Prashanth.S 19MID0020

Bivariate Analysis

Importing the libraries

```r
library(ggplot2)
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(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
library(tidyr)
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```r
library(RColorBrewer)
library(ggrepel)
library(ggthemes)
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```r
library(plotly)
```

```
##
## Attaching package: 'plotly'
```

```
## The following object is masked from 'package:ggplot2':
##
##     last_plot
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following object is masked from 'package:graphics':
##
##     layout
```

Importing the data-sets

```r
df = read.csv('pokemon.csv')
head(df)
```

	Name <chr>	Type.1 <chr>	Type.2 <chr>	Total <int>	HP <int>	Attack <int>	Defense <int>	Sp..Atk <int>	Sp..Def <int>
1	Bulbasaur	Grass	Poison	318	45	49	49	65	65
2	Ivysaur	Grass	Poison	405	60	62	63	80	80
3	Venusaur	Grass	Poison	525	80	82	83	100	100
4	VenusaurMega Venusaur	Grass	Poison	625	80	100	123	122	120
5	Charmander	Fire		309	39	52	43	60	50
6	Charmeleon	Fire		405	58	64	58	80	65

6 rows | 1-10 of 13 columns

PreProcessing

```r
cat("Number of instances  : ",nrow(df))
```

```
## Number of instances  :  800
```

```r
cat("\nNumber of attributes : ",ncol(df))
```

```
##
## Number of attributes :  12
```

```r
str(df)
```

```
## 'data.frame':    800 obs. of  12 variables:
##  $ Name      : chr  "Bulbasaur" "Ivysaur" "Venusaur" "VenusaurMega Venusaur" ...
##  $ Type.1    : chr  "Grass" "Grass" "Grass" "Grass" ...
##  $ Type.2    : chr  "Poison" "Poison" "Poison" "Poison" ...
##  $ Total     : int  318 405 525 625 309 405 534 634 634 314 ...
##  $ HP        : int  45 60 80 80 39 58 78 78 78 44 ...
##  $ Attack    : int  49 62 82 100 52 64 84 130 104 48 ...
##  $ Defense   : int  49 63 83 123 43 58 78 111 78 65 ...
##  $ Sp..Atk   : int  65 80 100 122 60 80 109 130 159 50 ...
##  $ Sp..Def   : int  65 80 100 120 50 65 85 85 115 64 ...
##  $ Speed     : int  45 60 80 80 65 80 100 100 100 43 ...
##  $ Generation: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Legendary : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

```r
summary(df)
```

```
##      Name              Type.1             Type.2              Total
##  Length:800         Length:800         Length:800         Min.   :180.0
##  Class :character   Class :character   Class :character   1st Qu.:330.0
##  Mode  :character   Mode  :character   Mode  :character   Median :450.0
##                                                           Mean   :435.1
##                                                           3rd Qu.:515.0
##                                                           Max.   :780.0
##        HP             Attack       Defense          Sp..Atk
##  Min.   :  1.00   Min.   :  5   Min.   :  5.00   Min.   : 10.00
##  1st Qu.: 50.00   1st Qu.: 55   1st Qu.: 50.00   1st Qu.: 49.75
##  Median : 65.00   Median : 75   Median : 70.00   Median : 65.00
##  Mean   : 69.26   Mean   : 79   Mean   : 73.84   Mean   : 72.82
##  3rd Qu.: 80.00   3rd Qu.:100   3rd Qu.: 90.00   3rd Qu.: 95.00
##  Max.   :255.00   Max.   :190   Max.   :230.00   Max.   :194.00
##     Sp..Def          Speed          Generation    Legendary
##  Min.   : 20.0   Min.   :  5.00   Min.   :1.000   Mode :logical
##  1st Qu.: 50.0   1st Qu.: 45.00   1st Qu.:2.000   FALSE:735
##  Median : 70.0   Median : 65.00   Median :3.000   TRUE :65
##  Mean   : 71.9   Mean   : 68.28   Mean   :3.324
##  3rd Qu.: 90.0   3rd Qu.: 90.00   3rd Qu.:5.000
##  Max.   :230.0   Max.   :180.00   Max.   :6.000
```

```r
for ( col in 1:ncol(df)) {
  cat(colnames(df)[col])
  cat(" --> ")
  print(sum(is.na(df[,col])))
}
```

