

Project Report: Understanding Audience - A Machine Learning Approach to Customer Segmentation

1 Introduction

1.1 Project Overview

In this rapidly evolving and competitive business landscape, the identification and segmentation of potential customers play a pivotal role in an organization's success. The project, "Understanding Audience," leverages the power of Machine Learning, with understanding and experimenting with different Machine Learning models to classify customers into segments based on various attributes.

1.2 Purpose

The primary goal of this project is to streamline customer segmentation, facilitating better decision-making, enhanced business processes, and the formulation of effective strategies. By integrating Data Science, Artificial Intelligence and Machine Learning, the project aims to optimize revenue generation and provide businesses with a competitive edge.

2 Literature Survey

2.1 Existing problem

One of the primary issues is the vast and diverse nature of data available for analysis. Customer information is often scattered across numerous channels, including social media, online transactions, and customer service interactions. Integrating and harmonizing these disparate datasets pose a significant hurdle. The heterogeneity of data formats, structures, and sources makes it challenging to create a unified and comprehensive view of the audience.

The main obstacle is the dynamic and evolving nature of customer behavior. Consumer preferences and habits change over time, influenced by various external factors such as cultural trends, economic conditions, and global events. Traditional static segmentation models struggle to capture these nuanced shifts in customer behavior, requiring a more adaptive and real-time approach to audience understanding. The interpretability of machine learning models also poses a significant challenge. As these models often operate as complex black boxes, understanding the rationale behind segmentation decisions becomes difficult. Businesses need to ensure that their machine learning models provide not only accurate predictions but also transparent insights into why specific segments are identified, fostering trust, and facilitating actionable decision-making.

2.2 References

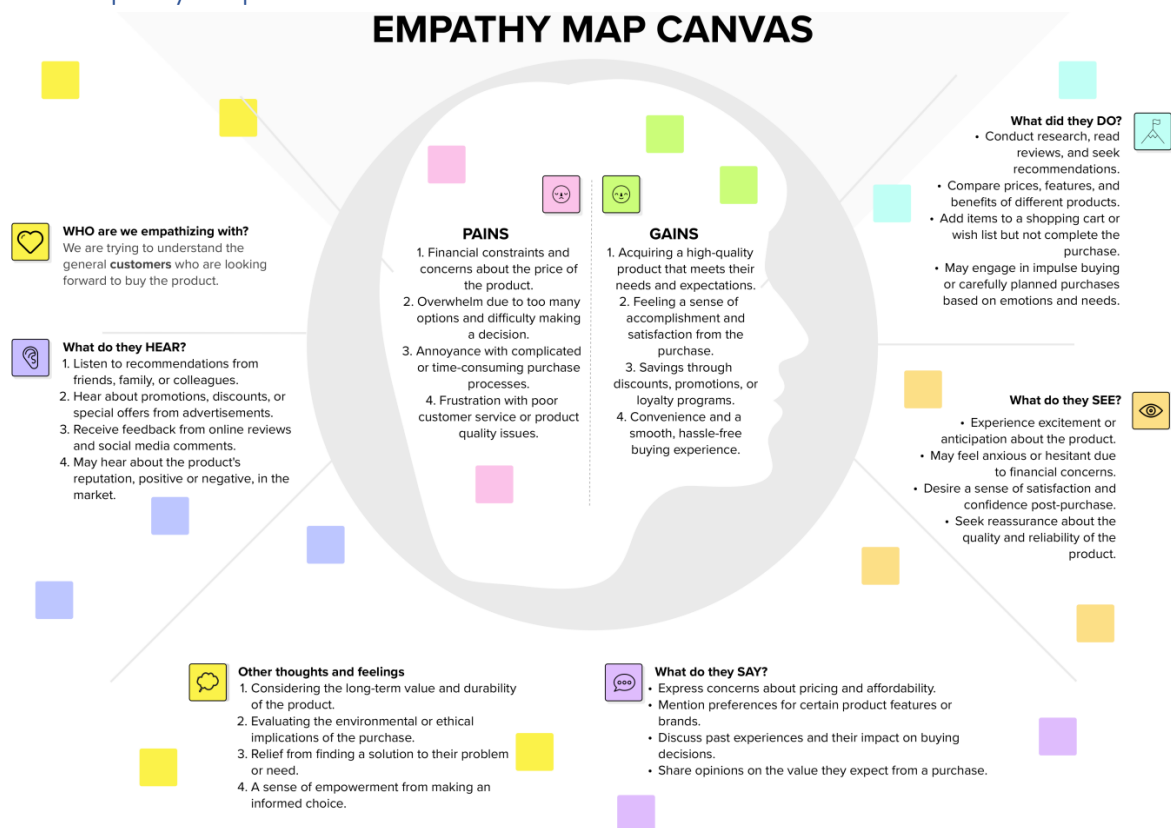
- Wyner, A., Olson, M., Bleich, J. and Mease, D. (2017). Explaining the Success of AdaBoost and Random Forests as Interpolating Classifiers. *Journal of Machine Learning Research*, [online] 18, pp.1–33. doi: <https://www.jmlr.org/papers/volume18/15-240/15-240.pdf>
- Pham, D.T., Dimov, S.S. and Nguyen, C.D. (2005). Selection of K in K-means clustering. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, [online] 219(1), pp.103–119. doi: [phamDN05-kmeans.pdf \(columbia.edu\)](http://phamDN05-kmeans.pdf(columbia.edu))
- Wyner, A.J., Olson, M., Bleich, J. and Mease, D. (2015). Explaining the Success of AdaBoost and Random Forests as Interpolating Classifiers. *arXiv (Cornell University)*.

2.3 Problem Statement Definition

The project aims to leverage machine learning techniques to enhance the understanding of audience behaviour through customer segmentation. By analysing diverse datasets encompassing customer demographics, purchasing history, and online interactions, the goal is to develop a sophisticated model that identifies distinct customer segments. This segmentation will not only provide valuable insights into the varying preferences and needs of different customer groups but also enable businesses to tailor their marketing strategies more effectively. The project seeks to contribute to a deeper comprehension of audience dynamics, ultimately empowering companies to optimize their products, services, and communication strategies for enhanced customer satisfaction and business success.

3 Ideation and Proposed Solution

3.1 Empathy Map Canvas



3.2 Ideation and Brainstorming

Brainstorm

Write down any ideas that come to mind that address your problem statement.

RAMANUJA CHANDURI

Research and gather relevant data sources, such as social media posts, customer reviews, or user interactions.

Develop a Machine Learning model for classification

Discuss potential biases in your data and models and strategies to mitigate them to ensure fairness and inclusivity.

Discuss strategies for communicating project goals and progress to non-technical team members or stakeholders.

Estimate the budget and resource requirements for the project, including data acquisition costs, hardware, and software expenses.

Discuss how to gather user feedback

CHINTHAKUNTA SRAVYA

Brainstorm potential features that could be extracted from the data, such as sentiment scores, keyword frequencies, or user engagement metrics.

Consider using Non supervised Learning algorithms for segmentation

Brainstorm ideas for user interfaces or dashboards that present the insights gained from the machine learning models to non-technical stakeholders.

Explore methods for deploying machine learning models in a scalable and production-ready manner.

Define key performance indicators (KPIs) that align with the project's business goals, such as revenue increase, user retention, or customer satisfaction.

Plan for potential constraints and contingencies to stay within budget.

SRINITHYA SIVA SAI

Explore data preprocessing techniques, including data cleaning, and other data cleaning techniques using defined modules such as Pandas or Numpy.

Consider the pros and cons of collaborative filtering, content-based filtering, deep learning, and traditional algorithms.

Explore methods for transparent and interpretable machine learning models to understand and mitigate unintended consequences.

Discuss techniques for Customer Analysis

Discuss resource allocation, such as the number of data engineers, data scientists, and software engineers needed for the project.

Consider strategies for model version control and continuous integration either via GitHub or Bit Bucket.

NIKHILA MANOGNA

Investigate techniques for creating user profiles or segmenting the audience based on behavior and preferences.

Plan for A/B testing or other experimental methodologies to measure the impact of the machine learning models on the audience.

Explore visualization techniques to represent the output via an application or a web application

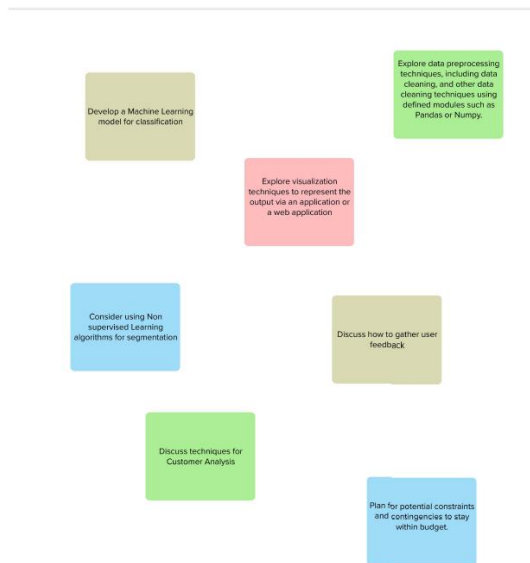
Discuss cloud infrastructure, containerization, and load balancing to handle increased user loads.

