Group Assignment: Product Survey Data Analysis

SUBMITTED

BY

Lee Yuan Yeh Derrick/U2021101E

Nicholas Chua Ming Hao/U2022058D

Georemon Russelraj/U2022586A

SCHOOL OF MECHANICAL AND AEROSPACE ENGINEERING

An MA4829 Group Project Report

presented to

Prof Moon Seung Ki

Submitted on 16th February 2024

Year 2023/2024

Abstract

This report is prepared for Assoc. Prof. Moon Seung Ki and submitted as part of the MA4829 Machine Intelligence course group assignment. It serves as a comprehensive analysis of a product survey dataset that aims to collect user feedback regarding car preferences. The survey features entries from 50 survey respondents, exploring factors influencing car purchase decisions, including customisation preferences and willingness to pay for personalised designs. Additionally, the dataset also includes demographic information, which serves as useful data that will aid the formulation of logical problem statements for insightful analysis.

Employing advanced data mining techniques, such as principal component analysis (PCA), classification, and association rule mining, this report utilises software tools including Visual Basic for Applications (VBA), matplotlib, and pandas, which ultimately contribute to a deeper understanding of consumer preferences and buyer decision making process in the automotive market.

Table of Contents

List of Figures

Chapter 1: Introduction

1.1 Motivation

In machine learning, data mining allows us to analyse large datasets to identify patterns or relationships that can help train models to make better predictions. In the automotive industry, data mining can be used to understand consumer behaviour and preferences, essential for companies aiming to stay competitive and relevant in the market. By identifying trends from data mining, businesses can gain valuable insights into consumer preferences for car customization, enabling them to tailor their products and services more effectively to meet the needs and desires of their target audience. This project applies these techniques to gain insights from a product survey dataset involving the preference for car customization among 50 survey respondents.

1.2 Objective and Scope

Our report will explore the relationship between the features of our survey dataset and car customisation likelihood. First, we will manipulate and preprocess our raw dataset by using the Visual Basic Analysis (VBA) language to perform one-hot encoding to extract data classes from columns represented by 0s and 1s. This will allow us to convert categorical data into a format suitable for machine learning algorithms. Subsequently, we will employ data mining techniques, including classification and principal component analysis (PCA) to help identify the feature most impactful to the overall dataset and from there, train a model capable of predicting the customisation likelihood. Our project also uncovers associations between various factors when customising a car by using the rule association mining technique. This will enable us to understand co-occurrences and relations between data classes.

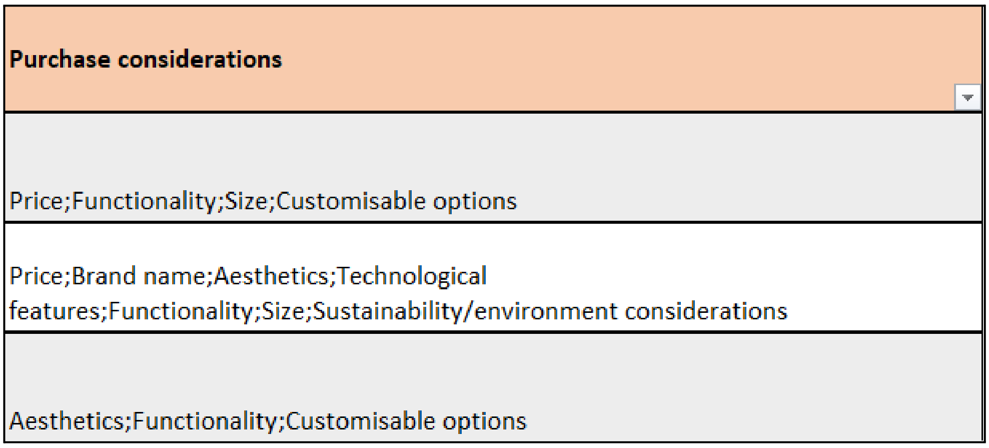
Chapter 2: Analysis

2.1 Data Selection

In this project, data selection has already been done for us. Thus, we will only be conducting data cleaning, transformation, mining and interpretation in this report. The initial dataset provided to us is made up of purely categorical data, hence, we will need to transform the data into a more useful form for data mining.

2.2: Data Cleaning & Transformation

To be able to datamine effectively, the data from the excel sheet cannot be used directly without transformation. The columns; "Purchase considerations", "Customisable exterior features" and "Customisable interior features" all have multiple variables in a single cell which make it difficult to process while data mining. This can be seen in Figure 1. Thus, we will use the built-in programming language, Visual Basic for Applications (VBA), found in Microsoft Excel to transform the data into a more interpretable form.



Purchase considerations
Price;Functionality;Size;Customisable options
Price;Brand name;Aesthetics;Technological features;Functionality;Size;Sustainability/environment considerations
Aesthetics;Functionality;Customisable options

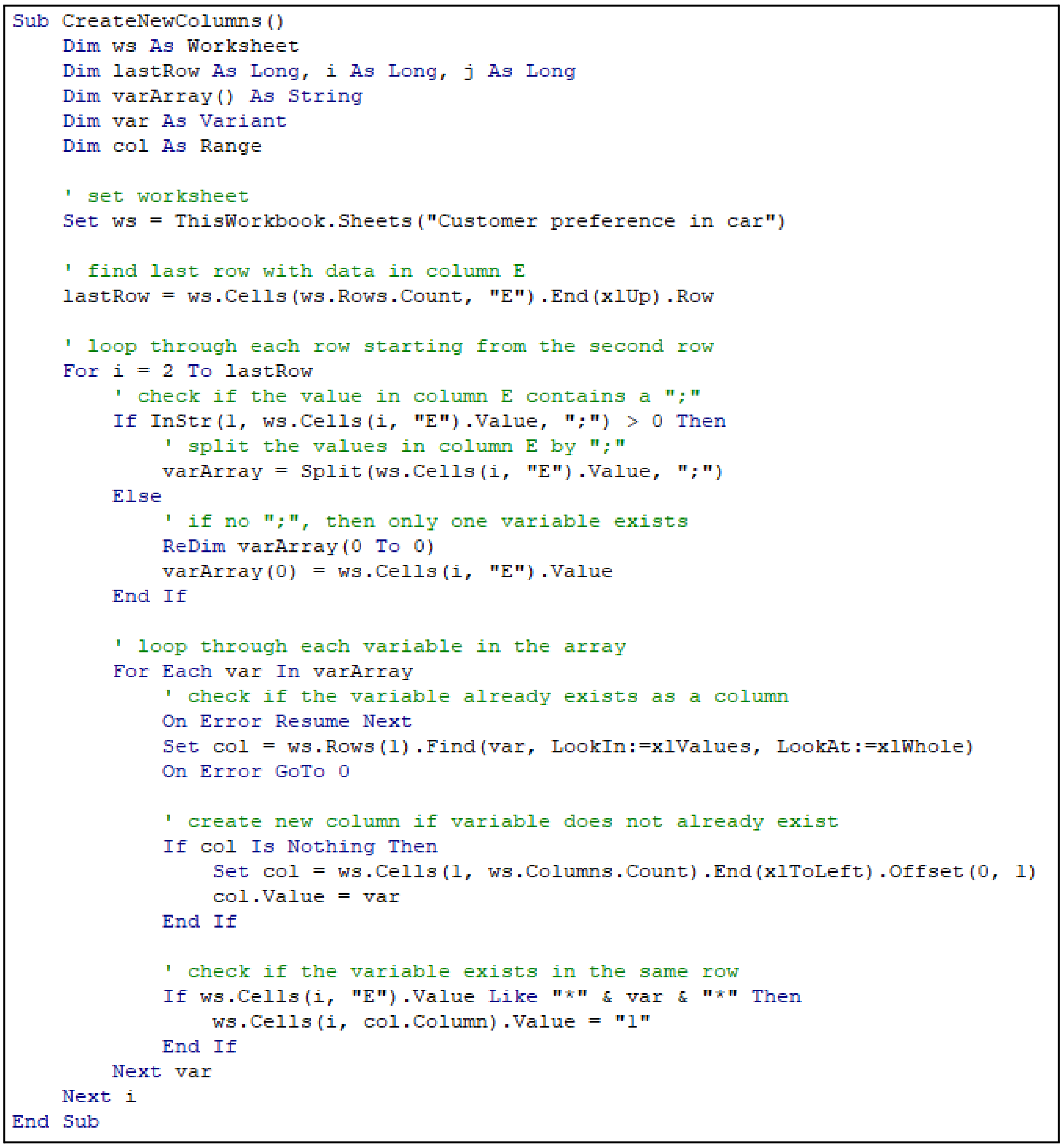
Figure 1. Multiple Variables Within a Single Column in a Single Cell

2.2.1 Visual Basic for Applications

There are many methods we could have used to solve the issue highlighted in Figure 1, we opted to use VBA to create a script that would automatically go down a column and split the string in each cell from ";" and turn each unique variable split found in the column into a new column to the right of the sheet. This method, commonly known as one-hot encoding [1], turns nominal (categorical) data into ordinal (numeric) data. By solving the issue from within the excel sheet itself, it makes the data mining process more streamlined as we would not have to clean up the data with each different mining approach.

2.2.2 VBA Code For One-Hot Encoding

Figure 2 shows our code written to transform the chosen data from a single column with multiple variables in a cell into multiple columns with binary input of either “1” or an empty cell.

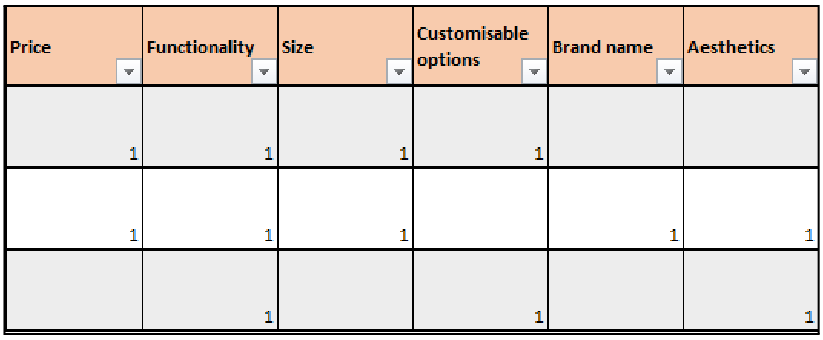


```
Sub CreateNewColumns()
    Dim ws As Worksheet
    Dim lastRow As Long, i As Long, j As Long
    Dim varArray() As String
    Dim var As Variant
    Dim col As Range

    ' set worksheet
    Set ws = ThisWorkbook.Sheets("Customer preference in car")

    ' find last row with data in column E
    lastRow = ws.Cells(ws.Rows.Count, "E").End(xlUp).Row

    ' loop through each row starting from the second row
    For i = 2 To lastRow
        ' check if the value in column E contains a ";"
        If InStr(1, ws.Cells(i, "E").Value, ";") > 0 Then
            ' split the values in column E by ";"
            varArray = Split(ws.Cells(i, "E").Value, ";")
        Else
            ' if no ";", then only one variable exists
            ReDim varArray(0 To 0)
            varArray(0) = ws.Cells(i, "E").Value
        End If

        ' loop through each variable in the array
        For Each var In varArray
            ' check if the variable already exists as a column
            On Error Resume Next
            Set col = ws.Rows(1).Find(var, LookIn:=xlValues, LookAt:=xlWhole)
            On Error GoTo 0

            ' create new column if variable does not already exist
            If col Is Nothing Then
                Set col = ws.Cells(1, ws.Columns.Count).End(xlToLeft).Offset(0, 1)
                col.Value = var
            End If

            ' check if the variable exists in the same row
            If ws.Cells(i, "E").Value Like "*" & var & "*" Then
                ws.Cells(i, col.Column).Value = "1"
            End If
        Next var
    Next i
End Sub
```

Figure 2. VBA Code for Columns with Multiple Variables in Single Cell

The code uses nested “For” loops and arrays in order to identify and split the multi-variable cells into its own columns. It creates new columns of identified unique variables and assigns a “1” in the new column if the same variable was previously found in the same row. This turns the data into something more directly usable for the data mining process. The code has to be run once for each of the columns we want to extract the variables of. As such, it was run a total of 3 times for each of the columns with multi-variable cells.

An example of the results after the code was run can be seen in the figure below.



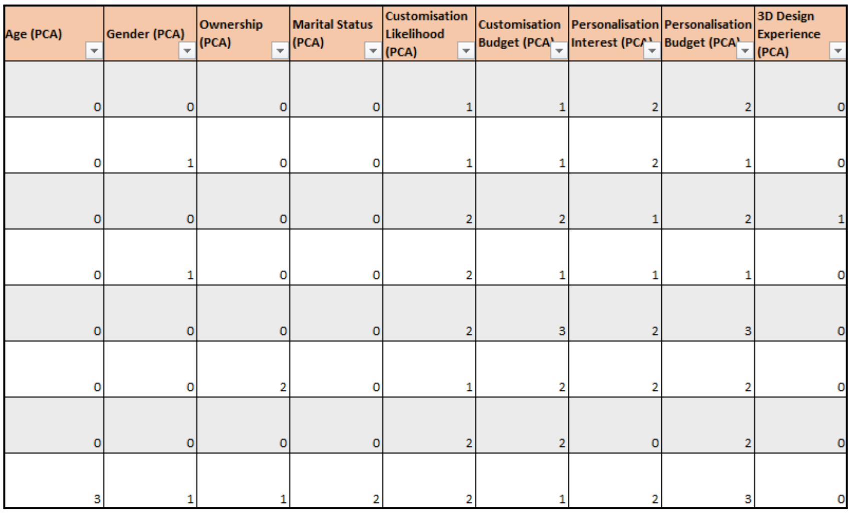
Price	Functionality	Size	Customisable options	Brand name	Aesthetics
1	1	1	1		
1	1	1		1	1
	1		1		1

Figure 3. New transformed columns automatically generated by VBA code

The transformed data set now has 52 columns instead of the original 13 columns due to the addition of new columns via the above one-hot encoding method. This would allow us to run more meaningful analysis on our dataset.

