GERMAN CREDIT DATA: KMEANS CLUSTERING

Statistical Learning, Deep Learning and Artificial Intelligence by Prof. Silvia Salini

Tejaswini Yadav Nakka - 06651A

A.A. 2022-2023

Customer segmentation plays a crucial role in the banking industry to better understand and cater to the diverse needs of customers. This project focuses on using the power of machine learning by implementing the K-Means Clustering Algorithm for Bank credit Customer Segmentation.Through this project, we will perform customer segmentation using R Studio, a popular statistical software.We will perform a descriptive analysis on the German Credit Data.This descriptive analysis will provide us with valuable insights into the characteristics of the customers and their preferences.

By understanding the credit requirements and behaviors of different segments, banks can design customized loan products, credit cards, and financial solutions that align with the specific needs and risk profiles of each segment.

Contents

1 Introduction

In recent years, with the availability of vast amounts of customer data and advancements in machine learning and data analytics, banks have been using sophisticated techniques like K-Means Clustering to perform customer segmentation more accurately and efficiently.

K-Means Clustering is a popular unsupervised learning algorithm that groups similar data points into clusters based on their similarities and differences. By applying this algorithm to bank credit customer data, banks can uncover hidden patterns and relationships, leading to more robust and actionable segmentation results.

Throughout the project, we will employ various statistical and visualization techniques to analyze and present the results in a comprehensive manner. This will aid in the interpretation of the segmentation outcomes and facilitate informed decision-making.

2 Methodology

All the analysis required for the unsupervised learning has been done in R programming.

- Exploratory Data Analysis
- Relationship Between Variables
- Visualisation of Data;
- Unsupervised Machine Learning
- Methods for finding optimal number of clusters

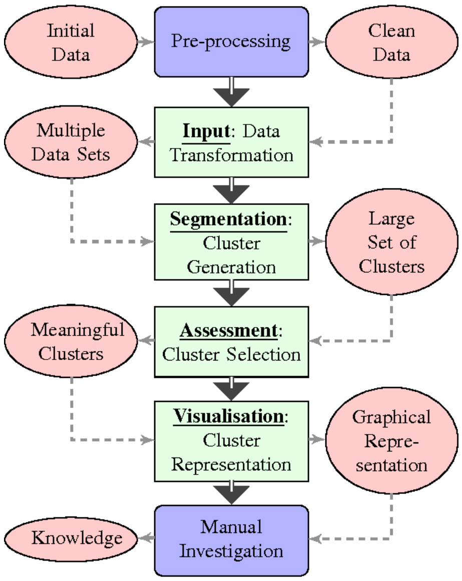

Initial Data
Pre-processing
Clean Data
Multiple Data Sets
Input: Data Transformation
Segmentation: Cluster Generation
Large Set of Clusters
Meaningful Clusters
Assessment: Cluster Selection
Visualisation: Cluster Representation
Graphical Repre-sentation
Knowledge
Manual Investigation

Figure 1: Analysis of Customer Segmentation

2.1 Usage of Unsupervised Learning

Unsupervised learning is a powerful branch of machine learning that plays a crucial role in extracting valuable insights from unstructured and unlabeled data. Unlike supervised learning, where the algorithm is trained on labeled data, unsupervised learning algorithms analyze the underlying patterns and structures within the data without any predefined labels or target variables. This makes unsupervised learning particularly useful in scenarios where there is limited or no prior knowledge about the data.

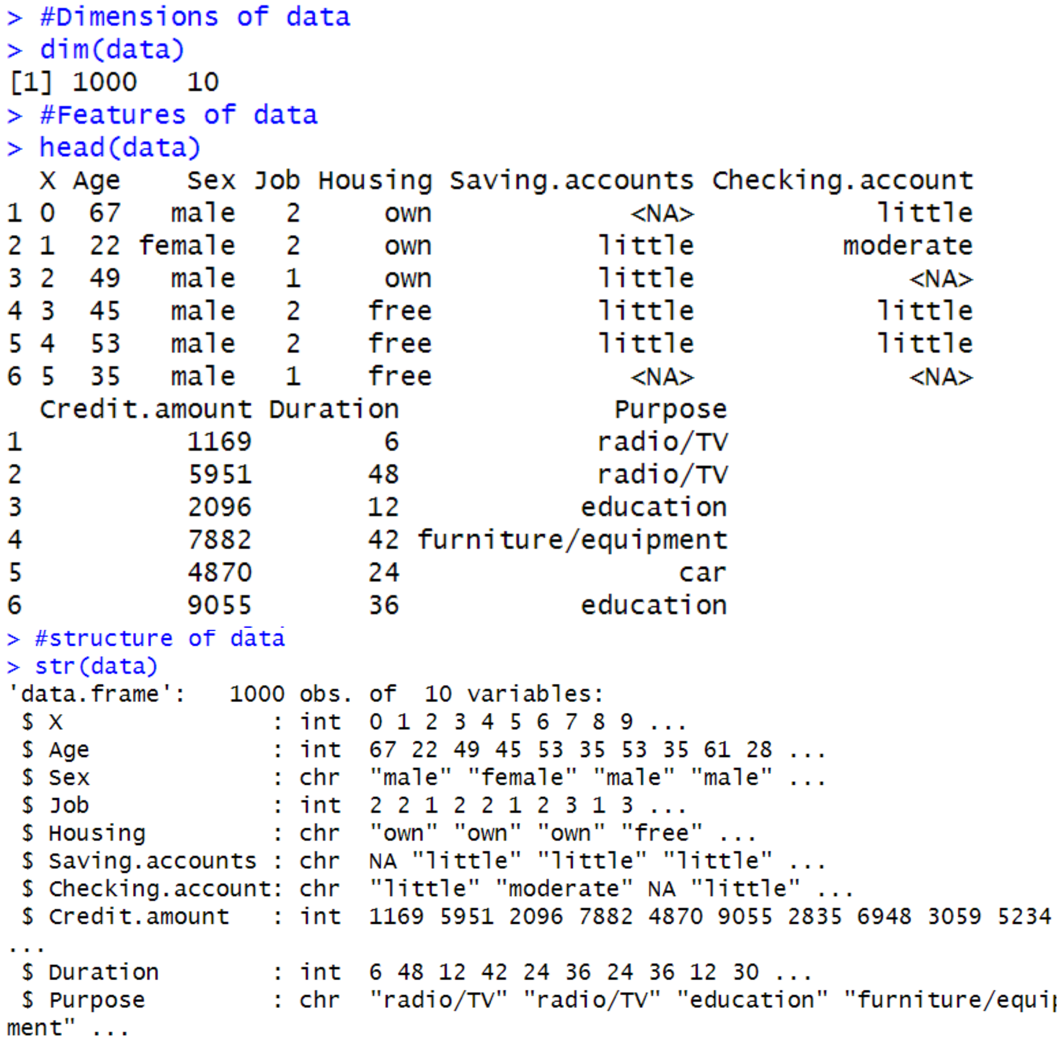
One of the primary applications of unsupervised learning is clustering, which involves grouping similar data points together based on their intrinsic characteristics. Clustering algorithms, such as K-Means, hierarchical clustering, and DBSCAN, help identify natural groupings or clusters within the data, allowing for better understanding and organization of complex datasets. By automatically uncovering patterns and similarities in the data, clustering enables researchers, businesses, and organizations to gain valuable insights, make data-driven decisions, and optimize processes.

3 Data Collection & Data Description

The dataset is a German bank customers dataset which contains 1000 observations and 10 variables. The source of this dataset is kaggle. The variables of this dataset are

- X - index
- Age
- Sex
- Job
- Housing
- Saving.accounts
- Checking.account
- Credit.amount
- Duration
- Purpose

