Assignment Project T3 2025

Lao Khanh Naiva - 22145720

T3 2025

Declaration

Libraries

Part 1

Inspecting and Presenting Data

Importing Data

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

Inspecting Data

```
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 ...

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" "Ign
ition (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"
...
```

```
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" ...
```

Merging data

Using left_join() function to merge the three datasets together in two step where variables overlap. The automobile data can be joined with engine data via EngineModel; the engine data can be joined with maintenance data via PlateNumber. Many-to-many relationships are enabled as engine models may be used in multiple cars, and one car can have multiple maintenance records.

```
df_merged <- automobile %>%
  left_join(engine, by = "EngineModel", relationship = "many-to-many") %>%
  left_join(maintenance, by = "PlateNumber", relationship = "many-to-many")
str(df_merged)

## '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-ro
mero" ...
##  $ BodyStyles    : chr  "convertible" "convertible" "convertible" "convert
ible" ...
##  $ 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" "Adjustme
nt" ...
```

Cleaning Missing Values

```
sum(is.na(df_merged)) # counts existing NAs

## [1] 29

sum(df_merged == "?", na.rm = TRUE) # checks how many ? exist

## [1] 23
```

There is a total of 23 missing values marked with “?” that need to be transformed into true missing values, presently amounting to 29 NAs. The count of actual missing values should correspond to the sum of existing missing values and the amount of question marks, estimated at 52.

```
NA_estimate <- sum(is.na(df_merged)) + sum(df_merged == "?", na.rm = TRUE)
NA_estimate

## [1] 52
```

We transform the question marks, then count all true missing values.

```
df_new <- df_merged
df_new[df_new == "?"] <- NA
sum(is.na(df_new)) # checks sum of all missing values NA

## [1] 52
```

These is now a total of 52 missing values, which matches the estimate and signifies that the transformation was successful. The question marks are converted into true missing data, which were previously recognised as characters. This changes the data distribution with more accurate missing values and affect the output of summary statistics. With the possibility to remove the missing data.

Converting variables into factors

```
variables <- c("BodyStyles", "FuelTypes", "ErrorCodes") # create index to sel
ect variables
sapply(df_new[variables], class) # check current class of variables
```

```
##  BodyStyles   FuelTypes  ErrorCodes 
## "character" "character"   "integer"
```

Currently, the variables BodyStyles, FuelTypes and ErrorCodes are categorical data, namely character and integer, which can be converted into factors with levels.

```
df_new[variables] <- lapply(df_new[variables], as.factor)
sapply(df_new[variables], class) # check if conversion worked

## BodyStyles  FuelTypes ErrorCodes 
##   "factor"   "factor"   "factor"
```

Replacing missing values in Horsepower variable

This process requires conversion of into numeric data, the calculation of the mean followed by replacement.

```
df_new$Horsepower = as.numeric(df_new$Horsepower) # convert into numeric vari
able
horsepower_NA <- is.na(df_new$Horsepower) # check for existing NA
sum(horsepower_NA)

## [1] 2
```

There are 2 missing values that need replacing with mean values.