```
## Name --> [1] 0
## Type.1 --> [1] 0
## Type.2 --> [1] 0
## Total --> [1] 0
## HP --> [1] 0
## Attack --> [1] 0
## Defense --> [1] 0
## Sp..Atk --> [1] 0
## Sp..Def --> [1] 0
## Speed --> [1] 0
## Generation --> [1] 0
## Legendary --> [1] 0
```

From the above observation, it seems that there are no missing values. But there are more empty values(i.e missing values) in the data-set

Converting the empty values in missing values

```r
df[df == ""] <- NA
```

```r
for ( col in 1:ncol(df)) {
  cat(colnames(df)[col])
  cat(" --> ")
  print(sum(is.na(df[,col])))
}
```

```
## Name --> [1] 0
## Type.1 --> [1] 0
## Type.2 --> [1] 386
## Total --> [1] 0
## HP --> [1] 0
## Attack --> [1] 0
## Defense --> [1] 0
## Sp..Atk --> [1] 0
## Sp..Def --> [1] 0
## Speed --> [1] 0
## Generation --> [1] 0
## Legendary --> [1] 0
```

```r
print("Percentage of Missing values in Type.2 attribute")
```

```
## [1] "Percentage of Missing values in Type.2 attribute"
```

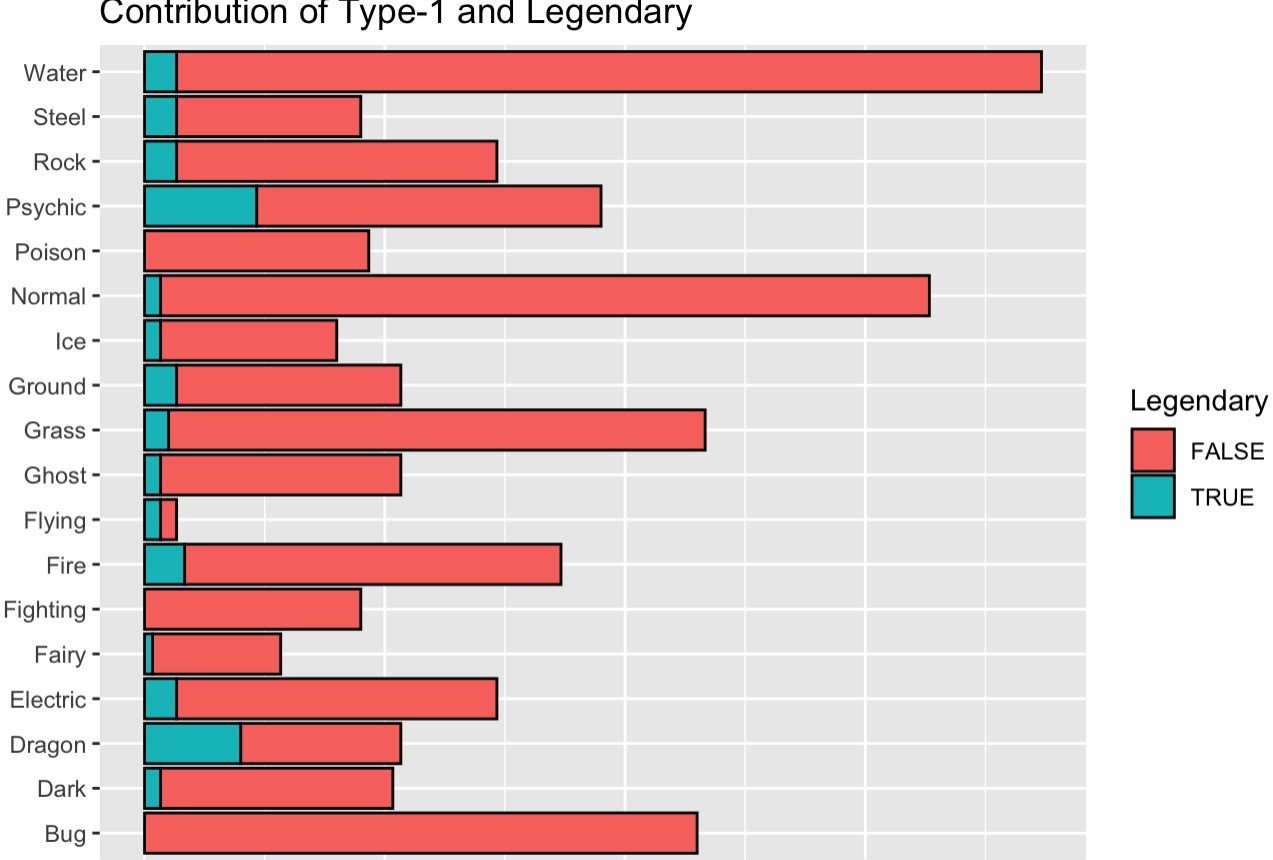
```r
print((abs(nrow(df) - sum(is.na(df['Type.2'])))) / nrow(df))
```

