Project Trimester 1 2025

COMP1013 Analytics Programming

Due Friday 04 April 2025 (Week 11)

Student name: Ngo Le Duy Anh

Student ID: 22106021

Declaration

By including this statement, we the authors of this work, verify that:

• We hold a copy of this assignment that we can produce if the original is lost or damaged.

• We hereby certify that no part of this assignment/product has been copied from any other student's work or from any other source except where due acknowledgement is made in the assignment.

• No part of this assignment/product has been written/produced for us by another person except where such collaboration has been authorised by the subject lecturer/tutor concerned.

• We are aware that this work may be reproduced and submitted to plagiarism detection software programs for the purpose of detecting possible plagiarism (which may retain a copy on its database
for future plagiarism checking).

• We hereby certify that we have read and understand what the School of Computing, Engineering and Mathematics defines as minor and substantial breaches of misconduct as outlined in the learning guide for this unit.

I/ Preparation phase

Install essential packages for the analysis

```
installed.packages("tidyverse")
library("tidyverse")
```

Import dataset into the project

```
Automobile <- read_csv("Automobile.csv")

Engine <- read_csv("Engine.csv")

Maintenance <- read_csv("Maintenance.csv")
```

II/ Assignment completion phase

Question 1

1.1 Inspect data frames that I have imported.

In order to ensure the effective workflow from start to end, I will inspect the data and check whether any duplicates exist in distinct key columns of each table.

```
Automobile #show the data from file Automobile
```

PlateNumber <chr>	Manufactures <chr>	BodyStyles <chr>	DriveWheels <chr>	EngineLocation <chr>	WheelBase <dbl>	Length <dbl>	Width <dbl>	Height <dbl>
53N-001	Alfa-romero	convertible	rwd	front	88.6	168.8	64.1	48.8
53N-002	Alfa-romero	hatchback	rwd	front	94.5	171.2	65.5	52.4
53N-003	Audi	sedan	fwd	front	99.8	176.6	66.2	54.3
53N-004	Audi	sedan	4wd	front	99.4	176.6	66.4	54.3
53N-005	Audi	sedan	fwd	front	99.8	177.3	66.3	53.1
53N-006	Audi	sedan	fwd	front	105.8	192.7	71.4	55.7
53N-007	Audi	wagon	fwd	front	105.8	192.7	71.4	55.7
53N-008	Audi	sedan	fwd	front	105.8	192.7	71.4	55.9
53S-001	Audi	hatchback	4wd	front	99.5	178.2	67.9	52.0
53S-002	Bmw	sedan	rwd	front	101.2	176.8	64.8	54.3

1-10 of 204 rows | 1-9 of 13 columns Previous 1 2 3 4 5 6 ... 21 Next

```
duplicated(Automobile$PlateNumber) #no duplicates exist
```

➔ Automobile table has 204 rows and 13 columns

```
Engine #Show the data from file Engine.csv
```

EngineModel <chr>	EngineType <chr>	NumCylinders <chr>	EngineSize <dbl>	FuelSystem <chr>	Horsepower <chr>	FuelTypes <chr>	Aspiration <chr>
E-0001	dohc	four	130	mpfi	111	gas	std
E-0002	ohcv	six	152	mpfi	154	gas	std
E-0003	ohc	four	109	mpfi	102	gas	std
E-0004	ohc	five	136	mpfi	115	gas	std
E-0005	ohc	five	136	mpfi	110	gas	std
E-0006	ohc	five	131	mpfi	140	gas	turbo
E-0007	ohc	five	131	mpfi	160	gas	turbo
E-0008	ohc	four	108	mpfi	101	gas	std
E-0009	ohc	six	164	mpfi	121	gas	std
E-0010	ohc	six	164	mpfi	121	gas	std

1-10 of 88 rows Previous 1 2 3 4 5 6 … 9 Next

```
duplicated(Engine$EngineModel) #identify 4 duplicates in this table

##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
FALSE
## [13] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
FALSE
## [25] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
FALSE
## [37] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE
FALSE
## [49] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
FALSE
## [61] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
FALSE
## [73] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
FALSE
## [85] FALSE FALSE FALSE  TRUE
```

➔ Engine table has 88 rows and 8 columns with 4 duplicates identified.

```
Engine <- distinct(Engine,EngineModel,.keep_all=TRUE) #remove duplicates
duplicated(Engine$EngineModel) #no duplicates exist
```

```
Maintenance #show the data from Maintenance.csv file
```

ID <dbl>	PlateNumber <chr>	Date <chr>	Troubles <chr>	ErrorCodes <dbl>	Price <dbl>	Methods <chr>
1	53N-001	15/02/2024	Break system	-1	110	Replacement
2	53N-001	16/03/2024	Transmission	-1	175	Replacement
3	53N-001	15/04/2024	Suspected clutch	-1	175	Adjustment
4	53N-001	15/05/2024	Ignition (finding)	1	180	Adjustment
5	53N-001	14/06/2024	Chassis	-1	85	Replacement
6	53N-002	15/02/2024	Cylinders	1	1000	Replacement
7	53N-002	16/03/2024	Ignition (finding)	1	180	Adjustment
8	53N-002	15/04/2024	No error	0	0	NA
9	53N-003	16/04/2024	Loss of driving ability	-1	180	Urgent care
10	53N-004	17/04/2024	Loss of driving ability	-1	180	Urgent care

1-10 of 374 rows Previous 1 2 3 4 5 6 … 38 Next

```
duplicated(Maintenance$ID) #no duplicates identified
```

