Untitled

2023-10-09

Import Libraries & Dataset

```
# Load required libraries
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
```

```
library(tidyr)
library(caret)
```

```
## Loading required package: lattice
```

```
library(kableExtra)
```

```
##
## Attaching package: 'kableExtra'
```

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

```
library(reshape2)
```

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

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

```
# Load Sales data
sales_data <- read.csv('turtle_sales.csv')

# Load Reviews data
reviews_data <- read.csv('turtle_reviews.csv')
```

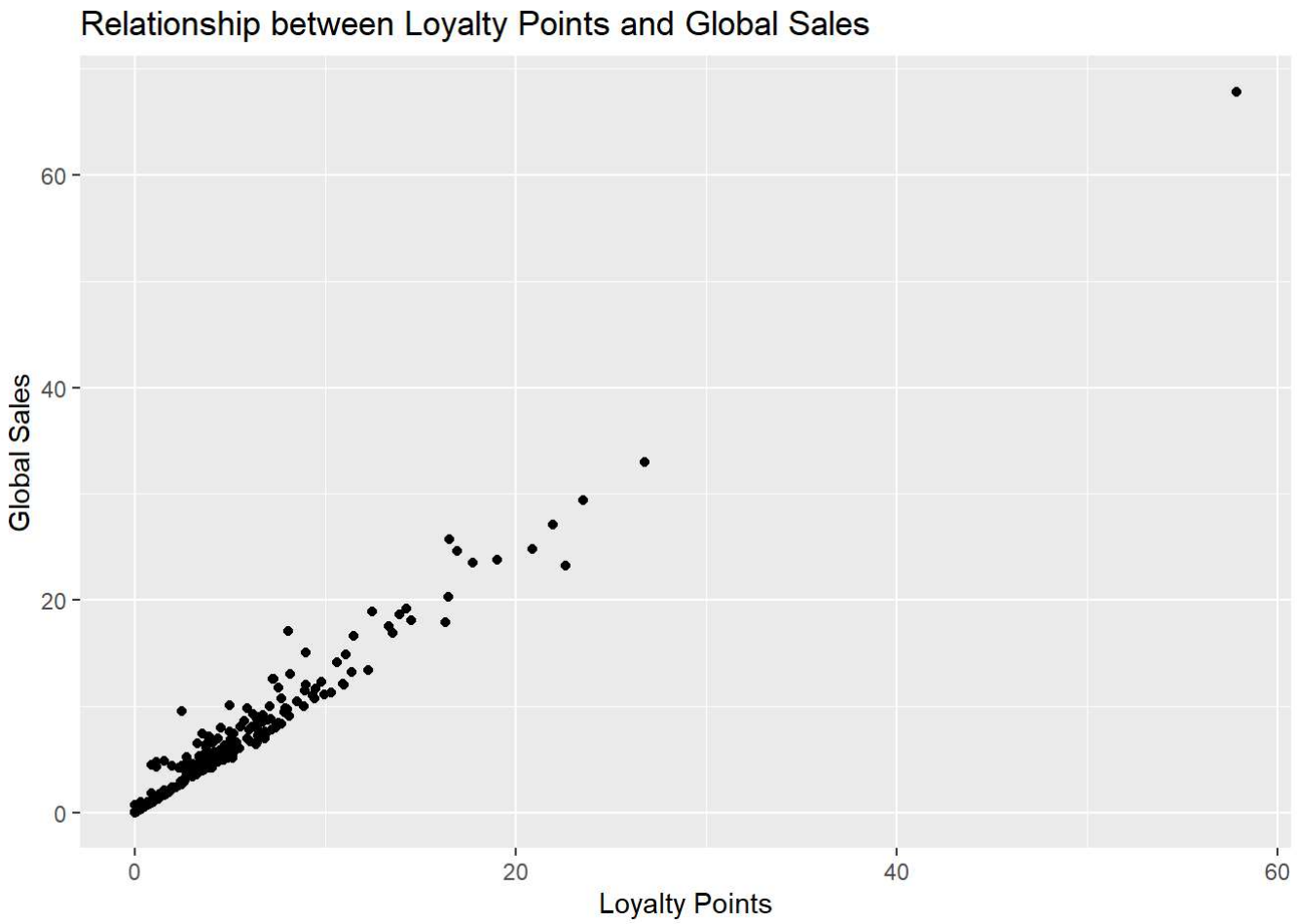
Cleaning **Dataset**

```
# Data cleaning
# Remove rows with missing values in sales_data and reviews_data
sales_data <- sales_data[complete.cases(sales_data), ]
reviews_data <- reviews_data[complete.cases(reviews_data), ]
```

