EDA LA 2 Pokemon Dataset

LAKSHMINARASIMHAN,KISHORE

EDA LA 2 "Pokemon Dataset" Group 25: LAKSHMINARASIMHAN P S (1NT20IS084), KISHORE M (1NT20IS080)

Github link: https://github.com/Kishoremanjunath09/Exploratory-data-analysis

#The below commands are used to load several libraries into the R environment. The libraries included here are ggplot2, ggthemes, GGally, plotrix and tidyverse. These are used for data manipulation and visualization in R. Finally we load the data from pokemondataset.csv into the object pokemon using the read.csv command

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✓ dplyr     1.1.2     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.2     ✓ tibble    3.2.1
## ✓ lubridate 1.9.2     ✓ tidyr     1.3.0
## ✓ purrr     1.0.1
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all
conflicts to become errors
```

```
library(ggplot2)
library(ggthemes)
library(GGally)
```

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

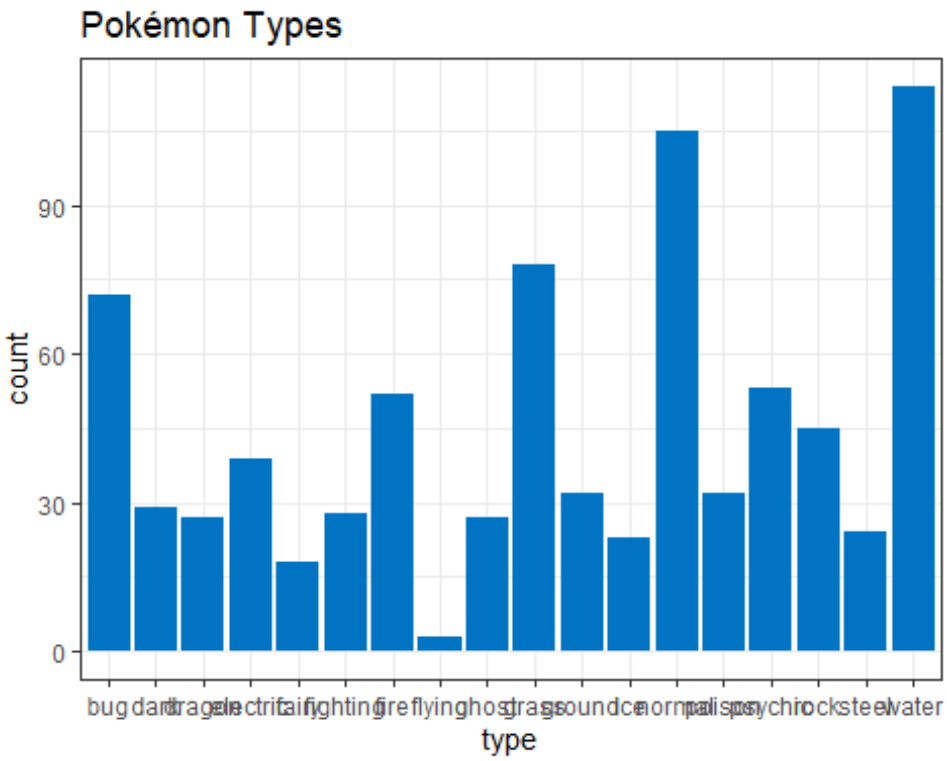
```
library(plotrix)
```

```
pokemon <- read.csv("pokemon.csv")
```

#The provided R commands utilize the "pokemon" dataset to create visualizations that explore different aspects of the data. These commands generate plots to understand the distribution of Pokémon types, compare attack and defense attributes, examine base experience by type, visualize the speed distribution, and analyze Pokémon generations.

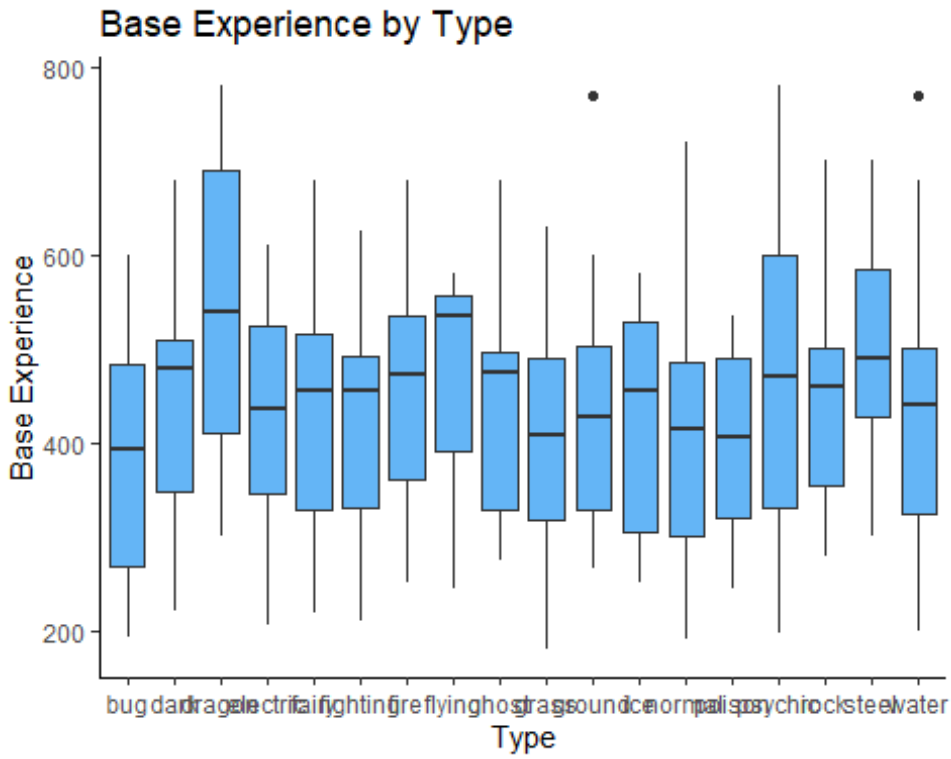
#The below set of commands work on the dataset of "type1" and creates a bar plot to display the count of Pokémon for each type. It provides an overview of the Pokémon types present in the dataset according to their types

```
pokemon %>%
  group_by(type1) %>%
  summarise(count = n()) %>%
  ggplot(aes(x = type1, y = count)) +
  geom_bar(stat = "identity", fill = "#0073C2FF") +
  ggtitle("Pokémon Types") +
  xlab("type") +
  ylab("count") +
  theme_bw()
```



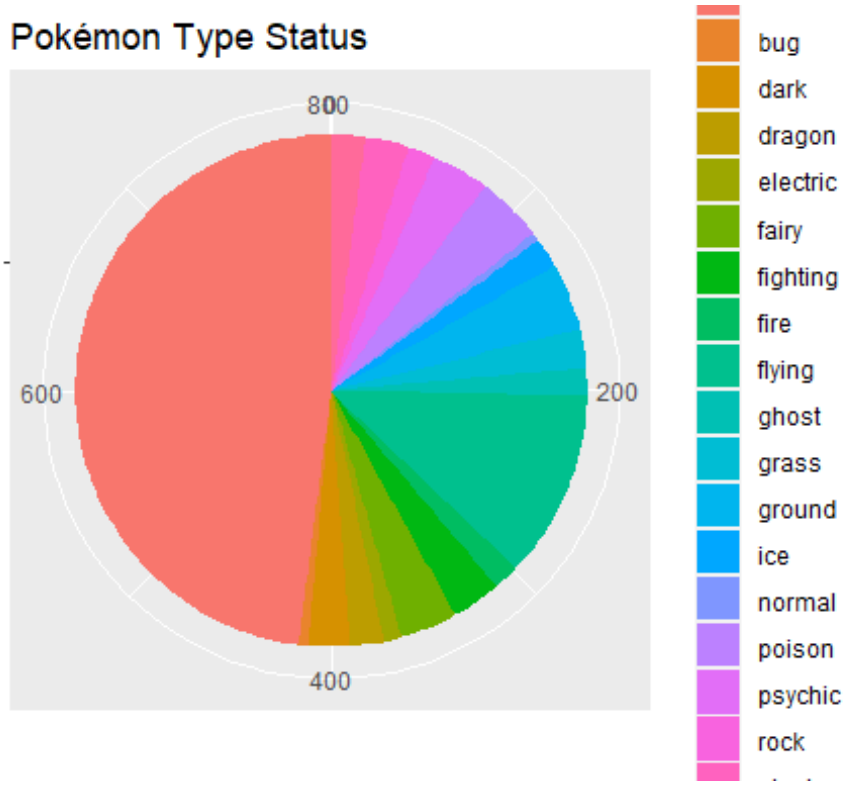
#The below command produces a boxplot that illustrates the distribution of base experience of each pokemon across different Pokémon types. It allows for comparisons of base experience between different types to know their ability of experience

```
pokemon %>%
  ggplot(aes(x = type1, y = base_total)) +
  geom_boxplot(fill = "#64B5F6FF") +
  ggtitle("Base Experience by Type") +
  xlab("Type") +
  ylab("Base Experience") +
  theme_classic()
```



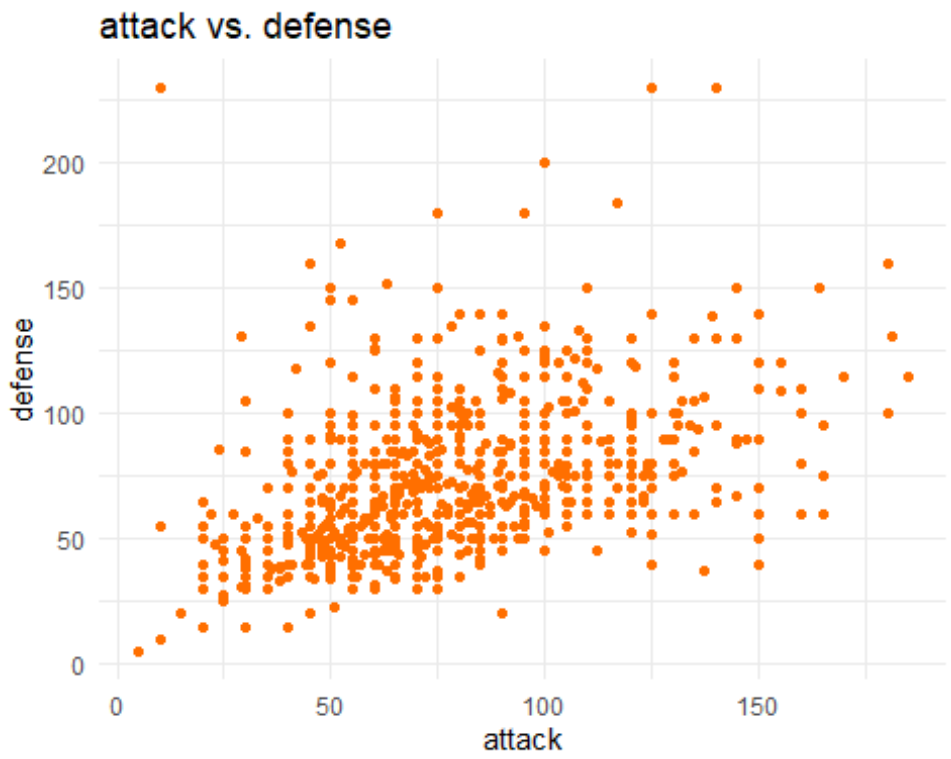
#The below command groups the dataset by “type2” and creates a polar bar plot to represent the count of Pokémon for each type. The angle of each sector corresponds to the count, providing a visual representation of the distribution of Pokémon types in the “type2” category.

```
pokemon %>%
  group_by(type2) %>%
  summarise(count = n()) %>%
  ggplot(aes(x = "", y = count, fill = type2)) +
  geom_bar(stat = "identity", width = 1) +
  coord_polar("y", start = 0) +
  ggtitle("Pokémon Type Status") +
  xlab("") +
  ylab("")
```



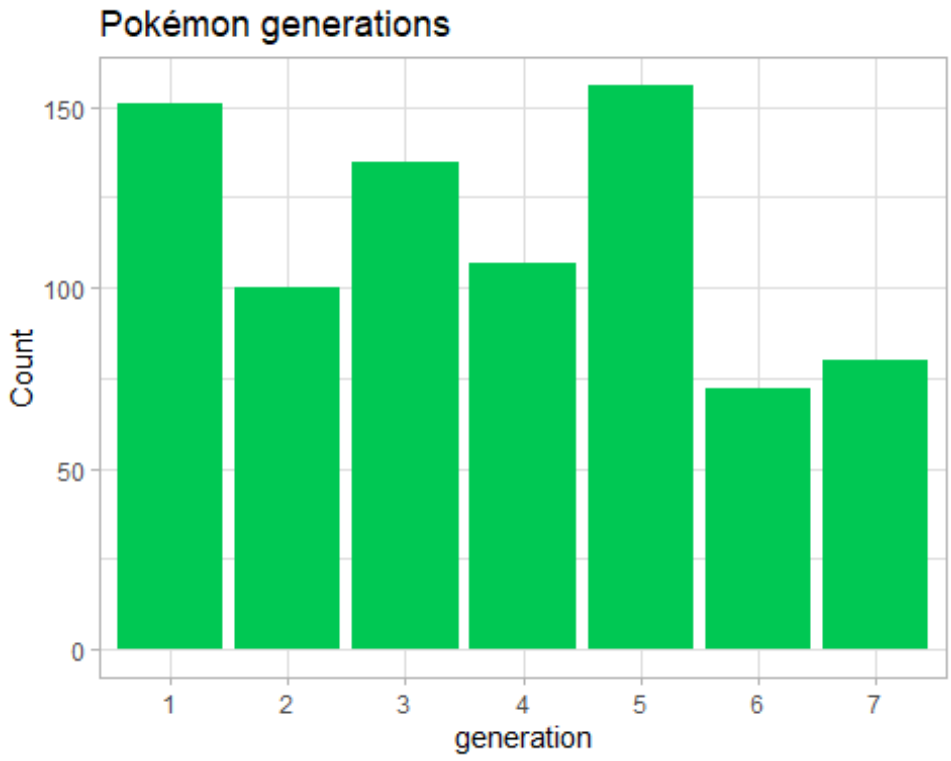
#The below command generates a scatter plot to compare the “attack” and “defense” attributes of each Pokémon. It helps identify any potential relationships or patterns between these attributes.

