WESTERN SYDNEY UNIVERSITY

SCHOOL of: Computer, Data and Mathematical Sciences

ASSIGNMENT COVER SHEET

STUDENT DETAILS

Name: Mai Tien Duc **Student ID:** 22201017

SUBJECT AND TUTORIAL DETAILS

Subject Name: Analytics Programming **Subject code:** COMP 1013

Tutorial Group: None **Day:** Sunday **Time:** 15:30 - 18:45

Lecturer or Tutor name: Assoc. Prof. Nguyen Tan Luy

ASSIGNMENT DETAILS

Title: Assignment **Length:** 21 pages

Due Date: 21/11/2025 **Date submitted:** 21/11/2025

Home campus: Vietnam

DECLARATION

By submitting your work using this link you are certifying that:

☒ You hold a copy of this submission if the original is lost or damaged.

☒ No part of this submission has been copied from any other student's work or from any other third party (including generative AI) except where due acknowledgment is made in the submission.

☒ No part of this submission has been submitted by you in another (previous or current) assessment, except where appropriately referenced, and with prior permission from the teacher/tutor/supervisor/ Subject Coordinator for this subject.

☒ No part of this submission has been written/produced for you by any other person or technology except where collaboration has been authorised by the teacher/tutor/ supervisor/Subject Coordinator either in the assessment resources section of the Learning Guide for this assessment task, in the instructions for this assessment task, or through vUWS.

☒ You are aware that this submission will be reproduced and submitted to detection software programs for the purpose of investigating possible breaches of the Student Misconduct Rule, for example, plagiarism, contract cheating, or unauthorised use of generative AI. Turnitin or other tools of investigation may retain a copy of the submission for the purposes of future investigation.

☒ You will not make this submission available to any other person unless required by the University.

***Instructions**: Please complete the requested details in the form, save it and convert to PDF before adding your signature below*

Student signature: Duc

Note: An examiner or lecturer/tutor has the right to not mark this assignment if the above declaration has not been completed. Staff may contact you for permission to share a de-identified extract or copy of your submission with students or staff for teaching purposes, following guidelines for requesting and sharing exemplar assessment tasks.

Analytics Programming - Assignment

2025-11-17

Table of Contents

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. ***

Question 1: Write the code to inspect the data structure and present the data: The missing values in the dataset were written as "?", replace any "?" with NA; Write code to check: after replacing, how many rows were affected in total? Does this change alter the data distribution?; Convert categorical variables BodyStyles, FuelTypes, ErrorCodes to factors; Replace the missing values in column Horsepower with the median horsepower; Select the appropriate chart type and display: horsepower distribution.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library("ggplot2")
```

```
engine.data = read.csv("Engine.csv")
automobile.data <- read.csv("Automobile (1).csv")
maintenance.data = read.csv("Maintenance (1).csv")
```

```
summary(engine.data)
```

```
##  EngineModel         EngineType        NumCylinders         EngineSize
##  Length:88          Length:88          Length:88          Min.   : 60.0
##  Class :character   Class :character   Class :character   1st Qu.:108.0
##  Mode  :character   Mode  :character   Mode  :character   Median :121.0
##                                                           Mean   :134.1
##                                                           3rd Qu.:151.2
##                                                           Max.   :320.0
##   FuelSystem         Horsepower         FuelTypes          Aspiration
##  Length:88          Length:88          Length:88          Length:88
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
```

```
summary(automobile.data)
```

```
##  PlateNumber        Manufactures        BodyStyles         DriveWheels
##  Length:204         Length:204         Length:204         Length:204
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
```

```
##
##  EngineLocation       WheelBase          Length          Width
##  Length:204         Min.   : 86.60   Min.   :141.1   Min.   :60.30
##  Class :character   1st Qu.: 94.50   1st Qu.:166.3   1st Qu.:64.08
##  Mode  :character   Median : 97.00   Median :173.2   Median :65.50
##                     Mean   : 98.81   Mean   :174.1   Mean   :65.92
##                     3rd Qu.:102.40   3rd Qu.:183.2   3rd Qu.:66.90
##                     Max.   :120.90   Max.   :208.1   Max.   :72.30
##      Height        CurbWeight   EngineModel           CityMpg
##  Min.   :47.80   Min.   :1488   Length:204         Min.   :10.00
##  1st Qu.:52.00   1st Qu.:2145   Class :character   1st Qu.:19.00
##  Median :54.10   Median :2414   Mode  :character   Median :24.00
##  Mean   :53.75   Mean   :2556                      Mean   :25.23
##  3rd Qu.:55.50   3rd Qu.:2939                      3rd Qu.:30.00
##  Max.   :59.80   Max.   :4066                      Max.   :50.00
##    HighwayMpg
##  Min.   :15.00
##  1st Qu.:25.00
##  Median :30.00
##  Mean   :30.76
##  3rd Qu.:34.50
##  Max.   :55.00
```

```
summary(maintenance.data)
```

```
##        ID          PlateNumber           Date             Troubles
##  Min.   :  1.00   Length:374         Length:374         Length:374
##  1st Qu.: 94.25   Class :character   Class :character   Class :character
##  Median :187.50   Mode  :character   Mode  :character   Mode  :character
##  Mean   :187.50
##  3rd Qu.:280.75
##  Max.   :374.00
##    ErrorCodes           Price          Methods
##  Min.   :-1.00000   Min.   :   0.0   Length:374
##  1st Qu.:-1.00000   1st Qu.:  85.0   Class :character
##  Median : 0.00000   Median : 120.0   Mode  :character
##  Mean   : 0.04813   Mean   : 204.8
##  3rd Qu.: 1.00000   3rd Qu.: 180.0
##  Max.   : 1.00000   Max.   :1000.0
```

- Before exploring the data structure, import the three datasets engine, automobile, and maintenance using `read.csv` (the files are in .csv format). Use `head()` to preview the first few rows, and `ls.str()` to inspect each variable's data type. Determining whether a variable is character or numeric is important for choosing the correct computational functions.