```
## [1] 0.5175
```

There are 51% missing values in 'Type-2' attribute

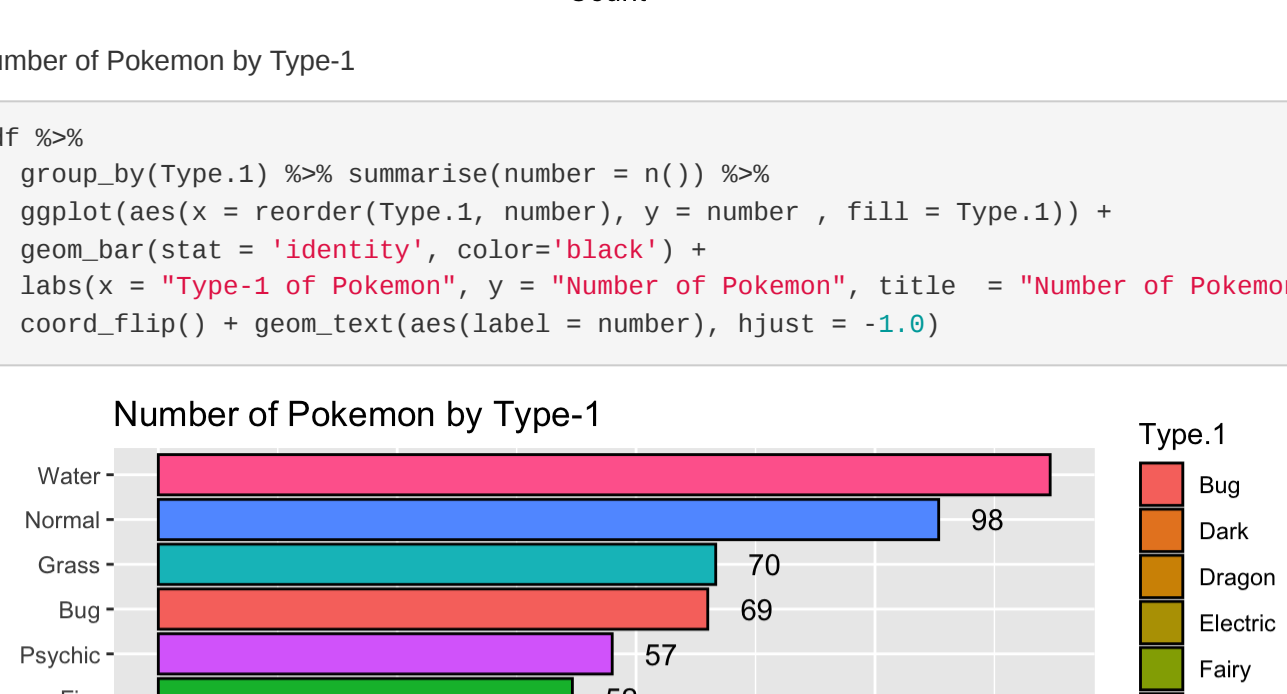
1)Bar Chart