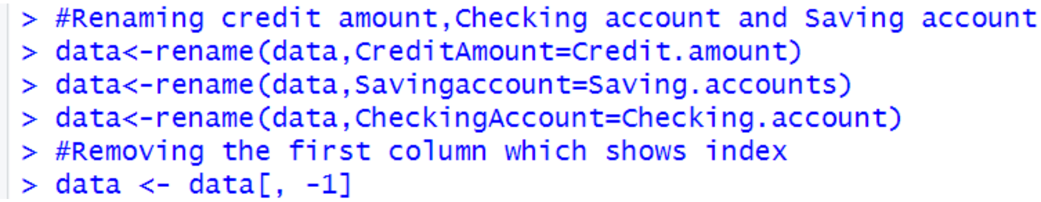
3.1 Dataset insights:



```
> #Dimensions of data
> dim(data)
[1] 1000   10
> #Features of data
> head(data)
  X Age    Sex Job Housing Saving.accounts Checking.account
1 0  67   male   2     own            <NA>           little
2 1  22 female   2     own          little         moderate
3 2  49   male   1     own          little             <NA>
4 3  45   male   2    free          little           little
5 4  53   male   2    free          little           little
6 5  35   male   1    free            <NA>             <NA>
  Credit.amount Duration             Purpose
1          1169        6            radio/TV
2          5951       48            radio/TV
3          2096       12           education
4          7882       42 furniture/equipment
5          4870       24                 car
6          9055       36           education
> #structure of data
> str(data)
'data.frame':   1000 obs. of  10 variables:
 $ X               : int  0 1 2 3 4 5 6 7 8 9 ...
 $ Age             : int  67 22 49 45 53 35 53 35 61 28 ...
 $ Sex             : chr  "male" "female" "male" "male" ...
 $ Job             : int  2 2 1 2 2 1 2 3 1 3 ...
 $ Housing         : chr  "own" "own" "own" "free" ...
 $ Saving.accounts : chr  NA "little" "little" "little" ...
 $ Checking.account: chr  "little" "moderate" NA "little" ...
 $ Credit.amount   : int  1169 5951 2096 7882 4870 9055 2835 6948 3059 5234
...
 $ Duration        : int  6 48 12 42 24 36 24 36 12 30 ...
 $ Purpose         : chr  "radio/TV" "radio/TV" "education" "furniture/equi
ment" ...
```

Figure 2: Dataset insights

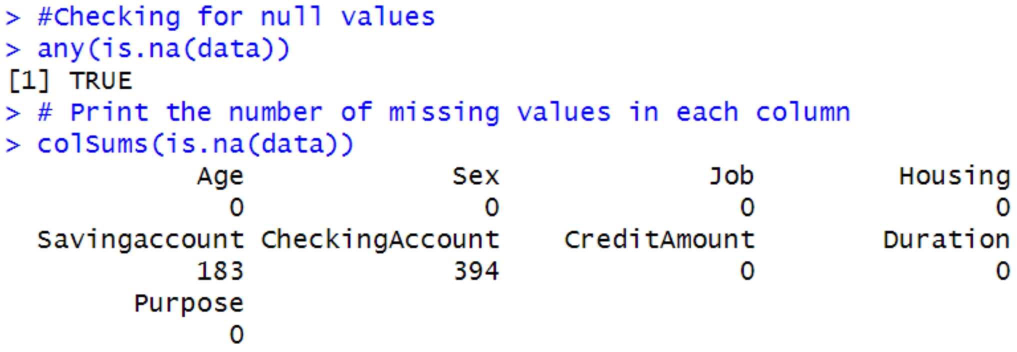
From the above figures, we can see that our dataset contains 1000 observations of 10 variables which includes basic customer information and also how much amount a customer has taken as a credit and for what purpose.



```
> #Renaming credit amount,Checking account and Saving account
> data<-rename(data,CreditAmount=Credit.amount)
> data<-rename(data,Savingaccount=Saving.accounts)
> data<-rename(data,CheckingAccount=Checking.account)
> #Removing the first column which shows index
> data <- data[, -1]
```

Figure 3: Renaming column names and removing unnecessary columns

Renaming the columns to make easy analysis in the future and it looks that the first column is simply an index which we can delete.So proceeding forward in removing the index column.

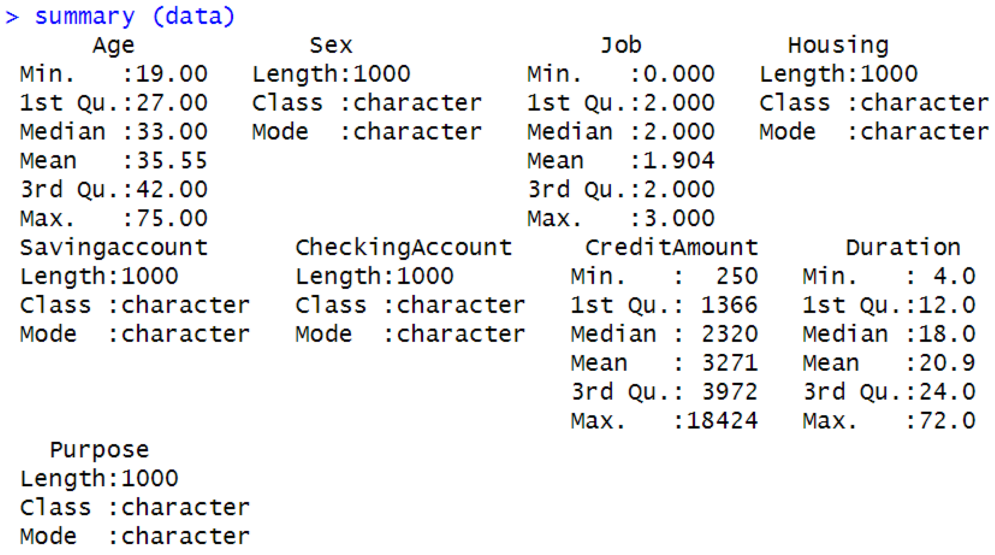


```
> #Checking for null values
> any(is.na(data))
[1] TRUE
> # Print the number of missing values in each column
> colSums(is.na(data))
            Age             Sex             Job         Housing 
              0               0               0               0 
  Savingaccount CheckingAccount    CreditAmount        Duration 
            183             394               0               0 
        Purpose 
              0 
```

Figure 4: Checking for null values

Out of 8 columns 2 contain missing values. Probably these are the customers who do not own one of these two accounts.In General, there are 3 numeric variables and 5 categorical ones.

From the below figure, you can see the summary of the dataset. From which, we can observe that the age of the customer varies from 19 to 75. The minimum credit amount a customer has taken is 250 and the maximum is 18424. The minimum duration for a customer to clear the loan is 4months and maximum is 72 months.



```
> summary (data)
      Age            Sex                 Job           Housing
 Min.   :19.00   Length:1000        Min.   :0.000   Length:1000
 1st Qu.:27.00   Class :character   1st Qu.:2.000   Class :character
 Median :33.00   Mode  :character   Median :2.000   Mode  :character
 Mean   :35.55                      Mean   :1.904
 3rd Qu.:42.00                      3rd Qu.:2.000
 Max.   :75.00                      Max.   :3.000
 Savingaccount      CheckingAccount     CreditAmount       Duration
 Length:1000        Length:1000        Min.   :  250   Min.   : 4.0
 Class :character   Class :character   1st Qu.: 1366   1st Qu.:12.0
 Mode  :character   Mode  :character   Median : 2320   Median :18.0
                                       Mean   : 3271   Mean   :20.9
                                       3rd Qu.: 3972   3rd Qu.:24.0
                                       Max.   :18424   Max.   :72.0
   Purpose
 Length:1000
 Class :character
 Mode  :character
```

Figure 5: Summary of the dataset

4 Exploratory Data Analysis & Feature Engineering:

4.1 Gender Breakdown

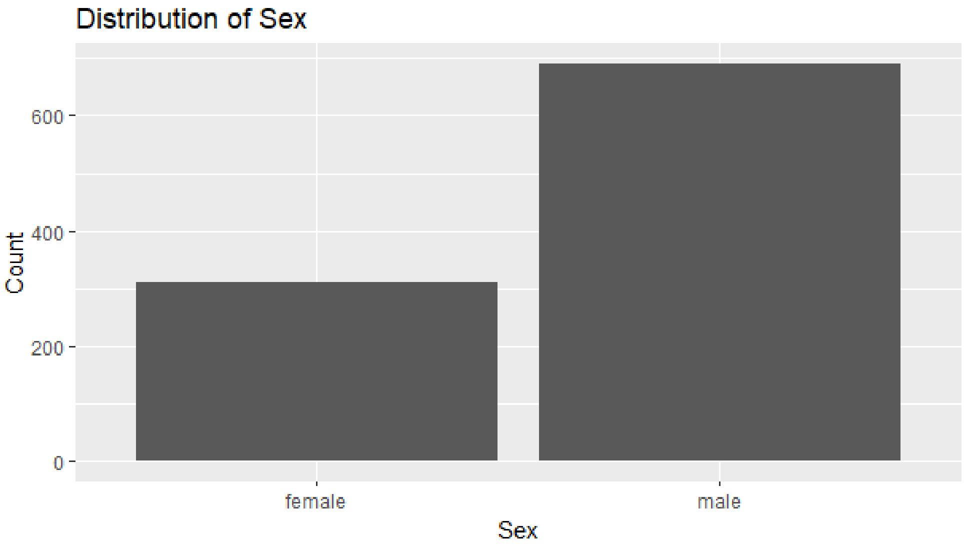
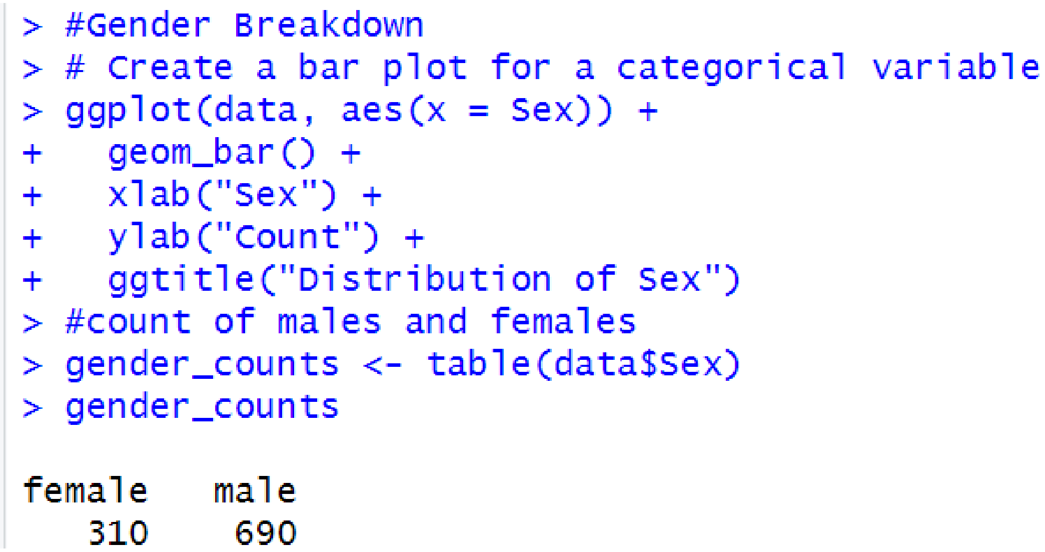


