Assignment 1

Lao Khanh Naiva - 22145720

2025-04-04

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.

TASK 1

Importing the data

```
library(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.3.3

library(tidyverse)

## Warning: package 'tidyverse' was built under R version 4.3.3

## Warning: package 'tibble' was built under R version 4.3.3

## Warning: package 'tidyr' was built under R version 4.3.3

## Warning: package 'readr' was built under R version 4.3.3

## Warning: package 'purrr' was built under R version 4.3.3

## Warning: package 'dplyr' was built under R version 4.3.3

## Warning: package 'forcats' was built under R version 4.3.3

## Warning: package 'lubridate' was built under R version 4.3.3

## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.5
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ lubridate 1.9.3     ✔ tibble    3.2.1
## ✔ purrr     1.0.2     ✔ tidyr     1.3.1
## ── Conflicts ──────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force
all conflicts to become errors

automobile = read.csv("Automobile.csv")
maintenance = read.csv("Maintenance.csv")
engine = read.csv("Engine.csv")
```

Presenting the data

```
# The head() function gives a preview of inputed data, providing the overall
layout of each dataset
head(automobile)
```

```
##   PlateNumber Manufactures  BodyStyles DriveWheels EngineLocation
WheelBase
## 1     53N-001  Alfa-romero convertible         rwd          front
88.6
## 2     53N-002  Alfa-romero   hatchback         rwd          front
94.5
## 3     53N-003         Audi       sedan         fwd          front
99.8
## 4     53N-004         Audi       sedan         4wd          front
99.4
## 5     53N-005         Audi       sedan         fwd          front
99.8
## 6     53N-006         Audi       sedan         fwd          front
105.8
##   Length Width Height CurbWeight EngineModel CityMpg HighwayMpg
## 1  168.8  64.1   48.8       2548      E-0001      21         27
## 2  171.2  65.5   52.4       2823      E-0002      19         26
## 3  176.6  66.2   54.3       2337      E-0003      24         30
## 4  176.6  66.4   54.3       2824      E-0004      18         22
## 5  177.3  66.3   53.1       2507      E-0005      19         25
## 6  192.7  71.4   55.7       2844      E-0005      19         25
```

```
head(maintenance)
```

```
##   ID PlateNumber       Date           Troubles ErrorCodes Price
Methods
## 1  1     53N-001 15/02/2024       Break system         -1   110
Replacement
## 2  2     53N-001 16/03/2024       Transmission         -1   175
Replacement
## 3  3     53N-001 15/04/2024    Suspected clutch        -1   175
Adjustment
## 4  4     53N-001 15/05/2024 Ignition (finding)          1   180
Adjustment
## 5  5     53N-001 14/06/2024            Chassis         -1    85
Replacement
## 6  6     53N-002 15/02/2024          Cylinders          1  1000
Replacement
```

```
head(engine)
```

```
##   EngineModel EngineType NumCylinders EngineSize FuelSystem Horsepower
## 1      E-0001       dohc         four        130       mpfi        111
## 2      E-0002       ohcv          six        152       mpfi        154
## 3      E-0003        ohc         four        109       mpfi        102
## 4      E-0004        ohc         five        136       mpfi        115
## 5      E-0005        ohc         five        136       mpfi        110
## 6      E-0006        ohc         five        131       mpfi        140
##   FuelTypes Aspiration
## 1       gas        std
## 2       gas        std
```

```
## 3       gas        std
## 4       gas        std
## 5       gas        std
## 6       gas      turbo
```

```r
# The str() function supports the above output in a more efficient way, with
a more comprehensive summary of each dataset's structure
str(automobile)
```

```
## 'data.frame':    204 obs. of  13 variables:
##  $ PlateNumber   : chr  "53N-001" "53N-002" "53N-003" "53N-004" ...
##  $ Manufactures  : chr  "Alfa-romero" "Alfa-romero" "Audi" "Audi" ...
##  $ BodyStyles    : chr  "convertible" "hatchback" "sedan" "sedan" ...
##  $ DriveWheels   : chr  "rwd" "rwd" "fwd" "4wd" ...
##  $ EngineLocation: chr  "front" "front" "front" "front" ...
##  $ WheelBase     : num  88.6 94.5 99.8 99.4 99.8 ...
##  $ Length        : num  169 171 177 177 177 ...
##  $ Width         : num  64.1 65.5 66.2 66.4 66.3 71.4 71.4 71.4 67.9 64.8
...
##  $ Height        : num  48.8 52.4 54.3 54.3 53.1 55.7 55.7 55.9 52 54.3
...
##  $ CurbWeight    : int  2548 2823 2337 2824 2507 2844 2954 3086 3053 2395
...
##  $ EngineModel   : chr  "E-0001" "E-0002" "E-0003" "E-0004" ...
##  $ CityMpg       : int  21 19 24 18 19 19 19 17 16 23 ...
##  $ HighwayMpg    : int  27 26 30 22 25 25 25 20 22 29 ...
```

```r
str(maintenance)
```

```
## 'data.frame':    374 obs. of  7 variables:
##  $ ID         : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ PlateNumber: chr  "53N-001" "53N-001" "53N-001" "53N-001" ...
##  $ Date       : chr  "15/02/2024" "16/03/2024" "15/04/2024" "15/05/2024"
...
##  $ Troubles   : chr  "Break system" "Transmission" "Suspected clutch"
"Ignition (finding)" ...
##  $ ErrorCodes : int  -1 -1 -1 1 -1 1 1 0 -1 -1 ...
##  $ Price      : int  110 175 175 180 85 1000 180 0 180 180 ...
##  $ Methods    : chr  "Replacement" "Replacement" "Adjustment" "Adjustment"
...
```

```r
str(engine)
```

```
## 'data.frame':    88 obs. of  8 variables:
##  $ EngineModel : chr  "E-0001" "E-0002" "E-0003" "E-0004" ...
##  $ EngineType  : chr  "dohc" "ohcv" "ohc" "ohc" ...
##  $ NumCylinders: chr  "four" "six" "four" "five" ...
##  $ EngineSize  : int  130 152 109 136 136 131 131 108 164 164 ...
##  $ FuelSystem  : chr  "mpfi" "mpfi" "mpfi" "mpfi" ...
##  $ Horsepower  : chr  "111" "154" "102" "115" ...
```

```
## $ FuelTypes   : chr  "gas" "gas" "gas" "gas" ...
## $ Aspiration  : chr  "std" "std" "std" "std" ...
```

