DTS301TC_32

Jiayuan Zhu, Zhiyi Zhao, Minshu Xu

10/3/2022

T1

T1-1

- Load the CSV file; show the dimensionality, structure and summary of the dataset.

```
# Load Data
library(readr)
ford <- read_csv("ford.csv")
```

```
## Rows: 17976 Columns: 9
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): model, transmission, fuelType
## dbl (6): year, price, mileage, tax, mpg, engineSize
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# Dimensionality
dim(ford)
```

```
## [1] 17976     9
```

```
# Structure
  str(ford)
```

```
## spec_tbl_df [17,976 x 9] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ model       : chr [1:17976] "Fiesta" "Focus" "Focus" "Fiesta" ...
##  $ year        : num [1:17976] 2017 2018 2017 2019 2019 ...
##  $ price       : num [1:17976] 12000 14000 13000 17500 16500 10500 22500 9000 25500 10000 ...
##  $ transmission: chr [1:17976] "Automatic" "Manual" "Manual" "Manual" ...
##  $ mileage     : num [1:17976] 15944 9083 12456 10460 1482 ...
##  $ fuelType    : chr [1:17976] "Petrol" "Petrol" "Petrol" "Petrol" ...
##  $ tax         : num [1:17976] 150 150 150 145 145 145 145 145 145 145 ...
##  $ mpg         : num [1:17976] 57.7 57.7 57.7 40.3 48.7 47.9 50.4 54.3 42.2 61.4 ...
##  $ engineSize  : num [1:17976] 1 1 1 1.5 1 1.6 1 1.2 2 1 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   model = col_character(),
##   ..   year = col_double(),
##   ..   price = col_double(),
##   ..   transmission = col_character(),
##   ..   mileage = col_double(),
##   ..   fuelType = col_character(),
```

```
##   ..   tax = col_double(),
##   ..   mpg = col_double(),
##   ..   engineSize = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
# Summary
summary(ford)
```

```
##     model                year          price       transmission
##  Length:17976       Min.   :1996   Min.   :  495   Length:17976
##  Class :character   1st Qu.:2016   1st Qu.: 8999   Class :character
##  Mode  :character   Median :2017   Median :11291   Mode  :character
##                     Mean   :2017   Mean   :12279
##                     3rd Qu.:2018   3rd Qu.:15298
##                     Max.   :2060   Max.   :54995
##
##     mileage         fuelType              tax             mpg
##  Min.   :     1   Length:17976       Min.   :  0.0   Min.   : 20.8
##  1st Qu.:  9987   Class :character   1st Qu.: 30.0   1st Qu.: 52.3
##  Median : 18242   Mode  :character   Median :145.0   Median : 58.9
##  Mean   : 23362                      Mean   :113.3   Mean   : 57.9
##  3rd Qu.: 31066                      3rd Qu.:145.0   3rd Qu.: 65.7
##  Max.   :177644                      Max.   :580.0   Max.   :201.8
##
##    engineSize
##  Min.   :0.000
##  1st Qu.:1.000
##  Median :1.200
##  Mean   :1.351
##  3rd Qu.:1.500
##  Max.   :5.000
##  NA's   :10
```