➔ Maintenance table has 374 rows and 7 columns

1.2 Replace any "?" with NA in dataset

```
#CHECK WHICH TABLE HAS MISSING VALUES
#Firstly, identify missing values in different tables
Auto_missing_check <- Automobile %>%
  filter(if_any(everything(),~.=="?"))

Engine_missing_check <- Engine %>%
  filter(if_any(everything(),~.=="?"))

Maintenance_missing_check <- Maintenance %>%
  filter(if_any(everything(),~.=="?"))
```

Description

In the script above, I used filter in the "Tidyverse" package to identify missing values existing in these tables. However, filter function only categorizes one column identified clearly, whereas the missing values can exist in any columns. Therefore, I added if_any(everything()) function to increase filtering scope with the condition (~.=="?") to make the code scan each row in a table containing "?".

```
#Secondly, check what tables have missing value
Auto_missing_check #There is no missing values
Maintenance_missing_check #There is no missing values
```

```
Engine_missing_check #Found missing values in this table
```

EngineModel <chr>	EngineType <chr>	NumCylinders <chr>	EngineSize <dbl>	FuelSystem <chr>	Horsepower <chr>	FuelTypes <chr>	Aspiration <chr>
E-0011	?	three	60	2bbl	48	gas	std
E-0049	?	four	120	mpfi	97	gas	std
E-0050	?	four	152	idi	95	diesel	turbo
E-0051	?	four	120	mpfi	95	gas	std
E-0052	?	four	134	mpfi	142	gas	turbo
E-0057	ohc	four	132	mpfi	?	gas	std

6 rows

Replace missing value with NA in the Engine table

```
Engine <- Engine %>% mutate(across(everything(),~ifelse(. =="?",NA,.)))
```

Description:

I used #mutate() function in the “tidyverse” package to replace the value in a table. Moreover, #across(everything()) is used in order to make this code applied in a whole table rather than a specific column. #Ifelse() is used instead of #if(), because #if() only returns the first value, while #ifelse() can repeatedly evaluate many values and conduct action when it meets the condition.

Check whether the missing values are replaced or not

```
Engine_missing_check #Before: The old table indicates "?" as MISSING VALUES
```

EngineModel <chr>	EngineType <chr>	NumCylinders <chr>	EngineSize <dbl>	FuelSystem <chr>	Horsepower <chr>	FuelTypes <chr>	Aspiration <chr>
E-0011	?	three	60	2bbl	48	gas	std
E-0049	?	four	120	mpfi	97	gas	std
E-0050	?	four	152	idi	95	diesel	turbo
E-0051	?	four	120	mpfi	95	gas	std
E-0052	?	four	134	mpfi	142	gas	turbo
E-0057	ohc	four	132	mpfi	?	gas	std

6 rows

```
Engine_Check <- Engine %>% filter(EngineModel %in% c("E-0011","E-0049","E-0050", "E-0051","E-0052","E-0057")) #choose engine models that contain "?" before

Engine_Check #After: The new table indicates missing values as NA
```

EngineModel <chr>	EngineType <chr>	NumCylinders <chr>	EngineSize <dbl>	FuelSystem <chr>	Horsepower <chr>	FuelTypes <chr>	Aspiration <chr>
E-0011	*NA*	three	60	2bbl	48	gas	std
E-0049	*NA*	four	120	mpfi	97	gas	std
E-0050	*NA*	four	152	idi	95	diesel	turbo
E-0051	*NA*	four	120	mpfi	95	gas	std
E-0052	*NA*	four	134	mpfi	142	gas	turbo
E-0057	ohc	four	132	mpfi	*NA*	gas	std

6 rows

1.3 Convert categorical variables BodyStyles, FuelTypes, ErrorCodes to factors

```
Automobile$BodyStyles <- as.factor(Automobile$BodyStyles)

Engine$FuelTypes <- as.factor(Engine$FuelTypes)

Maintenance$ErrorCodes <- as.factor(Maintenance$ErrorCodes)
```

Description

"as.factor" is a function that transform the data type from any types to Factor

```
#Check the datatype of each variable: All variables are converted to factors
str(Automobile$BodyStyles)

##  Factor w/ 5 levels "convertible",..: 1 3 4 4 4 4 5 4 3 4 ...


str(Engine$FuelTypes)

##  Factor w/ 2 levels "diesel","gas": 2 2 2 2 2 2 2 2 2 2 ...


str(Maintenance$ErrorCodes)

##  Factor w/ 3 levels "-1","0","1": 1 1 1 3 1 3 3 2 1 1 ...
```

1.4 Replace the missing value in Horsepower column with mean of Horsepower

```
Engine$Horsepower <- as.numeric(Engine$Horsepower) #change the data type
first
mean_Horsepower <- mean(Engine$Horsepower,na.rm=TRUE) #calculate the mean of
Horsepower column

mean_Horsepower <- round(mean_Horsepower,digits=2) #limit the digit of result


Engine <- Engine %>% mutate(Horsepower =
replace(Horsepower,is.na(Horsepower),mean_Horsepower)) #replace the NA value
```

```
#Check the result
checkNA_horsepower <- Engine_Check %>% filter(is.na(Horsepower))
checkNA_horsepower #Old table: NA still indicates in old table
```

EngineModel <chr>	EngineType <chr>	NumCylinders <chr>	EngineSize <dbl>	FuelSystem <chr>	Horsepower <chr>	FuelTypes <chr>	Aspiration <chr>
E-0057	ohc	four	132	mpfi	*NA*	gas	std

