Sentiment Analysis on FIFA 2018 World Cup

Chukwuekwu Musei

1.0 Introduction

1.1 *Background*

The era of advanced technology and innovation has given rise to social media platforms such as Twitter, a micro blogging application where people share their opinion or sentiment about any topic of their choice through tweets- a message containing 140 characters. Today, Twitter has over 353 Million subscribers who discuss various topics such as politics, arts and sports, in particular, football. Over the years, the game of football has stood as a symbol of unity globally, where countries in different continents come together to vie for the FIFA World Cup tournament every four years. More important is the significant way it has consistently kept Nigeria and other developing nations at peace, despite being polarized by ethnicity and religion. It is on this basis the FIFA 18 World Cup channel has been chosen for this task.

The purpose of this paper is to analyse which of the chosen machine learning algorithms- Linear regression (LR), Decision Tree (DT) and Random Forest (RF) that best performs well in performing sentiment analysis on the FIFA 18 World Cup competition dataset in order to determine the tweets likely to get the most Likes.

The paper has been organised in the following way. Session 2 provides the methodology used in carrying out the task. In session 3, the results and findings are interpreted and discussed, while session 4 concludes by highlighting the benefits and limitations of the models.

1.2 *Related Work*

1.2.1 Sentiment Analysis and Twitter

The success of sentiment analysis as it is known today cannot be written without the emergence of bigdata that makes sense of data through predictive analytics (Kuhn and Johnson, 2013). According to Barnaghi *et al.* (2016), they opine that Twitter is the best known micro-blogging application that people use to express and interact about their opinions. These opinions could be positive, negative or neutral(Barnaghi *et al.*, 2016). Pang et al. (2002) study text categorization with the use of Logistics Regression, Support Vector machine and Naïve Bayes models to classify sentiment in movie reviews. Barnaghi *et al.* (2016) performed sentiment analysis on the tweets of the FIFA 2014 World Cup to gather negative and positive opinions using Bayesian Logistic Regression and Naïve Bayes models.

This paper will use the LR, RF and DT for predicting top tweets in the FIFA 2018 World Cup Final using the provided dataset.

2.0 Methodology

The exploratory data analysis (EDA), preprocessing and cleaning of the FIFA 2018 World Cup Twitter dataset was carried out in python.

The dataset was preprocessed and additional features such as Hour of day, Time of day, and length of tweets were created and added to the FIFA 18 World Cup dataset, which are to form part of the predictors during modelling. The bar chart and line graph in matplotlib.pyplot were used to visualise the number of tweets per day, hour, and time of the day to gain insight at what time and day during the World cup competition that people tweeted most.

In addition, the words in the Tweet column were tokenized in order to separate each word as a stand-alone, leading to the effective removal of stopwords, where words less than seven were removed. The choice of seven words was based on the nature of the dataset and also when words with less than five words were executed, the worldcloud was dominated by meaningless words not helpful for the analysis.

Sentiment analysis was then carried out to get people's opinion whether positive, negative or neutral. This was done using the TextBlob in the python library where positive tweets was assigned polarity > 0, negative < 0 and neutral=0.

The machine learning methods used for this task were LR, DT and RF. The cleaned dataset was partitioned into train and test data at 80% and 20% respectively. The test data was used to check for the accuracy of the predicted train dataset in order to evaluate its performance by safeguarding it from overfitting to produce optimally good models (Hastie *et al.,* 2009).

The sklearn model selection package in python was used. LR was selected because it performs well when the predictors are linear (Hastie *et al.,* 2009). Similarly, RF was used because it has the capability to accommodate both linear and non-linear nature of the predictors as well as reduce overfitting of the model (Kuhn and Johnson, 2013).

.

3.0 Results and Discussions

3.1 Data Cleaning

3.1.1 Descriptive Statistics of fifa_1 dataframe

The result from summary statistics of the dataset show 530000 observations and 4 variables (See Appendix 2). Further analysis indicate 551 NAs in the Tweet variable, 238591 duplicates and 3808 punctuations. These were removed during data cleaning because they will cause errors and reduce the accuracy of the models (Hastie *et al.*, 2009). The Urls created from the Tweet variable were counted to create one of the features in the dataset, including such features as Hour of day, Time of day, Length of tweet into the fifa_1 dataframe, increasing the number of predictors from 4 to 11 and reducing the number of observation from 530000 to 290858 were created for the analysis as shown in Table 1 below.

