Lecture

INSTITUTE OF ACTUARIAL & QUANTITATIVE STUDIES

Mrs. Indrani Sen (MCA, Mphil Computer Science)

Class: SYBSc

Subject : Data Managemnt

Subject Code: Statistical modelling in R

Chapter: Unit 1 **Chapter2 chapter Name:** Data visualisation

Importance of shape

One application is **testing for normality**:

many statistics inferences require that a distribution be normal or nearly normal.

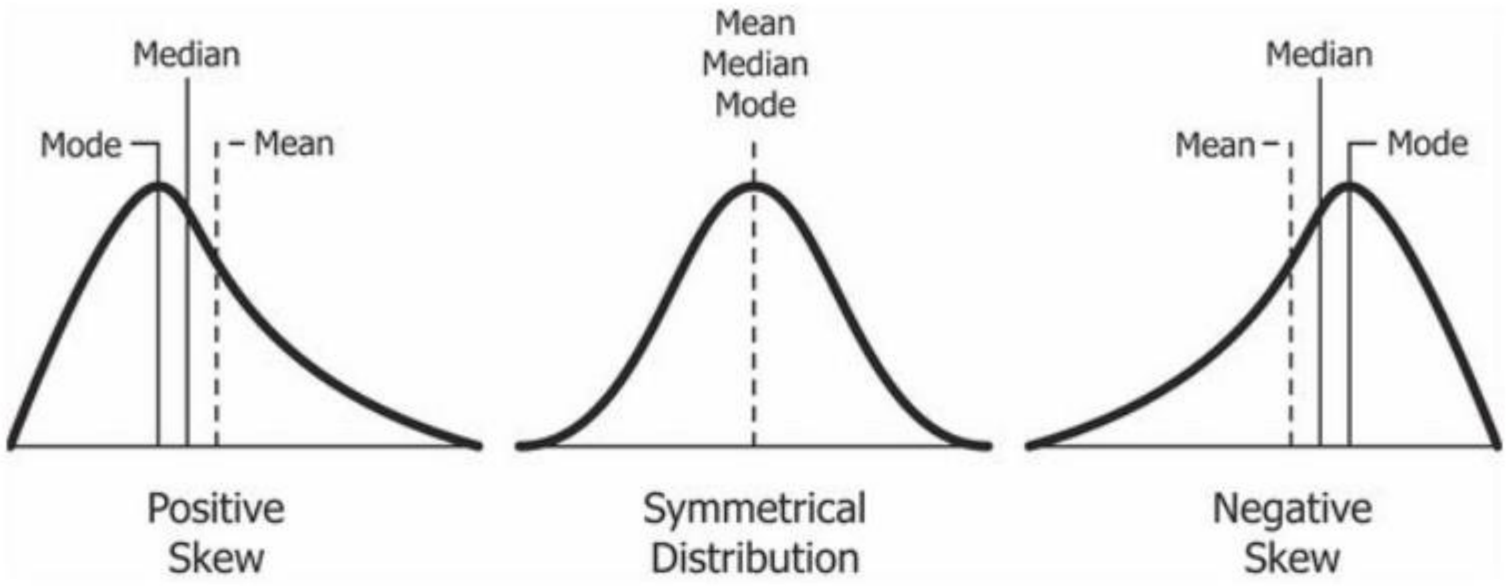
A normal distribution has skewness and excess kurtosis of 0,

so if your distribution is close to those values then it is probably close to normal.

Skewness

- It is the *degree of distortion* from the symmetrical bell curve or the normal distribution.
- It measures the lack of symmetry in data distribution.
  If it's **unimodal** (has just one peak), like most data sets, the next thing you notice is whether it's **symmetric or skewed** to one side.
- If the bulk of the data is at the left and the right tail is longer, we say that the distribution is **skewed right or positively skewed**;
- if the peak is toward the right and the left tail is longer, we say that the distribution is **skewed left or negatively skewed**.

Dist plot


Median
Mode
Mean
Positive Skew
Mean
Median
Mode
Symmetrical Distribution
Median
Mean
Mode
Negative Skew

Skewness and symmetry

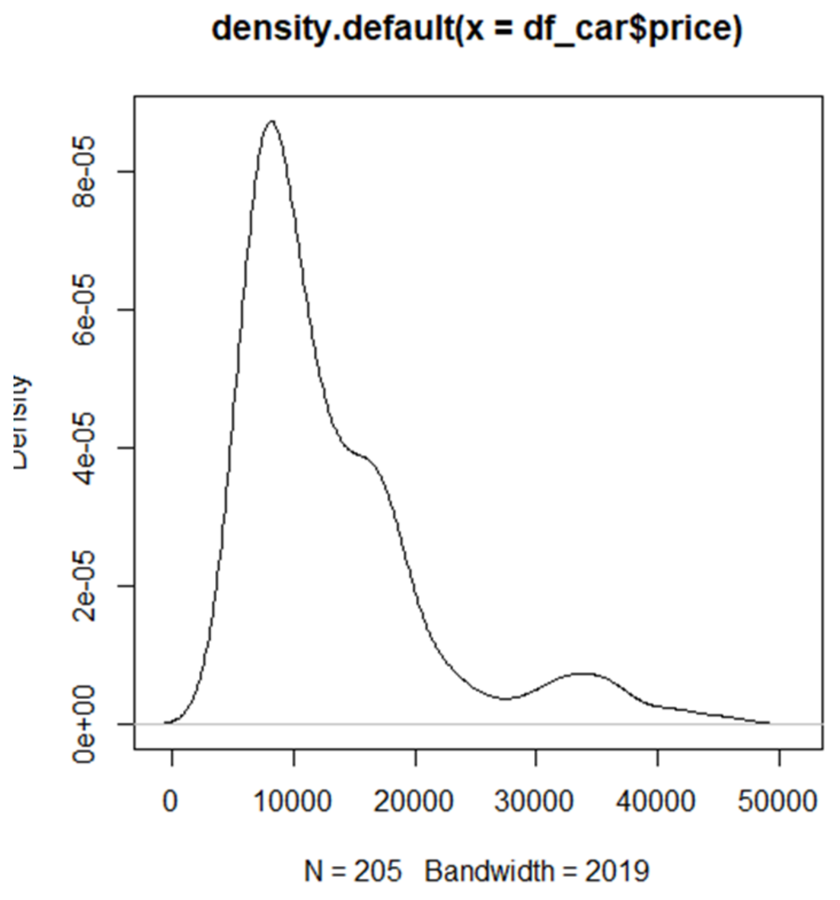
- If skewness is 0, the data are perfectly symmetrical, although it is quite unlikely for real-world data.
- If skewness is less than -1 or greater than 1, the distribution is highly skewed.
- If skewness is between -1 and -0.5 or between 0.5 and 1, the distribution is moderately skewed.
- If skewness is between -0.5 and 0.5, the distribution is approximately symmetric.