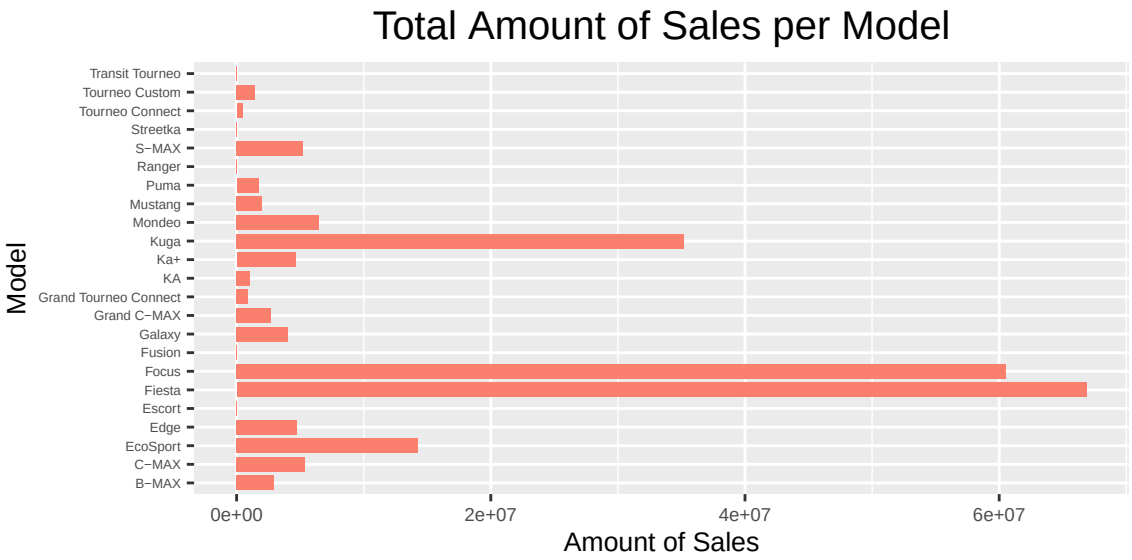
T1-2

- Calculate and visualize the total amount of sales for each model.

```
library(ggplot2)

amount_model = aggregate(ford$price ~ model, sum, data = ford)

ggplot(amount_model) +
  geom_bar(aes(x = model, y = `ford$price`),
           fill = '#FA7F6F',
           stat = 'identity',
           width = 0.8) +
  coord_flip() +
  labs(x = 'Model',
       y = 'Amount of Sales',
       title = "Total Amount of Sales per Model") +
  theme(plot.title = element_text(hjust = 0.5, size = 15),
        axis.text.x = element_text(size = 7),
        axis.text.y = element_text(size = 5),
        axis.title = element_text(size = 10))
```

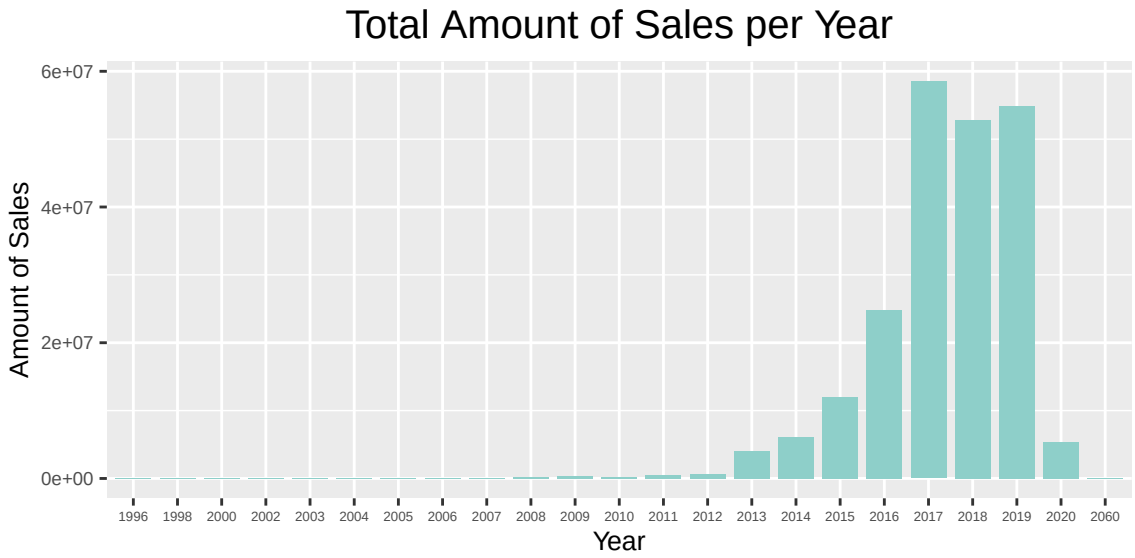


T1-3

- Calculate and visualize the total amount of sales per year.

```
amount_year = aggregate(ford$price ~ year, sum, data = ford)

ggplot(amount_year) +
  geom_bar(aes(x = as.character(year), y = `ford$price`),
           fill = '#8ECFC9',
           stat = 'identity',
           width = 0.8) +
  labs(x = 'Year',
       y = 'Amount of Sales',
       title = "Total Amount of Sales per Year") +
  theme(plot.title = element_text(hjust = 0.5, size = 15),
        axis.text.x = element_text(size = 5),
        axis.text.y = element_text(size = 7),
        axis.title = element_text(size = 10))
```