Table 1: Descriptive statistics of the final fifa_1 dataset

```
display(fifa_1)
```

	Date	Tweet	Likes	RTs	Url	Time_of_day	Hour_of_day	length_of_tweet	polarity	sentiment	Sentiment_score
0	2018-02-07 01:35:00	"'only", "'goalkeepers", "'have", "'saved", "'...	0	477	0	sunrise	1	101	0.000000	Neutral	0.000000
1	2018-02-07 01:35:00	"'scores", "'winning", "'penalty", "'send", "'...	0	1031	0	sunrise	1	123	0.500000	Positive	0.500000
2	2018-02-07 01:35:00	"'tonight", "'have", "'game",	0	488	0	sunrise	1	29	-0.400000	Negative	-0.400000
3	2018-02-07 01:35:00	"'stronger", "'turn", "'music", "'that", "'pow...	0	0	0	sunrise	1	60	0.000000	Neutral	0.000000
5	2018-02-07 01:35:00	"'looking", "'strong", "'going", "'into", "'kn...	0	153	0	sunrise	1	94	0.433333	Positive	0.433333
...	...	...	...	...	...	...	...	...	...	...	...
529982	2018-07-15 22:49:00	"'guys", "'voted", "'right", "'congratulations...	0	0	0	night	22	72	0.285714	Positive	0.285714
529985	2018-07-15 22:49:00	"'ronaldos", "'hattrick", "'germany", "'groups...	0	1138	0	night	22	144	-0.300000	Negative	-0.300000
529986	2018-07-15 22:49:00	"'long", "'time", "'very", "'young", "'griezma...	0	6	0	night	22	85	0.040000	Positive	0.040000
529989	2018-07-15 22:49:00	"'proud",	0	1099	0	night	22	9	0.800000	Positive	0.800000
529998	2018-07-15 22:49:00	"'starts", "'todays", "'final",	0	1013	0	night	22	31	0.000000	Neutral	0.000000

290858 rows × 11 columns

3.1.2 Visualisation of Number of tweets by date in the fifa_1 dataframe

The line graph in Figure 1 below show frequency of tweet by date, indicating that more tweets at the FIFA 2018 World Cup was experienced on 1st, 7th, and 11th of July 2018, respectively. According to FIFA (2018), the final result of the competition(See Appendix 3) show that on the 1st of July 2018, the host country Russia upset football giant Spain at the round of 16 game on penalty shootout. Also, the second game of the day ended in penalty shootout between Croatia and Denmark, which ended in favour of Croatia. Similarly, on the 7th of July 2018, the host played with Croatia in the quarter final stage, but this time losing on penalties. The second game of same day involving England and Sweden, ended in favour of the former. Lastly, on the 11th of July 2018 featured another upset in the semi-final match, where Croatia upset another football powerhouse, England. It can be claimed that the major talking point of the FIFA 2018 World Cup centered on these dates above because the dates featured the host, Russia as well as two power houses in world football, Spain and England who were both eliminated from the tournament by the underdogs.