2.2.3 Data Transformation for Principal Component Analysis

Principal Component Analysis (PCA) requires its own form of data transformation due to the nature of our pure categorical data. We need to assign arbitrary numerical weights to our data for PCA to be done more meaningfully, known as integer encoding [1]. This was done easily in Excel using a formula that identifies a specific string and outputs an integer that we arbitrarily associate with the unique string. An example of the output created is shown in the figure below.



Age (PCA)	Gender (PCA)	Ownership (PCA)	Marital Status (PCA)	Customisation Likelihood (PCA)	Customisation Budget (PCA	Personalisation Interest (PCA	Personalisation Budget (PCA	3D Design Experience (PCA)
0	0	0	0	1	1	2	2	0
0	1	0	0	1	1	2	1	0
0	0	0	0	2	2	1	2	1
0	1	0	0	2	1	1	1	0
0	0	0	0	2	3	2	3	0
0	0	2	0	1	2	2	2	0
0	0	0	0	2	2	0	2	0
3	1	1	2	2	1	2	3	0

Figure 4. Transformed data for PCA

2.2.4 Error correction

There is an error for one of the “budget” columns where there were 2 options; “under 500” and “100-500”. Since these 2 options mean the same thing, they were either given the same numerical integer for PCA, or were concatenated into the same “one-hot” column when doing classification.

2.3 Data Mining

We used a multitude of data mining methods to study the data set provided to us. This section touches on the methods used and the intricacies with each different method.

2.3.1 Association Rule Mining - Apriori Algorithm

The Apriori algorithm was used to conduct association rule mining with the help of Jupyter Notebook. For association rule mining, all variables from the sheet were used as variables for mining to facilitate a blanket mining approach to determine the associations between all the different variables within the data set. This is a form of market basket analysis.

2.3.1.1 Key Metrics in Association Rule Mining

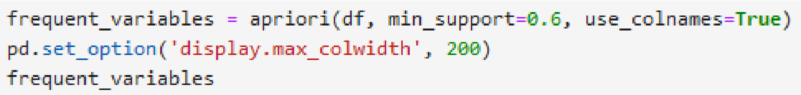
Lift: The lift value indicates to us how often the antecedent and consequent of a rule would occur together than if they were statistically independent. When lift value is higher than 1, it tells us that the antecedent and consequent appear together more often than would be expected by chance, which suggests a strong association between the antecedent and consequent. [2]

Support: The support value indicates to us the frequency of how often a rule’s antecedent and consequent appear together in a data set, it tells us the proportion of data in which both items occur. A high support value is a good indicator of whether or not the rule is applicable to a significant portion of the data set. [3]

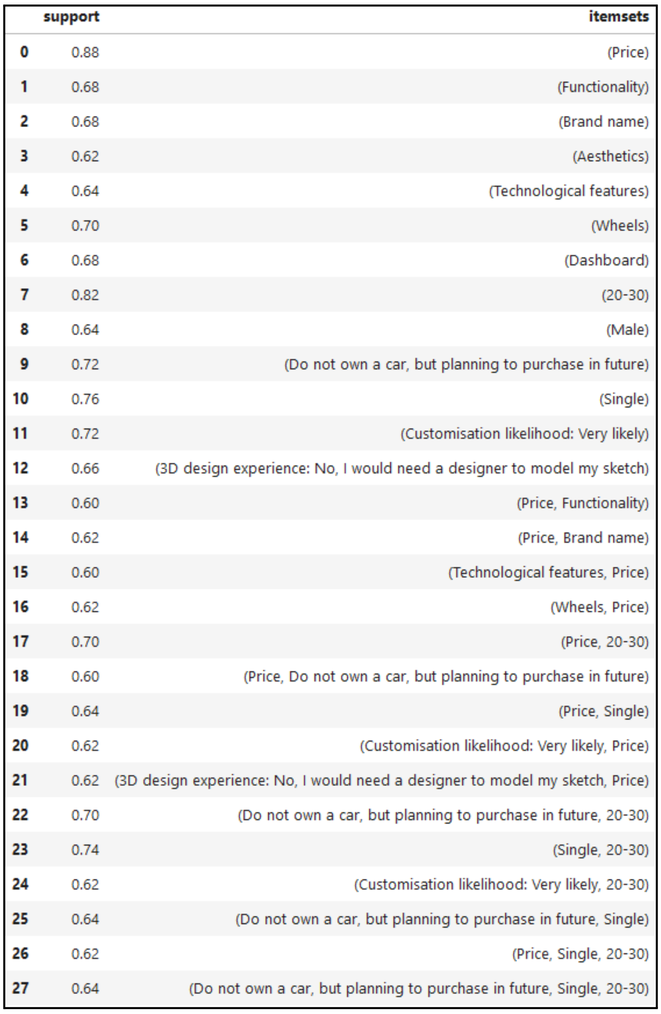
Confidence: The confidence value gives us a value to determine the reliability of a rule. It calculates the conditional probability of the consequent, given the antecedent. The numerical value indicates how often the rule holds true. Higher confidence value indicates a higher likelihood that the consequent will be present when the antecedent is also present. [3]

2.3.1.2 Frequent Itemset Raw Results:

The code below was used to generate the frequent itemset data. A support value of 0.6 was selected due to the high amount of variables we are working with after running the VBA code to split the multi-variable columns.



```
frequent_variables = apriori(df, min_support=0.6, use_colnames=True)
pd.set_option('display.max_colwidth', 200)
frequent_variables
```



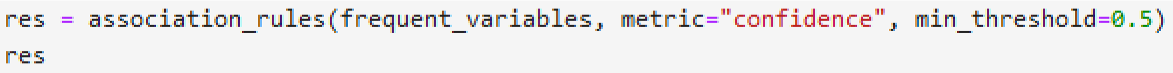
	support	itemsets
0	0.88	(Price)
1	0.68	(Functionality)
2	0.68	(Brand name)
3	0.62	(Aesthetics)
4	0.64	(Technological features)
5	0.70	(Wheels)
6	0.68	(Dashboard)
7	0.82	(20-30)
8	0.64	(Male)
9	0.72	(Do not own a car, but planning to purchase in future)
10	0.76	(Single)
11	0.72	(Customisation likelihood: Very likely)
12	0.66	(3D design experience: No, I would need a designer to model my sketch)
13	0.60	(Price, Functionality)
14	0.62	(Price, Brand name)
15	0.60	(Technological features, Price)
16	0.62	(Wheels, Price)
17	0.70	(Price, 20-30)
18	0.60	(Price, Do not own a car, but planning to purchase in future)
19	0.64	(Price, Single)
20	0.62	(Customisation likelihood: Very likely, Price)
21	0.62	(3D design experience: No, I would need a designer to model my sketch, Price)
22	0.70	(Do not own a car, but planning to purchase in future, 20-30)
23	0.74	(Single, 20-30)
24	0.62	(Customisation likelihood: Very likely, 20-30)
25	0.64	(Do not own a car, but planning to purchase in future, Single)
26	0.62	(Price, Single, 20-30)
27	0.64	(Do not own a car, but planning to purchase in future, Single, 20-30)

Figure 5. Raw Results from Frequent Itemset Mining

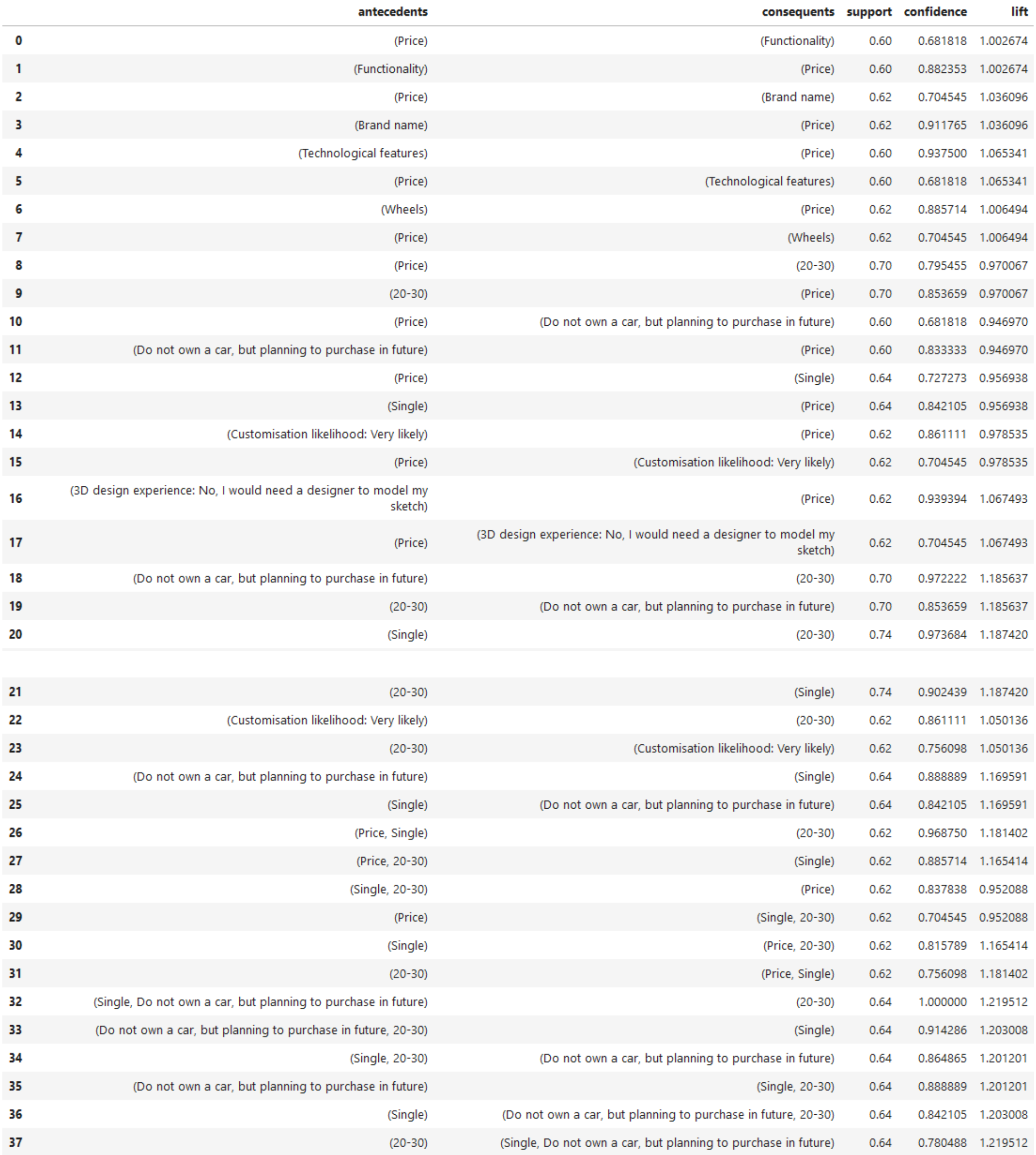
Before conducting full association rule mining, it is good practice to conduct frequent itemset mining on the data first. This would give us a better indicator of the possible antecedent and consequent combinations in association rule mining. By understanding these frequent item datasets, we can better focus on interesting associations found from association rule mining subsequently.