- Automobile Data: includes 204 automobiles with 13 variables describing various attributes of each vehicle.
- Maintenance Data: includes 374 observations with 7 variables describing various aspects of maintenance activities.
- Engine Data: includes 88 engine models with 8 variables describing various attributes of different engines.

Merging the datasets

For the next tasks, all three datasets are merged into a single dataframe called “df” for ease of analysis.

```
# Combining all three datasets into a single dataframe
df = automobile %>%
  left_join(engine, by = "EngineModel", relationship = "many-to-many") %>%
  left_join(maintenance, by = "PlateNumber", relationship = "many-to-many")
df = as.data.frame(df)
str(df)
```

```
## 'data.frame':    391 obs. of  26 variables:
## $ PlateNumber   : chr  "53N-001" "53N-001" "53N-001" "53N-001" ...
## $ Manufactures  : chr  "Alfa-romero" "Alfa-romero" "Alfa-romero" "Alfa-
romero" ...
## $ BodyStyles    : chr  "convertible" "convertible" "convertible"
"convertible" ...
## $ DriveWheels   : chr  "rwd" "rwd" "rwd" "rwd" ...
## $ EngineLocation: chr  "front" "front" "front" "front" ...
## $ WheelBase     : num  88.6 88.6 88.6 88.6 88.6 94.5 94.5 94.5 99.8 99.4
...
## $ Length        : num  169 169 169 169 169 ...
## $ Width         : num  64.1 64.1 64.1 64.1 64.1 65.5 65.5 65.5 66.2 66.4
...
## $ Height        : num  48.8 48.8 48.8 48.8 48.8 52.4 52.4 52.4 54.3 54.3
...
## $ CurbWeight    : int  2548 2548 2548 2548 2548 2823 2823 2823 2337 2824
...
## $ EngineModel   : chr  "E-0001" "E-0001" "E-0001" "E-0001" ...
## $ CityMpg       : int  21 21 21 21 21 19 19 19 24 18 ...
## $ HighwayMpg    : int  27 27 27 27 27 26 26 26 30 22 ...
## $ EngineType    : chr  "dohc" "dohc" "dohc" "dohc" ...
## $ NumCylinders  : chr  "four" "four" "four" "four" ...
## $ EngineSize    : int  130 130 130 130 130 152 152 152 109 136 ...
## $ FuelSystem    : chr  "mpfi" "mpfi" "mpfi" "mpfi" ...
## $ Horsepower    : chr  "111" "111" "111" "111" ...
## $ FuelTypes     : chr  "gas" "gas" "gas" "gas" ...
```

```
##  $ Aspiration    : chr  "std" "std" "std" "std" ...
##  $ ID            : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Date          : chr  "15/02/2024" "16/03/2024" "15/04/2024"
"15/05/2024" ...
##  $ Troubles      : chr  "Break system" "Transmission" "Suspected clutch"
"Ignition (finding)" ...
##  $ ErrorCodes    : int  -1 -1 -1 1 -1 1 1 0 -1 -1 ...
##  $ Price         : int  110 175 175 180 85 1000 180 0 180 180 ...
##  $ Methods       : chr  "Replacement" "Replacement" "Adjustment"
"Adjustment" ...
```

The left_join function is used to preserve the totality of automobiles used for the data. Many-to-many relationships are also accepted: some engine models might be used in multiple cars, and one car can have multiple maintenance records.