1 row

```
Check_New_Engine <- Engine %>% filter(EngineModel=="E-0057")
Check_New_Engine #New table: NA is replaced with the mean of Horsepower
```

EngineModel <chr>	EngineType <chr>	NumCylinders <chr>	EngineSize <dbl>	FuelSystem <chr>	Horsepower <dbl>	FuelTypes <fctr>	Aspiration <chr>
E-0057	ohc	four	132	mpfi	113.06	gas	std

1 row

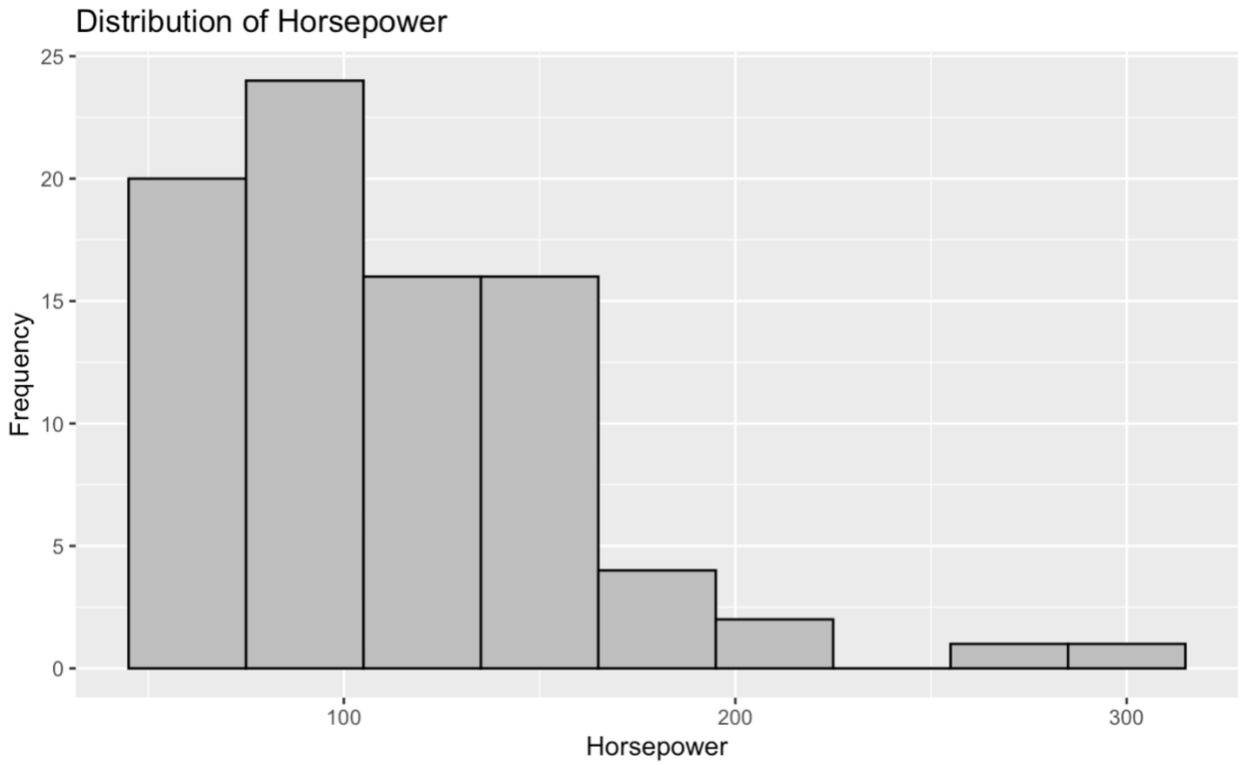
1.5 Display Horsepower distribution

To display the Horsepower distribution, I used histogram chart to make the distribution visualization transparent.

```
sqrt(nrow(Engine)) #calculate the number of bins.
```

```
## [1] 9.165151
```