Calculating skewness

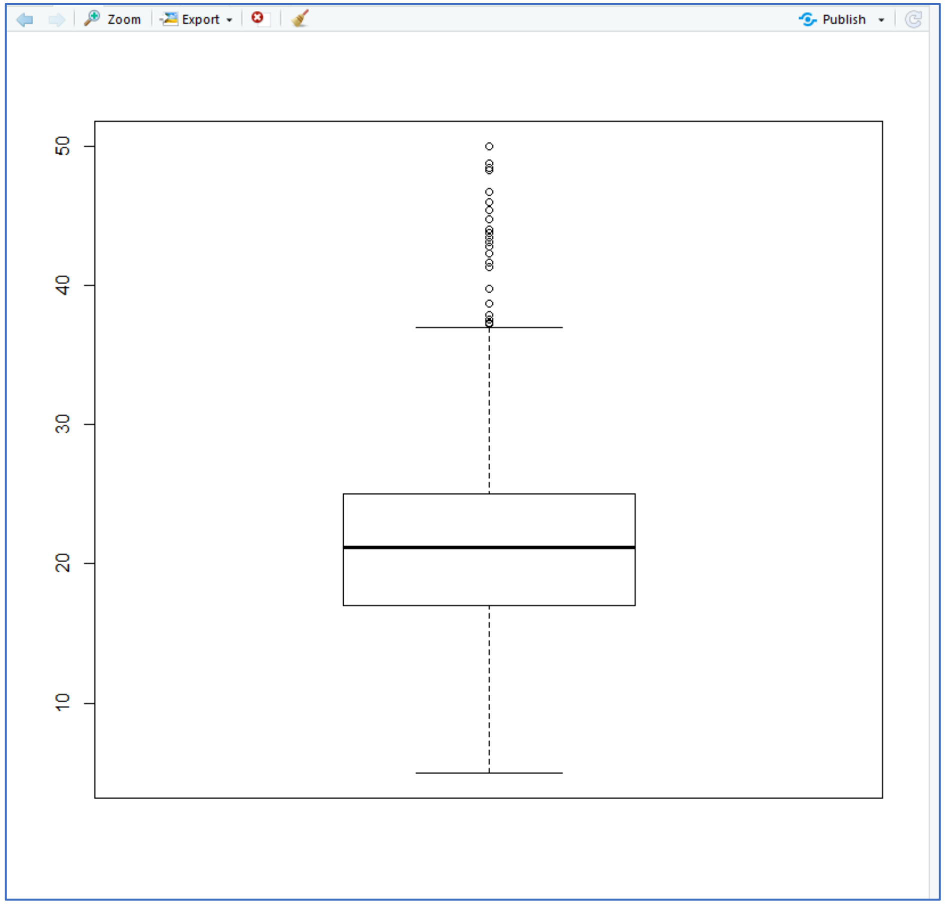
```
> library(moments)

> skewness(df_car$price)

[1] 1.764644

> plot(d)
```

Calculating skewness

```
> density(df_car$price)
x                y
 Min.   : -937.9   Min.   :1.088e-08
 1st Qu.:12160.6   1st Qu.:2.415e-06
 Median :25259.0   Median :6.402e-06
 Mean   :25259.0   Mean   :1.907e-05
 3rd Qu.:38357.4   3rd Qu.:3.378e-05
 Max.   :51455.9   Max.   :8.727e-05
> d=density(df_car$price)
> plot(d)
```



Outlier analysis



Kurtosis

- Kurtosis is all about the tails of the distribution — not the peakedness or flatness.
- It is used to describe the extreme values in one versus the other tail.
- It is actually the measure of outliers present in the distribution.

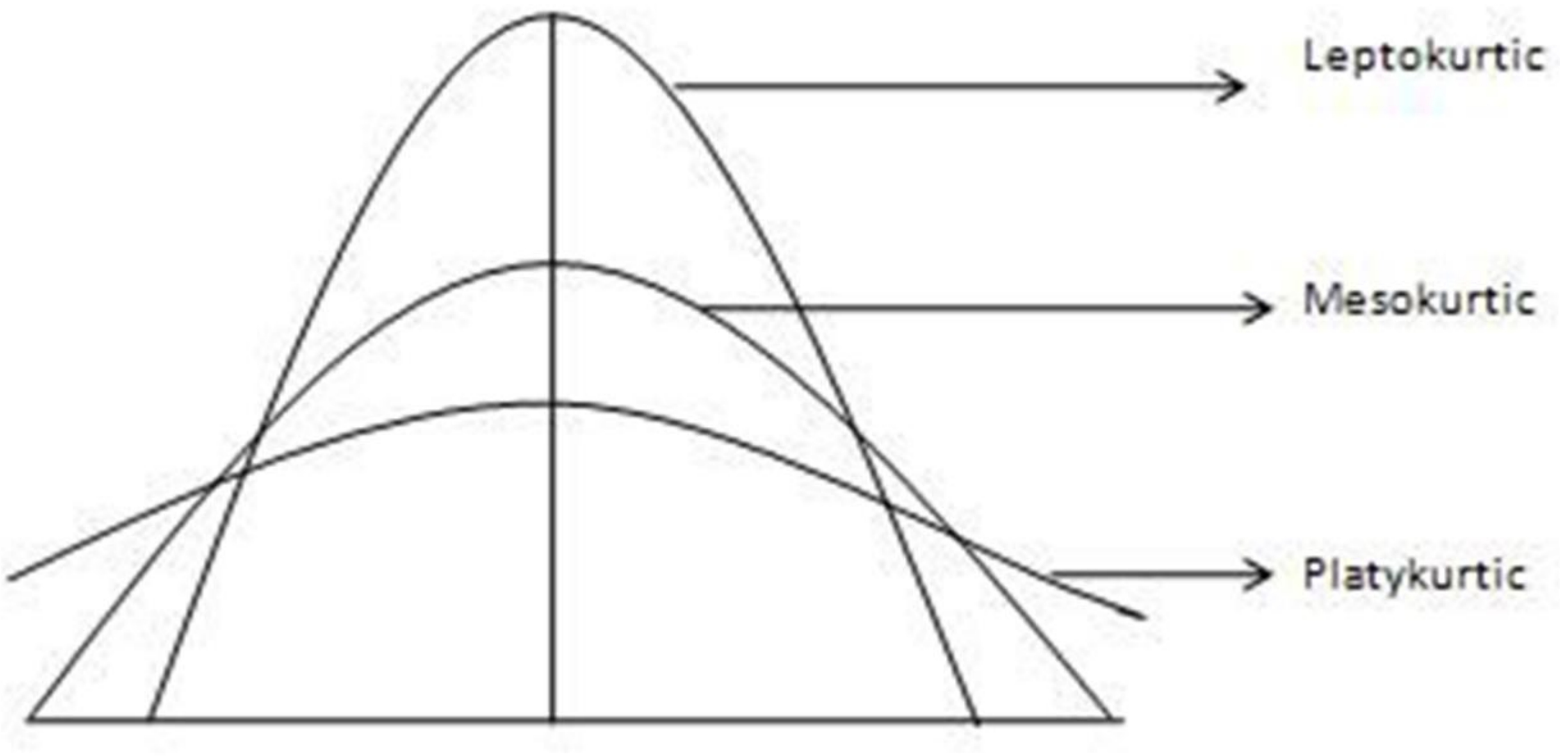
High kurtosis

- **High kurtosis** in a data set is an indicator that data has heavy tails or outliers.
- If there is a high kurtosis, then, we need to investigate why do we have so many outliers.
- It indicates a lot of things, maybe wrong data entry or other things. Investigate!

Low kurtosis

- Low kurtosis in a data set is an indicator that data has light tails or lack of outliers.
- If we get low kurtosis(too good to be true), then also we need to investigate the dataset

Types of kurtosis



Mesokurtic

- This distribution has kurtosis statistic similar to that of the normal distribution.
- It means that the extreme values of the distribution are similar to that of a normal distribution characteristic.
- This definition is used so that the standard normal distribution has a kurtosis of 3.

Leptokurtic

1. **Leptokurtic (*Kurtosis > 3*):**
2. Distribution is longer, tails are fatter.
3. Peak is higher and sharper than Mesokurtic,
4. which means that data are heavy-tailed or profusion of outliers.

Platykurtic

Platykurtic: (*Kurtosis* < *3*):
Distribution is shorter, tails are thinner than the normal distribution.
The peak is lower and broader than Mesokurtic,
which means that data are light-tailed or lack of outliers.