Replacing missing values “?” with NA

```
# The lapply() function will replace all missing values accross all column of
each datasets with ifelse().
df[] <- lapply(df, function(x) {
  ifelse(x == "?", NA, x)
})
```

Convert categorical variables BodyStyles, FuelTypes, ErrorCodes to factors

```
df$BodyStyles = as.factor(df$BodyStyles)
df$ErrorCodes = as.factor(df$ErrorCodes)
df$FuelTypes = as.factor(df$FuelTypes)
```

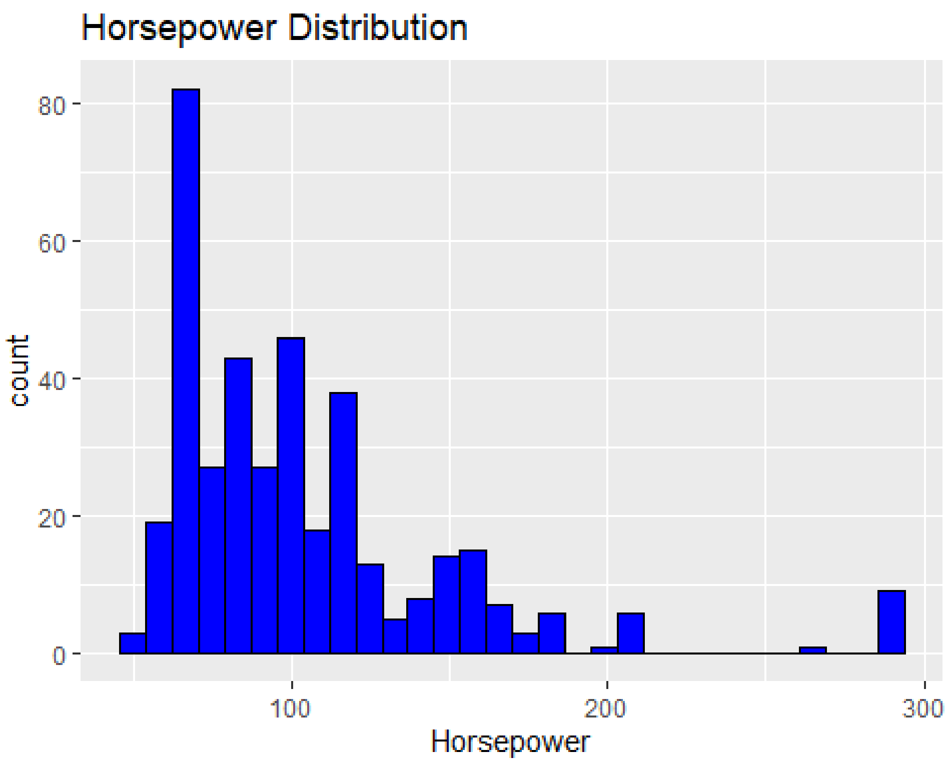
Replace the missing values in column Horsepower with the mean horsepower

```
# The Horsepower variable needs to be converted from a string variable to a
numeric varibale
df$Horsepower = as.numeric(df$Horsepower)

# The mean horsepower is then calculated, without taking into account the
missing values in order to produce an accurate result
mean_horsepower <- mean(df$Horsepower, na.rm = TRUE)

# Replacing the missing values with mean:
missing_horsepower <- is.na(df$Horsepower)
df$Horsepower[missing_horsepower] <- mean_horsepower
```

Horsepower distribution

```
ggplot(df, aes(x = Horsepower)) +
  geom_histogram(fill = "blue", color = "black") +
  ggtitle("Horsepower Distribution")
```

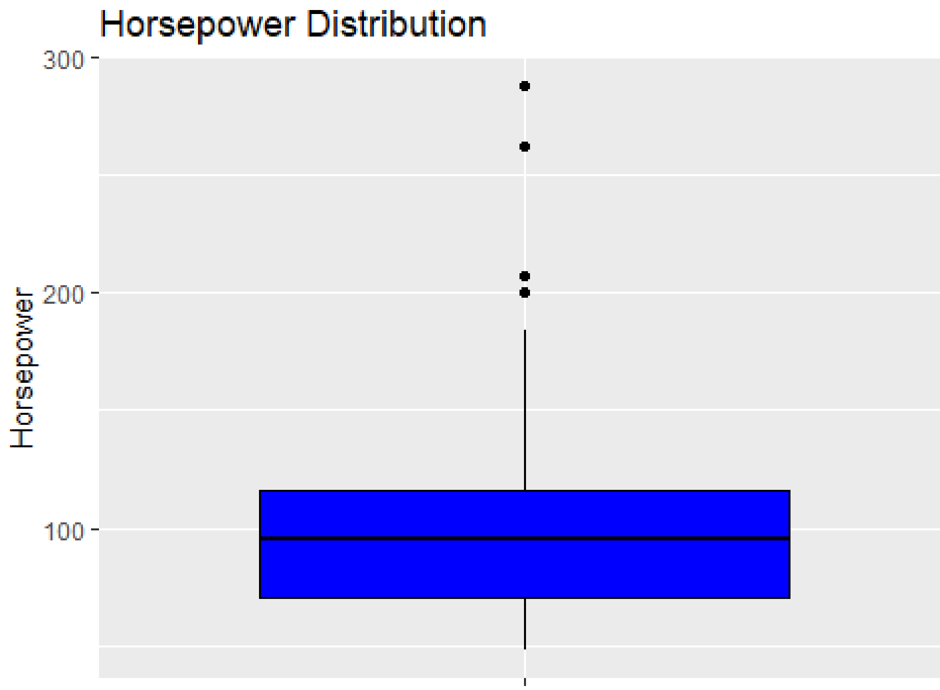
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



A histogram is be used to visualise horsepower distribution, as it provides a representation of frequency and visualisation of most common (or less common) values in the dataset. As can be seen, the data is right-skewed and most engines in the dataset have lower horsepower. Few engines extend towards higher horsepower values, creating a long tail and illustrating several outliers.