```
engine.data[engine.data == "?"] = NA
na.engine = sum(is.na(engine.data$Horsepower))

automobile.data[automobile.data == "?"] = NA
na.automobile = sum(is.na(automobile.data))

maintenance.data[maintenance.data == "?"] = NA
na.maintenance = sum(is.na(maintenance.data))

sum(na.engine,na.automobile,na.maintenance)
```

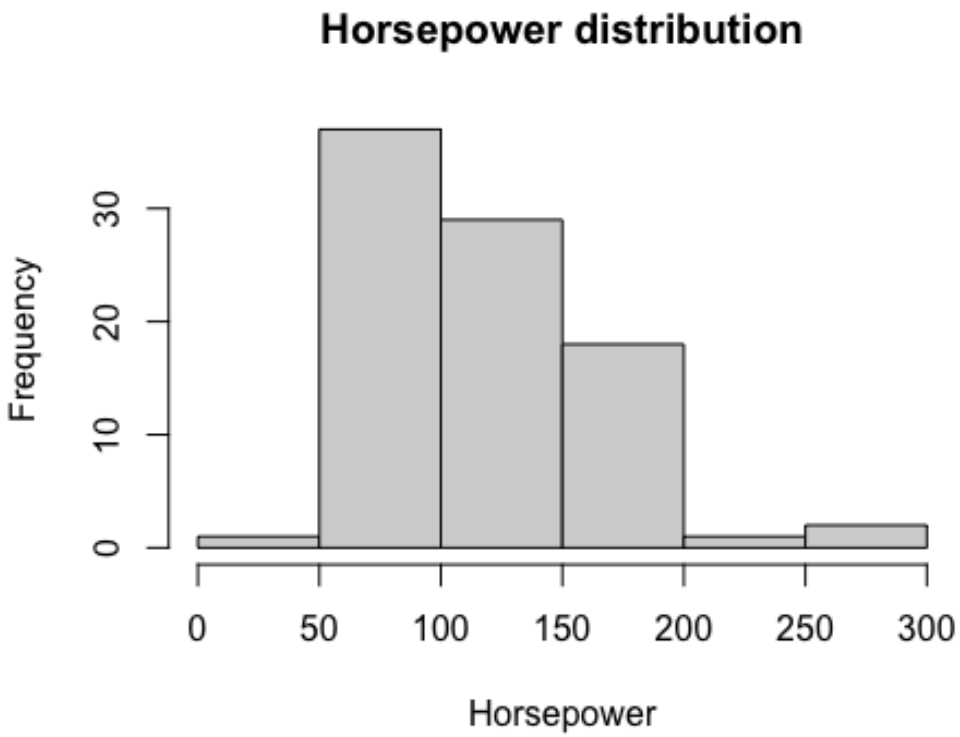
```
## [1] 29
```

- Missing values in the original dataset are recorded as "?". The requirement is to replace these missing values with NA, using [ ] to find row that contain "?" and substitute NA. The function `sum(is.na())` is used to count the cells containing NA, thereby determining how many rows in each dataset were changed from "?" to NA. Therefore, we knew there are 29 row in total were affected after replacement.
- Replacing "?" with NA does not affect the data distribution because all computations use `na.rm = TRUE` to exclude missing values, thereby leaving the calculation results unaffected.

```
engine.type = factor(engine.data$EngineType, levels = c("dohc", "dohcv", "ohc
","ohcf","ohcv","rotor","NA"))

automobile.bodystyle = factor(automobile.data$BodyStyles, levels = c("hardtop
","wagon","sedan", "hatchback", "covertible"))

maintenance.errorcodes = factor(maintenance.data$ErrorCodes, levels = c(-1,1,
0))
```

- To convert the categorical variables **BodyStyles**, **FuelTypes**, and **ErrorCodes** to factors, use the `factor()` function. Since these factors have categories, also use the `levels =` argument to specify the classifications.

```
engine.data$Horsepower[is.na(engine.data$Horsepower)] = median(engine.data$Ho
rsepower,na.rm = TRUE)

hist(as.numeric(engine.data$Horsepower),
     xlab = "Horsepower",
     main = "Horsepower distribution")
```

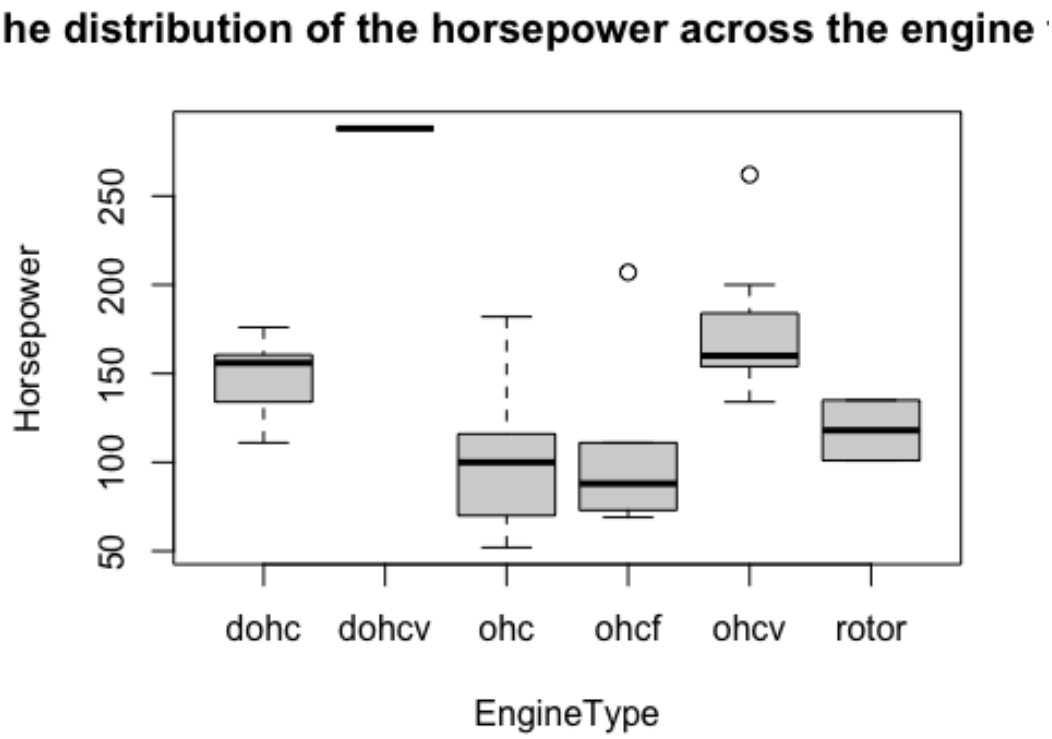


- A histogram is suitable for visualizing the distribution.

Question 2: Write the code to analyse the distribution of the horsepower across the engine types. Write the code to investigate the distribution of the horsepower across the groups of the engine sizes (e.g., 60-90, 91-190, 191-299, 300+). Visualise both the findings using the histogram. Explain your findings.

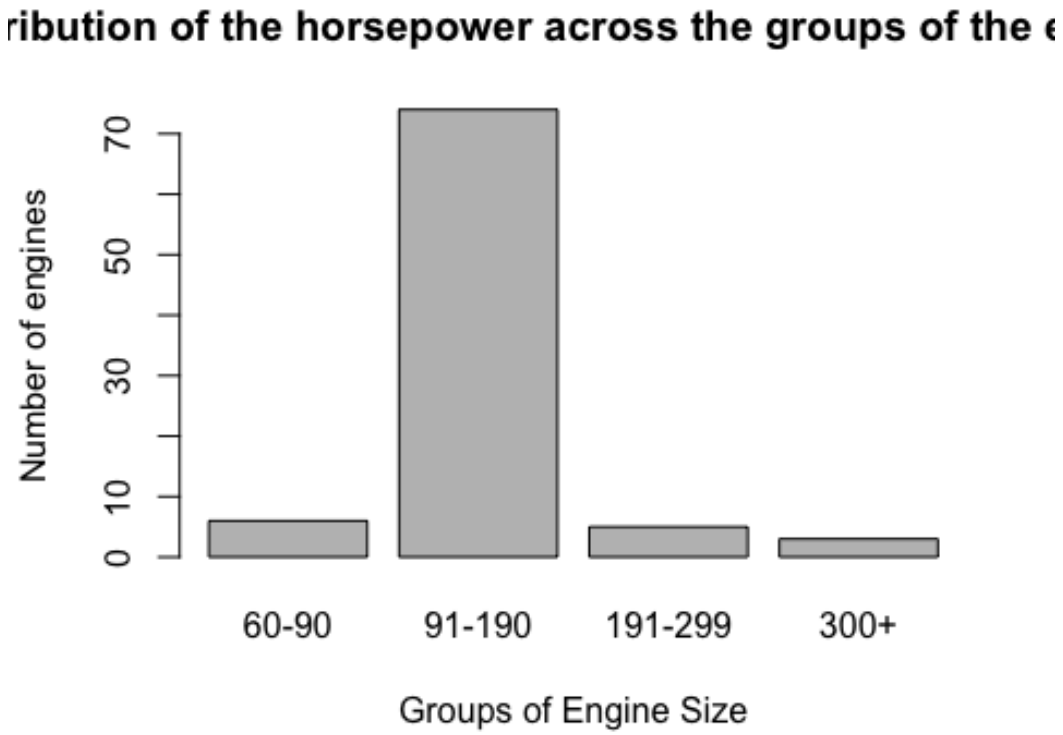
- Using `boxplot()`, we can evaluate the power of each engine type. The interquartile range (IQR) shows the horsepower range where most observations lie, and the plot also reveals the minimum and maximum power for each type.

