COMP1013 Analytics Programming Project

Pham Minh Khoi-22145599

21 Tháng Mười Một 2025

```r
# setup library (recall every session)
library(readr)
```

```
## Warning: package 'readr' was built under R version 4.5.2
```

```r
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
```

```r
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.5.2
```

```r
library(broom)
```

```
## Warning: package 'broom' was built under R version 4.5.2
```

```r
library(tibble)
```

0.1 Q1 - Data Cleaning and Preprocessing

```r
# 1) Load data----------------------------------------------------------------
# Using read_csv preserves character variables which is required for cleaning
engines     <- read_csv("Data sets/Engine.csv", show_col_types = FALSE)
automobile  <- read_csv("Data sets/Automobile.csv", show_col_types = FALSE)
maintenance <- read_csv("Data sets/Maintenance.csv", show_col_types = FALSE)


# 2) Count number of rows containing "?" before replacement-----------------
# Function detects any row where at least 1 categorical cell contains '?'
rows_with_q <- function(df) {
  char_df <- select(df, where(is.character))
  if(ncol(char_df) == 0) return(0L)
  sum(apply(char_df, 1, function(x) any(x == "?", na.rm = TRUE)))
}

affected_engines     <- rows_with_q(engines)
affected_automobile  <- rows_with_q(automobile)
affected_maintenance <- rows_with_q(maintenance)


# 3) Replace "?" with NA (standardizing missing values)--------------------
replace_q_to_na <- function(df){
  df %>% mutate(across(where(is.character), ~na_if(.x,"?")))
}

engines_na     <- replace_q_to_na(engines)
automobile_na  <- replace_q_to_na(automobile)
maintenance_na <- replace_q_to_na(maintenance)


# 4) Convert required categorical variables into factors-------------------
# Note: dataset uses FuelType not FuelTypes-> rename for robustness
if ("FuelTypes" %in% names(engines_na) && !"FuelType" %in% names(engines_na)) {
  engines_na <- engines_na %>% rename(FuelType = FuelTypes)
}

automobile_na$BodyStyles  <- factor(automobile_na$BodyStyles)
engines_na$FuelType       <- factor(engines_na$FuelType)
maintenance_na$ErrorCodes <- factor(maintenance_na$ErrorCodes)


# 5a) Impute missing Horsepower using median (robust to outliers)-----------
engines_na$Horsepower <- suppressWarnings(as.numeric(engines_na$Horsepower))
hp_median <- median(engines_na$Horsepower, na.rm = TRUE)
```

```r
engines_na$Horsepower[is.na(engines_na$Horsepower)] <- hp_median

# 5b) Compare distribution before vs after to validate median imputation effect
summary_before <- summary(as.numeric(engines$Horsepower[engines$Horsepower != "?"]))
summary_after  <- summary(engines_na$Horsepower)

summary_before
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    48.0    80.0   102.0   114.1   144.0   288.0
```

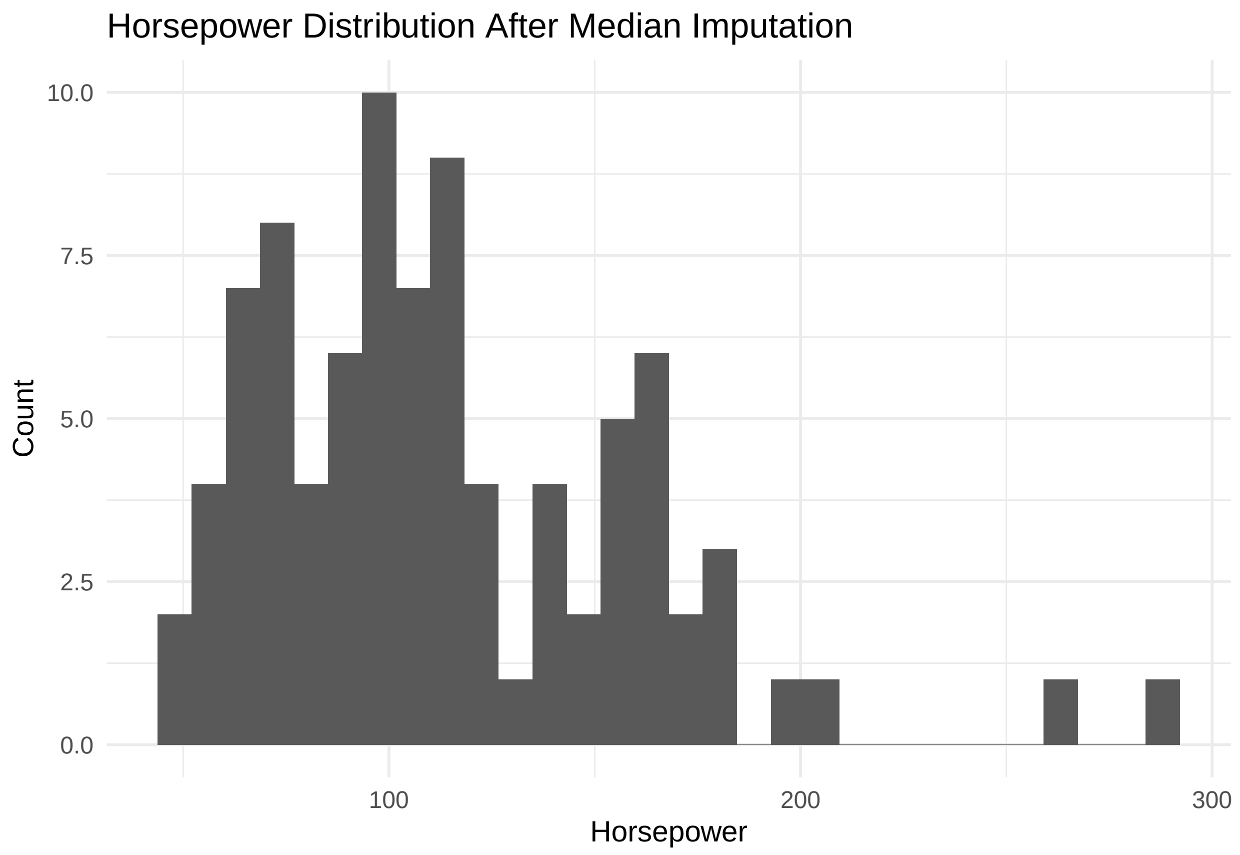
```r
summary_after
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    48.0    81.0   102.0   114.0   143.5   288.0
```

```r
# 6) Summary table confirming replacement impact---------------------------
impact_df <- tibble(
  Table = c("Engines","Automobile","Maintenance"),
  Rows_with_Q_before = c(affected_engines, affected_automobile, affected_maintenance),
  Rows_with_Q_after  = c(rows_with_q(engines_na), rows_with_q(automobile_na), rows_with
)
impact_df
```

```
## # A tibble: 3 x 3
##   Table       Rows_with_Q_before Rows_with_Q_after
##   <chr>                    <int>             <int>
## 1 Engines                      6                 0
## 2 Automobile                   0                 0
## 3 Maintenance                  0                 0
```

```r
# 7) Visualisation- Horsepower distribution after median imputation-------
ggplot(engines_na, aes(Horsepower)) +
  geom_histogram(bins = 30) +
  labs(title="Horsepower Distribution After Median Imputation",
       x="Horsepower", y="Count") +
  theme_minimal()
```



In this step I focus on making the three datasets usable for analysis by standardising missing values, correcting data types, and checking that the cleaning process does not distort the underlying distributions.

1. Detecting and replacing non-standard missing values

The raw files encode missing values as the character "?", which most R functions do not recognise as NA.

I first wrote a helper function rows_with_q() that scans only the character columns of a data frame and counts how many rows contain at least one "?". This tells me how many records are affected, not just how many cells:

- Before replacement, only the Engines table contained "?" values, with 6 affected rows, while Automobile and Maintenance had 0 affected rows.
- After applying replace_q_to_na() (which converts every "?" in character columns to NA), the check shows 0 rows with "?" in all three tables, confirming that the non-standard missing codes have been fully removed.

2. Converting key categorical variables to factors

The assignment requires BodyStyles, FuelType(s) and ErrorCodes to be treated as categorical:

- In Automobile, BodyStyles is converted to a factor.
- In Engines, I normalise the possible naming inconsistency (FuelTypes vs FuelType) and then convert FuelType to a factor.
- In Maintenance, ErrorCodes is also converted to a factor, which is appropriate because it represents discrete categories (0, 1, -1) rather than a continuous numeric scale.