Explore where and how we can proceed with the aim for the project.

Plan for user education and onboarding if your solution requires user interaction.

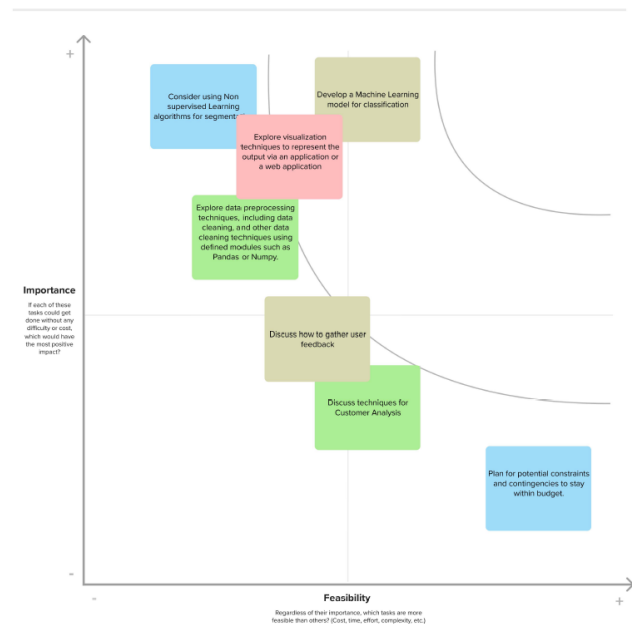
Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.



Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.



4 Requirement Analysis

4.1 Functional Requirements

1. Data Collection and Integration: The system should be able to collect and integrate data from various sources, including customer interactions, demographics, purchase history, and social media.

2. Data Preprocessing: Implement data preprocessing techniques to handle missing values, outliers, and ensure data quality. Also, normalize or standardize data to ensure consistency and comparability.

3. Feature Selection: Incorporate a feature selection process to identify the most relevant attributes for customer segmentation. Allow for customization in selecting features based on specific business requirements.

4. Machine Learning Model Development: Develop machine learning models for customer segmentation based on algorithms such as clustering (e.g., K-means, hierarchical clustering) or classification (e.g., decision trees, random forests). Implement a mechanism for model training, validation, and evaluation.

5. User Interface: Design a user-friendly interface for users to interact with the system. Include visualization tools for presenting segmentation results, trends, and insights.

4.2 Non-Functional Requirements

1. Model Accuracy: The machine learning models must achieve a minimum accuracy threshold of 95% in customer segmentation.

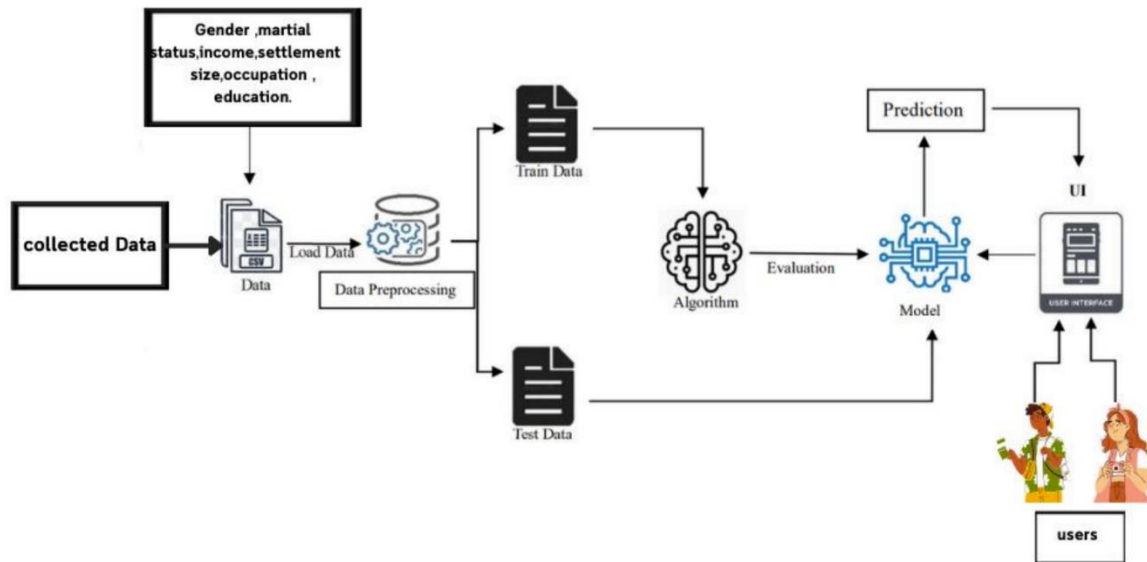
2. Response Time: The system should provide segmentation results in real-time, with a maximum response time of 1 second for 95% of requests.

3. Reliability: The system should maintain an uptime of at least 99.5%, providing users reliable platform.

4. Usability: The user interface should be designed for ease of use, with a learning curve of no more than one hour for new users.

5 Project Design

5.1 Data Flow Diagrams and User Stories



User Type	User Story Number	User Story / Task	Acceptance criteria	Priority
A small business owner struggling to identify their ideal customer base	USN-1	I want to be able to identify my most profitable customer segments so that I can tailor my marketing campaigns to them.	The segmented list of customers will be provided in a variety of formats, including CSV, Excel, and PDF.	High
A large retailer with a vast customer database	USN-2	I want to be able to segment my customer base into more granular and actionable groups so that I can personalize my marketing messages and product recommendations to each segment.	As a user, I will be able to create custom customer segments based on a variety of demographic and behavioral criteria.	High
A financial institution looking to reduce customer churn	USN-3	I want to be able to identify customers at risk of churn so that I can proactively engage with them and offer them targeted incentives to keep them as customers.	The list of customers at risk of churn will be ranked by their likelihood of churning, and will include recommendations for targeted interventions.	High

A non-profit organization seeking to optimize their fundraising efforts	USN-4	I want to be able to segment my customer base into more granular and actionable groups so that I can personalize my marketing messages and product recommendations to each segment.	The user will be able to save and manage their custom customer segments, and use them to create targeted marketing campaigns.	Medium
A healthcare provider aiming to improve patient outcomes	USN-5	I want to be able to identify patients with specific risk factors so that I can provide them with personalized interventions and preventive care.	The list of patients with specific risk factors will be ranked by their level of risk, and will include recommendations for personalized interventions	High
A marketing agency looking to create effective targeted campaigns for their clients	USN-6	I want to be able to create highly targeted campaigns for my clients that resonate with their ideal customer segments.	The segmented list of customers will be provided in a variety of formats, including CSV, Excel, and PDF.	Medium
An insurance company seeking to reduce fraud and abuse	USN-7	I want to be able to identify individuals with a high risk of fraud so that I can implement preventive measures and investigations.	The list of individuals with a high risk of fraud will be ranked by their likelihood of committing fraud, and will include recommendations for preventive measures.	High
A government agency aiming to improve resource allocation for social programs	USN-8	I want to be able to identify individuals and communities with the greatest need for social assistance so that I can allocate resources more effectively.	The list of individuals and communities with the greatest need for social assistance will be ranked by their level of need, and will include recommendations for resource allocation.	High

