**A logo of a university of windsor

Description automatically generated with low confidence**

|  |
| --- |
| Course Name |
| Advanced Database Topics (COMP-8157) |

|  |
| --- |
| Document Type |
| Lab 1 |

|  |
| --- |
| Professor |
| Dr. Shafaq Khan |

|  |
| --- |
| Student Name Student ID |
| Manjinder Singh 110097177 |

**Part 1: Data Exploration**

***QUESTION 1:***Import the Vehicle dataset, summarize it and explain the output (2 marks).

|  |
| --- |
| R Code |
| *#Data is imported in "vehicle\_data" data.frame*  vehicle\_data <- read.csv("C:/Users/rahul/Downloads/Vehicle.csv")  *#Checking the type of dataset to work on further*  dataset\_type\_check <- class(vehicle\_data)  print(dataset\_type\_check)  *#Summarizing the vehicle\_data which mainly shows the Minimum Value, 1st Quantile, Median, Mean, 3rd Quantile, Maximum Value of numeric data columns and for string columns, it will return the count of values.*  summary(vehicle\_data) |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
|  |

|  |
| --- |
| Explanation |
| In this, we imported the **Vehicles.csv** data and then we had summarized the **vehicle**\_**data** using a summary function which mainly shows   1. the Minimum Value, 1st Quantile, Median, Mean, 3rd Quantile, and Maximum Value of **numeric data columns**, and 2. for **string columns**, it will return the count of values of the string   These insights are useful to get an overview of the data. |

***QUESTION 2 :*** Show the structure and dimension of the dataset and explain it

|  |
| --- |
| R Code |
| # str will display the structure and Summary of the Vehicle Dataset  str(vehicle\_data)  # dim will display the dimensions of the Vehicle Dataset i.e. it will return column count and  dim(vehicle\_data) |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
|  |

|  |
| --- |
| Explanation |
| In this, str and dim functions are used for the following reasons:   * str function will display the structure and summary of the dataset(mainly overview of datatype of columns along with the values) whereas * dim function will display the number of row and columns. |

***QUESTION 3:*** Show the column names of the Vehicle dataset and the first 3 rows and the last 6 rows of it

|  |
| --- |
| R Code |
| # Method to display the column names of the Vehicle Dataset  colnames(vehicle\_data)  # Displaying the first 3 rows of the Vehicle Dataset using head function  head(vehicle\_data, n = 3)  # Displaying the last 6 rows of the Vehicle Dataset using tail function  tail(vehicle\_data, n = 6) |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
|  |

|  |
| --- |
| Explanation |
| In this question, I had used the following:   * colnames function to print column names of vehicle dataset. * head function is used to print top n rows and parameter n is mentioned to fetch top n i.e. 3 rows of vehicle dataset. * tail function is used to print last n rows and parameter n is mentioned to fetch last n i.e. 6 rows of vehicle dataset. |

***QUESTION 4 :*** Show the average Kms\_Driven for each type of car (Car\_Name) in the dataset.

|  |
| --- |
| R Code |
| # Calculating the average number of kilometers for each car and storing in average\_kms\_driven  average\_kms\_per\_car <- aggregate(vehicle\_data$Kms\_Driven, by = list(vehicle\_data$Car\_Name), FUN = mean)  # Naming the columns in the result  colnames(average\_kms\_per\_car) <- c("Car\_Name", "Average\_Kilometers\_Driven")  # Displaying the result  average\_kms\_per\_car |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
|  |

|  |
| --- |
| Explanation |
| In this, aggregate function is used for Kms\_Driven and Car\_Name Columns along with mean function and in return average kms per car is returned with the appropriate names of the table as "Car\_Name", "Average\_Kilometers\_Driven." |

**QUESTION 5 :** What is the average Selling\_Price of the cars in each year?

|  |
| --- |
| R Code |
| # Calculating the average Selling\_Price for each year and saving in average\_price\_per\_year dataframe  average\_price\_per\_year <- aggregate(Selling\_Price ~ Year, data = vehicle\_data, FUN = mean)  # Displaying the result after calculation  average\_price\_per\_year |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
|  |

|  |
| --- |
| Explanation |
| In this, aggregate function is used for Selling\_Price and Year Columns along with mean function and in return average selling price per year is returned.  Example: Average selling price in 2013 was 1.3 units |