```
pokemon %>%
  ggplot(aes(x = attack, y = defense)) +
  geom_point(color = "#FF6F00FF") +
  ggtitle("attack vs. defense") +
  xlab("attack") +
  ylab("defense") +
  theme_minimal()
```



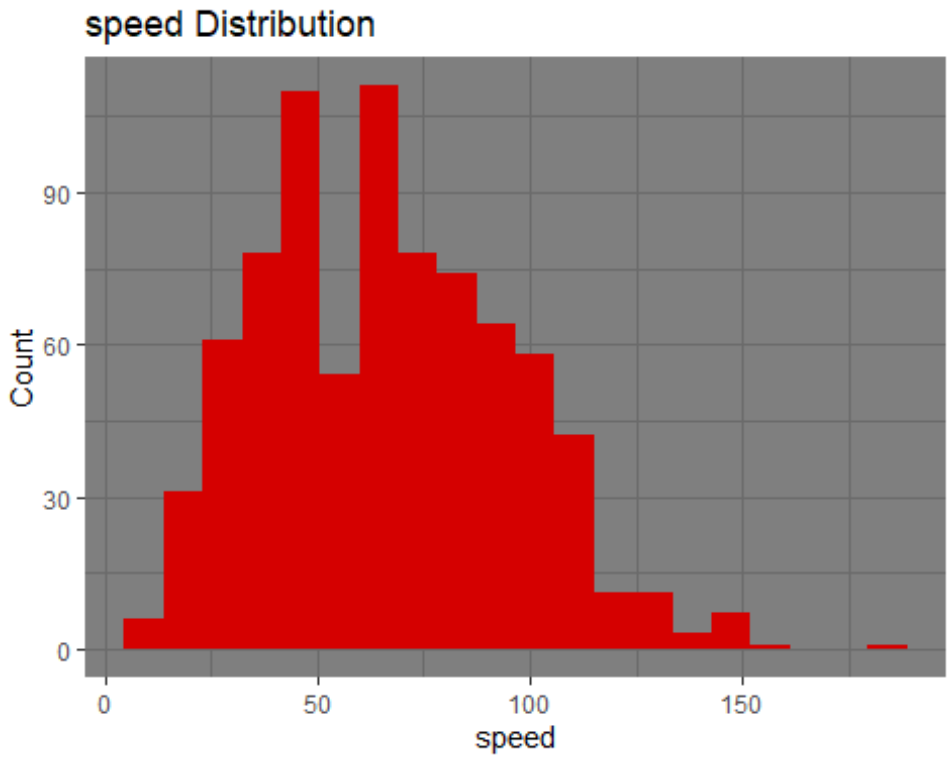
#The below command groups the dataset by "generation" and creates a bar plot to display the count of Pokémon for each generation. It helps understand the distribution of Pokémon across different generations.

```
pokemon %>%
  group_by(generation) %>%
  summarise(count = n()) %>%
  ggplot(aes(x = as.factor(generation), y = count)) +
  geom_bar(stat = "identity", fill = "#00C853FF") +
  ggtitle("Pokémon generations") +
  xlab("generation") +
  ylab("Count") +
  theme_light()
```



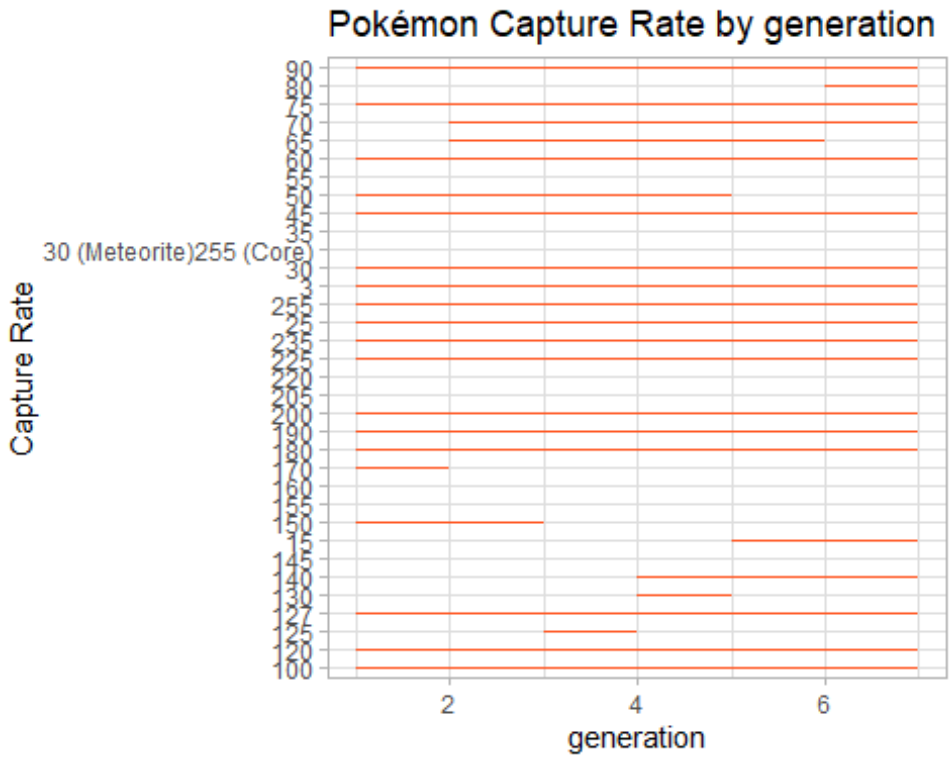
#The below command creates a histogram to visualize the distribution of Pokémon speeds. It provides insights into the frequency and range of speed values among the Pokémon in the dataset.

```
pokemon %>%
  ggplot(aes(x = speed)) +
  geom_histogram(fill = "#D50000FF", bins = 20) +
  ggtitle("speed Distribution") +
  xlab("speed") +
  ylab("Count") +
  theme_dark()
```



#The below command generates a line plot to visualize the capture rate of Pokémon across different generations. It helps identify any trends or patterns in capture rates over time.

```
pokemon %>%
  ggplot(aes(x = generation, y = capture_rate)) +
  geom_line(color = "#FF5722FF") +
  ggtitle("Pokémon Capture Rate by generation") +
  xlab("generation") +
  ylab("Capture Rate") +
  theme_light()
```

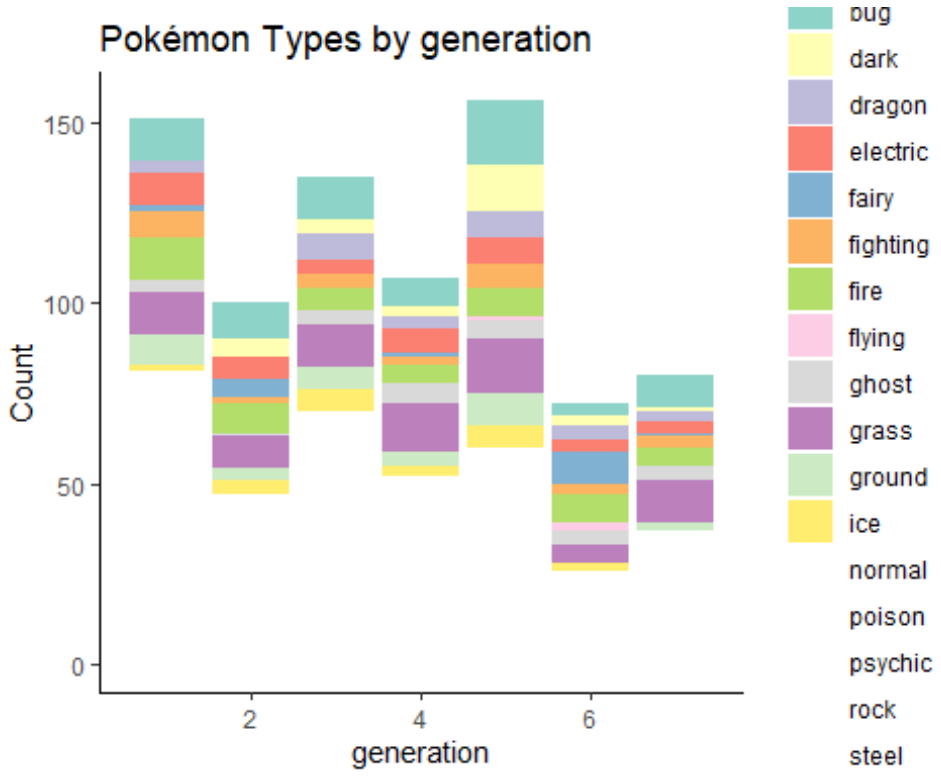


#The below command groups the dataset by both “generation” and “type1” and creates a stacked bar plot to showcase the count of Pokémon for each combination. It provides insights into the distribution of Pokémon types across different generations.

```
pokemon %>%
  group_by(generation, type1) %>%
  summarise(count = n()) %>%
  ggplot(aes(x = generation, y = count, fill = type1)) +
  geom_bar(stat = "identity") +
  ggtitle("Pokémon Types by generation") +
  xlab("generation") +
  ylab("Count") +
  theme_classic() +
  scale_fill_brewer(palette = "Set3")
```

```
## `summarise()` has grouped output by 'generation'. You can override using
the
## `.groups` argument.
```

```
## Warning in RColorBrewer::brewer.pal(n, pal): n too large, allowed maximum
for palette Set3 is 12
## Returning the palette you asked for with that many colors
```



#The below command calculates the correlation matrix for a subset of attributes (HP, attack, defense, special attack, special defense, and speed) from the "pokemon" dataset.

```
pokemon_stats <- pokemon[, c("hp", "attack", "defense", "sp_attack",
"sp_defense", "speed")]
corr_matrix <- cor(pokemon_stats)
corr_matrix

##                  hp    attack     defense sp_attack sp_defense
speed
## hp         1.0000000 0.4106158 0.242378156 0.3659726  0.3669707
0.160759818
## attack     0.4106158 1.0000000 0.468914914 0.3681540  0.2658362
0.352702640
## defense    0.2423782 0.4689149 1.000000000 0.2418820  0.5263483
0.007934069
## sp_attack  0.3659726 0.3681540 0.241882033 1.0000000  0.5114955
0.438981268
## sp_defense 0.3669707 0.2658362 0.526348276 0.5114955  1.0000000
0.225976980
## speed      0.1607598 0.3527026 0.007934069 0.4389813  0.2259770
1.000000000
```

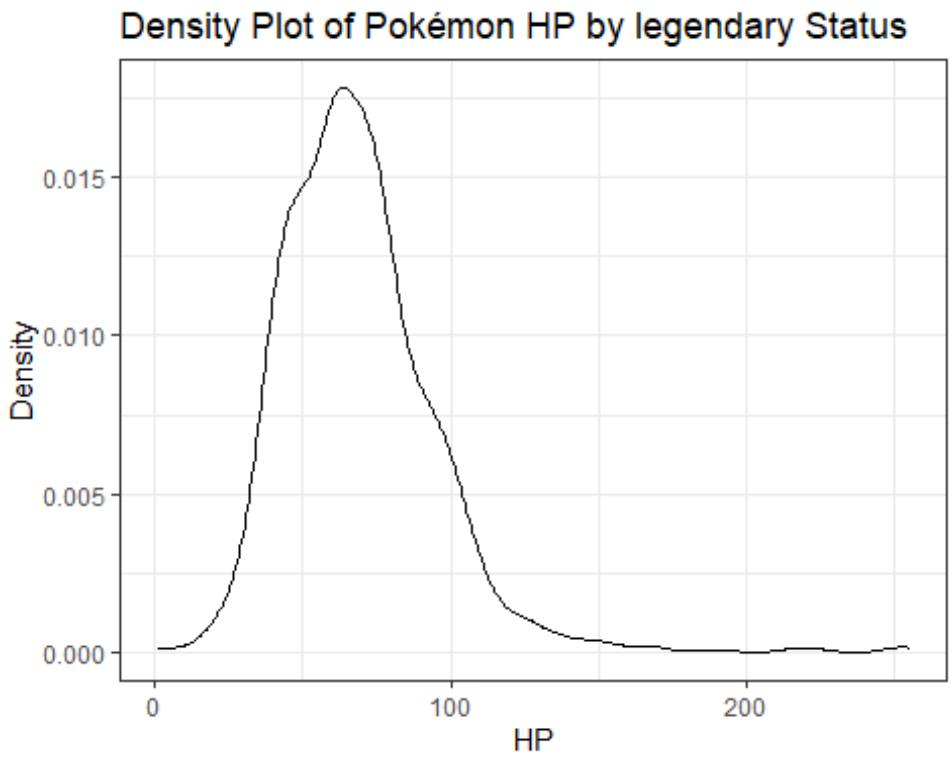
#The below command creates a subset of the "pokemon" dataset, containing only the attributes related to HP, attack, defense, special attack, special defense, and speed. This subset is stored in the "pokemon_stats" object.