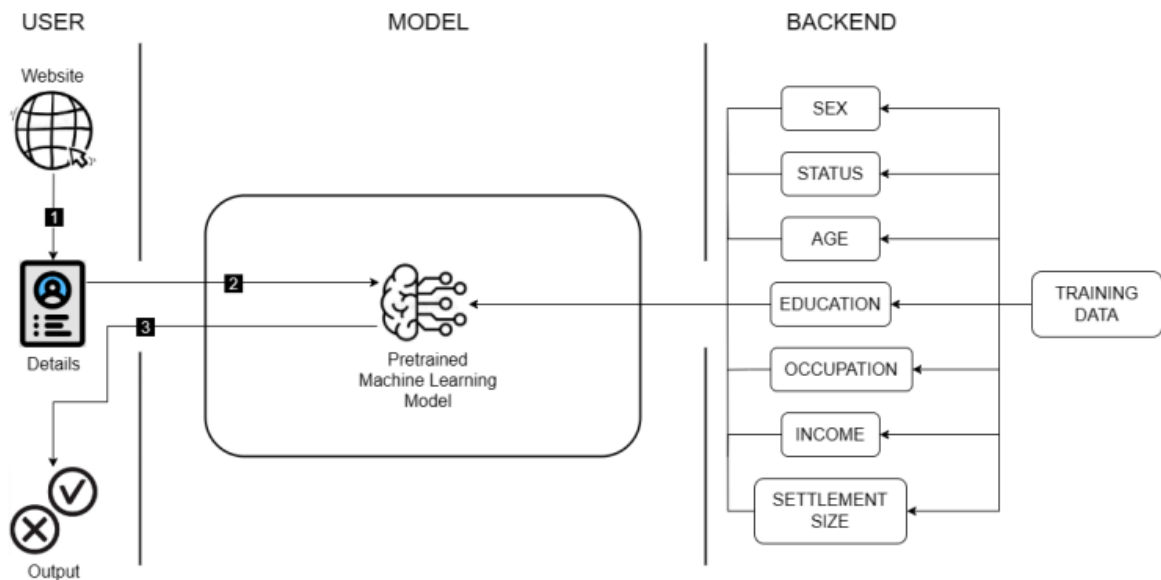
5.2 Solution Architecture

- **Data Ingestion:** Data is collected from the provided sources such as Kaggle/ Amazon Datasets and its structure is analysed.
- **Data Preprocessing:** Data is cleaned, normalized, and transformed to handle missing values and outliers. Feature engineering is performed to extract relevant customer attributes.
- **K-Means Clustering:** The pre-processed data is fed into a K-Means clustering algorithm to group customers with similar characteristics into clusters.
- **Cluster Analysis:** Analyse the resulting clusters and classify them into different groups for supervised machine learning algorithm.
- **Feature Selection:** Identify key features that contribute to the differences between customer clusters.
- **Data Classification:** Random Forest classifier, Adaboost classifier, Decision Tree classifier and XGBoost classifier are trained on the clustered data to predict customer behaviours or preferences within each cluster.
- **Model Evaluation:** Evaluate different model classification performance based on their accuracy score.
- **Saving the Model:** Consider the best solution among the four classification techniques based on the evaluation and save the model with the most considerate results.

- **User Interface:** Develop a user-friendly interface for business users to interact with the system, allowing them to explore customer segments and plan marketing strategies using Flask.

6 Project planning and scheduling

6.1 Technical Architecture



6.2 Sprint Planning and Estimation

Sprint	Functional Requirement(Epic)	User Story Number	UserStory/Task	Story Points	Priority	Team Members
Sprint 1	Customer Segmentation: Understanding Customer Potential	USN-1: Determine Customer Potential	As a customer, I want to use your website to determine if I am a potential customer or not and why.	11	High	Ramanuja Chanduri
Sprint 1	Customer Segmentation: Personalized Recommendations	USN-2: Provide Personalized Recommendations	As a customer, I want to receive personalized recommendations on products and services that are tailored to my needs and preferences.	9	High	Ramanuja Chanduri
Sprint 2	Customer Segmentation: Targeted Marketing Offers	USN-3: Deliver Targeted Marketing Offers	As a customer, I want to be able to access exclusive offers and promotions based on my segmentation information.	20	Low	Sravva Chinthakunta
Sprint 3	Customer Segmentation: Personalized Support	USN-4: Enhance Customer Support with Segmentation	As a customer, I want to be able to access exclusive offers and promotions based on my segmentation information.	20	Medium	Nikhila Manogna
Sprint 4	Customer Segmentation: User Feedback	USN-5: Gather User Feedback on Segmentation	As a customer, I want to be able to provide feedback on the segmentation process and make suggestions for improvement.	20	Medium	Srinithya Siva Sai

6.3 Sprint Delivery Schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2023	29 Oct 2023	20	29 Oct 2023
Sprint-2	20	6 Days	31 Oct 2023	05 Nov 2023	20	05 Nov 2023
Sprint-3	20	6 Days	07 Nov 2023	12 Nov 2023	20	12 Nov 2023
Sprint-4	20	6 Days	14 Nov 2023	19 Nov 2023	20	19 Nov 2023

7 Coding and Solutioning

7.1 Feature 1- Unsupervised Learning Algorithm (K-Means Clustering)

In the context of this project, the incorporation of k-means clustering serves as a pivotal feature in the segmentation process. K-means clustering is a powerful unsupervised machine learning algorithm that partitions the dataset into distinct clusters based on similarity. Leveraging this algorithm allows the system to automatically categorize customers into groups with similar traits and behaviours, enabling a more nuanced understanding of the audience. By analysing patterns and relationships within the data, k-means clustering facilitates the identification of customer segments with fulfilling the need of predefined labels. The flexibility and efficiency of k-means clustering make it an invaluable tool for enhancing the precision and effectiveness of customer segmentation within the broader machine learning framework of the project.

7.1.1 Code

```
from sklearn.cluster import KMeans
from scipy import spatial

#Elbow method to find the best number of clusters
wcss = []
for i in range(1,11):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42)
    kmeans.fit(data)
    wcss.append(kmeans.inertia_)

plt.plot(range(1,11), wcss)
plt.title('Elbow Method')
plt.xlabel('No.Of Clusters')
plt.ylabel('WCSS')
plt.show()

#Using the best found cluster count 3 for the model
km_model = KMeans(n_clusters = 3, init='k-means++', random_state= 42)

ykmeans = km_model.fit_predict(data)
```

7.2 Feature 2- Supervised Learning Algorithm (Adaboost Classifier)

The AdaBoost classifier serves as a pivotal feature in enhancing the accuracy and effectiveness of customer segmentation models. This adaptability of AdaBoost over other models makes it particularly well-suited for scenarios where the distribution of customer data may be complex or uneven. Leveraging the power of AdaBoost in the classification process contributes to the overall model accuracy, enabling more nuanced identification and understanding of diverse customer segments.

7.2.1 Code

```
from sklearn.ensemble import AdaBoostClassifier

adb_model = AdaBoostClassifier()

grid_search = GridSearchCV(adb_model, param_grid, cv=5,
scoring='accuracy')
grid_search.fit(X_train, y_train)

best_params = grid_search.best_params_
```



```
best_adaboost = AdaBoostClassifier(**best_params)
best_adaboost.fit(X_train, y_train)
```

```
# Predictions
```

```
y_pred = best_adaboost.predict(X_test)
```

```
# Model performance
```

```
accuracy = accuracy_score(y_test, y_pred)
```

#Here we used 4 different classifiers to obtain the best one which turned out to be AdaBoost.

7.3 Web App Development

The web page serves as UI for interacting with the ML model. The form embedded in the HTML code collects essential input features, such as gender, marital status, age, education, income, occupation, and settlement size, which are crucial for predicting customer segmentation. The styling elements ensure a visually appealing and user-friendly design, with a background image, form-container styling, and a distinctive colour scheme. The form utilizes the POST method to send user inputs to the '/predict' endpoint on the server for processing. Upon submitting the form, the user-triggered prediction is processed by the back-end, and the resulting customer segmentation prediction is dynamically displayed in the designated area with the class 'prediction-text'. This integration between the HTML front-end and the back-end prediction mechanism creates a seamless user experience, allowing individuals to input their demographic information and receive real-time predictions regarding their likely segmentation.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Customer Segmentation</title>
  <style>
    body {
      background-image: url("../static/images/s1.jpg");
      background-size: cover;
      background-position: center;
      color: black;
      font-family: 'Arial', sans-serif;
      margin: 0;
      display: flex;
      align-items: center;
      justify-content: center;
      height: 100vh;
    }

    .form-container {
      background-color: #f4f4f4;
      padding: 20px;
```

```
        border-radius: 10px;
        box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        width: 500px; /* Adjust the width as needed */
        text-align: center;
    }

    h1 {
        color: #333;
        margin-bottom: 20px;
    }

    form {
        display: flex;
        flex-direction: column;
        align-items: center;
    }

    label {
        margin: 10px 0;
    }

    select, input {
        padding: 8px;
        margin-bottom: 15px;
        width: 100%;
        box-sizing: border-box;
    }

    button {
        background-color: #4caf50;
        color: white;
        border: none;
        padding: 10px;
        border-radius: 5px;
        cursor: pointer;
        width: 100%;
    }

    .prediction-text {
        margin-top: 20px;
        font-weight: bold;
    }
</style>
</head>
<body>
    <div class="form-container">
        <h1>Customer Segmentation</h1>
```

```
<!-- Main Input For Receiving Query to our ML -->
<form action="/predict" method="POST">
  <label for="Sex">Sex:</label>
  <select id="Sex" name="Sex">
    <option value="0">Female</option>
    <option value="1">Male</option>
  </select>

  <label for="MaritalStatus">Marital Status:</label>
  <select id="MaritalStatus" name="Marital status">
    <option value="0">Single</option>
    <option value="1">Married</option>
  </select>

  <label for="Age">Age:</label>
  <input type="number" min="20" max="80" name="Age" placeholder="Age"
required="required"/>

