Assignment: Predicting Future Outcomes

Course 3 Final Project Report

DECEMBER 23, 2022

NAME: ABHISHEK PILLAI

Contents

I. INTRODUCTION

Turtle Games is a game manufacturer and retailer with a worldwide customer base. It produces and merchandises its personal commodities in addition to sourcing and selling other companies' products. In this manner, the product range of Turtle Games includes video games, books, toys, and board games, with Turtle Games continuously gathering data from the sales made and the customer reviews made. As such, Turtle Games has an aim of enhancing the overall sales performance through the exploration of the obtained data illustrating customer trends.

II. ANALYTICAL APPROACH

II. A. Python

II. A. 1. Data Cleaning in Python

Pythonic analysis using the "turtle_reviews.csv" was initially completed, with the csv file being imported as reviews using the read_csv() method. Afterwards, the "reviews" dataframe was sense checked using reviews.head(), reviews.columns, reviews.shape and reviews.dtypes. Then, the language and platform columns were also dropped before the remuneration(k£) and spending score (1 – 100) were renamed, with boxplots and user defined functions then being used to remove outliers within the 'loyalty_points', 'spend_score', 'age', 'product' and 'remuneration' columns, with an example of the outlier user defined function used being displayed in figure 1 of the appendix.

II. A. 2 Data Analysis in Python

After the data was cleaned, a linear regression analysis comparing loyalty_points against spend_score, remuneration & age respectively was initially undertaken, but it sould be noted that since none of the raw graphs of the 3 stated variable against loyalty_points displayed homoscedasticity when the Breusch – Pagan Test was completed on these initial plots, the loyalty_points values needed to be transformed for each respective graph as homoscedasticity is a primary assumption made when completing a regression analysis.

Next, the k – means clustering between remuneration and spend score was observed to better understand the usefulness of the remuneration and spend_score data obtained, which helps identification of groups within the customer base that can be used to target specific market segments. Therefore, elbow and silhouette methods were employed to observe where the optimal number of clusters to be used, with the clusters' optimal number being determined by the elbow feature of the elbow method and the peak of the silhouette. Thus, the remuneration and spend_score data could then be plotted using the optimal K – means value.

The final pythonic analysis undertaken was to identify the 15 most common words as well as observing the most positive and negative reviews received on social media via the use of NLP. This led to the creation of word clouds with and without stop words as well as frequency distributions for all reviews and summaries in the dataframe, through the act of ensuring that everything was in lowercase, removing the punctuation and tokenising the reviews and summaries. Thus, the frequency distributions created without the presence of stop words were plotted as bar plots to visualise the most common words

Afterwards, the polarity of each review and summary was obtained via a user defined function, with the values obtained being plotted in 2 respective histograms as well as being utilised to identify the top 20 most positive and negative reviews.

Note that all the important libraries, functions and examples of the codes used in the pythonic analysis can be found in figures 1 – 10 of the appendix.

II. B. R

II. B. 1. Data Cleaning in R

The next part of the analysis involved the use of R and the "turtle_sales.csv" file. Therefore, the "turtle_sales.csv" file was initially imported as data_TS into R using read.csv(), with the 'Ranking', 'Year', 'Genre' and 'Publisher' columns then being dropped using the subset method. Each sales column was then analysed for outliers via the use of boxplots, with the identified outliers then being removed. Thus, the data was cleaned sufficiently for further analysis to be undertaken.

II. B. 2. Data Analysis in R

After the data had been cleaned in R, an initial exploration of the data was undertaken using qplot to create scatterplots, histograms and histograms to show the variation of NA_Sales, EU_Sales and Global_Sales with respect to Product the ID.

Next, a further exploration of the data and its reliability was explored.Therefore, a new dataframe (Product_Sales) was created by grouping the data in the final dataframe created in week 4 (data_TS_2c) by the Product ID, with each of the sales columns being summed by Product ID (thus creating a NA_Sales_Sum, EU_Sales_Sum and Global_Sales_Sum column instead of the sales columns originally present), whilst the 'Platform' column was dropped, with further boxplots, scatterplots and boxplots created to view the variation of each Sales Sum with product ID via ggplot2. Afterwards, an initial analysis of the normality of the Sales Sum data was completed through the plotting of individual Q-Q Plots for each Sales Sum column, followed by an assessment of the skewness and kurtosis of each separate Sales Sum column.

Finally, the possible relationships between NA_Sales, EU_Sales and Global_Sales was completed using simple and multiple linear regression, after the correlation between the 3 Sales_Sum columns was determined. Through this it was then possible to make comparisons between the predicted Global Sales Value and the actual Global Sales value obtained.

Note that all the important libraries, functions and examples of the codes used in the R analysis can also be found in figures 11 – 16 of the appendix.

III. VISUALISTIONS AND INSIGHTS

III. A. Linear Regression Analysis of Loyalty Points Via Python

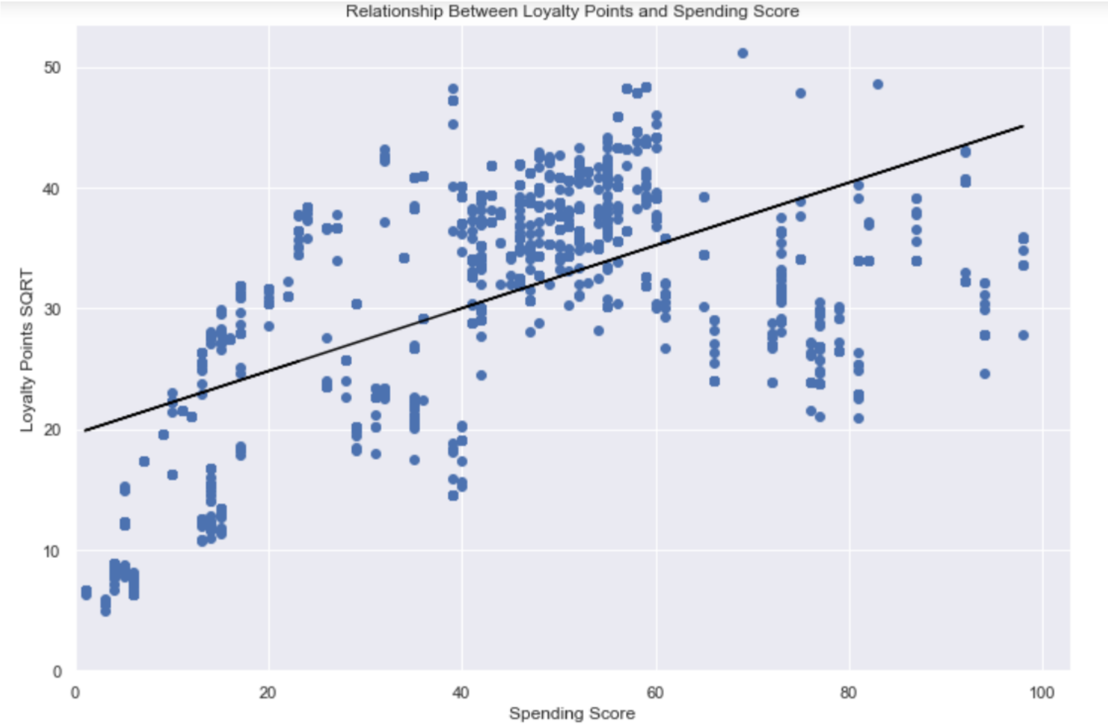


Figure I. Linear Regression Plot of the Square Root of Loyalty Points Against Spending Score

Figure I show the relationship between the loyalty points and spending score, with the square root of loyalty points used instead of the raw loyalty points values as loyalty points vs. spending score did not produce a graph displaying homoscedasticity whilst this one did (p value = 0.805). Additionally, the graph plotted shows strong positive correlation between the square root of loyalty points and spending score, although the adjusted R^2 value is relatively small (0.339), but this can be attributed to the fact that loyalty points has had to be transformed to produce this plot.