```r
ggplot(df,
       aes(x = Type.1, fill = Legendary)) +
       geom_bar(position = "stack", color='black') +
       labs(x = "Type-1", y = "Count", title  = "Contribution of Type-1 and Legendary") +
       coord_flip()
```



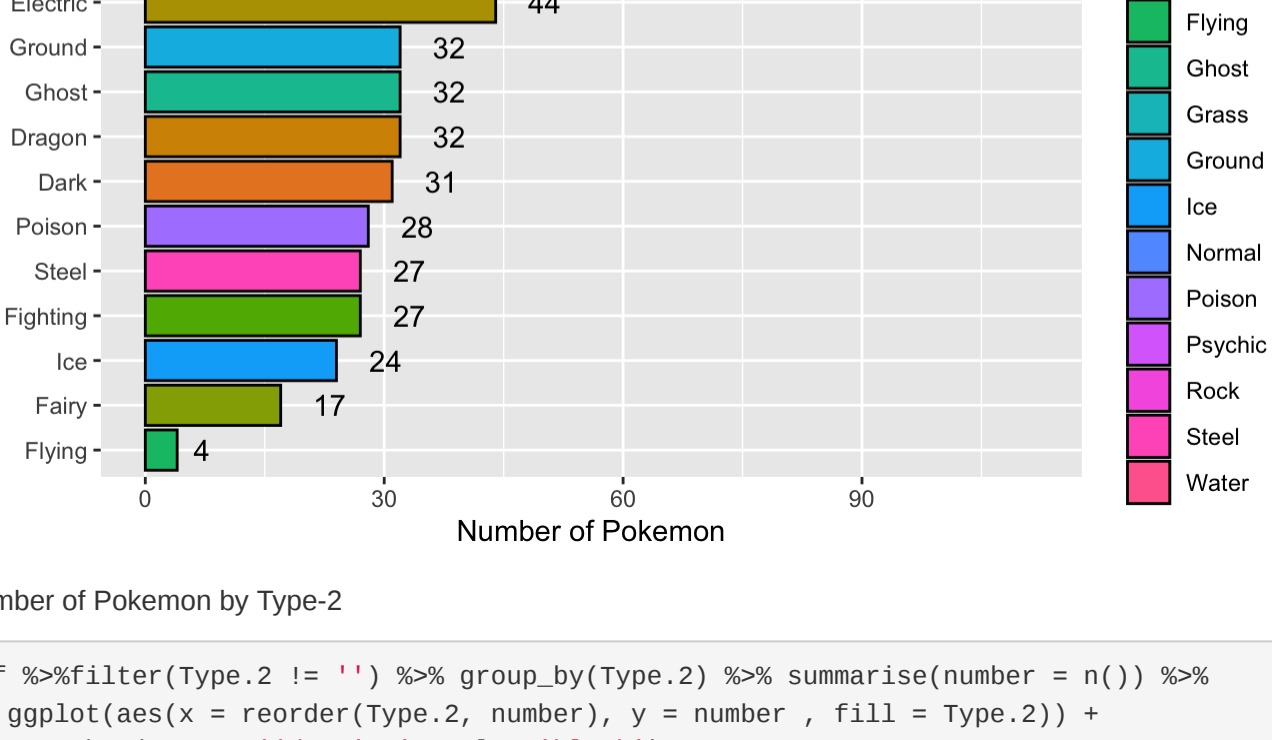
Number of Pokemon by Type-1