A boxplot can provide further insights regarding distribution, providing a visualisation of summary statistics.

```r
ggplot(df, aes(x = "", y = Horsepower)) +
  geom_boxplot(fill = "blue", color = "black") +
  ggtitle("Horsepower Distribution") +
  xlab("") +
  ylab("Horsepower")
```

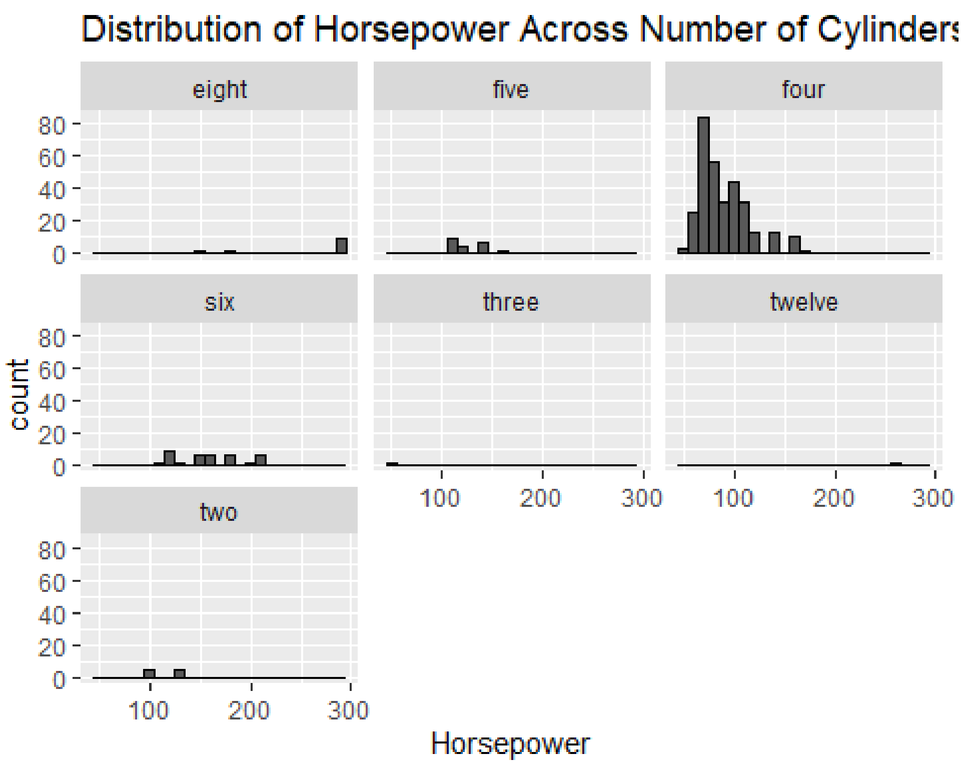


- IQR: Q3 stands at approximately 120 horsepower, while Q1 is approximately 50 horsepower - meaning that the values are spread within a range of 70. This is means that the data is moderately varied.
- Median: the median stands at approximately 90, indicating central tendency and that the data is skewed.
- Outliers: three outliers can be identified, indicating extreme values in horsepower and powerful engines.

TASK 2

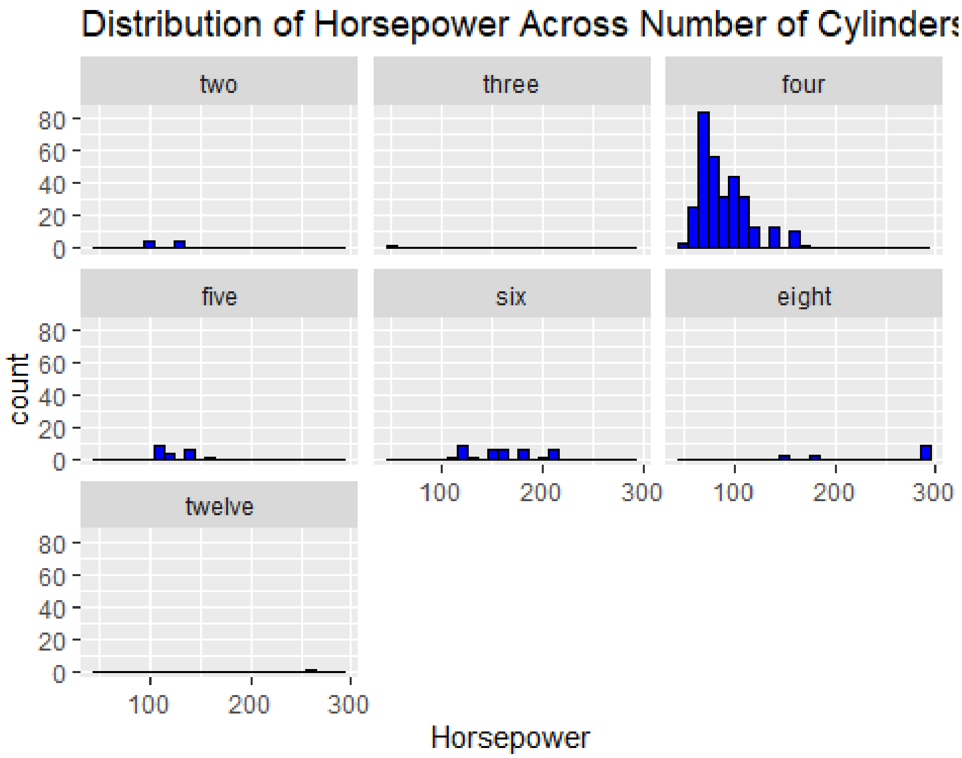
Distribution of horsepower across the number of cylinders

```
# Generating a histogram to visualise horsepower distribution according to
number of cylinders
ggplot(df, mapping = aes(x = Horsepower)) +
  geom_histogram(binwidth = 10, color = "black") +
  facet_wrap(~ NumCylinders, ncol = 3) +
  labs(title = "Distribution of Horsepower Across Number of Cylinders")
```

