-living patterns across U.S. regions. The design of the GUI will focus on user accessibility, allowing policymakers, businesses, and individuals to explore data insights in a clear and interactive manner.

Main Visualization Methods:

1. **Clustering Visualization:**
   a. **Visualization Type:** Scatter plots or bubble charts to display U.S. regions grouped into clusters based on their cost of living. Users can filter by different attributes (e.g., housing cost, transportation cost).

   b. **Interactive Features:** Users can zoom in on specific regions or time periods to analyze cluster characteristics.

2. **Classification Visualization:**
   a. **Visualization Type:** Heatmaps and bar charts that show the classification of regions into vulnerability categories (high, moderate, low).

   b. **Interactive Features:** Users can hover over specific regions to see details such as healthcare and housing costs that contributed to their classification.

3. **Regression Predictions Visualization:**
    a. **Visualization Type:** Line graphs and area charts that display the predicted future cost of living in selected regions.

    b. **Interactive Features:** Users can select different timeframes to see how the prediction model forecasts cost increases over time.
4. **Policy Impact Simulation:**
    a. **Visualization Type:** Interactive sliders and input boxes where users can modify policy variables (e.g., taxes, subsidies) and see real-time simulated changes in cost of living.

    b. **Interactive Features:** Users can input custom policy changes and immediately visualize the potential impact across regions.

User Input and Control:

Dropdown menus: Users can select regions, cost factors (housing, healthcare, etc.), and time periods to customize the visualizations according to their needs.

Sliders and Text Boxes: For the policy impact simulation feature, users can enter values or adjust sliders to change tax rates or subsidies, which updates the cost predictions in real-time.

Additional Tools

Jupyter Notebooks: Jupyter Notebooks are used during the initial development phases for exploratory data analysis (EDA), algorithm testing, and visualizing intermediate results. The notebooks allow easy debugging, testing, and collaboration among team members.

Source: Jupyter Notebooks are open-source and available at https://jupyter.org.

g) Performance Evaluation Plan

The performance evaluation of our application is critical to ensure that the data mining algorithms implemented (clustering, classification, regression, and association rule mining) provide accurate, meaningful, and actionable insights. Each task requires different metrics based on its objective. Below is a detailed description of the evaluation metrics, why they are chosen, and the methods we will use to assess the performance of the algorithms.

1. Clustering: K-Means Clustering

Performance Metrics:

Silhouette Score: The Silhouette Score evaluates how similar a point is to its own cluster (cohesion) compared to other clusters (separation). It ranges from -1 to +1, where a higher score indicates better-defined clusters.

Why Chosen: This metric provides an intuitive understanding of the quality of the clusters formed, as it considers both intra-cluster cohesion and inter-cluster separation.

Elbow Method (Within-Cluster Sum of Squares - WCSS): The elbow method plots WCSS against the number of clusters, and the "elbow point" indicates the optimal number of clusters. The WCSS measures the sum of squared distances between data points and their corresponding cluster centroids.

Why Chosen: It helps in determining the optimal number of clusters, which is crucial for ensuring that the clustering task is neither under- nor over-clustered.

Method of Evaluation:

Internal Validation: After clustering, the Silhouette Score and WCSS will be computed to assess the clustering quality. Different values of K (number of clusters) will be tested, and the elbow point will be identified using the Elbow Method. Additionally, the silhouette score for different K values will help confirm the optimal number of clusters.

2. Classification: Random Forest Classifier

Performance Metrics:

Accuracy:
The proportion of correctly classified regions out of the total number of regions.
Why Chosen:
Accuracy is a straightforward metric to evaluate how well the classification model performs on the entire dataset.

Precision, Recall, and F1-Score:

Precision: The proportion of true positive classifications out of all predicted positives.

Recall (Sensitivity): The proportion of true positives out of all actual positives.

F1-Score: The harmonic mean of precision and recall, providing a balanced metric when both false positives and false negatives are of concern.

Why Chosen:
These metrics are essential for imbalanced datasets where one class (e.g., high vulnerability) may have significantly fewer examples than others. The F1-score is particularly useful in such cases because it balances precision and recall.

Confusion Matrix:
A confusion matrix will show the breakdown of correct and incorrect classifications for each category (high, moderate, low vulnerability).
Why Chosen:
This matrix helps in understanding not just overall accuracy but also which categories are misclassified more often, allowing for targeted model improvements.

Method of Evaluation:

Cross-Validation:
We will apply k-fold cross-validation (with k=5 or 10) to assess the model's performance on different subsets of the data. This method ensures that the evaluation is not biased by a single train-test split.