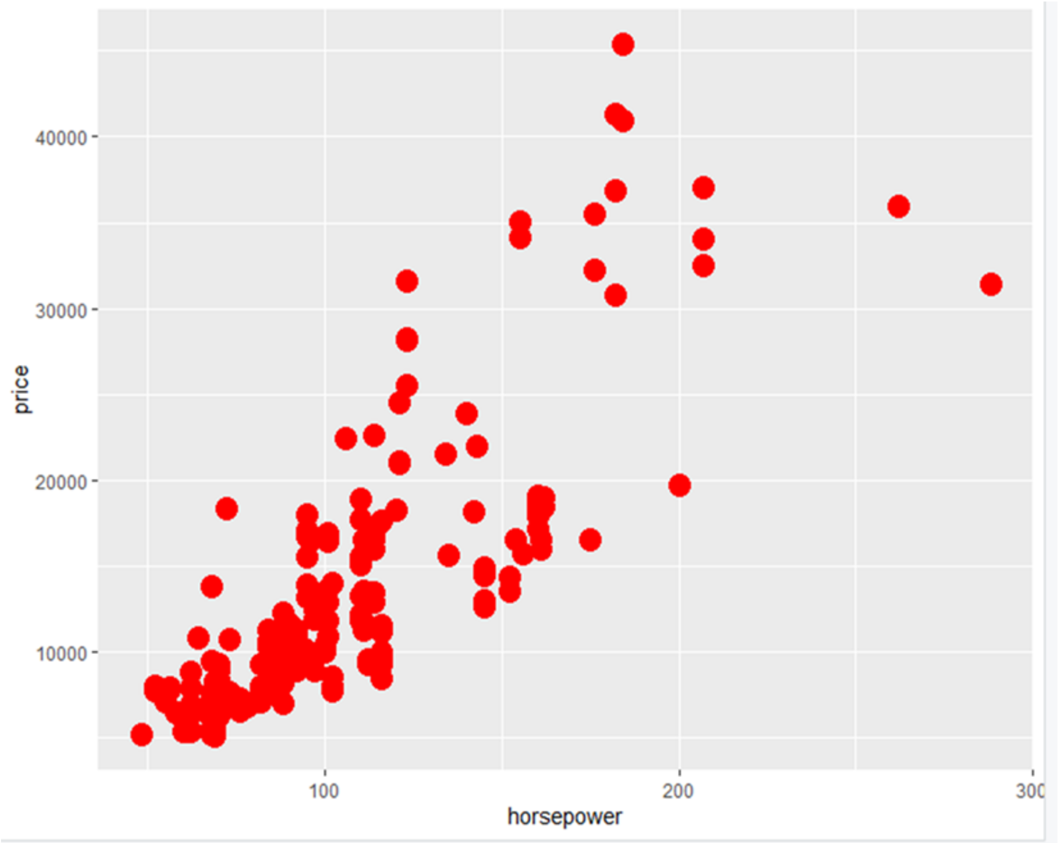
The reason for this is because the extreme values are less than that of the normal distribution.

Kurtosis of car price

```
> kurtosis(df_car$price)
[1] 5.948598

#kurtosis>3
```

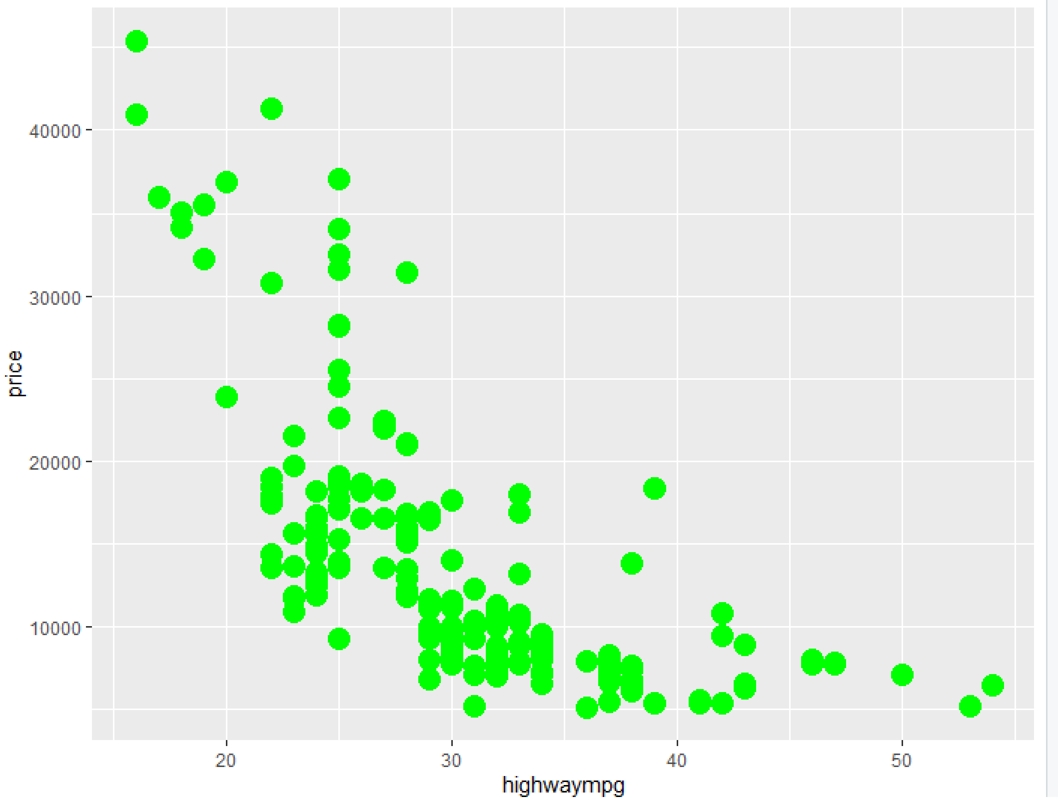
Scatterplot (To display price wrt horsepower)

- aes() function is used to specify the X and Y axes.
- That's because, any information that is part of the source dataframe has to be specified inside the aes() function.
- ggplot(df_car,aes(x=horsepower,
- y=price))+geom_point(size=5,col="red")



Scatterplot (To display price wrt highwaympg)

- aes() function is used to specify the X and Y axes.
- That’s because, any information that is part of the source
- dataframe has to be specified inside the aes() function.
- ggplot(df_car,aes(x=highwaympg,y =price))+geom_point(size=5,col="g reen")