```
pokemon_stats <- pokemon[, c("hp", "attack", "defense", "sp_attack",
"sp_defense", "speed")]
```

#The below command creates a density plot to examine the distribution of Pokémon HP values based on their legendary status. It allows for a comparison of HP distributions between legendary and non-legendary Pokémon.

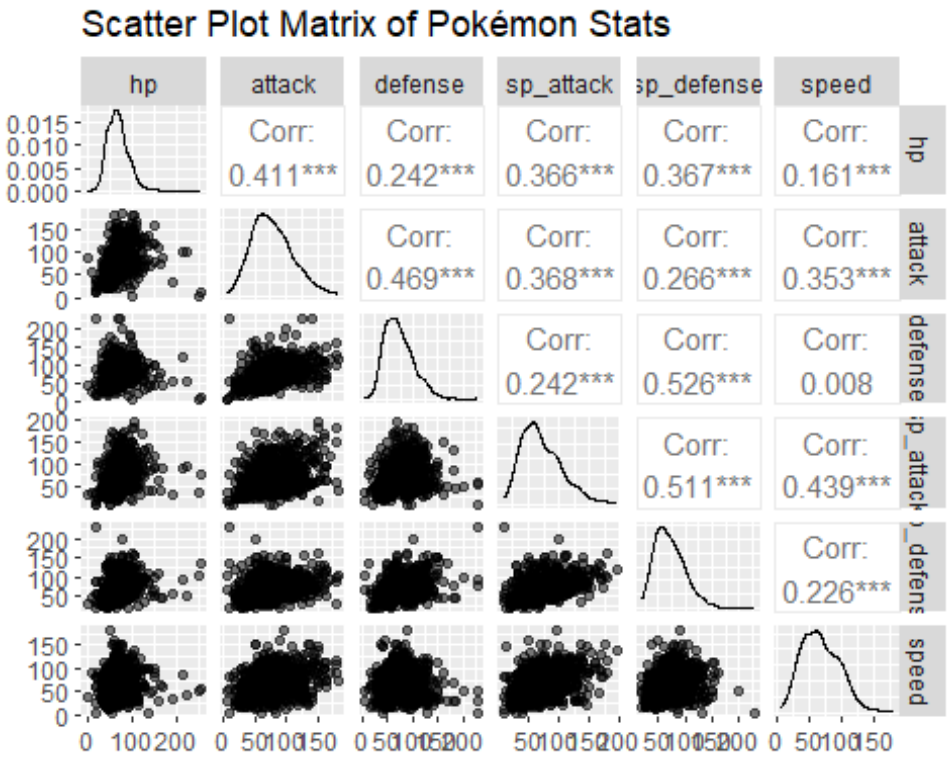
```
pokemon %>%
  ggplot(aes(x = hp, fill = is_legendary)) +
  geom_density(alpha = 0.5) +
  ggtitle("Density Plot of Pokémon HP by legendary Status") +
  xlab("HP") +
  ylab("Density") +
  theme_bw() +
  scale_fill_manual(values = c("#FFAB00FF", "#2962FFFF"))
```

```
## Warning: The following aesthetics were dropped during statistical
transformation: fill
## ℹ This can happen when ggplot fails to infer the correct grouping
structure in
##   the data.
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?
```



#The below command generates a scatter plot matrix using the attributes in the "pokemon_stats" dataset. Each scatter plot represents the pairwise relationship between two attributes. The plots have a title and transparent points.

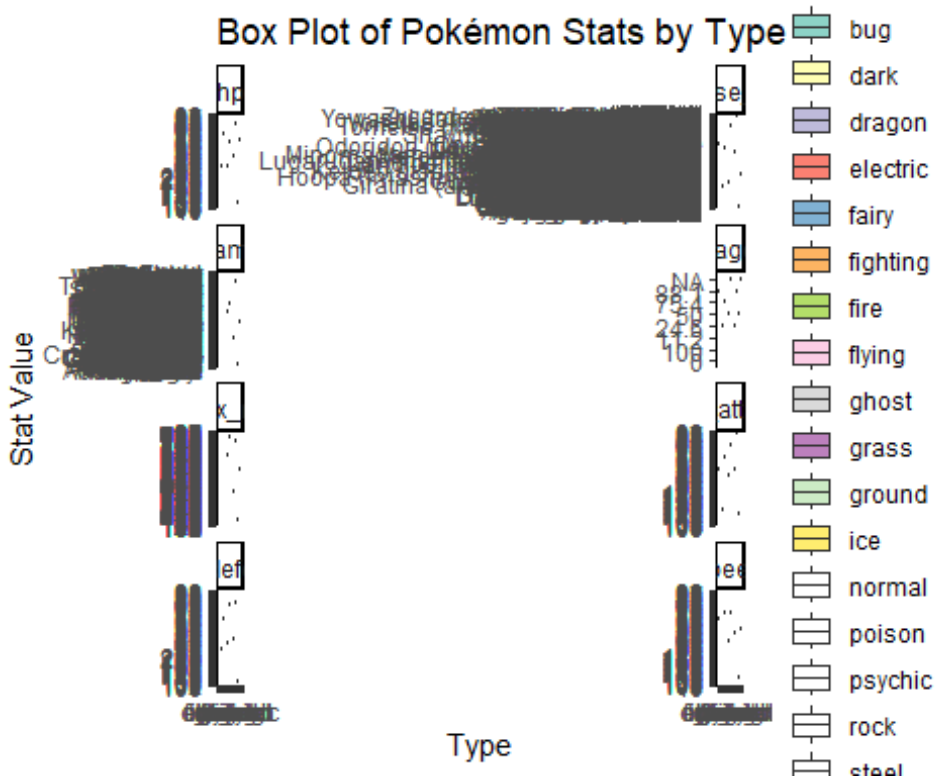
```
ggpairs(pokemon_stats, title = "Scatter Plot Matrix of Pokémon Stats", lower
= list(continuous = wrap("points", alpha = 0.5)))
```



#The below command reshapes the "pokemon" dataset from wide to long format, gathering the attributes from "hp" to "speed" into two columns: "Stat" and "Value". It then creates a box plot for each "Stat" (attribute) grouped by "type1" (Pokémon type). The plot has a title, x-label for type, y-label for stat value, a classic theme, and uses a color palette.

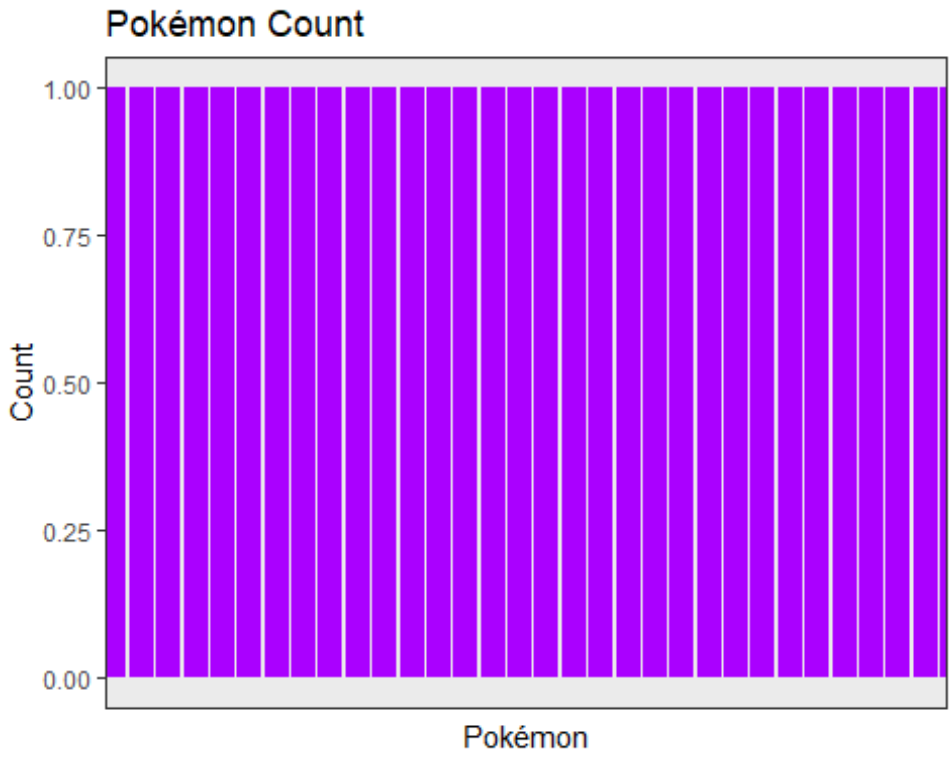
```
pokemon %>%
  gather(key = "Stat", value = "Value", hp:speed) %>%
  ggplot(aes(x = type1, y = Value, fill = type1)) +
  geom_boxplot() +
  facet_wrap(~Stat, scales = "free_y", ncol = 2) +
  ggtitle("Box Plot of Pokémon Stats by Type") +
  xlab("Type") +
  ylab("Stat Value") +
  theme_classic() +
  scale_fill_brewer(palette = "Set3")
```

```
## Warning in RColorBrewer::brewer.pal(n, pal): n too large, allowed maximum
for palette Set3 is 12
## Returning the palette you asked for with that many colors
```



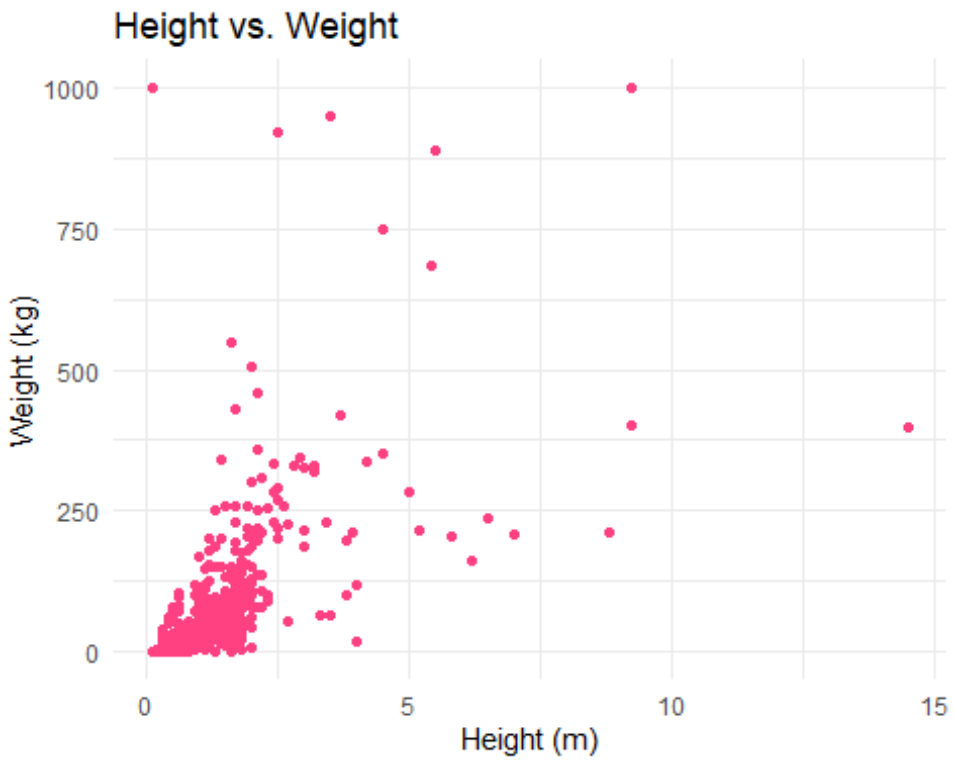
#The below command creates a bar plot to visualize the count of Pokémon. Each bar represents a Pokémon, and the plot has a title, x-label, y-label, and a black and white theme. The x-axis labels and ticks are hidden.