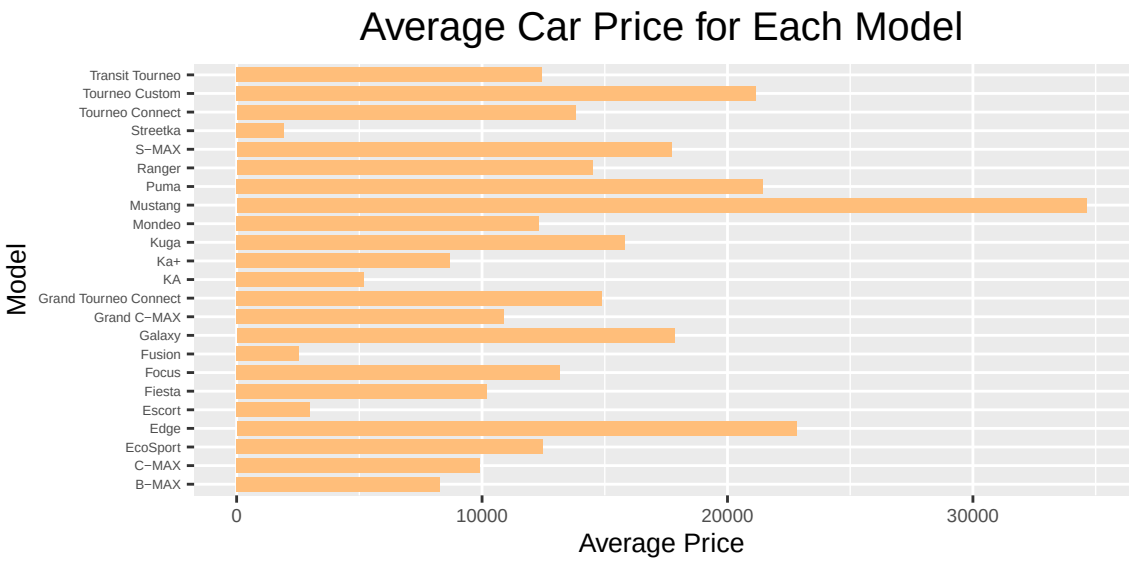


T1-4

- Calculate and visualize the average car price for each model.

```
mean_model = aggregate(ford$price ~ model, mean, data = ford)

ggplot(mean_model) +
  geom_bar(aes(x = model, y = `ford$price`),
           fill = '#FFBE7A',
           stat = 'identity',
           width = 0.8) +
  coord_flip() +
  labs(x = 'Model',
       y = 'Average Price',
       title = "Average Car Price for Each Model") +
  theme(plot.title = element_text(hjust = 0.5, size = 15),
        axis.text.x = element_text(size = 7),
        axis.text.y = element_text(size = 5),
        axis.title = element_text(size = 10))
```



