Customer Personality Analysis

Group 3

Team Details

1. Sushma Sagar
2. Ajay Sriram
3. Suyash Dahale
4. JITHIN
5. Santhosh
6. Rahul Mohan Gandhasiri
7. Mohammad Kamran Shaikh

Agenda

1. Introduction
2. Exploratory Data Analysis
3. Principal Component Analysis
4. K-Means Clustering
5. Hierarchical Clustering
6. Profiling
7. Model Building
8. Deployment
9. Conclusion

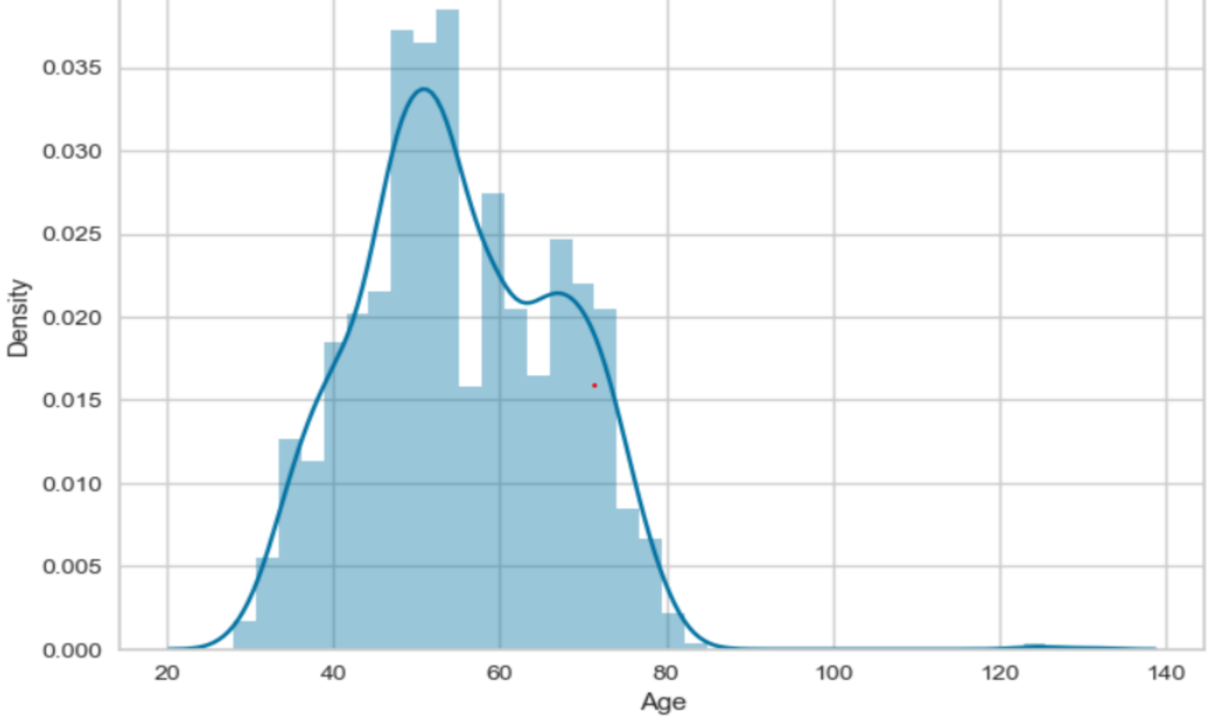
Introduction and Objectives

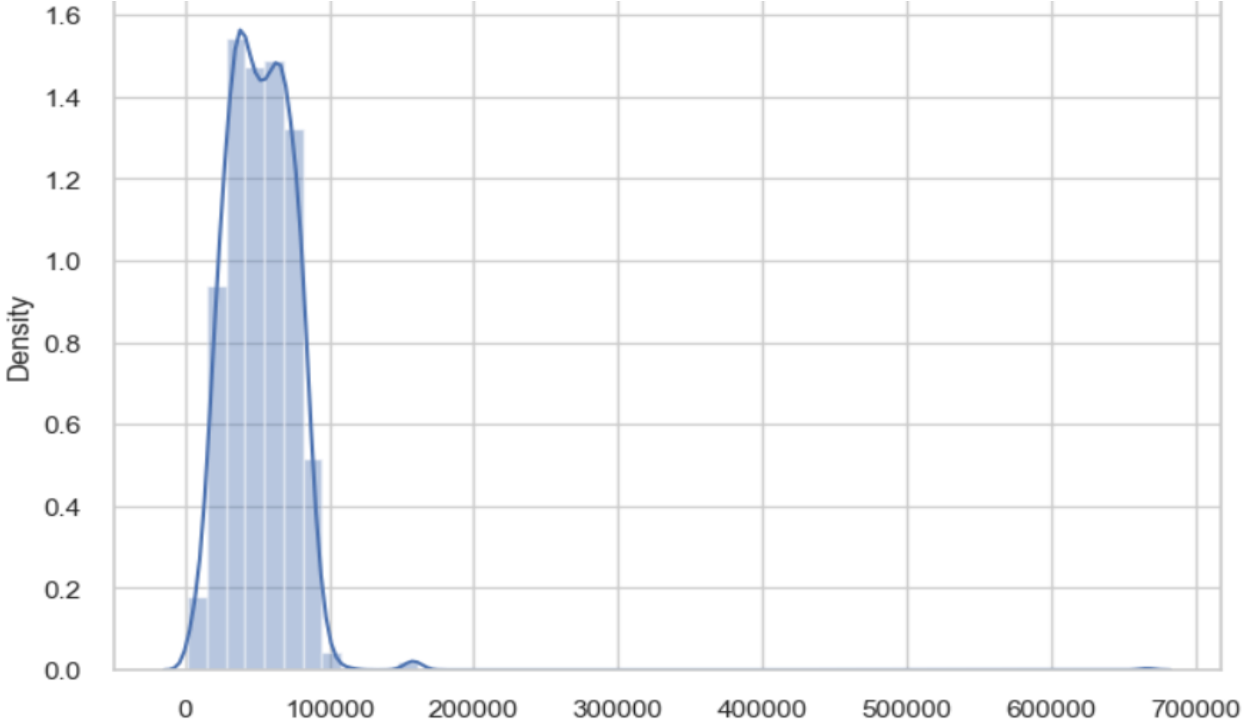
- Customer Personality Analysis is a method utilized by businesses to understand the psychological traits, behaviours, preferences, and motivations of their customers. It involves the use of various techniques and tools to gain insights into customers' personalities, allowing businesses to tailor their products, services, and marketing strategies effectively.
- The primary objective of Customer Personality Analysis is to enhance customer satisfaction, loyalty, and engagement by gaining a deeper understanding of customers' personalities. By identifying their preferences, needs, and decision-making processes, businesses can personalize their offerings, improve customer experiences, and build stronger, more meaningful relationships with their target audience. This ultimately leads to increased customer retention, higher sales, and sustainable business growth.