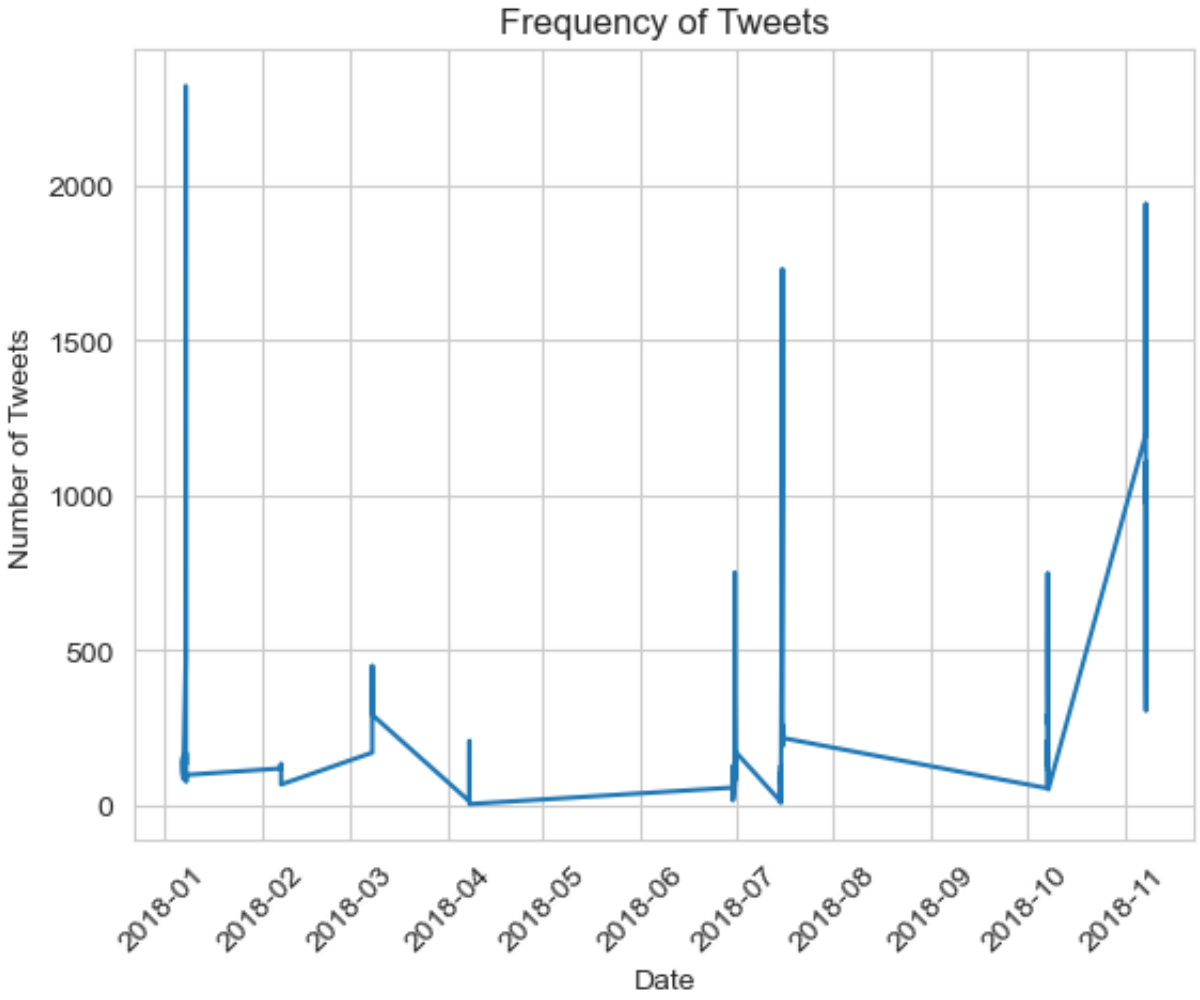


Fig. 1: Graph showing date with the highest number of tweets during the tournamant

3.1.3 Visualisation of Number of tweets by Hour of day in the fifa_1 dataframe

The bar chart in Figure 2 show the frequency of tweets by Hour of day during the tournamant. It can be observed that 17.00hrs has the highest number of people tweeting over 50,000 tweets. According to FIFA(2018), it showed that most of the matches were played in 17:00hrs GMT +1 which is two hours behind Russia at 19:00hrs GMT +3 (see Appendix 4). It can be observed that most matches, including the ones that recorded the highest tweets per day in Figure 1, were played at that time. This can be claimed on why 17:00hrs has the highest tweet by Hour of day.

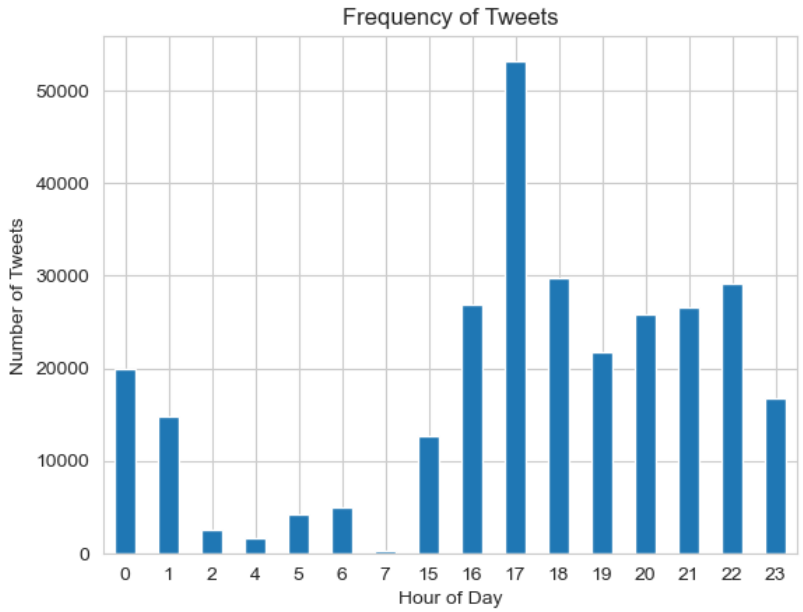


Fig.2: Bar chart showing frequency of tweets by Hour of day during the tournamant

3.1.4 Visualisation of Number of tweets by Time of day in the fifa_1 dataframe

Figure 3 show the result of the Time of day variable by number of tweets. The variable was categorised into 3 groups of sunrise(0-11hrs), sunset(12-16hrs) and night(17-23hrs). Result indicate that majority of the tweets during the FIFA 2018 World Cup tournament were at night between the hours of 17hrs to 23:00hrs. This can also be seen as indicated in Figure 2 above.