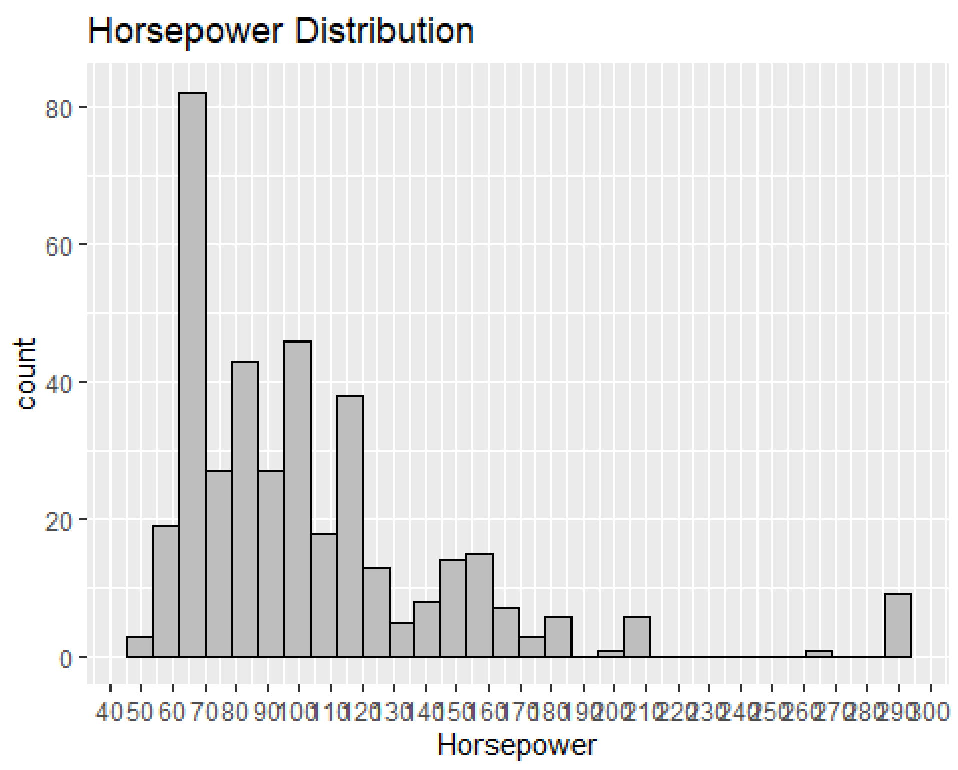
```
horsepower_mean <- mean(df_new$Horsepower, na.rm = TRUE) # calculate mean
df_new$Horsepower[horsepower_NA] <- horsepower_mean # replace in variable usi
ng horsepower_NA as index
sum(is.na(df_new$Horsepower)) # check for any NA left

## [1] 0
```

No missing values left, code ran properly.

Horsepower distribution

We can use a histogram to illustrate distribution and frequency of values:

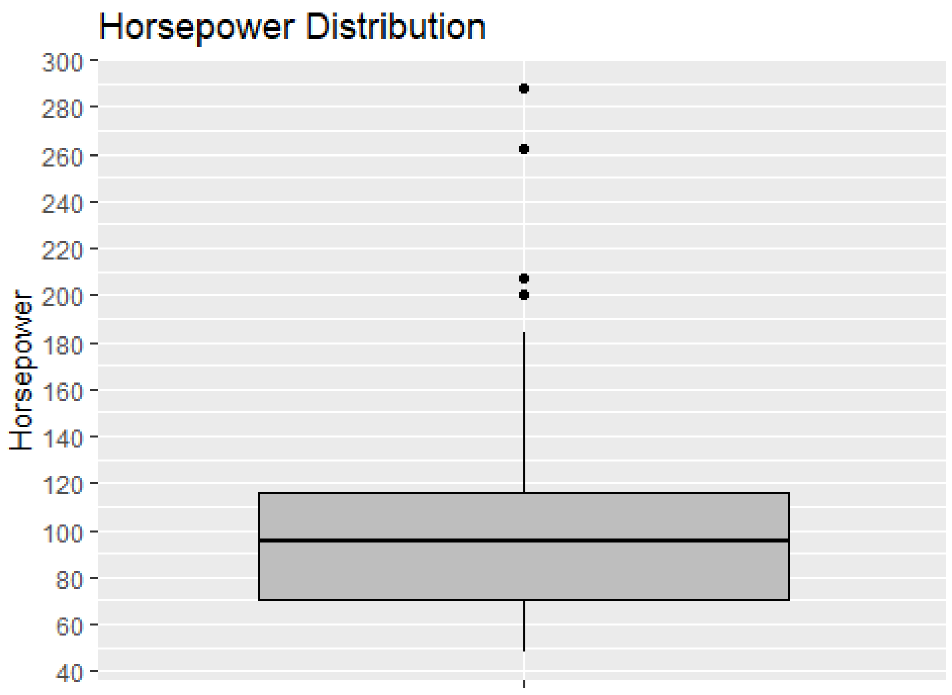
```
ggplot(df_new, aes(x = Horsepower)) +
  geom_histogram(fill = "gray", color = "black") +
  ggtitle("Horsepower Distribution") +
  scale_x_continuous(breaks = seq(0, 300, by = 10))

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



The distribution is right-skewed and most engines in the dataset range between approximately 45hp and 130hp. Less than 10 engines extend towards higher horsepower values, with outliers ranging from 200hp and 288hp.

We can also use a boxplot to visualise location statistics:

```
ggplot(df_new, aes(x = "", y = Horsepower)) +
  geom_boxplot(fill = "gray", color = "black") +
  ggtitle("Horsepower Distribution") +
  scale_y_continuous(breaks = seq(0, 300, by = 20)) +
  xlab("") +
  ylab("Horsepower")
```

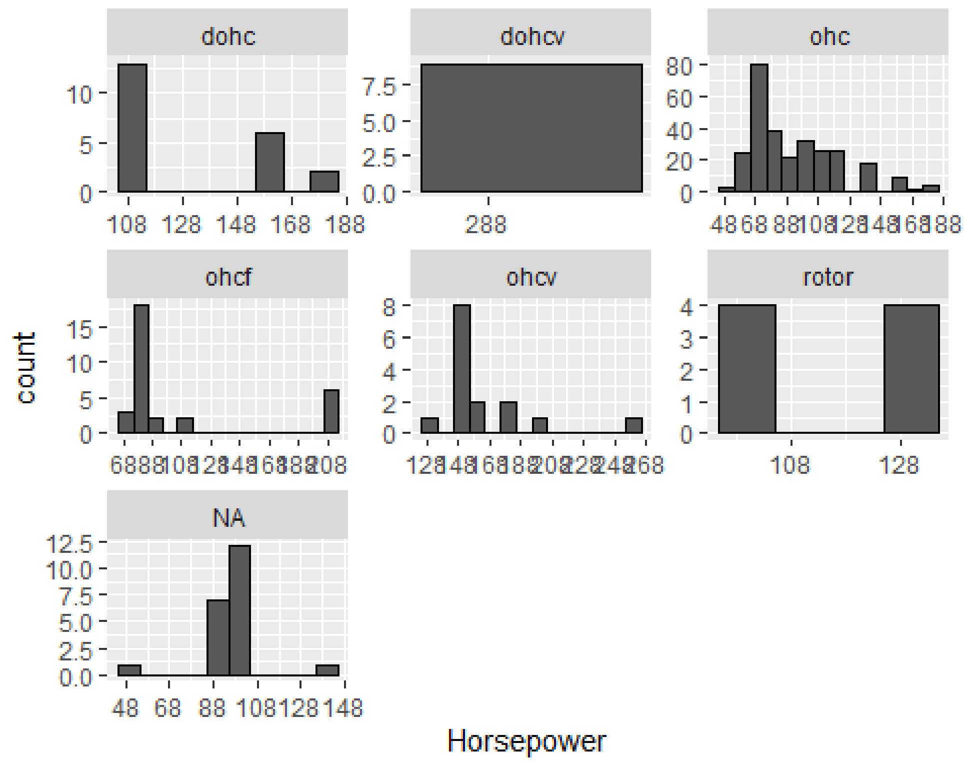


- The median is approximately at 95hp.
- The IQR is between the approximate range of 75hp to 117hp, 50% of the data falls in this range.
- 25% of the data falls below ~75hp, another 25% falls above 117hp.
- 4 outliers are identified, suggesting highly performant engines ranging from 200hp and 288hp.

Part 2

Horsepower distribution across engine types

```
ggplot(df_new, mapping = aes(x = Horsepower)) +
  geom_histogram(binwidth = 10, color = "black") +
  facet_wrap(~ EngineType, ncol = 3, scales = "free") +
  scale_x_continuous(
    breaks = seq(min(df_new$Horsepower), max(df_new$Horsepower), by = 20)
  )
```



```
labs(title = "Distribution of Horsepower by Engine Type")

## $title
## [1] "Distribution of Horsepower by Engine Type"
##
## attr(,"class")
## [1] "labels"
```

The distribution across engine types is uneven, with some engine types having very low counts while some dominate.

- DOHC engine: counts are concentrated at 108hp and unevenly distributed.
- DOHCV engine: the most performant engine type in horsepower is DOHCV, with 288hp. It is also the rarest type in the dataset.

- OHC engine: this type has the largest representation in the dataset with the most vehicle data and varied horsepower ranging from 48hp to 188hp, clustering around 68hp. Most vehicles at the Maintenance center seem to be of this engine type. This histogram is the most evenly distributed.

- OHCF engine: highly uneven frequency with lower extremes between 68hp and 88hp, with few counts at 208hp.

- OHCV engine: The second most performant engine in the dataset ranging from 128hp to 268hp.

- Rotor engine: Distinct concentration at 108 hp and 128 hp, with very few observations.

Horsepower distribution across engine sizes

We check for the minimum values and maximum in EngineSize to determine appropriate ranges:

```r
min(df_new$EngineSize, na.rm = TRUE)
```

```
## [1] 60
```

```r
max(df_new$EngineSize, na.rm = TRUE)
```

```
## [1] 320
```

We can then define the following size range:

- 60cc – 100cc

- 101cc – 150cc

- 151cc – 200cc

- 201cc – 250cc

- 251cc – 300cc

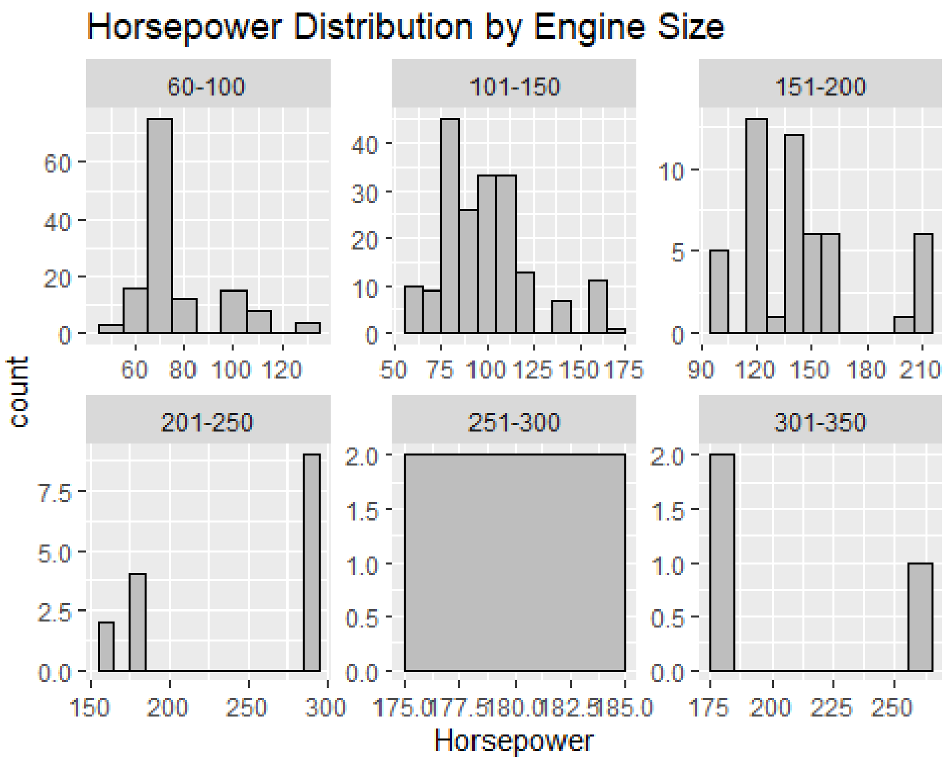
- 301cc – 350cc

We categorise and sort the engine sizes following the above ranges by creating a new column called EngineSizeGroup:

```r
df_new$EngineSizeGroup <- cut(
  df_new$EngineSize,
  breaks = c(60, 100, 150, 200, 250, 300, 350),
  labels = c("60-100", "101-150", "151-200", "201-250", "251-300", "301-350")
,
  include.lowest = TRUE
)
```

We then use this new column to generate the histogram:

```
ggplot(df_new, mapping = aes(x = Horsepower)) +
  geom_histogram(binwidth = 10, fill = "gray", color = "black") +
  facet_wrap(~ EngineSizeGroup, ncol = 3, scales = "free") +
  labs(title = "Horsepower Distribution by Engine Size")
```



The histogram once again shows highly uneven distribution across engine sizes. Most observations fall in the 60cc – 100cc, 101cc – 150cc and 151cc – 200cc groups. Larger engine sizes tend to be rare with very low counts below 10 and are characterised as outliers.

Part 3

Diesel vs Gas

Comparing the means of CityMPG by Fuel Type requires calculation of means:

```
diesel_mean <- mean(df_new$CityMpg[df_new$FuelType == "diesel"], na.rm = TRUE
)

gas_mean <- mean(df_new$CityMpg[df_new$FuelType == "gas"], na.rm = TRUE)

mean_diesel_gas <- matrix(c("Diesel", diesel_mean, "Gasoline", gas_mean),
                          ncol = 2, byrow = TRUE)
colnames(mean_diesel_gas) <- c("Fuel Type", "Average CityMpg")

mean_diesel_gas

##      Fuel Type  Average CityMpg
## [1,] "Diesel"   "31.3548387096774"
## [2,] "Gasoline" "24.8222222222222"
```

Diesel cars indeed have higher CityMPG than gasoline cars. This suggests diesel vehicles are more fuel-efficient in city commute compared to gasoline vehicles.

How DriveWheels affect fuel efficiency

We convert the data into the appropriate type for the following analysis:

```
df_new$DriveWheels <- as.factor(df_new$DriveWheels)
df_new$CityMpg <- as.numeric(df_new$CityMpg)
df_new$HighwayMpg <- as.numeric(df_new$HighwayMpg)
```

As we wish to compare means across multiple types of data, an ANOVA test will be used to check the impact of DriveWheels on CityMPG and HighwayMPG

- CityMPG
- H0: The type of DriveWheels does not affect CityMPG
- H1: At least one of the type of DriveWheels affects CityMPG

```
city_anova <- aov(CityMpg ~ DriveWheels, data = df_new)
summary(city_anova)

##              Df Sum Sq Mean Sq F value Pr(>F)
## DriveWheels   2   5848  2924.1   116.5 <2e-16 ***
## Residuals   388   9738    25.1
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p-value being smaller than 0.05, H0 is rejected and at least one type of DriveWheels affects CityMPG. We can verify with a Tukey HSD test:

```
TukeyHSD(city_anova)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = CityMpg ~ DriveWheels, data = df_new)
##
## $DriveWheels
##                diff       lwr        upr     p adj
## fwd-4wd  5.625623  2.665821  8.5854254 0.0000303
## rwd-4wd -2.490196 -5.519792  0.5394001 0.1305898
## rwd-fwd -8.115819 -9.378877 -6.8527616 0.0000000
```

The fwd-4wd comparison is significant with p-value < 0.05 and shows a positive difference: FWD types have positive impact on fuel efficiency compared to 4WD types.

The rwd-fwd comparison is also significant with p-value < 0.05, but with a negative difference: RWD types have a negative impact on fuel efficiency compared to FWD types.

Therefore, it can be said that FWD has a better effect on fuel efficiency when driving in the city, compared to 4WD and RWD types.