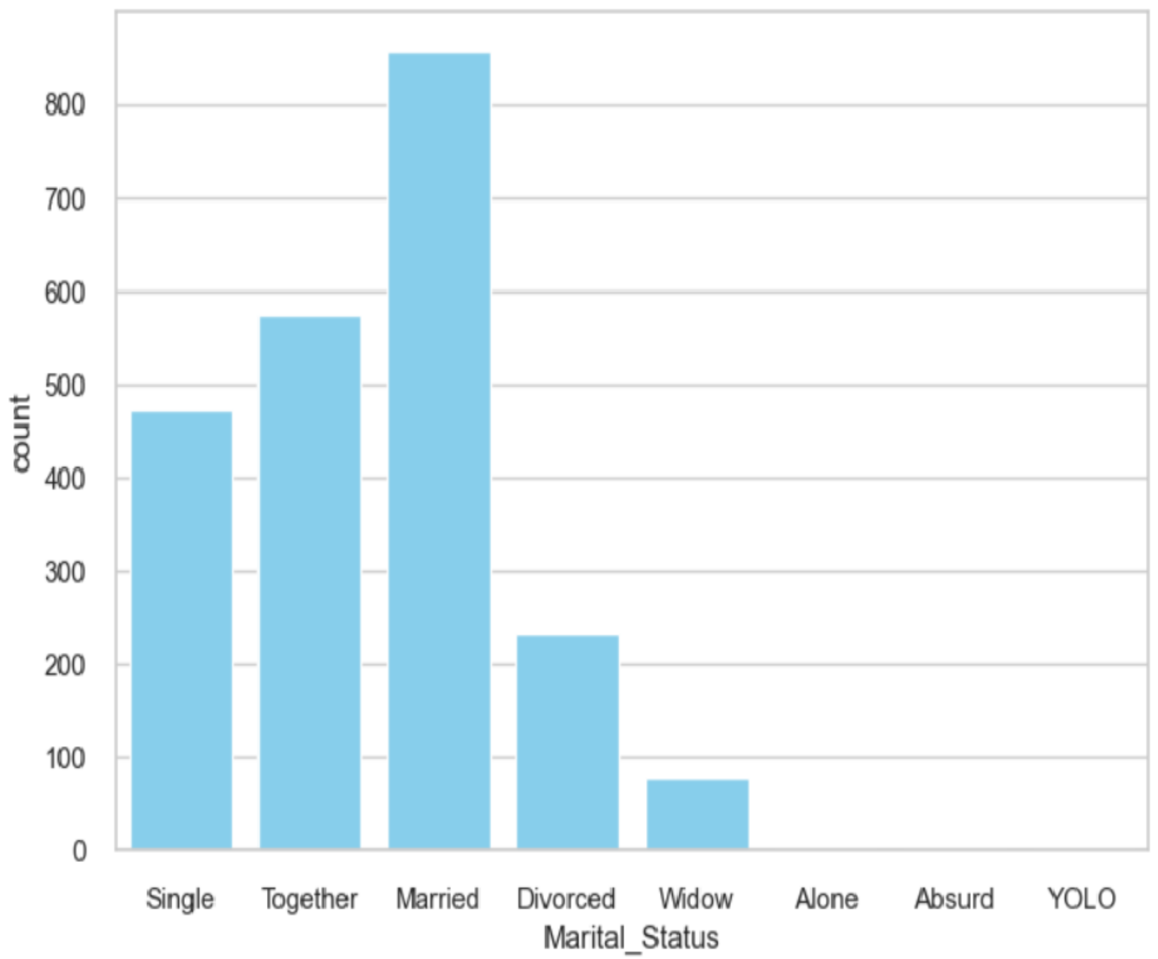
Exploratory Data Analysis(EDA)

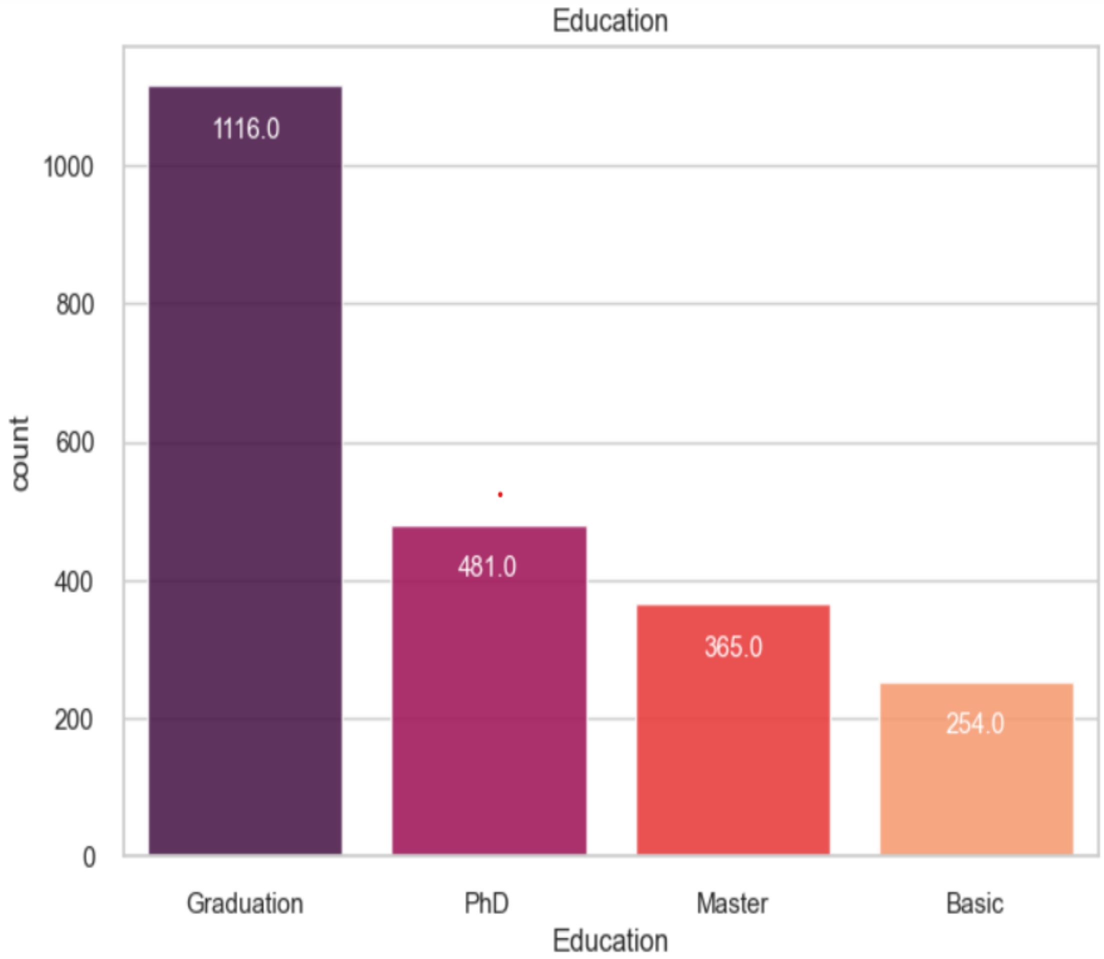
Distrubutions of graph plot with customers Age and Density.

Here the distribution of graph plot with Customers Income.