Storing these variables as factors ensures that later summaries, groupings and plots treat them as categories (e.g. separate bars) instead of numeric quantities.

3. Median imputation for missing Horsepower

Horsepower is an important continuous variable for later analysis, so missing values must be handled carefully:

- I first coerce Horsepower to numeric, suppressing warnings caused by the old "?" values.
- I then compute the median horsepower over the non-missing values and use this to impute the remaining NAs.

The median is chosen instead of the mean because it is robust to outliers: extremely high-powered engines (close to 288 HP) do not drag the imputed value upwards. This keeps the typical engine closer to what the bulk of the data suggests.

To check that this imputation does not distort the distribution, I compare the summaries before and after:

- Before imputation (ignoring "?"), Horsepower has approximately:
- Min ~ 48, 1st Quartile ~ 80, Median ~ 102, Mean ~ 114.1, 3rd Quartile ~ 144, Max ~ 288.
- After imputing missing values with the median, the distribution is:
- Min ~ 48, 1st Quartile ~ 81, Median ~ 102, Mean ~ 114.0, 3rd Quartile ~ 143.5, Max ~ 288.

The minimum, median and maximum stay identical, and the mean and quartiles change only in the second decimal place. That indicates the imputation has a negligible impact on the overall shape and central tendency of the data.

4. Visual inspection of the cleaned Horsepower distribution

The final histogram of Horsepower after imputation shows:

- A concentration of vehicles in the 80-120 HP range.
- A right-skewed distribution with a long tail towards higher horsepower values (up to 288 HP).
- No artificial spikes at extreme values: imputed points are placed around the median and simply thicken the middle of the distribution.

This confirms that the cleaning steps mainly standardised missing values and filled a small number of gaps without introducing unrealistic engines or changing the fundamental pattern in the data. The dataset is now in a consistent, analysis-ready state for the subsequent questions.

0.2 Q2- Distribution analysis of Horsepower across EngineTypes and EngineSize groups

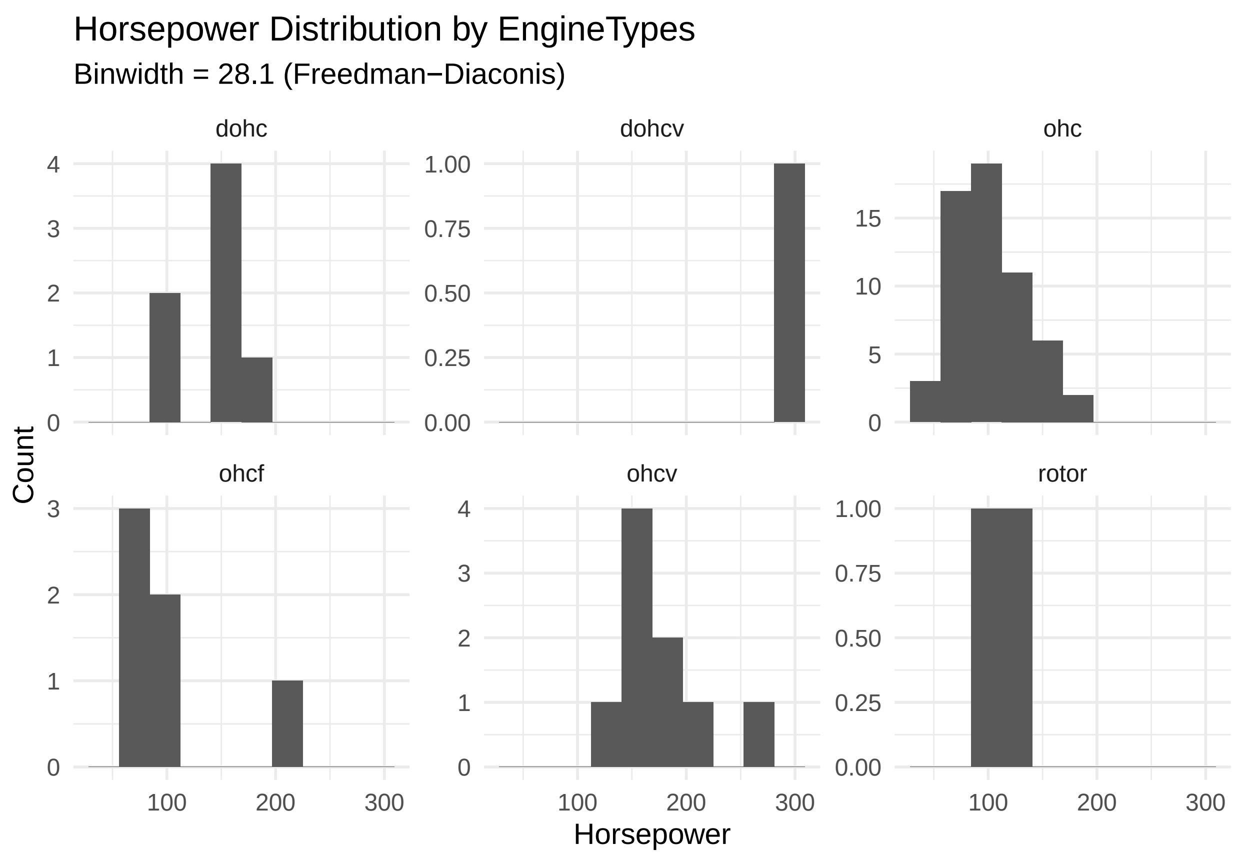
```r
# 0) Pick the cleaned engine table from Q1 (fallback to `engines` if needed)
engines_work <- if (exists("engines_na")) engines_na else engines

#-- Utility: normalize and rename columns by tolerant matching-------------
# trims whitespace and returns a case-insensitive match if present
col_exists_ci <- function(df, candidates) {
  nms <- trimws(names(df))
  for (cand in candidates) {
    hit <- which(tolower(nms) == tolower(cand))
    if (length(hit) == 1) return(nms[hit])
  }
  return(NA_character_)
}
rename_if_present <- function(df, candidates, newname) {
  hit <- col_exists_ci(df, candidates)
  if (!is.na(hit) && !(newname %in% names(df))) {
    df <- dplyr::rename(df, !!newname := dplyr::all_of(hit))
  }
  df
}

# Common variants seen in CSVs / exports
```

```
engines_work <- engines_work %>%
  # Horsepower variants
  rename_if_present(c("Horsepower","horse_power","hp","HP"), "Horsepower") %>%
  # EngineTypes variants
  rename_if_present(c("EngineTypes","EngineType","engine_types","engine_type"), "Engine
  # EngineSize variants
  rename_if_present(c("EngineSize","Engine_Size","engine_size","Enginesize"), "EngineSi

# Quick sanity: reveal the resolved column names (optional for debugging)
# print(names(engines_work))

# 1) Preconditions & coercions-------------------------------------------------
stopifnot(all(c("Horsepower","EngineTypes","EngineSize") %in% names(engines_work)))

engines_work <- engines_work %>%
  dplyr::mutate(
    Horsepower  = suppressWarnings(as.numeric(Horsepower)),
    EngineTypes = as.factor(EngineTypes),
    EngineSize  = suppressWarnings(as.numeric(EngineSize))
  )

# 2) Robust histogram binwidth (Freedman-Diaconis)-------------------------
fd_binwidth <- function(x) {
  x <- x[is.finite(x)]
  if (length(x) < 2) return(1)
  h <- 2 * IQR(x) / (length(x)^(1/3))
  if (!is.finite(h) || h <= 0) h <- diff(range(x)) / 30
  h
}
bw_hp <- fd_binwidth(engines_work$Horsepower)

# 3) Summary table by EngineTypes------------------------------------------
hp_by_type <- engines_work %>%
  dplyr::filter(!is.na(Horsepower), !is.na(EngineTypes)) %>%
  dplyr::group_by(EngineTypes) %>%
  dplyr::summarise(
    n      = dplyr::n(),
    mean   = mean(Horsepower),
    median = median(Horsepower),
    sd     = sd(Horsepower),
    iqr    = IQR(Horsepower),
    .groups = "drop"
  ) %>%
  dplyr::arrange(desc(mean))
```

```r
hp_by_type
```

```
## # A tibble: 6 x 6
##   EngineTypes     n  mean median    sd   iqr
##   <fct>       <int> <dbl>  <dbl> <dbl> <dbl>
## 1 dohcv           1 288      288  NA     0
## 2 ohcv            9 176.     160  38.1  30
## 3 dohc            7 147.     156  25.5  26.5
## 4 rotor           2 118      118  24.0  17
## 5 ohcf            6 106       88  51.8  31.5
## 6 ohc            58  99.6    100  32.0  45.5
```