  <label for="Education">Education:</label>
  <input type="number" min="0" max="3" name="Education" placeholder="Education"
required="required"/>

  <label for="Income">Income:</label>
  <input type="number" min="5000" name="Income" placeholder="Income"
required="required"/>

  <label for="Occupation">Occupation:</label>
  <select id="Occupation" name="Occupation">
    <option value="0">Not Working</option>
    <option value="1">Working</option>
    <option value="2">Business</option>
  </select>

  <label for="SettlementSize">Settlement size:</label>
  <select id="SettlementSize" name="Settlement size">
    <option value="1">1</option>
    <option value="0">0</option>
    <option value="2">2</option>
  </select>

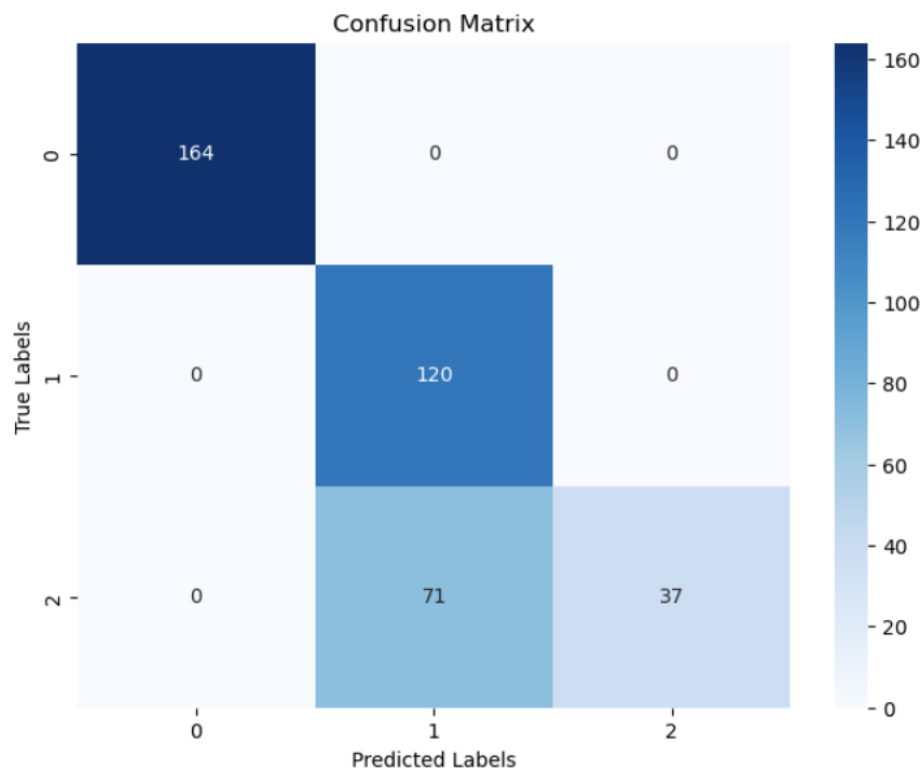
  <button type="submit">Predict</button>
</form>

<div class="prediction-text">{{prediction_text}}</div>
</div>
</body>
</html>
```

8 Performance Testing

8.1 Performance Metrics

8.1.1 Confusion Matrix



8.1.2 Accuracy Score

	Testing Accuracy	Validation Accuracy
0	0.787356	0.818878

8.1.3 Classification Report

	precision	recall	f1-score	support
0	1.00	1.00	1.00	164
1	0.63	1.00	0.77	120
2	1.00	0.34	0.51	108
accuracy			0.82	392
macro avg	0.88	0.78	0.76	392
weighted avg	0.89	0.82	0.80	392

9 Results

9.1 Outputs Screenshots

9.1.1 Initial Website

The screenshot shows the initial state of the 'Customer Segmentation' web application. The interface features a central form with various input fields for customer data, set against a background of stylized human figures in different colors (blue, orange, green) arranged in a circular pattern. The form fields are as follows:

- Sex: Female (dropdown)
- Marital Status: Single (dropdown)
- Age: (text input)
- Education: (text input)
- Income: (text input)
- Occupation: Not Working (dropdown)
- Settlement size: 1 (dropdown)

A green 'Predict' button is located at the bottom of the form.

9.1.2 Case I

This screenshot shows the application after inputting data for 'Case I'. The form fields are now populated with the following values:

- Sex: Female (dropdown)
- Marital Status: Single (dropdown)
- Age: 21 (text input)
- Education: 1 (text input)
- Income: 20000 (text input)
- Occupation: Not Working (dropdown)
- Settlement size: 1 (dropdown)

The green 'Predict' button is still present. Below the form, the text 'Highly potential customer' is displayed in a bold, black font.

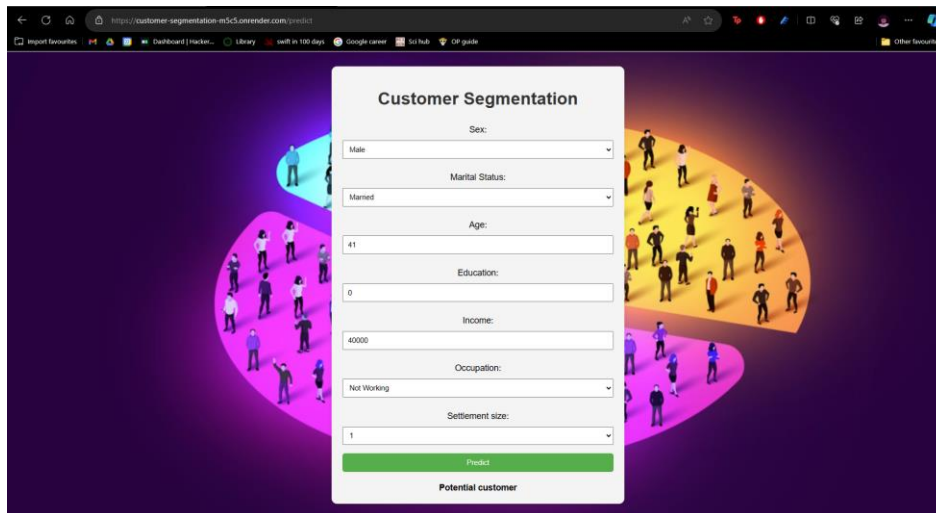
9.1.3 Case II

This screenshot shows the application after inputting data for 'Case II'. The form fields are populated with the following values:

- Sex: Male (dropdown)
- Marital Status: Single (dropdown)
- Age: 45 (text input)
- Education: 2 (text input)
- Income: 500000 (text input)
- Occupation: Business (dropdown)
- Settlement size: 1 (dropdown)

The green 'Predict' button is still present. Below the form, the text 'Not a potential customer' is displayed in a bold, black font.

9.1.4 Case III



The screenshot shows a web browser window with the URL <https://customer-segmentation-m3c5.onrender.com/predict>. The page features a central form titled "Customer Segmentation" with the following fields and values:

Field	Value
Sex	Male
Marital Status	Married
Age	41
Education	0
Income	40000
Occupation	Not Working
Settlement size	1

Below the form is a green "Predict" button and the text "Potential customer". The background of the page is a colorful illustration of a crowd of people.

10 Advantages and Disadvantages

Advantages:

- Enhanced Decision-Making:** The project allows for more informed decision-making by providing a data-driven understanding of customer segments. This can lead to targeted marketing strategies and personalized customer experiences.
- Increased Efficiency:** Adaboost can process large datasets quickly with precision, leading to more efficient customer segmentation compared to traditional methods.
- Precision and Accuracy:** This well-trained model can provide highly accurate and precise segmentation results, uncovering subtle patterns in customer data that might be challenging for manual approaches.
- Scalability:** This project can scale to handle large and diverse datasets, accommodating the growth of customer data and ensuring the continued effectiveness of segmentation models while optimising the accuracy of the model.

Disadvantages:

- Sensitive to noisy data:** The machine learning models such as Adaboost can be sensitive to noise data. This can pose a threat in ML modelling as it can lead to false outputs and hidden errors.
- Overfitting:** Overfitting, where the model learns the training data too well but struggles with new, unseen data, is an obvious challenge considering high accuracies obtained while generating the model. Careful model validation and tuning are necessary to mitigate this risk which is difficult for model running on a small dataset.
- Data limitations:** The data available only considers very limited number of characteristics in predicting the customer. This may pose a threat when introduced to large-scale real-life application in understanding the consumer.

11 Conclusion

The project "Understanding Audience - A Machine Learning Approach to Customer Segmentation" successfully integrates unsupervised learning, specifically K-Means clustering, and supervised learning using the Adaboost classifier to enhance customer segmentation. The combination of these machine learning techniques provides a robust framework for identifying distinct customer segments based on various attributes, contributing to more informed decision-making and personalized marketing strategies.

The project addresses the challenges of handling unlabelled customer data by employing K-Means clustering, which automatically categorizes customers into groups with similar traits. This unsupervised learning algorithm proves pivotal in achieving a nuanced understanding of the audience, enabling the system to adapt to evolving customer behaviours. The incorporation of the Adaboost classifier as a supervised learning feature enhances the accuracy and efficiency of customer segmentation models. The adaptability of Adaboost to complex and uneven data

In conclusion, the project demonstrates a comprehensive and well-executed approach to leveraging machine learning for customer segmentation. While acknowledging the challenges and limitations, the project lays a foundation for future advancements in the field of customer segmentation through machine learning.

12 Future Scope

In advancing this 'Understanding Audience' project, future avenues include real-time data integration for dynamic segmentation, employing advanced machine learning models like neural networks to enhance accuracy, and exploring personalization and recommendation systems. We can consider incorporating explainable AI techniques for transparent insights and implementing a feedback loop system for continuous model improvement. Enhance the user interface for better user experience, reinforce security and privacy measures, and optimize scalability for handling larger datasets.

13 Appendix

13.1 Source Code

Importing required packages