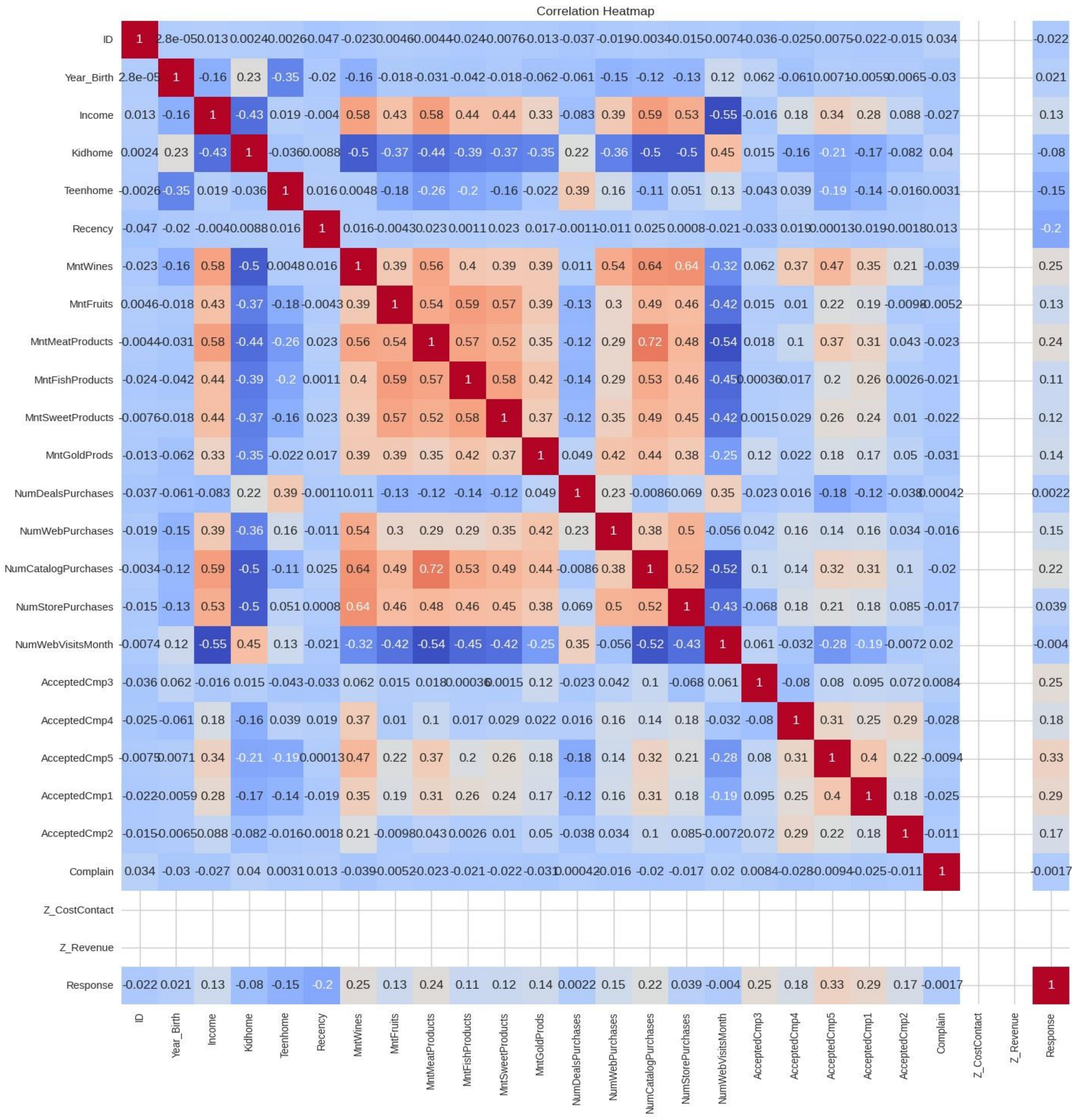


Count plot and Bar plot

Correlation Matrix through a heatmap

From the heatmap observation the data says Strong Positive Correlations: Look for dark red squares along the diagonal and off-diagonal areas. These represent strong positive correlations between variables.

Strong Negative Correlations: Look for dark blue squares, particularly off the diagonal. These represent strong negative correlations. Weak Correlations: Light or pastel shades of red or blue indicate weak positive or negative correlations, respectively.