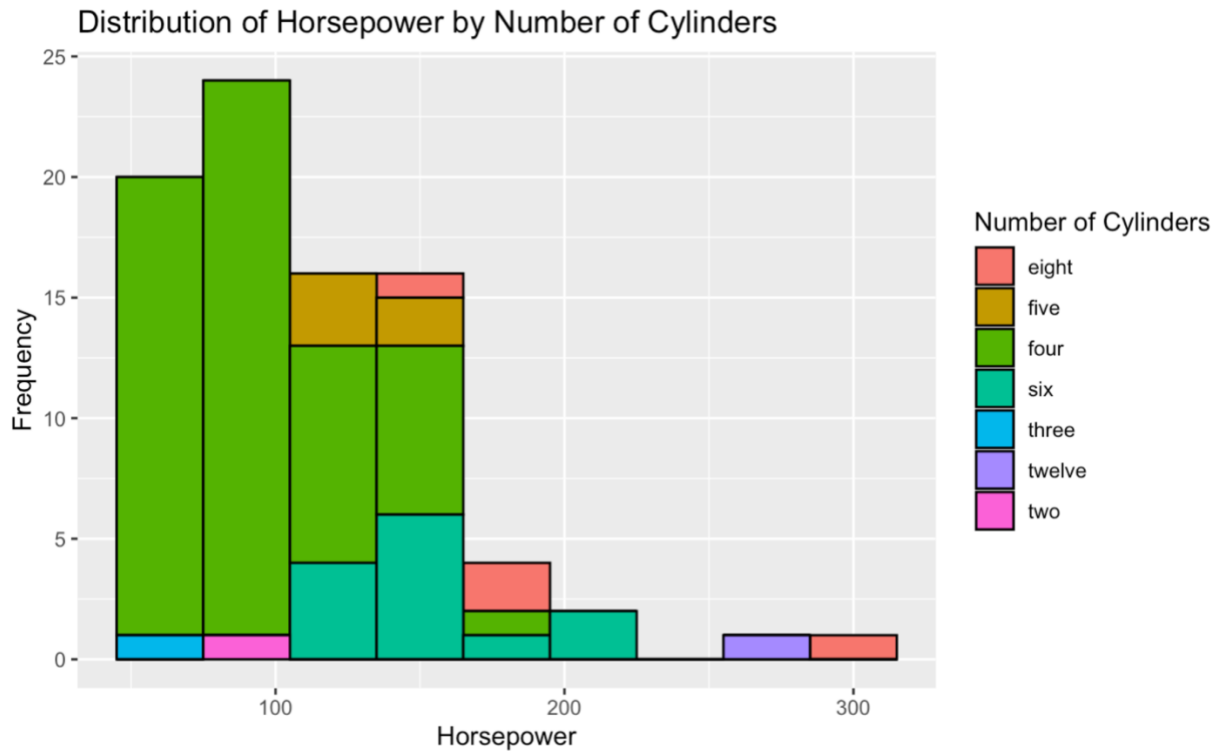
```r
ggplot(data=Engine, mapping=aes(x=Horsepower))+ #set up data and aesthetics for chart
    geom_histogram(bins=9,color="black",fill="gray")+ #set up design of the chart
    labs(title = "Distribution of Horsepower",x="Horsepower",y="Frequency")
#change and add name for a chart
```



Question 2

2.1 Analysis of Horsepower distribution across the number of cylinders

```r
ggplot(data= Engine,mapping=aes(x=Horsepower,fill=factor(NumCylinders)))+
#fill() to color the value of horsepower by each cylinder number, factor() is used to change the data type of this column to factor.
  geom_histogram(bins=9, color="black")+ #set up design of the chart
  labs(title = "Distribution of Horsepower by Number of Cylinders",
       x="Horsepower",y="Frequency",fill="Number of Cylinders")
```



Description

Instead of writing code for analysis and visualization separately, I chose to combine them into a single code block and visualized the pattern of horsepower distribution by cylinders as a histogram for straightforward analysis and understanding.

Analysis

The Horsepower distribution was skewed right. Overall, most engines frequently had a horsepower capacity lower than 200 hp, which was also regarded as the most popular range for vehicles. Among the number of cylinders, engines with 4 cylinders tend to have the horsepower ranging from 50 to 150 hp. In the medium range from 100 to 200 hp, 6-cylinder engines were distributed widely. Meanwhile, in the higher horsepower range, 8-cylinder engines showed a higher distribution from 150 to over 300 horsepower. Additionally, 2-,3- and 5-cylinders were regarded as rare engine that provided low horsepower capacity as 4 cylinders, similarly, 12-cylinder engine was also a scarce option in the high-range level with nearly 300 hp. In conclusion, the distribution data showcased that 4-cylinder engines dominated the low-capacity segment for everyday vehicles, while 6- and 8- cylinder engines were more ubiquitous in mid- and high-performance cars.

2.2 Distribution of the horsepower across the groups of the engine sizes (e.g., 60-100, 101-200, 201-300, 301+).

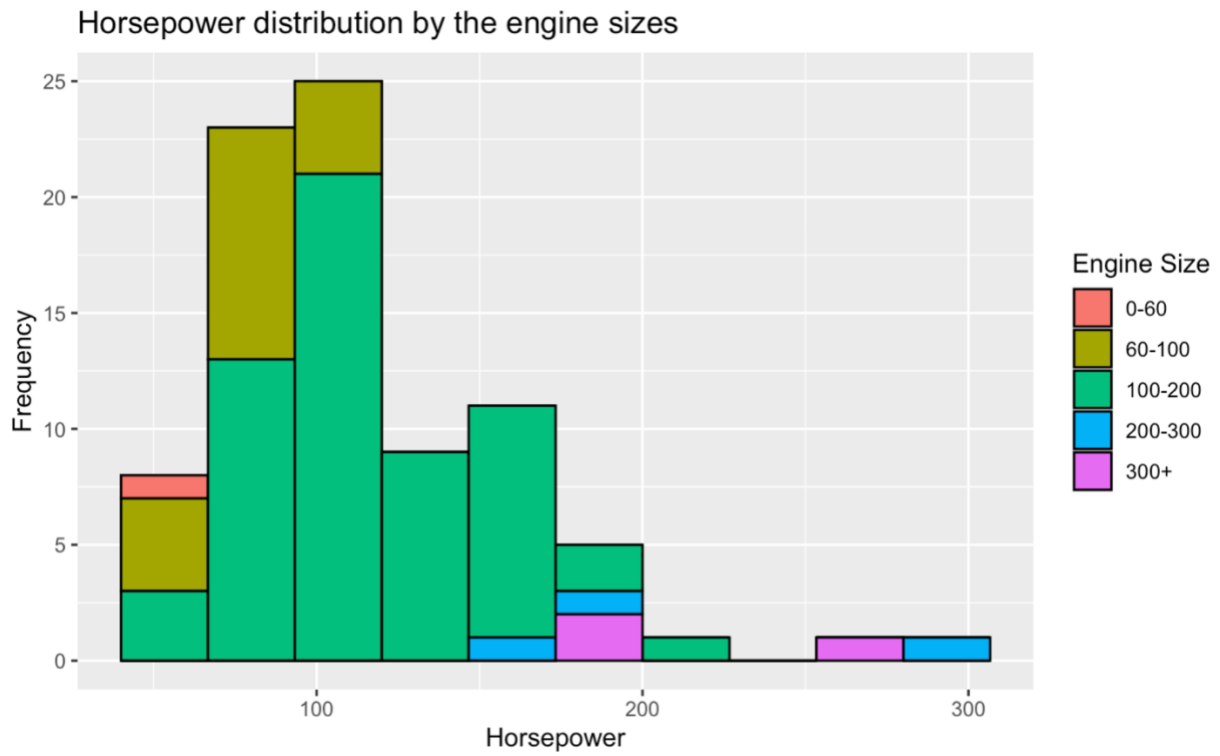
```r
#Create a new column that is categorized by engine size groups
Engine <- Engine %>% mutate( EngineGroup = cut(EngineSize,

  breaks=c(0,60,100,200,300,Inf),   #define the range for categorizing
  labels=c("0-60","60-100","100-200","200-300","300+"))) %>%
           group_by(EngineGroup) #group data by the EngineGroup column
Engine #check the new column named EngineGroup
```