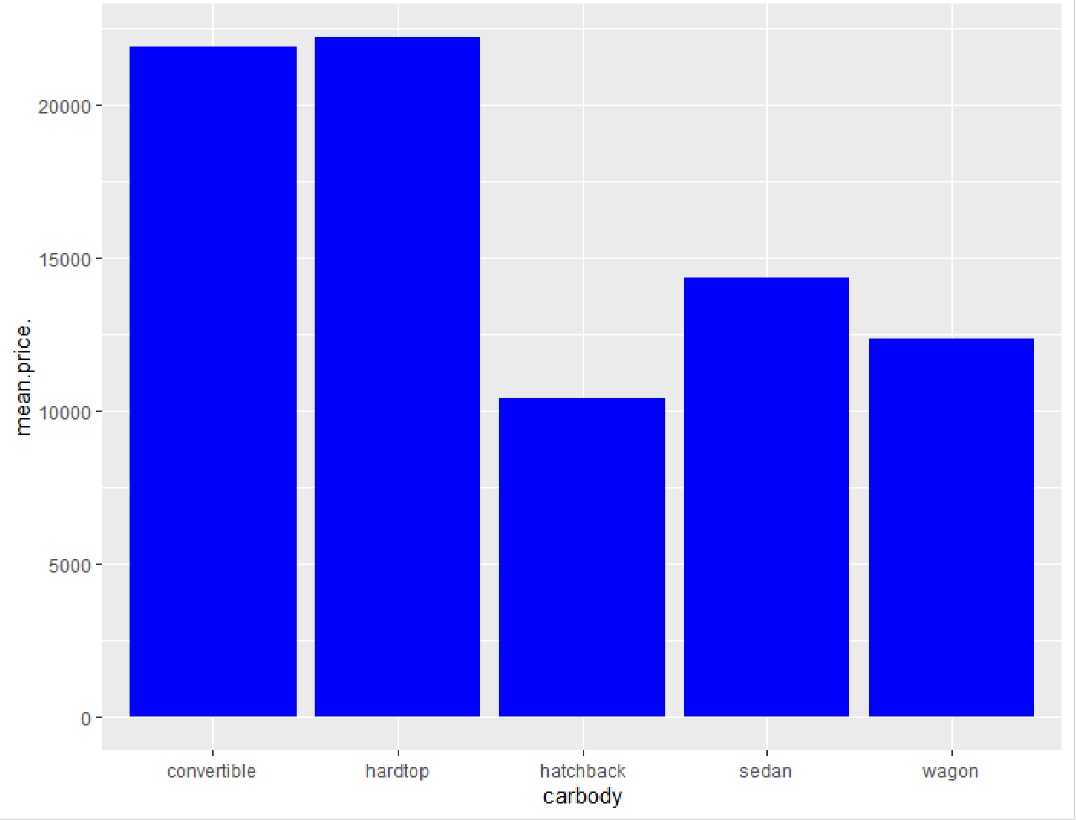


geom_bar()

- This is the most basic barplot you can build using the ggplot2 package. It follows those steps:
- always start by calling the ggplot() function.
- then specify the data object. It has to be a data frame. And it needs one numeric and one categorical variable.
- then come thes aesthetics, set in the aes() function: set the categoric variable for the X axis, use the numeric for the Y axis
- finally call geom_bar(). You have to specify stat="identity" for this kind of dataset.

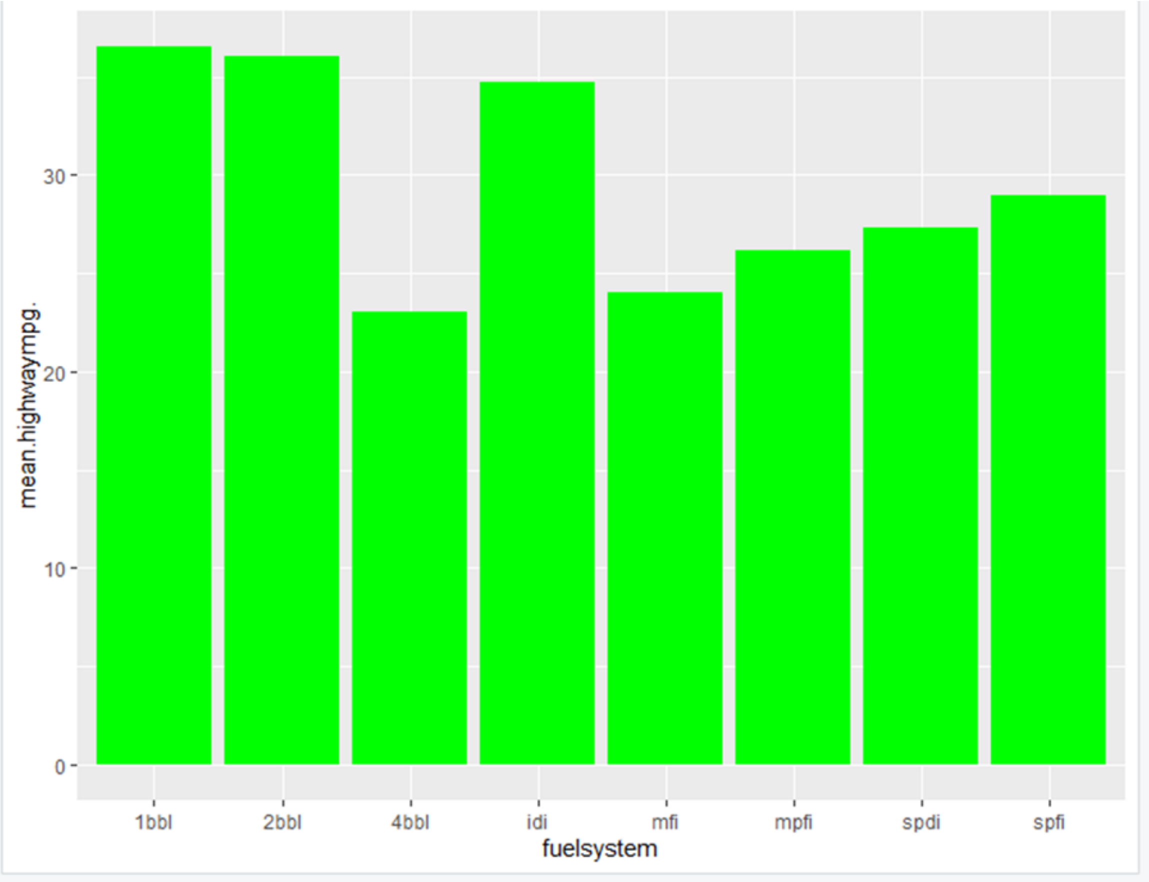
Bar plot of average price based on carbody

```
df_car_price=data.frame( df_car %>%
+ group_by(carbody) %>%
+ summarise(mean(price))
+ )
> df_car_price
      carbody mean.price.
1 convertible    21890.50
2     hardtop    22208.50
3   hatchback    10376.65
4       sedan    14344.27
5       wagon    12371.96
>ggplot(df_car_price,aes(carbody,mean.price.))+geom_bar(stat="identity",fill="blue")
```



Bar plot of average mileage based on fuelsystem

```
> df_car_mpg=
+ data.frame(df_car %>%
group_by(fuelsystem)
%>% summarise(mean(highwaympg)))
> df_car_mpg
  fuelsystem mean.highwaympg.
1       1bbl        36.54545
2       2bbl        36.01515
3       4bbl        23.00000
4        idi        34.75000
5        mfi        24.00000
6       mpfi        26.19149
7       spdi        27.33333
8   spfi        29.00000
9   ggplot(df_car_mpg,aes(fuelsystem,mean.highaympg.))+geom_bar(stat="identity",fill="green"
    )
```



Data Visualization

Data visualization in an effective method of data exploration.

Understanding the distribution of **prices** will help us to judge our predicted prices in relation to its accuracy.

```
#Data visualization
par(mfrow=c(1,2))
plot(density(car$price),main="Car Price Distribution plot")
boxplot(car$price, main="Car Price spread")
par(mfrow=c(1,1))
summary(car$price)
quantile(car$price)
```

Data Visualization

Data visualization in an effective method of data exploration.

Understanding the distribution of **prices** will help us to judge our predicted prices in relation to its accuracy.

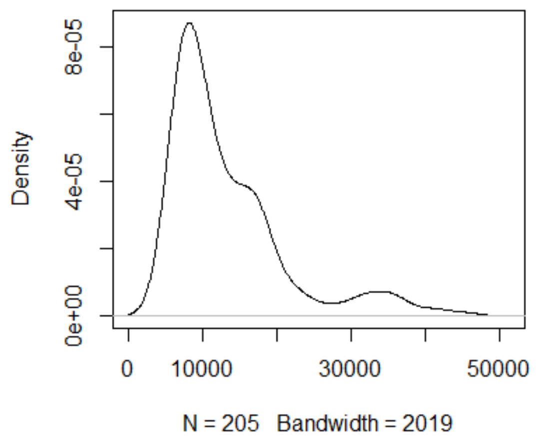
Data Visualization

Inference :

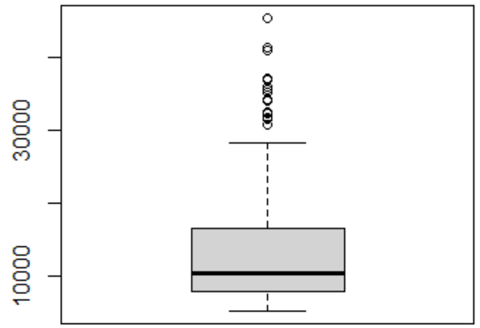
•The plot seemed to be right-skewed, meaning that the most prices in the dataset are low (Below 15,000).