```
pokemon %>%
  ggplot(aes(x = name)) +
  geom_bar(fill = "#AA00FFFF") +
  ggtitle("Pokémon Count") +
  xlab("Pokémon") +
  ylab("Count") +
  theme_bw() +
  theme(axis.text.x = element_blank(), axis.ticks.x = element_blank())
```



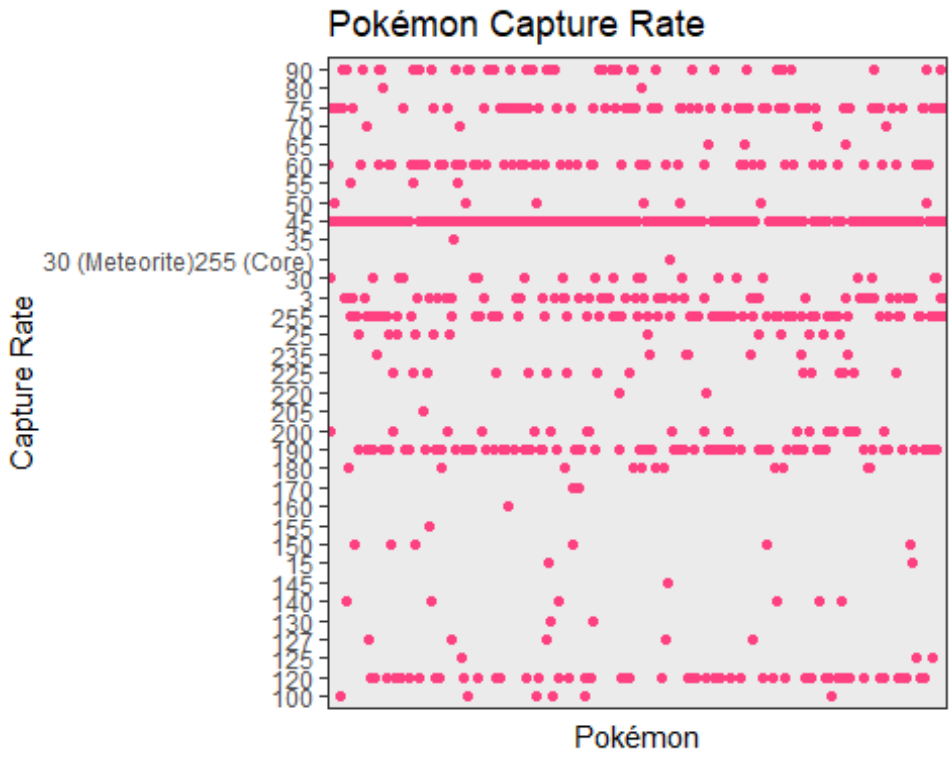
#The below command generates a scatter plot to compare the height and weight of Pokémon. Each point represents a Pokémon, and the plot has a title, x-label for height, y-label for weight, and a minimalistic theme.

```
pokemon %>%
  ggplot(aes(x = height_m, y = weight_kg)) +
  geom_point(color = "#FF4081FF") +
  ggtitle("Height vs. Weight") +
  xlab("Height (m)") +
  ylab("Weight (kg)") +
  theme_minimal()

## Warning: Removed 20 rows containing missing values (`geom_point()`).
```



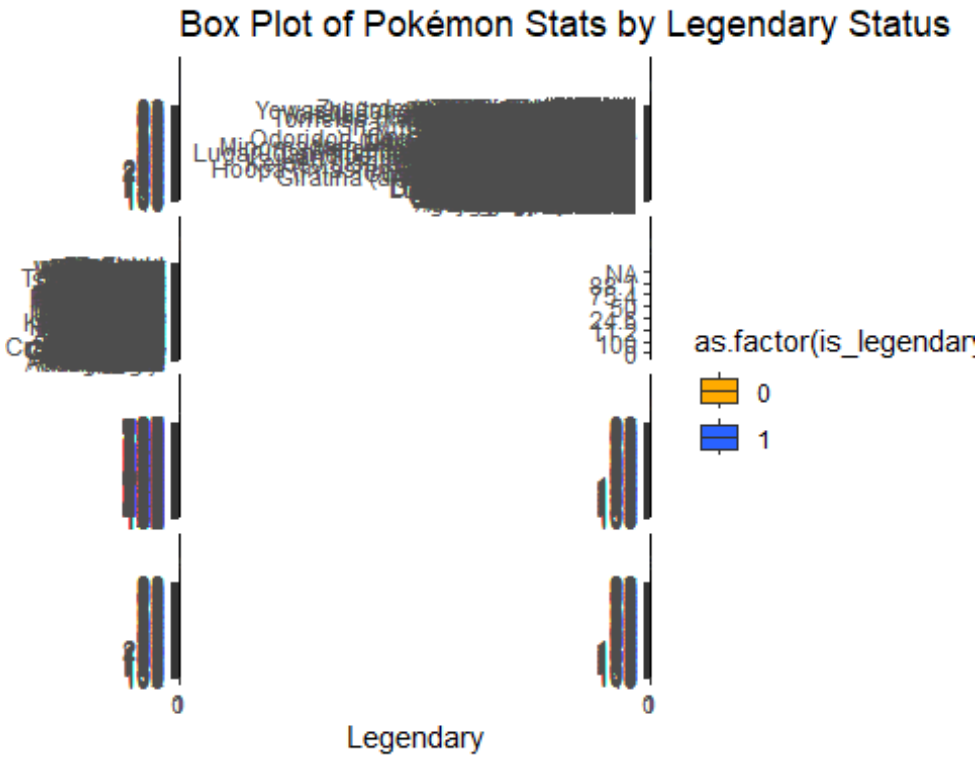
#The below command generates a scatter plot to visualize the capture rate of Pokémon. Each point represents a Pokémon, and the plot has a title, x-label for Pokémon names, y-label for capture rate, and a black and white theme. The x-axis labels and ticks are hidden.

```
pokemon %>%
  ggplot(aes(x = name, y = capture_rate)) +
  geom_point(color = "#FF4081FF") +
  ggtitle("Pokémon Capture Rate") +
  xlab("Pokémon") +
  ylab("Capture Rate") +
  theme_bw() +
  theme(axis.text.x = element_blank(), axis.ticks.x = element_blank())
```



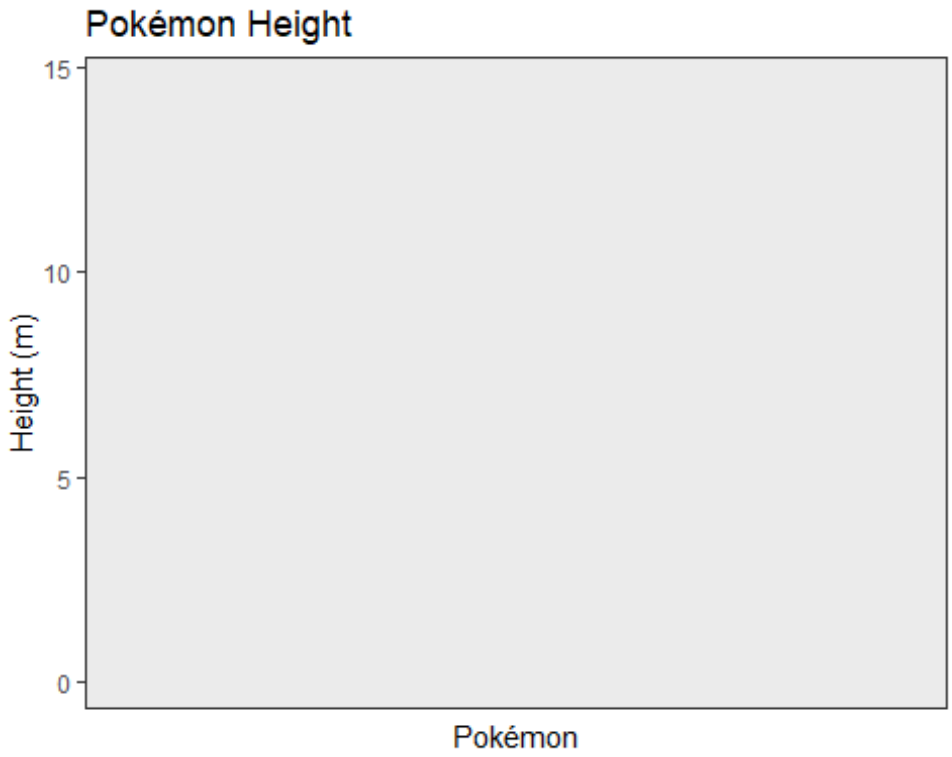
#The below command reshapes the "pokemon" dataset from wide to long format, gathering the attributes from "hp" to "speed" into two columns: "Stat" and "Value". It then creates box plots to compare the distribution of "Stat" (attribute) values for legendary and non-legendary Pokémon. The plot has a title, x-label for legendary status, y-label for stat value, a classic theme, and uses a manual color palette.

```
pokemon %>%
  gather(key = "Stat", value = "Value", hp:speed) %>%
  ggplot(aes(x = as.factor(is_legendary), y = Value, fill =
as.factor(is_legendary))) +
  geom_boxplot() +
  facet_wrap(~Stat, scales = "free_y", ncol = 2) +
  ggtitle("Box Plot of Pokémon Stats by Legendary Status") +
  xlab("Legendary") +
  ylab("Stat Value") +
  theme_classic() +
  scale_fill_manual(values = c("#FFAB00FF", "#2962FFFF"))
```



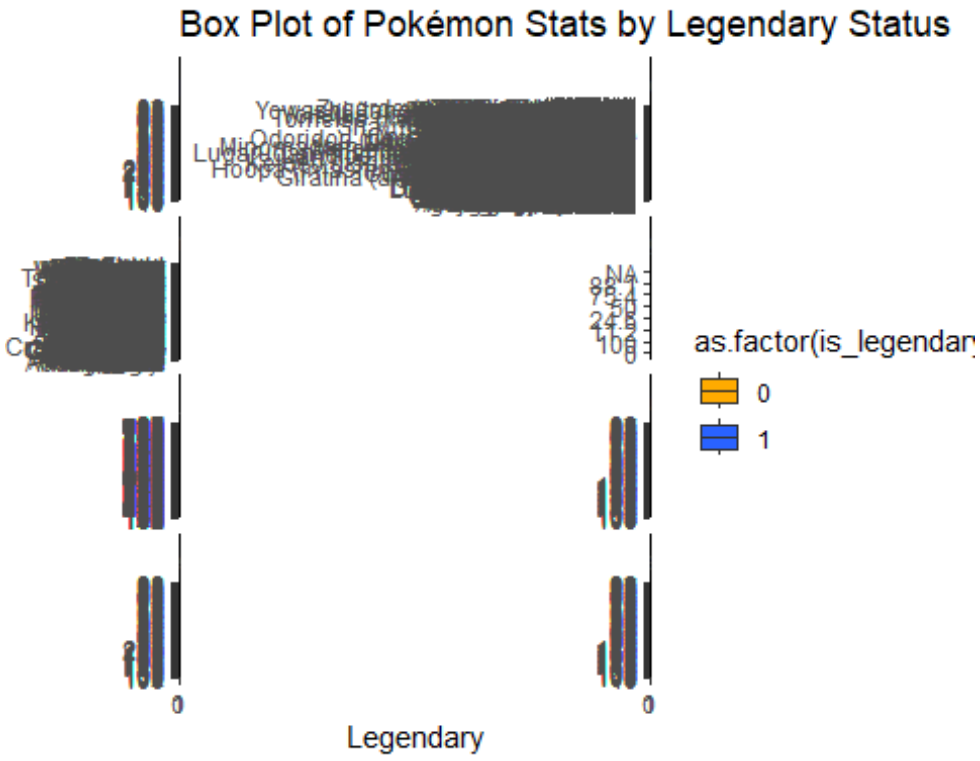
#The below command generates a line plot to visualize the heights of Pokémon. Each line represents the height of a Pokémon over its name. The plot has a title, x-label for Pokémon names, y-label for height in meters, and a black and white theme. The x-axis labels and ticks are hidden.