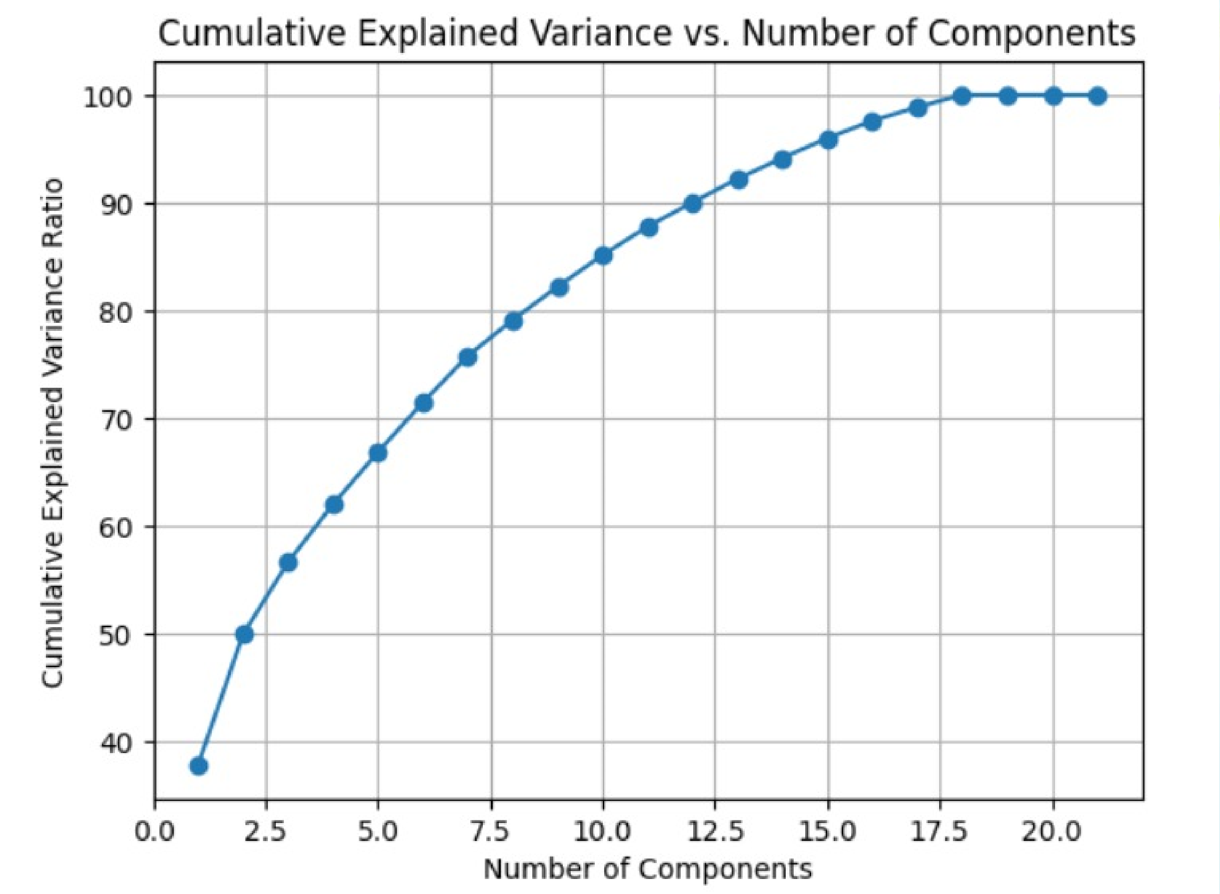


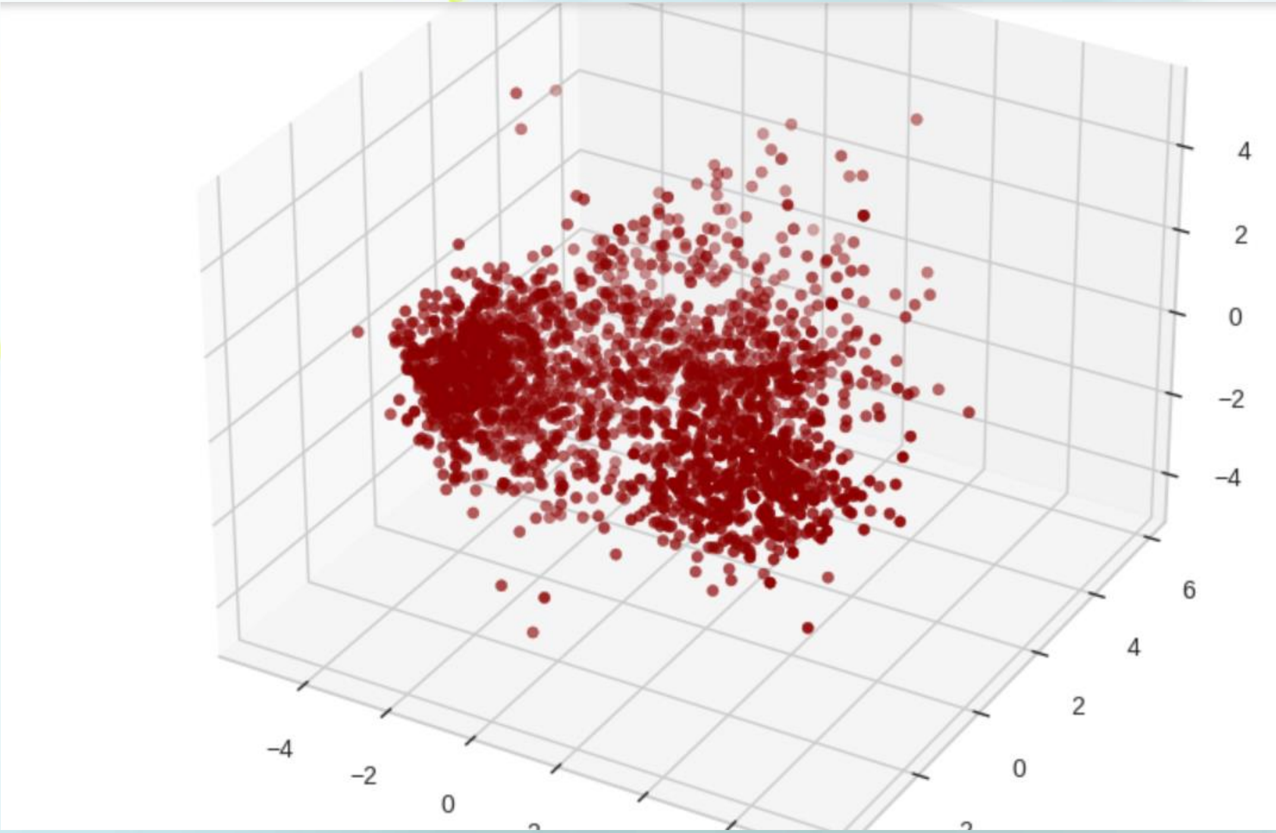
Principal Component Analysis(PCA)

Here the code plots the cumulative sum of the explained variance ratios of the principal components obtained from PCA.

In below 3D graph performs PCA on the dataset das, extracting three principal components.

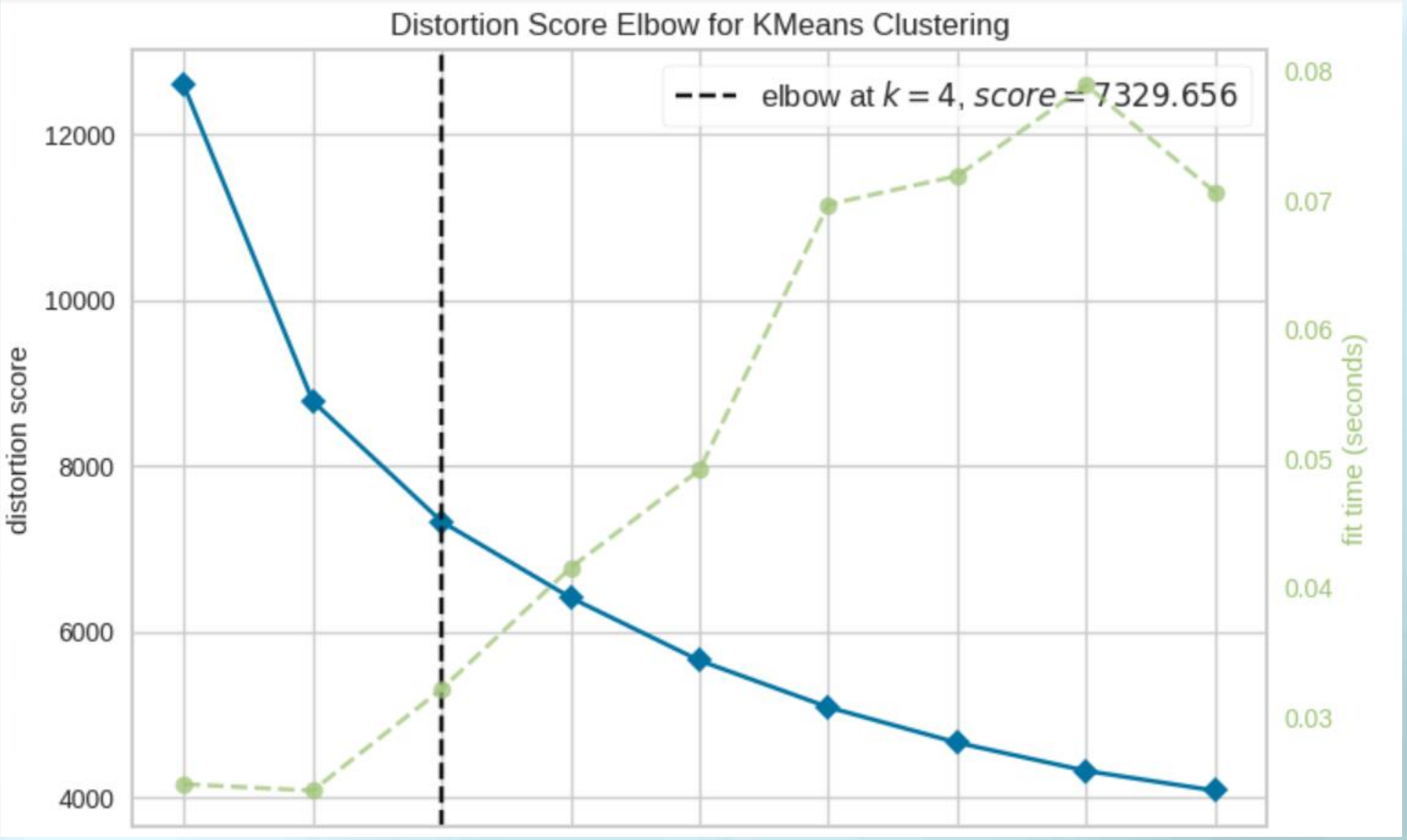
It then visualizes the transformed data in a 3D scatter plot, where each point represents an observation in the dataset projected onto the three principal components.