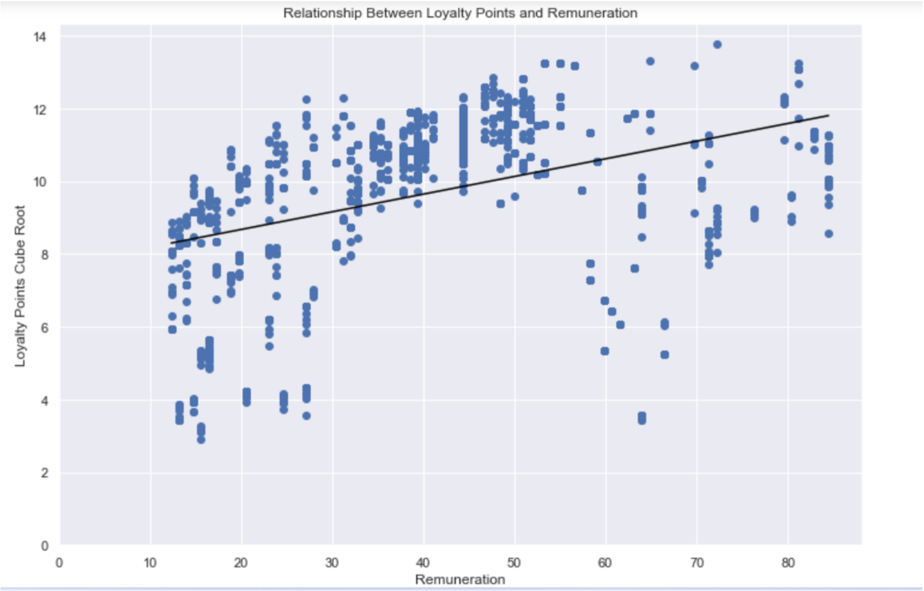


Figure II. Linear Regression Plot of the Cube Root of Loyalty Points Against Remuneration

Figure II shows the relationship between the loyalty points and remuneration, with the cube root of loyalty points used instead of the raw loyalty points values as loyalty points vs. remuneration did not produce a graph displaying homoscedasticity whilst this one did (p value = 0.581). Additionally, the graph plotted shows strong positive correlation between the cube root of loyalty points and remuneration, although the adjusted R^2 value is rather small (0.150), but this can be attributed to the fact that loyalty points has had to be transformed to produce this plot.

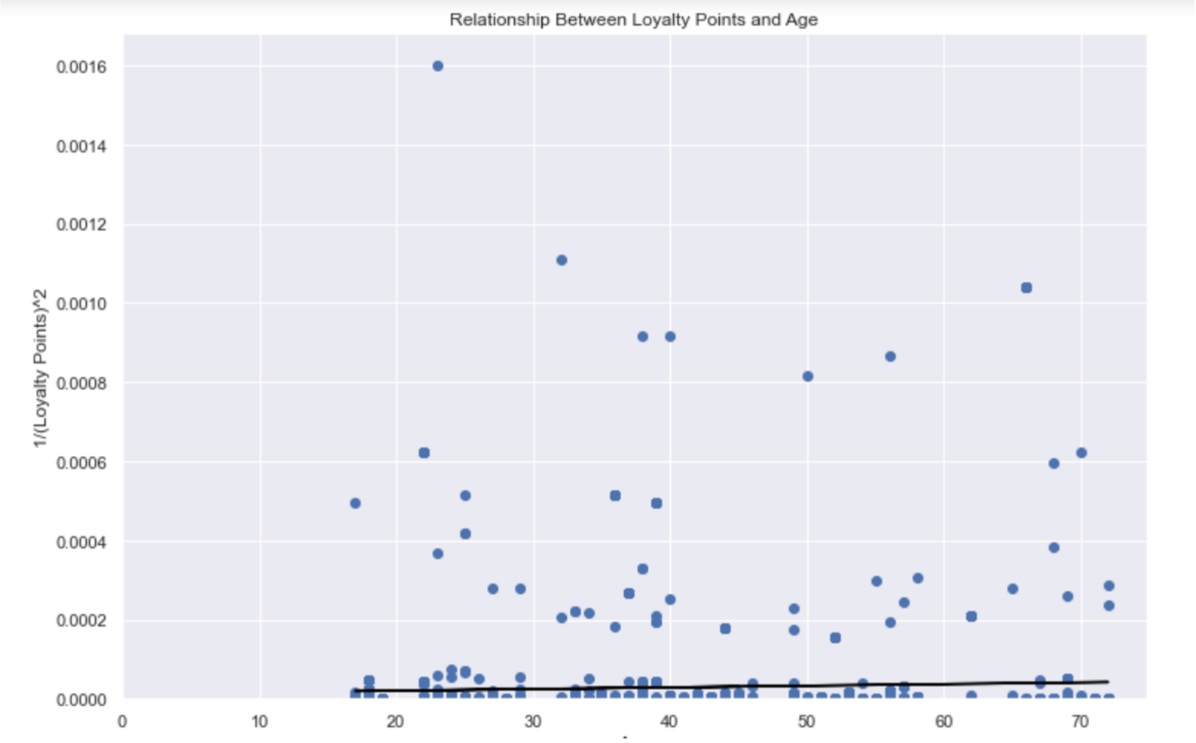


Figure III. Linear Regression Plot of 1/(Loyalty Points)^2 Against Remuneration

Figure III shows the relationship between the loyalty points and age. As can be seen, the reciprocal of loyalty points squared is used instead of the raw loyalty points values as loyalty points vs. age did not produce a graph displaying homoscedasticity whilst this one did, (p value = 0.0846). Additionally, the graph plotted shows very weak positive correlation between the reciprocal of loyalty points squared and age, although the adjusted R^2 value is extremely small (0.002) indicating that the graph may not be reliable, even if the loyalty points transformation is considered.