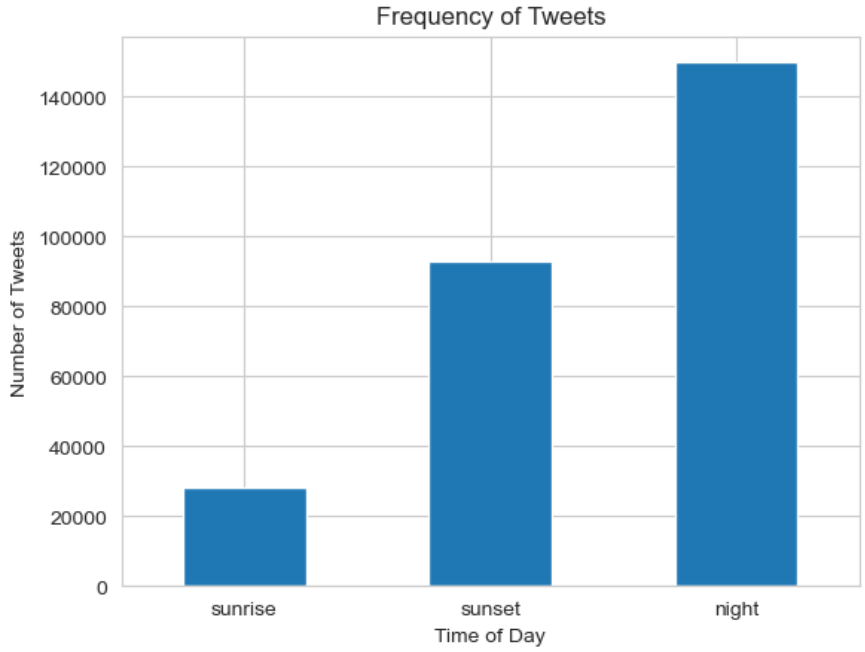


Fig.3:Bar chart showing frequency of tweets by Time of day during the tournamant

3.2 Data Mining Exploration

3.2.1 Wordcloud

Figure 4 show the wordcloud image after the words in the Tweet variable were tokenized in order to effectively remove stopwords. For this task, words below seven was preferred because below five words did not effectively remove meaningless words which where very visible on the wordcloud. It can be observed that quarter final, semifinal and penalty shootout are some of the words that were frequent according to the wordcloud. It can be claimed that the wordcloud result is in line with the results obtained in Figures 1,2 and 3. For instance, two of the penalty shootouts in the FIFA 2018 World Cup games involved the host country, Russia. One of the games was against Spain's elimination at the round of 16 and the other game in which Russia succumbed to Croatia in the quarter final. Also England qualified for the semifinal stage after defeating Sweden, the reason the phrase 'Coming Home' appearing on the wordcloud, the phrase synonymous with English fans for the trophy that has eluded them since 1966 to come back where modern football was birthed (Vincent et al., 2010).

Fig.4: Bar chart showing frequency of tweets by Time of day during the tournamant

3.2.2 Sentiment Analysis

The result from Figure 5 below reveal that the FIFA 2018 World Cup has more positive sentiments than negative ones almost by twice. It also indicates that majority of those that tweeted were neutral in their sentiment. It can be suggested that the positive sentiment expressed were those experienced when the host, Russia- an underdog, defeated football legends, Spain, during the round of 16 phase, as well as when Croatia, a country with 4million population (Croatia Population, 2018) defeated England at the semifinal stage of the competition. Additionally, other positive sentiment could be when England qualified for the semifinal stage, which must have turned negative after losing to Croatia in the semifinal phase.