1. How customers accumulate loyalty points:

```
# 1. How customers accumulate loyalty points
loyalty_points <- sales_data$na_sales + sales_data$eu_sales

# Visualize the relationship between sales and loyalty points
ggplot(data = sales_data, aes(x = loyalty_points, y = sales_data$global_sales)) +
  geom_point() +
  xlab('Loyalty Points') +
  ylab('Global Sales') +
  ggtitle('Relationship between Loyalty Points and Global Sales')
```



2. How groups within the customer base can be used to target

specific market segments:

```
# 2. How groups within the customer base can be used to target specific market segments
merged_data <- merge(sales_data, reviews_data, by.x = 'product', by.y = 'product', all.x = TRUE)

# Prepare features and target variable
X <- merged_data %>% select(na_sales, eu_sales)
y <- merged_data$loyalty_points

# Split the data into training and testing sets
set.seed(42)
train_index <- createDataPartition(y, p = 0.8, list = FALSE)
X_train <- X[train_index, ]
X_test <- X[-train_index, ]
y_train <- y[train_index]
y_test <- y[-train_index]

# Create a linear regression model
model <- lm(y_train ~ na_sales + eu_sales, data = data.frame(X_train, y_train))

# Make predictions
predictions <- predict(model, newdata = data.frame(X_test))

# Evaluate the model
accuracy <- cor(predictions, y_test)
mse <- mean((predictions - y_test)^2)

# Print accuracy and MSE
print(paste('Accuracy:', accuracy))
```

```
## [1] "Accuracy: 0.118416401860611"
```

```
print(paste('Mean Squared Error:', mse))
```

```
## [1] "Mean Squared Error: 1677404.22900061"
```

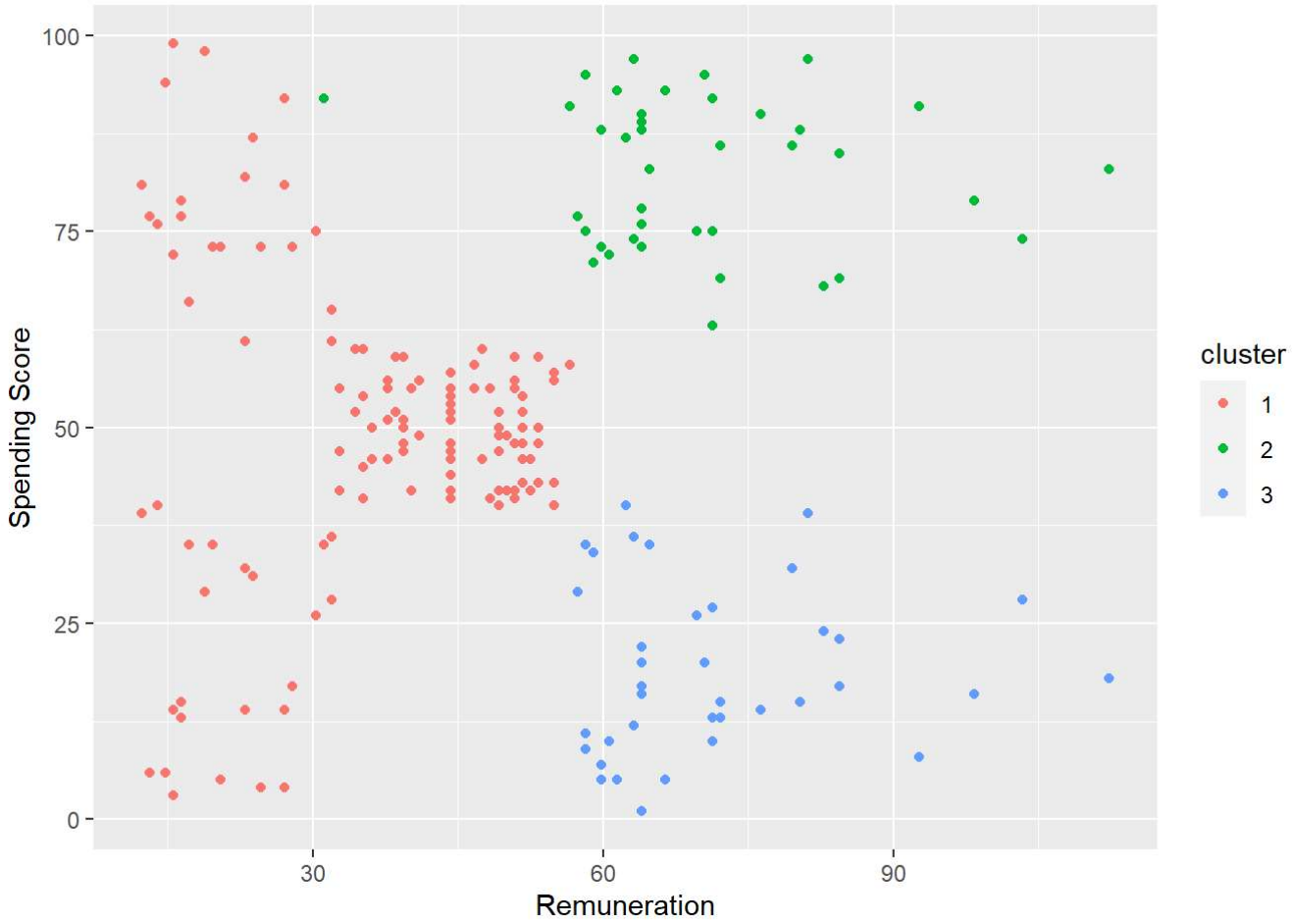
3. How social data (e.g. customer reviews) can be used to