•The data points are far spread out from the mean, which indicates a high variance in the car prices.(85% of the prices are below 18,500, whereas the remaining 15% are between 18,500 and 45,400.)

Car Price Distribution plot



Car Price spread

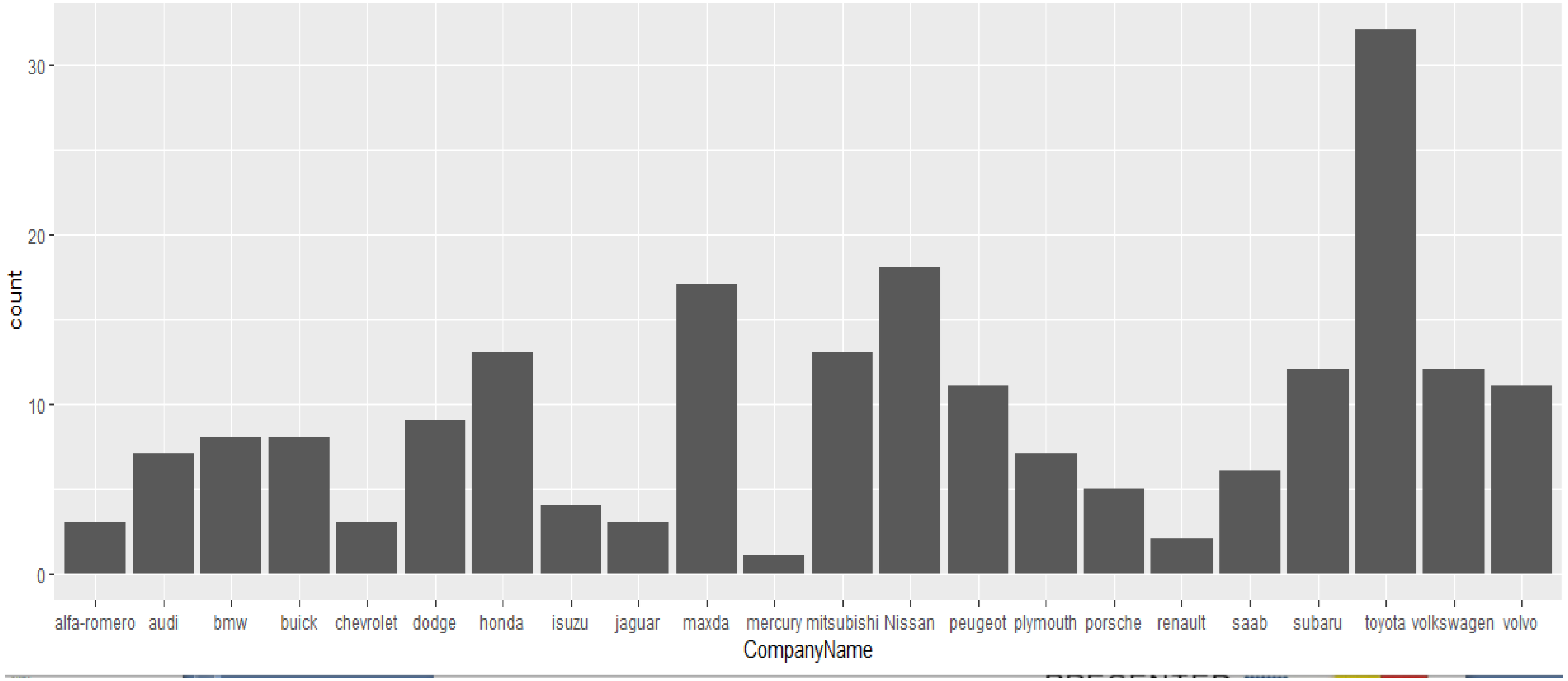


Data Summary

```
> summary(car$price)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   5118    7788   10295   13277   16503   45400 
> quantile(car$price)
   0%   25%   50%   75%  100% 
 5118  7788 10295 16503 45400 
```

Bar graph of CompanyName

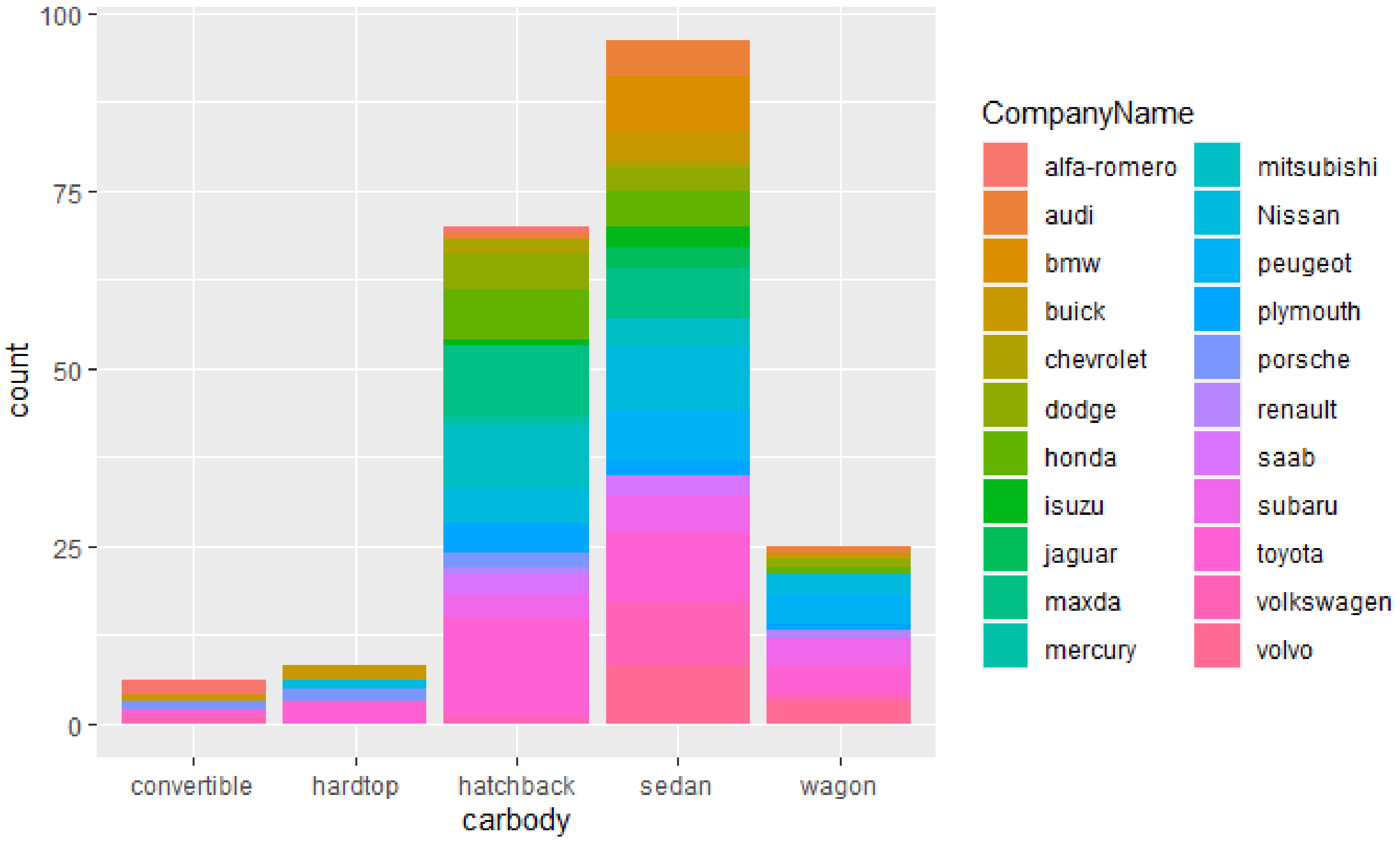
```
library(ggplot2)
car$CompanyName=as.factor(car$CompanyName)
class(car$CompanyName)
p1=ggplot(car,aes(x=CompanyName))
p1+geom_bar()
```