III. B. K – Means Clustering Analysis of Remuneration and Spending Score Via Python

III.B. 1. Elbow And Silhouette Plots

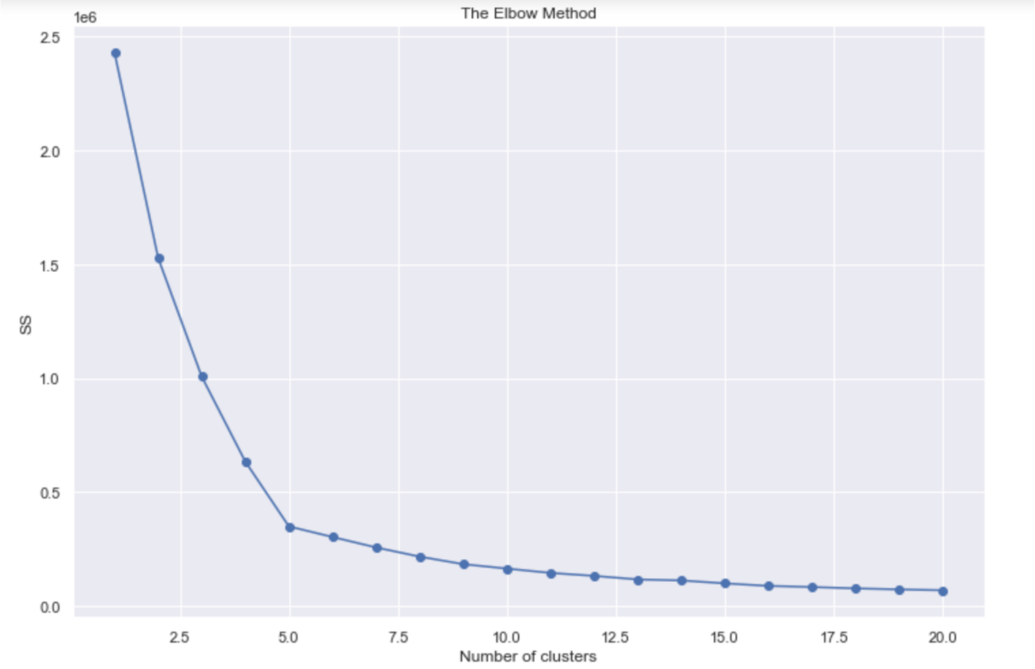


Figure IV. Elbow Method Plot

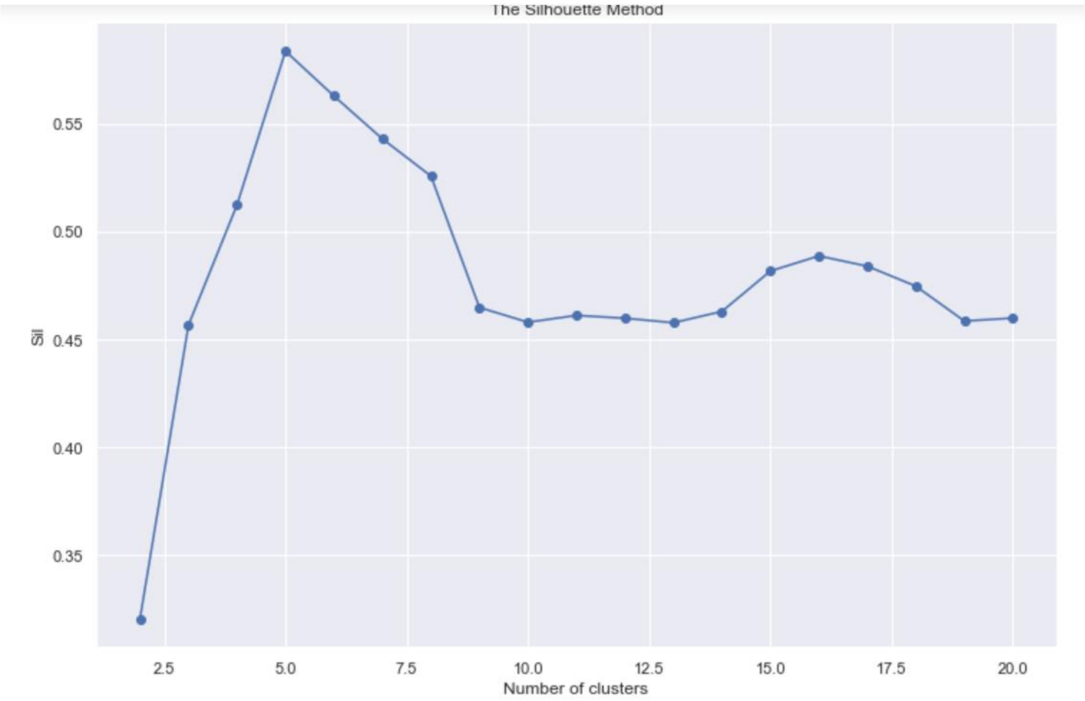


Figure V.

Silhouette Method Plot

Elbow and silhouette methods are methods that used to find the optimal number of clusters to be used in K – means clustering of the data set. In the case of the elbow method, the optimal number of clusters is denoted by the elbow feature whilst for the silhouette method, the optimal number of clusters exists at the peak Si value. Considering this, figures IV and V shows that the optimal number of clusters is 5.

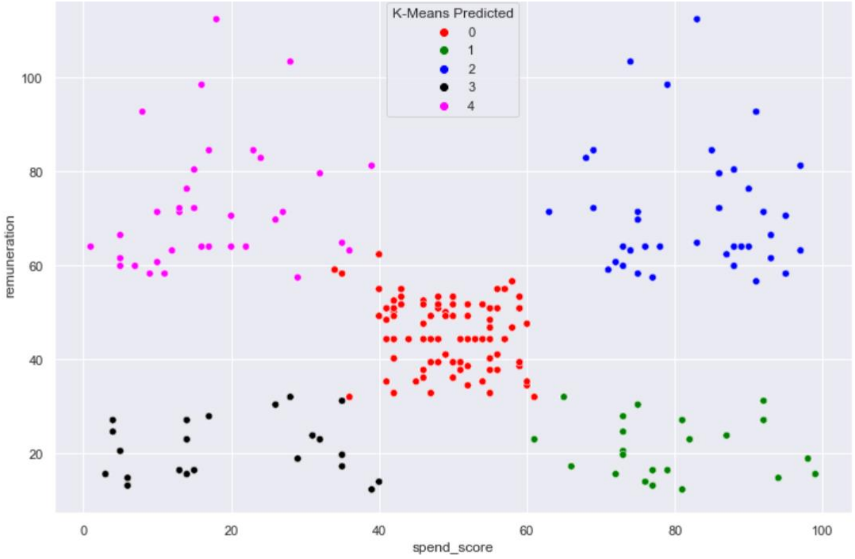


Figure VI. Final Fit Model for Remuneration Vs Spending Score

Figure VI shows the final fit K – means clustering model created for remuneration against spending score. It can be noted that some data points of a particular cluster may be in very close proximity to other clusters. For example, there are couple of points in the central red cluster which lie very close to the magenta and black clusters. Additionally, some data points within a particular cluster appear to lie in a position far away from other data points within the same cluster. However, attempting to break these clusters down into smaller clusters would make the final model less accurate as a cluster number greater than 5 would result in fewer points being put in the correct cluster.

III. C. NLP Analysis of Reviews and Summaries Via Python

III. C. 1. Common Words: Reviews

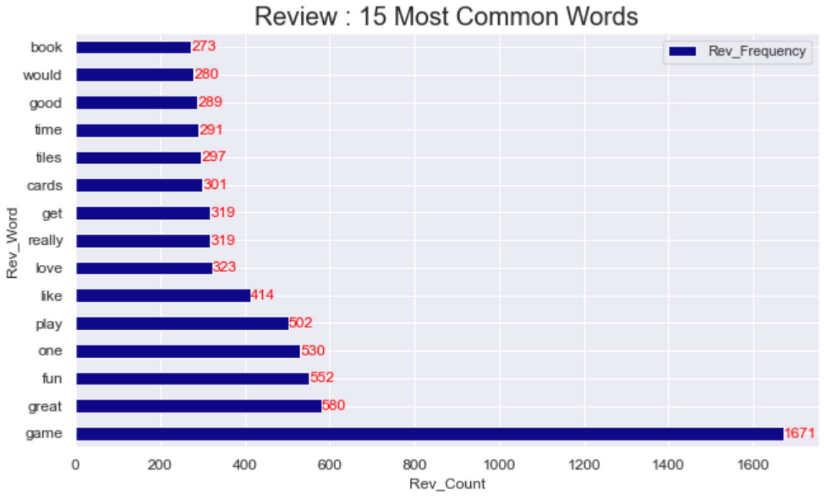


Figure VII. Review Frequency Distribution Bar Plot Without Stop Words

Figure VII shows the final frequency distribution bar plot created to identify the most common words in the reviews, with most of these pertaining to positive descriptive words such as great, love and good, with some of the other common words related to the goods sold such as game (which is the most common word, occurring with a frequency of approximately 2.9 times as much as the next most common word, great) and book. One other point to note would be that only the 5 most common words all have occurrence values of greater than 500, whilst the most common word (i.e., game) is the only word to occur over 1000 times.

III. C. 2. Common Words: Summaries

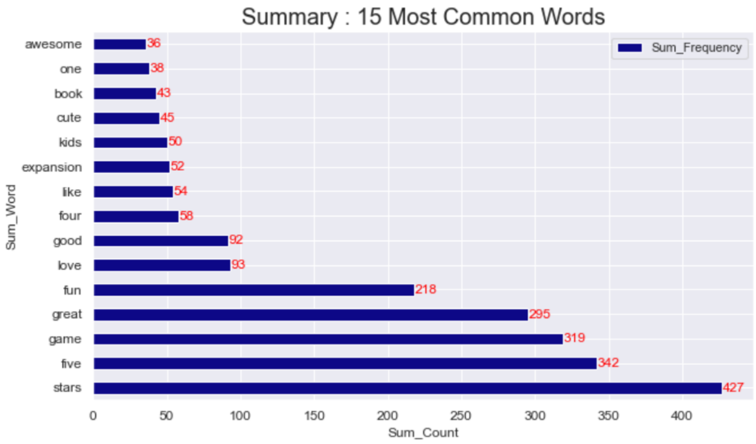


Figure VIII. Summary Frequency Distribution Bar Plot Without Stop Words

Figure VIII shows the final frequency distribution bar plot created to identify the most common words in the summaries, where stars were the most common word. Five being the next most common word afterwards, with stars exceeding the occurrence of five by a value of only 85. Apart from this the most common words for summary are also positive descriptive words, but unlike the reviews, no singular word exceeded 500, with stars (the most common word) only achieving an occurrence of 427.