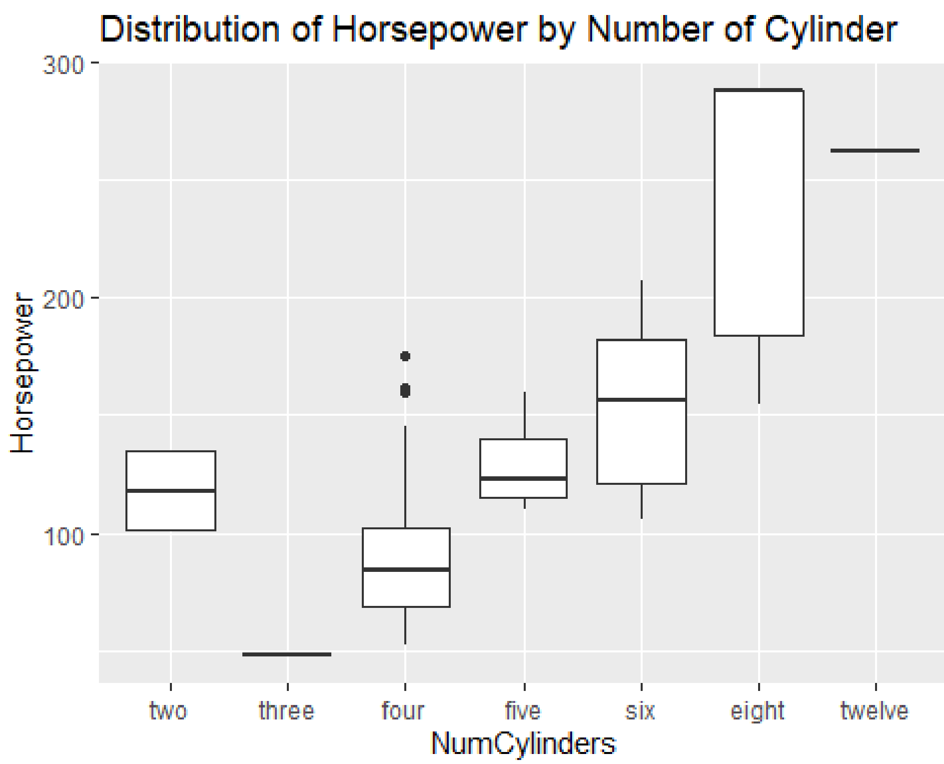


The resulting output is disordered, showing the histograms according to number of cylinders at random. Hence, this requires the definition of levels within the variable NumCylinders.

```
# The variable NumCylinders will be converted into a factor and ordered from small to large
df$NumCylinders <- factor(
  df$NumCylinders,
  levels = c("two", "three", "four", "five", "six", "eight", "twelve"),
  ordered = TRUE
)

# Regenerating the histogram:
ggplot(df, mapping = aes(x = Horsepower)) +
  geom_histogram(binwidth = 10, fill = "blue", color = "black") +
  facet_wrap(~ NumCylinders, ncol = 3) +
  labs(title = "Distribution of Horsepower Across Number of Cylinders")
```




```r
# Generating a boxplot
ggplot(df, aes(x = NumCylinders, y = Horsepower)) +
  geom_boxplot() +
  ggtitle("Distribution of Horsepower by Number of Cylinder")
```