Figure 6: Histogram showing the count of gender



```
> #Gender Breakdown
> # Create a bar plot for a categorical variable
> ggplot(data, aes(x = Sex)) +
+   geom_bar() +
+   xlab("Sex") +
+   ylab("Count") +
+   ggtitle("Distribution of Sex")
> #count of males and females
> gender_counts <- table(data$Sex)
> gender_counts

female   male
   310    690
```

Figure 7: No. of males and females

We can see that the number of male is 690, whereas the number of females are 310, less than half of the count of males. Showing a great difference.

4.2 Frequency of age

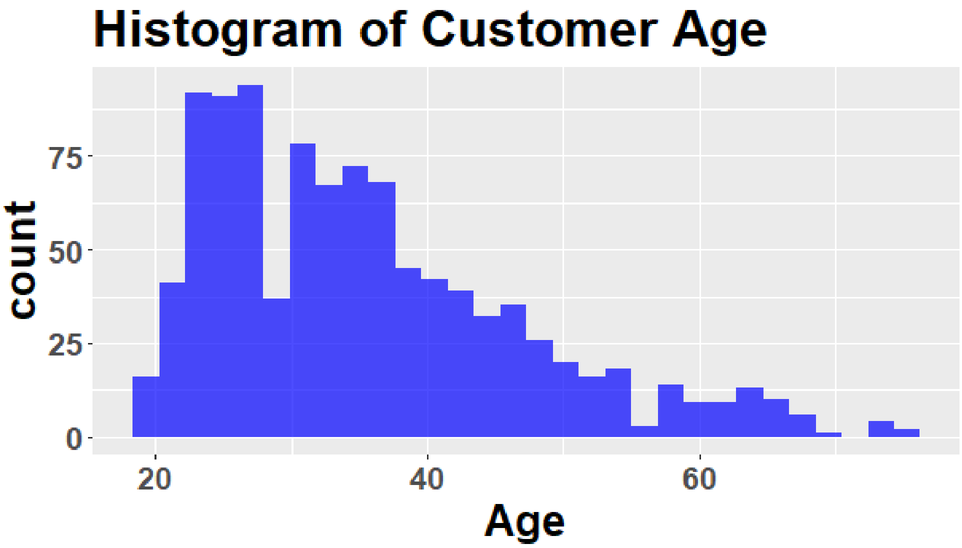


Figure 8: Histogram showing the count of Age

Most people who are taking credit are in the ages of 20 to 40. It makes sense when we think those are the people who are doing their graduation, either move into their own house or getting married and start living with their spouse. The older people are not taking up much credit as they are already settled and living their retirement.

4.3 Frequency of credit amount

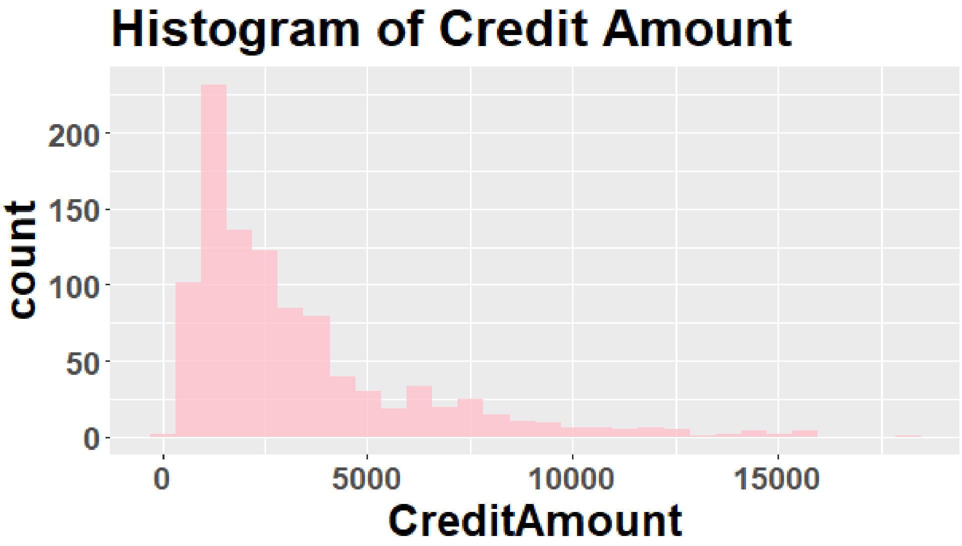


Figure 9: Histogram showing count Credit amount

The highest amount of credit people took up is 1000-5000 euros. 69.7 percentage of people has taken that amount. Intuitively its for radio/tv and furniture. These might be the people who moved into their own home and in need of buying electronics, equipment and furniture. As the amount increases, the number of people taking up the loan decreases. So the amount is negatively proportional to the count. As its huge amount, it maybe taken by the people who are doing their education.

4.4 Frequency of Duration

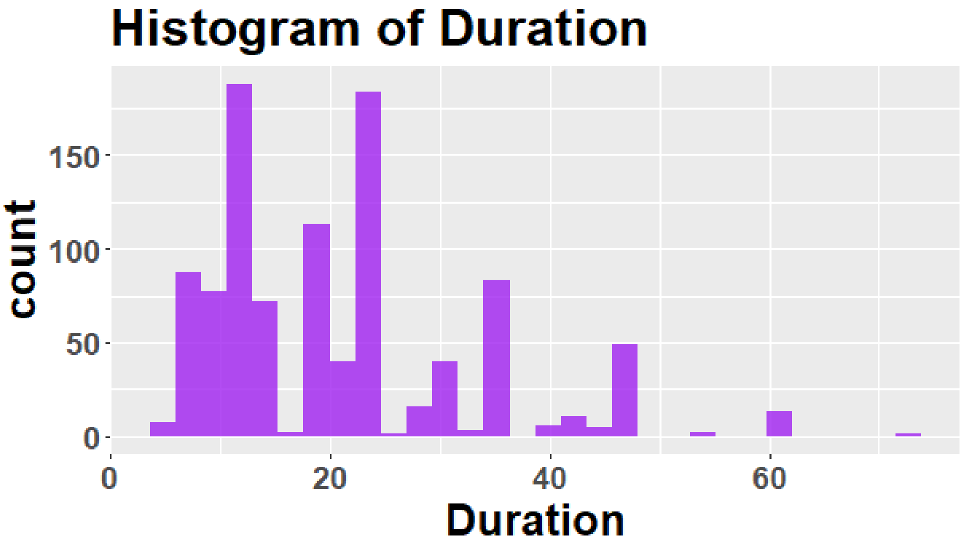


Figure 10: Histogram showing Count of Duration

From the above figure, we can see that many people are taking either 5,10,15,20,25 so on months to payback their credit amount. They are in the multiples of 5 between each other.Maybe this might be the bank's policy in declaring the period of time and customers chose which category they are eligible to pay back the loan. But we can also notice that few customers have paid their loan in different months compared to the counts of 5. This might be because those people have paid it before the due time of the loan.