2.3.1.3 Association Rule Mining Raw Results

The code below was used to conduct association rule mining. Confidence was set to 0.5.



```
res = association_rules(frequent_variables, metric="confidence", min_threshold=0.5)
res
```



	antecedents	consequents	support	confidence	lift
0	(Price)	(Functionality)	0.60	0.681818	1.002674
1	(Functionality)	(Price)	0.60	0.882353	1.002674
2	(Price)	(Brand name)	0.62	0.704545	1.036096
3	(Brand name)	(Price)	0.62	0.911765	1.036096
4	(Technological features)	(Price)	0.60	0.937500	1.065341
5	(Price)	(Technological features)	0.60	0.681818	1.065341
6	(Wheels)	(Price)	0.62	0.885714	1.006494
7	(Price)	(Wheels)	0.62	0.704545	1.006494
8	(Price)	(20-30)	0.70	0.795455	0.970067
9	(20-30)	(Price)	0.70	0.853659	0.970067
10	(Price)	(Do not own a car, but planning to purchase in future)	0.60	0.681818	0.946970
11	(Do not own a car, but planning to purchase in future)	(Price)	0.60	0.833333	0.946970
12	(Price)	(Single)	0.64	0.727273	0.956938
13	(Single)	(Price)	0.64	0.842105	0.956938
14	(Customisation likelihood: Very likely)	(Price)	0.62	0.861111	0.978535
15	(Price)	(Customisation likelihood: Very likely)	0.62	0.704545	0.978535
16	(3D design experience: No, I would need a designer to model my sketch)	(Price)	0.62	0.939394	1.067493
17	(Price)	(3D design experience: No, I would need a designer to model my sketch)	0.62	0.704545	1.067493
18	(Do not own a car, but planning to purchase in future)	(20-30)	0.70	0.972222	1.185637
19	(20-30)	(Do not own a car, but planning to purchase in future)	0.70	0.853659	1.185637
20	(Single)	(20-30)	0.74	0.973684	1.187420
21	(20-30)	(Single)	0.74	0.902439	1.187420
22	(Customisation likelihood: Very likely)	(20-30)	0.62	0.861111	1.050136
23	(20-30)	(Customisation likelihood: Very likely)	0.62	0.756098	1.050136
24	(Do not own a car, but planning to purchase in future)	(Single)	0.64	0.888889	1.169591
25	(Single)	(Do not own a car, but planning to purchase in future)	0.64	0.842105	1.169591
26	(Price, Single)	(20-30)	0.62	0.968750	1.181402
27	(Price, 20-30)	(Single)	0.62	0.885714	1.165414
28	(Single, 20-30)	(Price)	0.62	0.837838	0.952088
29	(Price)	(Single, 20-30)	0.62	0.704545	0.952088
30	(Single)	(Price, 20-30)	0.62	0.815789	1.165414
31	(20-30)	(Price, Single)	0.62	0.756098	1.181402
32	(Single, Do not own a car, but planning to purchase in future)	(20-30)	0.64	1.000000	1.219512
33	(Do not own a car, but planning to purchase in future, 20-30)	(Single)	0.64	0.914286	1.203008
34	(Single, 20-30)	(Do not own a car, but planning to purchase in future)	0.64	0.864865	1.201201
35	(Do not own a car, but planning to purchase in future)	(Single, 20-30)	0.64	0.888889	1.201201
36	(Single)	(Do not own a car, but planning to purchase in future, 20-30)	0.64	0.842105	1.203008
37	(20-30)	(Single, Do not own a car, but planning to purchase in future)	0.64	0.780488	1.219512

Figure 6. Raw results from Association Rule Mining

The above are the raw results after conducting association rule mining. Comments on the results will be added in Chapter 4.

2.3.2 Principal Component Analysis

Principal Component Analysis (PCA) is an exploratory data analysis procedure that simplifies complex multi-dimensional data. This is achieved by manipulating original data into a set of new variables called the principal components which are uncorrelated and are ordered such that the first few components are able to retain most of the variation that is present in all of the original variables [4]. This enables us to concentrate on the key variables that have the most significant influence on other data variables.

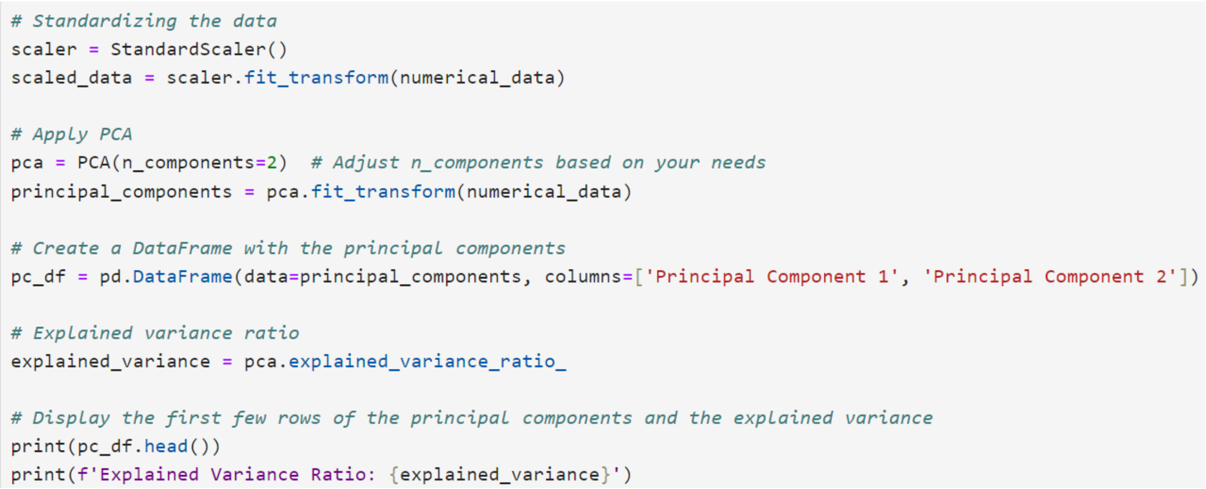
The steps for PCA are as follows. Firstly, standardisation takes place where data variables of different units and scales are normalised such that each variable equally contributes to the analysis, reducing bias placed towards variables with a higher magnitude. Covariance matrix computation then allows us to understand how variables vary from one another compared to the mean. PCA also finds the eigenvalues and eigenvectors of this covariance matrix which represent the magnitude of the variance for each principal component. The eigenvectors are then sorted by their eigenvalues in descending order and the number of principal components selected for analysis depends on the amount of covariance that is desired to be retained. Finally, the data is transferred to the coordinate system of the principal components [5].

2.3.2.1 Benefits of using Principal Component Analysis

PCA is a beneficial technique as it enables dimensional reduction, reducing the number of variables while also preserving most data. Furthermore, it enables noise reduction by keeping components with higher variance and neglecting the rest. PCA also allows for improved data visualisation as it lowers the principal components to 2 or 3 dimensional plot. This allows PCA to summarise complex data and capture its essence which reduces complexity and enables analysis to be more manageable without losing essential information. [4]

2.3.2.2 PCA on data

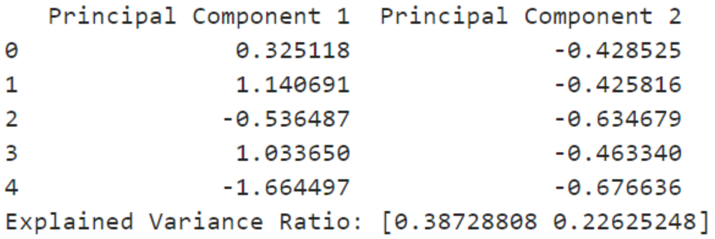
A pca.fit_tranform is first done on the numerical data and the explained variance ratio is obtained. The explained variance ratio provides a measure of how much of the information is captured by each principal component as shown in Figure 7.



```
# Standardizing the data
scaler = StandardScaler()
scaled_data = scaler.fit_transform(numerical_data)

# Apply PCA
pca = PCA(n_components=2)  # Adjust n_components based on your needs
principal_components = pca.fit_transform(numerical_data)

# Create a DataFrame with the principal components
pc_df = pd.DataFrame(data=principal_components, columns=['Principal Component 1', 'Principal Component 2'])

# Explained variance ratio
explained_variance = pca.explained_variance_ratio_

# Display the first few rows of the principal components and the explained variance
print(pc_df.head())
print(f'Explained Variance Ratio: {explained_variance}')
```

Figure 7. Initial code to prepare data for PCA

The following image shows the head of the DataFrame of the calculated Principle Components. The results of the explained variance ratio are also shown below (Figure 8).



```
   Principal Component 1  Principal Component 2
0               0.325118              -0.428525
1               1.140691              -0.425816
2              -0.536487              -0.634679
3               1.033650              -0.463340
4              -1.664497              -0.676636
Explained Variance Ratio: [0.38728808 0.22625248]
```

Figure 8. PCA results and explained variance ratio obtained

These PCA results are then used to plot the PCA for the dataset, allowing us to visualise the data in a scatter plot (Figure 9).

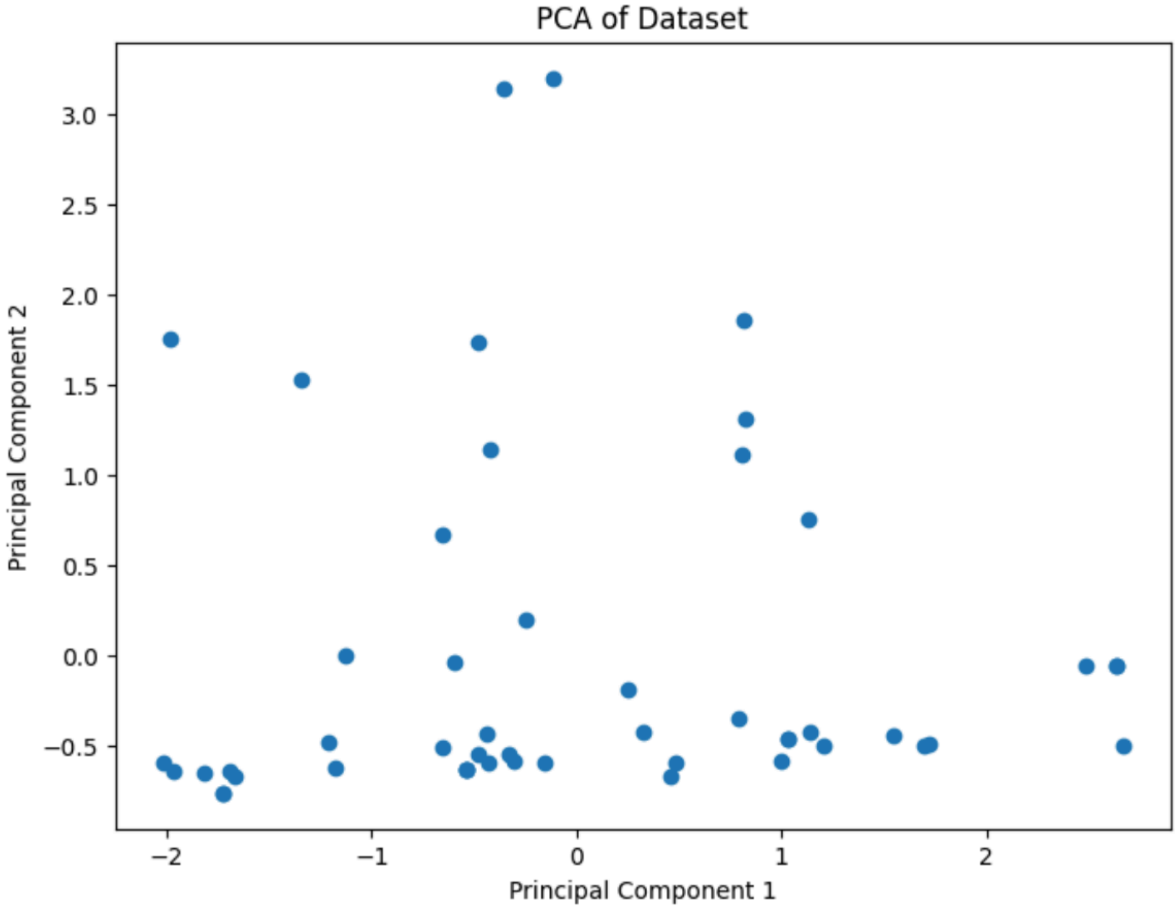


Figure 9. Scatter Plot of PCA Dataset

K-means clustering is then done on the data to discover hidden patterns in the complex dataset (Figure 10). K-means clustering works by grouping similar data points into clusters [6]. This is useful for exploratory data analysis, pattern recognition and simplifying complex datasets for further analysis. From the results obtained, three clusters are obtained and coloured accordingly. However no distinct clusters are visible. Therefore although K-means is used to find potential clusters, they are not significant for further analysis.

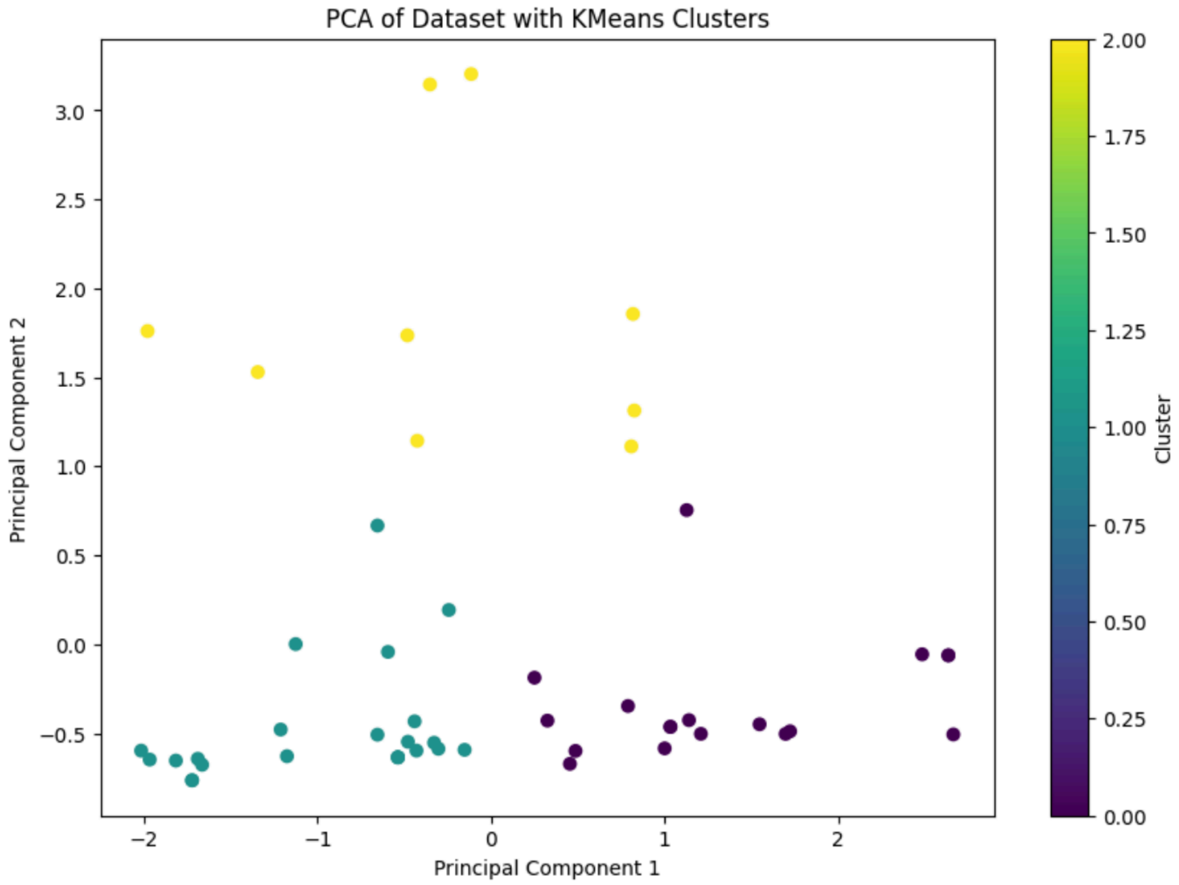


Figure 10. K-Means Cluster Coloured PCA Dataset

A PCA biplot was then done to visualise the loadings of the first two principal components to understand which variables contribute most to these principal components and therefore have the most influence. This can be seen from the length of the red lines which shows the magnitude of each variable from the origin (Figure 11).