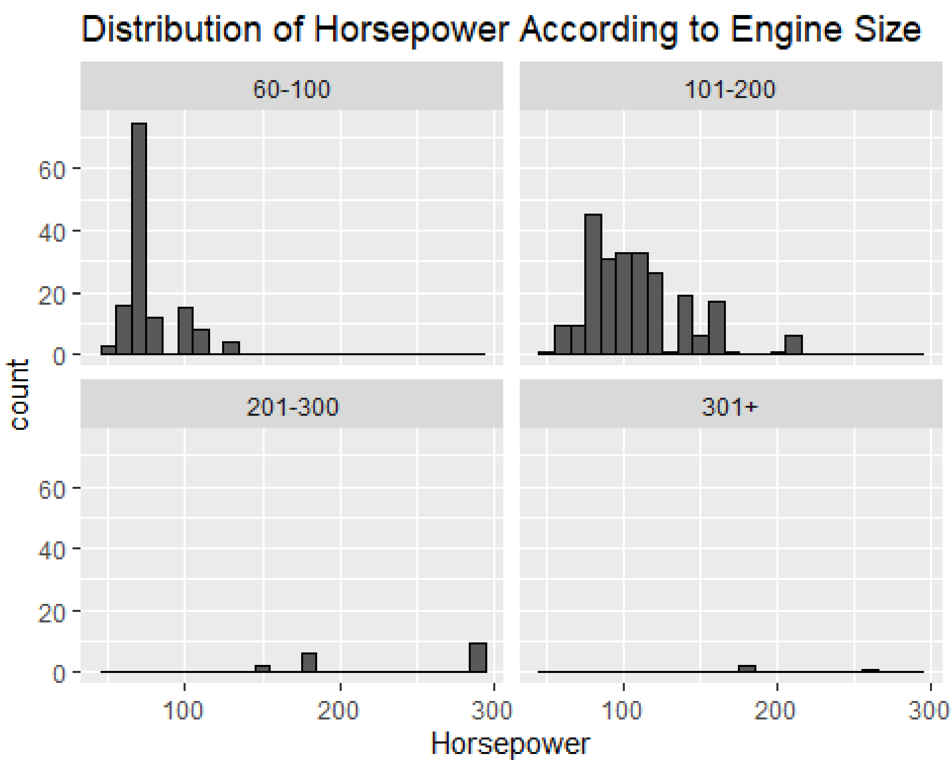
As can be seen, distribution of horsepower varies significantly depending on the number of cylinders. In the histograms, four cylinder counts are most common in the dataset, while three and twelve cylinder counts are least common. In the boxplots, variability of horsepower is most significant in the groups of six and eight cylinders, while it is the least significant in four cylinder counts due to its frequency of observations. This could suggest that the data collected may be biased and not well suited for complex statistical analysis.

Distribution of horsepower across different groups of engine sizes

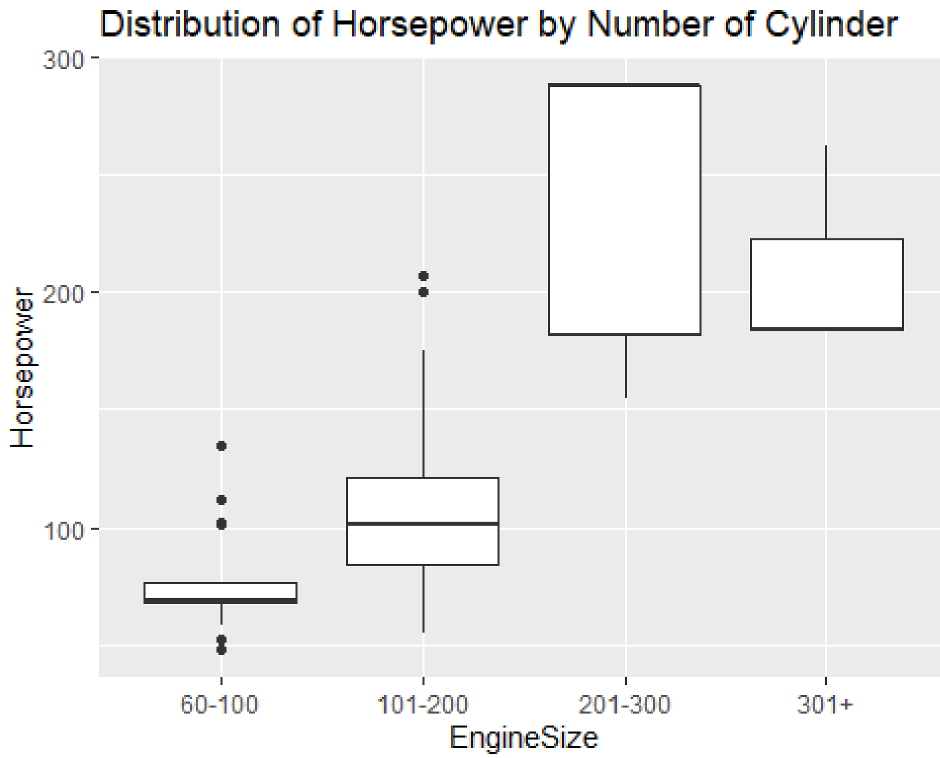
The engine sizes will be divided by 4 different groups: 60-100, 101-200, 201-300, 301+.

```
# Splitting the EngineSize variable by aforementioned groups

df$EngineSize <- ifelse(df$EngineSize >= 60 & df$EngineSize <= 100, "60-100",
                                   ifelse(df$EngineSize >= 101 &
df$EngineSize <= 200, "101-200",
                                          ifelse(df$EngineSize >= 201 &
df$EngineSize <= 300, "201-300",
                                                 ifelse(df$EngineSize > 300,
"301+", NA))))


# Classifying the groups from small to large for ease of comprehension in the
following visualisation
df$EngineSize <- factor(
  df$EngineSize,
```

```r
  levels = c("60-100", "101-200", "201-300", "301+"),
  ordered = TRUE
)

# Generating a histogram
ggplot(df, aes(x = Horsepower)) +
  geom_histogram(binwidth = 10, color = "black") +
  facet_wrap(~EngineSize) +
  ggtitle("Distribution of Horsepower According to Engine Size")
```




```r
# Generating a boxplot
ggplot(df, aes(x = EngineSize, y = Horsepower)) +
  geom_boxplot() +
  ggtitle("Distribution of Horsepower by Number of Cylinder")
```