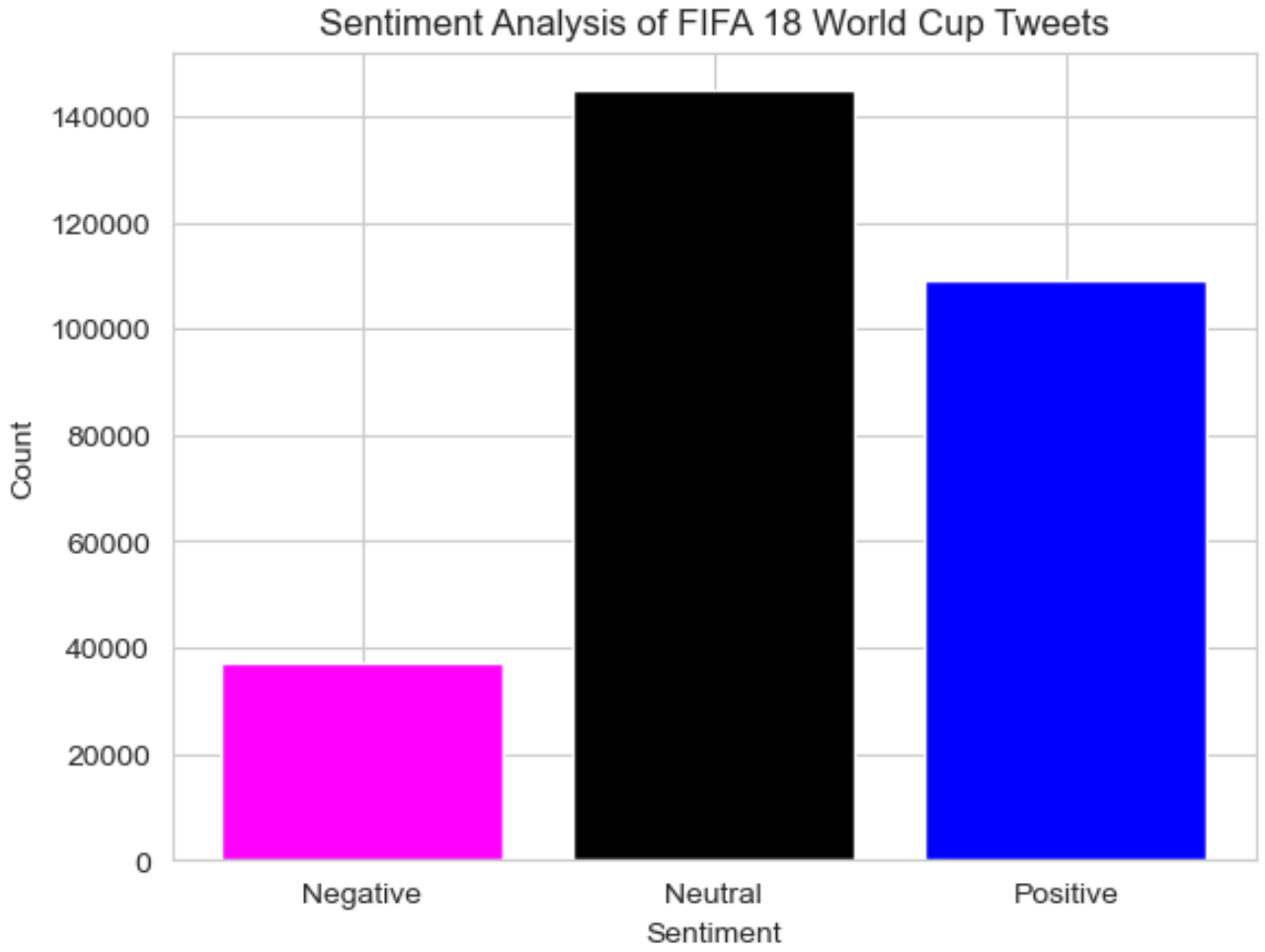


Fig.5: Bar chart showing sentiment analysis of tweets in the fifa_1 dataframe

Furthermore, Figure 6 show the sentiment analysis of the top five tweets. The result imply that there was no positive tweets in the five top tweets. It only showed negative sentiment, which was one-third of the sentiments to that of the neutral sentiment.