```
pokemon %>%
  ggplot(aes(x = name, y = height_m)) +
  geom_line(color = "#FF4081FF") +
  ggtitle("Pokémon Height") +
  xlab("Pokémon") +
  ylab("Height (m)") +
  theme_bw() +
  theme(axis.text.x = element_blank(), axis.ticks.x = element_blank())

## Warning: Removed 20 rows containing missing values (`geom_line()`).

## `geom_line()`: Each group consists of only one observation.
## ℹ Do you need to adjust the group aesthetic?
```



#The below command is the same as the first command, reshaping the dataset and creating box plots to compare the distribution of "Stat" (attribute) values for legendary and non-legendary Pokémon. The plot has a title, x-label for legendary status, y-label for stat value, a classic theme, and uses a manual color palette.

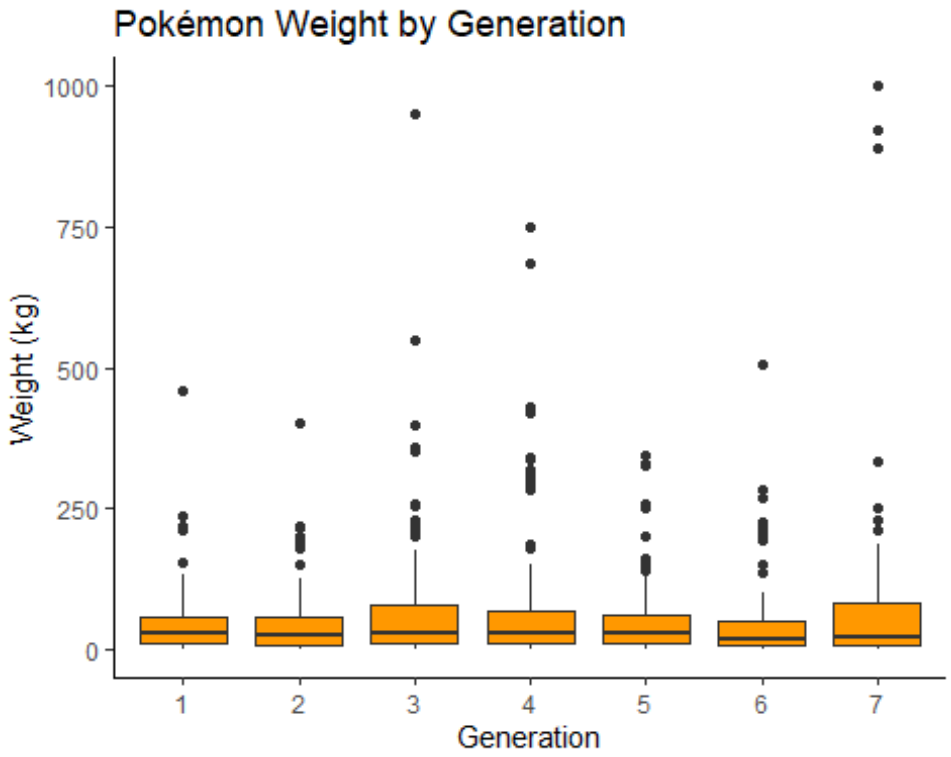
```
pokemon %>%
  gather(key = "Stat", value = "Value", hp:speed) %>%
  ggplot(aes(x = as.factor(is_legendary), y = Value, fill =
as.factor(is_legendary))) +
  geom_boxplot() +
  facet_wrap(~Stat, scales = "free_y", ncol = 2) +
  ggtitle("Box Plot of Pokémon Stats by Legendary Status") +
  xlab("Legendary") +
  ylab("Stat Value") +
  theme_classic() +
  scale_fill_manual(values = c("#FFAB00FF", "#2962FFFF"))
```



#The below command creates box plots to compare the weight of Pokémon across different generations. Each box plot represents the weight distribution within a generation. The plot has a title, x-label for generation, y-label for weight in kilograms, and a classic theme.

```
pokemon %>%
  ggplot(aes(x = as.factor(generation), y = weight_kg)) +
  geom_boxplot(fill = "#FF9800FF") +
  ggtitle("Pokémon Weight by Generation") +
  xlab("Generation") +
  ylab("Weight (kg)") +
  theme_classic()
```

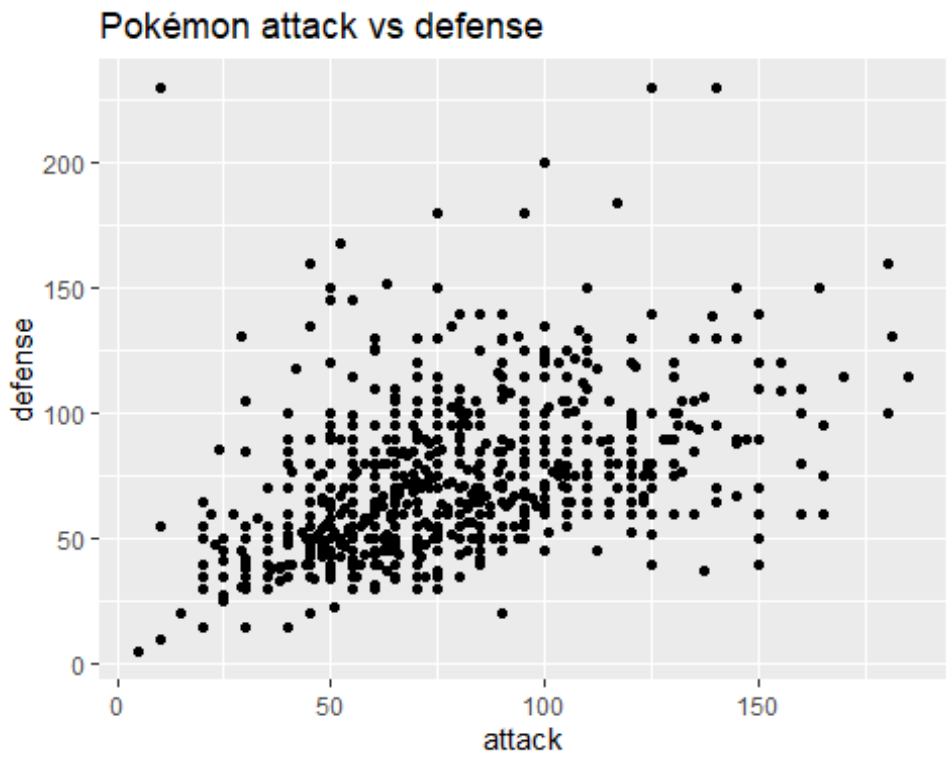
```
## Warning: Removed 20 rows containing non-finite values (`stat_boxplot()`).
```



``

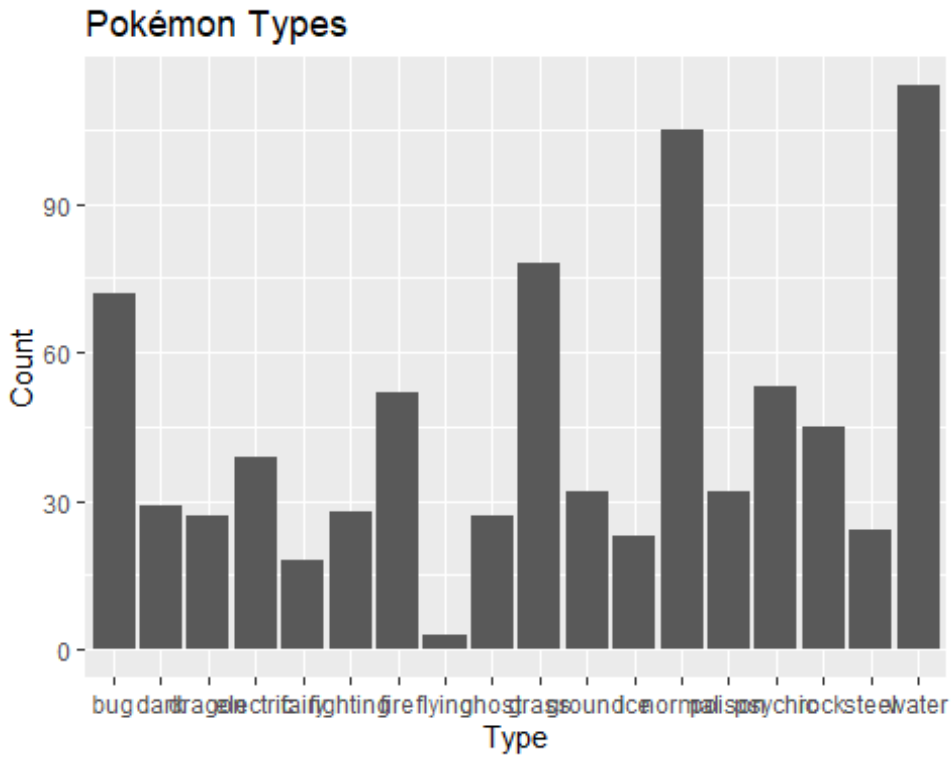
#The below command creates a scatter plot to visualize the relationship between Pokémon's attack and defense attributes. The x-axis represents the attack values, the y-axis represents the defense values. It has labeled axes for attack and defense, and a title stating "Pokémon attack vs defense".

```
ggplot(pokemon, aes(x = attack, y = defense)) +
  geom_point() +
  xlab("attack") +
  ylab("defense") +
  ggtitle("Pokémon attack vs defense")
```



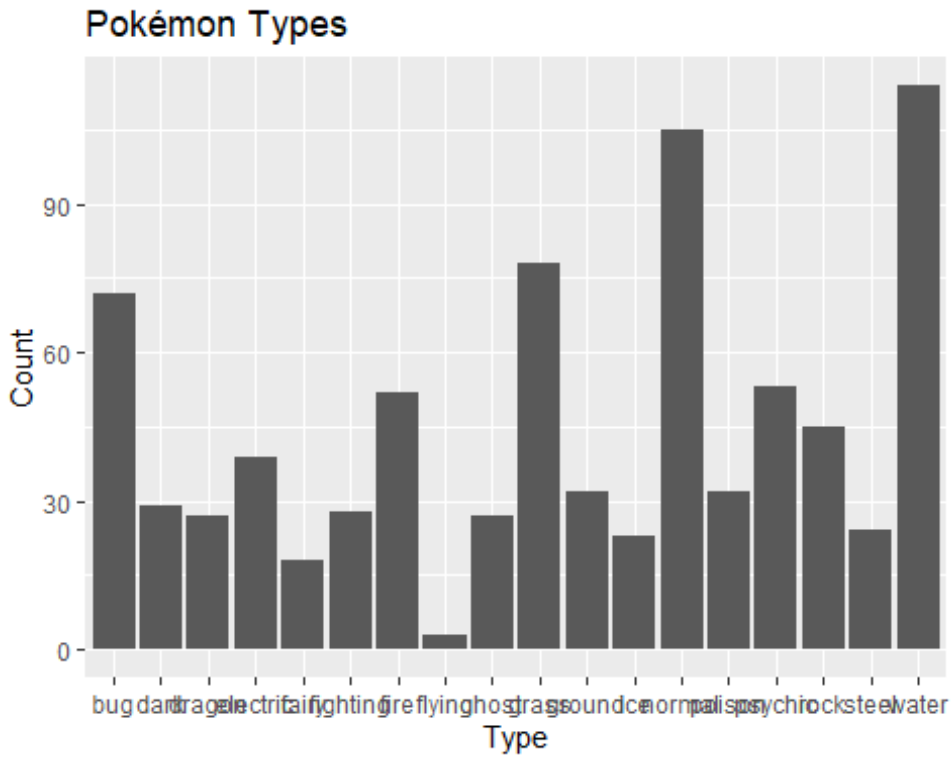
#The below command creates a bar plot to display the count of Pokémon for each type. The x-axis represents the Pokémon types, and the y-axis represents the count. It has labeled axes for type and count, and a title stating “Pokémon Types”.

```
ggplot(pokemon, aes(x = type1)) +
  geom_bar() +
  xlab("Type") +
  ylab("Count") +
  ggtitle("Pokémon Types")
```



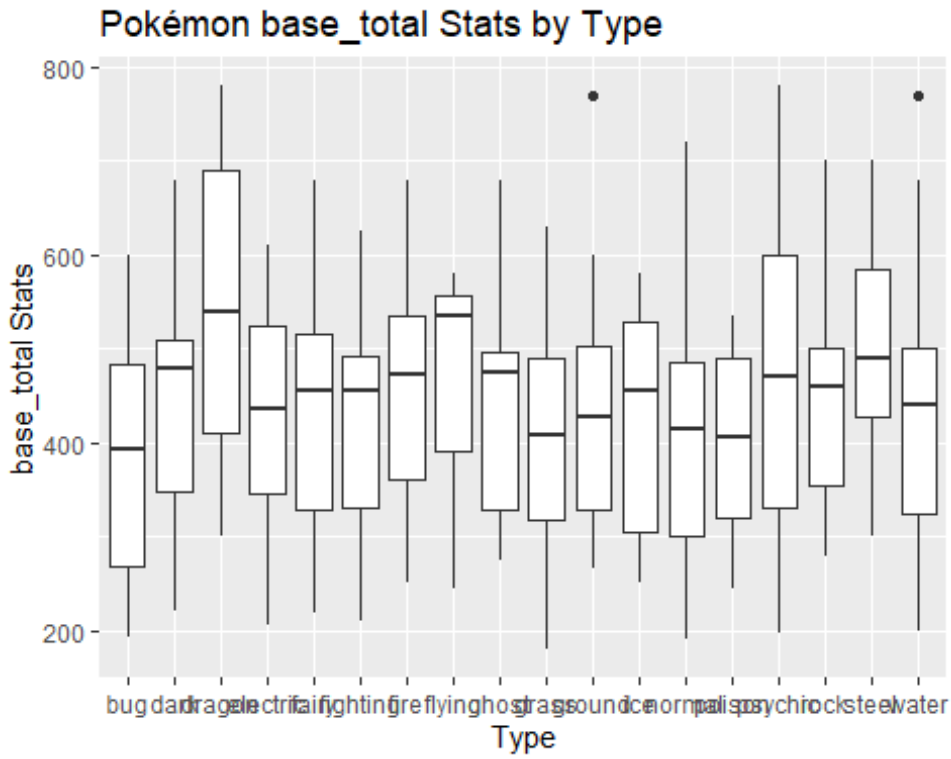
#The below command is the same as the second command, creating a bar plot to display the count of Pokémon for each type. It has the same x-axis, y-axis, and title.