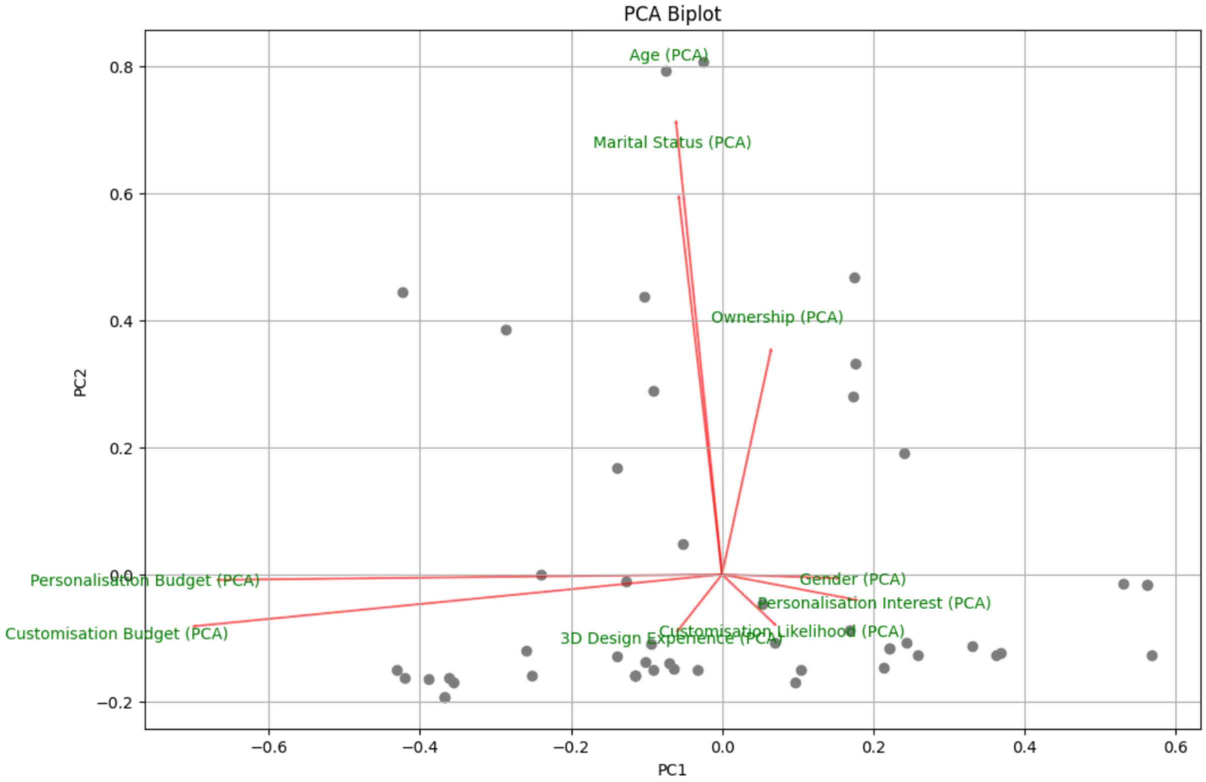


Figure 11. PCA Biplot results

Though the PCA biplot allows us to have a visual understanding of the variables that have the most impact on the data, a bar plot of the magnitudes of each variable on the Biplot was done to have a stronger understanding of the variables that cause the most significant impact that is ordered from the most to the least significant (Figure 12).

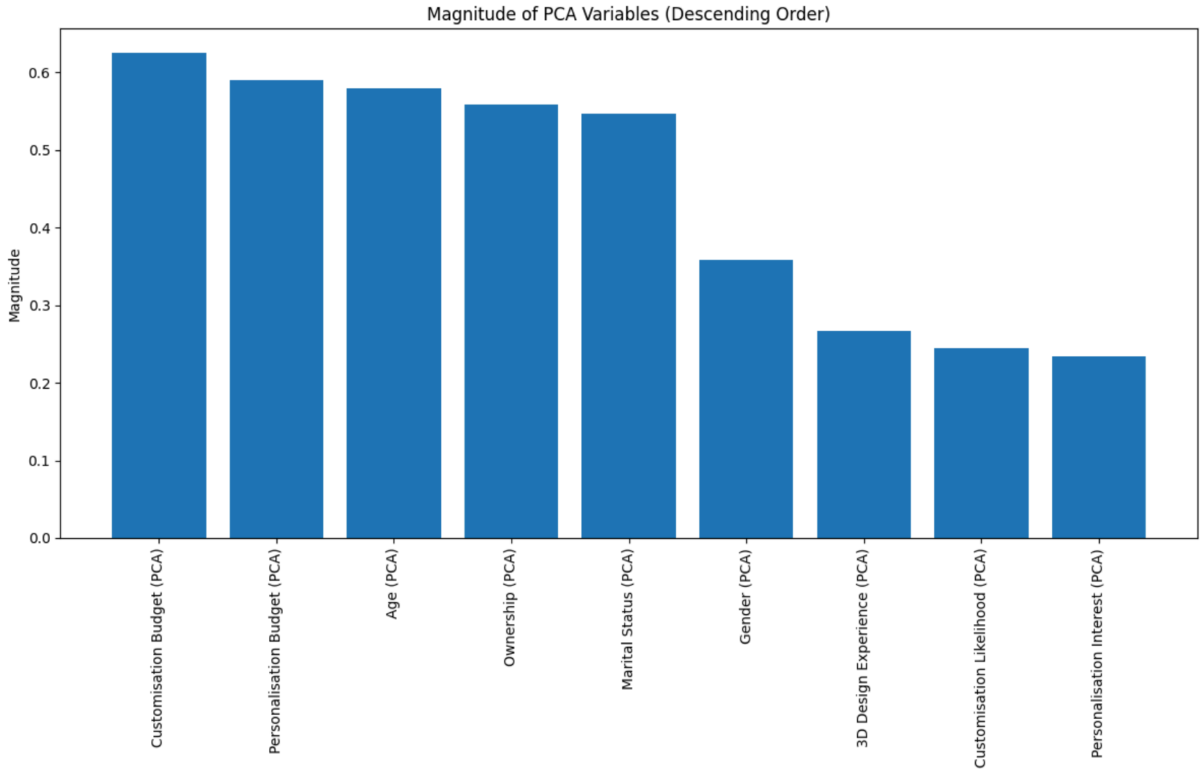
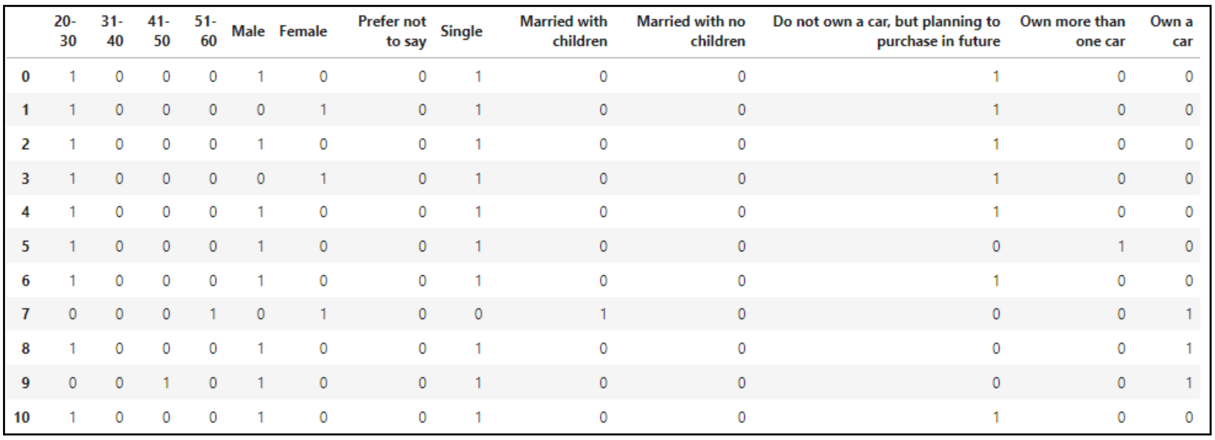


Figure 12. Magnitude of PCA Variables

This result shows us that ‘Customisation Budget’ has the most impact on the dataset with ‘Personalisation Interest’ having the least impact. This knowledge can now be used to choose the variables for comprehensive and in-depth analysis.

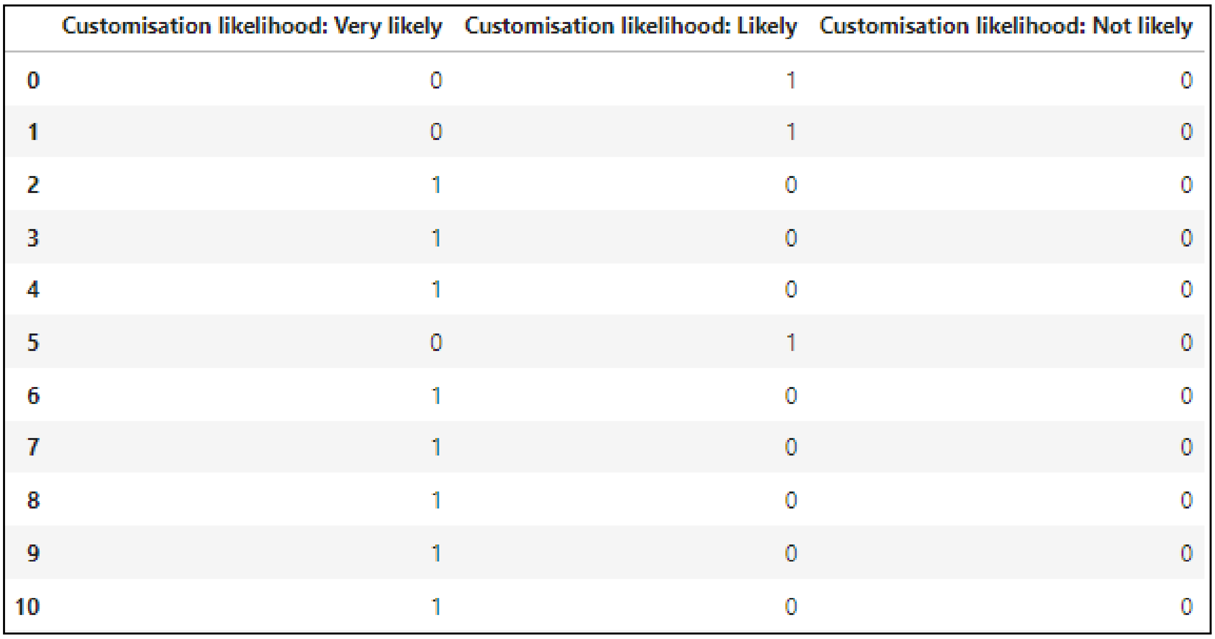
2.3.3 Classification

Classification is a useful data mining technique in analysing and interpreting results between features and the target variable [7]. In our case, PCA identified that demographics are the most impactful variables in the dataset. We will exclude customisation personalisation budget as those two are more akin to target variables. Classification offers a structured approach to further understand the effects between the demographics of the survey respondents and their willingness for car customisation. We will first partition the dependant variables (i.e. Age group, Gender, Marital status, Car ownership) from the target variable (i.e. Customisation Likelihood) into two separate dataframes X and y, respectively using the previously discussed one-hot encoded dataset.