K-Means Clustering

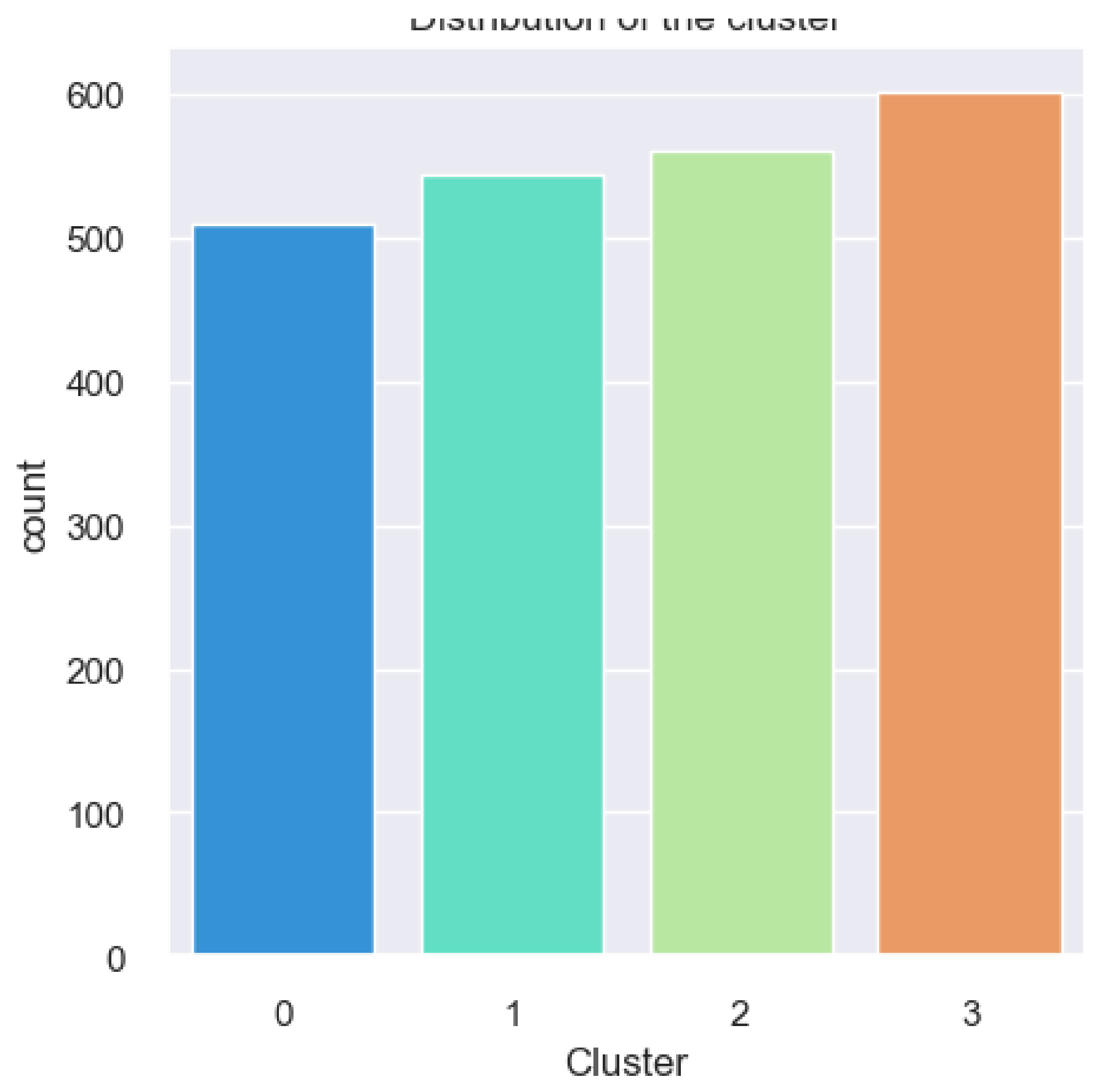
Elbow graph

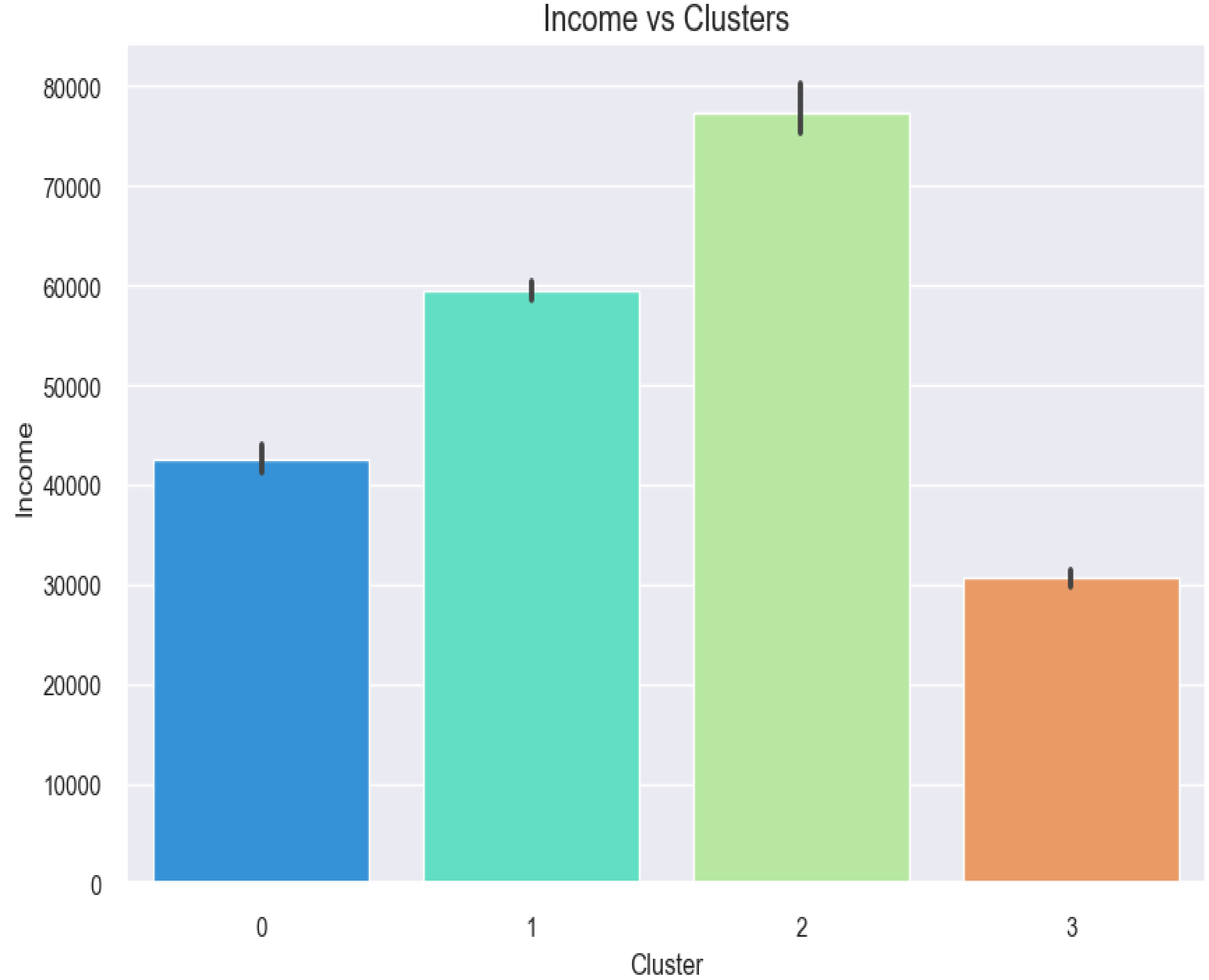


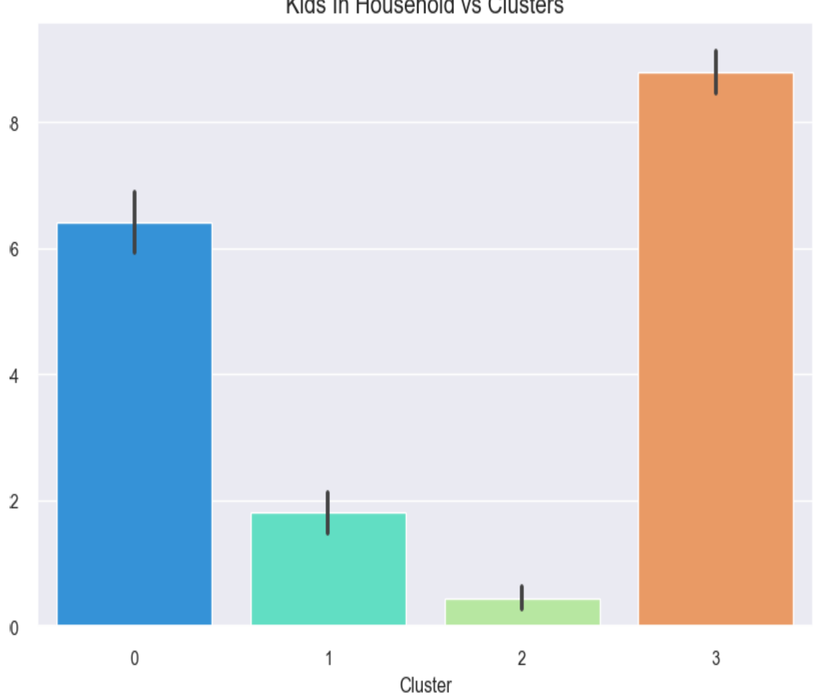
We see that the optimum number of cluster that should be used is K = 4

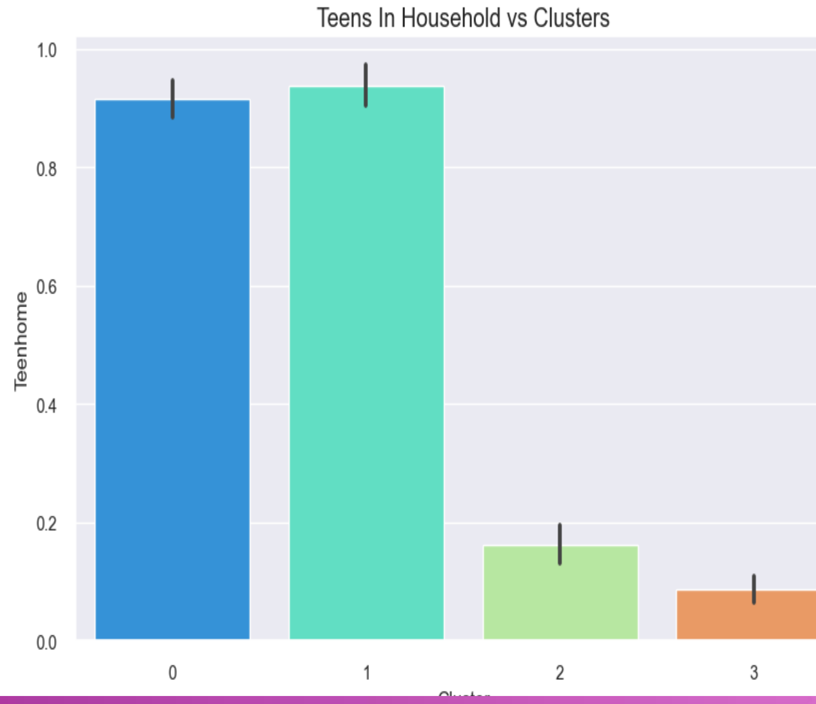
Visualization Of Clustering

Here we utilized Seaborn to create a count plot showing