```r
df %>%
  group_by(Type.1) %>% summarise(number = n()) %>%
  ggplot(aes(x = reorder(Type.1, number), y = number , fill = Type.1)) +
  geom_bar(stat = 'identity', color='black') +
  labs(x = "Type-1 of Pokemon", y = "Number of Pokemon", title  = "Number of Pokemon by Type-1") +
  coord_flip() + geom_text(aes(label = number), hjust = -1.0)
```



Number of Pokemon by Type-2

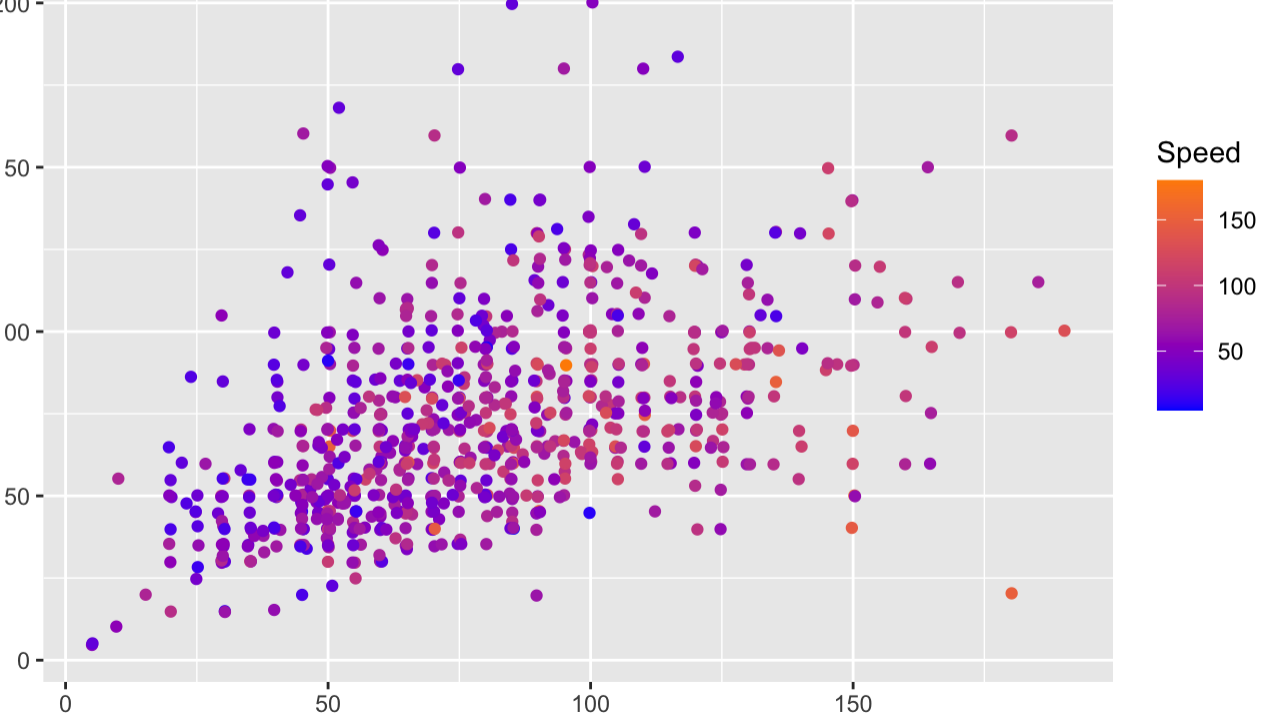
```r
df %>%filter(Type.2 != '') %>% group_by(Type.2) %>% summarise(number = n()) %>%
  ggplot(aes(x = reorder(Type.2, number), y = number , fill = Type.2)) +
  geom_bar(stat = 'identity', color='black') +
  labs(x = "Type-2 of Pokemon", y = "Number of Pokemon", title  = "Number of Pokemon by Type-2") +
  coord_flip() + geom_text(aes(label = number), hjust = -1.0)
```



2)Scatter plot

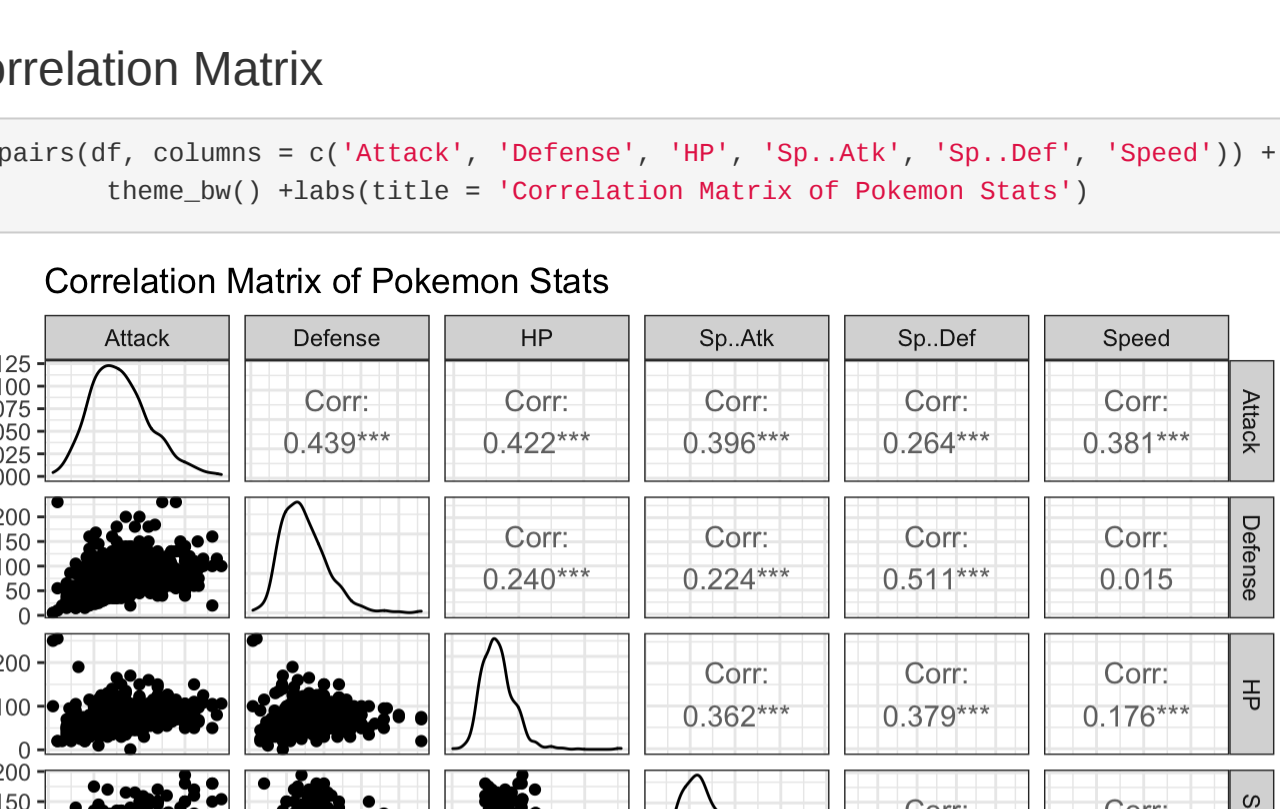
Pokemons with higher attack ratings are faster.