***QUESTION 6:*** Show the unique combinations of Car\_Name, Fuel\_Type, Seller\_Type, and Transmission in the Vehicle dataset.

|  |
| --- |
| R Code |
| # Selecting the columns of interest  required\_columns <- c("Car\_Name", "Fuel\_Type", "Seller\_Type", "Transmission")  # Extracting the unique combinations  uniq\_comb <- unique(vehicle\_data[required\_columns])  # Displaying the result  uniq\_comb |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
|  |

|  |
| --- |
| Explanation |
| In this, first required columsn are considered("Car\_Name", "Fuel\_Type", "Seller\_Type", "Transmission") then unique function is applied on that to get unique set of mentioned columns from vehicles dataset. |

***QUESTION 7:*** What are the different combinations of Car\_Name, Fuel\_Type, Seller\_Type, and Transmission in the Vehicle dataset, and how many times does it occur? (Display all such in both ascending and descending orders)

|  |
| --- |
| R Code |
| # Creating a combination column  reqd\_data$Combination <- paste(vehicle\_data$Car\_Name, vehicle\_data$Fuel\_Type, vehicle\_data$Seller\_Type, vehicle\_data$Transmission, sep = ", ")  # Calculating the frequencies of combinations  comb\_freq <- table(reqd\_data$Combination)  # Sorting the combinations in ascending order of frequency  sorted\_comb\_asc <- sort(comb\_freq)  # Sorting the combinations in descending order of frequency  sorted\_comb\_desc <- sort(comb\_freq, decreasing = TRUE)  # Displaying the combinations and their frequencies in ascending order  sorted\_comb\_asc  # Displaying the combinations and their frequencies in descending order  sorted\_comb\_desc |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
| \* The output is lengthy so wont be possible to display all in one screenshot but the R script works fine and it will display the complete output during execution. |

|  |
| --- |
| Explanation |
| First, the required columns are considered then we calculated the frequency of the combination. Then the results are saved in sorted\_comb\_asc with the help of sort function and by default, ascending order is applied. Similarly, for descending order we performed the similar approach but we defined the decareasing parameter to True for descending order. |

**Part 2: Data Pre-Processing**

***QUESTION 8 :*** Find if there are any missing values in the Vehicle dataset.

|  |
| --- |
| R Code |
| # Checking for missing values in dataset  find\_missing\_values <- any(is.na(vehicle\_data))  # Displaying the result for missing values in the dataset  find\_missing\_values |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
|  |

|  |
| --- |
| Explanation |
| In this, any function is used to check missing values with the is.na and in return the result is saved in find\_missing\_values variable.  The final result is FALSE as there is no missing value in the dataset. |

***QUESTION 9:*** Find which columns contain missing values in the vehicles dataset. What are the total missing values for each column?

|  |
| --- |
| R Code |
| # Checking for missing values in each column of the dataset  missing\_columns <- colSums(is.na(vehicle\_data))  # Displaying the columns with missing values and their total missing values for each column.  missing\_columns |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
|  |

|  |
| --- |
| Explanation |
| In this, I used colSums function to find the sum of missing values from the dataset.  The final result is FALSE as there is no missing value in the dataset. |

***QUESTION 10 :*** Replace the missing values in the dataset with the most repeated value of that field. Check if the missing values were replaced successfully.

|  |
| --- |
| R Code |
| # Replace the missing values with mode (most frequent value) for each column. It is also called as Outlier detection and treatment.  replace\_missing\_val\_with\_mode <- function(x) {  if (is.factor(x)) {  levels <- unique(x)  mode\_level <- levels[which.max(tabulate(match(x, levels)))]  x[is.na(x)] <- mode\_level  } else if (is.numeric(x)) {  x[is.na(x)] <- median(x, na.rm = TRUE)  }  return(x)  }  #Filling data with most frequent value  data\_filled <- as.data.frame(lapply(vehicle\_data, replace\_missing\_val\_with\_mode))  # Checking if all the missing values were replaced in complete data frame  missing\_val\_replaced <- any(is.na(data\_filled))  # Displaying the result after applying required functions  missing\_val\_replaced |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
|  |

|  |
| --- |
| Explanation |
| In this question, a function is created which checks if the input vector is factor or numeric and based on thatit performs operations which are self explanatory based on comments mentioned in code. Then the function replace\_missing\_val\_with\_mode is applied on vehicle\_data to fill the missing values with the most frequent value of the column. Furthermore, we also checked if all the missing values are replaced or not. Then we display the result after applying the functions which returns FALSE as in our dataset, we do not have any missing value so nothing is replaced. |