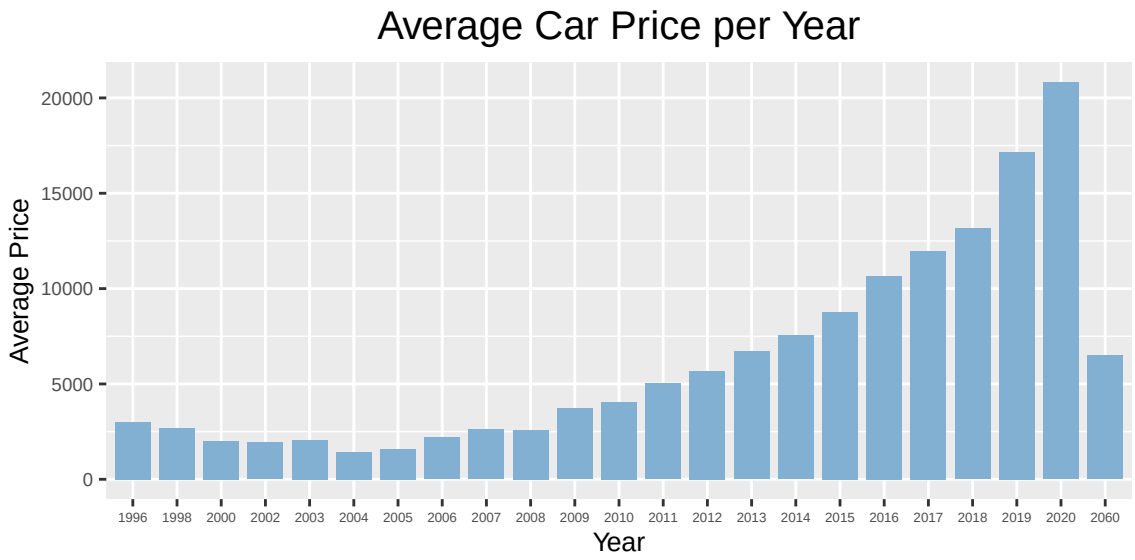
T1-5

- Calculate and visualize the average car price per year.

```
mean_year = aggregate(ford$price ~ year, mean, data = ford)

ggplot(mean_year) +
  geom_bar(aes(x = as.character(year), y = `ford$price`),
           fill = '#82B0D2',
           stat = 'identity',
           width = 0.8) +
  labs(x = 'Year',
       y = 'Average Price',
       title = "Average Car Price per Year") +
  theme(plot.title = element_text(hjust = 0.5, size = 15),
        axis.text.x = element_text(size = 5),
        axis.text.y = element_text(size = 7),
```

```
        axis.title = element_text(size = 10))
```



T1-6

- Analyze data visualization results and summarize your findings.

    1. The total amount of sales of different models varies greatly. The amount of sales of the Model Fiesta and Focus is much higher than that of the other models, exceeding 60 million dollars. Meanwhile, the majority of models have an amount of sales of less than 10 million dollars.
    2. Year records from 1996 to 2060, which might be outliners. The amount of sales increases rapidly with the year and peaks in 2017, then stabilises until a rapid decline in 2020.
    3. The average car price varies considerably between models, from less than $5,000 to nearly $40,000, with a median value close to $15,000.
    4. The average car price was relatively stable between 1996 and 2008 and has been increasing year on year since 2009.

T2

T2-1

1. Check each column for missing values.

```
colSums(is.na(ford))
```

```
##        model         year        price transmission      mileage     fuelType
##            0            0            0            0            0           10
##          tax          mpg   engineSize
##            0            0           10
```

The output shows that there are ten rows of data with a missing value in the engineSize column.

2. As there are only 10 rows with missing values and the total data has 17,976 rows, the number of missing rows is almost negligible in relation to the total data. Therefore, we decided to just drop the rows with missing values.

```
cleaned_miss_ford = na.omit(ford)
```

T2-2

1. Check if there are any duplicate rows in the data.

```
sum(duplicated(cleaned_miss_ford))
```

```
## [1] 162
```

The output shows that there are 150 duplicate rows of data in total.

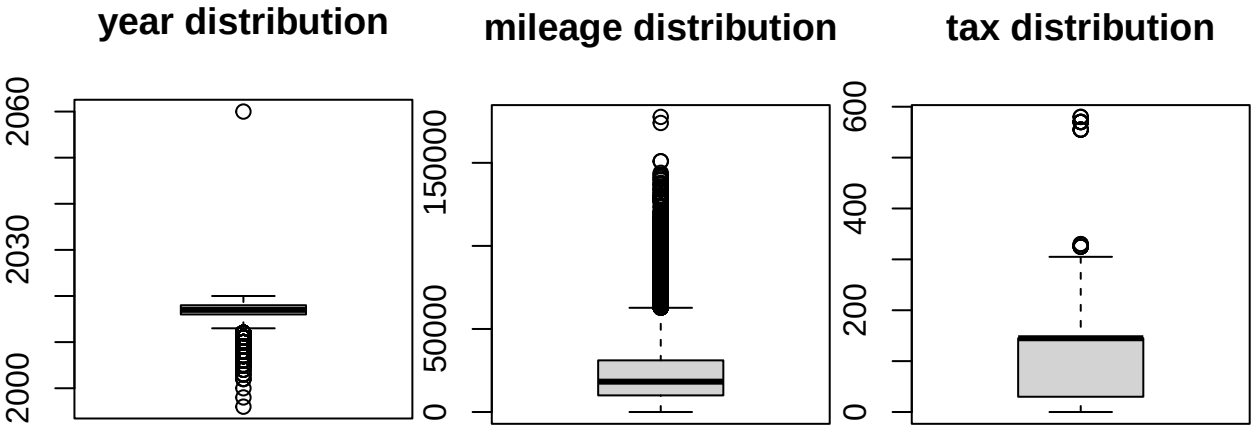
2. Remove the duplicate rows.