- HighwayMPG
- H0: The type of DriveWheels does not affect HighwayMPG
- H1: At least one of the type of DriveWheels affects HighwayMPG

```
highway_anova <- aov(HighwayMpg ~ DriveWheels, data = df_new)
summary(highway_anova)
```

```
##              Df Sum Sq Mean Sq F value Pr(>F)
## DriveWheels   2   6273  3136.3   119.4 <2e-16 ***
## Residuals   388  10189    26.3
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p-value being smaller than 0.05, H0 is rejected and at least one type of DriveWheels affects HighwayMPG. We can verify with a Tukey HSD test:

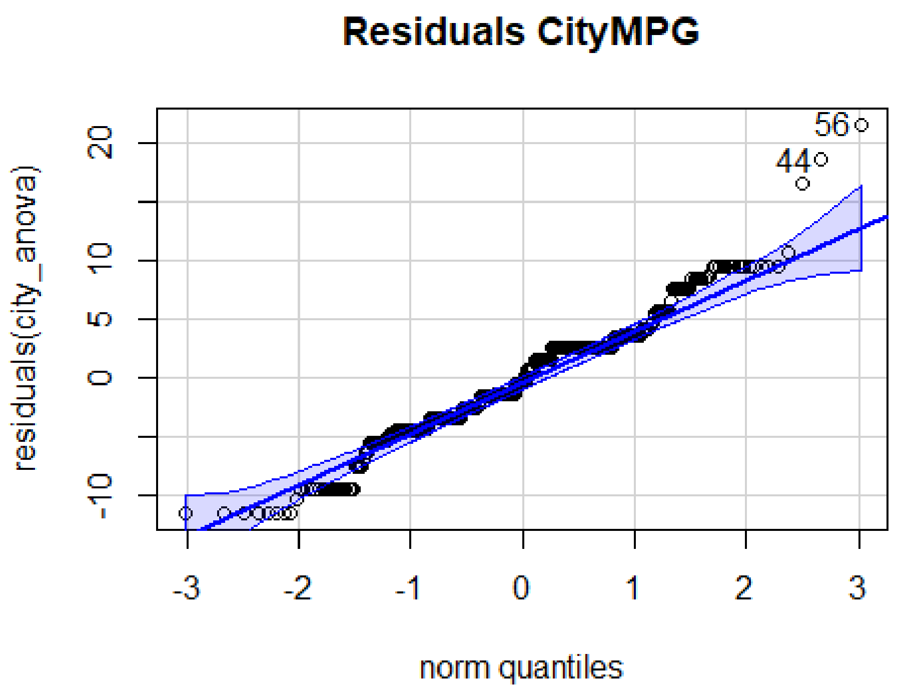
```
TukeyHSD(highway_anova)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = HighwayMpg ~ DriveWheels, data = df_new)
##
## $DriveWheels
##               diff       lwr       upr     p adj
## fwd-4wd  7.081505  4.053900 10.109111 0.0000002
## rwd-4wd -1.226343 -4.325341  1.872656 0.6209839
## rwd-fwd -8.307848 -9.599840 -7.015856 0.0000000
```

Same findings as above with p-values < 0.05 for fwd-4wd and rwd-fwd: FWD types have a positive influence on fuel efficiency in Highway context.

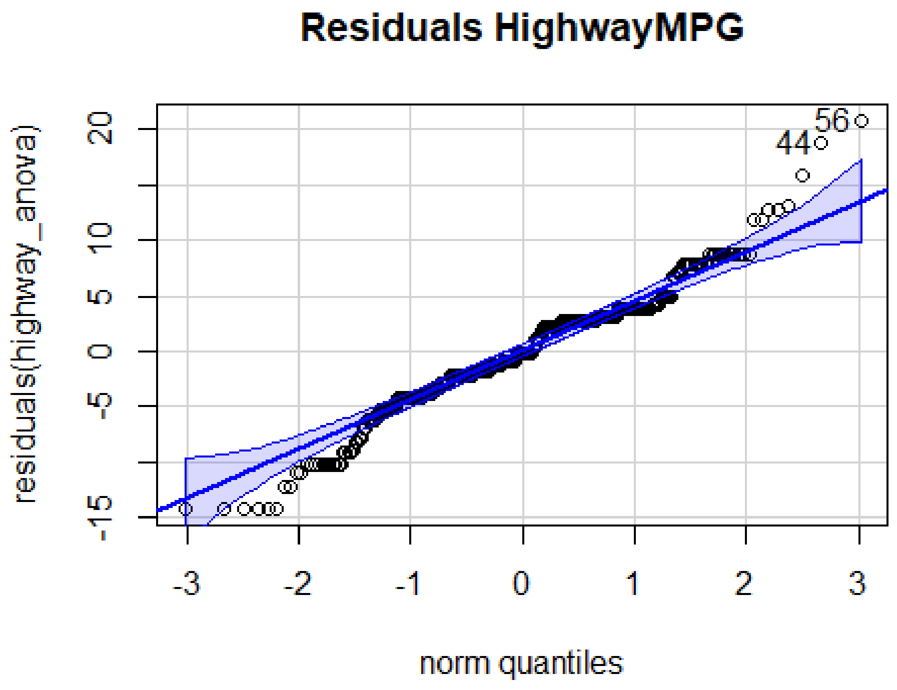
It should be noted that this statistical test is not the most reliable, as the residuals plot below does not strictly follow normality. It can be advised that further data collection or another statistical test should be employed to further study the effect of DriveWheels on fuel efficiency.

```
qqPlot(residuals(city_anova), main = "Residuals CityMPG")
```



```
## [1] 56 44
```

```
qqPlot(residuals(highway_anova), main = "Residuals HighwayMPG")
```



```
## [1] 56 44
```

Troubled engines

We can use R pipe code to filter out missing values and “No error”, then make sure that we only include observations where engine fails (ErrorCodes = 1). We then group troubles by most to least common, selecting only Top 5 troubles.