inform marketing campaigns:

```
# 3. How social data can be used to inform marketing campaigns
k <- 3
kmeans_model <- kmeans(reviews_data %>% select(remuneration, spending_score), centers = k, nstar
t = 10)
reviews_data$cluster <- as.factor(kmeans_model$cluster)

# Explore clusters and analyze customer segments
cluster_0 <- subset(reviews_data, cluster == 0)
cluster_1 <- subset(reviews_data, cluster == 1)
cluster_2 <- subset(reviews_data, cluster == 2)

# Visualize clusters
ggplot(reviews_data, aes(x = remuneration, y = spending_score, color = cluster)) +
  geom_point() +
  xlab('Remuneration') +
  ylab('Spending Score')
```

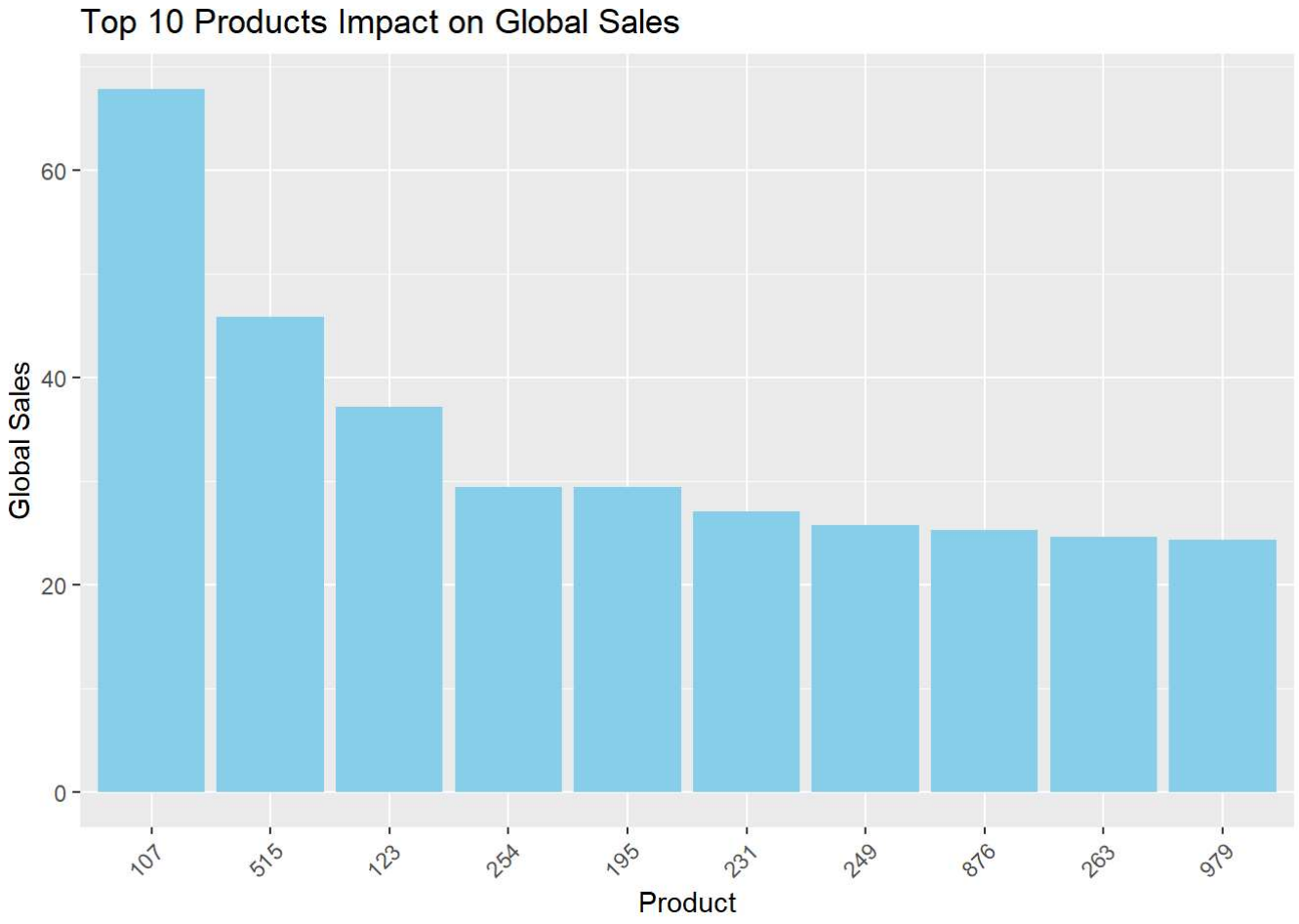


4. The impact that each product has on sales:

```
# 4. The impact that each product has on sales
top_10_products <- sales_data %>%
  group_by(product) %>%
  summarise(total_sales = sum(global_sales)) %>%
  arrange(desc(total_sales)) %>%
  top_n(10)
```

```
## Selecting by total_sales
```

```
# Visualize the impact of the top 10 products on sales
ggplot(top_10_products, aes(x = reorder(product, -total_sales), y = total_sales)) +
  geom_bar(stat = 'identity', fill = 'skyblue') +
  xlab('Product') +
  ylab('Global Sales') +
  ggtitle('Top 10 Products Impact on Global Sales') +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



5. How reliable the data is (e.g. normal distribution, skewness,

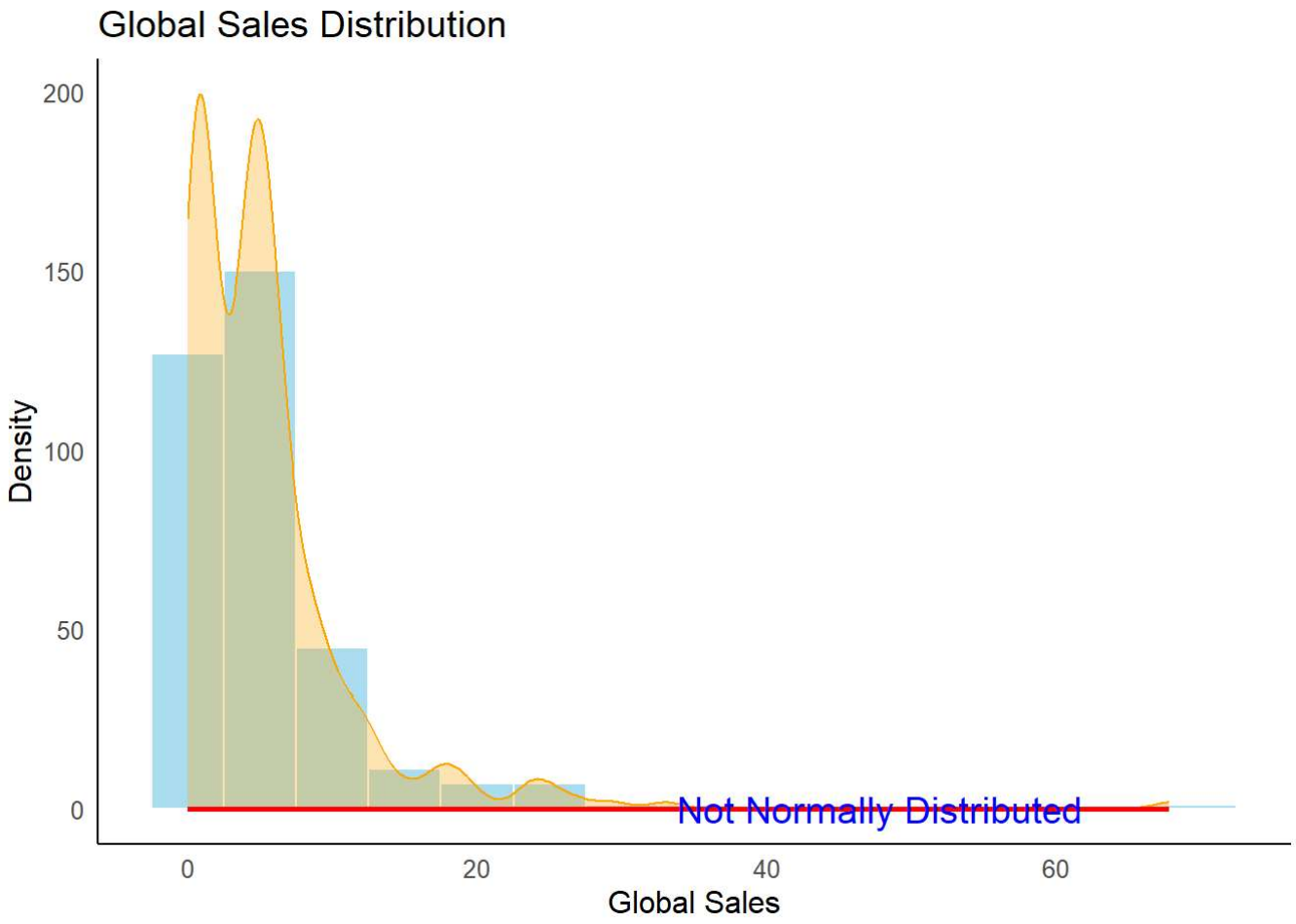
or kurtosis):