```r
ggplot(df, aes(Attack, Defense)) + geom_jitter(aes(col=Speed)) +
       scale_color_gradient(low="blue", high="darkorange") +
       ggtitle("Defense vs Attack wrt Speed")
```



Correlation Matrix

```r
ggpairs(df, columns = c('Attack', 'Defense', 'HP', 'Sp..Atk', 'Sp..Def', 'Speed')) +
        theme_bw() +labs(title = 'Correlation Matrix of Pokemon Stats')
```



3)Density Plot(Univariate analysis)

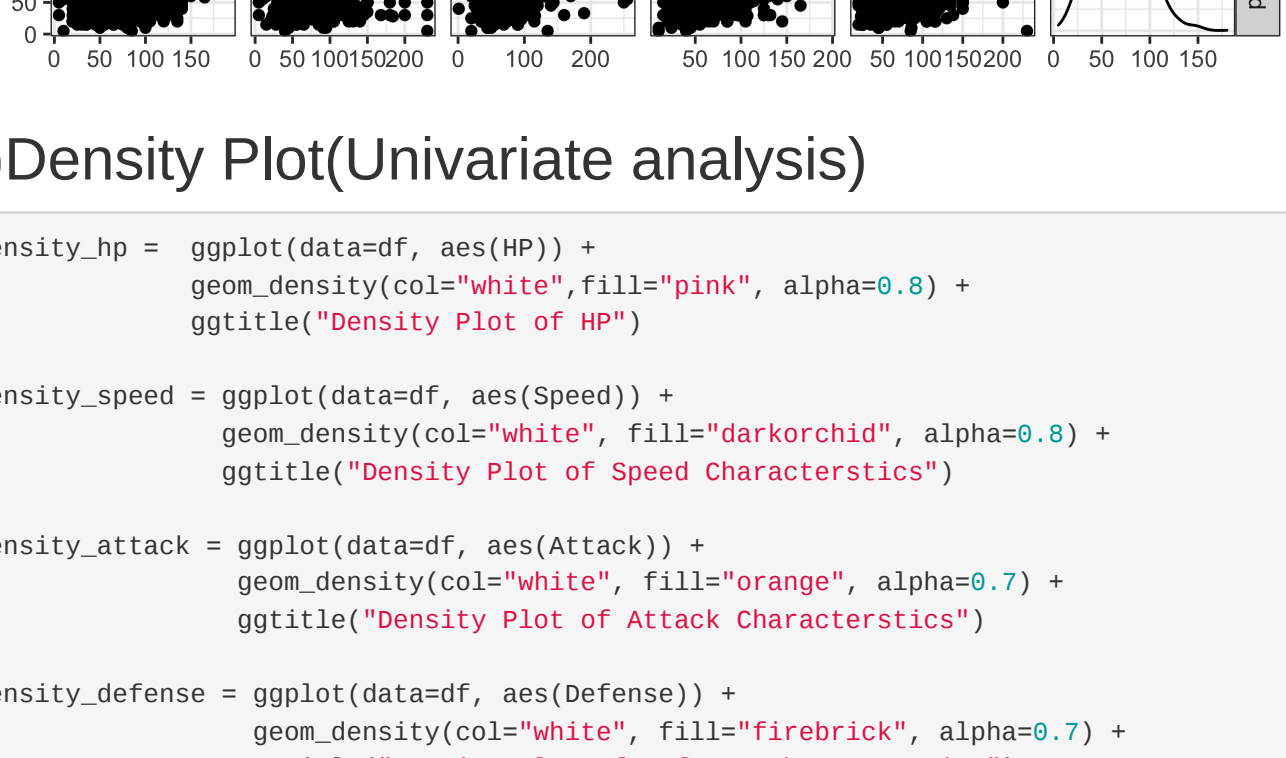
```r
density_hp =  ggplot(data=df, aes(HP)) +
              geom_density(col="white",fill="pink", alpha=0.8) +
              ggtitle("Density Plot of HP")