```
boxplot(as.numeric(Horsepower) ~ EngineType,
        data = engine.data,
        ylab ="Horsepower",
        main = "The distribution of the horsepower across the engine types")
```



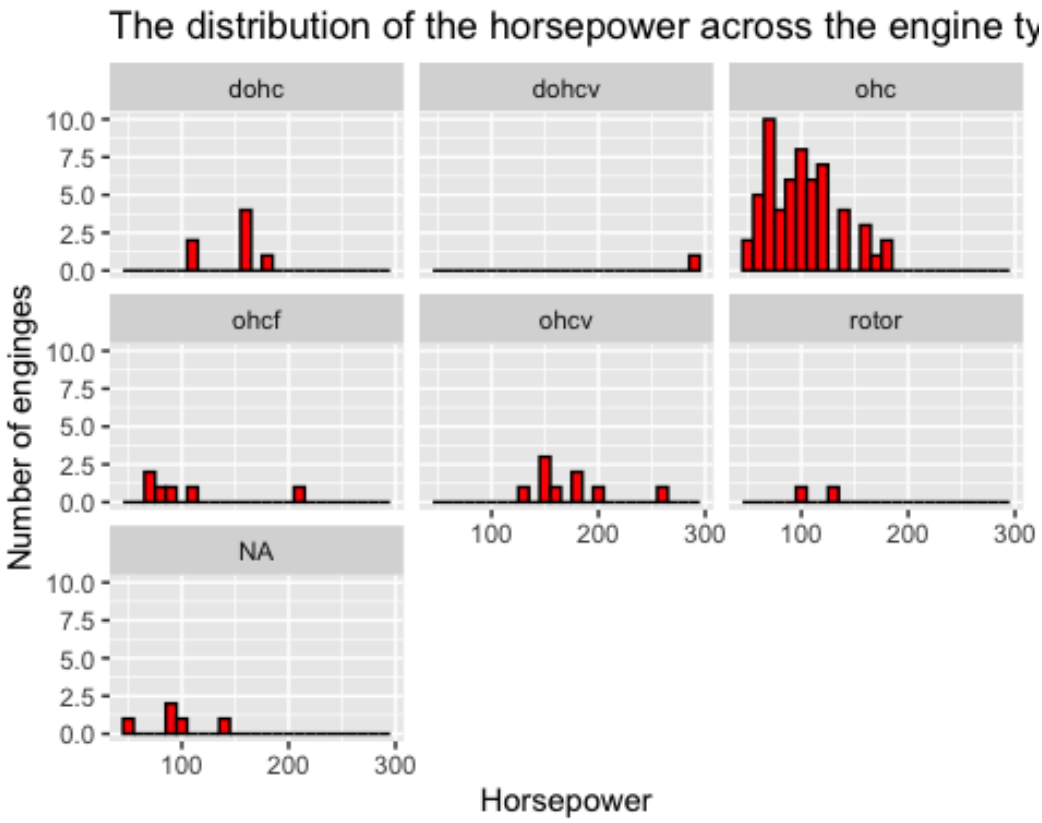
- Using `boxplot()`, we can evaluate the power of each engine type. The interquartile range (IQR) shows the horsepower range where most observations lie, and the plot also reveals the minimum and maximum power for each type.