```
ggplot(pokemon, aes(x = type1)) +
  geom_bar() +
  xlab("Type") +
  ylab("Count") +
  ggtitle("Pokémon Types")
```



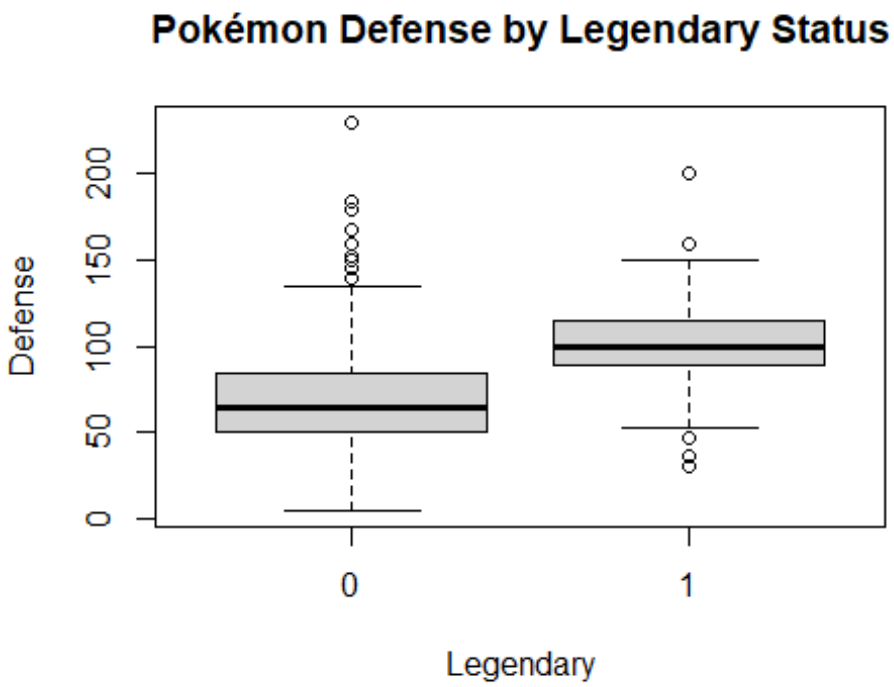
#The below command generates a box plot to compare the distribution of "base_total" stats across different Pokémon types. The x-axis represents the Pokémon types, and the y-axis represents the "base_total" stats. It has labeled axes for type and base_total stats, and a title stating "Pokémon base_total Stats by Type".

```
ggplot(pokemon, aes(x = type1, y = base_total)) +
  geom_boxplot() +
  xlab("Type") +
  ylab("base_total Stats") +
  ggtitle("Pokémon base_total Stats by Type")
```



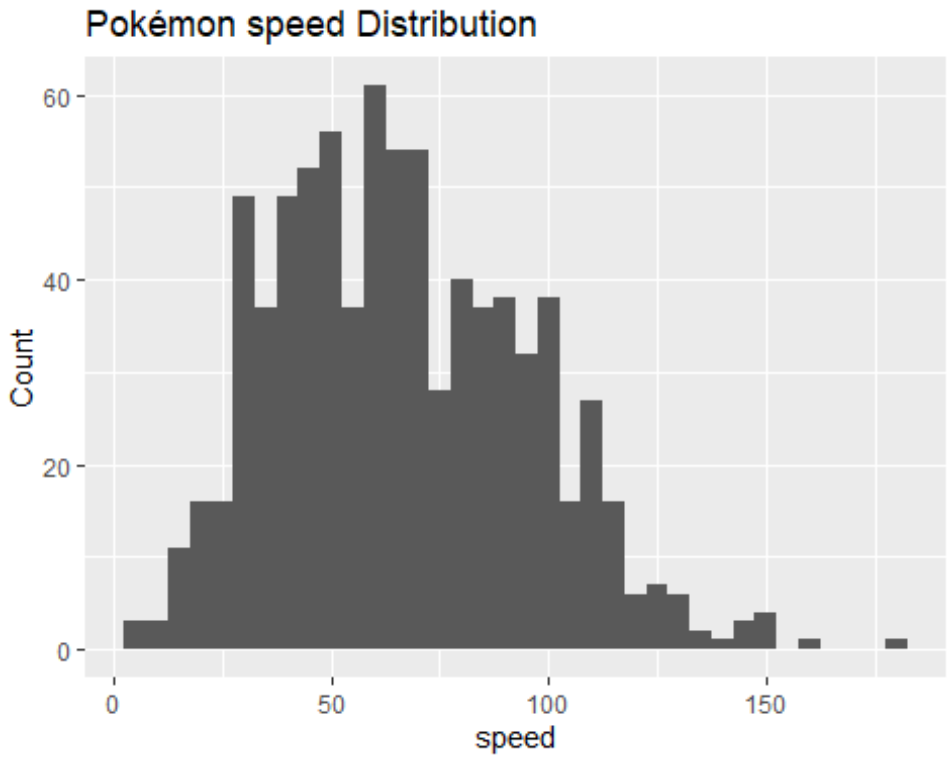
#The below command produces a box plot to compare the defense attribute of Pokémon based on their legendary status. It divides the plot into two groups: legendary and non-legendary. The x-axis represents the legendary status, and the y-axis represents the defense values. It has labeled axes for legendary and defense, and a title stating "Pokémon Defense by Legendary Status".

```
boxplot(defense ~ is_legendary, data = pokemon,
        xlab = "Legendary",
        ylab = "Defense",
        main = "Pokémon Defense by Legendary Status")
```



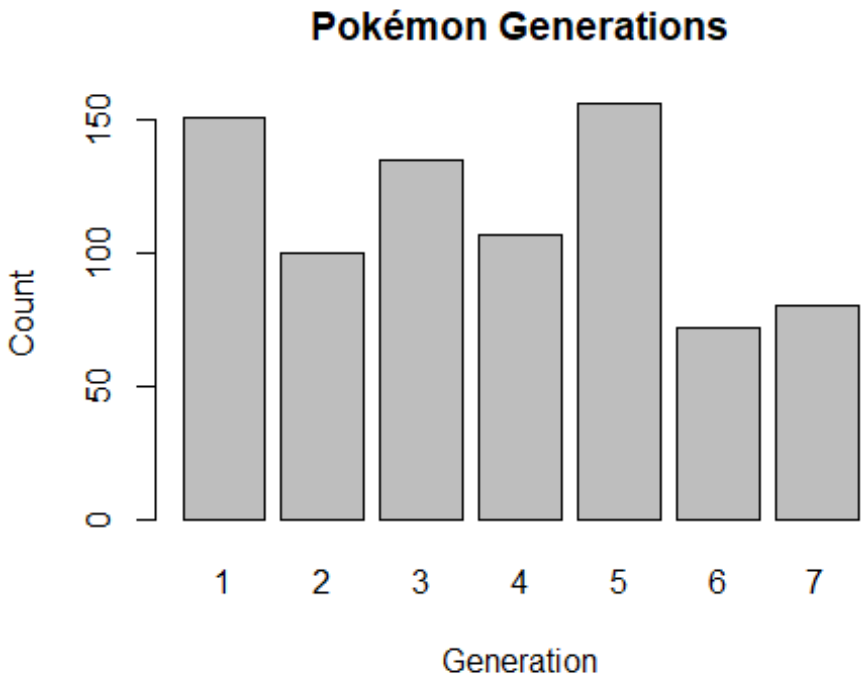
#The below command creates a histogram to visualize the distribution of Pokémon speeds. The x-axis represents the speed values, and the y-axis represents the count of Pokémon in each speed bin. It has a labeled x-axis for speed, a labeled y-axis for count, and a title stating "Pokémon speed Distribution"

```
ggplot(pokemon, aes(x = speed)) +
  geom_histogram(binwidth = 5) +
  xlab("speed") +
  ylab("Count") +
  ggtitle("Pokémon speed Distribution")
```



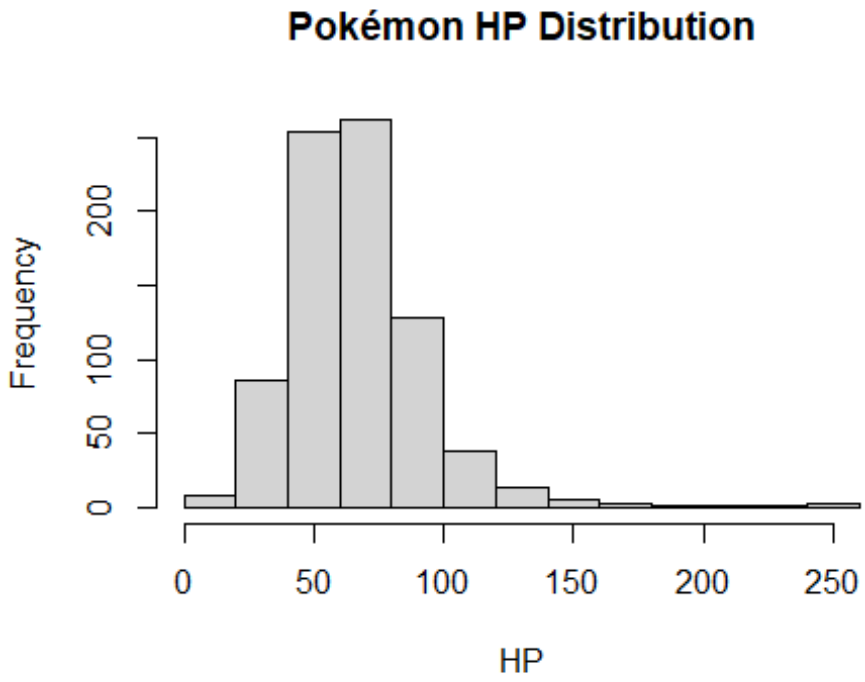
#The below command creates a bar plot to display the count of Pokémon for each generation. The x-axis represents the generations, and the y-axis represents the count. It has labeled axes for generation and count, and a title stating “Pokémon Generations”.

```
barplot(table(pokemon$generation),
        xlab = "Generation",
        ylab = "Count",
        main = "Pokémon Generations")
```



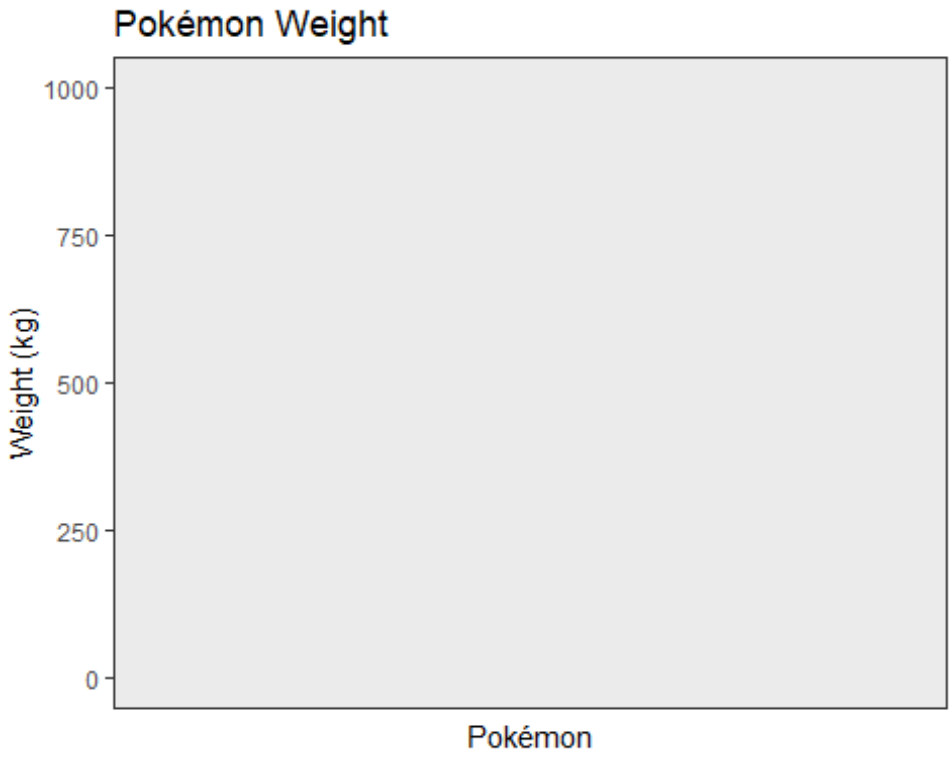
#The below command generates a histogram to visualize the distribution of Pokémon's HP (Hit Points). The x-axis represents the HP values, and the y-axis represents the frequency of occurrence. It has labeled axes for HP and frequency, and a title stating "Pokémon HP Distribution".