Confusion Matrix Analysis:
We will analyze the confusion matrix for insights into which categories of socioeconomic vulnerability are misclassified. Precision, recall, and the F1-score will be calculated for each category to ensure balanced performance across all classes.

3. Regression: Gradient Boosting Regression (GBR)

Performance Metrics:

Root Mean Squared Error (RMSE):
RMSE is a widely used metric that measures the average magnitude of prediction errors. It gives higher weight to large errors, making it sensitive to significant deviations between predicted and actual values.
Why Chosen:
RMSE is useful for understanding the overall performance of the regression model and how far the predictions are from the true cost-of-living values. This is important in our application, where large prediction errors (e.g., incorrect future cost-of-living projections) could have significant consequences for users.

Mean Absolute Error (MAE):
MAE measures the average of the absolute differences between predicted and actual values, providing a more interpretable metric.
Why Chosen:
Unlike RMSE, MAE treats all errors equally and is more robust to outliers, offering another perspective on prediction accuracy.

R-squared (R^2):
R^2 measures the proportion of variance in the dependent variable (cost of living) that is explained by the independent variables.
Why Chosen:
This metric provides an indication of how well the model explains the variability in the cost of living across regions.

Method of Evaluation:

Train-Test Split:
The model will be trained on 80% of the dataset and tested on the remaining 20%. RMSE, MAE, and R^2 will be calculated on the test set to evaluate the performance of the model.

Cross-Validation:
We will use k-fold cross-validation to further assess the generalizability of the regression model and ensure that it performs well across different subsets of the data.

4. Association Rule Mining: Apriori Algorithm

1.Performance Metrics:

Support:
The frequency with which an itemset (e.g., "high income" and "high housing cost") appears in the dataset.
Why Chosen:
Support indicates how often certain patterns occur, helping us understand the commonality of relationships between socioeconomic factors.

Confidence:
The proportion of transactions that contain an itemset (e.g., "high income") where a specific association (e.g., "high housing cost") also occurs.
Why Chosen:
Confidence measures the strength of a rule, indicating how likely it is that the second part of the rule occurs when the first part does. This is particularly useful for uncovering reliable associations between income and cost-of-living factors.

Lift:
Lift measures the strength of an association rule compared to random chance. A lift greater than 1 indicates a positive association between the items in the rule.
Why Chosen:
Lift helps identify whether the association is meaningful beyond random co-occurrence, which is important for finding actionable patterns in the data.

2.Method of Evaluation:

Rule Validation:
We will evaluate the strength of the association rules using support, confidence, and lift. Rules with high confidence and lift values will be considered strong and meaningful.

Pruning:
We will prune rules with low support and confidence to avoid generating irrelevant or weak associations. The lift metric will help focus on significant patterns that are more likely to provide actionable insights.

5) Work to Be Done

The following tasks are critical to completing the project. They include algorithm optimization, dashboard development, final model evaluation, and report preparation. The table below outlines the tasks, deliverables, timelines, status, and responsible group members.

TASK	DELIVERABLES	START DATE	END DATE	STATUS COMPARED TO ORIGINAL SCHEDULE	RESPONSIBLE GROUP MEMBER
Clustering model finalization	Optimized K-Means clustering model, evaluation report (Silhouette Score, WCSS)	October 17, 2024	October 20, 2024	On Schedule	Sasank Sribhashyam
Classification Model Finalization	Final Random Forest classification model, precision-recall metrics, and confusion matrix	October 17, 2024	October 22, 2024	On Schedule	Venkat Tarun Adda
Regression Model Optimization	Optimized Gradient Boosting Regression model, RMSE, MAE, and R^2 evaluation report	October 20, 2024	October 24, 2024	On Schedule	Sasank Sribhashyam
Association Rule Mining Implementation	Apriori algorithm, association rules with support, confidence, and lift	October 22, 2024	October 25, 2024	On Schedule	Venkat Tarun Adda
Anomaly Detection Implementation	Isolation Forest model, anomaly score, and outlier analysis report	October 21, 2024	October 27, 2024	On Schedule	Hima Deepika Mannam
Dashboard Development	Fully functional interactive dashboard for clustering, classification, regression, and simulation	October 20, 2024	November 5, 2024	On Schedule	Hima Deepika Mannam
Model Performance Evaluation	Final evaluation report of all models using cross-validation and test metrics	October 25, 2024	November 5, 2024	On Schedule	Sasank Sribhashyam

Final Report Writing	Comprehensive report covering objectives, methodology, results, and conclusions	November 6, 2024	November 20, 2024	On Schedule	All Group Members
Presentation Preparation	PowerPoint slides and project demonstration video	November 21, 2024	December 1, 2024	On Schedule	All Group Members

Explanation of Remaining Tasks

1. Clustering Model Finalization

Deliverables: Final optimized K-Means clustering model with evaluation using Silhouette Score and WCSS.