```
import os
import pandas as pd
import numpy as np
import seaborn as sns
import scipy
import matplotlib.pyplot as plt
```

Reading and understanding data

```
data = pd.read_csv('segmentation data.csv', header='infer')
```

```
data.head()
```

	ID	Sex	Marital status	Age	Education	Income	Occupation	\
0	100000001	0	0	67	2	124670	1	
1	100000002	1	1	22	1	150773	1	
2	100000003	0	0	49	1	89210	0	
3	100000004	0	0	45	1	171565	1	

```
4 100000005      0              0  53              1 149031              1
```

```
Settlement size
0      2
1      2
2      0
3      1
4      1
```

```
data.describe()
```

	ID	Sex	Marital status	Age	Education
\					
count	2.000000e+03	2000.000000	2000.000000	2000.000000	2000.000000
mean	1.000010e+08	0.457000	0.496500	35.909000	1.03800
std	5.774946e+02	0.498272	0.500113	11.719402	0.59978
min	1.000000e+08	0.000000	0.000000	18.000000	0.00000
25%	1.000005e+08	0.000000	0.000000	27.000000	1.00000
50%	1.000010e+08	0.000000	0.000000	33.000000	1.00000
75%	1.000015e+08	1.000000	1.000000	42.000000	1.00000
max	1.000020e+08	1.000000	1.000000	76.000000	3.00000

	Income	Occupation	Settlement size
count	2000.000000	2000.000000	2000.000000
mean	120954.419000	0.810500	0.739000
std	38108.824679	0.638587	0.812533
min	35832.000000	0.000000	0.000000
25%	97663.250000	0.000000	0.000000
50%	115548.500000	1.000000	1.000000
75%	138072.250000	1.000000	1.000000
max	309364.000000	2.000000	2.000000

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 8 columns):
#   Column              Non-Null Count  Dtype
---  -
0   ID                   2000 non-null   int64
1   Sex                  2000 non-null   int64
2   Marital status       2000 non-null   int64
3   Age                  2000 non-null   int64
4   Education             2000 non-null   int64
5   Income                2000 non-null   int64
6   Occupation            2000 non-null   int64
7   Settlement size       2000 non-null   int64
dtypes: int64(8)
memory usage: 125.1 KB
```

Null values handling

```
data.isnull().sum()
```

```
ID          0
Sex          0
Marital status  0
Age          0
Education    0
Income       0
Occupation   0
Settlement size  0
dtype: int64
```