Inference:
Toyota seems to be a favored company

Bar graph of CompanyName

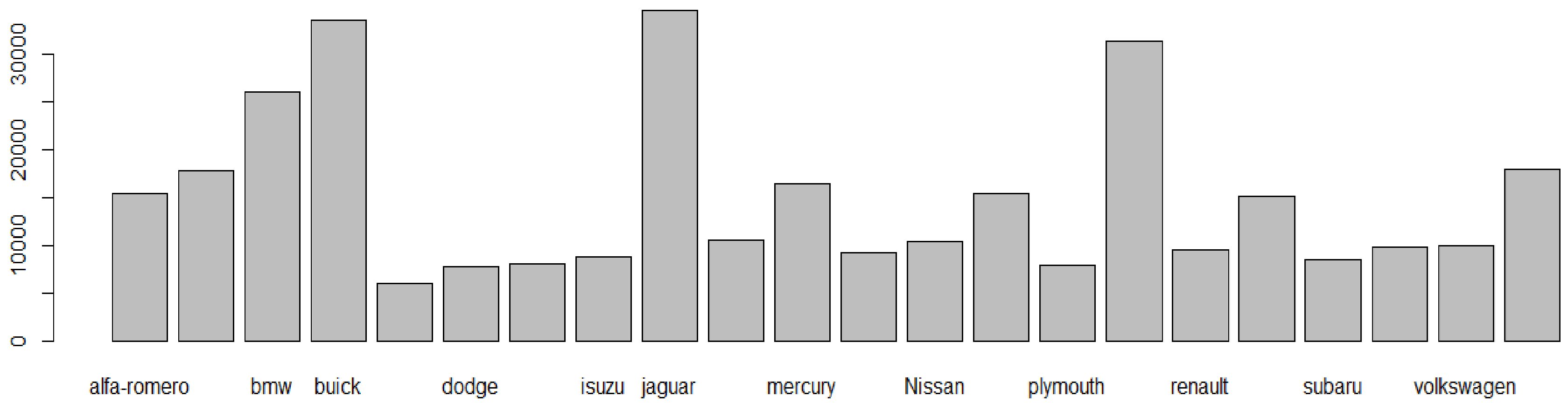
```
pl2=ggplot(car,aes(x=carbody))
pl2+geom_bar(aes(fill=CompanyName))
```



Inference :
Sedan is the top car type preferred

Barplot of companyName with average price

```
> df=car%>%group_by(CompanyName)%>%summarise(mean=mean(price)) #using pipe operator
`summarise()` ungrouping output (override with `.groups` argument)
> barplot(df$mean,names.arg = df$CompanyName)
```



Inference:

Jaguar and Buick seem to have highest average price.

Barplot of companyName with average price

Create barplot for

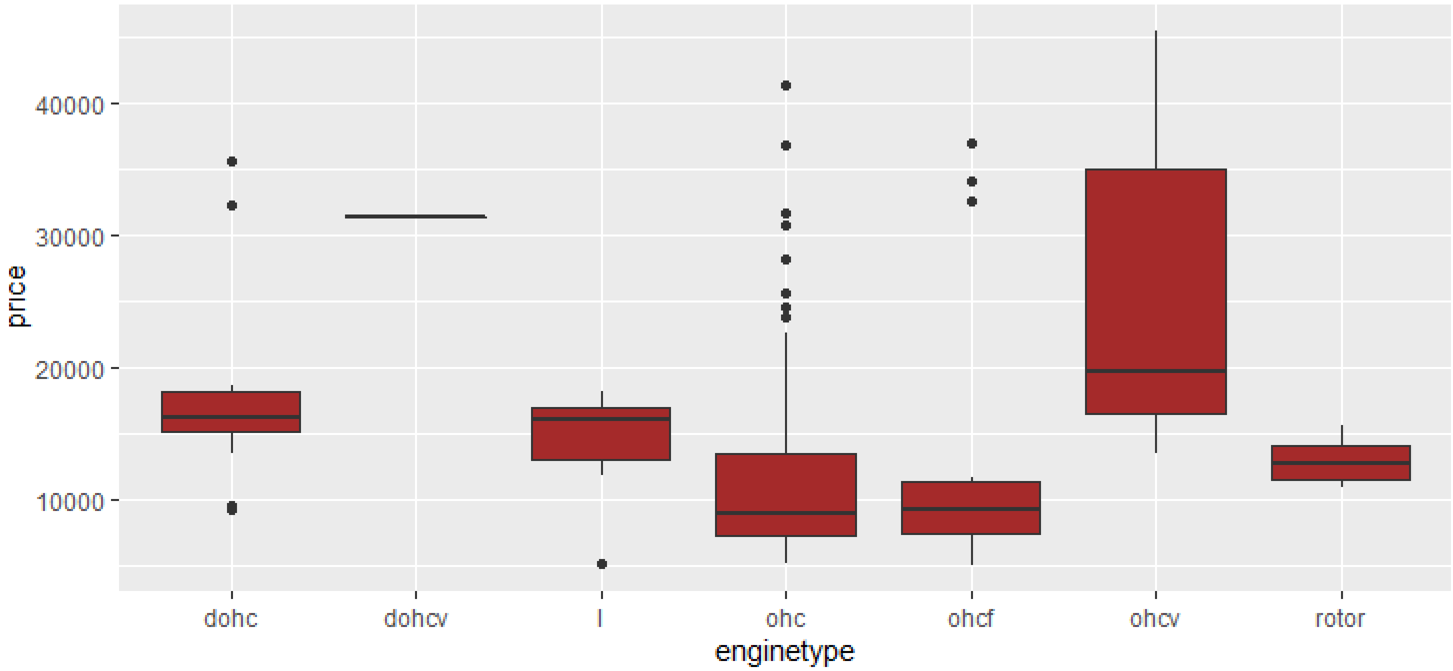
- fueltype with average price
- enginetype with average price

Inference:

- ➢diesel has higher average price than gas.
- ➢hardtop and convertible have higher average price

Boxplot with enginetype and price

```
#enginetype versus price
pl4=ggplot(car,aes(enginetype,price))
pl4+geom_boxplot(fill="brown")
par(mfrow=c(1,1))
```



Inference:

➢'ohc' seems to have the highest price range.
There seems to be outliers in this data.

➢We discuss this the business team
Business team has asked us to use the field irrespective of the outliers because it has important information.

So we proceed.