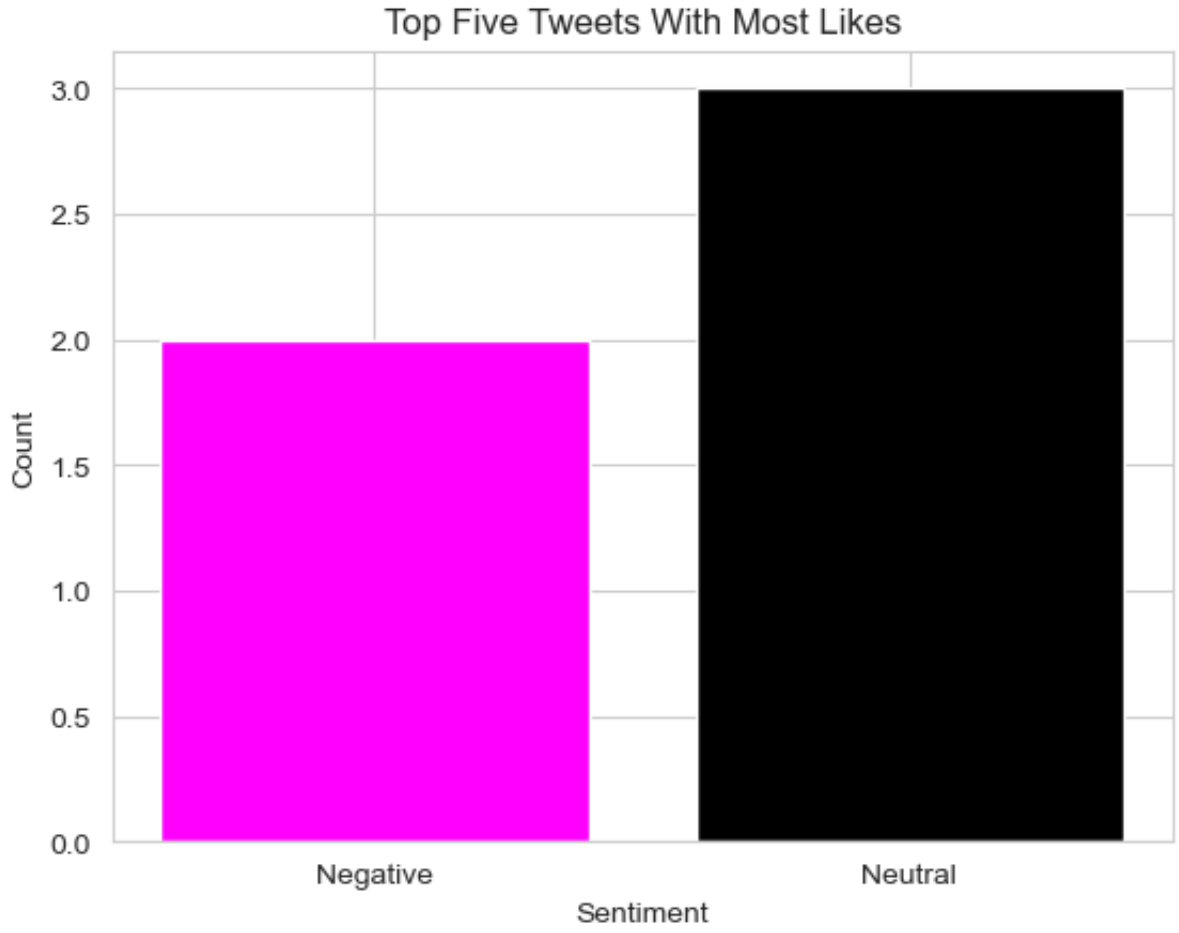


Fig.6: Bar chart showing sentiment analysis of top five tweets

3.3 Machine Learning

Table 2: Machine Learning Results

	Linear Regression(LR)	**Decision Tree (DT)**	**Random Forest(RF)**
Root Squared Mean Error (RSME)	48.76	56.72	53.32
Mean Absolute Error (MAE)	3.38	3.85	3.93
Mean Square Error (MSE)	2377.67	3217.06	2843.99

Table 2 show the results of the three models used to predict how many number of Likes a tweet is likely to get in the fifa_1 dataframe. Comparing the three models, LR performed better than both DT and RF in all the metrics. LR models are known to perform well with small amount of dataset (Hastie *et al*., 2012). It can be claimed that the predictors are linear in nature as well as being able to be interpreted.

On the other hand, DT performed worse than RF because its RSME and MSE values are greater, except for the MAE value where DT performed slightly better than RF. According to Field *et al*. (2012), DT is susceptible to overfitting, leading to why it performs badly when generalized with new data. It can be suggested that the DT may have overfitted the train data of the fifa_1 dataframe, resulting to poor performance of the model.

Furthermore, RF, whose algorithm is built on the ensemble technique (Kuhn and Johnson, 2013) with the capability to prevent overfitting. In the result above, RF did not perform well compared to LR in the three metrics shown above. It could be claimed that in an attempt to create a balance between non-linear relationship and overfitting, it slightly performed less than LR.

Ultimately, these results are related to the study Barnaghi *et al.* (2016) conducted with the dataset of the FIFA 2014 World Cup using regression methods.

4.0 Conclusion

This study was designed to predict the number of likes a tweet is likely to get using three machine learning models (LR, DT, and RF) on the fifa_1 dataset. It can be observed that majority of the engagement on the FIFA twitter handle took place between 17:00hrs to 23:00hrs. It can be useful that the FIFA handle should constantly engage its audience with appealing contents when there is no game or competition at hand. In the case of the tweets of the FIFA 2018 World Cup, it recorded low tweets during the sunrise and sunset periods. Additionally, it is recommended when carrying out a task of this nature, LR model could be the choice for analysis because of its simplicity to interpretation and performance with linear predictors. It was observed that the limitation to this task was in deciding the number of lettered words to remove because this can affect the result of the sentiment analysis and wordcloud as in this task when removing words less than five gave different sentiment analysis and wordcloud to that which is less than seven words. Therefore datasets should be analysed before making a choice as to the number of words to remove in order to make the analysis meaningful.