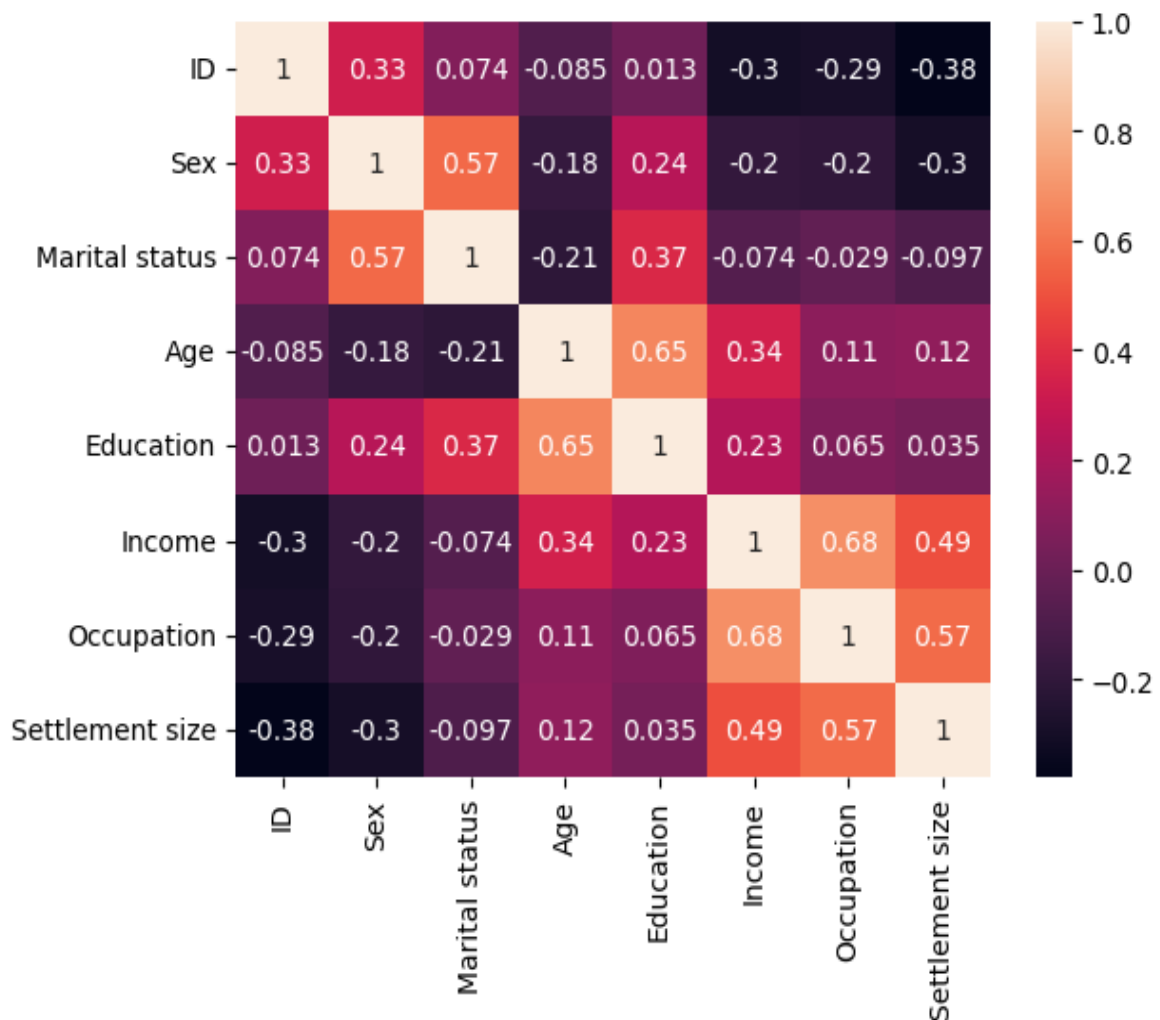
There are no null values

Visualising Data

Heatmap

```
sns.heatmap(data.corr(), annot=True)
```

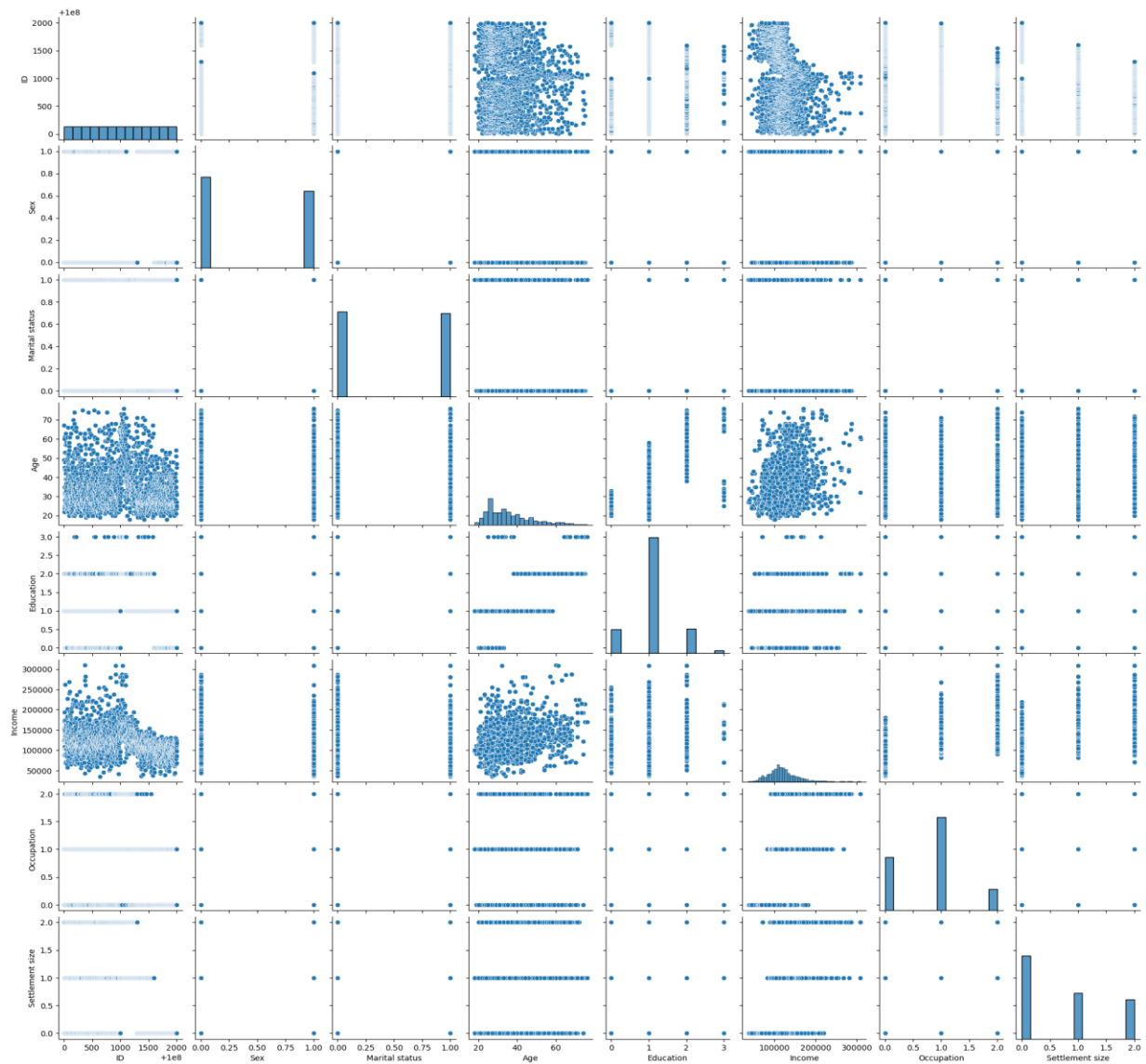
<Axes: >



Pairplot

```
sns.pairplot(data)
```

<seaborn.axisgrid.PairGrid at 0x23841966ad0>



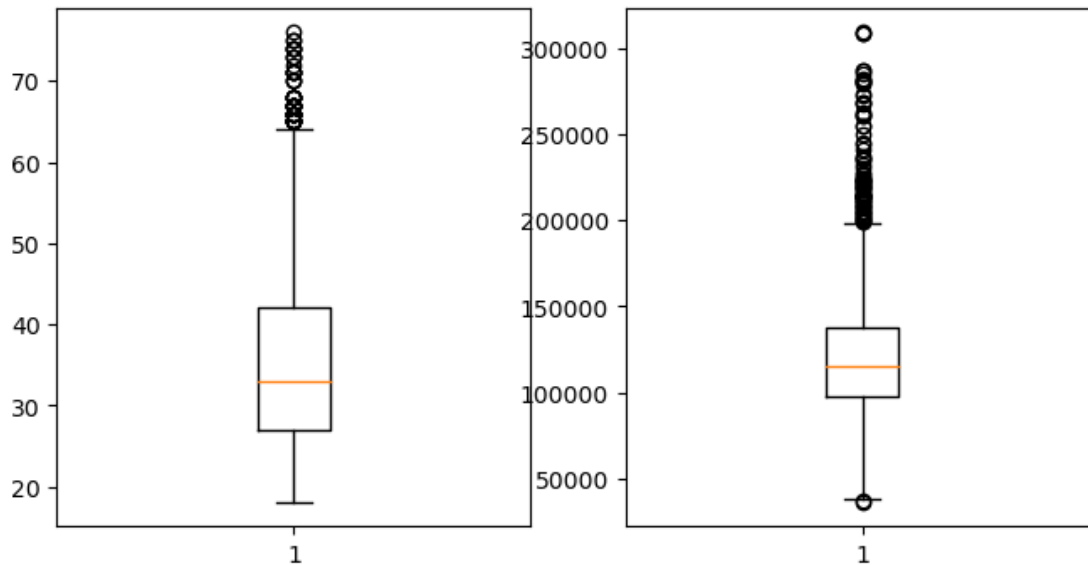
Boxplot

data.columns

```
Index(['ID', 'Sex', 'Marital status', 'Age', 'Education', 'Income',
      'Occupation', 'Settlement size'],
      dtype='object')
```

Uncategorized numerical columns from this dataset are age and income

```
plt.figure(figsize=(8,4))
plt.subplot(121)
plt.boxplot(data['Age'])
plt.subplot(122)
plt.boxplot(data['Income'])
plt.show()
```



There are too many outliers in these two categories. We need to remove them.

Removing Outliers

```
from scipy.stats import zscore
```

```
z_scores = zscore(data['Age'])
outliers = data[(z_scores > 3) | (z_scores < -3)]
data = data[(z_scores <= 3) & (z_scores >= -3)]
```

```
z_scores = zscore(data['Income'])
outliers = data[(z_scores > 3) | (z_scores < -3)]
data = data[(z_scores <= 3) & (z_scores >= -3)]
```

```
data.describe()
```

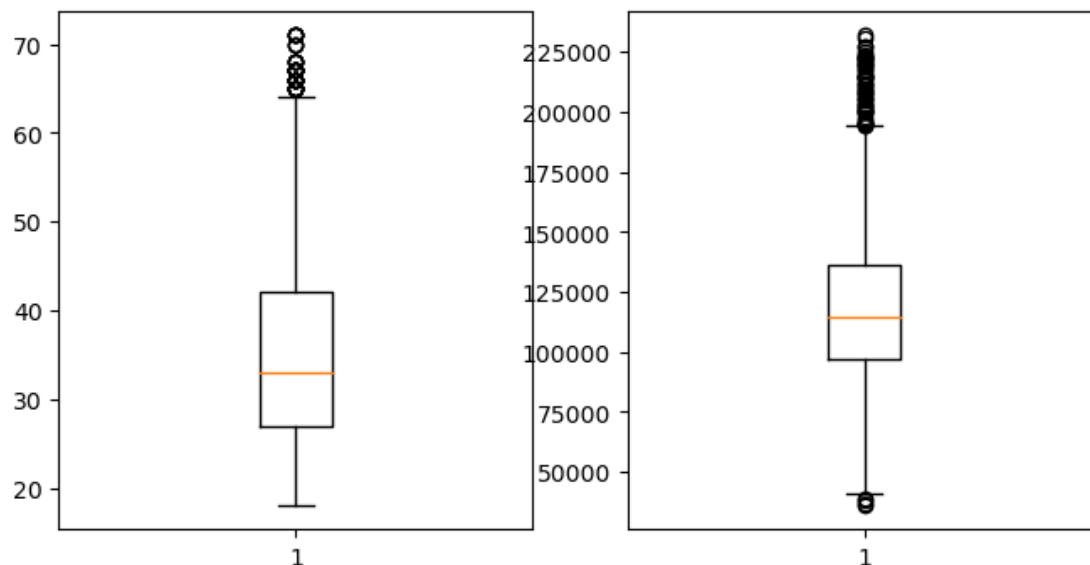
	ID	Sex	Marital status	Age	Education
\					
count	1.958000e+03	1958.000000	1958.000000	1958.000000	1958.000000
mean	1.000010e+08	0.460674	0.499489	35.530644	1.024515
std	5.798969e+02	0.498578	0.500127	11.284894	0.587218
min	1.000000e+08	0.000000	0.000000	18.000000	0.000000
25%	1.000005e+08	0.000000	0.000000	27.000000	1.000000
50%	1.000010e+08	0.000000	0.000000	33.000000	1.000000
75%	1.000015e+08	1.000000	1.000000	42.000000	1.000000
max	1.000020e+08	1.000000	1.000000	71.000000	3.000000