	20-30	31-40	41-50	51-60	Male	Female	Prefer not to say	Single	Married with children	Married with no children	Do not own a car, but planning to purchase in future	Own more than one car	Own a car
0	1	0	0	0	1	0	0	1	0	0	1	0	0
1	1	0	0	0	0	1	0	1	0	0	1	0	0
2	1	0	0	0	1	0	0	1	0	0	1	0	0
3	1	0	0	0	0	1	0	1	0	0	1	0	0
4	1	0	0	0	1	0	0	1	0	0	1	0	0
5	1	0	0	0	1	0	0	1	0	0	0	1	0
6	1	0	0	0	1	0	0	1	0	0	1	0	0
7	0	0	0	1	0	1	0	0	1	0	0	0	1
8	1	0	0	0	1	0	0	1	0	0	0	0	1
9	0	0	1	0	1	0	0	1	0	0	0	0	1
10	1	0	0	0	1	0	0	1	0	0	1	0	0

Figure 13. Truncated X dependant variable dataframe



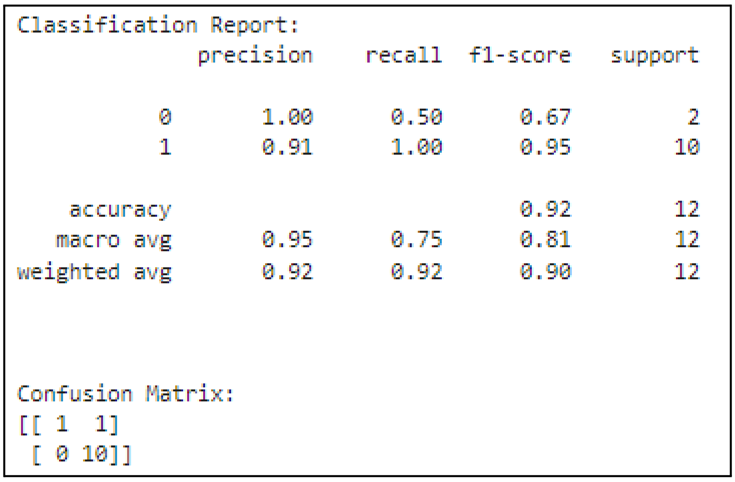
	Customisation likelihood: Very likely	Customisation likelihood: Likely	Customisation likelihood: Not likely
0	0	1	0
1	0	1	0
2	1	0	0
3	1	0	0
4	1	0	0
5	0	1	0
6	1	0	0
7	1	0	0
8	1	0	0
9	1	0	0
10	1	0	0

Figure 14. Truncated y target variable dataframe

However, certain options were chosen only once throughout the entire dataset, potentially causing them to be outliers and leading to an overfitted model. Overfitting occurs when the model is not able to generalise the data and fits too closely to the training data, often failing to see unseen data when tasked to predict on the test data. In the context of our dataset, only one respondent replied “Prefer not to say " as the chosen option for the question “What is your gender?”. Other similar instances include “Own more than one car” for “Do you own a car?” and “Not likely” for “How likely are you to customise your car?”. With such limited unique data points, it is challenging to classify them and train our model to do a proper prediction. To maintain the validity of our results, we opt to drop these 3 outliers from the original dataset.

2.3.3.1 Classification Report

Based on trial and error, we determined that a training-testing data split ratio of 75:25 yielded the highest accuracy score of 0.92. This implies that the trained model is capable of accurately predicting 92% of the survey respondent's customisation likelihood should the model be presented with new data. We then used the trained model to predict the customisation likelihood on the test dataset. The classification report and the confusion matrix are as follows:



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

           0       1.00      0.50      0.67         2
           1       0.91      1.00      0.95        10

    accuracy                           0.92        12
   macro avg       0.95      0.75      0.81        12
weighted avg       0.92      0.92      0.90        12