the distribution of clusters present in the Data Frame , with each cluster represented on the x-axis and their respective counts depicted by the height of the bars. The palette parameter specifies the colour scheme.



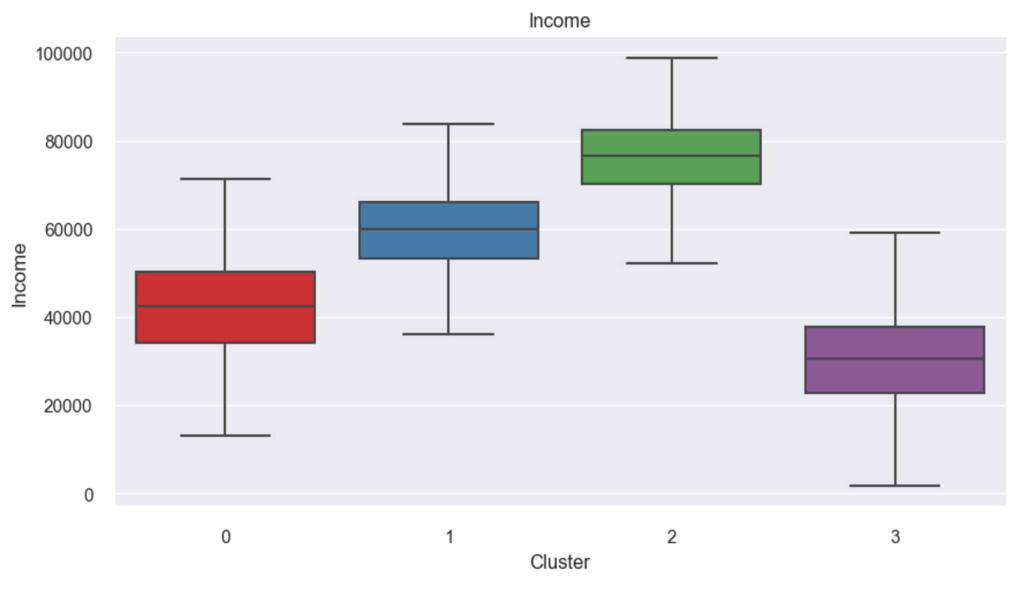

Income vs Clusters
80000
70000
60000
50000
40000
30000
20000
10000
0
Income
0
1
2
3
Cluster