```
# 5. How reliable the data is
# Perform Shapiro-Wilk test for normality
shapiro_test <- shapiro.test(sales_data$global_sales)

# Check the results
if (shapiro_test$p.value > 0.05) {
  distribution <- "Normally Distributed"
} else {
  distribution <- "Not Normally Distributed"
}

# Plot histogram and normal distribution curve
ggplot(sales_data, aes(x = global_sales)) +
  geom_histogram(binwidth = 5, fill = 'skyblue', color = 'white', alpha = 0.7) +
  geom_density(aes(y = ..count.. * 5), color = 'orange', fill = 'orange', alpha = 0.3) +
  stat_function(fun = dnorm, args = list(mean = mean(sales_data$global_sales), sd = sd(sales_dat
a$global_sales)), color = 'red', linewidth = 1) +
  labs(x = 'Global Sales', y = 'Density', title = 'Global Sales Distribution') +
  theme_minimal() +
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        axis.line = element_line(colour = "black"), text = element_text(size = 12)) +
  annotate("text", x = max(sales_data$global_sales) - 20, y = 0.03, label = distribution, color
= "blue", size = 5)
```

```
## Warning: The dot-dot notation (`..count..`) was deprecated in ggplot2 3.4.0.
## ℹ Please use `after_stat(count)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




```
# Print Shapiro-Wilk test results
print(paste("Shapiro-Wilk Test p-value: ", format(shapiro_test$p.value, scientific = FALSE)))
```

```
## [1] "Shapiro-Wilk Test p-value:  0.000000000000000000000003951602"
```

6. What the relationship(s) is/are (if any) between North

American, European, and global sales?

```
# 6. Relationship(s) between North American, European, and global sales
correlation_matrix <- cor(sales_data[c('na_sales', 'eu_sales', 'global_sales')])

# Reshape the correlation matrix for better visualization
correlation_melted <- melt(correlation_matrix)

# Create a heatmap using ggplot2
ggplot(data = correlation_melted, aes(x = Var1, y = Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradient2(low = "blue", mid = "white", high = "red",
                       midpoint = 0, name = "Correlation") +
  labs(title = 'Correlation Matrix between Sales Regions') +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  coord_fixed() +
  scale_x_discrete(name = "Sales Region") +
  scale_y_discrete(name = "Sales Region") +
  theme(legend.position = "bottom") +
  geom_text(aes(label = round(value, 2)), vjust = 1)
```