***QUESTION 11 :*** Find if the dataset has duplicate rows. Remove them, if exist.

|  |
| --- |
| R Code |
| # Check if duplicate rows exists in the dataset  any\_duplicates <- any(duplicated(vehicle\_data))  # Remove duplicate rows if they exist in the dataframe  if (any\_duplicates) {  data\_unique <- unique(vehicle\_data)  } else {  data\_unique <- vehicle\_data  }  # Display the status after checking the dataset if it has duplicate rows  any\_duplicates |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
|  |

|  |
| --- |
| Explanation |
| In this, I used any of duplicated function to check if there are redundant rows in the vehicles dataset. Then only unique rows are selected if duplicates are found or else the data frame is returned. In the end status is returned which is TRUE in our case as in our data few rows are duplicate.  If loop will keep only 1 set of unique record. |

***QUESTION 12 :*** Replace the values of the following attributes:

a Fuel\_Type: “Petrol”: 0, “Diesel”: 1, “CNG”: 2

b Seller\_Type: “Dealer”: 0, “Individual”: 1

c Transmission: “Manual”: 0, “Automatic”: 1

Show the conversion output of the specific attribute.

|  |
| --- |
| R Code |
| # Convert factor variables to character to avoid errors  vehicle\_data$Fuel\_Type <- as.character(vehicle\_data$Fuel\_Type)  vehicle\_data$Seller\_Type <- as.character(vehicle\_data$Seller\_Type)  vehicle\_data$Transmission <- as.character(vehicle\_data$Transmission)  # Check levels before value replacements before performing operations  levels\_before <- levels(vehicle\_data$Fuel\_Type)  levels\_before <- levels(vehicle\_data$Seller\_Type)  levels\_before <- levels(vehicle\_data$Transmission)  # Replacing Fuel\_Type values as per problem statement  vehicle\_data$Fuel\_Type <- replace(vehicle\_data$Fuel\_Type, vehicle\_data$Fuel\_Type == "Petrol", 0)  vehicle\_data$Fuel\_Type <- replace(vehicle\_data$Fuel\_Type, vehicle\_data$Fuel\_Type == "Diesel", 1)  vehicle\_data$Fuel\_Type <- replace(vehicle\_data$Fuel\_Type, vehicle\_data$Fuel\_Type == "CNG", 2)  # Replacing Seller\_Type values as per problem statement  vehicle\_data$Seller\_Type <- replace(vehicle\_data$Seller\_Type, vehicle\_data$Seller\_Type == "Dealer", 0)  vehicle\_data$Seller\_Type <- replace(vehicle\_data$Seller\_Type, vehicle\_data$Seller\_Type == "Individual", 1)  # Replacing Transmission values as per problem statement  vehicle\_data$Transmission <- replace(vehicle\_data$Transmission, vehicle\_data$Transmission == "Manual", 0)  vehicle\_data$Transmission <- replace(vehicle\_data$Transmission, vehicle\_data$Transmission == "Automatic", 1)  # Convert back to factor variables to stay consistent  vehicle\_data$Fuel\_Type <- as.factor(vehicle\_data$Fuel\_Type)  vehicle\_data$Seller\_Type <- as.factor(vehicle\_data$Seller\_Type)  vehicle\_data$Transmission <- as.factor(vehicle\_data$Transmission)  # Check levels after value replacements to stay consistent  levels\_after <- levels(vehicle\_data$Fuel\_Type)  levels\_after <- levels(vehicle\_data$Seller\_Type)  levels\_after <- levels(vehicle\_data$Transmission)  # Display the conversion output for attributes of("Fuel\_Type", "Seller\_Type", "Transmission")  conversed\_output <- vehicle\_data[, c("Fuel\_Type", "Seller\_Type", "Transmission")]  # Show the conversion output as  conversed\_output |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
| (Contd.) |

|  |
| --- |
| Explanation |
| First, we converted factor variables to characters to avoid errors then levels were checked before value replacement. Then as per the problem statement, replacements were done using replace function. Once it is done, had converted back to factor variables to stay consistent.In the end, those columns with data is displayed where replace function was applied. |