```
trouble_engine <- df_new %>%
  filter(!is.na(Troubles) & Troubles != "No error" & ErrorCodes == "1") %>%
# exclude NA, No error, filter where engine fails
  group_by(Troubles) %>%
  summarise(count = n()) %>%
  arrange(desc(count))
head(trouble_engine, 5)
```

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

Top 5 troubles that may have caused engine failure include: cylinders, ignition, noise, valve clearance and fans.

Top 5 troubles by engine type

- Diesel engine

```
diesel_troubles <- df_new %>%
  filter(!is.na(Troubles) & Troubles != "No error" & ErrorCodes == "1" & Fuel
Types == "diesel") %>%
  group_by(Troubles) %>%
  summarise(count = n()) %>%
  arrange(desc(count))
head(diesel_troubles, 5)
```

```
## # A tibble: 5 × 2
##   Troubles    count
##   <chr>       <int>
## 1 Cam shaft       3
## 2 Cylinders       3
## 3 Crank shaft     2
## 4 Stroke          2
## 5 ECU's power     1
```

Top 5 troubles where diesel engine failed include: cam shaft, cylinders, crank shaft, stroke, ECU's power.

- Gasoline engine

```
diesel_troubles <- df_new %>%
  filter(!is.na(Troubles) & Troubles != "No error" & ErrorCodes == "1" & Fuel
Types == "gas") %>%
  group_by(Troubles) %>%
  summarise(count = n()) %>%
  arrange(desc(count))
head(diesel_troubles, 5)
```

```
## # A tibble: 5 × 2
##   Troubles            count
##   <chr>               <int>
## 1 Cylinders              36
## 2 Ignition (finding)     22
## 3 Noise (finding)        19
## 4 Valve clearance        15
## 5 Fans                   13
```

Top 5 troubles for gas engine failure are in much higher numbers and in fact make up the majority of error cases. These include cylinders, ignition, noise, valve clearance and fans which reflect overall Top 5 engine troubles.

Part 4

Error type frequency

```
error_counts <- df_new %>%
  filter(!is.na(ErrorCodes)) %>%
  count(ErrorCodes, sort = TRUE)
rownames(error_counts)<- c("Engine Failure", "Other Component Failure", "No E
rror")
error_counts
```

```
##                         ErrorCodes   n
## Engine Failure                   1 191
## Other Component Failure         -1 171
## No Error                         0  29
```

Engine failure and other component failure occur the most, with respective counts of 191 and 171 cases,

Maintenance methods by engine type and manufacturer

The dataset is not suitable for statistical tests, which is why descriptive statistics will be used. Cross tabulation and histogram visualisations will provide a general and surface level representation of what factors may influence the maintenance methods.

Engine type & maintenance methods

```
trouble_vehicles <- df_new %>% filter(ErrorCodes == 1 | ErrorCodes == -1)
crosstab_enginetype <- table(trouble_vehicles$EngineType, trouble_vehicles$Me
thods)
print(crosstab_enginetype)
```

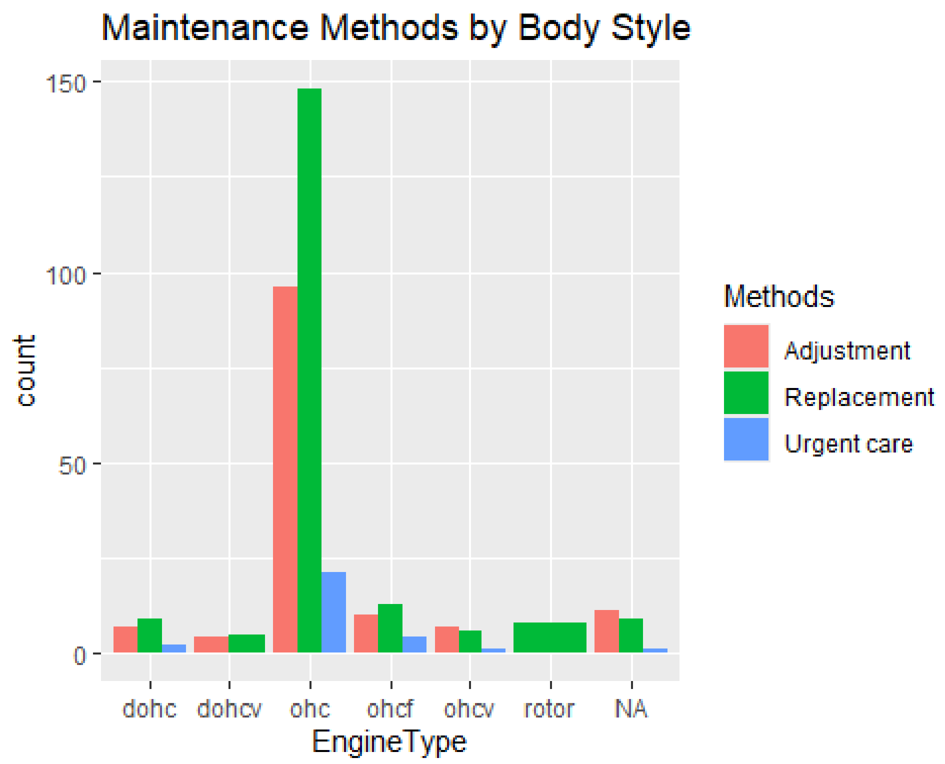
```
##
##          Adjustment Replacement Urgent care
##   dohc            7           9           2
##   dohcv           4           5           0
##   ohc            96         148          21
##   ohcf           10          13           4
##   ohcv            7           6           1
##   rotor           0           8           0
```

It can be seen that troubles occur mostly among in OHC engine type, followed by OHCF engine type. OHC engine types mostly need replacements with 148 cases, 96 adjustment cases and 21 urgent cases. This is because this engine type accounts for 286 observations among 391.