III. C. 3. Sentiment Scores: Reviews and Summaries

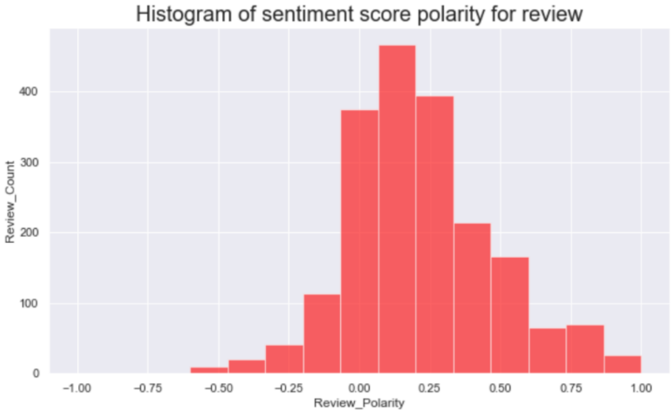


Figure IX. Sentiment Score Histogram - Reviews

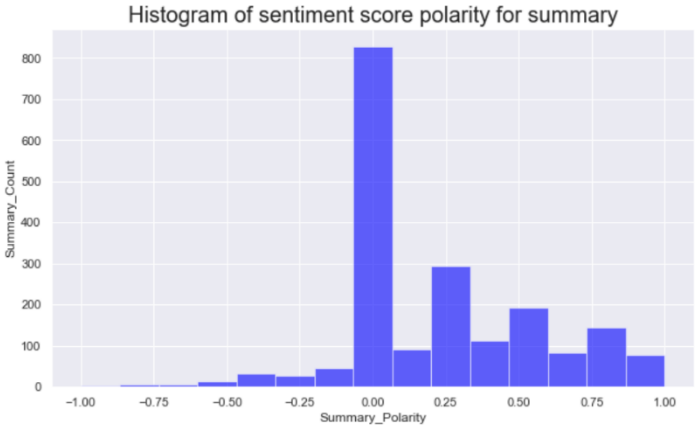


Figure X. Sentiment Score Histogram - Summaries

Figures IX and X illustrate that the sentiment score polarity histograms for reviews and summaries both had a positive sentiment score on average, although summary had a stronger positive sentiment on average compared to review (0.137 for review and 0.282 for summary). Additionally, note that the modal polarity score for both histograms are 0.00 (+/- 0.05) and that most other reviews/ summary occur to the right of the mode, with review having a weaker positive polarity score due to the fact that there appears to be a greater number of negative reviews compared to that of summaries.

III. D. Effect of Product ID on the Sales Sum Analysis in R

III. D. 1. Scatterplots of Each Sales Sum Vs. Product ID

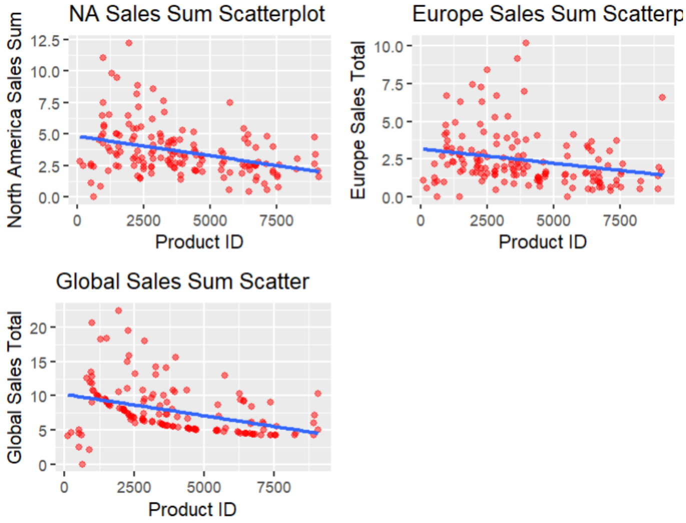


Figure XI.

Sales Sum Variation Scatterplots

Figure XIII shows that there is a strong negative correlation between that of each sales sum column and product ID. Therefore, as Product ID increases, each respective sales sum tends to decrease.

III. D. 2. Histograms of Each Sales Sum

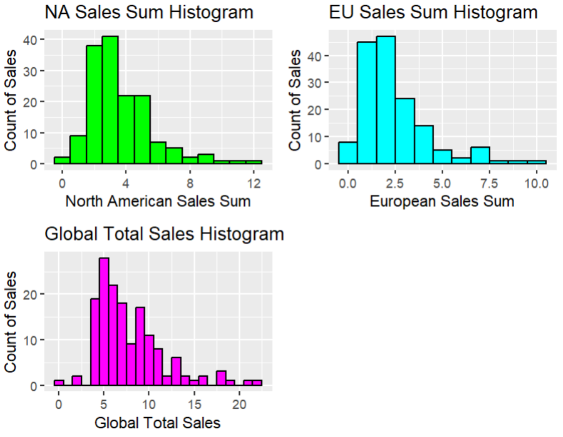


Figure XII. Sales Sum Variation Histograms

Figure XII shows that the 'NA Sales Sum Histogram' follows a bell shape curve depicting a right skew, with the peak of the North American Total Sales being within the 2.5 - 3.5 NA Sales Sum bin. This shape is also observed for the EU Total Sales histogram, with the EU Sales Sum Histogram having its peak within the 1.5 - 2.5 EU Sales Sum bin. However, whilst the shape of the 'Global Sales Sum Histogram' also follows a similar shape to the other two histograms for most of the sales sum values obtained, but there are unexpected increases between the 7.5 - 8.5 & 8.5 - 9.5 + 11.5 - 12.5 & 12.5 - 13.5 Global Sales Sum bins. Note that the modal global total sales lies within the 4.5 - 5.5 Global Sales Sum bin.

III. D. 3. Boxplots of Each Sales Sum Vs. Product ID

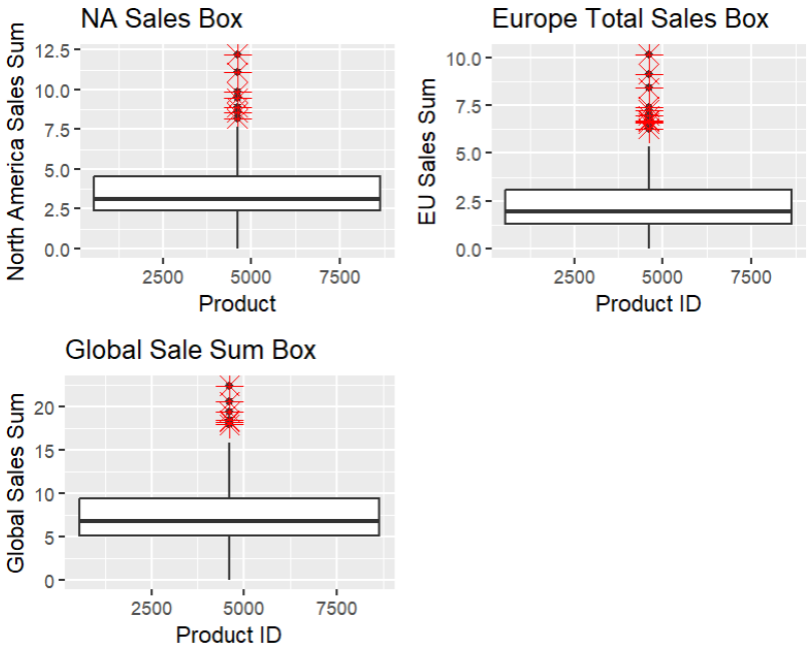


Figure XIII. Sales Sum Variation Boxplots

III. D. 3.1. Boxplot Statistics

	NA_Sales	EU_Sales	Global_Sales
Minimum	0.00	0.00	0.00
Maximum	7.625	5.375	15.75
Lower quartile	2.375	1.25	5.00
Upper quartile	4.625	3.25	9.75
Mean	3.125	2.125	7.00
IQR	2.25	2.00	4.75