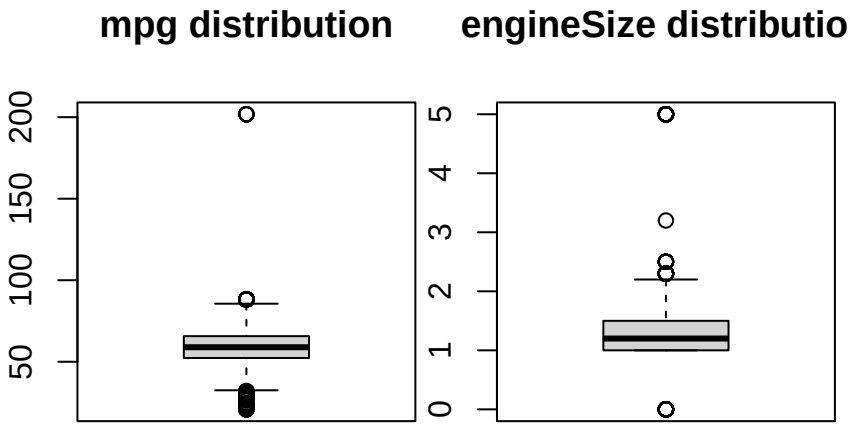
```
cleaned_dup_ford = cleaned_miss_ford[!duplicated(cleaned_miss_ford),]
```

T2-3

1. Plot data distribution of numeric attributes and check for outliers

```
# Check the outliers for the numeric attributes
# Numeric attributes
num_cols <- list('year', 'mileage', 'tax', 'mpg', 'engineSize')
for (i in num_cols){
 # Box-plot the distribution
 boxplot(cleaned_dup_ford[, c(i)], main = paste(i, 'distribution'))
}
```





2. The box-plot shows that these six numeric columns of data all have outliners. Remove the outliners for the columns.

```
# Remove the outliers
# Create a new dataframe
cleaned_out_ford <- cleaned_dup_ford
for (i in num_cols) {
  # Get the outliers using box-plot
  outliers <- boxplot(cleaned_out_ford[[i]], plot=FALSE)$out
  # Remove the outliers in the data frame
  cleaned_out_ford <- cleaned_out_ford[-which(cleaned_out_ford[[i]] %in% outliers),]
}
```

T2-4

- Apply min-max normalization for numric columns.

```
min_max_norm <- function(x) {
  (x - min(x)) / (max(x) - min(x))
}
# New a data frame
cleaned_norm_ford <- cleaned_out_ford
for (i in num_cols) {
  # Apply min-max norm in each column
  cleaned_norm_ford[c(i)] <- as.data.frame(lapply(cleaned_out_ford[c(i)], min_max_norm))
}
```

T2-5

- Encode categorical values.

```
library(mltools)
library(data.table)
# Categorical attributes
cat_cols <- list('model', 'transmission', 'fuelType')
# New a data frame
cleaned_cat_ford <- cleaned_norm_ford
# Apply one hot in the columns
cleaned_cat_ford <- one_hot(as.data.table(cleaned_cat_ford))
```

T2-6

- Store the preprocessed dataset into a new CSV file.

```
# Save the dataset into a CSV file
write.csv(cleaned_cat_ford, file = "cleaned_ford.csv", row.names = F)
```