So Far, we have learned that the count of female are less than fifty percent of the count of men. Most people who take up loan are in the age of 20 to 40.Their demographic percentage are:

- 0-18
  0%

- 19-30
  37.2%

- 31-40
  33.1%

- 41-50
  17.4%

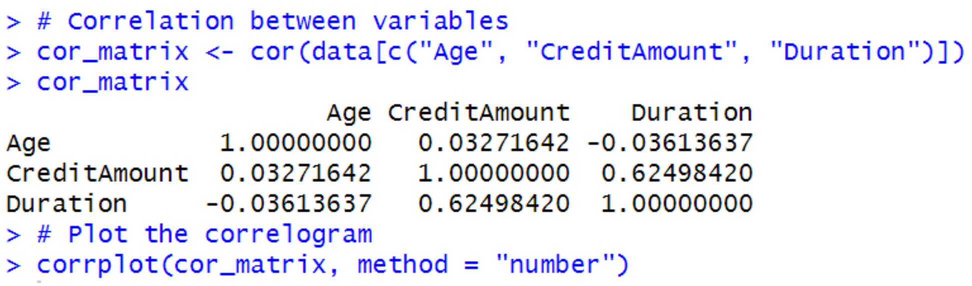
- 51-60
  7.4%

- 61+
  4.9%

69.7% of people have taken credit between 1000-5000 euros. Mean is 3271.258 euros. The mean off duration is 21 months.

5 In-Depth Data Exploration (relationship between variables)

5.1 Correlation matrix



```
> # Correlation between variables
> cor_matrix <- cor(data[c("Age", "CreditAmount", "Duration")])
> cor_matrix
                     Age CreditAmount    Duration
Age           1.00000000   0.03271642 -0.03613637
CreditAmount  0.03271642   1.00000000  0.62498420
Duration     -0.03613637   0.62498420  1.00000000
> # Plot the correlogram
> corrplot(cor_matrix, method = "number")
```

Figure 11: Correlation Matrix

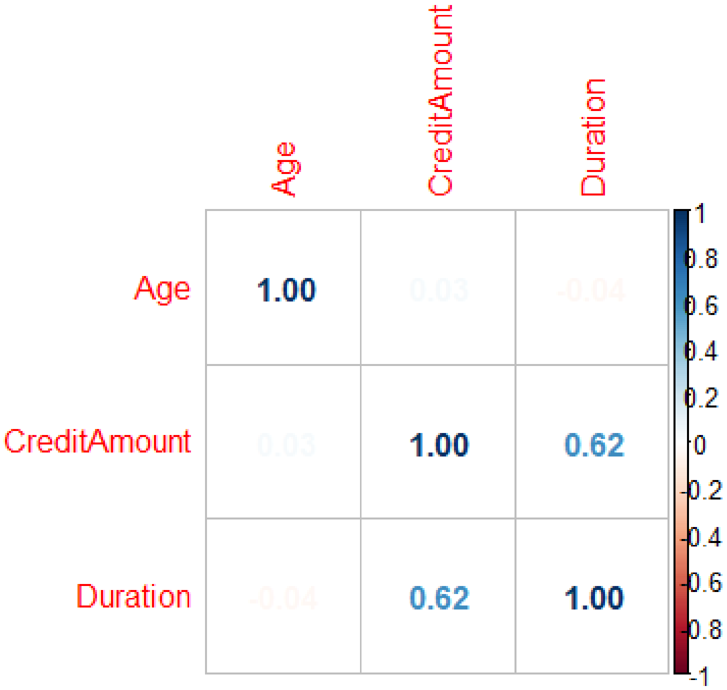
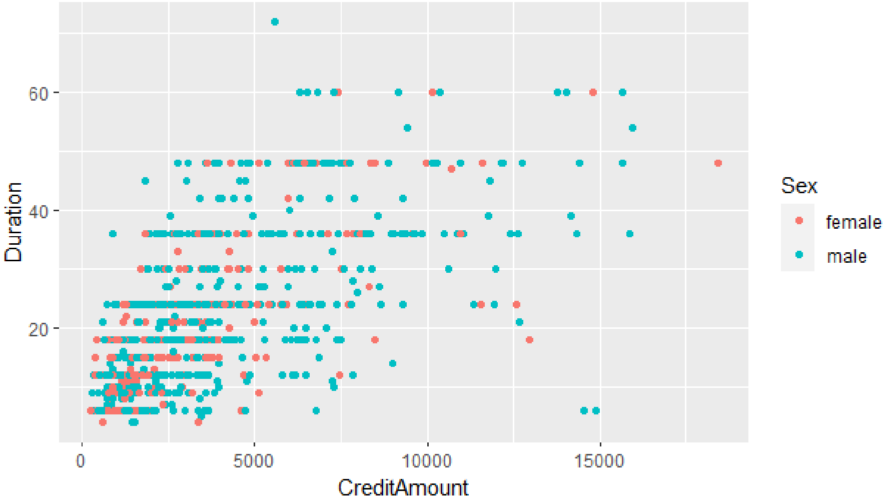


Figure 12: Correlation plot

The correlation matrix provides insights into the strength and direction of the linear relationship between pairs of variables. Positive values indicate a positive correlation, negative values indicate a negative correlation, and values closer to 1 or -1 represent stronger correlations. The correlogram plot provides a visual summary of these correlations.

We can see that the Variables Credit Amount and duration has positive relationship between them. But its still important to find the relationships between the other variables

5.2 Scatter Plots



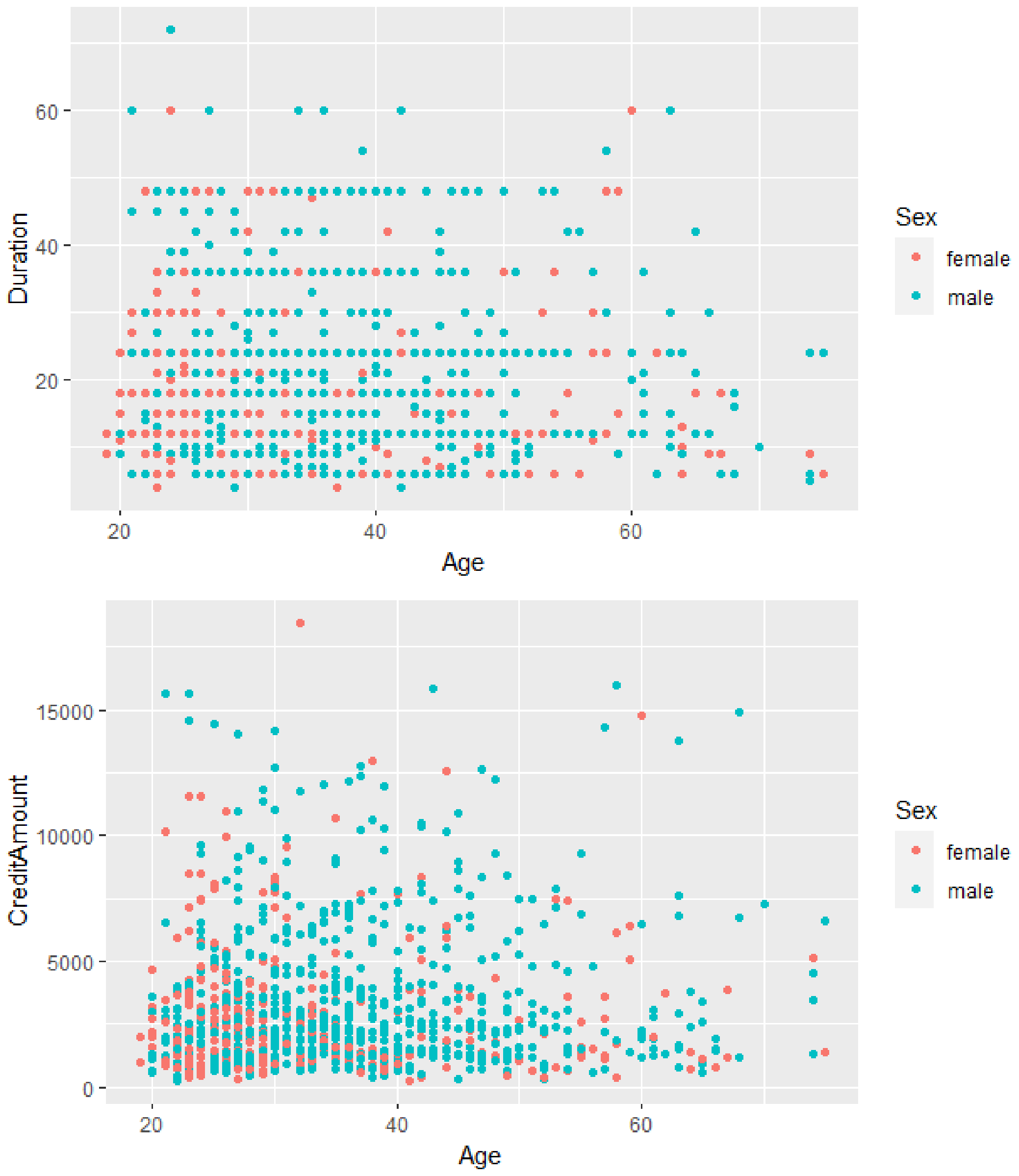
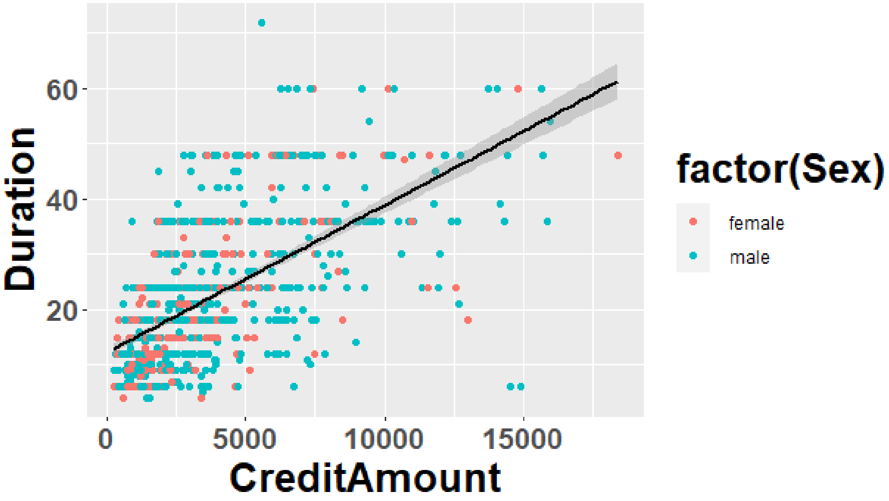