III. E. Normality of The Data in R

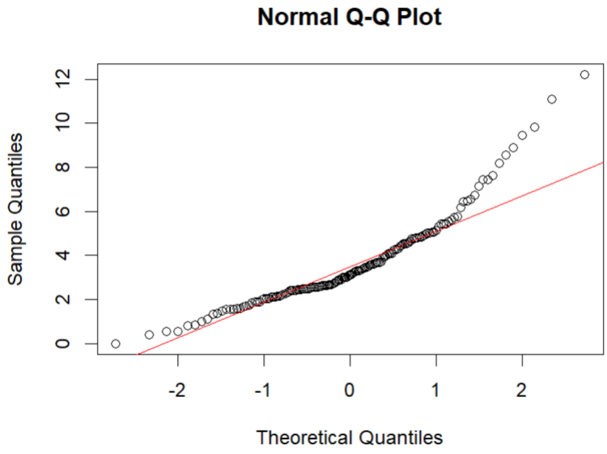


Figure XIV. Q – Q Plot: NA_Sales_Sum

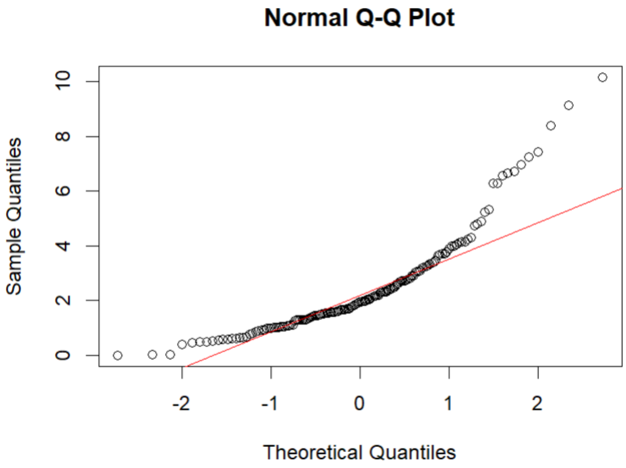


Figure XV. Q – Q Plot: EU_Sales_Sum

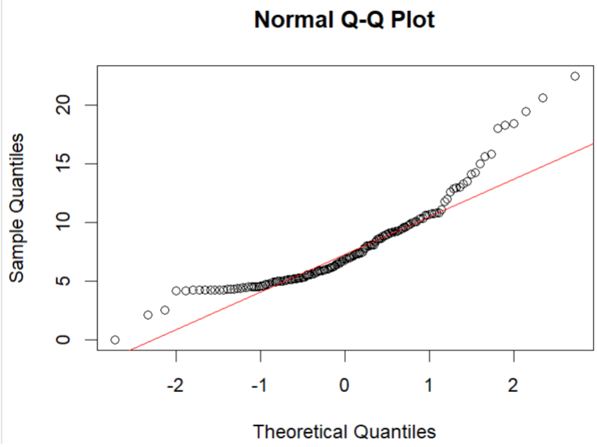


Figure XVI. Q – Q Plot: Global_Sales_Sum

Q – Q Plots can help to determine the normality of the data. Considering this, looking at figures XIV – XVI, it can be determined that all the sales sum data is right skewed since the upper part of the Q – Q plots deviate from the reference lines provided.

III. F. Relationships Between the Sales Sum Data in R

III. E. 1. Simple Linear Regression Models

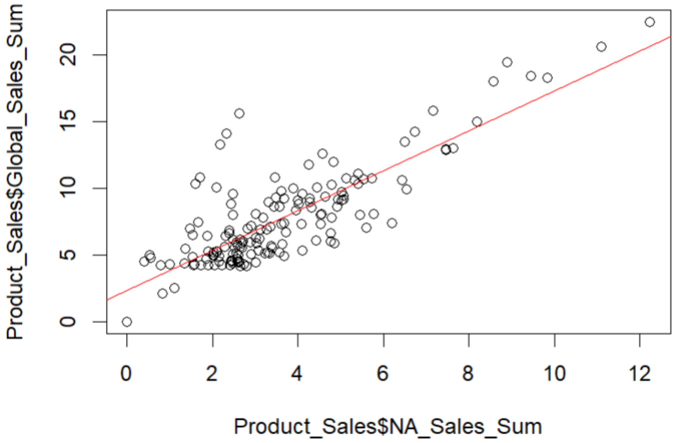


Figure XVII. Simple Linear Regression - NA_Sales_Sum Vs. Global_Sales_Sum

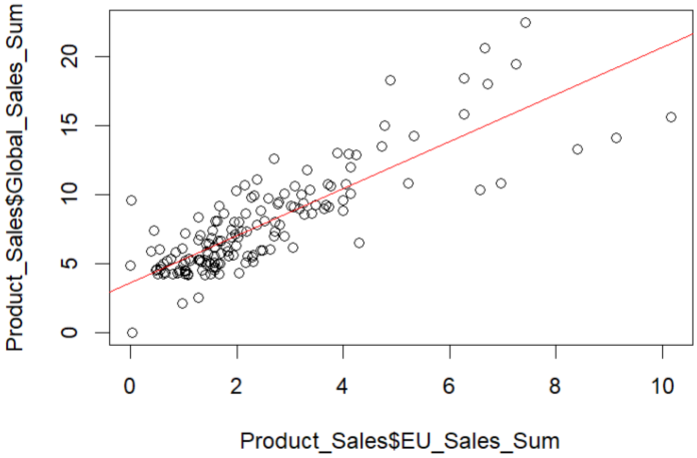


Figure XVIII. Simple Linear Regression - EU_Sales_Sum Vs. Global_Sales_Sum

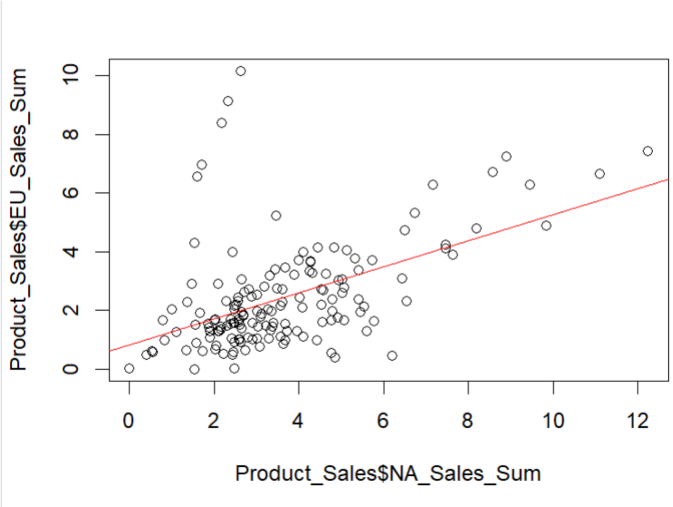


Figure XIX. Simple Linear Regression - NA_Sales_Sum Vs. EU_Sales_Sum

Figures XVII - XIX show simple linear regression plots comparing the 3 sales sum data obtained. These plots show that there is a strong positive correlation between NA_Sales_Sum & Global_Sales_Sum + EU_Sales_Sum & Global_Sales_Sum, with this observation backed up by their correlation values of 0.818 and 0.834 respectively. However, the correlation between NA_Sales_Sum & EU_Sales_Sum is only moderately positive, and is backed up by its correlation value of 0.498.

IV. Patterns and Predictions

IV. A. Accumulation of Loyalty Points

From figures I – III, it can be determined that there is a strong positive correlation between the square root of loyalty points & spending score as well as there being a strong positive correlation between the cube root of loyalty points and remuneration. However, it should also be noted there isn't much correlation between loyalty points and age. Therefore, the conclusion to this question is that customers tend to accumulate loyalty points through spending score & remuneration, but age plays a very small or no part in the overall accumulation.

IV. B. How can groups within the customer base be used to target specific market segments?

As can be seen in figure VI, there are 5 groups of customers within the customer base. These groups can be observed as follows:

- low spend scores (spend < 40) + low remuneration (remuneration < 35)
- moderate spend scores (spend = 40 - 60) + moderate remuneration (remuneration = 35 - 55)
- high spend scores (spend > 60) + high remuneration (remuneration > 55)
- low spend scores + high remuneration
- high spend scores + low remuneration

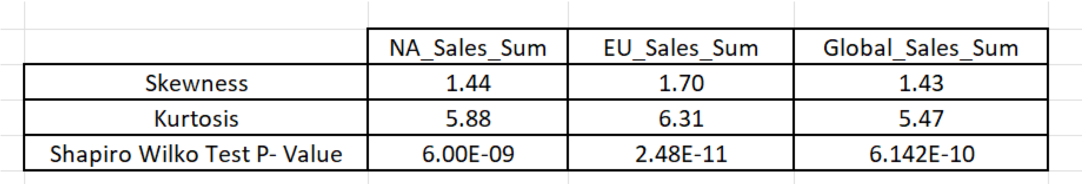
IV. C. Social Media Information

Although, there is a positive sentiment towards Turtle Games overall, it should be noted that a large majority of customers remain neutral towards the firm. Additionally, looking at the most common words, it can be concluded that most customers believe that the games sold by Turtle Games are fun or great, with many opting to rate it as four or five stars, with very little mention of the other merchandise sold by Turtle Games. On the other hand looking at the most positive/negative reviews/summaries, it showed that the consistency of the products sold is lacking as some people found the product to be awesome and come in good quality whilst others found it disappointing, complex and coming in poor condition.

IV. D. Sum of Sales Per Product ID

From figure XI, it can be determined that there is a strong negative correlation between Product ID and each sales sum. Additionally, each sales sum appears to follow a bell-shaped curve around a peak, which exists in a position to the left of the central portion indicating that each sales sum is right skewed. Note that by looking at each respective product ID/Sales Sum boxplots, Global_Sales_Sum has the greatest mean and IQR indicating that the sales sum per product ID is greatest on a global scale as opposed to North America or Europe on average, and using this same logic it can also be determined that on average, Europe has the least sales sum per product ID.