Inference on univariate analysis

- **Use boxplot to understand the impact of other variables on price**

▪aspiration
▪enginelocation
▪cylindernumber
▪fuelsystem
▪drivewheel

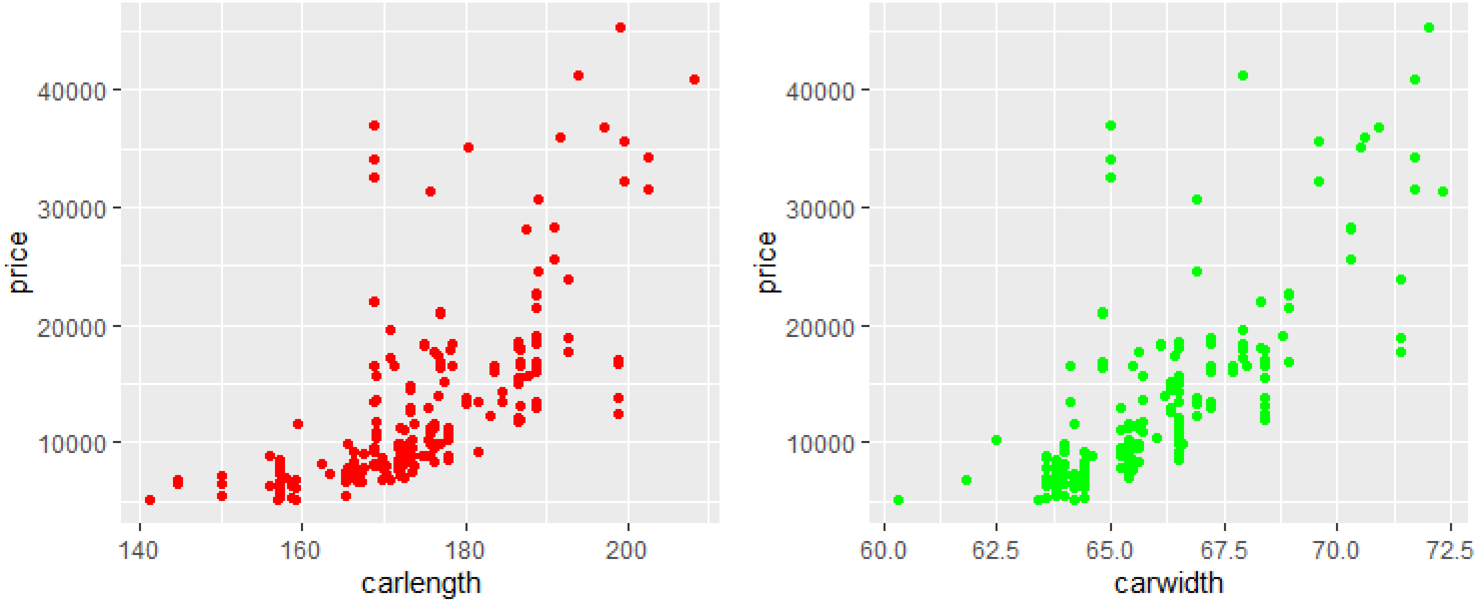
- **Inference:**

➢It seems aspiration with turbo have higher price range than the std(though it has some high values outside the whiskers.)
➢Very few datapoints for enginelocation categories to make an inference.
➢Eight cylinders have the highest price range.
➢mpfi and idi having the highest price range. But there are few data for other categories to derive any meaningful inference
➢Most high ranged cars seemed to prefer rwd drivewheel.

Scatter plot of car width & price, car length & price

Scatterplot helps to check correlation.

```
library(ggplot2)
library(gridExtra)
gg1=ggplot(car,aes(carlength,price))+geom_point(color="red")
gg2=ggplot(car,aes(carwidth,price))+geom_point(color="green")
grid.arrange(gg1,gg2,ncol=2)
```

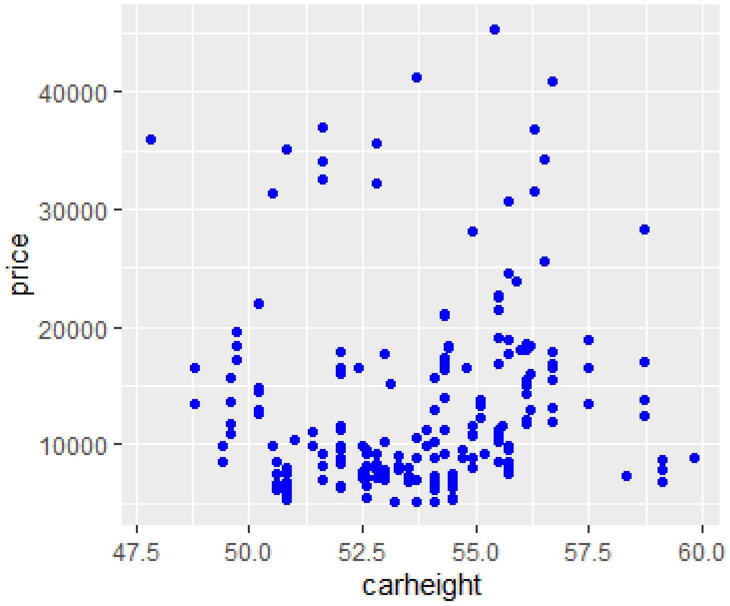


Inference:

Carlength & carwidth have positive correlation with price.

Scatter plot of car height & price, curb weight & price

```
gg3=ggplot(car,aes(carheight,price))+
  geom_point(color="blue")
gg4=ggplot(car,aes(curbweight,price))+
  geom_point(color="red")
grid.arrange(gg3,gg4,ncol=2)
```



Inference:

➢**curbweight** seems to have a positive correlation with price.

➢**carheight** doesn't show any significant trend with price.

Inference on univariate analysis

- Create scatterplot of following variables with price
- Enginesize
- Boreratio
- Stroke
- Compressionratio
- Horsepower
- Peakrpm
- Wheelbase
- Citympg
- Highwaympg

- **Inference:**
- enginesize, boreratio, horsepower, wheelbase - seem to have a significant positive
  - correlation with price.
- citympg, highwaympg - seem to have a significant negative correlation with price.

Feature Engineering

Feature engineering is the process of using domain knowledge to extract features
from raw data via data mining techniques.

The **fuel economy** of an automobile relates distance traveled by a vehicle and the amount of fuel consumed. We have two variables which show distance travelled traveled with respect to fuel consumed : citympg and highwaympg. Both have strong correlation with price, we can combine them for our analysis.

fueleconomy= 0.55*citympg +0.45*highwaympg

Discussion with business team resulted in this combination weights of **citympg** & **highwaympg**. Correlation values of these fields with price could have been used as weights too.