```r
GroupEngineSize = engine.data %>%
  filter(!is.na(EngineSize)) %>%
  mutate(
    engsize.group = cut(EngineSize,
      breaks = c(60, 90, 190, 299, Inf),
      labels = c("60-90", "91-190", "191-299", "300+"),
      right = TRUE,
      include.lowest = TRUE
    )
  )
barplot(summary(GroupEngineSize$engsize.group),
        xlab = "Groups of Engine Size",
        ylab = "Number of engines",
        main = "The distribution of the horsepower across the groups of the e
ngine sizes ")
```



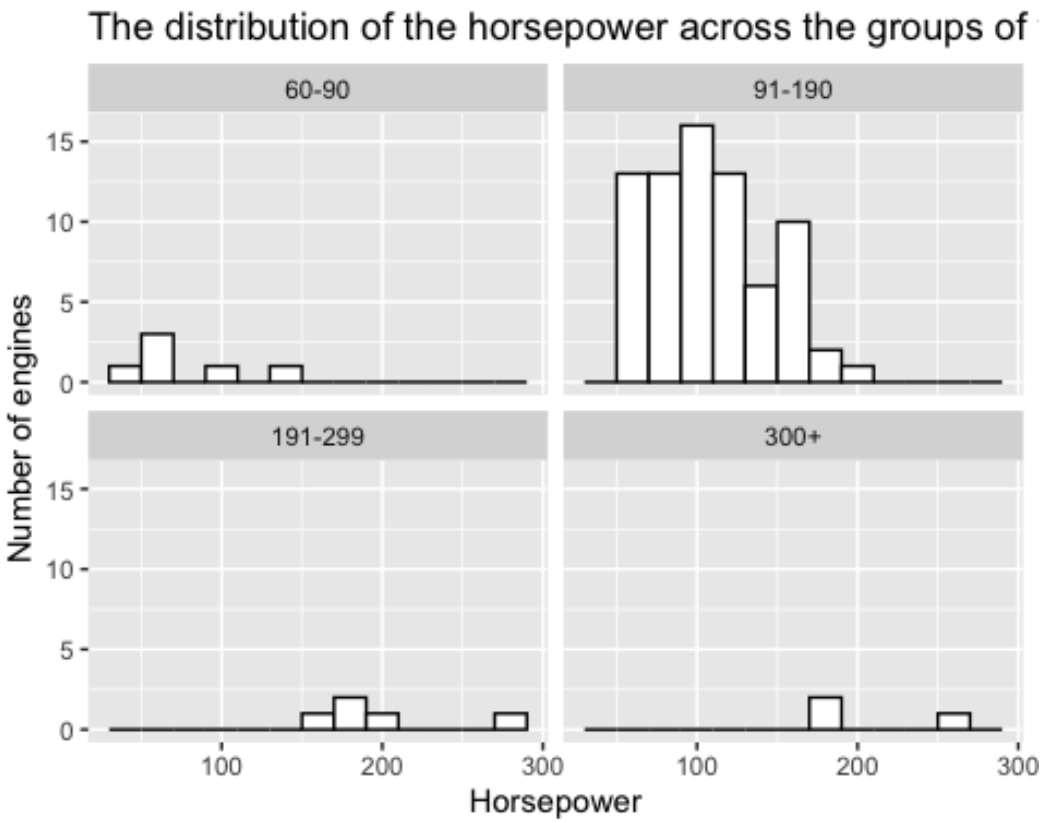
- To group engines size by any horsepower levels, use the pipe operator %>%. First, remove missing values with `filter(!is.na())`. Then groups horsepower with `cut()`, specifying `breaks =` for the cut points, `labels =` to name each group, `right =` to include right point (e.g., with cut points 60 and 90, `right = TRUE` means (60, 90]), and `include.lowest = TRUE` to include the lowest (e.g., 60) in the first interval.

```
ggplot(data = engine.data) +
  geom_histogram(mapping = aes(x = as.numeric(Horsepower)), binwidth = 10, co
l = "black", fill="red") +
  facet_wrap(~ EngineType) +
  labs(title = "The distribution of the horsepower across the engine types",
       x = "Horsepower",
       y = "Number of enginges")
```



- Based on the histogram, different engine types align with different horsepower. **dohc**, **ohcv**, and **rotor** engines typically have horsepower (hp) in the **100–200** range. **ohc** and **ohcf** are used more often in engines with **less than 100 hp**. **dohcv** appears to be suited to high performance, as cars use it only at **around 300 hp**, at lower horsepower, no cars use this engine type. Above **200 hp**, only **ohcf** and **ohcv** appear suitable. Therefore, none single type has an advantage across the entire horsepower range.

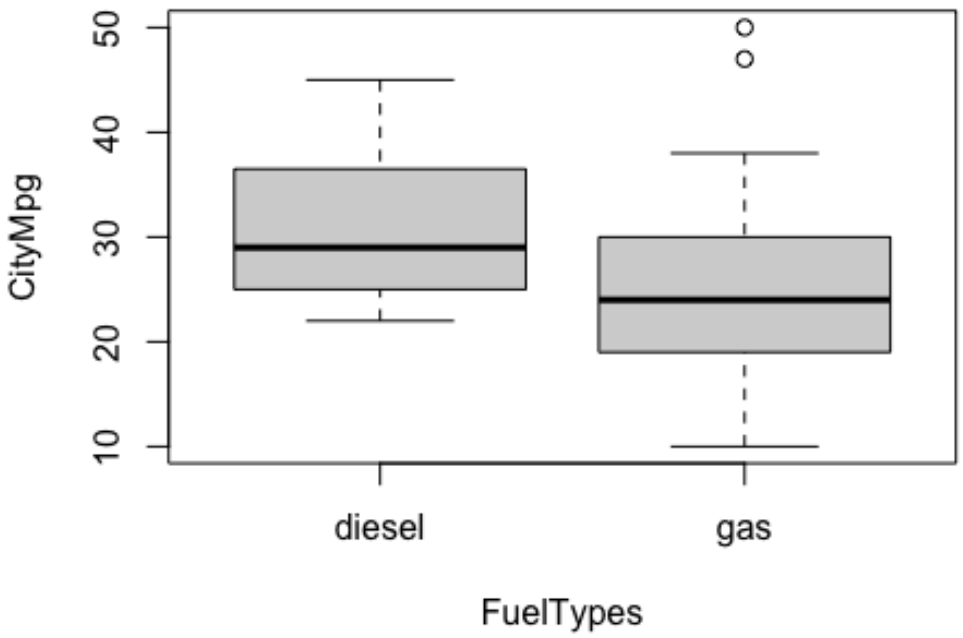
```
ggplot(data = GroupEngineSize) +
  geom_histogram(mapping = aes(x = as.numeric(Horsepower)), binwidth = 20, co
l = "black", fill="white") +
  facet_wrap(~ GroupEngineSize$engsize.group) +
  labs(title = "The distribution of the horsepower across the groups of the e
ngine sizes",
       x = "Horsepower",
       y = "Number of engines")
```



- From the histogram, engine **size** appears related to the horsepower produced. For the **60–90** size group, the **maximum** achievable output is about **150 hp**. The **91–190** group generates output similar to the 60–90 group, but its upper limit extends **to over 200 hp**. The **two largest size groups** both produce higher horsepower than the two smaller groups, with **minimums starting around 150 hp** and **maximums reaching about 300 hp**. From these observations, larger engine sizes yield **higher maximum horsepower**, and the **engine size group** should be considered when crafting the engine with a desired horsepower level.

Question 3: Do diesel cars have higher average CityMpg than gasoline cars? Provide statistical evidence. How does DriveWheels affect fuel efficiency (CityMpg and HighwayMpg)? Elaborate on the findings. Filter out those engines in the dataset that have trouble or are suspected of having trouble; what are the top 5 most common troubles related to the engines? Do the troubles differ between engine types?