IV. E. Data Reliability

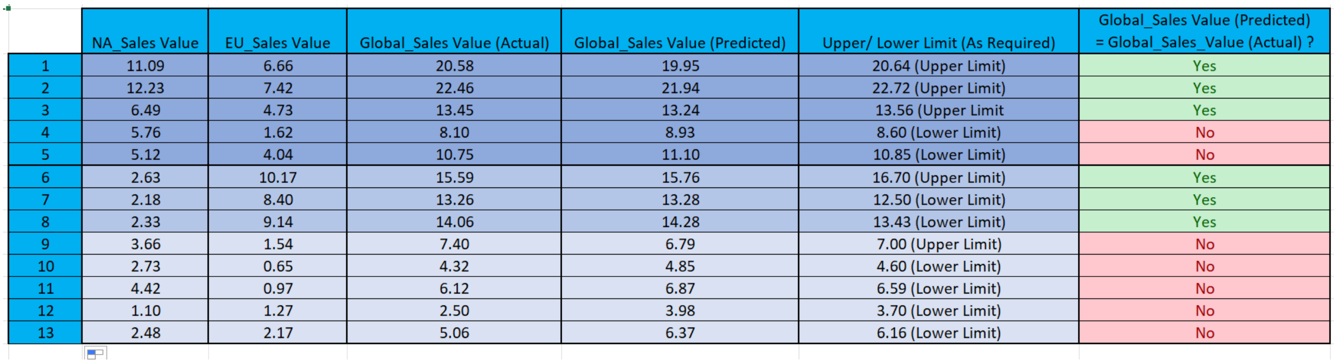


	NA_Sales_Sum	EU_Sales_Sum	Global_Sales_Sum
Skewness	1.44	1.70	1.43
Kurtosis	5.88	6.31	5.47
Shapiro Wilko Test P- Value	6.00E-09	2.48E-11	6.142E-10

The above skewness and kurtosis data, along with figures XIV – XVI show that all 3 sales sum data are right skewed, with EU_Sales_Sum being significantly more right skewed compared to the other 2 sales sum. Therefore since all 3 sales sum are right skewed, have kurtosis values suggesting the data is leptokurtic and have Shapiro – Wilko Test p – values of less than 0.05, it can be deduced that the data cannot be considered to be normally distributed and that therefore the data isn't reliable.

IV. F. Relationships Between North American, European, and Global Sales?

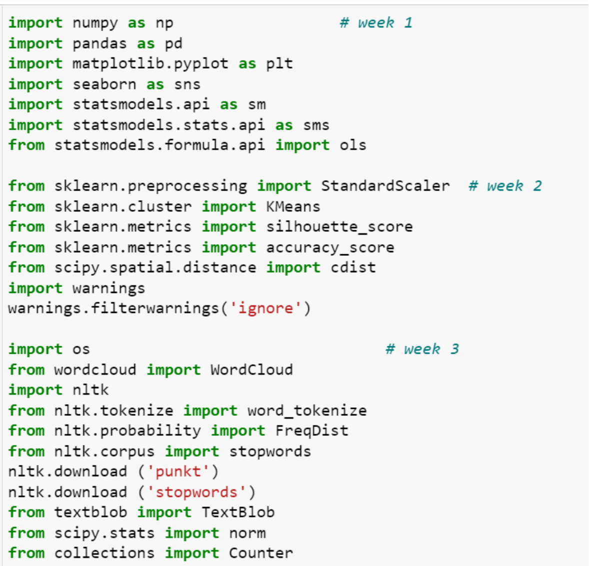
The simple linear regression relationships between the North American Sales Sum & Global Sales Sum + European Sales Sum + Global Sales Sum shows strong positive correlation whilst linear regression of North American Sales Sum + European Sales Sum only shows moderate positive correlation. As such the prediction of the Global sales sum using multiple linear regression can only be predicted within the desired confidence interval when the EU_Sales_Sum + NA_Sales_Sum is significantly large enough (i.e., i.e., EU_Sales_Sum + NA_Sales_Sum > 10) as shown by the following table:



	NA_Sales Value	EU_Sales Value	Global_Sales Value (Actual)	Global_Sales Value (Predicted)	Upper/ Lower Limit (As Required)	Global_Sales Value (Predicted) = Global_Sales_Value (Actual) ?
1	11.09	6.66	20.58	19.95	20.64 (Upper Limit)	Yes
2	12.23	7.42	22.46	21.94	22.72 (Upper Limit)	Yes
3	6.49	4.73	13.45	13.24	13.56 (Upper Limit	Yes
4	5.76	1.62	8.10	8.93	8.60 (Lower Limit)	No
5	5.12	4.04	10.75	11.10	10.85 (Lower Limit)	No
6	2.63	10.17	15.59	15.76	16.70 (Upper Limit)	Yes
7	2.18	8.40	13.26	13.28	12.50 (Lower Limit)	Yes
8	2.33	9.14	14.06	14.28	13.43 (Lower Limit)	Yes
9	3.66	1.54	7.40	6.79	7.00 (Upper Limit)	No
10	2.73	0.65	4.32	4.85	4.60 (Lower Limit)	No
11	4.42	0.97	6.12	6.87	6.59 (Lower Limit)	No
12	1.10	1.27	2.50	3.98	3.70 (Lower Limit)	No
13	2.48	2.17	5.06	6.37	6.16 (Lower Limit)	No

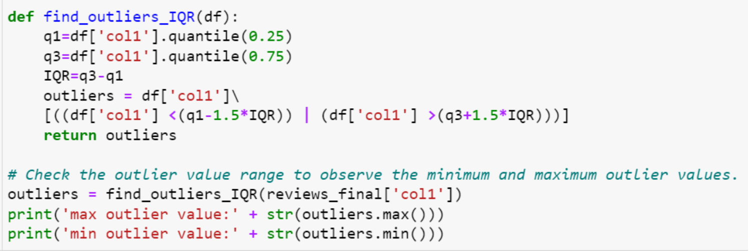
V. Appendix

V. A. Python Code



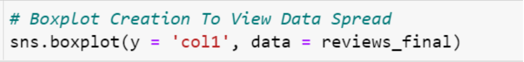
```
import numpy as np                          # week 1
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm
import statsmodels.stats.api as sms
from statsmodels.formula.api import ols

from sklearn.preprocessing import StandardScaler  # week 2
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from sklearn.metrics import accuracy_score
from scipy.spatial.distance import cdist
import warnings
warnings.filterwarnings('ignore')

import os                                        # week 3
from wordcloud import WordCloud
import nltk
from nltk.tokenize import word_tokenize
from nltk.probability import FreqDist
from nltk.corpus import stopwords
nltk.download ('punkt')
nltk.download ('stopwords')
from textblob import TextBlob
from scipy.stats import norm
from collections import Counter
```

Figure 1. All Modules and Libraries



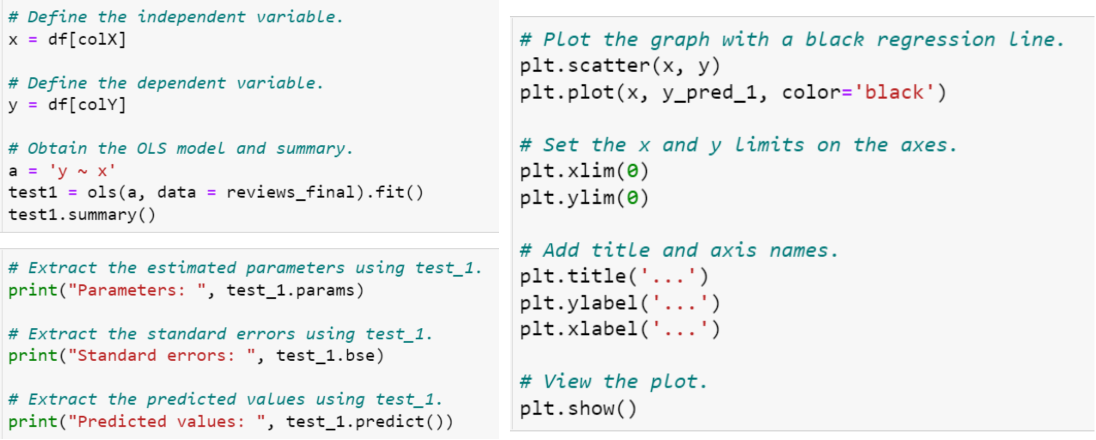
```
def find_outliers_IQR(df):
    q1=df['col1'].quantile(0.25)
    q3=df['col1'].quantile(0.75)
    IQR=q3-q1
    outliers = df['col1']\
    [((df['col1'] <(q1-1.5*IQR)) | (df['col1'] >(q3+1.5*IQR)))]
    return outliers

# Check the outlier value range to observe the minimum and maximum outlier values.
outliers = find_outliers_IQR(reviews_final['col1'])
print('max outlier value:' + str(outliers.max()))
print('min outlier value:' + str(outliers.min()))
```