EngineType <chr>	NumCylinders <chr>	EngineSize <dbl>	FuelSystem <chr>	Horsepower <dbl>	FuelTypes <fctr>	Aspiration <chr>	EngineGroup <fctr>
dohc	four	130	mpfi	111.00	gas	std	100-200
ohcv	six	152	mpfi	154.00	gas	std	100-200
ohc	four	109	mpfi	102.00	gas	std	100-200
ohc	five	136	mpfi	115.00	gas	std	100-200
ohc	five	136	mpfi	110.00	gas	std	100-200
ohc	five	131	mpfi	140.00	gas	turbo	100-200
ohc	five	131	mpfi	160.00	gas	turbo	100-200
ohc	four	108	mpfi	101.00	gas	std	100-200
ohc	six	164	mpfi	121.00	gas	std	100-200
ohc	six	164	mpfi	121.00	gas	std	100-200

1-10 of 84 rows | 2-9 of 9 columns Previous 1 2 3 4 5 6 … 9 Next

```r
ggplot(data = Engine, mapping=aes(x=Horsepower,fill=factor(EngineGroup)))+
#define main parameters of chart
  geom_histogram(bins=10,color="black")+ #set the specification for the chart
  labs(title="Horsepower distribution by the engine sizes",
       x="Horsepower",y="Frequency",fill="Engine Size") # add details for the
histogram chart
```



Description

To create a histogram of horsepower distribution by the groups of engine sizes, I firstly had to create a column named “EngineGroup” that categorized each value in the Horsepower column into different groups as the question required. Therefore, I used cut() to categorize value in the EngineSize column into different group as the code above. Then, I grouped the data by EngineGroup column for creating chart after that.

Analysis

The Horsepower distribution by the engine sizes was skewed right. Overall, engine in the group 100-200 was the most popular size in vehicles. Particularly, this engine size could have the horsepower ranging from 50 to roundly 200 hp, which gave the wide range of capacity from low to medium. Besides, engine size in the 60-100 group was also the second popular group, which had lower frequency compared to the first one. Moreover, this group was distributed more in the the low level of horsepower, ranging from 50 to approximately 150. Additionally, the low range also contained group 0-60, regarded as a rare size that had the lowest horsepower at 50 hp. Meanwhile, from the medium- to high-capacity level, there were 2 engine groups, including 200-300, and over 300, which were also unpopular sizes distributed from 150 to 300 hp, and roughly 200 to 250, respectively. In conclusion, engine size 100-200 was the most popular size for normal vehicles with the low (~50 hp) to medium (~200 hp) capacity, while other groups, such as 0-60 or 60-100, were less used in this range. In the medium-high range, this was also a rare range for a typical everyday vehicle with engine size 200-300 and over 300. However, since they had

high horsepower capacity (from 200 to over 300 hp), they tend to be used for high-performance vehicles.

Question 3

3.1 Filter out those engines in the dataset that have trouble or are suspected of having trouble

```
engine_trouble <- Maintenance %>% filter((ErrorCodes=="1")) #filter based on condition
engine_trouble #check the result, the new table is completely created
```

ID <dbl>	PlateNumber <chr>	Date <chr>	Troubles <chr>	ErrorCodes <fctr>	Price <dbl>	Methods <chr>
4	53N-001	15/05/2024	Ignition (finding)	1	180	Adjustment
6	53N-002	15/02/2024	Cylinders	1	1000	Replacement
7	53N-002	16/03/2024	Ignition (finding)	1	180	Adjustment
12	53N-004	18/06/2024	Ignition (finding)	1	180	Adjustment
14	53N-004	19/08/2024	Cylinders	1	1000	Replacement
15	53N-005	20/08/2024	Noise (finding)	1	10	Adjustment
18	53N-008	19/01/2024	Ignition	1	180	Adjustment
20	53N-008	19/05/2024	Cylinders	1	1000	Replacement
21	53N-008	19/07/2024	Noise (finding)	1	10	Adjustment
26	53S-002	19/05/2024	Oil filter	1	90	Replacement

1-10 of 182 rows Previous 1 2 3 4 5 6 … 19 Next

Description

According to the question, ErrorCodes can display the status of vehicle, so ErrorCodes having “1”, which was for engine trouble vehicles, would be filtered out into another table.