```
fueltype = unique(subset(engine.data, select = c("EngineModel", "FuelTypes"))
)
citympg = subset(automobile.data, select = c("EngineModel", "CityMpg"))

fueltype.citympg = citympg %>% inner_join(fueltype, by = c("EngineModel" = "E
ngineModel"))

aggregate(CityMpg ~ FuelTypes,
          mean,
        data = fueltype.citympg)

##   FuelTypes  CityMpg
## 1    diesel 30.30000
## 2       gas 24.67935

boxplot(CityMpg ~ FuelTypes,
        data = fueltype.citympg)
```



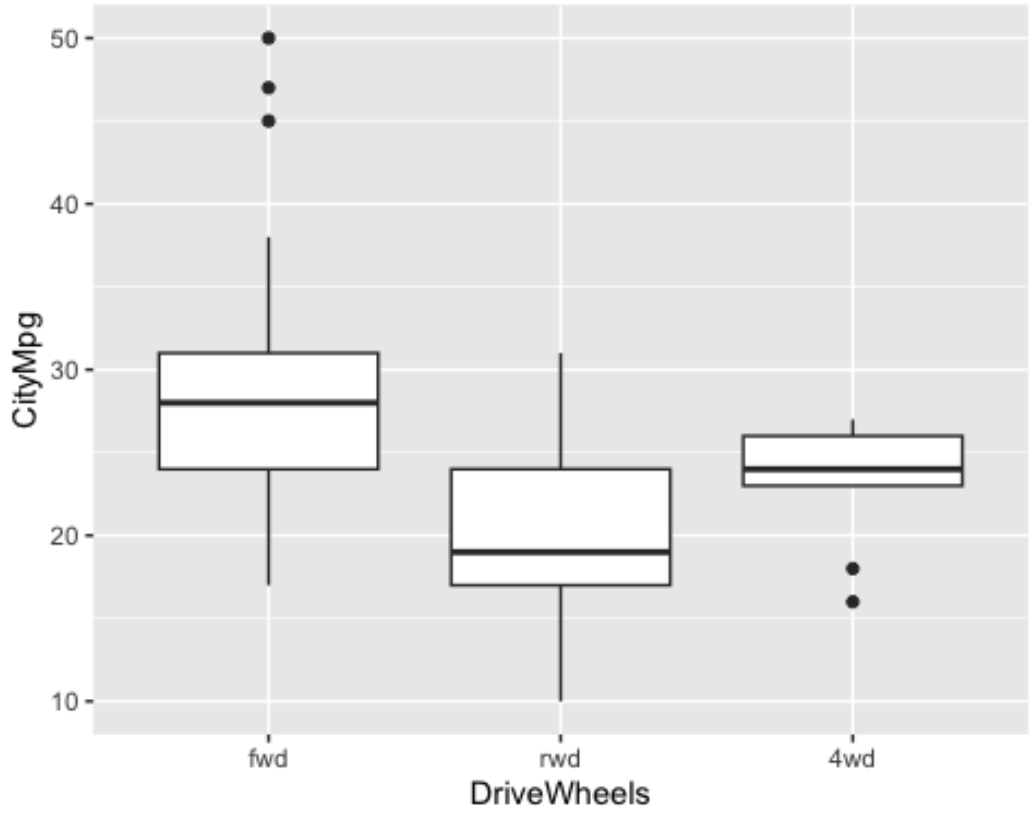
```r
wilcox.test(CityMpg ~ FuelTypes, fueltype.citympg, alternative = "greater")
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  CityMpg by FuelTypes
## W = 2670, p-value = 0.000449
## alternative hypothesis: true location shift is greater than 0
```

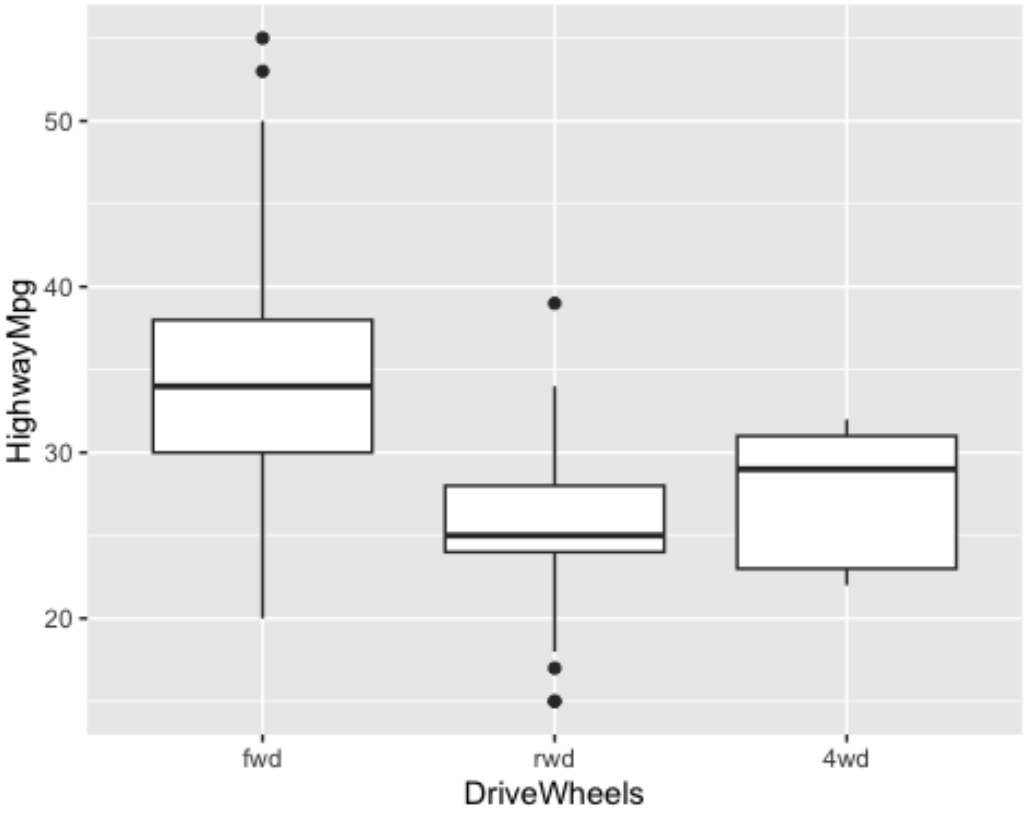
- Using the dataset, when calculating the mean CityMpg for diesel, and gas cars, the results indicate that diesel cars consume more gallons per mile in city driving. Visualizing with a boxplot also shows that the IQR for diesel engines is higher than for gasoline. To ensure statistical evidence, a Wilcoxon test is used to compare whether the mean for diesel is higher than for gas. The Wilcoxon test is chosen here because the data's normality has not been verified. We set up hypotheses as follows:
    - **H0:** Diesel engines consume the same or less fuel than gasoline engines per mile.
    - **H1:** Diesel engines consume more fuel than gasoline engines per mile.
- A one-tailed test is conducted with `alternative = "greater"` and a 5% significance level. With p-value < significance level, we reject H0. Therefore, we can confirm statistically that diesel engines consume more fuel than gas engines for each mile in city driving.