Figure 13: Scatter plot

The general impression is that women tend to be younger than men, however, the top plot shows that there is no clear difference between men and women in terms of amount and duration of the credit. From visual inspection, it seems that there is some positive correlation between duration and amount of credit, what makes sense.
Further, we will check the linear relationship between Credit amount and duration



The plot above shows a linear correlation.That makes sense because usually, people take bigger credits for longer periods.The plot above indicates that there is no significant difference between men and women. And we can also notice that customers took less credit for little amount of time and more credit for much longer period

6 Data Preprocessing

Data preprocessing is a crucial step in preparing data for analysis and modeling. It involves transforming raw data into a format that is suitable for further analysis. One common preprocessing technique is standardization, which aims to transform the data in such a way that it has zero mean and unit variance. This ensures that all variables are on a similar scale, allowing for fair comparison and interpretation.

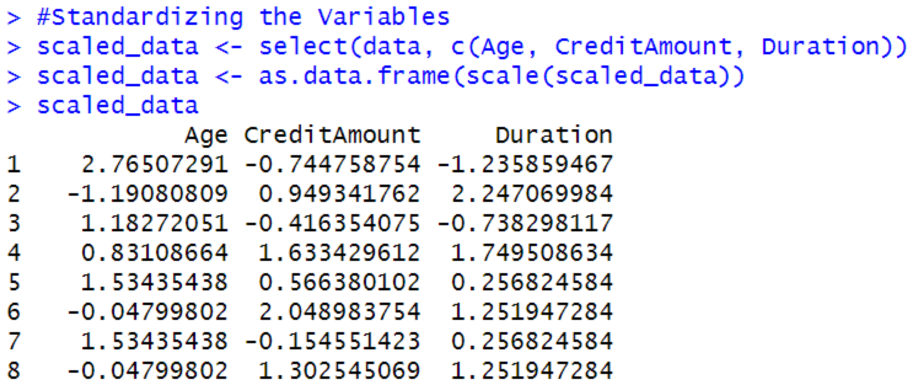
Standardization, also known as z-score normalization, can be achieved using the following formula:

$$Z = \frac{x - \mu}{\sigma}$$

Where:
Z is the standardized value
X is the original value

μ is the mean of the variable
σ is the standard deviation of the variable

By standardizing the data, you eliminate the differences in scales between variables, which is particularly useful when dealing with features with different units of measurement. It ensures that all variables contribute equally to the analysis and prevents one variable from dominating the results due to its larger scale.

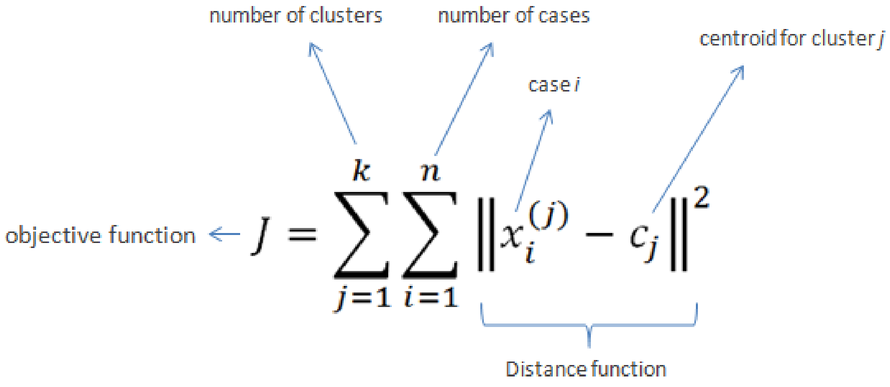


```
> #Standardizing the Variables
> scaled_data <- select(data, c(Age, CreditAmount, Duration))
> scaled_data <- as.data.frame(scale(scaled_data))
> scaled_data
          Age CreditAmount     Duration
1  2.76507291 -0.744758754 -1.235859467
2 -1.19080809  0.949341762  2.247069984
3  1.18272051 -0.416354075 -0.738298117
4  0.83108664  1.633429612  1.749508634
5  1.53435438  0.566380102  0.256824584
6 -0.04799802  2.048983754  1.251947284
7  1.53435438 -0.154551423  0.256824584
8 -0.04799802  1.302545069  1.251947284
```

7 Model Building

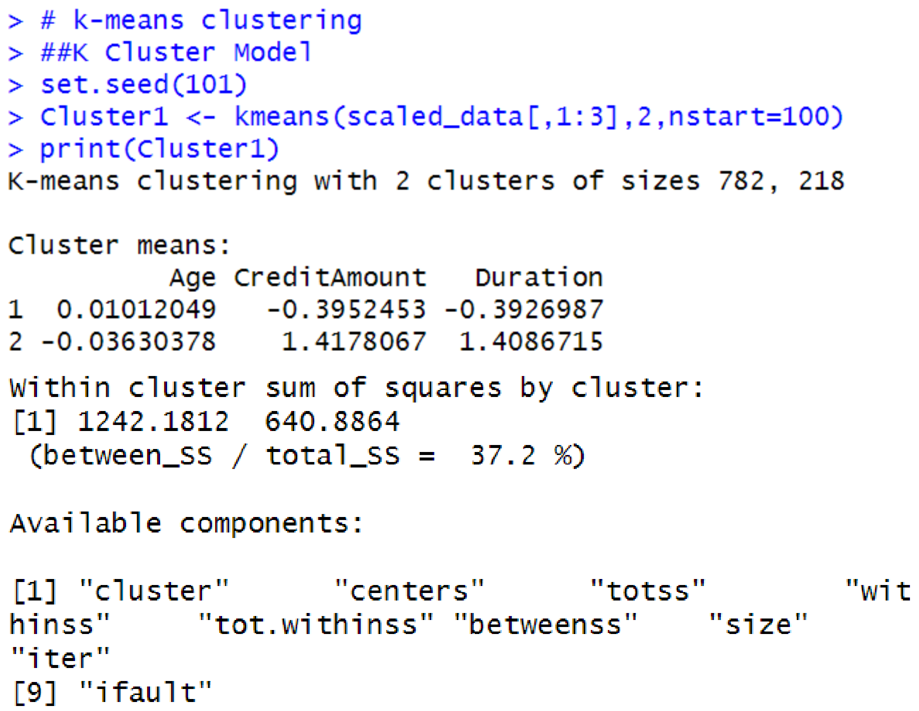
7.1 K-means clustering

K-means clustering is a popular unsupervised machine learning algorithm used to partition a dataset into distinct groups or clusters. It iteratively assigns data points to clusters based on their proximity to cluster centroids. The algorithm aims to minimize the within-cluster sum of squares, making observations within the same cluster more similar to each other than to those in other clusters. K-means clustering is widely used for exploratory data analysis, customer segmentation, image processing, and many other applications. It provides a simple and effective approach to uncover underlying patterns and structure within data.