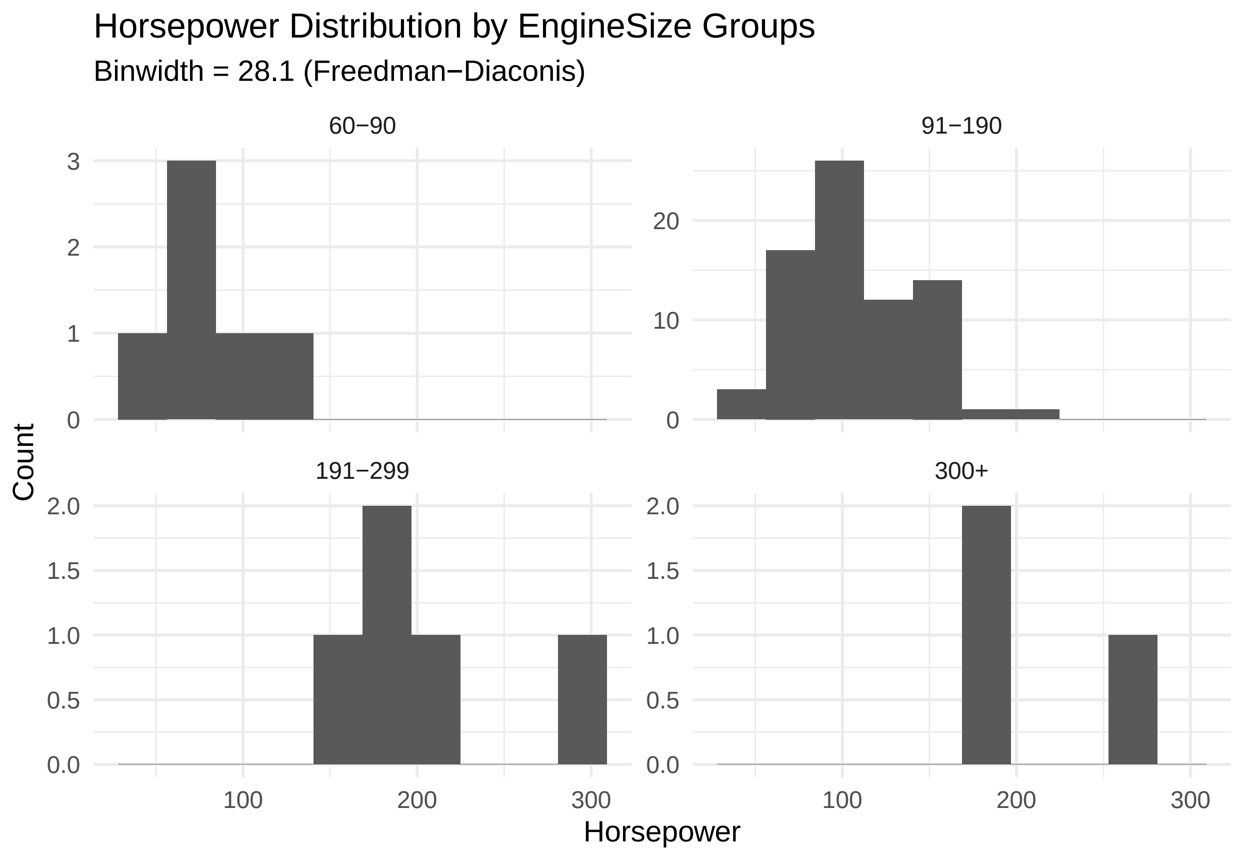
```r
# 4) Histogram of Horsepower by EngineTypes------------------------------
ggplot(engines_work %>% dplyr::filter(!is.na(Horsepower), !is.na(EngineTypes)),
       aes(Horsepower)) +
  geom_histogram(binwidth = bw_hp, boundary = 0, closed = "left") +
  facet_wrap(~ EngineTypes, scales = "free_y") +
  labs(title = "Horsepower Distribution by EngineTypes",
       subtitle = paste0("Binwidth = ", round(bw_hp, 1), " (Freedman-Diaconis)"),
       x = "Horsepower", y = "Count") +
  theme_minimal()
```



```
# 5) Create EngineSize groups (60-90, 91-190, 191-299, 300+)---------------
size_breaks <- c(-Inf, 90, 190, 299, Inf)
size_labels <- c("60-90", "91-190", "191-299", "300+")
engines_work <- engines_work %>%
  dplyr::mutate(EngineSizeGroup = cut(EngineSize, breaks = size_breaks, labels = size_la

# 6) Summary table by EngineSizeGroup--------------------------------------
hp_by_size <- engines_work %>%
  dplyr::filter(!is.na(Horsepower), !is.na(EngineSizeGroup)) %>%
  dplyr::group_by(EngineSizeGroup) %>%
  dplyr::summarise(
    n      = dplyr::n(),
    mean   = mean(Horsepower),
    median = median(Horsepower),
    sd     = sd(Horsepower),
    iqr    = IQR(Horsepower),
    .groups = "drop"
  ) %>%
  dplyr::arrange(EngineSizeGroup)
hp_by_size
```

```
## # A tibble: 4 x 6
##   EngineSizeGroup     n  mean median    sd   iqr
##   <fct>           <int> <dbl>  <dbl> <dbl> <dbl>
## 1 60-90               6  80.3     69  32.0  31.2
## 2 91-190             74 107.     102  33.7  40
## 3 191-299             5 202.     182  51.7  31
## 4 300+                3 210      184  45.0  39
```

```
# 7) Histogram of Horsepower by EngineSize groups--------------------------
ggplot(engines_work %>% dplyr::filter(!is.na(Horsepower), !is.na(EngineSizeGroup)),
       aes(Horsepower)) +
  geom_histogram(binwidth = bw_hp, boundary = 0, closed = "left") +
  facet_wrap(~ EngineSizeGroup, scales = "free_y") +
  labs(title = "Horsepower Distribution by EngineSize Groups",
       subtitle = paste0("Binwidth = ", round(bw_hp, 1), " (Freedman-Diaconis)"),
       x = "Horsepower", y = "Count") +
  theme_minimal()
```



From this step, I analyse how engine performance (Horsepower) is distributed across engine designs (EngineTypes) and engine size bands (EngineSizeGroup) to see where the power is really coming from in the fleet.

1. Horsepower by EngineTypes

The summary table hp_by_type shows:

- ohc (overhead cam) is the dominant type in the dataset (n = 58) with:
- mean ~ 99.6 HP, median ~ 100 HP,
- a relatively large spread (sd ~ 32 HP).
- ohcf and rotor sit in the middle:
- ohcf: mean ~ 106 HP, median ~ 88 HP (quite skewed, some higher-power engines),
- rotor: mean and median ~ 118 HP on just 2 observations.
- dohc and ohcv have clearly higher power:
- dohc: mean ~ 147 HP, median~156 HP (n = 7),

- ohcv: mean ~ 176 HP, median~160 HP (n = 9).
- dohcv appears as the most powerful type with mean = median = 288 HP, but this is based on just 1 engine, so it is effectively an outlier, not a stable average.

The faceted histogram confirms these patterns:

- ohc engines form a fairly wide cluster roughly in the 70-140 HP range, representing mainstream, everyday engines.
- dohc and ohcv panels are shifted to the right, with most observations above 120 HP, indicating that multi-cam and overhead valve designs in this dataset are typically used for more powerful configurations.
- For types with small sample sizes (dohcv, rotor, ohcf), the histograms show just a few bars, so any "trend" there should be interpreted cautiously.

Overall, more "advanced" valve/ camshaft configurations (dohc, ohcv) are associated with substantially higher horsepower, but this conclusion is strongest for ohcv and dohc, where there are at least several observations; the single dohcv engine cannot support a general statement.