3.2 what are the top 5 most common troubles related to the engines?

```
top_5_engine_trouble <- engine_trouble %>% select(Troubles) %>% #take the Troubles column from data engine_trouble
  group_by(Troubles) %>% #group Troubles column
  summarise(Times= n()) %>% #add another column that calculate the sum of duplicates of each group
  arrange(desc(Times)) %>% slice_head(n=5) #arrange data in descending order and extract top 5

top_5_engine_trouble #show the top 5 problems related to engines
```

Troubles <chr>	Times <int>
Cylinders	38
Ignition (finding)	22
Noise (finding)	19
Valve clearance	15
Fans	13

5 rows

Description

To create the top 5 problems in the engine, I grouped the values in the Troubles column and then created a new column that calculated the number of values in each group and arranged it in descending order. Finally, I only took the first five values and saved it into a new table named “top_5_engine_trouble”.

Result

There are 5 main problems happened frequently in engines, including cylinders, ignition, noise, valve clearance, and fans.

3.3 Do the troubles differ between fuel types?

```
engine_by_fuel <- engine_trouble %>% left_join(Automobile, by="PlateNumber")
#combine first two tables (engine_trouble & Automobile)
engine_by_fuel #check the first combination of Automobile and engine_trouble data
```

ID <dbl>	PlateNumber <chr>	Date <chr>	Troubles <chr>	ErrorCodes <fctr>	Price <dbl>	Methods <chr>	Manufactures <chr>
4	53N-001	15/05/2024	Ignition (finding)	1	180	Adjustment	Alfa-romero
6	53N-002	15/02/2024	Cylinders	1	1000	Replacement	Alfa-romero
7	53N-002	16/03/2024	Ignition (finding)	1	180	Adjustment	Alfa-romero
12	53N-004	18/06/2024	Ignition (finding)	1	180	Adjustment	Audi
14	53N-004	19/08/2024	Cylinders	1	1000	Replacement	Audi
15	53N-005	20/08/2024	Noise (finding)	1	10	Adjustment	Audi
18	53N-008	19/01/2024	Ignition	1	180	Adjustment	Audi
20	53N-008	19/05/2024	Cylinders	1	1000	Replacement	Audi
21	53N-008	19/07/2024	Noise (finding)	1	10	Adjustment	Audi
26	53S-002	19/05/2024	Oil filter	1	90	Replacement	Bmw

1-10 of 182 rows | 1-8 of 19 columns Previous 1 2 3 4 5 6 … 19 Next

```r
engine_by_fuel_2 <- engine_by_fuel %>% left_join(Engine, by="EngineModel")
#combine the first table with Engine table using Left_join()
engine_by_fuel_2 #check the final combination between 3 tables
```

ID <dbl>	PlateNumber <chr>	Date <chr>	Troubles <chr>	ErrorCodes <fctr>	Price <dbl>	Methods <chr>	Manufactures <chr>
4	53N-001	15/05/2024	Ignition (finding)	1	180	Adjustment	Alfa-romero
6	53N-002	15/02/2024	Cylinders	1	1000	Replacement	Alfa-romero
7	53N-002	16/03/2024	Ignition (finding)	1	180	Adjustment	Alfa-romero
12	53N-004	18/06/2024	Ignition (finding)	1	180	Adjustment	Audi
14	53N-004	19/08/2024	Cylinders	1	1000	Replacement	Audi
15	53N-005	20/08/2024	Noise (finding)	1	10	Adjustment	Audi
18	53N-008	19/01/2024	Ignition	1	180	Adjustment	Audi
20	53N-008	19/05/2024	Cylinders	1	1000	Replacement	Audi
21	53N-008	19/07/2024	Noise (finding)	1	10	Adjustment	Audi
26	53S-002	19/05/2024	Oil filter	1	90	Replacement	Bmw

1-10 of 182 rows | 1-8 of 27 columns Previous 1 2 3 4 5 6 ... 19 Next

```r
trouble_by_fuel <- engine_by_fuel_2 %>% select(Troubles,FuelTypes) #create a different table that take selected columns inside that
trouble_by_fuel #check the troubles filtered by fuel types
```

Troubles <chr>	FuelTypes <fctr>
Ignition (finding)	gas
Cylinders	gas
Ignition (finding)	gas
Ignition (finding)	gas
Cylinders	gas
Noise (finding)	gas
Ignition	gas
Cylinders	gas
Noise (finding)	gas
Oil filter	gas

1-10 of 182 rows Previous 1 2 3 4 5 6 ... 19 Next

Description

- To filter the troubles by fuel types, I had to combine the first two tables (the engine_trouble and Automobile table), which have a similar key column (PlateNumber). After that, I joined it with the second table (Engine), which allowed me to extract the values of FuelTypes column using the key column (EngineModel).
- Using left_join() helped me to extract only the values for engines with defined troubles, instead of all values.

- Creating a different table named “trouble_by_fuel” allowed me to take only essential columns, which are clearer than a table full of 27 columns.