Confusion Matrix:
[[ 1  1]
 [ 0 10]]
```

Figure 15. Classification report of trained model

The confusion matrix shows that the model is able to accurately predict 11 out of 12 data points from the test dataset, which proves the relatively high accuracy score. True Positive (TP) and True Negative (TN) cells represent correctly predicted data points for class 1 (Very likely) and class 0 (Likely), respectively. In addition, the precision indicates the rate of positive predictions for each class. Recall on the other hand refers to the model's capability to detect positive data points. Ultimately, the F-1 score evens out both precision and recall by returning the harmonic mean and indicates the reliability of each class. Both classes 0 and 1 have a moderate to high F-1 score of 0.67 and 0.95 respectively, indicating that the model is capable of distinguishing between false and true predictions.

2.3.3.2 Decision Tree

We proceed to plot our model with a decision tree. This would allow us to visualise the decision making process of our model and understand how it separates the data into class 0 and 1. The annotated classification tree is shown in Figure below. All left branches represent the answer ‘no’ to the nodes and vice versa.

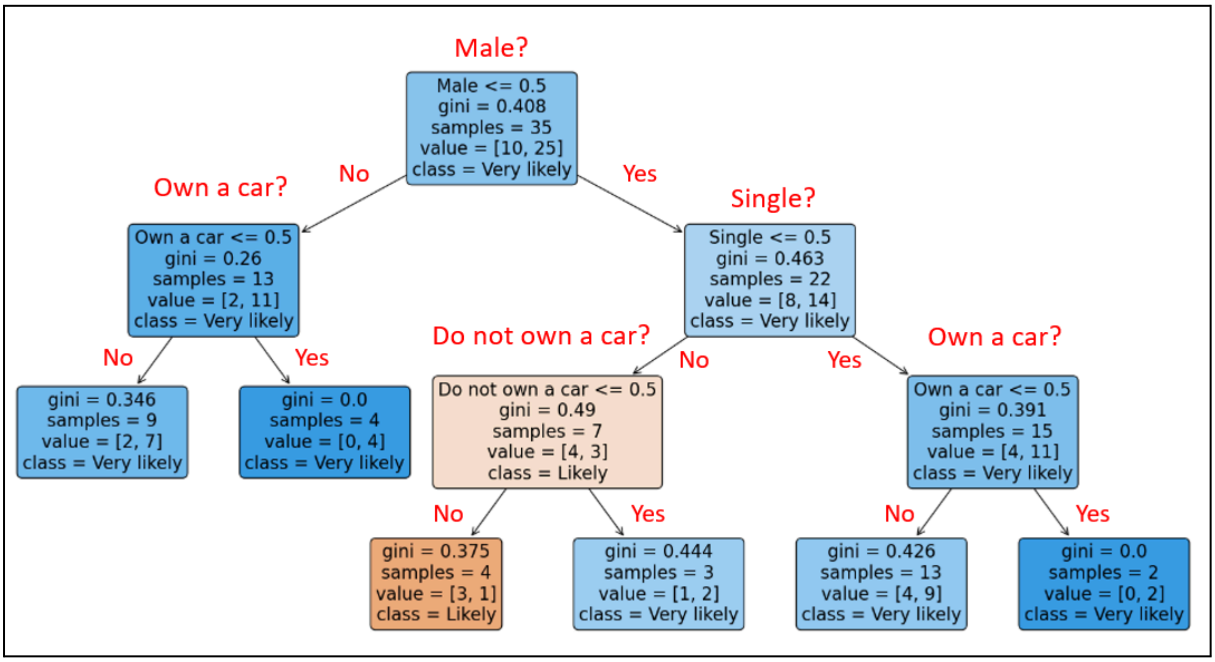


Figure 16. Trained model visualised with classification tree with annotations

By using the DecisionTreeClassifier() method from scikit-learn, we are able to visualise our model through a decision tree. Based on our dataframe of dependent and target variables, the DecisionTreeClassifier() method uses the information gain or the gini index algorithm taken as a parameter, to determine the attribute to further branch each node.

To maintain a balance of generalisation and model complexity, we restricted the tree's maximum depth to three, ensuring it captures essential patterns without becoming overly specific at the leaf nodes (an example of the overfitted tree can be found in Appendix C). However, it can be noticed from Figure 16 that some leaf nodes have the same class outcome as their sibling nodes. To streamline our classification tree and improve interpretability, we will prune it by tuning the Cost Complexity Pruning parameter (ccp_alpha). Pruning allows us to simplify the tree by removing leaf nodes that provide weak predictive power and is often used to prevent the model from over analysing the training dataset instead of searching for general patterns. A higher ccp_alpha value will result in a more pruned tree with fewer nodes.

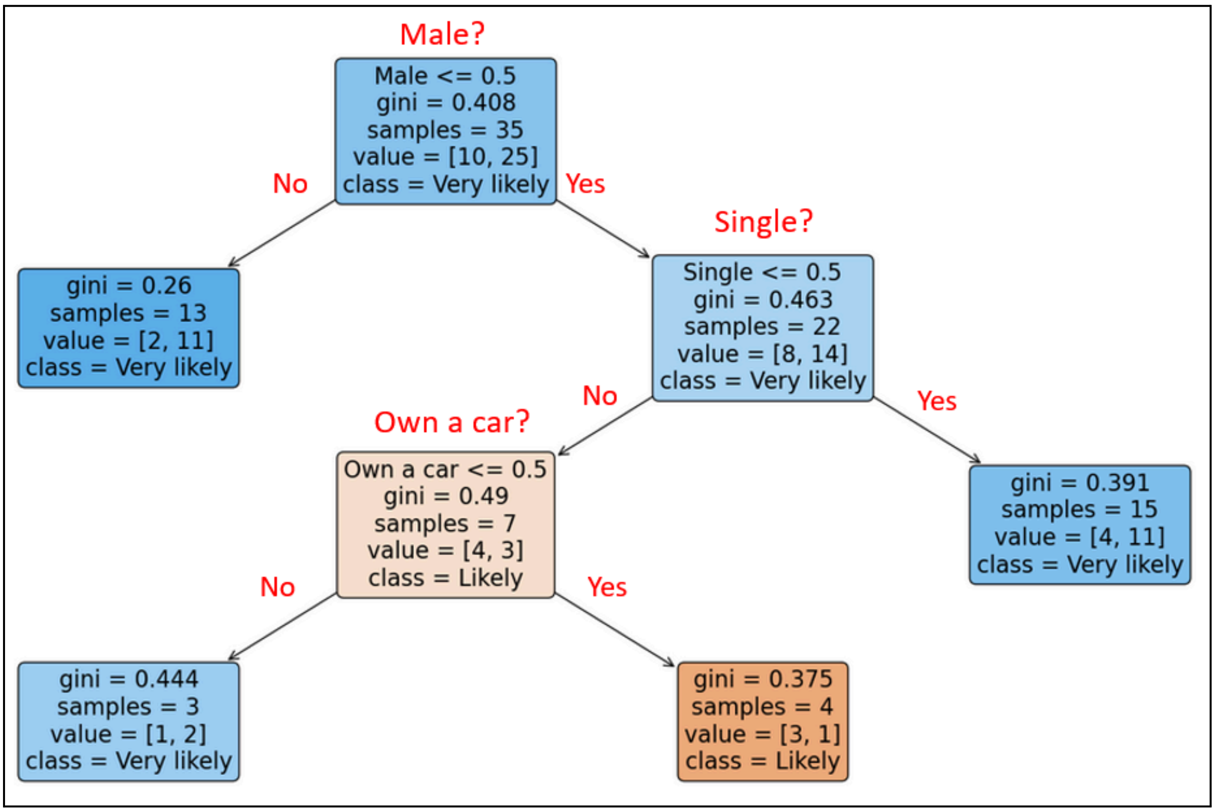


Figure 17. Pruned classification tree with annotations

By using a ccp_alpha of 0.01, the classification tree removes nodes with weak predictive power. For instance, in the original tree, further classification of non-males (i.e., females) based on car ownership results in both child nodes predicting "Very likely" customisation likelihood. Similarly, single males are subdivided by car ownership, yet both child nodes yield the same outcome. The pruned tree in Figure 17 removes those nodes. Consequently, the prediction indicates that if the respondent identifies as female, the likelihood of customising their car is deemed "Very likely."

Chapter 3: Results & Discussion

3.1 Association Rule Mining

As association rule mining is an unsupervised mining approach with a lot of output information, it is not possible to comment on all the insights that could be gained from it. This section will be going through the more interesting information we extracted.

3.1.1 Frequent Itemset Results

There are a few highly frequent individual items that we can see from our frequent itemsets analysis, notably, “Price” and the age group “20-30” were the highest with their supports being 0.88 and 0.82 respectively. This indicates that most of our survey respondents take “Price” into consideration and most of them were also between the age of 20 to 30.

When looking at item pairs, “singles” in their “20-30”s have the highest support at 0.74. This does not necessarily give us insight to car purchase considerations but it still serves as a good indicator that the data is accurate as it makes logical sense that younger people would tend to be single when compared to older people. There are also other item pairs with relatively high support values that indicate to us what customers might be looking out for, such as, “Price” & “Brand Name”, “Technological Features” & “Price”, etc.

3.1.2 Association Rule Mining Results

A triple association rule that has a high support value of 0.64 is “Do not own a car,but planning to purchase in the future”, “Single” & “20-30”. Although this does not give direct insight into a customer's buying habits, it gives further confirmation that the data is accurate as it suggests that young people who are single will probably not own a car. Furthermore, it gives us useful insights into the majority demographic of respondents in the survey.

From the data we can see that many rules have bidirectional associations, this tells us that the relationship between the items are symmetric and also reinforces the relationships between items as both directions are found to have high confidence values. Some examples of bidirectional pairings are; “Price” & “Functionality” and “Price” & “Brand name”.

Another insight is that certain demographic factors may strongly influence purchase decisions. For example, “Price” and “Single” have high confidence and support values bidirectionally. This indicates that singles are more likely to consider price when making purchasing decisions. “20-30” also shows high confidence and support values with “Price” bidirectionally.

Lastly, another pairing, “20-30” & “Customisation likelihood: Very likely”, has high support and confidence values, telling us that younger people are more likely to want to customise their vehicles.

3.2 PCA & Classification

From our PCA and classification results, the decision tree begins by splitting the dataset with the attribute gender, with females branching to the left and males to the right. Notably, based on the data from our dataset, our model predicts that most females are very likely to customise their cars, with a gini impurity of 0.26, indicating that most females fall into the “Very likely” class. This segmentation serves as an indicator that there is a relationship between gender and customization likelihood in our dataset. Although these results may not be representative of females in general outside of the dataset, it is to note that our dataset does not have further attributes that could distinctively segment females into the classes, as can be seen in our non-constricted decision tree in Appendix C. Given that PCA has also detected that demographics served as the most impactful variables of our dataset, more data points involving females with various likelihood responses are required to further segment them beyond just their gender.

The right side of the node representing the males further branches out, indicating that there is a more balanced mix of males that fall in either classes, thus using additional attributes to split the subdataset. The algorithm detects marital status as the optimal attribute to segment the male subdataset, showing that single males are “Very likely” to customise their cars. While this is plausible due to single males having fewer commitments, it is still tough to draw conclusive evidence due to the small sample size. On the other hand, married males are further segmented based on their car ownership, showing that married males who do not own a car are “Very likely” while those that do own a car are just “Likely” to customise their car. To an extent, this result is plausible as it might suggest that males that do not own a car might find it tougher to estimate the cost of customisation while those that do own a car find it easier. Again, given the small sample size, this insight may be sheer coincidence and not completely factual.

3.3 Limitations of the Study

One limitation of our model is its inability to distinguish between respondents categorised as 'Very likely', 'Likely', and 'Not likely' to customise their cars, due to the model only having two classification classes. This is a result of us dropping the only datapoint that indicated ‘Not likely’ as their customisation likelihood to mitigate overfitting. To solve this limitation, more data from respondents that share the same demographic (i.e. ‘Not likely’) will be needed for a comprehensive and nuanced model capable of classifying a third class within the dataset.

Chapter 4: Conclusion

In conclusion, our analysis revealed valuable insights into customer preferences and purchase behaviour. Association rule mining highlighted key factors influencing decisions, such as price considerations and demographic groups. PCA and classification results detailed gender and marital status’s influence on customisation likelihood. Overall, these findings provide us with several valuable insights from our provided dataset.

Word Count: 3508 words

References

[1] J. Brownlee, "Why One-Hot Encode Data in Machine Learning: Benefits and Implementation," Machine Learning Mastery, [Online]. Available: https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/. [Accessed: February 2, 2024].

[2] Data Overload, "Discovering Patterns in Data: The Importance of Lift in Association Rule Mining," Medium, Available: https://medium.com/@data-overload/discovering-patterns-in-data-the-importance-of-lift-in-association-rule-mining-3ae63de9dc79. [Accessed: February 3, 2024].

[3] TechTarget, "Association rules in data mining," SearchBusinessAnalytics, Available: https://www.techtarget.com/searchbusinessanalytics/definition/association-rules-in-data-mining. [Accessed: February 3, 2024].

[4] Built In, "Step-by-Step Explanation of Principal Component Analysis," Available: https://builtin.com/data-science/step-step-explanation-principal-component-analysis. [Accessed: February 5, 2024].

[5] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction (Second Edition)*. Beijing: Shi jie tu shu chu ban gong si, 2023.

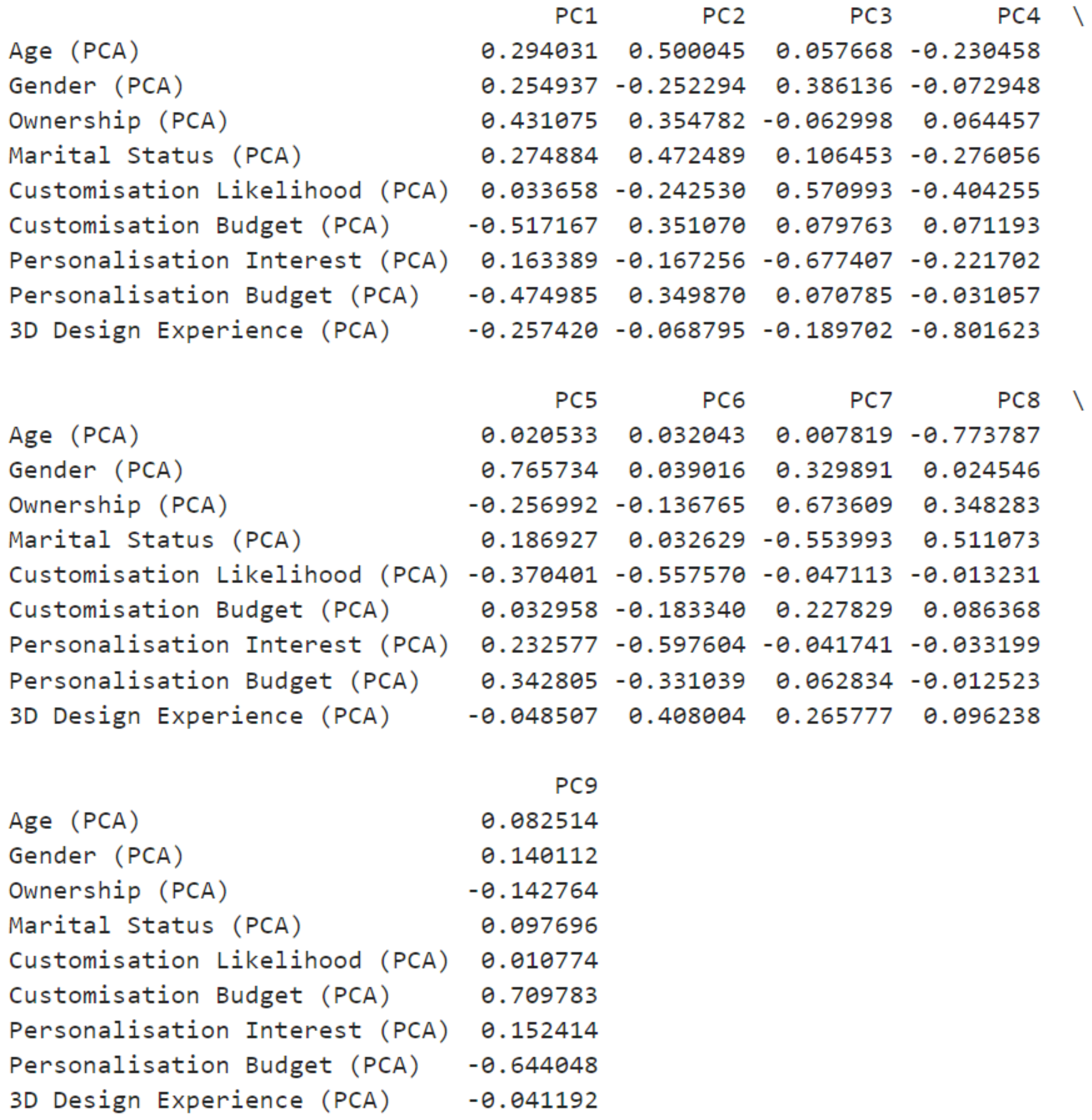
[6] N. Sharma, “K-means clustering explained,” neptune.ai, https://neptune.ai/blog/k-means-clustering#:~:text=It%20is%20an%20iterative%20process,data%20point%20should%20belong%20to. (accessed Feb. 15, 2024).

[7] GeeksforGeeks, "Basic Concept of Classification in Data Mining," Available: https://www.geeksforgeeks.org/basic-concept-classification-data-mining/. [Accessed: February 5, 2024].

Appendices

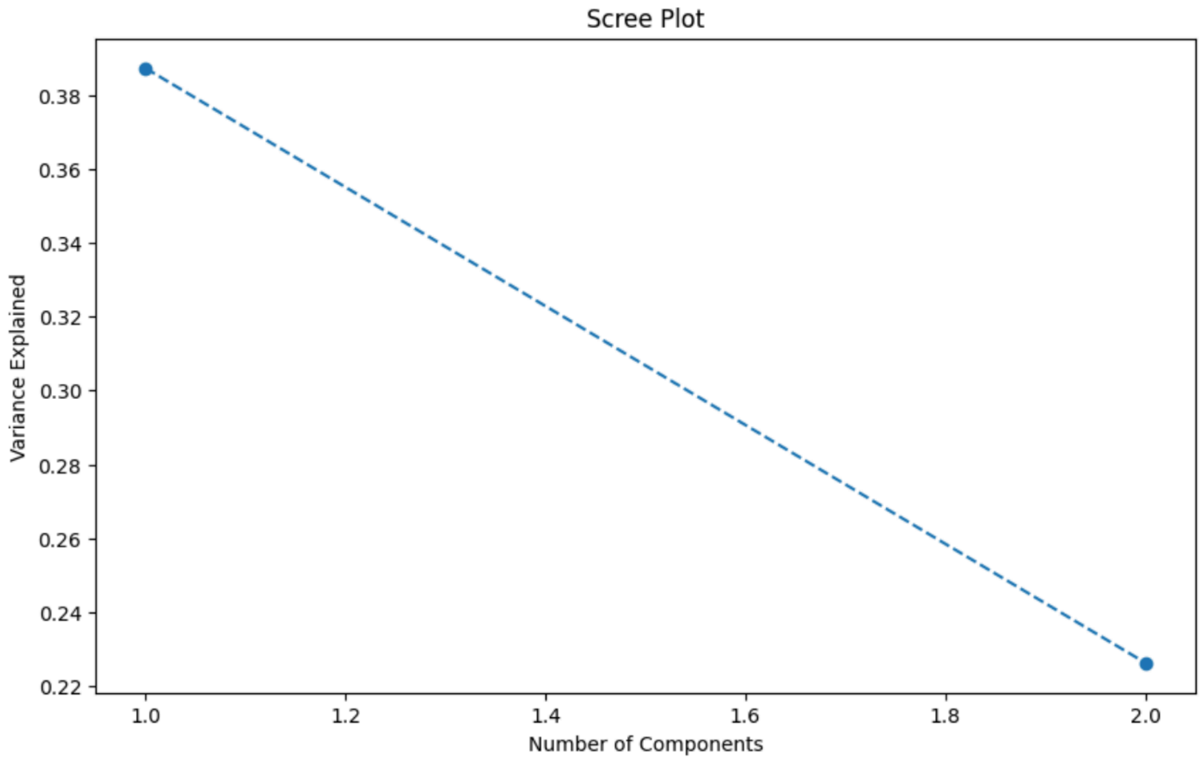
Appendix A

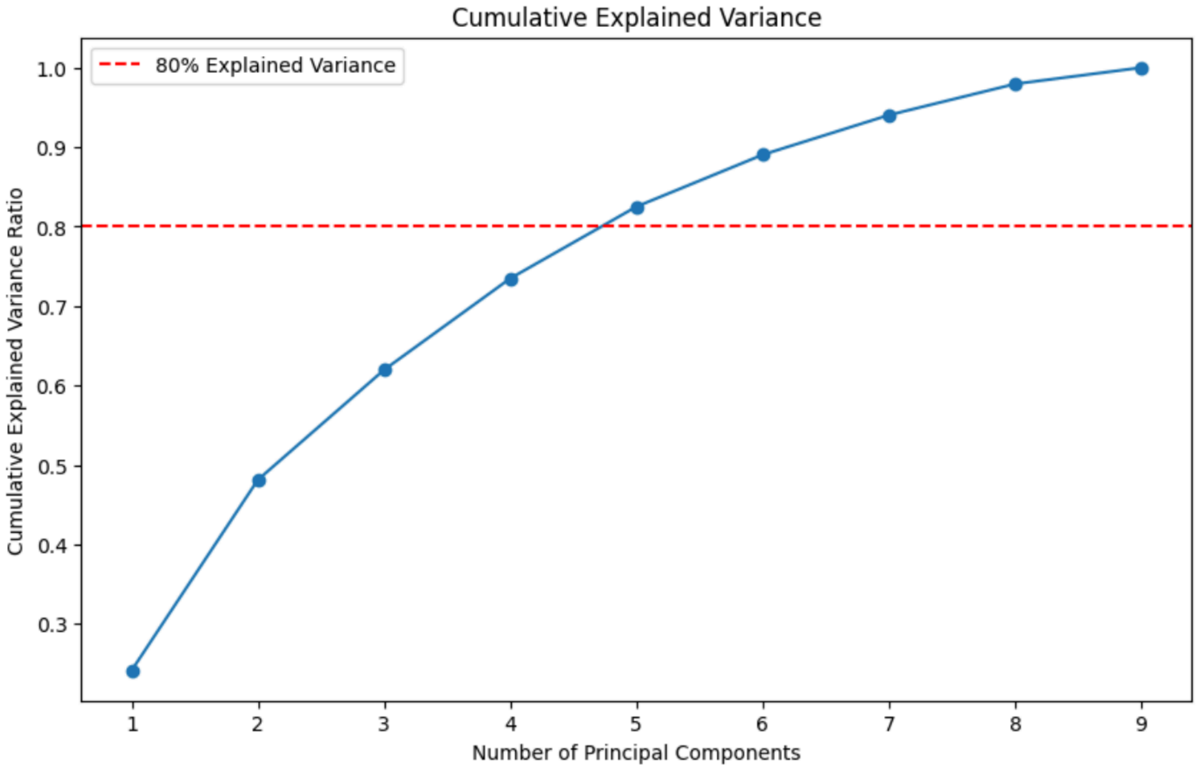
A list of all the PCs are listed as shown below.