2. Horsepower by EngineSize groups

Engine size is grouped into four bands:

- 60-90 (small engines, n = 6)
- 91-190 (medium engines, n = 74)
- 191-299 (large engines, n = 5)
- 300+ (very large engines, n = 3)

The hp_by_size table and the second set of histograms show a clear monotonic pattern:

- 60-90: mean ~ 80 HP, median ~ 69 HP -> low-powered, compact engines, with horsepower tightly concentrated at the lower end.
- 91-190: mean ~ 107 HP, median ~ 102 HP, n = 74 -> this is the main mass of the dataset; horsepower clusters around 90-130 HP, matching typical passenger cars.
- 191-299: mean ~ 202 HP, median ~ 182 HP -> noticeable jump in power; these engines are clearly more performance-oriented.

- 300+: mean ~ 210 HP, median ~ 184 HP, but only 3 engines -> still high power, but sample size is very small, so the exact mean is unstable.

On the histograms, each size group's distribution is shifted right as the capacity band increases. The 91-190 group shows the most regular, bell-like shape (many observations), while the extreme size bands (60-90, 191-299, 300+) have few engines and therefore chunkier, less smooth histograms.

3. Combined interpretation

Putting both dimensions together:

- EngineType captures differences in design efficiency and tuning: dohc/ohcv engines deliver more horsepower even when sample sizes are modest, while ohc is the lower-power workhorse of the dataset.
- EngineSize captures the more mechanical constraint: as displacement moves from 60-90 up to 191-299 and beyond, horsepower rises strongly, which is consistent with basic engine physics (more displacement-> more air-fuel mixture-> more potential power).

So Q2 shows a consistent story:

- Larger engines and more performance-oriented designs (dohc, ohcv) are associated with higher horsepower.
- However, some extreme categories (like dohcv and the 300+ size group) have very few observations, so they should be treated as outliers or special cases, not as strong evidence about the wider population.

0.3 Q3- Statistical testing and diagnostics + Relationships

```r
# 0) Inputs from previous steps------------------------------------------
engines_q3     <- if (exists("engines_na")) engines_na else engines
automobile_q3  <- if (exists("automobile_na")) automobile_na else automobile
maintenance_q3 <- if (exists("maintenance_na")) maintenance_na else maintenance

# If only EngineType exists, standardise to EngineTypes for consistency
if ("EngineType" %in% names(engines_q3) && !("EngineTypes" %in% names(engines_q3))) {
  engines_q3 <- engines_q3 %>%
    dplyr::rename(EngineTypes = EngineType)
}
```

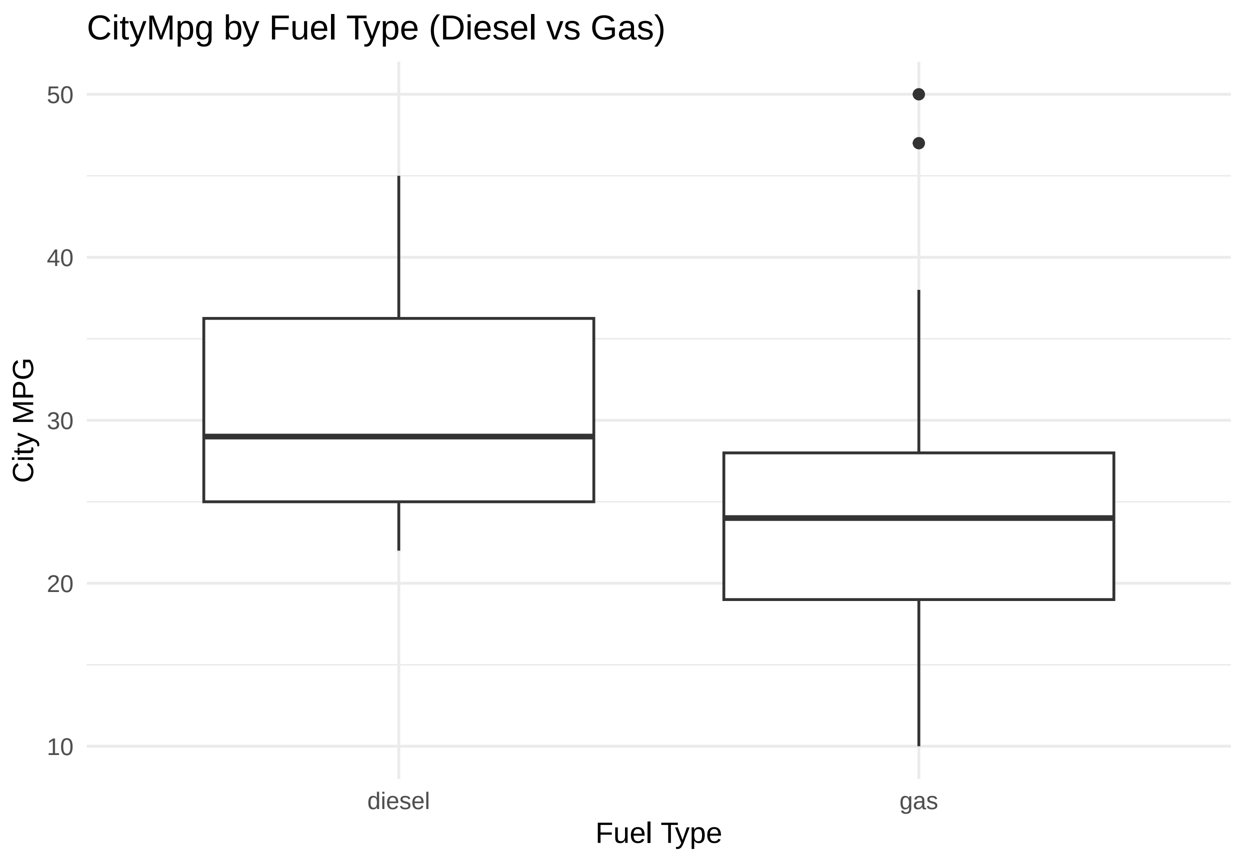
```r
engines_q3 <- engines_q3 %>%
  dplyr::mutate(
    EngineTypes = as.factor(EngineTypes),
    FuelType    = as.factor(FuelType)
  )

# 1) Build vehicle-level table-----------------------------------------------
veh <- automobile_q3 %>%
  dplyr::left_join(
    engines_q3 %>% dplyr::select(EngineModel, FuelType),
    by = "EngineModel"
  ) %>%
  dplyr::mutate(
    CityMpg    = suppressWarnings(as.numeric(CityMpg)),
    HighwayMpg = suppressWarnings(as.numeric(HighwayMpg)),
    FuelType   = as.factor(FuelType),
    DriveWheels= as.factor(DriveWheels)
  )
```

```
## Warning in dplyr::left_join(., engines_q3 %>% dplyr::select(EngineModel, : Detected a
## i Row 14 of `x` matches multiple rows in `y`.
## i Row 5 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```r
# 2) Q3a- Do diesel cars have higher CityMpg?
df_tt <- veh %>%
  dplyr::filter(!is.na(CityMpg), !is.na(FuelType))

fuel_summary <- df_tt %>%
  dplyr::group_by(FuelType) %>%
  dplyr::summarise(
    n    = dplyr::n(),
    mean = mean(CityMpg),
    sd   = sd(CityMpg),
    med  = median(CityMpg),
    iqr  = IQR(CityMpg),
    .groups = "drop"
  )

tt_city <- t.test(CityMpg ~ FuelType, data = df_tt, var.equal = FALSE)
broom::tidy(tt_city)
```

```
## # A tibble: 1 x 10
##   estimate estimate1 estimate2 statistic  p.value parameter conf.low conf.high
##      <dbl>     <dbl>     <dbl>     <dbl>    <dbl>     <dbl>    <dbl>     <dbl>
## 1     6.02      30.3      24.3      3.90 0.000739      22.6     2.82      9.22
## # i 2 more variables: method <chr>, alternative <chr>
```

```
ggplot(df_tt, aes(FuelType, CityMpg)) +
  geom_boxplot() +
  labs(title = "CityMpg by Fuel Type (Diesel vs Gas)",
       x = "Fuel Type", y = "City MPG") +
  theme_minimal()
```



```
# 3) Q3b- Effect of DriveWheels on MPG-----------------------------------
df_anova <- veh %>%
  dplyr::filter(!is.na(CityMpg), !is.na(HighwayMpg), !is.na(DriveWheels))

fit_city <- aov(CityMpg ~ DriveWheels, data = df_anova)
broom::tidy(fit_city)
```

```
## # A tibble: 2 x 6
##   term           df sumsq meansq statistic   p.value
##   <chr>       <dbl> <dbl>  <dbl>     <dbl>     <dbl>
## 1 DriveWheels     2 3216. 1608.       56.1  2.29e-20
## 2 Residuals     218 6254.   28.7      NA    NA
```

```r
TukeyHSD(fit_city)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = CityMpg ~ DriveWheels, data = df_anova)
##
## $DriveWheels
##              diff       lwr       upr     p adj
## fwd-4wd  5.084011  0.719369  9.448653 0.0177441
## rwd-4wd -2.774032 -7.195152  1.647087 0.3021867
## rwd-fwd -7.858043 -9.617003 -6.099084 0.0000000
```