***QUESTION 13 :*** Add a new field called ‘Age’, and input the values by using the field Year. Show the output.

|  |
| --- |
| R Code |
| # Saving current year in current\_year variable as integer  current\_year <- as.integer(format(Sys.Date(), "%Y"))  # Calculating 'Age' field based on 'Current Year' and Year Column of Dataset  vehicle\_data$Age <- current\_year - vehicle\_data$Year  # Display the output with the new 'Age' field with "Car\_Name", "Year", "Age" columns  reqd\_output\_with\_age <- vehicle\_data[, c("Car\_Name", "Year", "Age")]  reqd\_output\_with\_age |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
|  |

|  |
| --- |
| Explanation |
| First, current year is saved from system date function. Then Age column is created where difference of current year and Year column of vehicle dataset is updated.  Then data is displayed with the Car\_Name, Year, Age columns to show the operation performed. |

***QUESTION 14*** : Create a new dataset by selecting only the columns “Car\_name”, “Selling\_Price”, “Present\_Price”, and “Kms\_Drive”. Show the output of the new dataset.

|  |
| --- |
| R Code |
| # Creating a new dataset with the selected columns such as "Car\_Name", "Selling\_Price", "Present\_Price", "Kms\_Driven"  reqd\_dataset <- vehicle\_data[, c("Car\_Name", "Selling\_Price", "Present\_Price", "Kms\_Driven")]  # Displaying the output of the new dataset with the selected columns such as "Car\_Name", "Selling\_Price", "Present\_Price", "Kms\_Driven"  reqd\_dataset |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
|  |

|  |
| --- |
| Explanation |
| In this , a new data frame is created with selected columns("Car\_Name", "Selling\_Price", "Present\_Price", "Kms\_Driven") of vehicles dataset. Then in the end, that dataframe is displayed. |

***QUESTION 15:*** Shuffle the rows of the Vehicle dataset randomly and show the output.

|  |
| --- |
| R Code |
| # Shuffling the rows randomly using nrow of sample function  my\_shuffled\_data <- vehicle\_data[sample(nrow(vehicle\_data)), ]  # Displaying the output of the shuffled dataset  my\_shuffled\_data |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
|  |

|  |
| --- |
| Explanation |
| In this Vehicle dataset is shuffled with the help of nrow function of sample. Then shuffled data is displayed. |

**Part 3: Data Visualization**

***QUESTION 16*** : Import the Vehicle dataset. Create a scatter plot of the Selling\_Price Vs Present\_Price. Colour code the points based on the Transmission (5 marks).

a. Add labels, title and colour to the plot. The colour should be red for Transmission type ‘0’ and blue for ‘1’.

b. Add open triangles to the plot.

c. What do you understand from the output (5 marks)?

|  |
| --- |
| R Code |
| # Importing the dataset  vehicle\_data <- read.csv("C:/Users/rahul/Downloads/Vehicle.csv")  # Import the ggplot2 library  library(ggplot2)  # Create scatter plot  my\_scatter\_plot <- ggplot(vehicle\_data, aes(x = Present\_Price, y = Selling\_Price, color = factor(Transmission))) +  geom\_point(shape = 2, size = 3) +  scale\_color\_manual(values = c('Automatic'= "red",'Manual'= "blue")) +  labs(title = "Scatter Plot: Selling\_Price vs Present\_Price", x = "Present\_Price", y = "Selling\_Price")  # Show the scatter plot  print(my\_scatter\_plot) |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
| A graph with red and blue triangles  Description automatically generated with low confidence |

|  |
| --- |
| Explanation |
| **ABOUT CODE:**  In this ggplot library is used to scatter plot the vehicles data based on Present Price and Selling Price.Also for transmission, red color code is used for automatic and blue color is used for manual. Then suitable titles were provided to plot the graph.  **ABOUT GRAPH:**  In this, we are plotting graph with triangles. Here red represents Automatic and blue represents Manual. From the observation of graph, it is evident that present price of Automatic car is getting sold at the price lesser than that whereas the value of selling price and present price is mostly consistent for manual vehicles.   In addition to this, it is also clearly visible that the selling price and present price of Automatic vehicles are mostly high as compared to Manual vehicles. |