```
hist(pokemon$hp,
     xlab = "HP",
     ylab = "Frequency",
     main = "Pokémon HP Distribution")
```



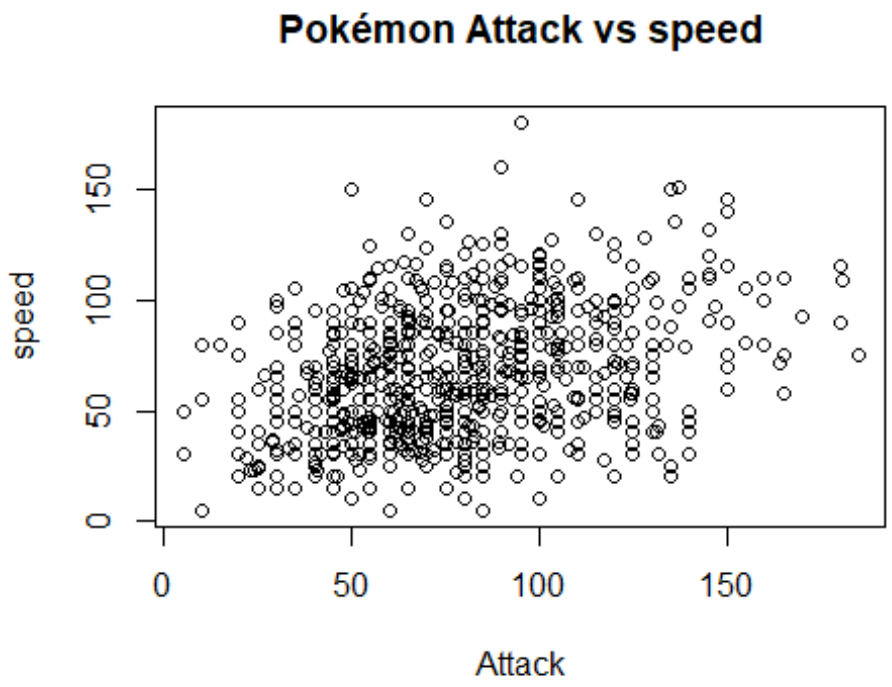
#The below command generates a line plot to visualize the weights of Pokémon. Each line represents the weight of a Pokémon over its name. The plot has a title, x-label for Pokémon names, y-label for weight in kilograms, and a black and white theme. The x-axis labels and ticks are hidden.

```
pokemon %>%
  ggplot(aes(x = name, y = weight_kg)) +
  geom_line(color = "#FF4081FF") +
  ggtitle("Pokémon Weight") +
  xlab("Pokémon") +
  ylab("Weight (kg)") +
  theme_bw() +
  theme(axis.text.x = element_blank(), axis.ticks.x = element_blank())

## Warning: Removed 20 rows containing missing values (`geom_line()`).

## `geom_line()`: Each group consists of only one observation.
## ℹ Do you need to adjust the group aesthetic?
```



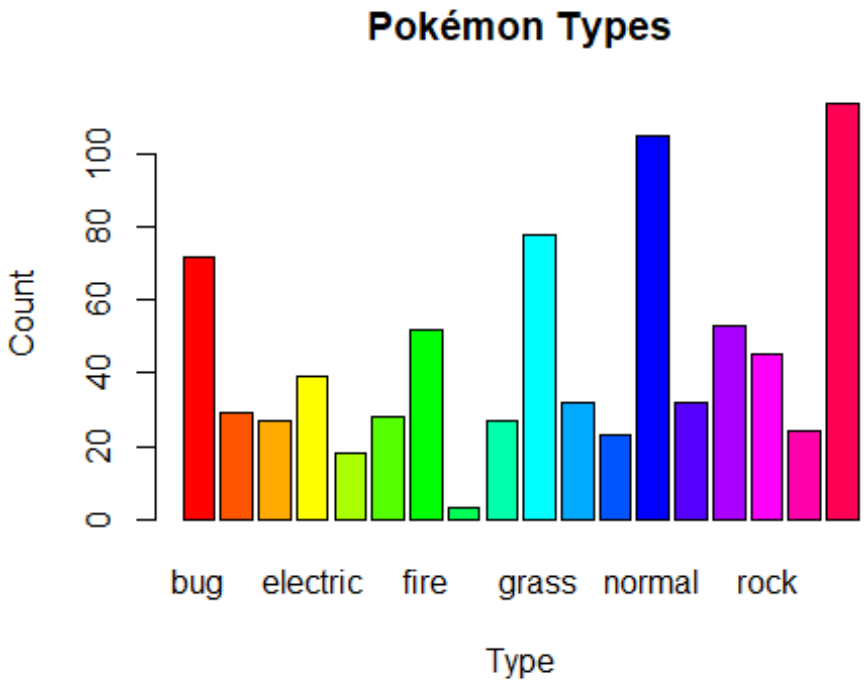
#The below command creates a scatter plot to compare the Pokémon's attack and speed attributes. The x-axis represents the attack values, and the y-axis represents the speed values. It has labeled axes for attack and speed, and a title stating "Pokémon Attack vs Speed".

```
plot(pokemon$attack, pokemon$speed,
     xlab = "Attack",
     ylab = "speed",
     main = "Pokémon Attack vs speed")
```



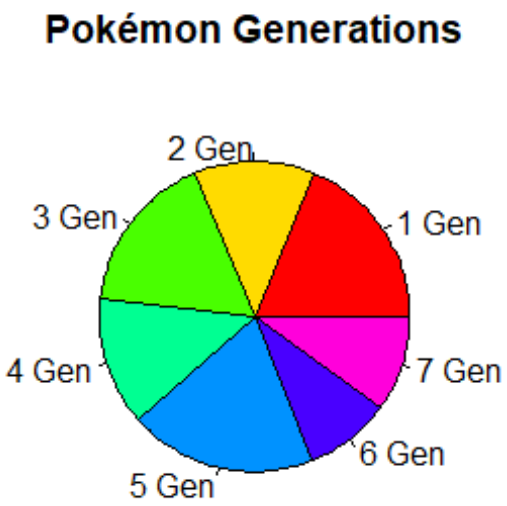
#The below command creates a bar plot to display the count of Pokémon for each type. The x-axis represents the Pokémon types, and the y-axis represents the count. It has labeled axes for type and count, and a title stating “Pokémon Types”. Additionally, it uses a rainbow color palette to differentiate the bars representing different types.

```
type_counts <- table(pokemon$type1)
barplot(type_counts,
        xlab = "Type",
        ylab = "Count",
        main = "Pokémon Types",
        col = rainbow(length(type_counts)))
```



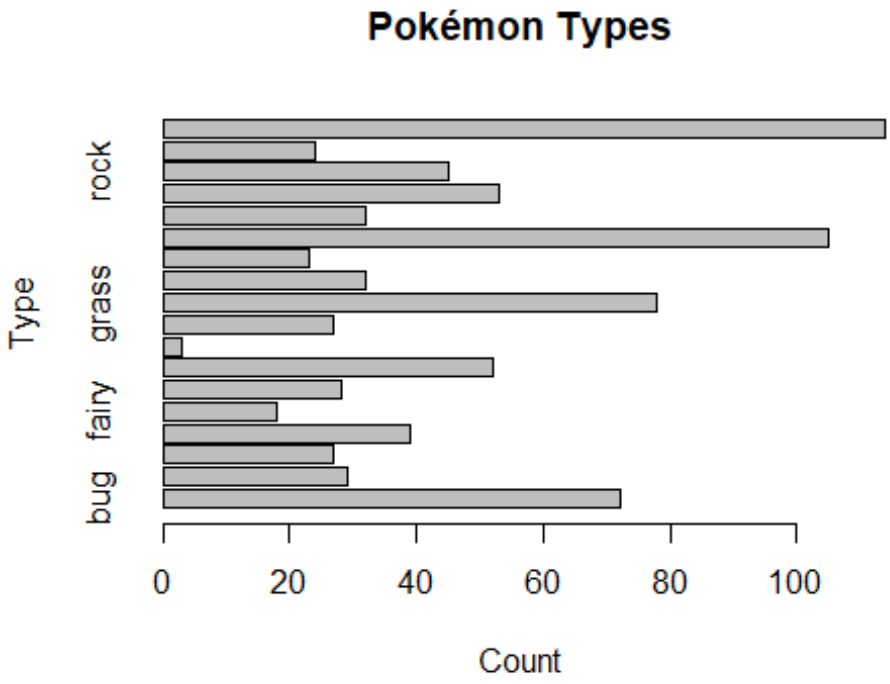
#The below command generates a pie chart to visualize the distribution of Pokémon generations. Each slice of the pie represents a generation, and the size of each slice is proportional to the count of Pokémon in that generation. It has a title stating “Pokémon Generations”. The colors of the slices are determined by the rainbow color palette, and the labels on the pie slices are a combination of generation names and the term “Gen”.

```
generation_counts <- table(pokemon$generation)
pie(generation_counts,
    main = "Pokémon Generations",
    col = rainbow(length(generation_counts)),
    labels = paste(names(generation_counts), "Gen"))
```



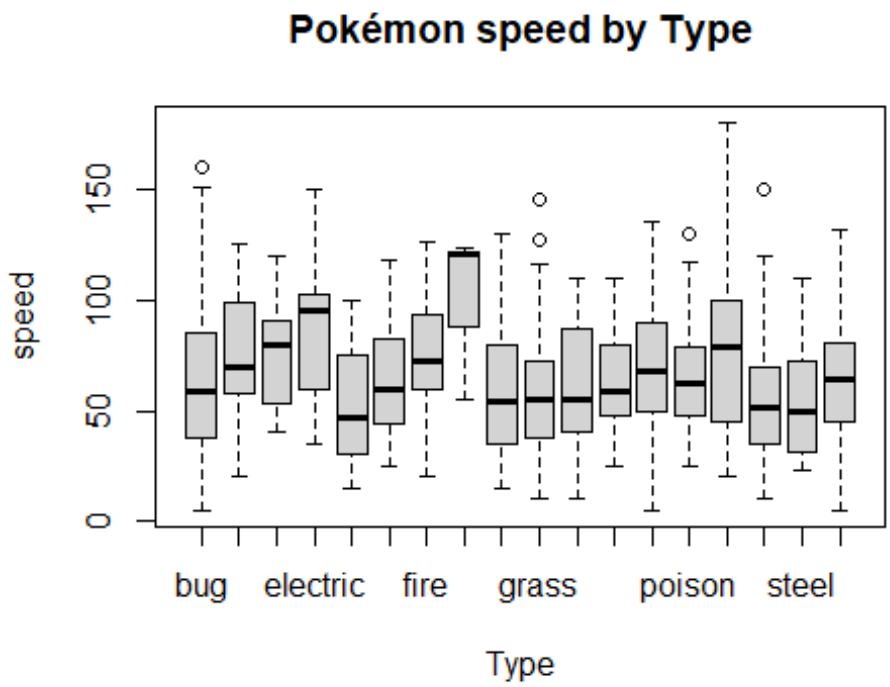
#The below command creates a horizontal bar plot to display the count of Pokémon for each type. The y-axis represents the types, and the x-axis represents the count. The plot has a title stating “Pokémon Types” and labeled axes for count and type.

```
barplot(table(pokemon$type1),
        horiz = TRUE,
        xlab = "Count",
        ylab = "Type",
        main = "Pokémon Types")
```



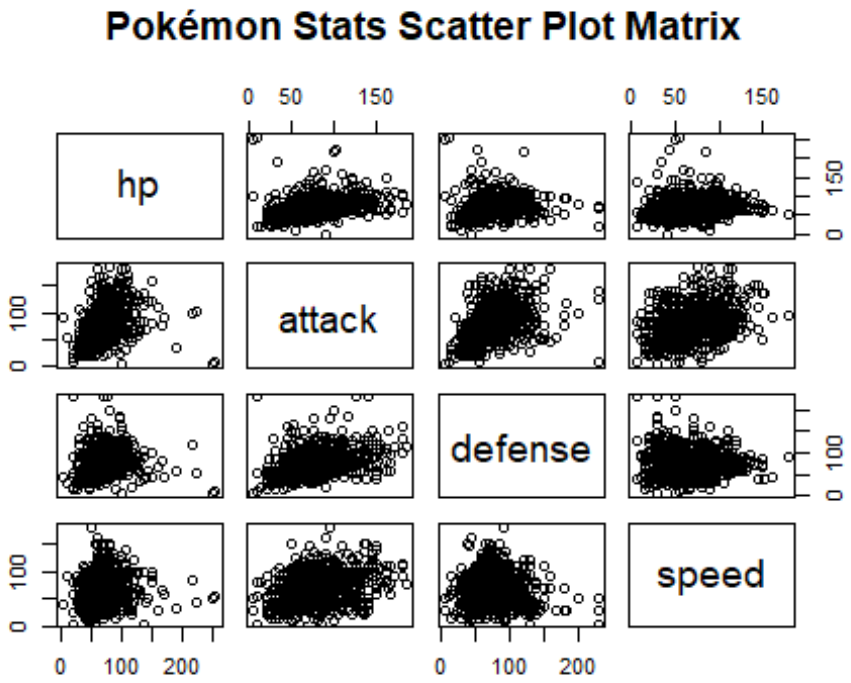
#The below command generates a box plot to compare the speed of Pokémon across different types. The x-axis represents the types, and the y-axis represents the speed values. It has a title stating “Pokémon speed by Type” and labeled axes for type and speed.