Feature Engineering

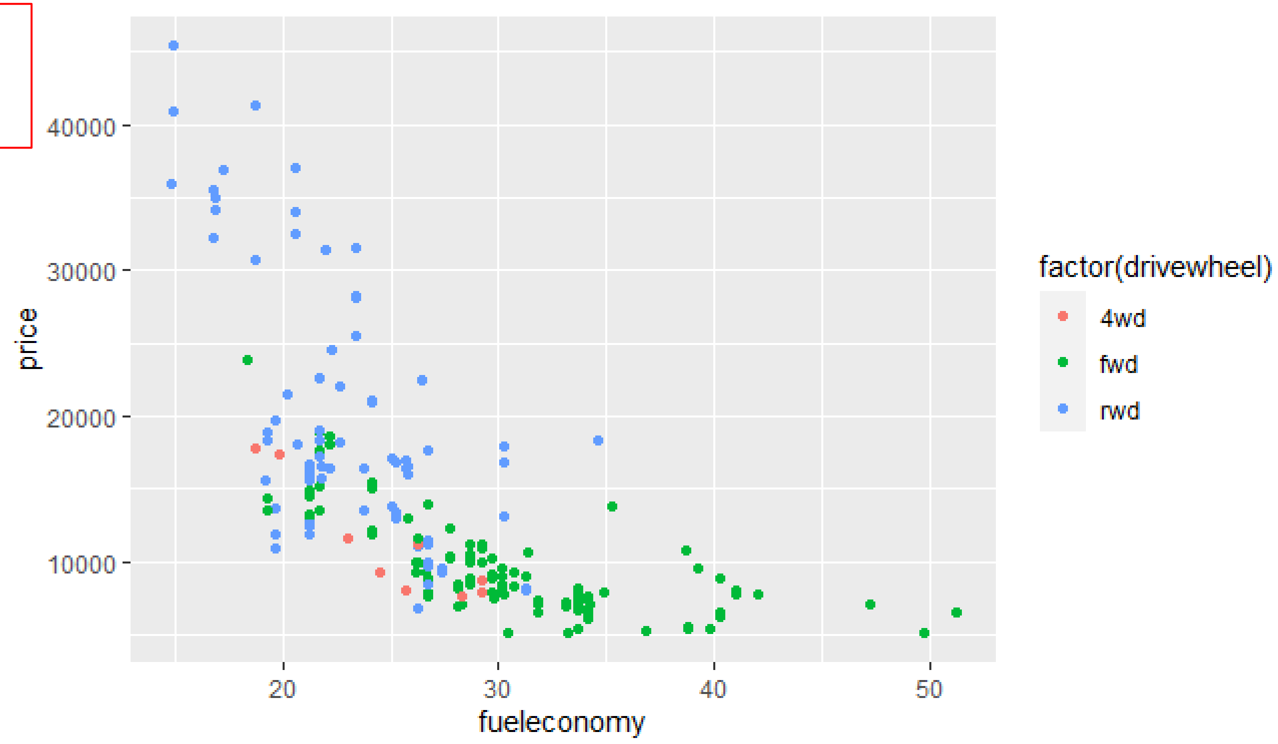
```
#feature engineering
car$fueleconomy=(0.55*car$citympg)+(0.45*car$highwaympg)
head(car$fueleconomy)
```

Output:

```
> head(car$fueleconomy)
[1] 23.70 23.70 22.15 26.70 19.80 21.70
```

Scatter plot to confirm correlation between fueleconomy and price

```
p16=ggplot(car,aes(x=fueleconomy,y=price))
p16+geom_point(aes(color=factor(drivewheel)))
```



Inference:

➢fueleconomy has an obvious negative correlation with price and is significant.

Significant variables after visual analysis

After analyzing the scatter plots and other charts, we identify fields that influence the price of car.

Engine Type Drivewheel Engine Size Fuel Economy

Fuel type Curbweight Boreratio carbody

Aspiration Car Length Horse Power fueltype

Cylinder Number Car width Wheel base

Let us create a data frame containing only variables required for price prediction. We have stored it in the variable named ‘use’.

```
fields=c('wheelbase', 'curbweight', 'enginesize', 'boreratio',
         'horsepower','fueleconomy','carlength','carwidth','price','enginetype',
        'fueltype','carbody','aspiration','cylindernumber','drivewheel' )
use=car[,fields]
head(use)
```

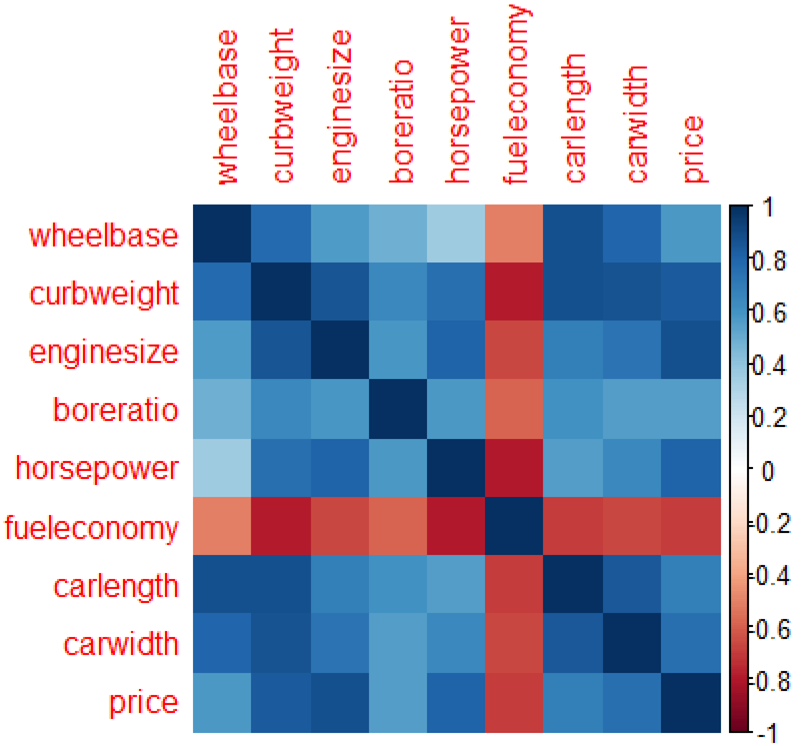
```
> head(use)
# A tibble: 6 x 15
  wheelbase curbweight enginesize boreratio horsepower fueleconomy carlength carwidth
      <dbl>      <dbl>      <dbl>     <dbl>      <dbl>       <dbl>     <dbl>    <dbl>
1      88.6       2548        130      3.47        111        23.7      169.     64.1
2      88.6       2548        130      3.47        111        23.7      169.     64.1
3      94.5       2823        152      2.68        154        22.2      171.     65.5
4      99.8       2337        109      3.19        102        26.7      177.     66.2
5      99.4       2824        136      3.19        115        19.8      177.     66.4
6      99.8       2507        136      3.19        110        21.7      177.     66.3
# ... with 7 more variables: price <dbl>, enginetype <chr>, fueltype <chr>,
#   carbody <chr>, aspiration <chr>, cylindernumber <chr>, drivewheel <chr>
```

Correlation Analysis

```
library(corrplot)
#Understanding the correlation between numeric fields using visualization.
num.cols=sapply(use,is.numeric)
cor.data=cor(use[,num.cols])
round(cor.data,4)
corrplot(cor.data,method="color")
```

Outp

	price
wheelbase	0.5778
curbweight	0.8353
enginesize	0.8741
boreratio	0.5532
horsepower	0.8081
fueleconomy	-0.6962
carlength	0.6829
carwidth	0.7593
price	1.0000



Correlation is above |0.5| for all variables with price indicating their significant impact

Inference:

The strength of correlation between variables is indicated using color scales. Price has positive correlation with all variables except fueleconomy. It has maximum correlation with enginesize and least with boreratio.

Date & Time functions

```
> today=Sys.Date()
> class(today)
[1] "Date"
> today
[1] "2020-08-02"
```

```
> c="1990-01-01"
> class(c)
[1] "character"
> my.date=as.Date(c)
> class(my.date)
[1] "Date"
```

This can be converted easily when the input is in standard date format.

```
> as.Date("Nov-03-1990")
Error in charToDate(x) :
  character string is not in a standard unambiguous format
```

“format” argument can be used to handle this.

Date & Time functions

```
> as.Date("Nov-03-1990",format="%b-%d-%Y")
[1] "1990-11-03"
```

R uses POSIXct or POSIXlt to store time information. It stands for Portable Operating System Interface. It is mostly used in time series analysis.

```
> as.POSIXct("11:31:09",format="%H:%M:%S")
[1] "2020-08-02 11:31:09 IST"
```

It adds the date and timestamp "IST" itself. strptime() is more commonly used.

```
> strptime("12:00:09",format="%H:%M:%S")
[1] "2020-08-02 12:00:09 IST"
```

Notations for format argument

%d – day of the month(decimal no)
%m – Month(decimal no)
%b – Month(abbreviated)
%B – Month(fulll name)
%y – year(2 digit)
%Y – year(4 digit)

You can check the value for format argument by calling help on strptime().