***QUESTION 17 :*** Create a box plot of the Selling\_Price Vs Transmission and Fuel\_Type (5 marks).

|  |
| --- |
| R Code |
| # Converting Transmission and Fuel\_Type to factors to avoid errros well in advance  vehicle\_data$Transmission <- as.factor(vehicle\_data$Transmission)  vehicle\_data$Fuel\_Type <- as.factor(vehicle\_data$Fuel\_Type)  # Creating box plot with suitable labels  my\_box\_plot <- ggplot(vehicle\_data, aes(x = Transmission, y = Selling\_Price, fill = Fuel\_Type)) +  geom\_boxplot() +  labs(title = "Box Plot: Selling\_Price VS Transmission and Fuel\_Type(Columns Considered)", x = "Transmission", y = "Selling\_Price")  # Show the box plot  print(my\_box\_plot) |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
| (CONTD.)  A picture containing text, screenshot, diagram, plot  Description automatically generated |

|  |
| --- |
| Explanation |
| In this first we converted Transmission and Fuel\_Type to factors to avoid erros. Then we created box plot on vehicle dataset with Transmission(X Axis), Selling price(Y Axis), the data plotted is of fuel type with geom box plot. Also suitable titles are provided. In the end, box plot is displayed. |

***QUESTION 18 :*** Create a scatter plot of the Selling\_Price Vs Kms\_Driven, and use k-means clustering to cluster the points into 4 clusters. Colour-code based on the cluster they belong to (10 marks).

|  |
| --- |
| R Code |
| # Install stats package  #install.packages("stats")  library(stats)  # Select the columns of interest  reqd\_data <- vehicle\_data[, c("Selling\_Price", "Kms\_Driven")]  # Perform k-means clustering with 4 clusters  k <- 4  kmeans\_clusters <- kmeans(reqd\_data, centers = k)  # Add cluster labels to the dataset  vehicle\_data$Cluster <- kmeans\_clusters$cluster  # Create scatter plot with cluster coloring  my\_scatter\_plot\_cluster <- ggplot(vehicle\_data, aes(x = Selling\_Price, y = Kms\_Driven, color = as.factor(Cluster))) +  geom\_point() +  labs(title = "Scatter Plot: Selling\_Price vs Kms\_Driven with K-means Clustering", x = "Selling\_Price", y = "Kms\_Driven")  # Show the scatter plot  print(my\_scatter\_plot\_cluster) |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
| (CONTD.)  A picture containing text, screenshot, diagram, plot  Description automatically generated |

|  |
| --- |
| Explanation |
| In this, columns ("Selling\_Price", "Kms\_Driven") are selected then K means clustering is applied with 4 clusters. Then cluster labels are applied to the dataset. Next, scatter plot is drawn with cluster coloring with appropriate labels.  From the graph, it is evident that out of 4 clusters, cluster number 1 seems to be an outlier and clustersr number 2 and 3 are quite nearby to each other. |

***QUESTION 19***: Create a scatter plot of the Selling\_Price Vs Present\_Price, and use hierarchical clustering to cluster the points into 3 clusters? Colour-code the points based on the cluster they belong to (10 marks).

|  |
| --- |
| R Code |
| # Performing Hierarchical clustering on Selling\_Price Vs Present\_Price  library(ggplot2)  # Choosing required columns as per problem statement("Selling\_Price", "Present\_Price")  reqd\_data <- vehicle\_data[, c("Selling\_Price", "Present\_Price")]  # Create Scatter plot  plot(reqd\_data$Selling\_Price, reqd\_data$Present\_Price, main = "Scatter Plot of Comparison Selling\_Price and Present\_Price", xlab = "Selling\_Price", ylab = "Present\_Price")  # Calculating the distance matrix  dist\_matrix <- dist(reqd\_data[, c("Selling\_Price", "Present\_Price")])  # Performing the hierarchical clustering on distance matrix  hc <- hclust(dist\_matrix, method = "ward.D2")  # Cutting into 3 clusters  clusters <- cutree(hc, k = 3)  # Defining Color codes to show for points based on clusters  colors <- c("cyan", "yellow", "green")  points(reqd\_data$Selling\_Price, reqd\_data$Present\_Price, pch = 16, col = colors[clusters])  # Adding cluster labels now  text(reqd\_data$Selling\_Price, reqd\_data$Present\_Price, labels = clusters, pos = 3)  # Adding a legend with appropriate legends, col and pch parameters  legend("topright", legend = c("Cluster 1", "Cluster 2", "Cluster 3"), col = colors, pch = 16) |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
| A picture containing text, screenshot, diagram, line  Description automatically generated |