Figure 2. User – Defined Function: Outliers



```
# Boxplot Creation To View Data Spread
sns.boxplot(y = 'col1', data = reviews_final)
```

Figure 3. Boxplot Template Code

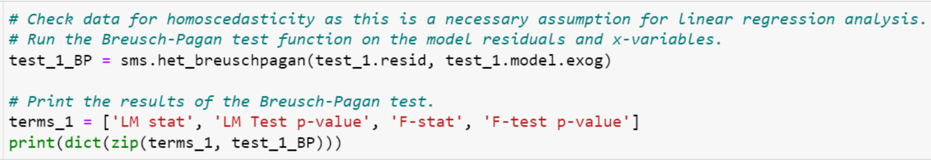


```
# Define the independent variable.
x = df[colX]

# Define the dependent variable.
y = df[colY]

# Obtain the OLS model and summary.
a = 'y ~ x'
test1 = ols(a, data = reviews_final).fit()
test1.summary()
```

```
# Extract the estimated parameters using test_1.
print("Parameters: ", test_1.params)

# Extract the standard errors using test_1.
print("Standard errors: ", test_1.bse)

# Extract the predicted values using test_1.
print("Predicted values: ", test_1.predict())
```

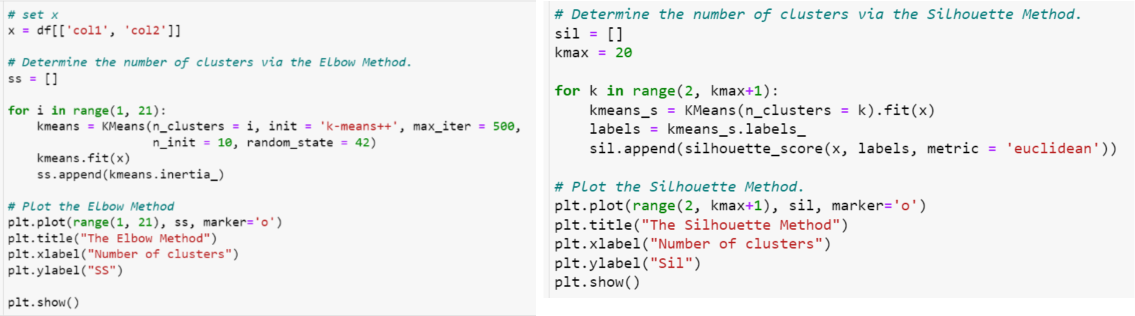
```
# Plot the graph with a black regression line.
plt.scatter(x, y)
plt.plot(x, y_pred_1, color='black')

# Set the x and y limits on the axes.
plt.xlim(0)
plt.ylim(0)

# Add title and axis names.
plt.title('...')
plt.ylabel('...')
plt.xlabel('...')

# View the plot.
plt.show()
```

Figure 4. Linear Regression Analysis Template Code



```
# Check data for homoscedasticity as this is a necessary assumption for linear regression analysis.
# Run the Breusch-Pagan test function on the model residuals and x-variables.
test_1_BP = sms.het_breuschpagan(test_1.resid, test_1.model.exog)

# Print the results of the Breusch-Pagan test.
terms_1 = ['LM stat', 'LM Test p-value', 'F-stat', 'F-test p-value']
print(dict(zip(terms_1, test_1_BP)))
```

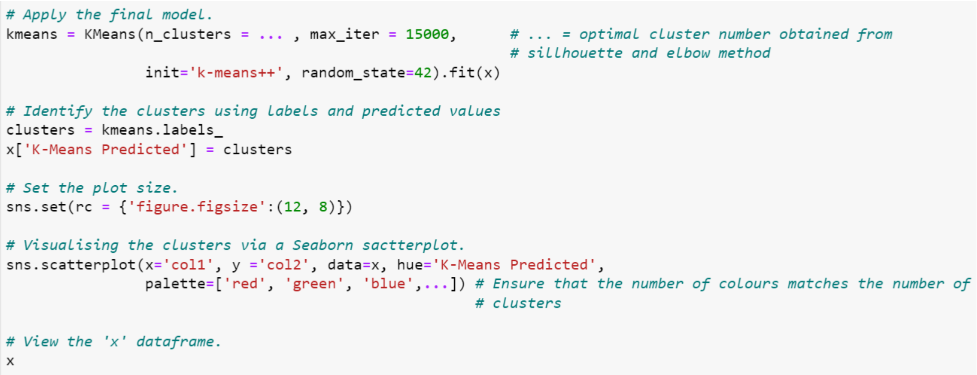
Figure 5. Homoscedasticity Analysis: Breusch – Pagan Test Template Code



```
# set x
x = df[['col1', 'col2']]

# Determine the number of clusters via the Elbow Method.
ss = []

for i in range(1, 21):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', max_iter = 500,
                    n_init = 10, random_state = 42)
    kmeans.fit(x)
    ss.append(kmeans.inertia_)

# Plot the Elbow Method
plt.plot(range(1, 21), ss, marker='o')
plt.title("The Elbow Method")
plt.xlabel("Number of clusters")
plt.ylabel("SS")

plt.show()
```

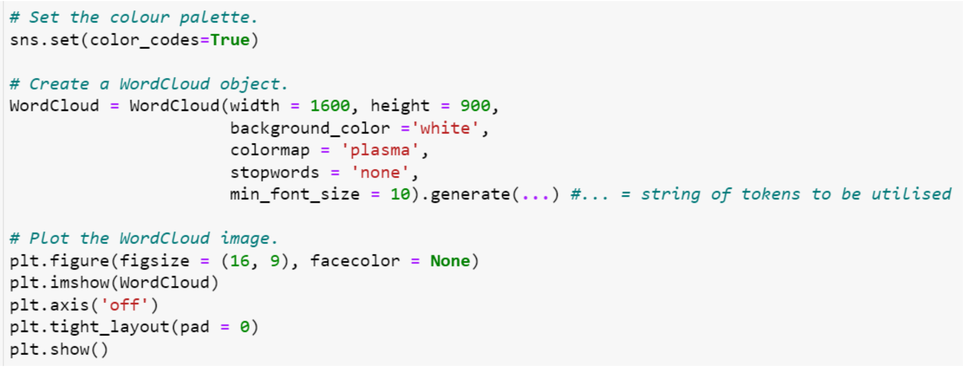
```
# Determine the number of clusters via the Silhouette Method.
sil = []
kmax = 20

for k in range(2, kmax+1):
    kmeans_s = KMeans(n_clusters = k).fit(x)
    labels = kmeans_s.labels_
    sil.append(silhouette_score(x, labels, metric = 'euclidean'))

# Plot the Silhouette Method.
plt.plot(range(2, kmax+1), sil, marker='o')
plt.title("The Silhouette Method")
plt.xlabel("Number of clusters")
plt.ylabel("Sil")
plt.show()
```

Figure 6. K – Means Clustering: Elbow and Silhouette Method Template Code



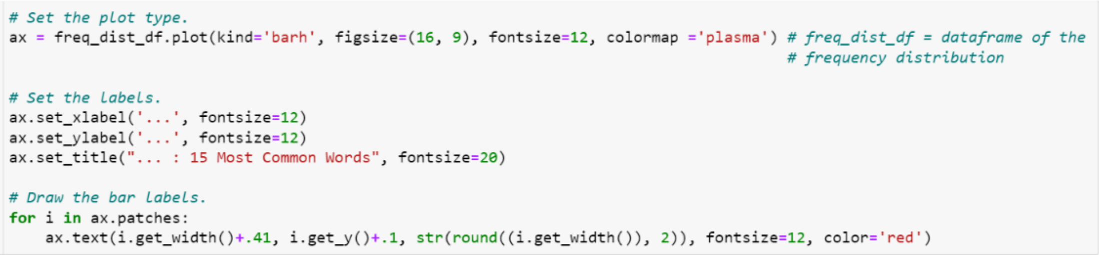
```
# Apply the final model.
kmeans = KMeans(n_clusters = ... , max_iter = 15000,      # ... = optimal cluster number obtained from
                                                          # sillhouette and elbow method
                init='k-means++', random_state=42).fit(x)

# Identify the clusters using labels and predicted values
clusters = kmeans.labels_
x['K-Means Predicted'] = clusters

# Set the plot size.
sns.set(rc = {'figure.figsize':(12, 8)})

# Visualising the clusters via a Seaborn sactterplot.
sns.scatterplot(x='col1', y ='col2', data=x, hue='K-Means Predicted',
                palette=['red', 'green', 'blue',...]) # Ensure that the number of colours matches the number of
                                                      # clusters

# View the 'x' dataframe.
x
```