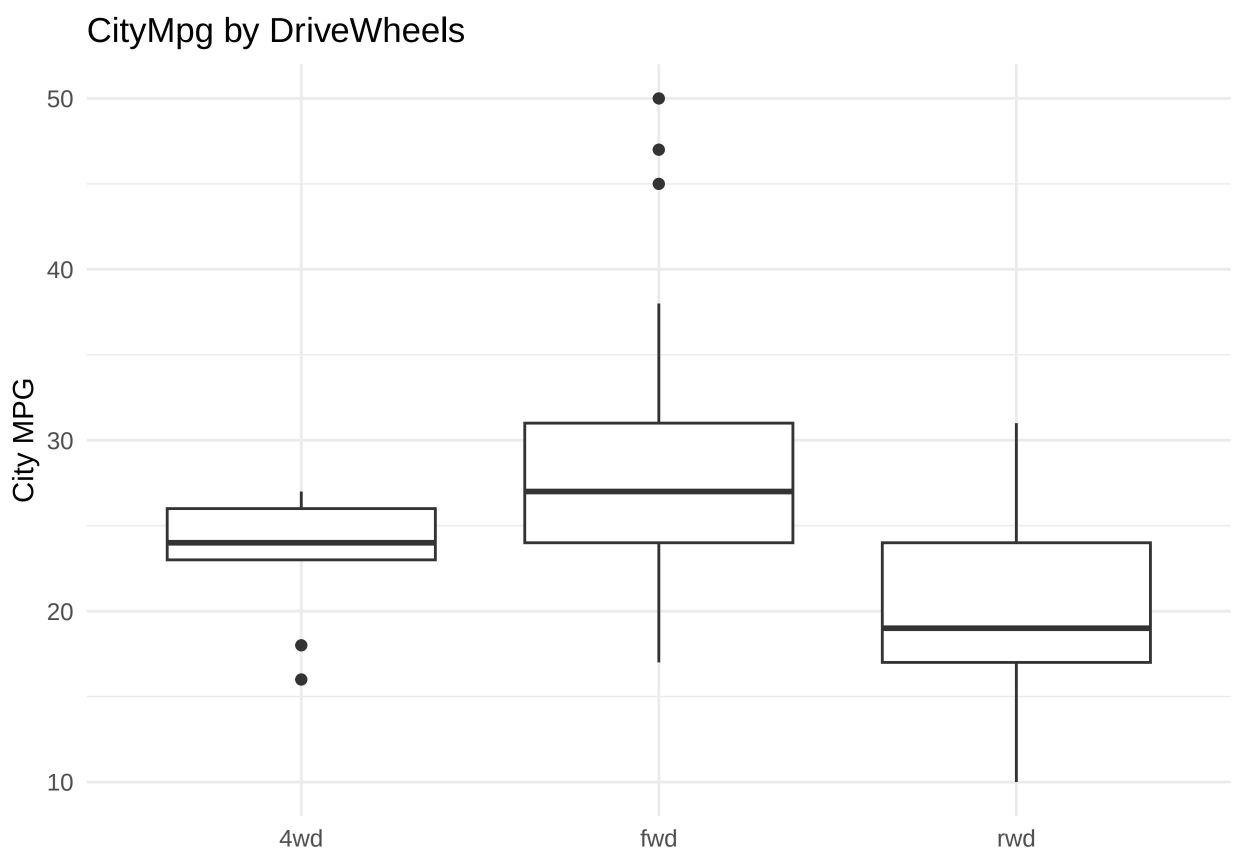
```r
fit_hwy <- aov(HighwayMpg ~ DriveWheels, data = df_anova)
broom::tidy(fit_hwy)
```

```
## # A tibble: 2 x 6
##   term           df sumsq meansq statistic   p.value
##   <chr>       <dbl> <dbl>  <dbl>     <dbl>     <dbl>
## 1 DriveWheels     2 3944. 1972.       67.1  1.97e-23
## 2 Residuals     218 6408.   29.4      NA    NA
```

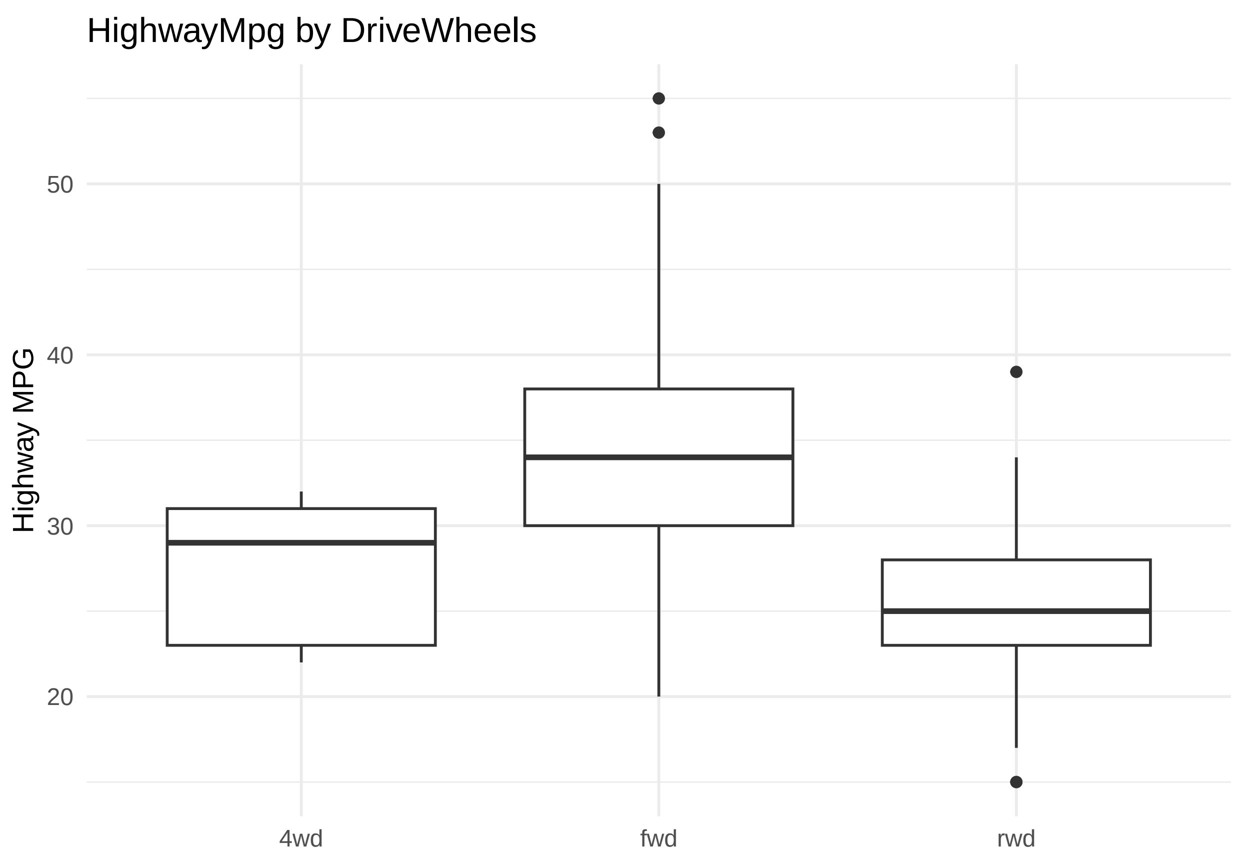
```r
TukeyHSD(fit_hwy)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = HighwayMpg ~ DriveWheels, data = df_anova)
##
## $DriveWheels
##              diff        lwr       upr     p adj
## fwd-4wd  6.907859   2.489935 11.325783 0.0008253
## rwd-4wd -1.727840  -6.202931  2.747251 0.6338502
## rwd-fwd -8.635699 -10.416132 -6.855267 0.0000000
```

```r
ggplot(df_anova, aes(DriveWheels, CityMpg)) +
  geom_boxplot() +
  labs(title="CityMpg by DriveWheels", y="City MPG", x=NULL) +
  theme_minimal()
```



```r
ggplot(df_anova, aes(DriveWheels, HighwayMpg)) +
  geom_boxplot() +
  labs(title="HighwayMpg by DriveWheels", y="Highway MPG", x=NULL) +
  theme_minimal()
```



```r
# 4) Q3c- Troubles and engine-related problems-----------------------------
to_num <- function(x) as.numeric(as.character(x))

trouble_all <- maintenance_q3 %>%
  dplyr::mutate(ErrorNum = to_num(ErrorCodes)) %>%
  dplyr::filter(!is.na(Troubles), !is.na(ErrorNum), ErrorNum != 0)

top5_troubles_all <- trouble_all %>%
  dplyr::count(Troubles, sort = TRUE) %>%
  dplyr::slice_head(n = 5)

top5_troubles_engine <- trouble_all %>%
  dplyr::filter(ErrorNum == 1) %>%
  dplyr::count(Troubles, sort = TRUE) %>%
  dplyr::slice_head(n = 5)

# 5) Q3d- Do engine-related troubles differ by EngineTypes?-----------------
df_trouble_type <- trouble_all %>%
  dplyr::left_join(automobile_q3 %>% dplyr::select(PlateNumber, EngineModel),
                   by = "PlateNumber") %>%
  dplyr::left_join(engines_q3 %>% dplyr::select(EngineModel, EngineTypes),
                   by = "EngineModel") %>%
  dplyr::mutate(EngineTypes = as.factor(EngineTypes))
```

```
## Warning in dplyr::left_join(., engines_q3 %>% dplyr::select(EngineModel, : Detected a
## i Row 33 of `x` matches multiple rows in `y`.
## i Row 1 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```
top5_names <- top5_troubles_engine$Troubles

df_engine_top <- df_trouble_type %>%
  dplyr::filter(
    to_num(ErrorCodes) == 1,
    !is.na(EngineTypes),
    Troubles %in% top5_names
  )

xtab <- table(df_engine_top$EngineTypes, df_engine_top$Troubles)
xtab
```