|  |
| --- |
| Explanation |
| * First, required columns are saved in another data frame for further processing. * Then scatter plot diagram is created for the data. * Then Distance matrix is calculated for selling price and present price and further hierarchical clustering is performed on distance matrix. * Then we cut into 3 clusters and 3 color codes are defined. * These colors are applied on points for clusters. * Furthermore, cluster labels, legend, color is applied to the final data visualization. |

***QUESTION 20:*** Add a new field called ‘Age’, and calculate it using the field ‘Year’. Create a barplot for the following fields of the dataset:

a. ‘Age’, ‘Year’, ‘Transmission’, ‘Seller\_Type’, ‘Fuel\_Type’ and ‘Owner’

b. Add labels, titles, and colours to the plot

|  |
| --- |
| R Code |
| # Calculating the Age field by computing difference with the current year  vehicle\_data$Age <- as.integer(format(Sys.Date(), "%Y")) - vehicle\_data$Year  # Creating the bar plot for the calculated column i.e. Age which is computed above  barplot(table(vehicle\_data$Age),main = "Age Distribution of Vehicles with their frequency",xlab = "Age",ylab = "Frequency of Occurence",col = "pink")  # Add color legends to fill with pink  legend("topright",legend = c("Age"),fill = "pink")  # Create bar plot for Year vs the frequency of occurrence  barplot(table(vehicle\_data$Year),main = "Year Distribution wrt Frequency of Vehicles",xlab = "Year",ylab = "Frequency",col = "yellow")  # Add color legend to fill with yellow  legend("topright",legend = c("Year"),fill = "yellow")  # Create bar plot for Transmission vs the frequency of occurrence  barplot(table(vehicle\_data$Transmission),main = "Transmission Distribution wrt Frequency",xlab = "Transmission Type",ylab = "Frequency",col = "cyan")  # Add color legend(Transmission) to fill with cyan  legend("topright",legend = c("Transmission Type"),fill = "cyan")  # Create bar plot for Seller\_Type vs the frequency of occurrence  barplot(table(vehicle\_data$Seller\_Type),main = "Seller Type Distribution wrt Frequency",xlab = "Seller\_Type",ylab = "Frequency",col = "magenta")  # Add color legend(Seller\_Type) to fill with magenta  legend("topright",legend = c("Seller\_Type"),fill = "magenta")  # Create bar plot for Fuel\_Type vs the frequency of occurrence  barplot(table(vehicle\_data$Fuel\_Type),main = "Fuel Type wrt Frequency",xlab = "Fuel\_Type",ylab = "Frequency",col = "red")  # Add color legend(Fuel\_Type) to fill with red  legend("topright",legend = c("Fuel\_Type"),fill = "red")  # Create bar plot for Owner vs the frequency of occurrence  barplot(table(vehicle\_data$Owner),main = "Owner Type and its Frequency of occurence",xlab = "Owner",ylab = "Frequency",col = "brown")  # Add color legend(Owner) to fill with brown  legend("topright",legend = c("Owner"),fill = "brown") |

|  |
| --- |
| Code Screenshot |
| (CONTD.) |

|  |
| --- |
| Output |
| AGE Barplot  A picture containing text, diagram, screenshot, plot  Description automatically generated  YEAR Barplot  A picture containing text, diagram, screenshot, font  Description automatically generated  Transmission Barplot  A picture containing text, screenshot, diagram, number  Description automatically generated  Seller\_Type Barplot  A picture containing text, screenshot, purple, diagram  Description automatically generated  Fuel\_Type Barplot  A picture containing text, screenshot, diagram, font  Description automatically generated  **Owner**  A picture containing text, screenshot, diagram, rectangle  Description automatically generated |

|  |
| --- |
| Explanation |
| First, Age is calculated from current year. Then barplot function is used to plot between one column and its frequency of occurrence. Along with that legend, color, labels were added for 6 columns in sequence as Age, Year, Transmission, Seller Type,Fuel Type, owner.  The output is attached under output section which is above. |