```
boxplot(speed ~ type1, data = pokemon,
        xlab = "Type",
        ylab = "speed",
        main = "Pokémon speed by Type")
```



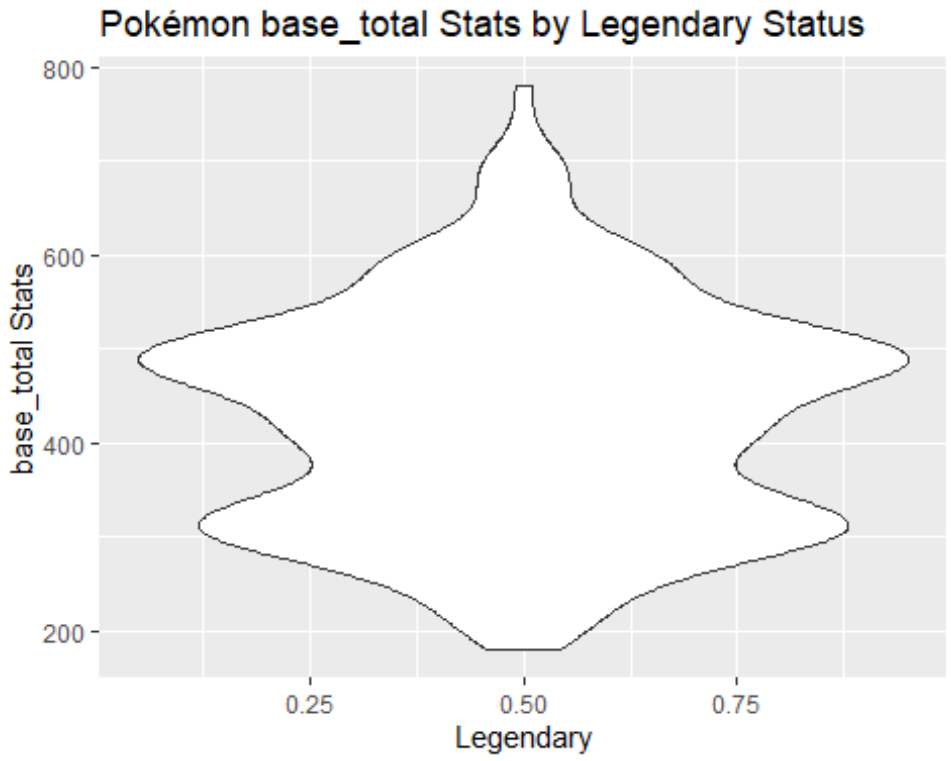
#The below command creates a scatter plot matrix to visualize the relationships between different Pokémon stats (hp, attack, defense, and speed). Each scatter plot represents the relationship between two stats. The plot has a title stating “Pokémon Stats Scatter Plot Matrix”.

```
pairs(pokemon[, c("hp", "attack", "defense", "speed")],
      main = "Pokémon Stats Scatter Plot Matrix")
```



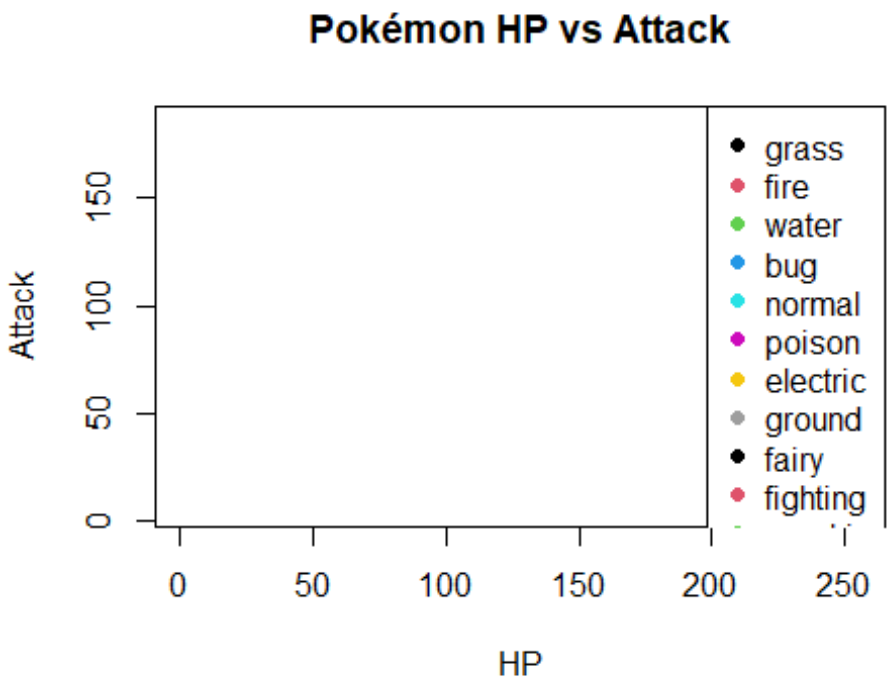
#The below command uses the ggplot library to create a violin plot, showing the distribution of base_total stats of Pokémon based on their legendary status. The x-axis represents legendary status, and the y-axis represents base_total stats. It has a title stating "Pokémon base_total Stats by Legendary Status" and labeled axes for legendary and base_total stats.

```
ggplot(pokemon, aes(x = is_legendary, y = base_total)) +
  geom_violin() +
  xlab("Legendary") +
  ylab("base_total Stats") +
  ggtitle("Pokémon base_total Stats by Legendary Status")
```



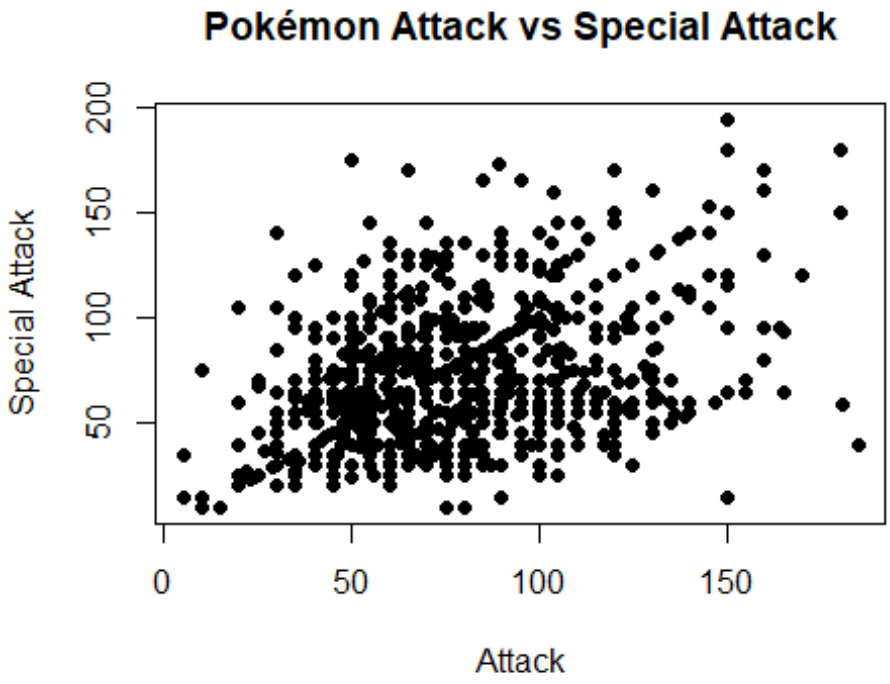
#The below command creates a scatter plot to compare the HP (Hit Points) and attack attributes of Pokémon. The x-axis represents HP values, and the y-axis represents attack values. Each point in the plot is colored based on the numeric representation of the Pokémon's type. It has labeled axes for HP and attack, and a title stating "Pokémon HP vs Attack". The points are represented by a solid circle symbol (pch = 16). The legend is added to the top right corner, displaying unique Pokémon types with corresponding colors and symbols.

```
plot(pokemon$hp, pokemon$attack,
     col = as.numeric(pokemon$type1),
     xlab = "HP",
     ylab = "Attack",
     main = "Pokémon HP vs Attack",
     pch = 16)

## Warning in plot.xy(xy, type, ...): NAs introduced by coercion

legend("topright", legend = unique(pokemon$type1),
       col = 1:length(unique(pokemon$type1)), pch = 16)
```



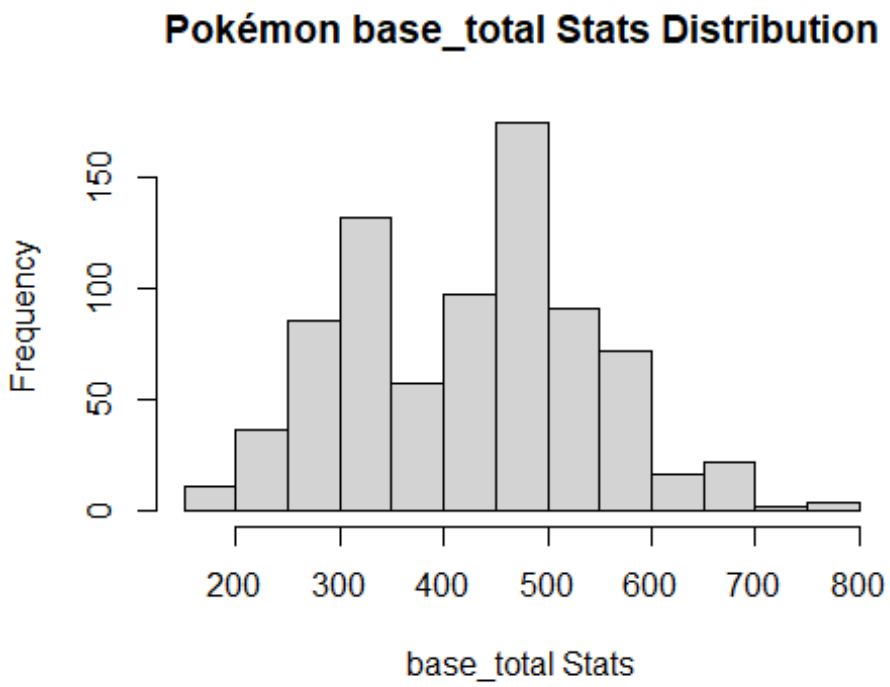
#The below command creates a scatter plot to compare the attack and special attack attributes of Pokémon. The x-axis represents attack values, and the y-axis represents special attack values. Each point in the plot is represented by a solid circle symbol (pch = 16). It has a title stating "Pokémon Attack vs Special Attack" and labeled axes for attack and special attack.

```
plot(pokemon$attack, pokemon$sp_attack,
     xlab = "Attack",
     ylab = "Special Attack",
     main = "Pokémon Attack vs Special Attack",
     pch = 16)
```



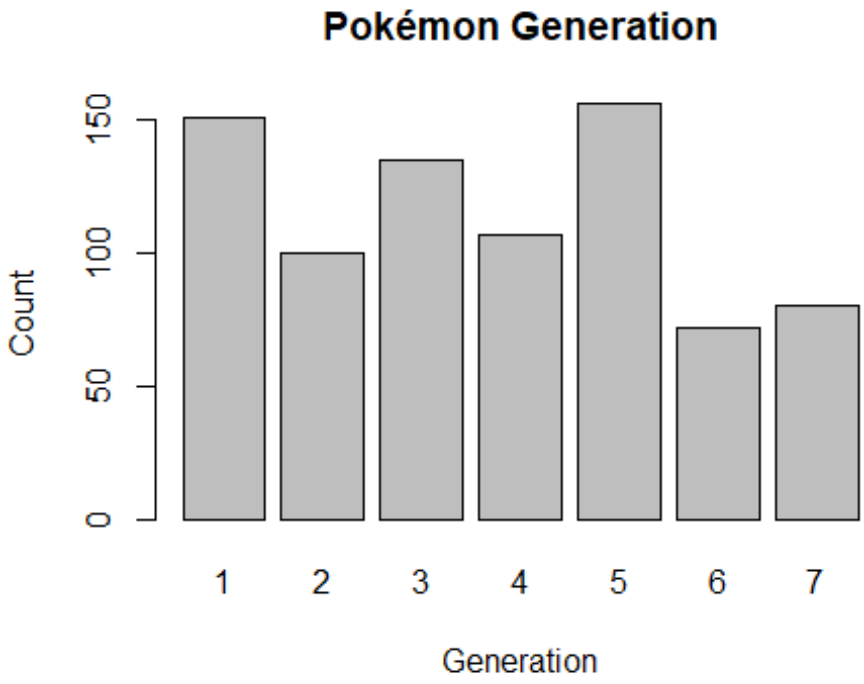
#The below command generates a histogram to visualize the distribution of base_total stats among Pokémon. The x-axis represents the base_total stats, and the y-axis represents the frequency of occurrence. It has a title stating "Pokémon base_total Stats Distribution" and labeled axes for base_total stats and frequency.

```
hist(pokemon$base_total,
     xlab = "base_total Stats",
     ylab = "Frequency",
     main = "Pokémon base_total Stats Distribution")
```



#The below command generates a bar plot to display the count of Pokémon for each generation. The x-axis represents the generations, and the y-axis represents the count. It has a title stating “Pokémon Generation” and labeled axes for generation and count.

```
barplot(table(pokemon$generation),
        xlab = "Generation",
        ylab = "Count",
        main = "Pokémon Generation")
```


Pokémon Generation
Count
0
50
100
150
1
2
3
4
5
6
7
Generation