```
                                    PC1       PC2       PC3       PC4  \
Age (PCA)                      0.294031  0.500045  0.057668 -0.230458
Gender (PCA)                   0.254937 -0.252294  0.386136 -0.072948
Ownership (PCA)                0.431075  0.354782 -0.062998  0.064457
Marital Status (PCA)           0.274884  0.472489  0.106453 -0.276056
Customisation Likelihood (PCA) 0.033658 -0.242530  0.570993 -0.404255
Customisation Budget (PCA)    -0.517167  0.351070  0.079763  0.071193
Personalisation Interest (PCA) 0.163389 -0.167256 -0.677407 -0.221702
Personalisation Budget (PCA)  -0.474985  0.349870  0.070785 -0.031057
3D Design Experience (PCA)    -0.257420 -0.068795 -0.189702 -0.801623

                                    PC5       PC6       PC7       PC8  \
Age (PCA)                      0.020533  0.032043  0.007819 -0.773787
Gender (PCA)                   0.765734  0.039016  0.329891  0.024546
Ownership (PCA)               -0.256992 -0.136765  0.673609  0.348283
Marital Status (PCA)           0.186927  0.032629 -0.553993  0.511073
Customisation Likelihood (PCA) -0.370401 -0.557570 -0.047113 -0.013231
Customisation Budget (PCA)     0.032958 -0.183340  0.227829  0.086368
Personalisation Interest (PCA) 0.232577 -0.597604 -0.041741 -0.033199
Personalisation Budget (PCA)   0.342805 -0.331039  0.062834 -0.012523
3D Design Experience (PCA)    -0.048507  0.408004  0.265777  0.096238

                                    PC9
Age (PCA)                      0.082514
Gender (PCA)                   0.140112
Ownership (PCA)               -0.142764
Marital Status (PCA)           0.097696
Customisation Likelihood (PCA) 0.010774
Customisation Budget (PCA)     0.709783
Personalisation Interest (PCA) 0.152414
Personalisation Budget (PCA)  -0.644048
3D Design Experience (PCA)    -0.041192
```

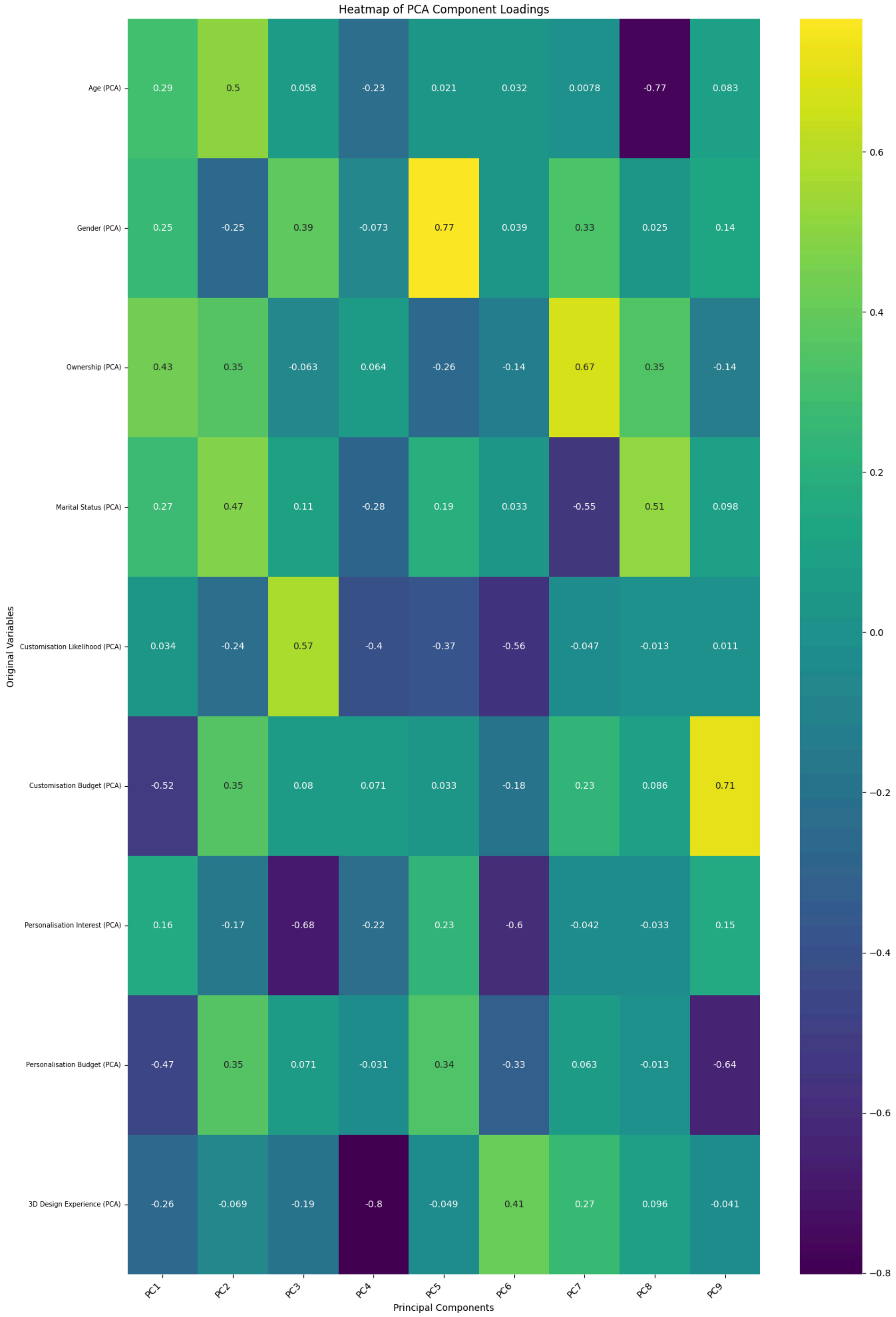
A Scree chart and a cumulative explained variance vs number of principal components chart is then plotted where it is found that there are 5 PCs that are able to explain 80% variance. This result can be seen in the charts below.



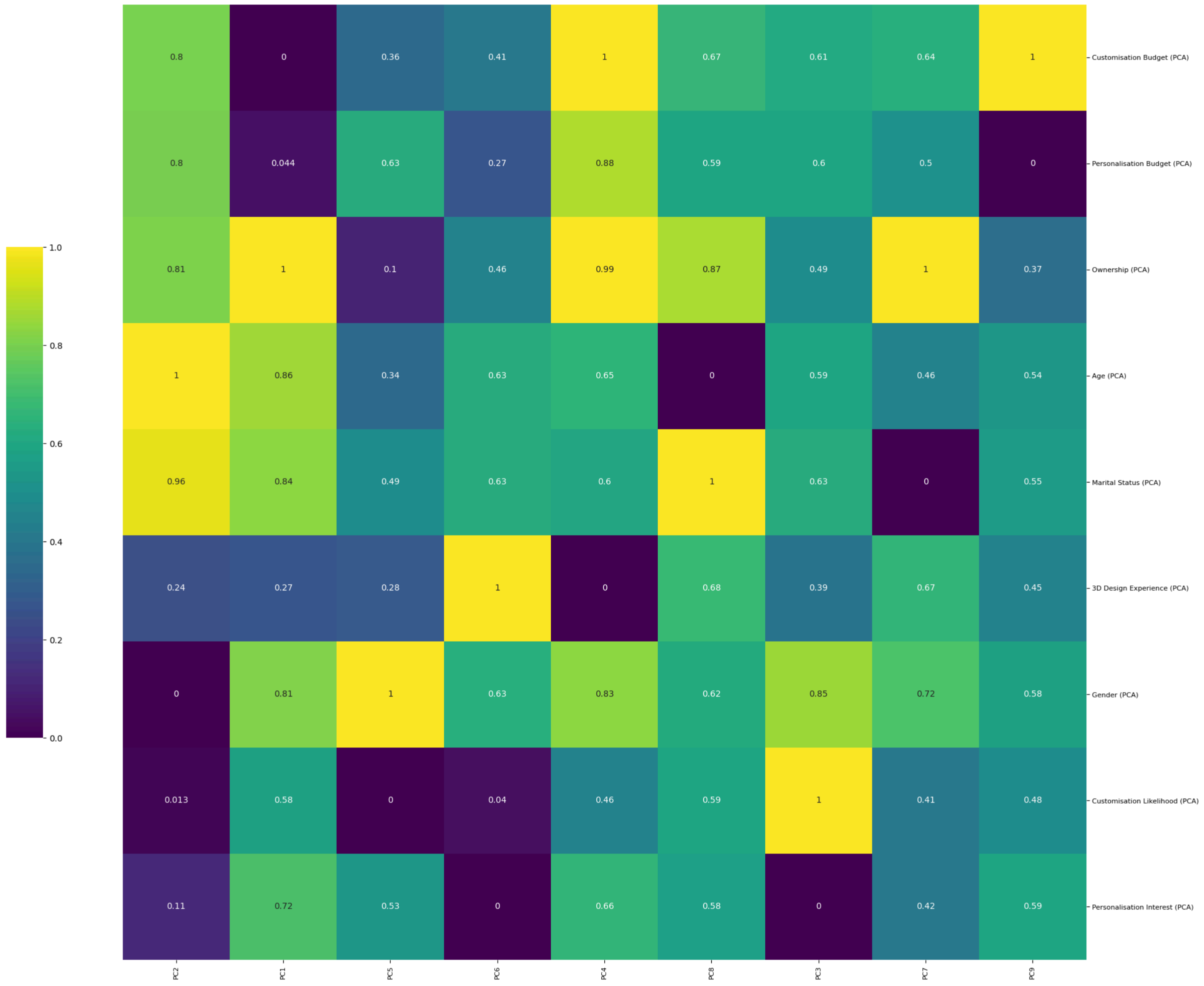


To have a better understanding of the data, a heatmap is done of the component loadings to visualise the weights of the loadings on the first 10 principal components. This is done as an exploratory step to discover more information about the first 10 principal components instead of the first 2. A high absolute value indicates variables that significantly contribute to the component. This means that the variable is strongly associated with that component's variance. This visualisation is therefore used in identifying which original variables are the most influential in defining each principal variable.

Appendix B

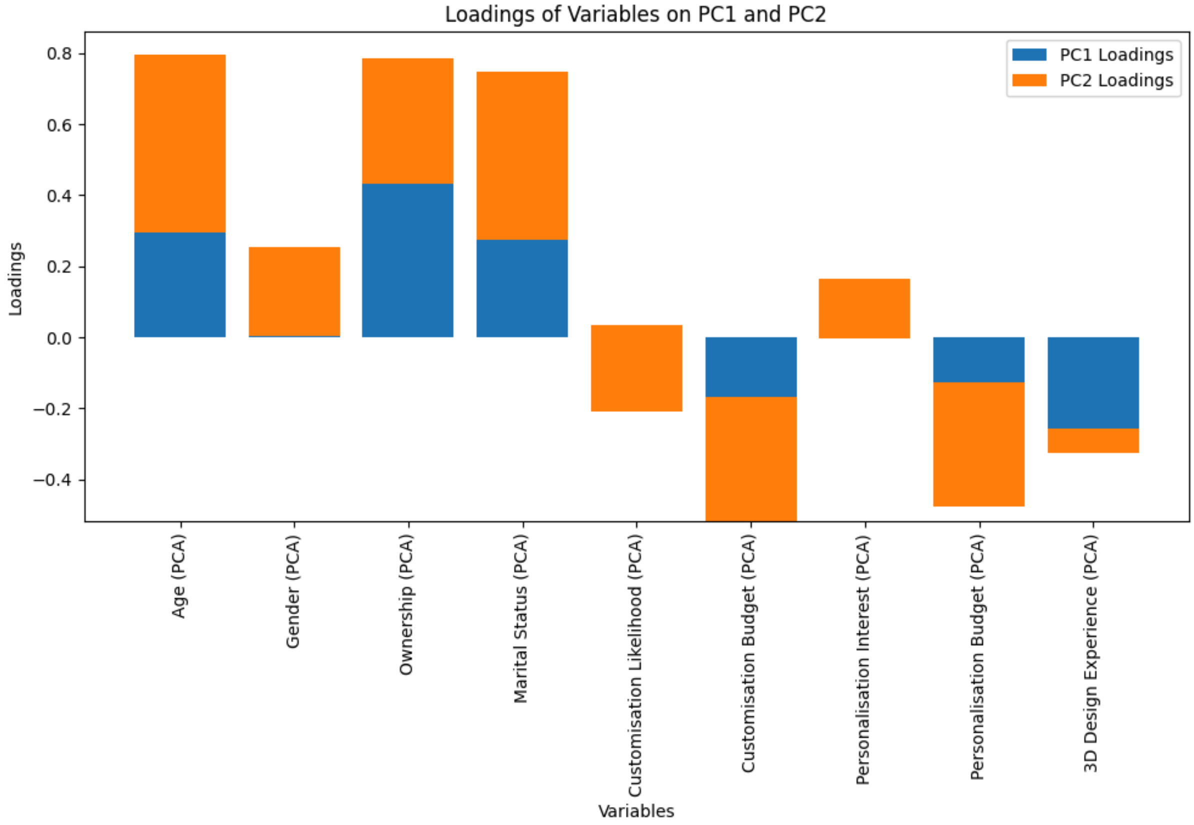

Heatmap of PCA Component Loadings
Original Variables
Age (PCA)
0.29
0.5
0.058
-0.23
0.021
0.032
0.0078
-0.77
0.083
Gender (PCA)
0.25
-0.25
0.39
-0.073
0.77
0.039
0.33
0.025
0.14
Ownership (PCA)
0.43
0.35
-0.063
0.064
-0.26
-0.14
0.67
0.35
-0.14
Marital Status (PCA)
0.27
0.47
0.11
-0.28
0.19
0.033
-0.55
0.51
0.098
Customisation Likelihood (PCA)
0.034
-0.24
0.57
-0.4
-0.37
-0.56
-0.047
-0.013
0.011
Customisation Budget (PCA)
-0.52
0.35
0.08
0.071
0.033
-0.18
0.23
0.086
0.71
Personalisation Interest (PCA)
0.16
-0.17
-0.68
-0.22
0.23
-0.6
-0.042
-0.033
0.15
Personalisation Budget (PCA)
-0.47
0.35
0.071
-0.031
0.34
-0.33
0.063
-0.013
-0.64
3D Design Experience (PCA)
-0.26
-0.069
-0.19
-0.8
-0.049
0.41
0.27
0.096
-0.041
PC1
PC2
PC3
PC4
PC5
PC6
PC7
PC8
PC9
Principal Components
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8

A cluster map analysis is then done which provides a heatmap combined with hierarchical clustering on both the variables and principal components. It groups variables and components that show similar patterns of loadings. By examining the colour intensity, we can identify clusters of variables that contribute similar to the principal components. This helps in understanding the relationships among the variables and identify potential underlying patterns in the data.



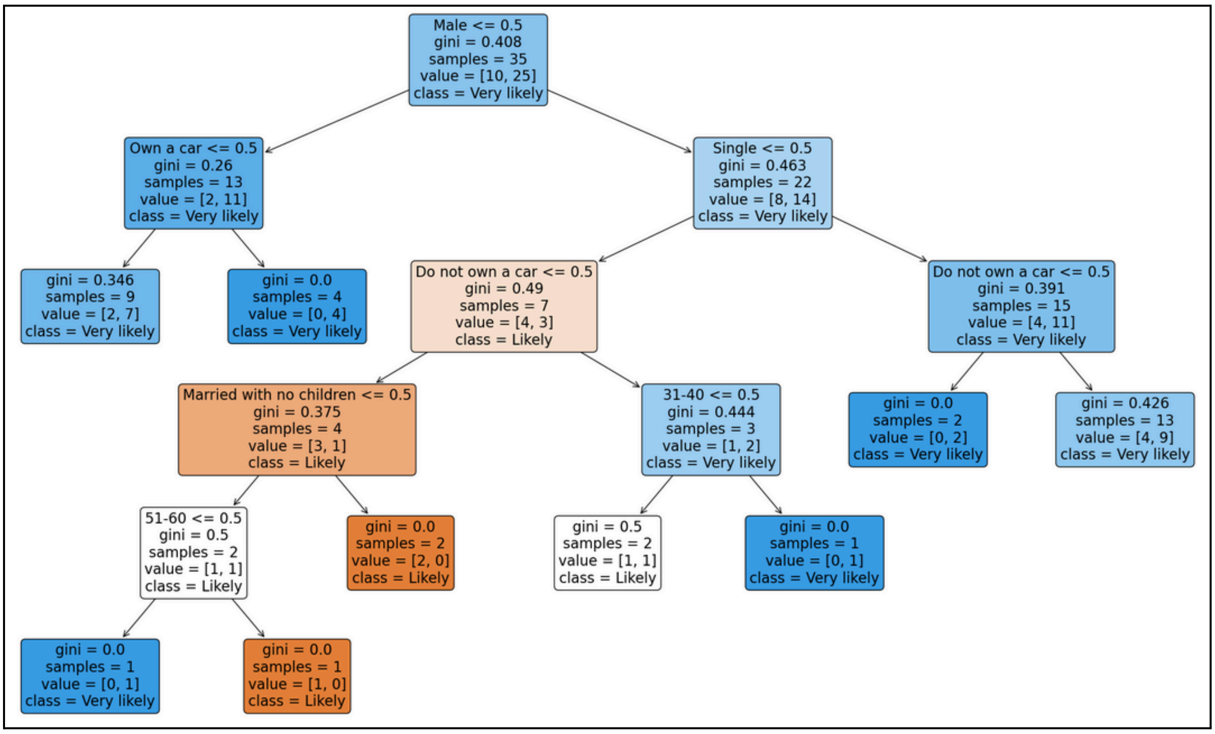
	PC2	PC1	PC5	PC6	PC4	PC8	PC3	PC7	PC9
Customisation Budget (PCA)	0.8	0	0.36	0.41	1	0.67	0.61	0.64	1
Personalisation Budget (PCA)	0.8	0.044	0.63	0.27	0.88	0.59	0.6	0.5	0
Ownership (PCA)	0.81	1	0.1	0.46	0.99	0.87	0.49	1	0.37
Age (PCA)	1	0.86	0.34	0.63	0.65	0	0.59	0.46	0.54
Marital Status (PCA)	0.96	0.84	0.49	0.63	0.6	1	0.63	0	0.55
3D Design Experience (PCA)	0.24	0.27	0.28	1	0	0.68	0.39	0.67	0.45
Gender (PCA)	0	0.81	1	0.63	0.83	0.62	0.85	0.72	0.58
Customisation Likelihood (PCA)	0.013	0.58	0	0.04	0.46	0.59	1	0.41	0.48
Personalisation Interest (PCA)	0.11	0.72	0.53	0	0.66	0.58	0	0.42	0.59

Finally, we generate a bar plot that visualises the loadings of variables on the first two principal components which allows us to visualise a direct comparison of the key variables that significantly influence these components. It allows us to understand the underlying structure of the dataset and highlights the variables with the most substantial impact on the dataset’s variance.



Loadings of Variables on PC1 and PC2
PC1 Loadings
PC2 Loadings
Loadings
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
Age (PCA)
Gender (PCA)
Ownership (PCA)
Marital Status (PCA)
Customisation Likelihood (PCA)
Customisation Budget (PCA)
Personalisation Interest (PCA)
Personalisation Budget (PCA)
3D Design Experience (PCA)
Variables

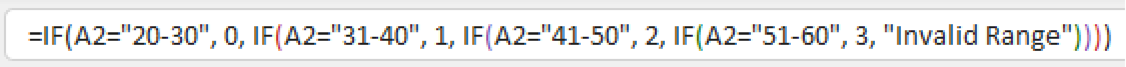
Appendix C



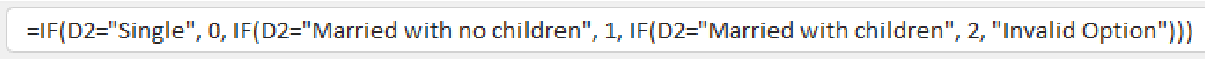
For the overfitted tree, we did not limit the max depth and allow scikit-learn to produce a decision tree without constraints. This results in an overly complex tree with many attributes. It is important to note that each attribute should distinctively split the subset into separate classes as much as possible. Additionally, the more attributes there are, the higher the variance of the model will be. This results in overfitting, as the model focuses on the noise of the training data, losing generalisation and resulting in poor predictions.

Appendix D

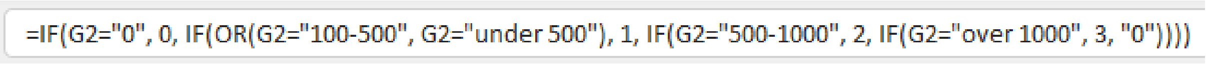
Below are examples of Formulas used in Excel for integer encoding for PCA.