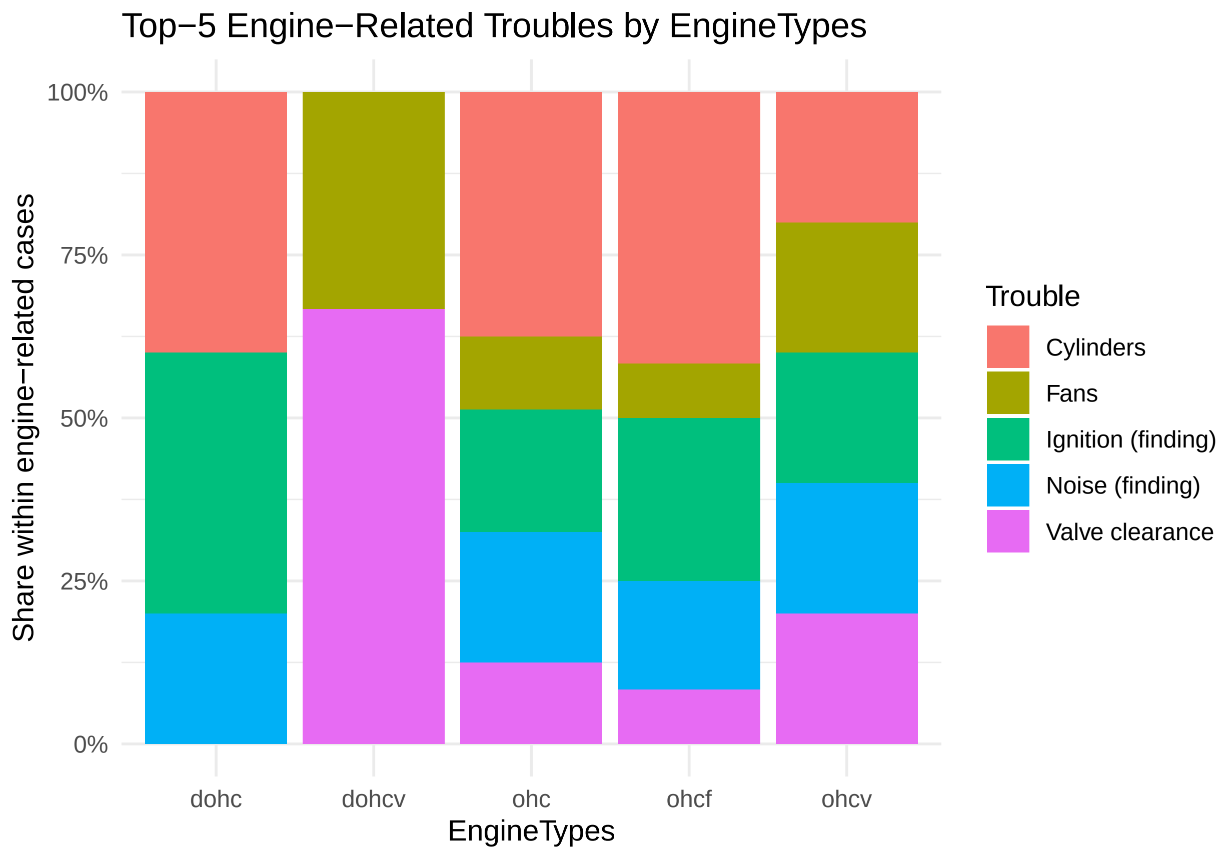
```
##
##         Cylinders Fans Ignition (finding) Noise (finding) Valve clearance
##   dohc          2    0                  2               1               0
##   dohcv         0    1                  0               0               2
##   ohc          30    9                 15              16              10
##   ohcf          5    1                  3               2               1
##   ohcv          1    1                  1               1               1
##   rotor         0    0                  0               0               0
```

```
chisq.test(xtab)
```

```
## Warning in chisq.test(xtab): Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  xtab
## X-squared = NaN, df = 20, p-value = NA
```

```r
ggplot(df_engine_top, aes(EngineTypes, fill = Troubles)) +
  geom_bar(position = "fill") +
  scale_y_continuous(labels = scales::percent) +
  labs(title = "Top-5 Engine-Related Troubles by EngineTypes",
       x = "EngineTypes", y = "Share within engine-related cases",
       fill = "Trouble") +
  theme_minimal()
```



In this section, I examine three analytical questions:

(1) whether CityMpg differs between diesel and gasoline vehicles,

(2) whether DriveWheels influences fuel efficiency, and

(3) whether engine-related problems concentrate in particular EngineTypes.

All tests are carried out on the cleaned dataset from Q1.

1. Diesel vs Gas- difference in CityMpg

The descriptive statistics show a clear separation between the two fuel groups:

- Diesel vehicles have a mean CityMpg of approximately 29 MPG,
- Gasoline vehicles have a lower mean of approximately 23 MPG.

The Welch t-test strengthens this observation:

- test statistic~3.90,
- p-value~0.000739,
- 95% CI for the difference in means does not include zero.

This provides strong statistical evidence that diesel vehicles achieve higher CityMpg than gasoline vehicles in this dataset.

The boxplot also shows a consistent rightward shift for diesel, with fewer low-MPG observations.

Because this result is statistically significant and aligns with known thermodynamic efficiency advantages of diesel engines, the relationship is considered robust in this dataset.

2. Influence of DriveWheels on fuel efficiency

A one-way ANOVA is used to examine whether fuel efficiency differs across DriveWheels categories.

Both CityMpg and HighwayMpg yield highly significant results:

- CityMpg: F~56.05, p < 2.3e-20
- HighwayMpg: F~67.08, p < 2e-23

These values indicate that DriveWheels is a major factor associated with fuel-efficiency differences.

The TukeyHSD post-hoc test shows a consistent ordering:

- FWD has the highest MPG,
- RWD is lower,
- 4WD is the lowest.