7.2 Steps to perform kmeans algorithm

- Clusters the data into k groups where k is predefined.
- Select k points at random as cluster centers.
- Assign objects to their closest cluster center according to the Euclidean distance function.
- Calculate the centroid or mean of all objects in each cluster.
- Repeat steps 2, 3 and 4 until the same points are assigned to each cluster in consecutive rounds.



```
> # k-means clustering
> ##K Cluster Model
> set.seed(101)
> Cluster1 <- kmeans(scaled_data[,1:3],2,nstart=100)
> print(Cluster1)
K-means clustering with 2 clusters of sizes 782, 218

Cluster means:
          Age CreditAmount   Duration
1  0.01012049   -0.3952453 -0.3926987
2 -0.03630378    1.4178067  1.4086715

Within cluster sum of squares by cluster:
[1] 1242.1812  640.8864
 (between_SS / total_SS =  37.2 %)

Available components:

[1] "cluster"      "centers"      "totss"        "wit
hinss"     "tot.withinss" "betweenss"    "size"
"iter"
[9] "ifault"
```

8 Tuning the Model:

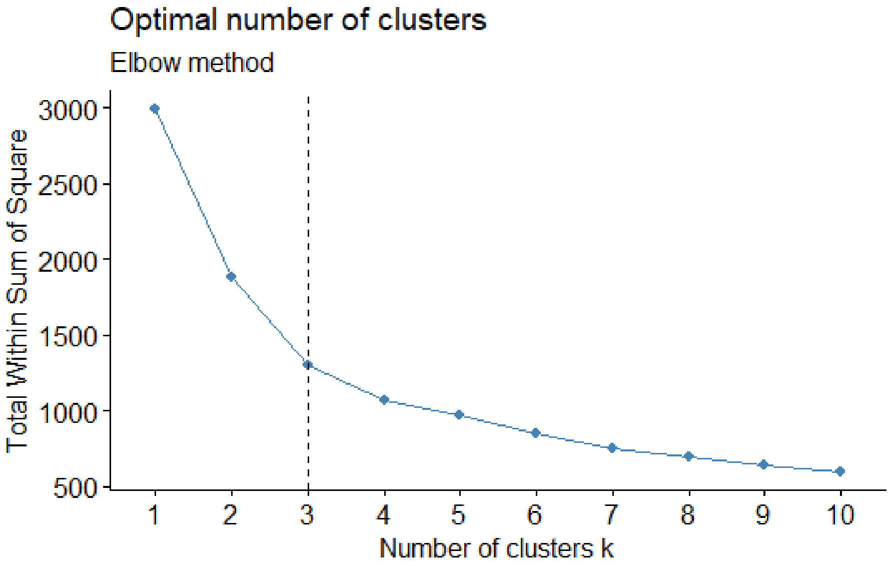
When using Kmeans, the number of clusters (k), is a value to be set by the user. Few methods to determine the appropriate number of clusters, as shown below.

8.1 The Elbow Method

The Elbow Method is a technique used to determine the optimal number of clusters in K-means clustering. It helps to identify the point of diminishing returns, where adding more clusters does not significantly improve the clustering performance. The method is based on plotting the Within-cluster sum of squares (WCSS) against the number of clusters and looking for the ”elbow” point in the resulting curve.
The formula for calculating the WCSS is as follows:

$$WCSS = \sum_{i \in n} (X_i - Y_i)^2$$

Where:
where Yi is centroid for observation Xi. The main goal is to maximize number of clusters and in limiting case each data point becomes its own cluster centroid.

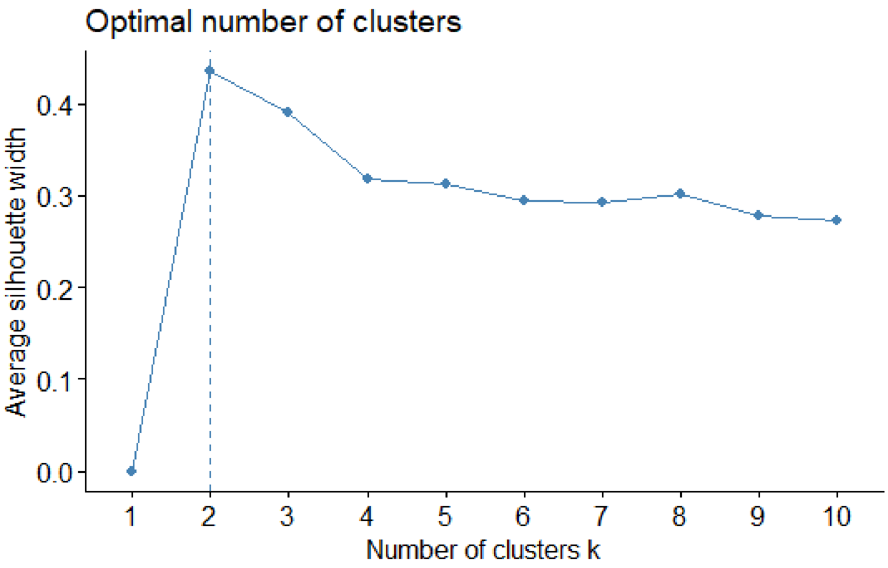


To apply the Elbow Method, you would perform K-means clustering for a range of cluster numbers, calculate the WCSS for each clustering solution, and plot the WCSS against the number of clusters. The elbow point on the resulting curve indicates the optimal number of clusters.

8.2 Average Silhouette Method

The silhouette plot displays a measure of how close each point in one cluster is to points in the neighboring clusters and thus provides a way to assess parameters like number of clusters visually. This measure has a range of [-1, 1].

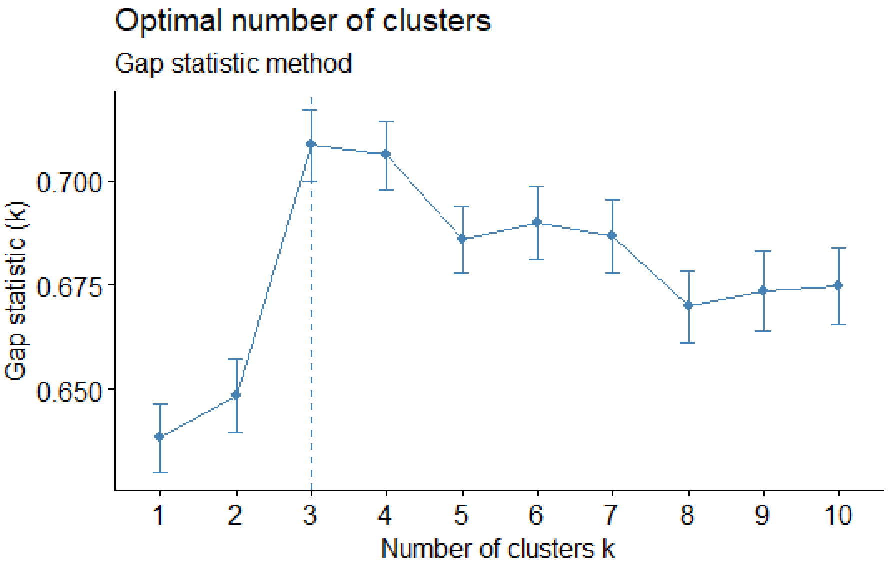
Silhouette coefficients (as these values are referred to as) near +1 indicate that the sample is far away from the neighboring clusters. A value of 0 indicates that the sample is on or very close to the decision boundary between two neighboring clusters and negative values indicate that those samples might have been assigned to the wrong cluster.



Silhouette analysis allows you to calculate how similar each observation is with the cluster it is assigned relative to other clusters. This metric ranges from -1 to 1 for each observation in your data and can be interpreted as a poor fit (-1), a loose fit that is borderline between clusters (0), and a great fit (1). Maximizing the silhouette metric is the goal, and should yield the optimal amount of clusters.

8.3 Gap Statistic Method

The Gap Statistic Method is a technique used to determine the optimal number of clusters in clustering algorithms, such as K-means. It compares the Within-cluster sum of squares (WCSS) of the data against a null reference distribution to assess the clustering quality. The method calculates the gap statistic for different numbers of clusters and identifies the number of clusters that maximizes the gap statistic.