	Income	Occupation	Settlement size
count	1958.000000	1958.000000	1958.000000
mean	118277.183861	0.789581	0.724208
std	33104.213185	0.624531	0.810342
min	35832.000000	0.000000	0.000000
25%	97324.250000	0.000000	0.000000
50%	114807.000000	1.000000	0.000000
75%	136135.750000	1.000000	1.000000
max	231992.000000	2.000000	2.000000

```
data.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 1958 entries, 0 to 1999
Data columns (total 8 columns):
 #   Column                Non-Null Count  Dtype
---  -
 0   ID                    1958 non-null   int64
 1   Sex                   1958 non-null   int64
 2   Marital status        1958 non-null   int64
 3   Age                   1958 non-null   int64
 4   Education              1958 non-null   int64
 5   Income                 1958 non-null   int64
 6   Occupation             1958 non-null   int64
 7   Settlement size        1958 non-null   int64
dtypes: int64(8)
memory usage: 137.7 KB
```

```
plt.figure(figsize=(8,4))
plt.subplot(121)
plt.boxplot(data['Age'])
plt.subplot(122)
plt.boxplot(data['Income'])
plt.show()
```



Feature Scaling

```
from sklearn.preprocessing import MinMaxScaler

minmax = MinMaxScaler()
columns = data.columns
data = pd.DataFrame(minmax.fit_transform(data), columns=columns)

data.drop(['ID'], axis=1, inplace=True)
data.head()
```


	Sex	Marital status	Age	Education	Income	Occupation \
0	0.0	0.0	0.924528	0.666667	0.452885	0.5
1	1.0	1.0	0.075472	0.333333	0.585955	0.5
2	0.0	0.0	0.584906	0.333333	0.272115	0.0
3	0.0	0.0	0.509434	0.333333	0.691950	0.5
4	0.0	0.0	0.660377	0.333333	0.577075	0.5

	Settlement size
0	1.0
1	1.0
2	0.0
3	0.5
4	0.5

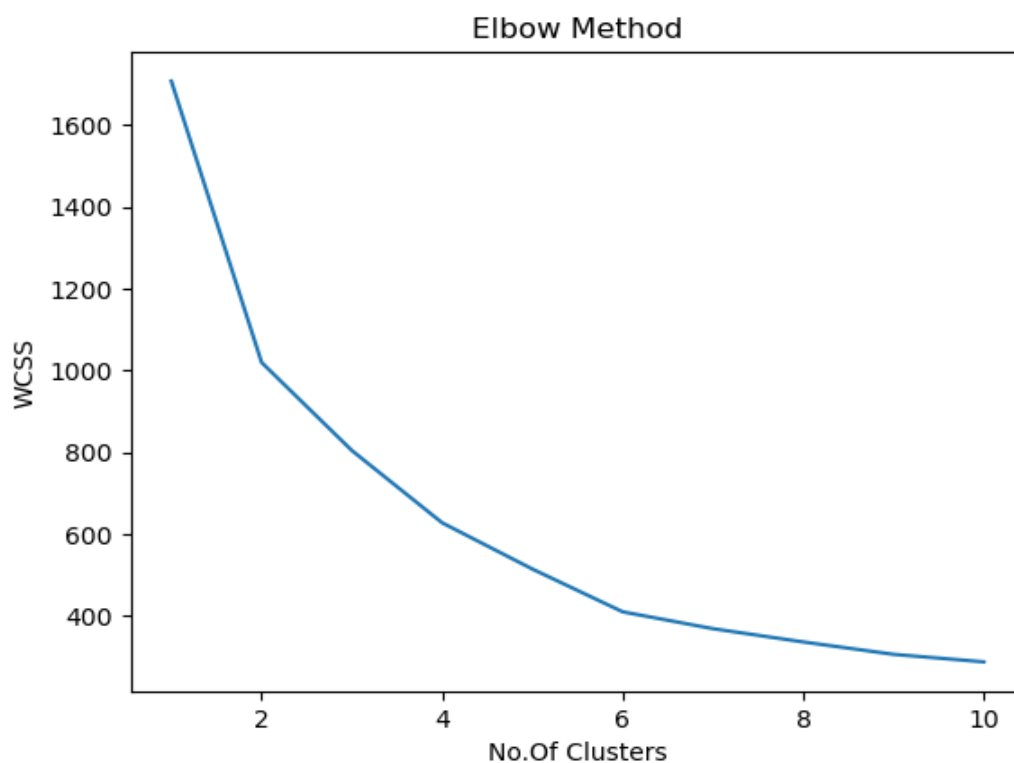
Unsupervised model for clustering

```
from sklearn.cluster import KMeans
from scipy import spatial
```

Elbow method to find the best number of clusters

```
wcss = []
for i in range(1,11):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=42)
    kmeans.fit(data)
    wcss.append(kmeans.inertia_)

plt.plot(range(1,11), wcss)
plt.title('Elbow Method')
plt.xlabel('No.Of Clusters')
plt.ylabel('WCSS')
plt.show()
```



Since the curve has three bends: (0-2, 2-6, 6-10) we will take three clusters

```
km_model = KMeans(n_clusters = 3, init='k-means++', random_state= 42)
```

```
ykmeans = km_model.fit_predict(data)
```

```
C:\Users\raman\AppData\Roaming\Python\Python311\site-  
packages\sklearn\cluster\_kmeans.py:1416: FutureWarning: The default value  
of `n_init` will change from 10 to 'auto' in 1.4. Set the value of  
`n_init` explicitly to suppress the warning  
    super()._check_params_vs_input(X, default_n_init=10)
```

```
data.head()
```

	Sex	Marital status	Age	Education	Income	Occupation \
0	0.0	0.0	0.924528	0.666667	0.452885	0.5
1	1.0	1.0	0.075472	0.333333	0.585955	0.5
2	0.0	0.0	0.584906	0.333333	0.272115	0.0
3	0.0	0.0	0.509434	0.333333	0.691950	0.5
4	0.0	0.0	0.660377	0.333333	0.577075	0.5

	Settlement size
0	1.0
1	1.0
2	0.0
3	0.5
4	0.5

```
data['kclus'] = pd.Series(ykmeans)
```

```
data.head()
```

	Sex	Marital status	Age	Education	Income	Occupation \
0	0.0	0.0	0.924528	0.666667	0.452885	0.5
1	1.0	1.0	0.075472	0.333333	0.585955	0.5
2	0.0	0.0	0.584906	0.333333	0.272115	0.0
3	0.0	0.0	0.509434	0.333333	0.691950	0.5
4	0.0	0.0	0.660377	0.333333	0.577075	0.5

	Settlement size	kclus
0	1.0	1
1	1.0	0
2	0.0	2
3	0.5	1
4	0.5	1

```
data['kclus'].unique()
```

```
array([1, 0, 2])
```

We have successfully divided the model into three categories. Now we need a supervised model to understand the trends and categorize the future/test data

```
from sklearn.model_selection import train_test_split
```

```
x = data.drop(columns = ['kclus'], axis = 1)  
y = data['kclus']
```

```
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size=0.2,
random_state=42)
```

```
print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)
```

```
(1566, 7) (392, 7) (1566,) (392,)
```

Supervised model building for classification

We are going to try four different models and select the best one out of all four

```
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
from xgboost import XGBClassifier
```

```
rand_model = RandomForestClassifier()
tree_model = DecisionTreeClassifier()
xgb_model = XGBClassifier()
adb_model = AdaBoostClassifier()
```

```
rand_model.fit(X_train, y_train)
tree_model.fit(X_train, y_train)
xgb_model.fit(X_train, y_train)
adb_model.fit(X_train, y_train)
```

```
AdaBoostClassifier()
```

```
rand_pred = rand_model.predict(X_train)
tree_pred = tree_model.predict(X_train)
xgb_pred = xgb_model.predict(X_train)
adb_pred = adb_model.predict(X_train)
```

```
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
```

Test Scores

```
scores = {'Random Forest': accuracy_score(rand_pred, y_train), 'Decision
Tree': accuracy_score(tree_pred, y_train), 'XGBoost':
accuracy_score(xgb_pred, y_train), 'Adaboost': accuracy_score(adb_pred,
y_train)}
test_scores = pd.DataFrame([scores])
print(test_scores)
```

	Random Forest	Decision Tree	XGBoost	Adaboost
0	1.0	1.0	1.0	0.787356

```
rand_pred = rand_model.predict(X_test)
tree_pred = tree_model.predict(X_test)
xgb_pred = xgb_model.predict(X_test)
adb_pred = adb_model.predict(X_test)
```

Validation Scores

```
scores = {'Random Forest': accuracy_score(rand_pred, y_test), 'Decision
Tree': accuracy_score(tree_pred, y_test), 'XGBoost':
accuracy_score(xgb_pred, y_test), 'Adaboost': accuracy_score(adb_pred,
```

```
y_test})
valid_scores = pd.DataFrame([scores])
print(valid_scores)
```

```
Random Forest  Decision Tree  XGBoost  Adaboost
0              1.0           1.0       1.0  0.818878
```