3.4 Provide a table to rank the top 5 troubles for diesel and gas engines separately.

```
top_5_diesel_trouble <- trouble_by_fuel %>% filter(FuelTypes=="diesel") %>%
  select(Troubles) %>% #take the Troubles column only
  group_by(Troubles) %>% #group data in the Troubles column
  summarise(TroubleCount = n()) %>% #add another column calculate each
Trouble group
  arrange(desc(TroubleCount)) %>% slice_head(n=5) #arrange in descending
order and extract first 5 rows
top_5_diesel_trouble # check top 5 trouble in diesel engine
```

Troubles <chr>	TroubleCount <int>
Cam shaft	3
Cylinders	3
Crank shaft	2
Stroke	2
ECU's power	1

5 rows

```
top_5_gas_trouble <- trouble_by_fuel %>% filter(FuelTypes=="gas") %>%
  select(Troubles) %>% #only take the Troubles column
  group_by(Troubles) %>% #group data in Troubles column
  summarise(TroubleCount = n()) %>% #add a column that count each data in
group
  arrange(desc(TroubleCount)) %>% slice_head(n=5) #arrange in descending
order and extract first 5 rows
top_5_gas_trouble #Check top 5 trouble in gas engine
```

Troubles <chr>	TroubleCount <int>
Cylinders	35
Ignition (finding)	21
Noise (finding)	18
Valve clearance	15
Fans	13

5 rows

Conclusion for question 3.3 and 3.4

The engine troubles, in general, had one similar problem, which was cylinders with the cases ranked first and second for both fuel types (gas & diesel). However, the rest of

problems differed between the two fuel types. In particular, while popular issues for the diesel engines were cam shaft, crank shaft, stroke issues, and ECU power, the most common problems in gas engines were ignition, noise, valve clearance, and fans.

Question 4

Target description

Based on the question, I chose 2 of factors, including BodyStyles, and EngineGroup, to analyze the features that each maintenance method (Urgent care, Adjustment, Replacement) frequently had.

4.1 Conduct combining factors from 3 tables that may have effect on the maintenance methods.

```
# JOINING 3 TABLES
error_vehicle <- Maintenance %>% filter((ErrorCodes=="1")|(ErrorCodes=="-1"))
#create table of vehicles that have troubles related to both engine and
component

error_vehicle_comp1 <- error_vehicle %>%
left_join(Automobile,by="PlateNumber") #combine the first two tables
(engine_trouble & Automobile)


error_vehicle_comp2 <- error_vehicle_comp1 %>%

left_join(Engine, by="EngineModel") %>%  #combine the first table with Engine
table using left_join()
select(Methods,Troubles,Manufactures,BodyStyles,DriveWheels,EngineModel,NumCy
linders,EngineGroup) #only take factors might affect maintenance methods


error_vehicle_comp2 #check factors that might affect maintenance methods
```

Methods <chr>	Troubles <chr>	Manufactures <chr>	BodyStyles <fctr>	DriveWheels <chr>	EngineModel <chr>	NumCylinders <chr>
Replacement	Break system	Alfa-romero	convertible	rwd	E-0001	four
Replacement	Transmission	Alfa-romero	convertible	rwd	E-0001	four
Adjustment	Suspected clutch	Alfa-romero	convertible	rwd	E-0001	four
Adjustment	Ignition (finding)	Alfa-romero	convertible	rwd	E-0001	four
Replacement	Chassis	Alfa-romero	convertible	rwd	E-0001	four
Replacement	Cylinders	Alfa-romero	hatchback	rwd	E-0002	six
Adjustment	Ignition (finding)	Alfa-romero	hatchback	rwd	E-0002	six
Urgent care	Loss of driving ability	Audi	sedan	fwd	E-0003	four
Urgent care	Loss of driving ability	Audi	sedan	4wd	E-0004	five
Adjustment	Suspected clutch	Audi	sedan	4wd	E-0004	five

1-10 of 346 rows | 1-7 of 8 columns Previous 1 2 3 4 5 6 ... 35 Next

4.2 Analyze 2 features (BodyStyle, EngineGroup) affecting each maintenance method (Urgent care, Replacement, Adjustment)

Urgent Care method analysis

```
UrgentCare_BodyStyles <- error_vehicle_comp2 %>%
  filter(Methods =="Urgent care") %>% #Only take 1 method
  select(BodyStyles) %>% #Find pattern of BodyStyles in this method
  group_by(BodyStyles) %>% #Group data for counting values in the next column
  summarise(Count= n()) %>% #Create a column that count value in each
category
  mutate(Percent = round(Count/sum(Count)*100,digits=2 )) %>% #Calculate the
% of each category
  arrange(desc(Percent)) #Arrange it in descending order to showcase the
pattern

UrgentCare_EngineGroup <- error_vehicle_comp2 %>%
  filter(Methods =="Urgent care") %>% #Only take 1 method
  select(EngineGroup) %>% #Find pattern of EngineGroup in this method
  group_by(EngineGroup) %>% #Group data for counting values in the next
column
  summarise(Count= n()) %>% #Create a column that count value in each
category
  mutate(Percent = round(Count/sum(Count)*100, digits =2)) %>% #Calculate the
% of each category
  arrange(desc(Percent)) #Arrange it in descending order to showcase the
pattern
```

```
UrgentCare_BodyStyles #check body style pattern of the urgent care method
```

BodyStyles <fctr>	Count <int>	Percent <dbl>
sedan	14	51.85
hatchback	9	33.33
hardtop	2	7.41
wagon	2	7.41

4 rows

```
UrgentCare_EngineGroup #check engine range pattern of the urgent care method
```

EngineGroup <fctr>	Count <int>	Percent <dbl>
100-200	16	59.26
60-100	10	37.04
300+	1	3.70

3 rows

Urgent care findings

- Body Styles: Trouble vehicles that are maintained through Urgent Care method tend to be sedan and hatchback cars. In particular, 51.85 percent of vehicles maintained are Sedans, while 33.3 percent of it are Hatchback designs. In contrast, hardtop and wagon cars are only recorded a few cases under this method.