References

Barnaghi, P., Ghaffari, P., & Breslin, J. G. (2016) 'Opinion mining and sentiment polarity on twitter and correlation between events and sentiment. I*nternational Conference on Big Data Computing Service and Applications.* Available at: DOI. 10.1109/BigDataService.2016.36.

Field, A., Miles, J. and Field Zoë (2012) *Discovering statistics using R.* Thousand Oaks: SAGE/Texts.

FIFA(2018), FIFA 2018 World Cup result. Available at: https://www.google.com/search?q=fifa+2018+world+cup+results&oq=fifa+2018+world+cup+results&aqs=chrome..69i57j0i22i30l4j69i60l3.21528j0j7&sourceid=chrome&ie=UTF-8#sie=lg;/m/06qjc4;2;/m/030q7;br;fp;1;;;

FIFA (2018) The match center. Available at: https://www.fifa.com/tournaments/mens/worldcup/2018russia/match-center

Guyon, I. & Elisseeff, A. (2003) 'An introduction to variable and feature selection' *Journal of Machine Learning Research*, 3, pp. 1157-1182.

Hastie, T., Tibshirani, R. & Friedman, J. (2009) *An introduction to statistical learning*, Springer Science and Business Media, New York.

Kuhn, M. and Johnson, K (2013) *Applied Predictive Modelling*. London, U.K.: Springer.

Pang,B., Lee, L. & Vaithyanathan, S. (2002) 'Thumbs up? sentiment classification using machine learning techniques' *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Available at: https://arxiv.org/pdf/cs/0205070.pdf

Razia Sulthana, A., Jaithunbi, A. K., & Sai Ramesh, L. (2018) 'Sentiment analysis in twitter data using data analytic techniques for predictive modelling', *Journal of Physics: Conference Series, 1000, 012130*. Available at: https://doi/10.1088/1742-6596/1000/1/012130.

Schröer, C., Kruse, F., & Gómez, J. M. (2021) 'A systematic literature review on applying CRISP-DM process model', Procedia Computer Science, 181, pp. 526–534. https://doi/10.1016/j.procs.2021.01.199.

Statista (2023) Number of twitter users. Available at: https://www.statista.com/statistics/303681/twitter-users-worldwide/

Vincent, J., Kian, E. M., Pedersen, P. M., Kuntz, A., & Hill, J. S. (2010) 'England expects: English newspapers' narratives about the English football team in the 2006 world cup. *International Review for the Sociology of Sport*, 45(2), pp. 199–223. Available at: https://doi.org/10.1177/1012690209360084.

Appendix 1: Python Code

```
import nltk

nltk.download([
    "names",
    "stopwords",
    "state_union",
    "twitter_samples",
    "movie_reviews",
    "averaged_perceptron_tagger",
    "vader_lexicon",
    "punkt",
 ])


# In[330]:


nltk.download('brown')
nltk.download('stopwords')
nltk.download('punkt')


# In[331]:


from nltk.corpus import brown
brown.categories()


# In[332]:


import pandas as pd

# Loading FIFA 18 WC CSV file
fifa = pd.read_csv('FIFA 18 WC.csv')


# In[333]:


fifa.head
fifa


# In[334]:


fifa_1 = fifa.copy()
```

DATA CLEANING

In[335]:

```
#checking for ant missing values
fifa_1.isna().any()
```

In[336]:

```
#removing NAs in the Tweet column to enable the text data to be concatenating into a string
fifa_1 = fifa_1.dropna(subset=['Tweet'])
```

In[337]:

```
#checking duplicates from the dataset
number_of_duplicates = fifa_1.duplicated().sum()
print("Number of duplicate rows: ", number_of_duplicates)
```

In[338]:

```
#removing duplicates from the dataset
fifa_1 = fifa_1.drop_duplicates()
```

In[339]:

```
#Removing any punctuations from Tweet column
fifa_1['Tweet'] = fifa_1['Tweet'].str.replace("[^a-zA-Z#]", " ", regex=True)
```

In[340]:

```
#checking for url in tweets
import numpy as np

number_of_urls = (fifa_1['Tweet'].str.contains('http')).sum()
print("Rows in the observations with URLs: ", number_of_urls)
```

```
#creating rows with url in the dataset
fifa_1['Url'] = np.where(fifa_1['Tweet'].str.contains('http'), 1, 0)
```

In[342]:

```
#converting words in the Tweet to lowercase
fifa_1['Tweet'] = fifa_1['Tweet'].str.lower()
print(fifa_1['Tweet'])
```