```r
factor.wheel = factor(automobile.data$DriveWheels, levels = c("fwd", "rwd", "
4wd"), ordered = TRUE)

wheel.mpg = subset(automobile.data, select = c("DriveWheels","CityMpg", "High
wayMpg"))

ggplot(data = wheel.mpg) +
  geom_boxplot(mapping = aes(x = factor.wheel, y = wheel.mpg$CityMpg)) +
  labs(x = "DriveWheels", y = "CityMpg")
```

```
## Warning: Use of `wheel.mpg$CityMpg` is discouraged.
## ℹ Use `CityMpg` instead.
```




```
ggplot(data = wheel.mpg) +
  geom_boxplot(mapping = aes(x = factor.wheel, y = wheel.mpg$HighwayMpg))+
  labs(x = "DriveWheels", y = "HighwayMpg")

## Warning: Use of `wheel.mpg$HighwayMpg` is discouraged.
## ℹ Use `HighwayMpg` instead.
```



- From the boxplots, **RWD** is the most fuel-efficient under both **city** and **highway** conditions, follow by **4WD** and then **FWD**. **FWD** consistently shows higher fuel consumption in both driving conditions. For **RWD**, highway fuel use appears better than in the city: the **IQR is smaller**, indicating small spread and thus more eco consumption in fuel. In contrast, **4WD** shows a **larger spread** on the highway, which implying consume more there, while displaying a **smaller spread** (better saving) in city driving.

```
errorcar.data = maintenance.data %>% filter(!is.na(Methods))

engine.error = maintenance.data %>% filter(ErrorCodes == 1)

trouble.frequency = (as.data.frame(table(engine.error$Troubles)))
names(trouble.frequency) = c("Troubles", "Frequency")

plate.enginetype = subset(automobile.data, select=c("PlateNumber","EngineMode
l")) %>%
  inner_join(unique(subset(engine.data, select = c("EngineModel", "EngineType
"))),
             by = c("EngineModel" = "EngineModel"))

plateerror = engine.error %>% inner_join(plate.enginetype, by = c("PlateNumbe
```

```
r" = "PlateNumber"))

with(plateerror, table(Troubles, EngineType))

##                       EngineType
## Troubles               dohc dohcv ohc ohcf ohcv rotor
##   Cam shaft               0     0  11    0    0     0
##   Crank shaft             0     0   5    0    0     1
##   Cylinders               2     0  29    5    1     0
##   ECU's power             0     0   6    0    0     0
##   Fans                    0     1   9    1    1     0
##   Ignition                1     0   8    0    0     0
##   Ignition (finding)      2     0  14    3    1     0
##   Noise (finding)         1     0  15    2    1     0
##   O2 sensors              0     0   1    0    0     0
##   Oil filter              