- Engine Group: In Urgent Care method, there are frequently everyday vehicles ranging from low to medium engine size. Specifically, maintenance cases appear more in medium engine group (100-200) with the rate of 59,26 percent, and low engine group (60-100) with the rate of 37.04 percent.

Replacement method analysis

```
Replacement_BodyStyles <- error_vehicle_comp2 %>%
  filter(Methods =="Replacement") %>% #Only take 1 method
  select(BodyStyles) %>% #Find pattern of BodyStyles in this method
  group_by(BodyStyles) %>% #Group data for counting values in the next column
  summarise(Count= n()) %>% #Create a column that count value in each
category
  mutate(Percent = round(Count/sum(Count)*100,digits=2 )) %>% #Calculate the
% of each category
  arrange(desc(Percent)) #Arrange it in descending order to showcase the
pattern

Replacement_EngineGroup <- error_vehicle_comp2 %>%
  filter(Methods =="Replacement") %>% #Only take 1 method
  select(EngineGroup) %>% #Find pattern of EngineGroup in this method
  group_by(EngineGroup) %>% #Group data for counting values in the next
```

```
column
  summarise(Count= n()) %>% #Create a column that count value in each
category
  mutate(Percent = round(Count/sum(Count)*100, digits =2)) %>% #Calculate the
% of each category
  arrange(desc(Percent)) #Arrange it in descending order to showcase the
pattern

Replacement_BodyStyles #check body style pattern of the replacement method
```

BodyStyles <fctr>	Count <int>	Percent <dbl>
sedan	89	47.34
hatchback	63	33.51
wagon	24	12.77
convertible	8	4.26
hardtop	4	2.13

5 rows

```
Replacement_EngineGroup #check engine range pattern of the replacement method
```

EngineGroup <fctr>	Count <int>	Percent <dbl>
100-200	120	63.83
60-100	60	31.91
200-300	7	3.72
300+	1	0.53

4 rows

Replacement findings:

- Body Styles: With Replacement method, there are more cases recorded than the Urgent Care method, particularly 5 different body styles. However, similarly to the previous one, Sedans and Hatchbacks dominate maintenance cases with 47.34 and 33.51 percent, respectively. Moreover, Wagon cars are the third frequent cases under this method, at 12.77 percent, which is in contrast with other body styles, such as Convertible (4.26%) and Hardtop (2.13%).
- Engine Group: Engine size Group that are prevalent in this method are 100-200 and 60-100 group, which is considered low to medium engine group. Particularly, 63.83 percent of vehicles belongs to 100-200 group, and 31.91 percent of them is in 60-100 group. On the other hand, 200-300 and over 300 group are rarely seen with low percentage, with 3.72 and 0.53 percent, respectively.

Adjustment method analysis

```
Adjustment_BodyStyles <- error_vehicle_comp2 %>%
  filter(Methods =="Adjustment") %>% #Only take 1 method
  select(BodyStyles) %>% #Find pattern of BodyStyles in this method
  group_by(BodyStyles) %>% #Group data for counting values in the next column
  summarise(Count= n()) %>% #Create a column that count value in each
category
  mutate(Percent = round(Count/sum(Count)*100,digits=2 )) %>% #Calculate the
% of each category
  arrange(desc(Percent)) #Arrange it in descending order to showcase the
pattern

Adjustment_EngineGroup <- error_vehicle_comp2 %>%
  filter(Methods =="Adjustment") %>% #Only take 1 method
  select(EngineGroup) %>% #Find pattern of EngineGroup in this method
  group_by(EngineGroup) %>% #Group data for counting values in the next
column
  summarise(Count= n()) %>% #Create a column that count value in each
category
  mutate(Percent = round(Count/sum(Count)*100, digits =2)) %>% #Calculate the
% of each category
  arrange(desc(Percent)) #Arrange it in descending order to showcase the
pattern

Adjustment_BodyStyles #check body style pattern of the adjustment method
```

BodyStyles <fctr>	Count <int>	Percent <dbl>
sedan	59	45.04
hatchback	48	36.64
wagon	15	11.45
convertible	7	5.34
hardtop	2	1.53

5 rows

```
Adjustment_EngineGroup #check engine range pattern of the adjustment method
```

EngineGroup <fctr>	Count <int>	Percent <dbl>
100-200	75	57.25
60-100	48	36.64
200-300	6	4.58
0-60	1	0.76
300+	1	0.76

5 rows

Adjustment findings

- Body Styles: Sedans and Hatchbacks still dominate cases with 45.04 and 36.64 percent of vehicles, respectively. Furthermore, similarly to Replacement method, Wagon cars are the third frequent cases with 11.45 percent, which is significantly higher than Convertible (5.34%) and Hardtop (1.53%) cars.
- Engine Group: The engine groups tend to concentrate on low and medium sizes with 57.25 percent belong to 100-200 group, and 36.64 percent are in 60-100 group. In contrast, other engine groups, such as 200-300, 0-60, or over 300, are recorded less rate, at 4.58, 0.76, and 0.76 percent, respectively.