```
df_merged %>% filter(EngineType == "ohc") %>% summarise(Frequency = n())
```

```
##   Frequency
## 1       286
```

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

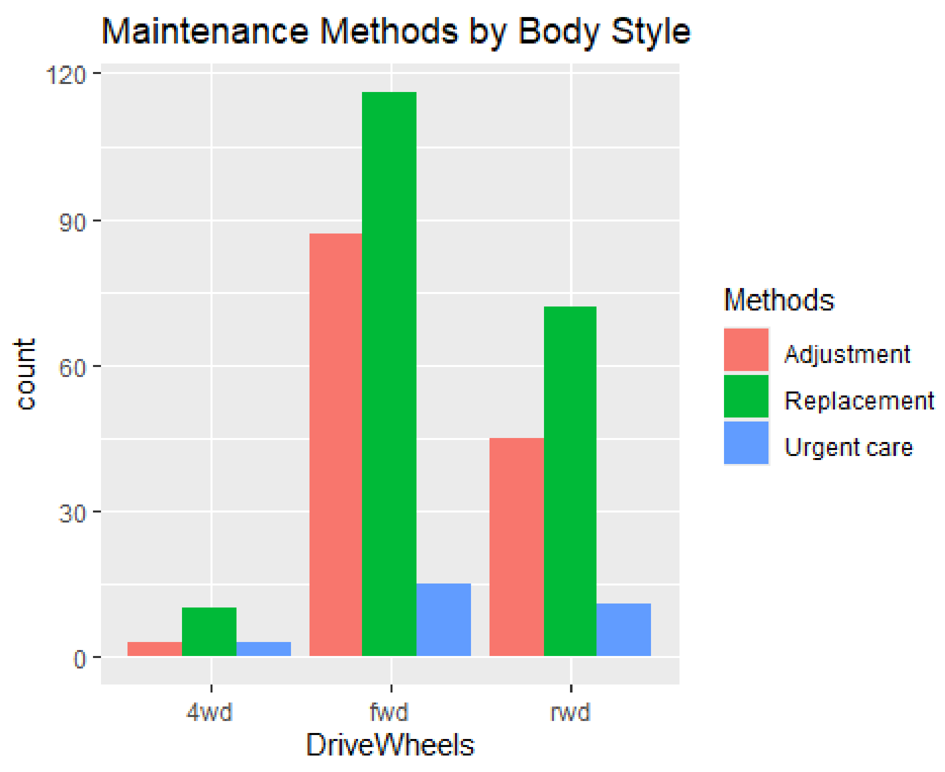


Drive wheels & maintenance methods

```
crosstab_manufacturer <- table(trouble_vehicles$DriveWheels, trouble_vehicles
$Methods)
print(crosstab_manufacturer)

##
##       Adjustment Replacement Urgent care
##   4wd          3          10           3
##   fwd         87         116          15
##   rwd         45          72          11
```

FWD drivewheels encounter the most troubles, with 116 replacement cases, 87 adjustment cases and 15 urgent care cases. IT can be assumed that most modern FWD cars use OHC engines, which explains the large frequency of troubles.

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



Due to the structure of this dataset, it is not suitable to use complex statistical tests to analyse trends. This dataset does not adequately follow normality and variance assumptions, therefore descriptive methods were used.

Part 5

Git repository link: https://github.com/naivalao/Assignment_Project_T3_2025