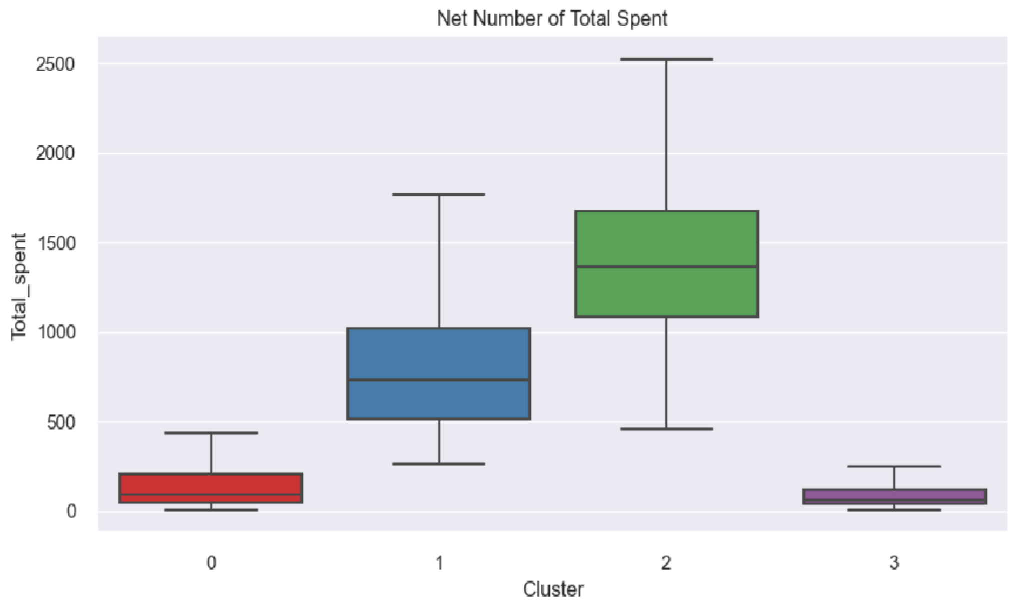

Kids In Household vs Clusters
8
6
4
2
0
0
1
2
3
Cluster


Teens In Household vs Clusters
1.0
0.8
0.6
0.4
0.2
0.0
Teenhome
0
1
2
3

Box Plot

- the distribution of total spent for each cluster in the dataset excluding outliers, providing insights into the variation of spending behaviour across different clusters.





Model Bilding

SVM

```
[ ] from sklearn.svm import SVC
    classifier_svm = SVC(kernel = 'linear',random_state = 0)
    classifier_svm.fit(rescaledx,y_train)
```

```
SVC
SVC(kernel='linear', random_state=0)
```

```
[ ] pred_svm = classifier_svm.predict(rescaledxtest)
```

```
[ ] cm_svm = confusion_matrix(y_test,pred_svm)
    acc_svm = accuracy_score(y_test,pred_svm)
    print(cm_svm)
    print(acc_svm)
```

```
[[108   0   1   2]
 [  0 124   1   0]
 [  1   1 103   0]
 [  1   1   0 105]]
0.9821428571428571
```

Random Forest Classification

Random Forest

```
[ ] from sklearn.model_selection import KFold
    from sklearn.model_selection import cross_val_score
    from sklearn.ensemble import BaggingClassifier
    from sklearn.tree import DecisionTreeClassifier
```

```
# Random Forest Classification

from sklearn.ensemble import RandomForestClassifier


num_trees = 100
max_features = 3
kfold = KFold(n_splits=10, random_state=None)
model_rf = RandomForestClassifier(n_estimators=num_trees, max_features=max_features)
results_rf = cross_val_score(model_rf, rescaledx,y_train, cv=kfold)
print(results_rf.mean())
```

```
0.9468708806729019
```

K-NEAREST NEIGHBOR

```
from sklearn.neighbors import KNeighborsClassifier
model_knn = KNeighborsClassifier(n_neighbors=2)
cv_knn = cross_val_score(model_knn,rescaledx,y_train,cv=kfold)
```

```
[ ]  print(cv_knn)
     print('mean:',cv_knn.mean()*100)
```

```
[0.89944134 0.92178771 0.90502793 0.94413408 0.91620112 0.88268156
 0.91061453 0.94972067 0.89325843 0.89325843]
mean: 91.16125792480071
```

```
[ ]  pred_knn = model_knn.fit(rescaledx,y_train).predict(rescaledxtest)
```

```
[ ]  cm_knn = confusion_matrix(y_test,pred_knn)
     acc_knn = accuracy_score(y_test,pred_knn)
     print(cm_knn)
     print(acc_knn)
```

```
[[104   0   3   4]
 [  0 121   4   0]
 [ 17   7  80   1]
 [  9   1   1  96]]
0.8950892857142857
```

Decision Tree

```
cm_dt = confusion_matrix(y_test,pred_dt)
acc_dt = accuracy_score(y_test,pred_dt)
print(cm_dt)
print(acc_dt)
```

```
[[ 98   0   2  11]
 [  0 121   4   0]
 [  5  17  82   1]
 [  5   0   2 100]]
0.8950892857142857
```

```
#display classification report
from sklearn.metrics import classification_report
print(classification_report(y_test,pred_dt))
```

```
              precision    recall  f1-score   support

           0       0.91      0.88      0.89       111
           1       0.88      0.97      0.92       125
           2       0.91      0.78      0.84       105
           3       0.89      0.93      0.91       107

    accuracy                           0.90       448
   macro avg       0.90      0.89      0.89       448
weighted avg       0.90      0.90      0.89       448
```


Naïve Bayes

Naive Bayes

```
[ ] from sklearn.naive_bayes import GaussianNB
    classifier_nb = GaussianNB()
    classifier_nb.fit(rescaledx,y_train)
```

```
▾ GaussianNB
GaussianNB()
```

```
[ ] pred_nb = classifier_nb.predict(rescaledxtest)
```

```
[ ] cm_nb = confusion_matrix(y_test,pred_nb)
    acc_nb = accuracy_score(y_test,pred_nb)
    print(cm_nb)
    print(acc_nb)
```

```
[[102   0   6   3]
 [  0 116   9   0]
 [  0   1 104   0]
 [ 10   1   3  93]]
0.9263392857142857
```

Choosing the model with High accuracy

COMPARING THE ACCURACIES OF THE MODELS

```
accu = pd.DataFrame({
    'Model': ['Decision Tree(Entropy)','Decision Tree(Gini)', 'Random Forest', 'SVM', 'Naive Bayes', 'KNN'],
    'Accuracy': [acc_dt.mean() * 100, acc_gini.mean() * 100, acc_rf.mean() * 100, acc_svm * 100, acc_nb * 100, acc_knn * 100]
})


accuracy = accu.sort_values(by='Accuracy', ascending=False)


print(accuracy)
```