***QUESTION 21:*** Create a correlation plot of the whole dataset variables and explain the output. Do not forget to convert some of the variable’s datatype if required and possible.

|  |
| --- |
| R Code |
| # Conversion of variables(Car\_Name,Fuel\_Type,Transmission,Seller\_Type) to a factor with numerical values as these are string  vehicle\_data$Car\_Name <- as.numeric(factor(vehicle\_data$Car\_Name))  vehicle\_data$Fuel\_Type <- as.numeric(factor(vehicle\_data$Fuel\_Type))  vehicle\_data$Transmission <- as.numeric(factor(vehicle\_data$Transmission))  vehicle\_data$Seller\_Type <- as.numeric(factor(vehicle\_data$Seller\_Type))  # Computing correlation matrix using cor function on vehicle data after conversion of datatype of some columns  my\_correlation\_matrix <- cor(vehicle\_data)  # Create correlation plot as color method  library(corrplot)  corrplot(my\_correlation\_matrix, method = "color", type = "full", tl.cex = 0.7)  # Create correlation plot as number method  corrplot(my\_correlation\_matrix, method = "number", type = "full", tl.cex = 0.7) |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
| A picture containing text, screenshot, diagram, line  Description automatically generated  A picture containing text, number, screenshot, diagram  Description automatically generated |

|  |
| --- |
| Explanation |
| About Code:   * First conversion of a variables(Car\_Name, Fuel\_Type, Transmission, Seller\_Type) to a factor with numerical values as these are string type. Then we compute correlation matrix using cor function on vehicle data after conversion of datatype of some columns. Finally we created correlation plot using color and number.   About Output:   * All the diagonal elements of the correlation matrix will always be 1 as the correlation of a variable with itself is always fully perfect. * Present\_Price and Selling\_Price shows more correlation which is evident with the number method of correlation plot whereas * Seller\_Type and Selling\_Price shows the least correlation. * Similarly, other trends can be observed with the logic, more the value then more it will be related and vice versa. |

***QUESTION 22:*** Create a scatter plot of the Selling\_Price Vs Kms\_Driven, and use DBSCAN clustering to cluster the points into 3 clusters. Color-code based on the cluster they belong to. Add a legend to the plot.

|  |
| --- |
| R Code |
| # Create a subset of the dataset with only the required columns  reqd\_data <- vehicle\_data[, c("Selling\_Price", "Kms\_Driven")]  #Installing and fetching required libraries like dbscan,ggplot2  #install.packages('dbscan')  library('dbscan')  library('ggplot2')  # Applying DBSCAN clustering with eps value of 10000 and MinPts of 3  dbscan\_res <- dbscan(reqd\_data, eps = 10000, MinPts = 3)  # Adding the cluster labels to the considered data  subset\_dataset$Cluster <- dbscan\_res$cluster  # Creating a scatter plot with 3 different colors and 3 labelled clusters  my\_scatter\_plot\_DBSCAN <- ggplot(subset\_dataset, aes(x = Selling\_Price, y = Kms\_Driven, color = factor(subset\_dataset$Cluster))) +  geom\_point() +  labs(title = "Scatter Plot of Selling\_Price vs Kms\_Driven with DBSCAN Clustering",  x = "Selling\_Price", y = "Kms\_Driven") +  scale\_color\_manual(values = c("yellow", "pink", "purple"), labels = c("Cluster 1", "Cluster 2", "Cluster 3")) +  theme\_bw()  # Display the scatter plot with the print function  print(my\_scatter\_plot\_DBSCAN) |

|  |
| --- |
| Code Screenshot |
|  |

|  |
| --- |
| Output |
| A picture containing text, screenshot, diagram, line  Description automatically generated |

|  |
| --- |
| Explanation |
| * Firstly, the required columns("Selling\_Price", "Kms\_Driven") are saved in new data frame. * Then dbscan and ggplot2 is fetched with the library function to use in the subsequent steps. * Furthermore dbscan clustering is applied with the value of eps and MinPts(minimum number of points required to form a cluster) as 10000 and 3 respectively. Then cluster labels were added to the required data. * Finally, scatter plot is displayed with the appropriate legend, title and different color for clusters. |