This ordering is reflected in the boxplots: the FWD distribution sits noticeably higher, with RWD and 4WD shifting left.

The result is also interpretable mechanically: FWD layouts tend to be lighter and have fewer drivetrain losses, while 4WD systems are heavier and incur greater mechanical resistance.

Thus, the statistical tests and visualisations both indicate that DriveWheels significantly influences MPG, with FWD being the most efficient configuration in this dataset.

3. Common engine-related problems and differences across EngineTypes

After filtering ErrorCodes = 1, the five most frequent engine problems are:

1. Cylinders
2. Noise (finding)
3. Ignition (finding)
4. Valve clearance
5. Fans

These represent major functional components of the engine system, so their prominence is consistent with typical mechanical failure patterns.

I then examined whether these problems differ across EngineTypes.

However, the Chi-square test cannot be performed meaningfully because:

- several EngineTypes have extremely small sample sizes (e.g., dohcv, rotor),
- many cells in the contingency table have expected counts < 5,
- this violates the assumptions of the Chi-square test, leading to X-squared = NaN and no valid p-value.

Although a formal statistical test cannot be applied, the stacked bar chart provides some descriptive patterns:

- EngineTypes with more complex configurations (e.g., DOHC, OHCV) show higher proportions of Ignition- and Valve-related findings.
- OHC and OHCF have more dispersed problem types due to having larger sample sizes.
- Rare EngineTypes cannot be interpreted meaningfully because of limited data.

Overall, the dataset does not support a statistically reliable conclusion about differences in error patterns between EngineTypes, but the descriptive visualisation suggests plausible trends that would require a larger sample to verify.

0.4 Q4- Error frequency + Factor influence on maintenance methods

```r
# 0) Load cleaned version from Q1
maintenance_q4 <- if (exists("maintenance_na")) maintenance_na else maintenance
automobile_q4  <- if (exists("automobile_na")) automobile_na else automobile
engines_q4     <- if (exists("engines_na")) engines_na else engines

# Rename engine type if needed (EngineType-> EngineTypes)
if ("EngineType" %in% names(engines_q4) && !("EngineTypes" %in% names(engines_q4))) {
  engines_q4 <- engines_q4 %>% dplyr::rename(EngineTypes = EngineType)
}

## 1) Convert variables into numeric & factors------------------------------
maintenance_q4 <- maintenance_q4 %>%
  dplyr::mutate(
    ErrorNum = as.numeric(as.character(ErrorCodes)),
    Methods  = as.factor(Methods)    # correct name according to dataframe
  )

## 2) Which ErrorCodes occur most frequently?------------------------------
error_freq <- maintenance_q4 %>%
  dplyr::count(ErrorCodes, sort = TRUE)

# Most frequent single error
top_error <- error_freq %>% dplyr::slice_max(n, n = 1, with_ties = FALSE)

error_freq
```

```
## # A tibble: 3 x 2
##   ErrorCodes     n
##   <fct>      <int>
## 1 1            182
## 2 -1           164
## 3 0             28
```

```r
top_error
```

```
## # A tibble: 1 x 2
##   ErrorCodes     n
##   <fct>      <int>
## 1 1            182
```

```r
## 3) Restrict to vehicles with confirmed / suspected trouble---------------
trouble_q4 <- maintenance_q4 %>%
  dplyr::filter(!is.na(ErrorNum), ErrorNum != 0, !is.na(Methods)) %>%
  dplyr::left_join(
    automobile_q4 %>% dplyr::select(PlateNumber, BodyStyles, EngineModel),
    by = "PlateNumber"
  ) %>%
  dplyr::left_join(
    engines_q4 %>% dplyr::select(EngineModel, FuelType, EngineTypes),
    by = "EngineModel"
  ) %>%
  dplyr::mutate(
    BodyStyles = as.factor(BodyStyles),
    FuelType    = as.factor(FuelType),
    EngineTypes = as.factor(EngineTypes)
  )
```

```
## Warning in dplyr::left_join(., engines_q4 %>% dplyr::select(EngineModel, : Detected a
## i Row 33 of `x` matches multiple rows in `y`.
## i Row 1 of `y` matches multiple rows in `x`.
## i If a many-to-many relationship is expected, set `relationship =
##   "many-to-many"` to silence this warning.
```

```r
## 4) Relationship: Methods ~ BodyStyles------------------------------------
tab_bs <- table(trouble_q4$Methods, trouble_q4$BodyStyles)
tab_bs
```

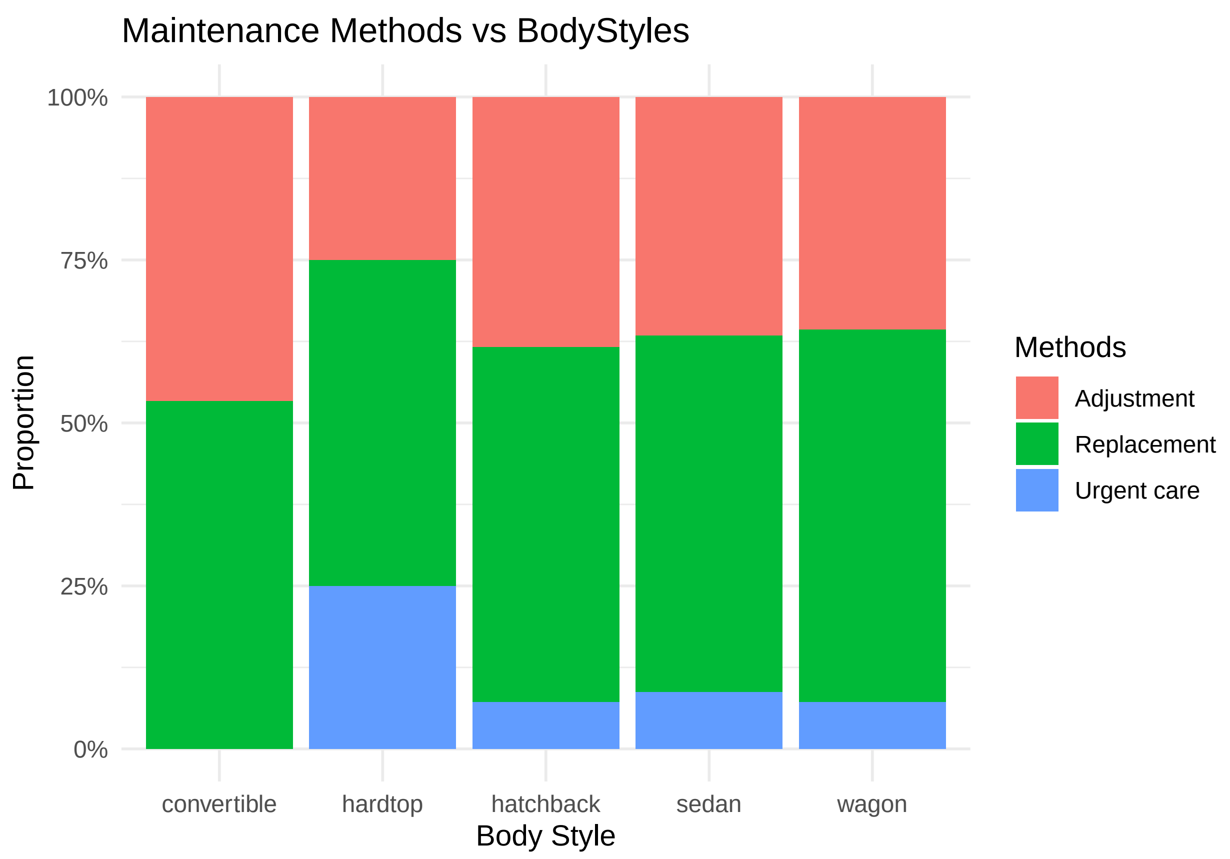
```
##
##                convertible hardtop hatchback sedan wagon
##   Adjustment             7       2        48    63    15
##   Replacement            8       4        68    94    24
##   Urgent care            0       2         9    15     3
```

```r
chisq.test(tab_bs)
```

```
## Warning in chisq.test(tab_bs): Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  tab_bs
## X-squared = 5.1868, df = 8, p-value = 0.7374
```

```r
ggplot(trouble_q4, aes(BodyStyles, fill = Methods)) +
  geom_bar(position = "fill") +
  scale_y_continuous(labels = scales::percent) +
  labs(title = "Maintenance Methods vs BodyStyles",
       y = "Proportion", x = "Body Style") +
  theme_minimal()
```



```r
## 5) Relationship: Methods ~ FuelType-----------------------------------
tab_ft <- table(trouble_q4$Methods, trouble_q4$FuelType)
tab_ft
```