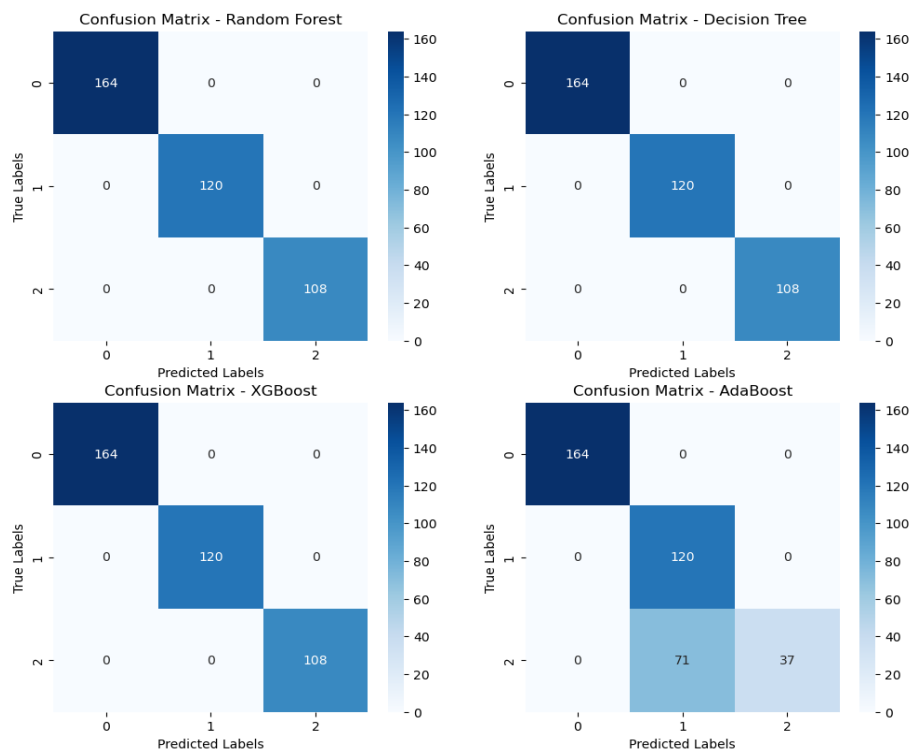
Confusion Matrices

```
rand_conf_matrix = confusion_matrix(y_test, rand_pred)
tree_conf_matrix = confusion_matrix(y_test, tree_pred)
xgb_conf_matrix = confusion_matrix(y_test, xgb_pred)
adb_conf_matrix = confusion_matrix(y_test, adb_pred)
```

```
conf_matrices = [rand_conf_matrix, tree_conf_matrix, xgb_conf_matrix,
adb_conf_matrix]
model_names = ['Random Forest', 'Decision Tree', 'XGBoost', 'AdaBoost']
```

Plotting the matrices

```
fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(12, 10))
for i, ax in enumerate(axes.flatten()):
    sns.heatmap(conf_matrices[i], annot=True, fmt="d", cmap="Blues",
ax=ax)
    ax.set_title(f'Confusion Matrix - {model_names[i]}')
    ax.set_xlabel('Predicted Labels')
    ax.set_ylabel('True Labels')
```



Classification Reports

```
rand_classification_report = classification_report(y_test, rand_pred)
tree_classification_report = classification_report(y_test, tree_pred)
xgb_classification_report = classification_report(y_test, xgb_pred)
adb_classification_report = classification_report(y_test, adb_pred)
```

```
# Print classification reports
print("Classification Report - Random Forest:\n",
      rand_classification_report)
print("Classification Report - Decision Tree:\n",
      tree_classification_report)
print("Classification Report - XGBoost:\n", xgb_classification_report)
print("Classification Report - AdaBoost:\n", adb_classification_report)
```

```
Classification Report - Random Forest:
              precision    recall  f1-score   support

    0           1.00        1.00        1.00        164
    1           1.00        1.00        1.00        120
    2           1.00        1.00        1.00        108

 accuracy          1.00          1.00          1.00        392
 macro avg         1.00          1.00          1.00        392
 weighted avg      1.00          1.00          1.00        392
```

```
Classification Report - Decision Tree:
              precision    recall  f1-score   support

    0           1.00        1.00        1.00        164
    1           1.00        1.00        1.00        120
    2           1.00        1.00        1.00        108

 accuracy          1.00          1.00          1.00        392
 macro avg         1.00          1.00          1.00        392
 weighted avg      1.00          1.00          1.00        392
```

```
Classification Report - XGBoost:
              precision    recall  f1-score   support

    0           1.00        1.00        1.00        164
    1           1.00        1.00        1.00        120
    2           1.00        1.00        1.00        108

 accuracy          1.00          1.00          1.00        392
 macro avg         1.00          1.00          1.00        392
 weighted avg      1.00          1.00          1.00        392
```

```
Classification Report - AdaBoost:
              precision    recall  f1-score   support

    0           1.00        1.00        1.00        164
    1          0.63         1.00        0.77        120
    2           1.00        0.34        0.51        108

 accuracy          0.82          0.78          0.80        392
 macro avg         0.88          0.78          0.76        392
 weighted avg      0.89          0.82          0.80        392
```

Hyperparameter tuning for Adaboost

```
from sklearn.model_selection import GridSearchCV

# Define the hyperparameter grid
param_grid = {
    'n_estimators': [3, 4, 5, 6, 7, 8, 9, 10],
    'learning_rate': [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0],
    'estimator': [None, DecisionTreeClassifier(max_depth=1)],
}

# Use GridSearchCV for hyperparameter tuning
grid_search = GridSearchCV(adaboost_model, param_grid, cv=5,
                           scoring='accuracy')
grid_search.fit(X_train, y_train)

best_params = grid_search.best_params_
best_adaboost = AdaBoostClassifier(**best_params)
best_adaboost.fit(X_train, y_train)

# Predictions
y_pred = best_adaboost.predict(X_test)

# Model performance
accuracy = accuracy_score(y_test, y_pred)

print("Best Hyperparameters:", best_params)
print("Accuracy on Test Set:", accuracy)

Best Hyperparameters: {'estimator': None, 'learning_rate': 0.9,
                       'n_estimators': 4}
Accuracy on Test Set: 0.8188775510204082

Since all the previous models are overfitting, we will use parameter tuned adaboost for our model
adaboost_model = AdaBoostClassifier(estimator=None, learning_rate=0.9,
                                    n_estimators=4)

adaboost_model.fit(X_train, y_train)

AdaboostClassifier(learning_rate=0.9, n_estimators=4)

adaboost_test = adaboost_model.predict(X_train)
adaboost_valid = adaboost_model.predict(X_test)

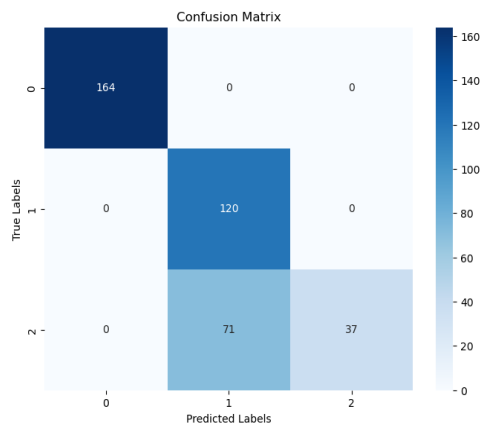
# Accuracy Scores
scores = {'Testing Accuracy': accuracy_score(adaboost_test, y_train),
          'Validation Accuracy': accuracy_score(adaboost_valid, y_test)}
valid_scores = pd.DataFrame([scores])
print(valid_scores)

   Testing Accuracy  Validation Accuracy
0             0.787356                0.818878

# Confusion Matrix
cm = confusion_matrix(y_test, adaboost_valid)
```



```
# Create a heatmap for the confusion matrix
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
            xticklabels=np.unique(y), yticklabels=np.unique(y))
plt.title('Confusion Matrix')
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.show()
```



```
# Classification Report
print(classification_report(y_test, adaboost_valid))
```

	precision	recall	f1-score	support
0	1.00	1.00	1.00	164
1	0.63	1.00	0.77	120
2	1.00	0.34	0.51	108
accuracy			0.82	392
macro avg	0.88	0.78	0.76	392
weighted avg	0.89	0.82	0.80	392

We can conclude that the Adaboost model generalized enough for the outside data. Hence, we will choose Adaboost model

```
import pickle
pickle.dump(adaboost_model, open('Flask/adbmodel.pkl', 'wb'))
pickle.dump(minmax, open('Flask/scaler.pkl', 'wb'))
```

13.2 GitHub and Demo Project Link

13.2.1 GitHub Link

<https://github.com/ch-sravva1712/CustomerSegmentation>

Output link: <https://customer-segmentation-m5c5.onrender.com/>

13.2.2 Demo Project Link

https://drive.google.com/file/d/1qpjRTRNzZc1_7iaUVMauiFegj7XJ7boF/view?usp=sharing