There is a noticeable trend of increasing horsepower as the engine size gets larger, as can be seen in the histograms. However, the frequency of observations in each category of engine size is highly uneven, with 201-300 and 301+ containing minimal observations. In the boxplot, large engine sizes show the most variability and extreme skewness due to their low observations counts, while smaller and mid-size engine sizes tend to have many outliers.

Overall, horsepower shows a potential positive trend as the number of cylinders and engine size increases which could be further analysed with more complex techniques. However, the structure of the data may not be optimal for further analysis and may require more data collection and / or cleaning.

TASK 3

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

In order to study troubles related to engines, we want to keep all observations filtered by the variable ErrorCodes related to engine failure.

```
troubled_engines = df %>% filter(ErrorCodes == 1)
```

Top 5 most common troubles related to the engines

```
top_troubles_engines = troubled_engines %>%
  group_by(Troubles) %>%
  summarise(Frequency = n()) %>%
  arrange(desc(Frequency))
head(top_troubles_engines)
```

```
## # A tibble: 6 × 2
##   Troubles           Frequency
##   <chr>                  <int>
## 1 Cylinders                 39
## 2 Ignition (finding)        23
## 3 Noise (finding)           20
## 4 Valve clearance           15
## 5 Fans                      13
## 6 Cam shaft                 12
```

The most common causes for engine failure are related to cylinders, ignition, noise, valve and fans.

Top 5 troubles according to Fuel Types

```
unique(troubled_engines$FuelTypes) # Show the types of fuel in the dataset:
gas and diesel
```

```
## [1] gas    diesel
## Levels: diesel gas
```

```
# Diesel fuel
diesel_troubles = troubled_engines %>% filter(FuelTypes == "diesel") %>%
count(Troubles, sort = TRUE)
head(diesel_troubles)
```

```
##      Troubles n
## 1   Cam shaft 3
## 2   Cylinders 3
## 3 Crank shaft 2
## 4      Stroke 2
## 5 ECU's power 1
## 6    Ignition 1
```

```
# Gas
gas_troubles = troubled_engines %>% filter(FuelTypes == "gas") %>%
count(Troubles, sort = TRUE)
head(gas_troubles)
```

```
##             Troubles  n
## 1          Cylinders 36
## 2 Ignition (finding) 22
## 3    Noise (finding) 19
## 4    Valve clearance 15
```

```
## 5               Fans 13
## 6   Pressure sensors 10
```

The troubles differ significantly between fuel types. Overall, it seems that troubled engines mostly occur within gas vehicles, but this could also be due to lack of data for diesel vehicles.

In both types of fuel, common issues related to engine failure involve cylinder troubles which rank Top 1 and Top 2 in diesel and gas vehicles, respectively. Engine failure in diesel vehicles mostly occur as a result of troubles in cam shaft and crank shaft, whereas engine failure in gas vehicles are often due to ignition and noise.

TASK 4

Cross-tabulations

As the dataset is not optimised for complex statistical analysis, simple techniques like cross tabulation and histogram visualisations will give a general overview of factors that might influence the maintenance methods.

Cross tabulations will display the frequency distribution of combinations and reveal general patterns between variables in a concise and understandable summary of their relationships.

- Fuel Types & Methods

```
fuel_crosstab = table(troubled_engines$FuelTypes, troubled_engines$Methods)
print(fuel_crosstab)

## 
##          Adjustment Replacement
##   diesel          5          12
##   gas            87          87
```

Overall, diesel cars have fewer maintenance occurrences compared to gas cars, indicating that gas cars are more prone to maintenance issues. However, diesel cars are more likely to require replacements, while gas cars have an equal likelihood of needing adjustments or replacements.

- Body Styles & Methods

```
body_crosstab = table(troubled_engines$BodyStyles, troubled_engines$Methods)
print(body_crosstab)

## 
##               Adjustment Replacement
##   convertible          3           2
##   hardtop              1           1
##   hatchback           33          32
##   sedan               46          53
##   wagon                9          11
```