0     0   2    0    0     1
##   Pedals                  0     0   5    0    1     0
##   Pressure sensors        0     0  10    0    0     0
##   Stroke                  0     0   3    0    0     0
##   Suspected battery       0     0   8    0    1     0
##   Temperature sensors     0     0   6    0    1     0
##   Valve clearance         0     2  10    1    1     0
```

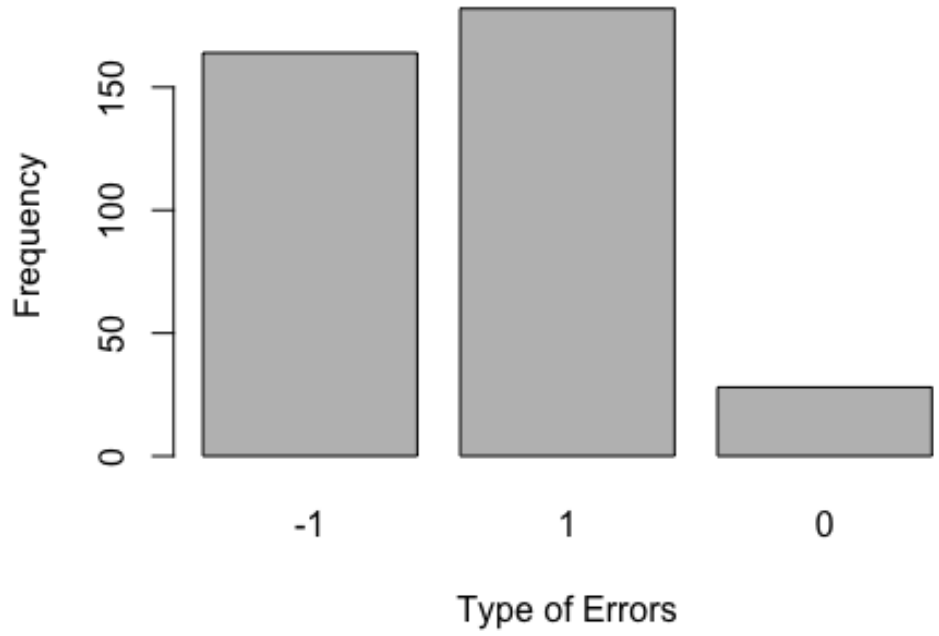
- For cars without issues, the Method field is set to `NA`. To filter vehicles that are troubled or suspected of having trouble, exclude rows with `NA` in Method using `filter(!is.na(Method))`. Similarly, to isolate engine related troubles, keep only records with ErrorCodes = 1 (engine error code equals 1). The top five most common engine troubles are: Cylinders, Ignition, Noise, Valve clearance, and Fans, in descending order of frequency.

- Because engine type is stored in engine.csv, identifying the engine types of cars coming in for maintenance can only be done by cross-referencing via the plate number. However, from the plate number you can retrieve only the Engine Model, and from the Engine Model you can then determine the Engine Type. Therefore, I use `inner_join()` twice with Engine Model and Plate Number as the intermediaries. After that, use `with()` to build a table identifying errors by engine type.

- From this table, cars still exhibit issues unique to specific engine types. However, there are five problems that appear across four or more engine types: Cylinders, Fans, Ignition (finding), Noise (finding), and Valve clearance. This shows that while each engine type has some specific troubles, but there are also five common issues that appear in most engine types in the maintenance data.

Question 4: Write the code to show which error type (ErrorCodes) occurs most frequently; Write the code to analyze the factors that might influence the maintenance methods (Urgent care, Adjustment, Replacement) for the trouble vehicles (confirmed or suspected) in the dataset. Any factors in the dataset, such as BodyStyles, FuelTypes, and ErrorCodes, can be considered. Pick 2 of the factors and explain if there is a trend that explains the variation.

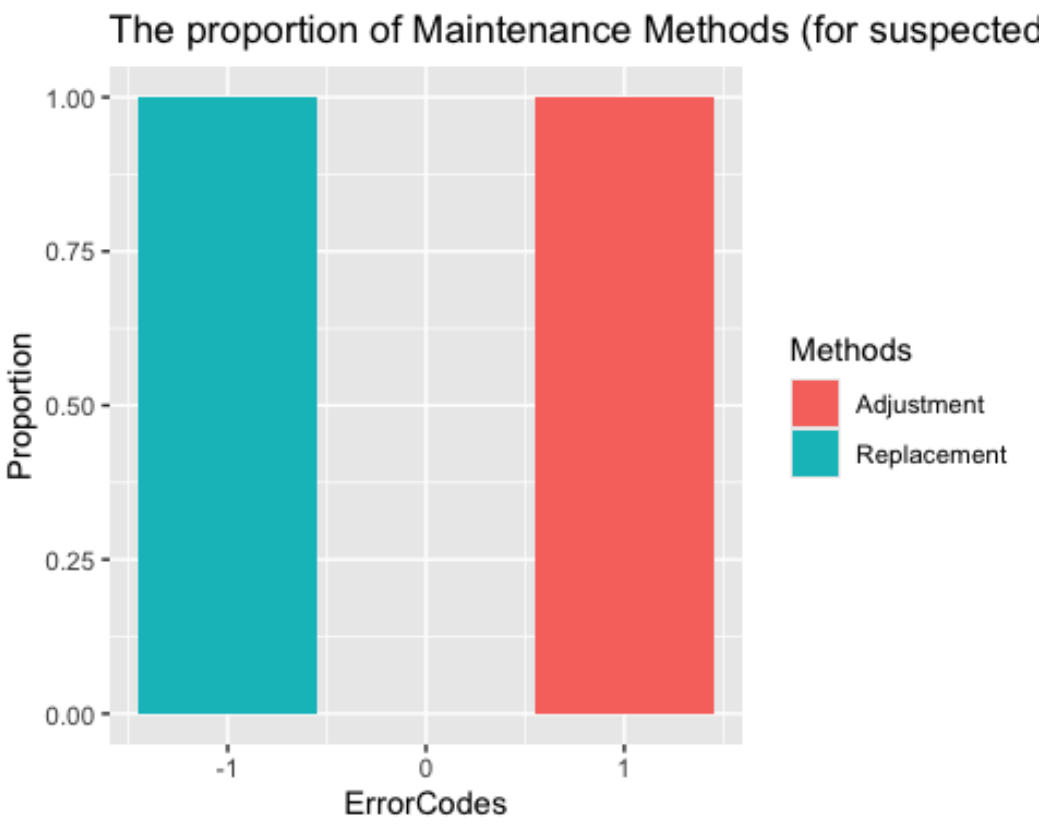
```
barplot(summary(maintenance.errorcodes), ylab = "Frequency", xlab = "Type of
Errors")
```



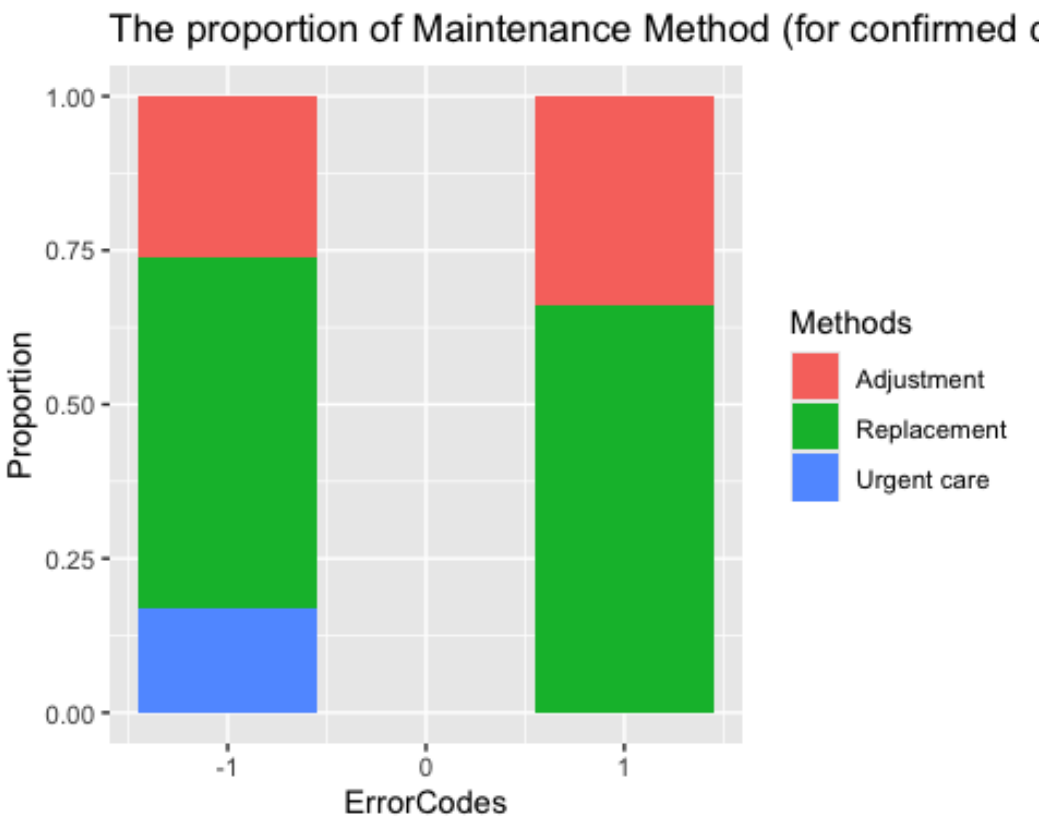
- From the bar plot, engine failures are the most common, followed by other vehicle component failures.