```
=IF(A2="20-30", 0, IF(A2="31-40", 1, IF(A2="41-50", 2, IF(A2="51-60", 3, "Invalid Range"))))
```

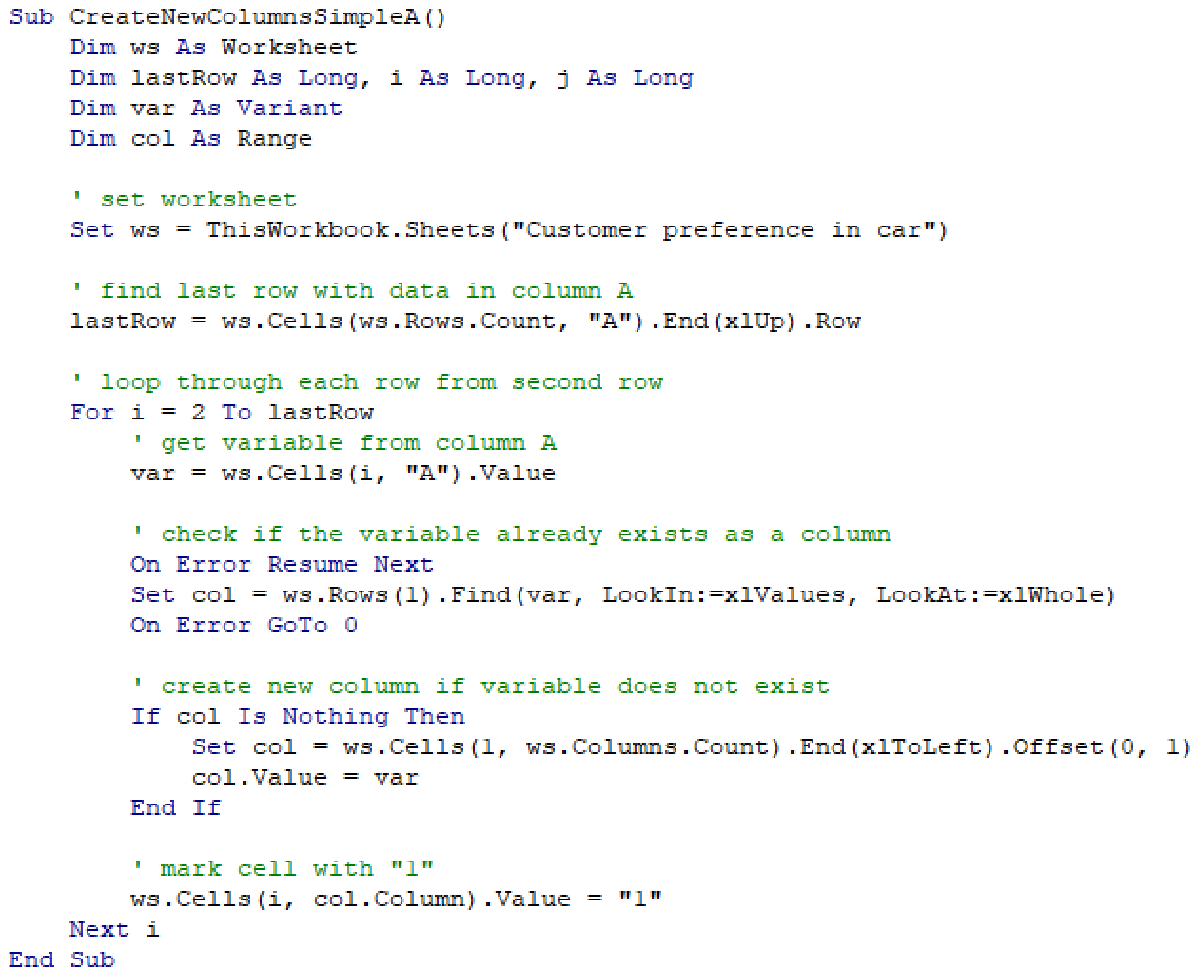


```
=IF(D2="Single", 0, IF(D2="Married with no children", 1, IF(D2="Married with children", 2, "Invalid Option")))
```



```
=IF(G2="0", 0, IF(OR(G2="100-500", G2="under 500"), 1, IF(G2="500-1000", 2, IF(G2="over 1000", 3, "0"))))
```

Below is VBA code used for one-hot encoding for columns without multiple variables.



```
Sub CreateNewColumnsSimpleA()
    Dim ws As Worksheet
    Dim lastRow As Long, i As Long, j As Long
    Dim var As Variant
    Dim col As Range

    ' set worksheet
    Set ws = ThisWorkbook.Sheets("Customer preference in car")

    ' find last row with data in column A
    lastRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).Row

    ' loop through each row from second row
    For i = 2 To lastRow
        ' get variable from column A
        var = ws.Cells(i, "A").Value

        ' check if the variable already exists as a column
        On Error Resume Next
        Set col = ws.Rows(1).Find(var, LookIn:=xlValues, LookAt:=xlWhole)
        On Error GoTo 0

        ' create new column if variable does not exist
        If col Is Nothing Then
            Set col = ws.Cells(1, ws.Columns.Count).End(xlToLeft).Offset(0, 1)
            col.Value = var
        End If

        ' mark cell with "1"
        ws.Cells(i, col.Column).Value = "1"
    Next i
End Sub
```