Figure 7. Python Code Template Used to Fit Final K – Means Model



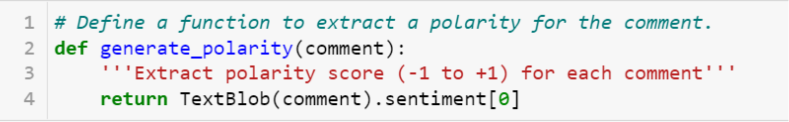
```
# Set the colour palette.
sns.set(color_codes=True)

# Create a WordCloud object.
WordCloud = WordCloud(width = 1600, height = 900,
                      background_color ='white',
                      colormap = 'plasma',
                      stopwords = 'none',
                      min_font_size = 10).generate(...) #... = string of tokens to be utilised

# Plot the WordCloud image.
plt.figure(figsize = (16, 9), facecolor = None)
plt.imshow(WordCloud)
plt.axis('off')
plt.tight_layout(pad = 0)
plt.show()
```

Figure 8. Word Cloud Template Code



```
# Set the plot type.
ax = freq_dist_df.plot(kind='barh', figsize=(16, 9), fontsize=12, colormap ='plasma') # freq_dist_df = dataframe of the
                                                                                     # frequency distribution

# Set the labels.
ax.set_xlabel('...', fontsize=12)
ax.set_ylabel('...', fontsize=12)
ax.set_title("... : 15 Most Common Words", fontsize=20)

# Draw the bar labels.
for i in ax.patches:
    ax.text(i.get_width()+.41, i.get_y()+.1, str(round((i.get_width()), 2)), fontsize=12, color='red')
```

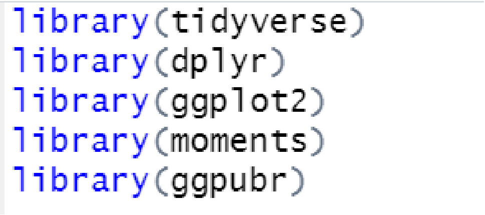
Figure 9. Frequency Distribution Bar Plot Template Code



```
# Define a function to extract a polarity for the comment.
def generate_polarity(comment):
    '''Extract polarity score (-1 to +1) for each comment'''
    return TextBlob(comment).sentiment[0]
```

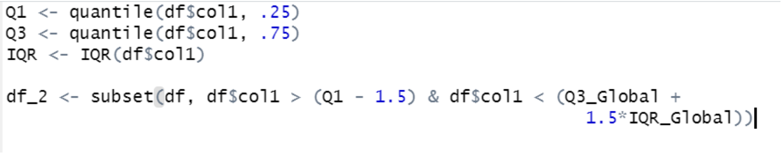
Figure 10. User Defined Function: Polarity

V. B. R Code



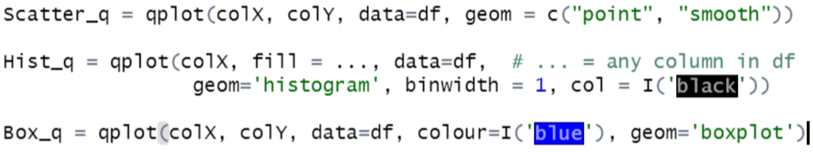
```
library(tidyverse)
library(dplyr)
library(ggplot2)
library(moments)
library(ggpubr)
```

Figure 11. All Modules and Libraries



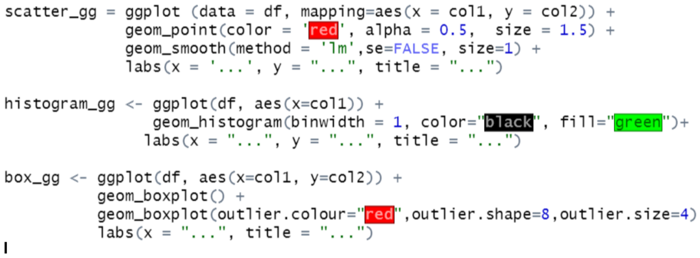
```
Q1 <- quantile(df$col1, .25)
Q3 <- quantile(df$col1, .75)
IQR <- IQR(df$col1)

df_2 <- subset(df, df$col1 > (Q1 - 1.5) & df$col1 < (Q3_Global +
                                                      1.5*IQR_Global))
```

Figure 12. Function Template Used to Remove Outliers in R



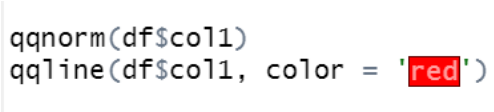
```
Scatter_q = qplot(colX, colY, data=df, geom = c("point", "smooth"))

Hist_q = qplot(colX, fill = ..., data=df,  # ... = any column in df
               geom='histogram', binwidth = 1, col = I('black'))

Box_q = qplot(colX, colY, data=df, colour=I('blue'), geom='boxplot')
```

Figure 12. qplot Template Code for scatterplots, histograms, and boxplots



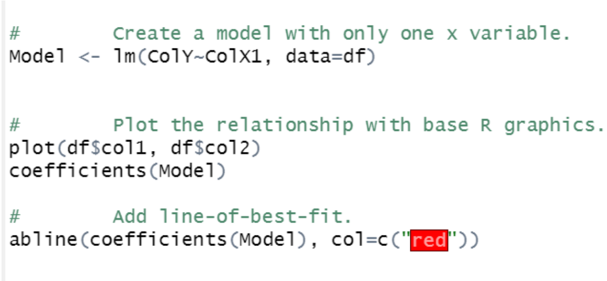
```
scatter_gg = ggplot (data = df, mapping=aes(x = col1, y = col2)) +
             geom_point(color = 'red', alpha = 0.5,  size = 1.5) +
             geom_smooth(method = 'lm',se=FALSE, size=1) +
             labs(x = '...', y = "...", title = "...")

histogram_gg <- ggplot(df, aes(x=col1)) +
                geom_histogram(binwidth = 1, color="black", fill="green")+
               labs(x = "...", y = "...", title = "...")

box_gg <- ggplot(df, aes(x=col1, y=col2)) +
          geom_boxplot() +
          geom_boxplot(outlier.colour="red",outlier.shape=8,outlier.size=4)
          labs(x = "...", title = "...")
```

Figure 13. ggplot Template Code for scatterplots, histograms, and boxplots



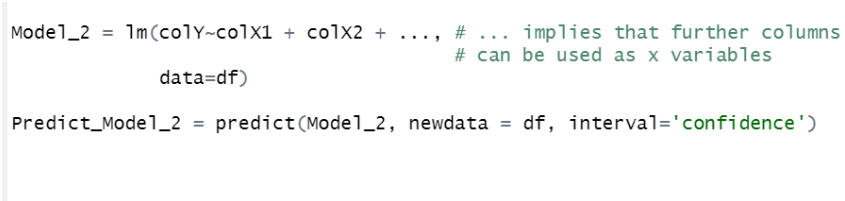
```
qqnorm(df$col1)
qqline(df$col1, color = 'red')
```

Figure 14. Template Code for q – q plots



```
#        Create a model with only one x variable.
Model <- lm(ColY~ColX1, data=df)


#        Plot the relationship with base R graphics.
plot(df$col1, df$col2)
coefficients(Model)

#        Add line-of-best-fit.
abline(coefficients(Model), col=c("red"))
```

Figure 15. Template Code in R for Simple Linear Regression



```
Model_2 = lm(colY~colX1 + colX2 + ..., # ... implies that further columns
                                       # can be used as x variables
             data=df)

Predict_Model_2 = predict(Model_2, newdata = df, interval='confidence')
```

Figure 16. Template Code for Multiple Linear Regression