```
suspect = c("Ignition (finding)", "Noise (finding)","Gear box (finding)")
confirm.trouble = subset(errorcar.data, !(Troubles %in% suspect))

suspect.trouble = subset(errorcar.data, Troubles == "Ignition (finding)" |  T
roubles == "Noise (finding)" | Troubles == "Gear box (finding)")

ggplot(suspect.trouble) +
  geom_bar(mapping = aes(x = ErrorCodes, fill = Methods), position = "fill")
+
  labs(title = "The proportion of Maintenance Methods (for suspected car)",
```

```
       x = "ErrorCodes",
       y = "Proportion")
```




```
ggplot(confirm.trouble) +
  geom_bar(mapping = aes(x = ErrorCodes, fill = Methods), position = "fill")
+
  labs(title = "The proportion of Maintenance Method (for confirmed car",
       x = "ErrorCodes",
       y = "Proportion")
```

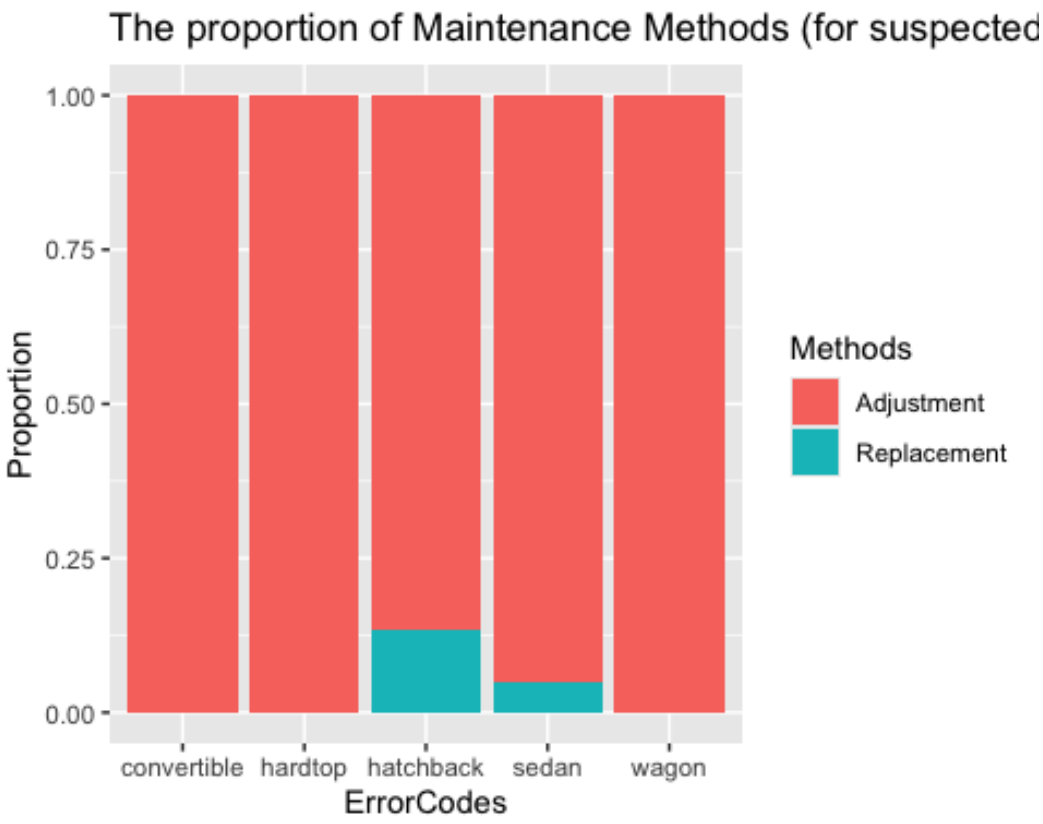


- For the suspected troubles vehicle category, each specific fault code is associated with a unique repair method. According to the maintenance data, faults coded as –1 (other component fails) are treated exclusively with replacement, whereas faults coded as 1 (engine fails) are fixed only with adjustment. Therefore, we can predict that, for suspected vehicles, only one repair method will be applied, determined directly by the error code.

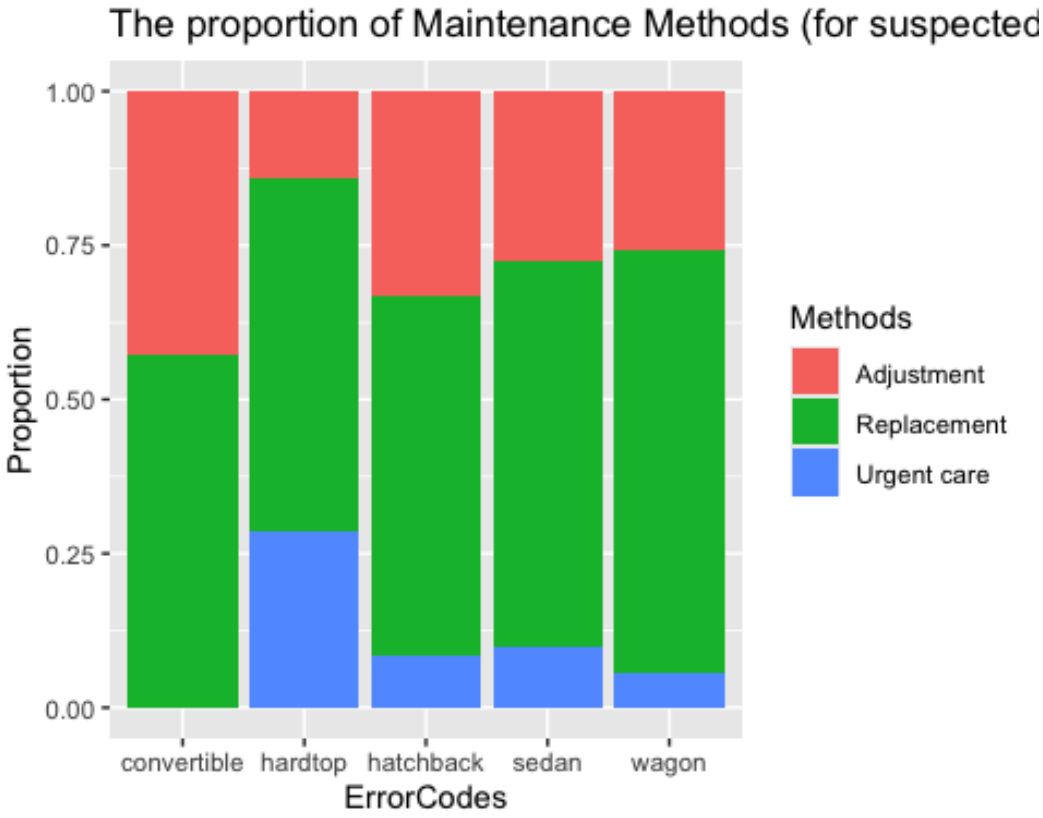
- The confirm troubles vehicle view shows the opposite pattern: engine-related faults are handled more often via replacement. However, engine faults generally do not require urgent care compared with failures in other components. For vehicles with confirmed troubles, even the -1 error code is most frequently fixed through replacement.

```
bodystyle = subset(automobile.data, select = c("PlateNumber", "BodyStyles"))
bodystyle.confirmed = confirm.trouble %>% inner_join(bodystyle, by = c("Plate
Number" = "PlateNumber"))
bodystyle.suspect = suspect.trouble %>% inner_join(bodystyle, by = c("PlateNu
mber" = "PlateNumber"))

ggplot(bodystyle.suspect) +
  geom_bar(mapping = aes(x = BodyStyles, fill = Methods), position = "fill")
+
```

```
  labs(title = "The proportion of Maintenance Methods (for suspected car)",
       x = "ErrorCodes",
       y = "Proportion")
```




```
ggplot(bodystyle.confirmed) +
  geom_bar(mapping = aes(x = BodyStyles, fill = Methods), position = "fill")
+
  labs(title = "The proportion of Maintenance Methods (for suspected car)",
       x = "ErrorCodes",
       y = "Proportion")
```



- For suspected troubles, the method used across body styles is mostly adjustment, with only a few cars requiring replacement (notably hatchback and sedan). For confirmed troubles, replacement is used more frequently. A few vehicles, excluding convertibles require urgent care, but the porportion is small, only the hardtop body style shows about 25% of cars needing urgent care.

Question 5:

Github: https://github.com/Itommtd2907/Analytics-Programming---Assignment

SSH key: git@github.com:Itommtd2907/Analytics-Programming—Assignment.git

HTTPS: https://github.com/Itommtd2907/Analytics-Programming---Assignment.git

Code:

In the Terminal, I code the follow line to push the repo into GitHub:

git init

git add ApAssignment.Rmd

git commit -m “Adding AP Assignment into Github”

git remote add origin https://github.com/Itommtd2907/Analytics-Programming---Assignment.git

git push -u origin master