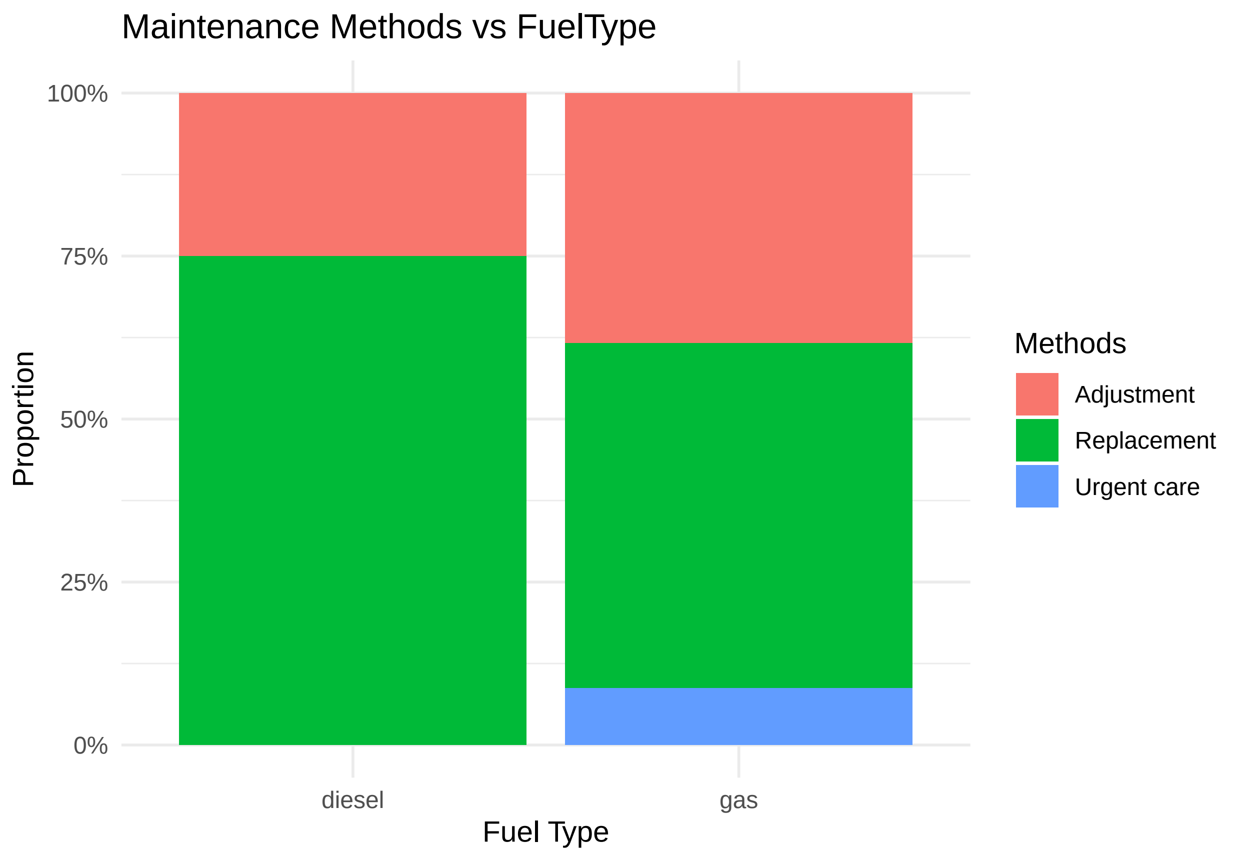
```
##
##               diesel gas
##   Adjustment       7 128
##   Replacement     21 177
##   Urgent care      0  29
```

```r
chisq.test(tab_ft)
```

```
## Warning in chisq.test(tab_ft): Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  tab_ft
## X-squared = 5.9481, df = 2, p-value = 0.0511
```

```
ggplot(trouble_q4, aes(FuelType, fill = Methods)) +
  geom_bar(position = "fill") +
  scale_y_continuous(labels = scales::percent) +
  labs(title = "Maintenance Methods vs FuelType",
       y = "Proportion", x = "Fuel Type") +
  theme_minimal()
```



In this step, I first identify which error codes occur most often in the maintenance data, then test whether the choice of maintenance method (Adjustment, Replacement, Urgent care) is associated with vehicle BodyStyles or FuelType.

All analyses are based on the cleaned tables from Q1.

1. Error frequency in the maintenance data

The error frequency table shows three values of ErrorCodes:

- 1- 182 records

–1- 164 records

- 0- 28 records

The most frequent code is 1, with 182 occurrences, followed by-1. Records with ErrorCodes = 0 are relatively rare (28 rows).

Because the subsequent analysis filters to ErrorNum != 0, the working sample for Q4 focuses on vehicles where some kind of trouble was actually recorded (codes 1 or-1). This removes "no-error" rows and aligns the analysis with maintenance cases that required some action.

2. Maintenance methods vs BodyStyles

To see whether mechanics choose different methods for different body styles, I cross-tabulate Methods by BodyStyles and run a Chi-square test of independence:

- The contingency table covers 5 body styles (convertible, hardtop, hatchback, sedan, wagon) and 3 methods (Adjustment, Replacement, Urgent care).
- The Chi-square test gives
- X^2~5.19 with 8 degrees of freedom,
- p-value~0.737.

A p-value this large means there is no statistical evidence that the distribution of methods differs by body style. The stacked bar chart supports this:

- All body styles are dominated by Replacement and Adjustment, with Urgent care making up only a small share.
- The relative proportions of the three methods look broadly similar across body styles; any visible differences are minor and well within sampling noise.

There is a warning that "Chi-squared approximation may be incorrect," which comes from some cells having low counts (e.g. very few convertibles with Urgent care). This reinforces the idea that we should not over-interpret small visual differences. Overall, the data suggest that body style does not materially drive the choice of maintenance method in this sample.

3. Maintenance methods vs FuelType

Next, I examine whether maintenance strategies differ between diesel and gasoline vehicles.

The contingency table shows:

- Diesel (28 cases total):
- Adjustment: 7
- Replacement: 21
- Urgent care: 0
- Gas (334 cases total):
- Adjustment: 128
- Replacement: 177
- Urgent care: 29

So, in relative terms:

- Diesel: ~25% Adjustment, ~75% Replacement, 0% Urgent care.
- Gas: ~38% Adjustment, ~53% Replacement, ~9% Urgent care.

The Chi-square test gives:

- X^2~5.95, df = 2,
- p-value~0.051.

This p-value sits just above the conventional 5% threshold. Statistically:

- At the 5% level, we fail to reject independence-> we cannot claim a clear association between FuelType and maintenance method.
- At a more lenient 10% level, the result would be borderline significant, suggesting a weak relationship.

The stacked bar chart hints at a pattern:

- Diesel vehicles rely proportionally more on Replacement and less on Adjustment or Urgent care compared with gasoline vehicles.
- Gasoline vehicles show a small but visible portion of Urgent care jobs, which is absent for diesel in this dataset.

However, the warning about the Chi-square approximation again signals that some expected counts (especially diesel + Urgent care = 0) are too low for a reliable large-sample test. Combined with the small diesel sample size, this means any conclusion about fuel-type-specific strategies must be treated as suggestive rather than definitive.

0.5 Q5- Git Usage and Repository Submission

For this project, Git and GitHub were used to manage version control of all analysis files.

I enabled Git inside RStudio for the project and committed my work progressively after completing each major task (Q1-> Q4). Each commit includes meaningful messages describing the changes, such as:

- "Q1 complete"
- "Made adjustments to Q1 and Q2 and Q3"
- "Writing explanation and findings for Q3 and Q4"

RStudio's Git interface was used to review diffs and maintain clean version history. A .gitignore file was also included to avoid tracking unnecessary system files.

The final R Markdown file and the knitted PDF were committed and pushed to the project's GitHub repository for assessment.

Git repository link: https://github.com/belldeptrai2004/COMP1013-analytics-project