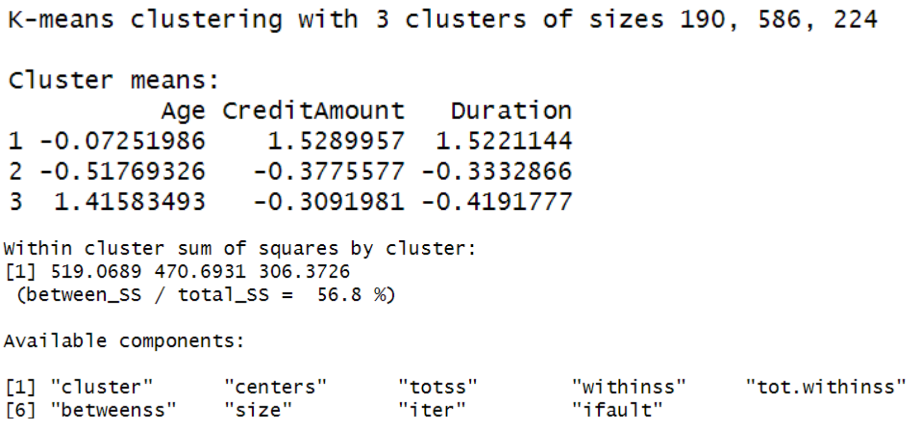


The formula for calculating the gap statistic is as follows:
Gap statistic = E(log(WCSSref)) - log(WCSS)
Where:
WCSSref is the WCSS of the reference distribution (randomly generated data with the same dimensions and range as the original dataset). WCSS is the WCSS of the actual clustering solution.

To apply the Gap Statistic Method, you would perform clustering with varying numbers of clusters, calculate the WCSS for each clustering solution, generate the reference distribution, and then calculate the gap statistic for each number of clusters. The number of clusters that maximizes the gap statistic is considered the optimal number of clusters.

Given all of the above, I felt most comfortable moving forward with 3 clusters. Because 2 methods indicate 6 clusters are best for your model.

9 Final Model

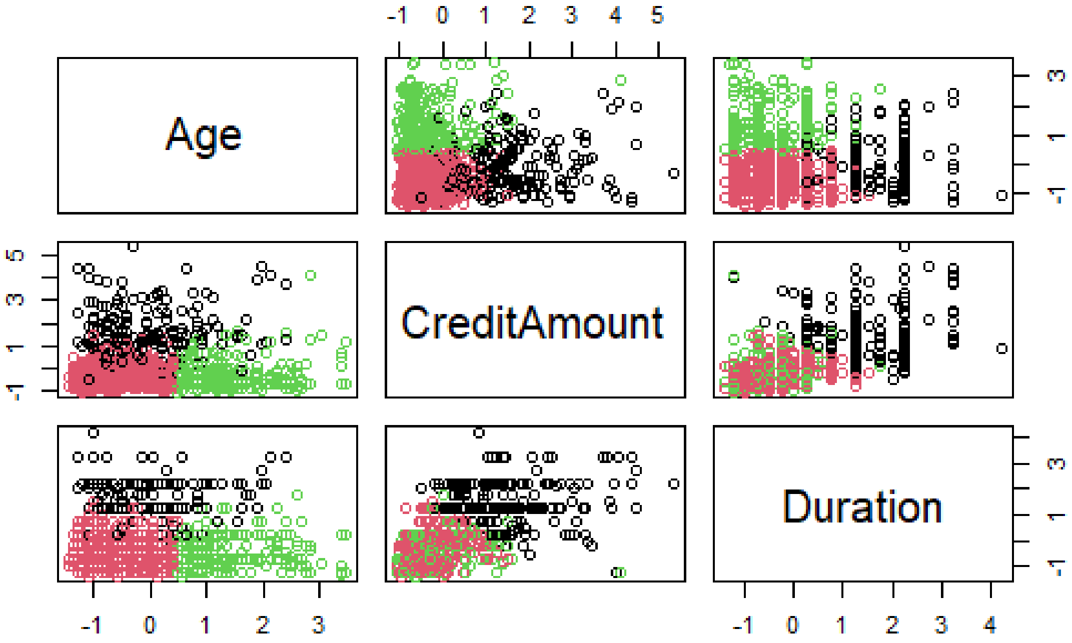


```
K-means clustering with 3 clusters of sizes 190, 586, 224

Cluster means:
          Age CreditAmount   Duration
1 -0.07251986    1.5289957  1.5221144
2 -0.51769326   -0.3775577 -0.3332866
3  1.41583493   -0.3091981 -0.4191777

Within cluster sum of squares by cluster:
[1] 519.0689 470.6931 306.3726
 (between_SS / total_SS =  56.8 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
```

Now we can see the cluster goodness has improved to 56.8%

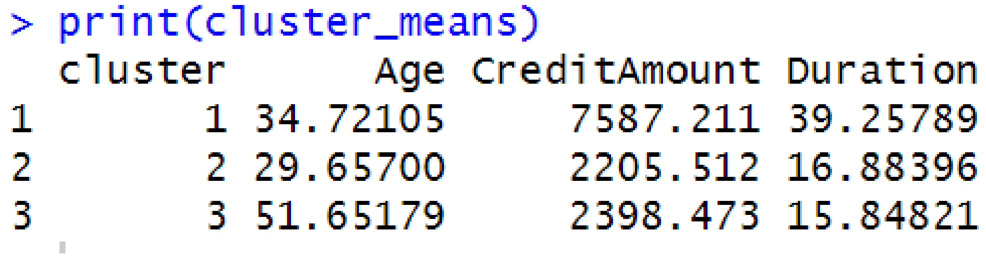
10 Matrix of Tuned Model

This is a matrix of clusters built on two-variable combinations of age, CreditAmount, and Duration. After tuning the model to have 3 clusters, the clusters explained 56.8% of the variance within the data.

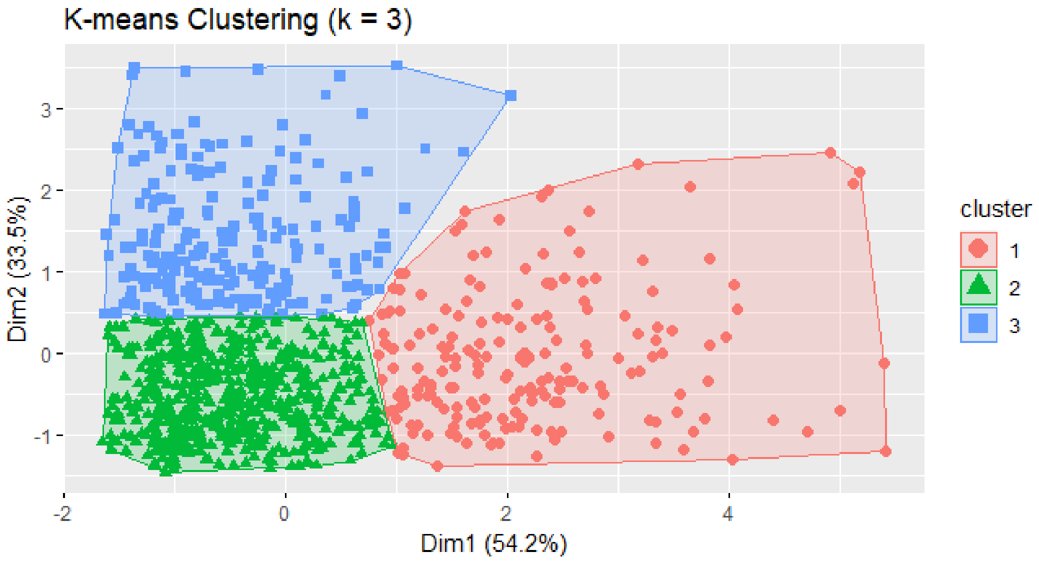


11 Results

Clustered data (K-Means) into 3 customer segments based on age, CreditAmount and Duration



```
> print(cluster_means)
  cluster      Age CreditAmount Duration
1       1 34.72105     7587.211 39.25789
2       2 29.65700     2205.512 16.88396
3       3 51.65179     2398.473 15.84821
```



11.1 Cluster 1

- **Age:** The average age in this cluster is around 34.72. It indicates that there is a group of customers in their early to mid-thirties who have certain credit characteristics.
- **CreditAmount:** The average credit amount in this cluster is 7587.21. It suggests that customers in this cluster tend to have higher credit amounts, indicating a higher borrowing capacity or a preference for larger loans.
- **Duration:** The average loan duration in this cluster is 39.26. This implies that customers in this cluster tend to choose longer loan durations, which may indicate their ability to manage longer-term financial commitments.
- **Conclusion for Cluster 1:** This cluster represents a group of relatively younger customers (in their early to mid-thirties) who have higher credit amounts and prefer longer loan durations. They may be individuals who are at a stage in their lives where they require larger financial assistance, such as buying a home or financing a significant investment.

11.2 Cluster 2

–

- **Age:** The average age in this cluster is around 29.66. It suggests that there is a group of relatively younger customers in their late twenties to early thirties who exhibit specific credit characteristics.
- **CreditAmount:** The average credit amount in this cluster is 2205.51. It indicates that customers in this cluster have lower credit amounts compared to other clusters, suggesting they might be borrowing smaller amounts or have a lower borrowing capacity.