Details: This involves tuning the number of clusters (K) and ensuring that the clustering results provide meaningful segmentation of U.S. regions. The evaluation will ensure the optimal number of clusters is selected based on internal validation metrics.

2. Classification Model Finalization

Deliverables: Final Random Forest classification model with performance metrics (accuracy, precision, recall, F1-score) and a confusion matrix.

Details: The focus here will be on optimizing the Random Forest classifier to ensure high accuracy in categorizing regions by socioeconomic vulnerability. Cross-validation and a confusion matrix will be used to assess the model's performance.

3. Regression Model Optimization

Deliverables: Optimized Gradient Boosting Regression model with RMSE, MAE, and R^2 values.

Details: The regression model will be fine-tuned to improve its predictive performance on cost-of-living increases. The evaluation will ensure that the model performs well on unseen data.

4. Association Rule Mining Implementation

Deliverables: Implemented Apriori algorithm, with discovered association rules between cost variables, including support, confidence, and lift.

Details: This task involves mining patterns and relationships between cost-of-living factors, providing users with actionable insights on how different socioeconomic variables relate across regions.

5. Anomaly Detection Implementation

Deliverables: Isolation Forest model with anomaly scores and analysis of outliers in the dataset.

Details: The model will identify regions with anomalous cost patterns, which may be of particular interest for further analysis or policy intervention.

6. Dashboard Development

Deliverables: A fully functional interactive dashboard that integrates clustering, classification, regression, and policy impact simulation features.

Details: The dashboard will allow users to visualize results from the various models and interact with the data (e.g., filtering by regions, adjusting time ranges, and simulating policy changes).

7. Model Performance Evaluation

Deliverables: Final evaluation report comparing the performance of all models (clustering, classification, regression, and anomaly detection).

Details: This involves a detailed analysis of model performance using test metrics like accuracy, RMSE, and silhouette scores, ensuring that the best-performing models are selected for the final system.

8. Final Report Writing

Deliverables: A comprehensive final report, including objectives, methodology, results, and conclusions.

Details: This report will summarize all aspects of the project, from the initial problem statement to the final evaluation of the models and the significance of the findings.

9. Presentation Preparation

Deliverables: PowerPoint slides and a video demonstration of the project.

Details: This task involves preparing a clear and engaging presentation that demonstrates the project's functionality, findings, and potential applications. A video demo will showcase how the dashboard and models work in real time.

6) References

Witten, I. H., Frank, E., & Hall, M. A. (2011). *Data Mining:* Practical Machine Learning Tools and Technique*s* (3rd ed.). Morgan Kaufmann.

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics, 20, 53-65. DOI: 10.1016/0377-0427(87)90125-7.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. Annals of Statistics, 29(5), 1189-1232. DOI: 10.1214/aos/1013203451.

Agrawal, R., & Srikant, R. (1994). Fast Algorithms for Mining Association Rules. Proceedings of the 20th International Conference on Very Large Data Bases, 487-499.

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90-95. DOI: 10.1109/MCSE.2007.55.

“Pang-Ning Tan, Michael Steinbach, Anuj Karpatne, and Vipin Kumar : Introduction to Data Mining, 2ndEdition, Pearson Education, Inc., (2019)”

“Georgius Andrian Halima, Patrice Agustina, Elbert Adiwijayantoa, Margaretha Ohyver, Estimation of cost of living in a particular city using multiple regression analysis and correction of residual assumptions through appropriate methods, 7th International Conference on Computer Science and Computational Intelligence 2022 (2022)”

“P.Ketha Vardhan Reddy, K.Hari Haran, P.Sahith Chowdary, Dr.G.Victo Sudha George. GLOBAL COST OF LIVING USING DATA SCIENCE, 2022 IJCSPUB | Volume 12, Issue 1 March 2022 | ISSN: 2250-1770 (2022)"

“John Friesen 1 ID, Lea Rausch 1 1 ID , Peter F. Pelz 1,* and Johannes Fürnkranz, Determining Factors for Slum Growth with Predictive Data Mining Methods, Urban Sci. (2018), 2, 81; doi:10.3390/urbansci203008”

“Prediction of the Cost of Living Index Using Machine Learning Techniques, Domingos, P. (2012). [www.medium.com] Date Accessed: 09/12/2024.”

"A few useful things to know about machine learning," Communications of the ACM, vol.

55, pp. 78-87."

"Dataset: https://www.kaggle.com/datasets/asaniczka/us-cost-of-living-dataset-3171-counties"