density_speed = ggplot(data=df, aes(Speed)) +
                geom_density(col="white", fill="darkorchid", alpha=0.8) +
                ggtitle("Density Plot of Speed Characterstics")

density_attack = ggplot(data=df, aes(Attack)) +
                 geom_density(col="white", fill="orange", alpha=0.7) +
                 ggtitle("Density Plot of Attack Characterstics")

density_defense = ggplot(data=df, aes(Defense)) +
                  geom_density(col="white", fill="firebrick", alpha=0.7) +
                  ggtitle("Density Plot of Defense Characterstics")

grid.arrange(density_hp, density_speed, density_attack, density_defense, ncol=2)
```

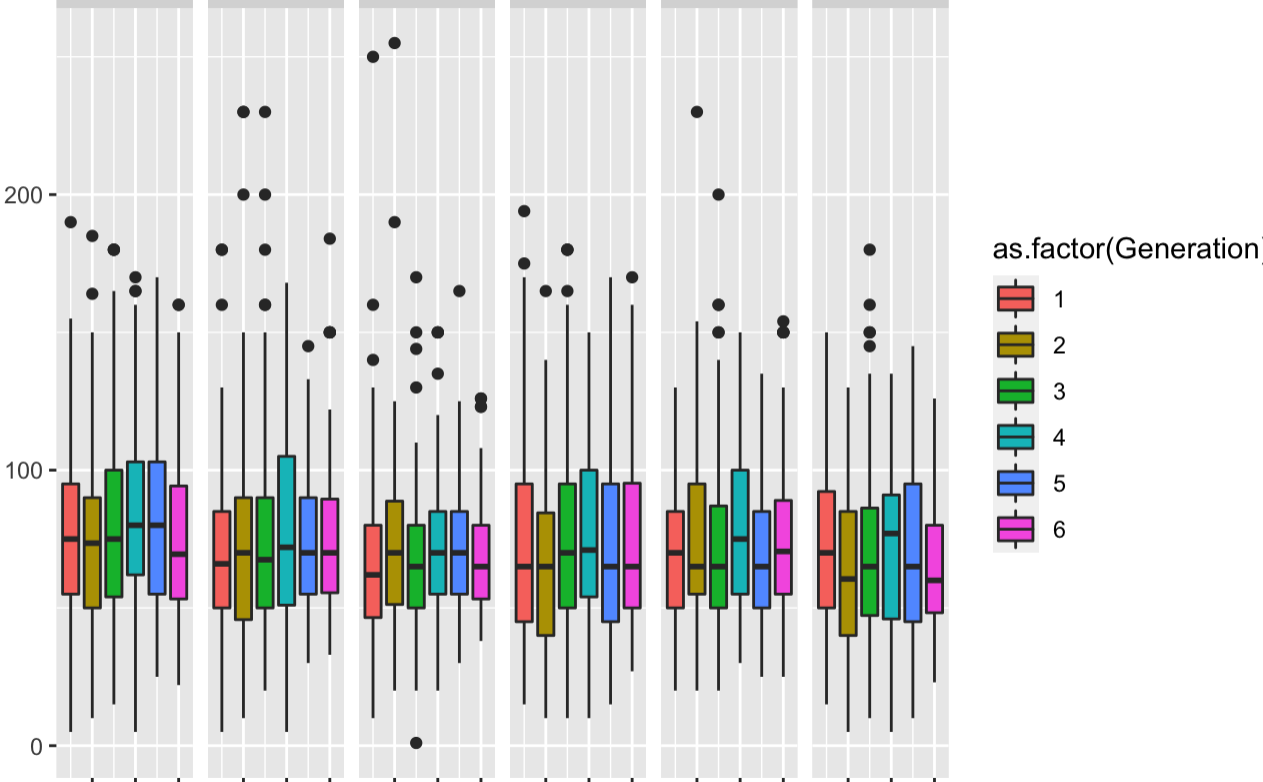


4)Box-Plot(Multi-variate Analysis)

Score of Pokemon by generation

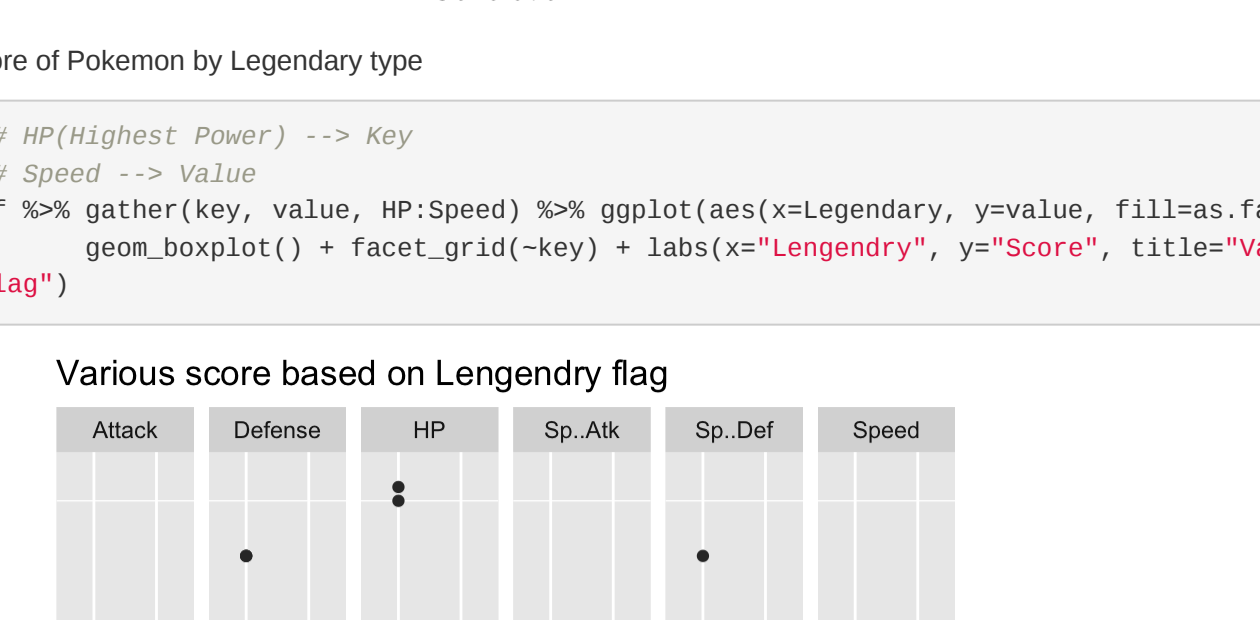
```r
## HP(Highest Power) --> Key
## Speed --> Value

df %>% gather(key, value, HP:Speed) %>% ggplot(aes(x = Generation, y = value, fill = as.factor(Generation))) +
       geom_boxplot() + facet_grid(~key) + labs(x="Generation", y="Score", title="Various score based on Generati
on flag")
```



Score of Pokemon by Legendary type

```r
## HP(Highest Power) --> Key
## Speed --> Value
df %>% gather(key, value, HP:Speed) %>% ggplot(aes(x=Legendary, y=value, fill=as.factor(Legendary))) +
       geom_boxplot() + facet_grid(~key) + labs(x="Lengendry", y="Score", title="Various score based on Lengendry
flag")
```



5) Pie-Chart(Multi-Variate Analysis)

```r
df1 = data.frame(unique(df$Type.1),aggregate(df,by=list(df$Type.1),FUN=median)["HP"])
```

```
## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA

## Warning in mean.default(sort(x, partial = half + 0L:1L)[half + 0L:1L]): argument
## is not numeric or logical: returning NA
```

```r
plot_ly(df1,labels=~unique.df.Type.1.,values =~HP,type ="pie")
```