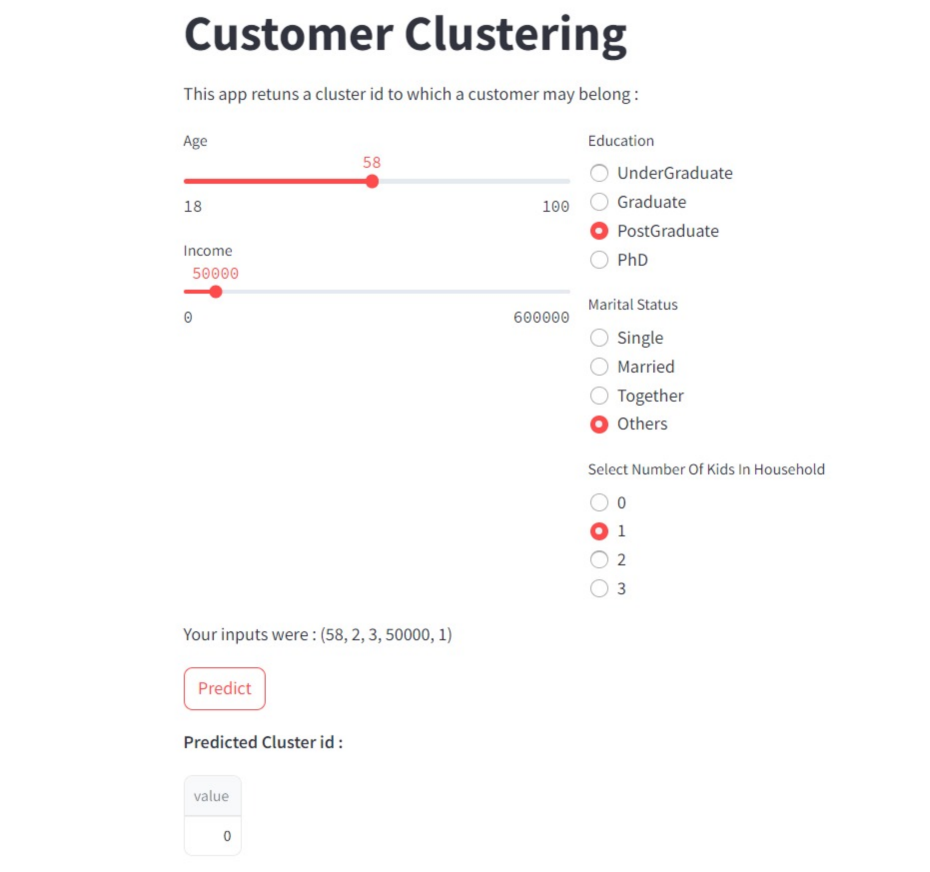
```
                    Model   Accuracy
3                     SVM  98.214286
2           Random Forest  94.196429
4             Naive Bayes  92.633929
0  Decision Tree(Entropy)  89.508929
5                     KNN  89.508929
1     Decision Tree(Gini)  88.616071
```

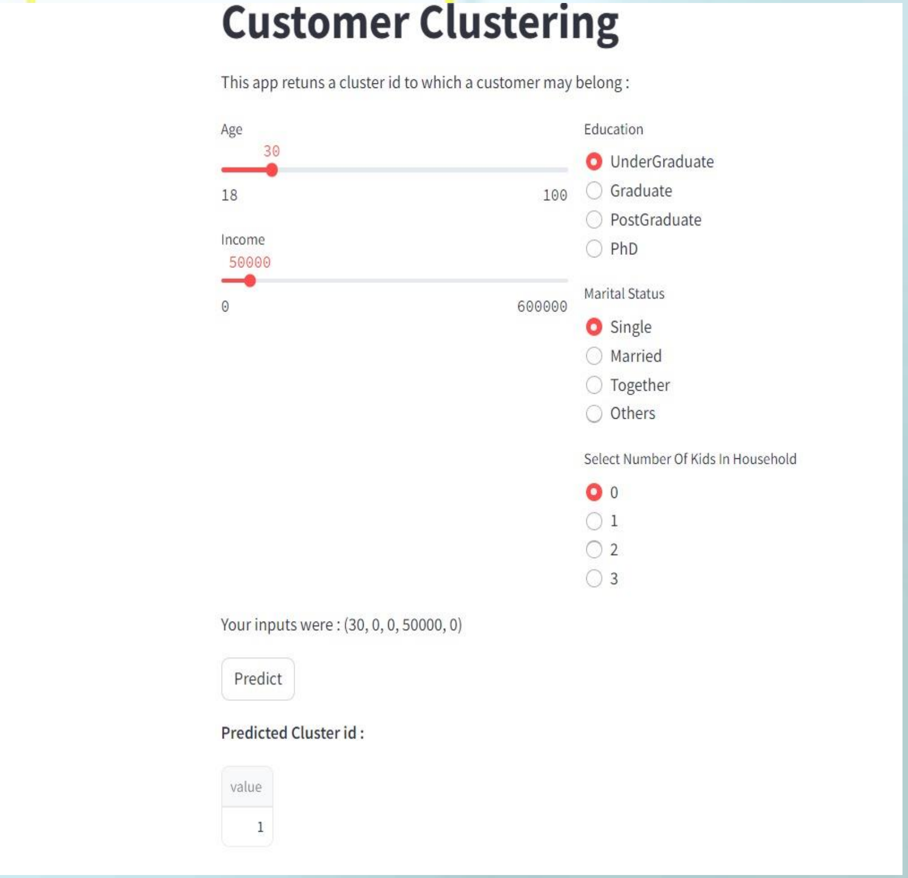

Here we got SVM with the highest accuracy of 98.214 for the model building

Deployment

- **Loading Data:**
- The script loads historical Customers data from a CSV file. Data Preparation:
- Unnecessary columns (“unnamed") are removed from the columns.
- **Streamlit Model Fitting:**
- The Stresmlit model is used for time series forecasting. It is trained on a specified portion of the data.
- **User Interaction:**
- The Streamlit app allows users to select a data to be displa, and the script filters the data accordingly.
- **Displaying Data:**
- The selected data is displayed in a table format on the web page.
- **Model Prediction and Evaluation:**
- The model predicts the Customers personality based on the training data. The predictions are compared , and the Root Mean Squared Error (RMSE) is calculated to measure the model's accuracy.
- **Visualizations:**
- The app provides various options to predict the customers prediction .
- **User Interaction (Forecast):**
- Users can use a app.

Customer Prediction User Interface





CONCLUSION

- Customer Personality Analysis offers valuable insights into the psychological traits, behaviors, and preferences of customers, aiding businesses in tailoring their products, services, and marketing strategies effectively.
- Through techniques like PCA, hierarchical clustering, and visualization tools such as scatter plots and box plots, businesses can identify distinct customer segments, uncover patterns, and make data-driven decisions to meet diverse customer needs.
- Overall, Customer Personality Analysis serves as a powerful tool for businesses striving to build stronger, more personalized relationships with their customers and drive success in today's competitive market.

THANK YOU