- **Duration:** The average loan duration in this cluster is 16.88. This implies that customers in this cluster tend to choose shorter loan durations, indicating a preference for quicker repayment or shorter-term financial commitments.
- **Conclusion for Cluster 2:** This cluster represents a group of relatively younger customers (in their late twenties to early thirties) who have lower credit amounts and prefer shorter loan durations. They may be individuals who require smaller loans or prefer to repay their debts quickly

11.3 Cluster 3

- **Age:** The average age in this cluster is around 51.65. It indicates that there is a group of older customers, likely in their fifties, who exhibit specific credit characteristics.
- **CreditAmount:** The average credit amount in this cluster is 2398.47, which is similar to Cluster 2. It suggests that customers in this cluster have moderate credit amounts, similar to the younger cluster.
- **Duration:** The average loan duration in this cluster is 15.85, slightly shorter than Cluster 2. This implies that customers in this cluster prefer relatively shorter loan durations, similar to the younger cluster.
- **Conclusion for Cluster 3:** This cluster represents a group of older customers (in their fifties) who have moderate credit amounts and prefer shorter loan durations. They may be individuals who are more financially established, have moderate borrowing needs, and prefer shorter loan repayment periods.

12 Conclusion

Based on the conclusions derived from the analysis of the German credit data, a bank can take further care in the following ways:

- **Tailored Marketing Strategies:** The bank can customize its marketing strategies based on the identified clusters. For example, for Cluster 1 customers who require larger loans and longer durations, the bank can offer personalized loan products with flexible repayment options. For Cluster 2 customers who prefer smaller loans and shorter durations, the bank can provide targeted loan offers with quick processing and attractive interest rates. For Cluster 3 customers who are older and prefer moderate-sized loans, the bank can offer specialized loan packages tailored to their specific needs.
- **Risk Assessment and Loan Approval:** By understanding the characteristics of each cluster, the bank can perform more accurate risk assessments for loan applications. For instance, Cluster 1 customers with higher credit amounts and longer durations may require more thorough scrutiny due to the potential higher risk associated with larger loan amounts. Cluster 2 customers, on the other hand, may present lower risk profiles, given their lower credit amounts and shorter durations. The bank can adjust its loan approval processes and criteria accordingly.

- **Customer Segmentation and Service Personalization:** The identified clusters can help the bank segment its customer base for personalized services. Each cluster may have unique preferences, needs, and financial goals. By tailoring banking services, such as account types, investment options, and financial planning advice, to each cluster, the bank can enhance customer satisfaction and loyalty.

- **Product Development:** The bank can use the insights gained from the analysis to develop new financial products or refine existing ones. For example, based on the demand and characteristics of Cluster 1 customers, the bank can introduce specialized mortgage or investment products that cater to their specific requirements. Similarly, for Cluster 2 customers, the bank can design products that focus on quick and hassle-free loan approvals with competitive interest rates.

13 Appendix

https://github.com/TejaswiniYadavNakka/Statistical_Learning_Unsupervised

```
install.packages("ggplot2")
install.packages("dplyr")
install.packages("corrplot")
install.packages("factoextra")

library(ggplot2)
library(dplyr)
library(corrplot)
library(factoextra)



#Loading in file
data=read.csv("C:/Users/Tejaswini␣yadav/Desktop/bank␣data/german_credit_data.csv")

#Dimensions of data
dim(data)

#Features of data
head(data)


#structure of data
str(data)


#Renaming credit amount,Checking account and Saving account
data<-rename(data,CreditAmount=Credit.amount)
data<-rename(data,Savingaccount=Saving.accounts)
data<-rename(data,CheckingAccount=Checking.account)


#Removing the first column which shows index
data <- data[, -1]
head(data)


#Checking for null values
any(is.na(data))

# Print the number of missing values in each column
colSums(is.na(data))

# Print the data types of each column
print(sapply(data, class))
```

```
#Exploratory data analysis


# Plot Themes
bold_axis <- element_text(face = "bold", color = "black", size = 20)
axis_text <- element_text(face = "bold", size = 14)

#Gender Breakdown
# Create a bar plot for a categorical variable
ggplot(data, aes(x = Sex)) +
  geom_bar() +
  xlab("Sex") +
  ylab("Count") +
  ggtitle("Distribution␣of␣Sex")

#count of males and females
gender_counts <- table(data$Sex)
gender_counts

# Plot Customers by Age
Plotage <- ggplot(data,aes(x=Age))
Plotage + geom_histogram(fill="blue", alpha = 0.7) + theme(axis.text = axis_text) + theme


#Plot by CreditAmount
PlotCredit <- ggplot(data, aes(x = CreditAmount))
PlotCredit + geom_histogram(fill="pink", alpha = 0.8) + theme(axis.text = axis_text) + th
mean(data$CreditAmount)
sd(data$CreditAmount)


#Plot by Duration
PlotDuration <- ggplot(data, aes(x = Duration))
PlotDuration + geom_histogram(fill="Purple", alpha = 0.8) + theme(axis.text = axis_text)
mean(data$Duration)
sd(data$Duration)


#InDepth Data Exploration(Relationship between variables)

# Correlation between variables
cor_matrix <- cor(data[c("Age", "CreditAmount", "Duration")])

cor_matrix
# Plot the correlogram
corrplot(cor_matrix, method = "number")

#Scattered plot of CreditAmount and Duration
```

```
ggplot(data,aes(x = CreditAmount, y = Duration, col=Sex))+geom_point()

#Scattered plot of Age and Duration
ggplot(data,aes(x =Age, y = Duration, col=Sex))+geom_point()

#Scattered plot of CreditAmount and Age
ggplot(data,aes(x = Age, y = CreditAmount,col=Sex))+geom_point()

#Checking the linear relationship between credit amount and duration
scatter2 <- ggplot(data, aes(x = CreditAmount, y = Duration))  + geom_point(aes(color = f
scatter2 +geom_smooth(method = "lm", color ="black") + theme(axis.text = axis_text) + ther

##histogram of Gender and Spending Score
#Clustering with KMeans

# Select numerical columns for K-means clustering
numerical_cols <- data[c("Age", "CreditAmount", "Duration")]



# Create histograms for the columns
ggplot(data, aes(x = Age)) +
  geom_histogram( fill = "blue", color = "white") +
  xlab("Age") +
  ylab("Frequency") +
  ggtitle("Histogram␣of␣Age")


ggplot(data, aes(x = CreditAmount)) +
  geom_histogram( fill = "green", color = "white") +
  xlab("Credit␣Amount") +
  ylab("Frequency") +
  ggtitle("Histogram␣of␣Credit␣Amount")

ggplot(data, aes(x = Duration)) +
  geom_histogram( fill = "orange", color = "white") +
  xlab("Duration") +
  ylab("Frequency") +
  ggtitle("Histogram␣of␣Duration")


#Standardizing the Variables
scaled_data <- select(data, c(Age, CreditAmount, Duration))
scaled_data <- as.data.frame(scale(scaled_data))
scaled_data

# k-means clustering
##K Cluster Model
set.seed(101)
```

```r
Cluster1 <- kmeans(scaled_data[,1:3],2,nstart=100)
print(Cluster1)
# deciding optimal number of cluster
# Elbow method
fviz_nbclust(scaled_data, kmeans, method = "wss")+
  geom_vline(xintercept = 3, linetype = 2)+labs(subtitle = "Elbow␣method")

#Average Silhouette Method
set.seed(101)
fviz_nbclust(scaled_data, kmeans, method = "silhouette")

#Gap Statistic Method
set.seed(101)
fviz_nbclust(scaled_data, kmeans, nstart = 25,  method = "gap_stat", nboot = 50)+
  labs(subtitle = "Gap␣statistic␣method")




#Adjusting Kmeans Model
set.seed(101)
kmeans_result <- kmeans(scaled_data[,1:3],3,iter.max=100, nstart=100)
kmeans_result

plot(scaled_data[,1:3], col=kmeans_result$cluster)

kmeans_result$centers

# Visualize the clusters
fviz_cluster(kmeans_result, data = scaled_data, geom = "point",
             frame = FALSE, stand = FALSE, pointsize = 2) +
  ggtitle("K-means␣Clustering␣(k␣=␣3)")

# making cluster as factor
kmeans_result$cluster <- as.factor(kmeans_result$cluster)
# assgining cluster to the original  data set
data.clust <- cbind(numerical_cols, cluster = kmeans_result$cluster)

# Aggregate the clustered data by cluster and calculate the means
cluster_means <- aggregate(data.clust[, 1:ncol(data.clust) - 1],
                           by = list(cluster = data.clust$cluster),
                           mean)

# Print the cluster means
print(cluster_means)
```