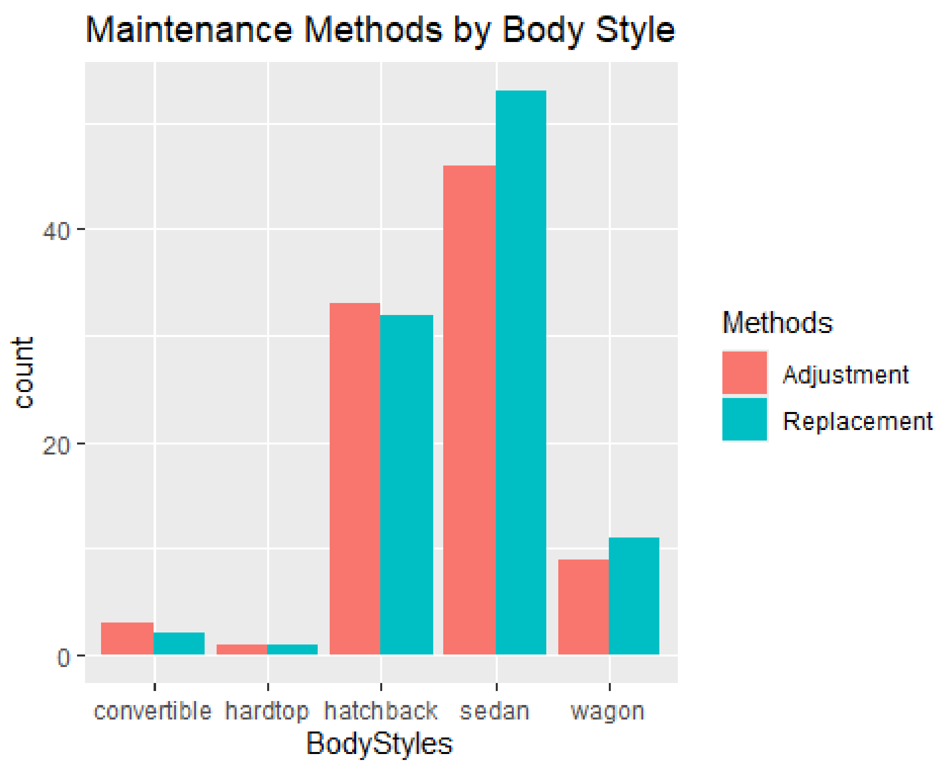
Among different styles of vehicles, Sedans and wagons tend to require more replacements, while hatchbacks have a balanced need for both adjustments and replacements. Convertibles and hardtops show no strong preference for either maintenance method.

Histograms

Histograms can better visualise the relationships illustrated in the cross-tabulations above.

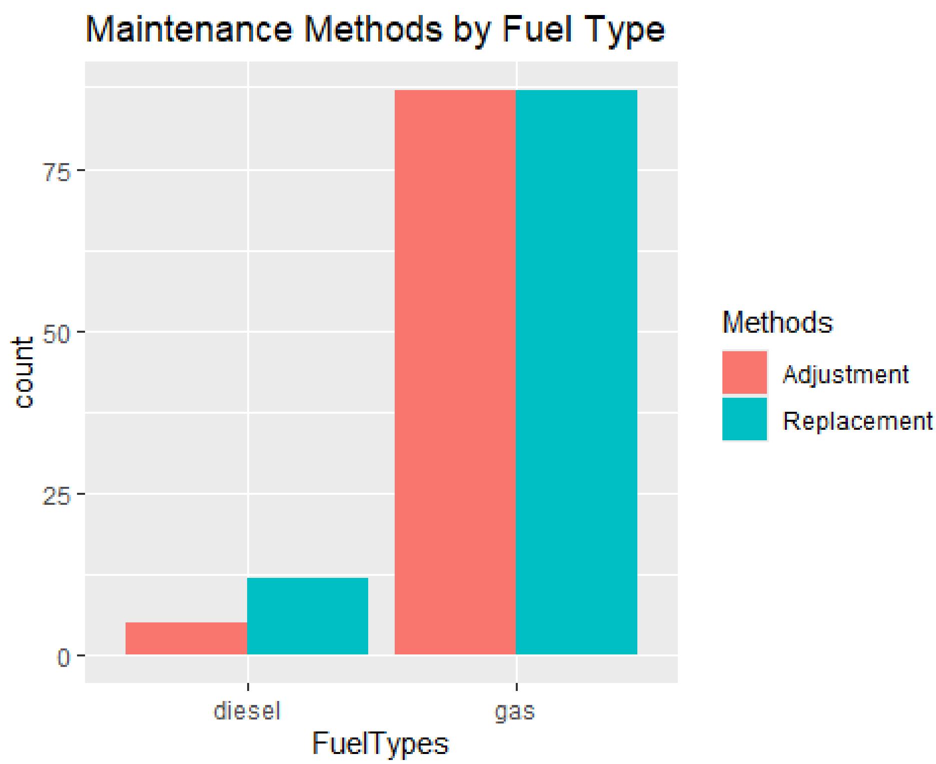
- Body Styles & Methods

```
ggplot(troubled_engines, aes(x = BodyStyles, fill = Methods)) +
  geom_bar(position = "dodge") +
  ggtitle("Maintenance Methods by Body Style")
```



- Fuel Types & Methods

```
ggplot(troubled_engines, aes(x = FuelTypes, fill = Methods)) +
  geom_bar(position = "dodge") +
  ggtitle("Maintenance Methods by Fuel Type")
```


Maintenance Methods by Fuel Type
count
75
50
25
0
diesel
gas
FuelTypes
Methods
Adjustment
Replacement