STATISTICAL ANALYSIS

In[343]:

```
#converting the dates in the Date column to be in the same format
fifa_1['Date'] = pd.to_datetime(fifa_1['Date'])
```

```
# grouping the dataset DataFrame by date and calculate the mean of Likes and RTs for each
date
average_likes_AND_rts = fifa_1.groupby('Date')[['Likes', 'RTs']].mean()

print(average_likes_AND_rts)
```

In[344]:

```
#checking the frquency of tweets by date
frequency_of_tweet = fifa_1['Date'].value_counts().sort_index()
print(frequency_of_tweet)
```

In[345]:

```
import matplotlib.pyplot as plt

# Counting daily tweets and sorting by date
frequency_of_tweet = fifa_1['Date'].value_counts().sort_index()

# Ploting the frequency of tweets on a histogram to see its distribution
plt.hist(frequency_of_tweet.values, bins=10)

# labelling x and y axes labels and that of the title
plt.xlabel('Number of Tweets')
plt.ylabel('Frequency')
plt.title('Distribution of Number of Tweets')

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

In[346]:

```
# Converting the date column to a pandas DatetimeIndex to extract the timeframe from the
date
hour = pd.DatetimeIndex(fifa_1['Date']).hour

# Categorizing the hours into sunrise, sunset, and night
bins = [0, 12, 17, 23]
labels = ['sunrise', 'sunset', 'night']
day_time = pd.cut(hour, bins=bins, labels=labels)
fifa_1 ['Time_of_day'] = day_time
fifa_1 ['Hour_of_day'] = hour


# In[347]:


# Creating a DataFrame that computes the sum total of Likes and RTs happening daily
Total_Likes_AND_RTs = fifa_1.groupby(['Date'])[['Likes', 'RTs']].sum().reset_index()

#scatter plot of Likes vs RTs
plt.scatter(Total_Likes_AND_RTs['Likes'], Total_Likes_AND_RTs['RTs'])
plt.title('Likes vs. RTs')
plt.xlabel('Likes')
plt.ylabel('RTs')
plt.show()


# In[348]:


# Count the number of tweets per day and sort by date
frequency_of_tweet = fifa_1['Date'].value_counts().sort_index()

# Plot the frequency of tweets as a line graph
plt.plot(frequency_of_tweet.index, frequency_of_tweet.values)

# Add x-axis and y-axis labels and a title
plt.xlabel('Date')
plt.ylabel('Number of Tweets')
plt.title('Frequency of Tweets')

# Customize the x-axis tick labels
plt.xticks(rotation=45)

# Display the plot
plt.show()


# In[349]:


sum_likes_hourly = fifa_1.groupby('Hour_of_day')['Tweet'].count().plot.bar()

# Add x-axis and y-axis labels and a title
plt.xlabel('Hour of Day')
```

```
plt.ylabel('Number of Tweets')
plt.title('Frequency of Tweets')

# labelling the x-axis
plt.xticks(rotation=360)

# Displaying the plot
plt.show()


# In[350]:


sum_tweets_time = fifa_1.groupby('Time_of_day')['Tweet'].count().plot.bar()

# Add x-axis and y-axis labels and a title
plt.xlabel('Time of Day')
plt.ylabel('Number of Tweets')
plt.title('Frequency of Tweets')

# Customize the x-axis tick labels
plt.xticks(rotation=360)

# Display the plot
plt.show()


# # **DATA MININIG EXPLORATION**
#
# ### **1. Tokenizing and removing Stop words**

# In[351]:


import nltk
nltk.download('punkt')
from nltk.tokenize import sent_tokenize, word_tokenize
fifa_1['Tweet'] = fifa_1['Tweet'].apply(lambda x: word_tokenize(x))


# In[352]:


def tokenize_tweet(tweet):
    return nltk.word_tokenize(tweet)

fifa_1['Tweet'] = fifa_1['Tweet'].astype(str)
fifa_1['Tweet'] = fifa_1['Tweet'].apply(lambda x: nltk.word_tokenize(x))
fifa_1['Tweet'] = fifa_1['Tweet'].astype(str)


# In[353]:


## removing stopwords
```

```
import nltk
import pandas as pd
nltk.download('stopwords') # Downloading the NLTK stopwords


# In[354]:


from nltk.corpus import PlaintextCorpusReader
from nltk.tokenize import TweetTokenizer
from nltk.corpus import stopwords
mus_stop = stopwords.words("english")
mus_stopwords = nltk.corpus.stopwords.words('english') #Defining lists of stopwords


# In[355]:


# Defining function to remove stopwords in tokens' list
def remove_stopwords(tokens):
      list_of_stopwords = stopwords.words('english')
      return[token for token in tokens if token.lower() not in list_of_stopwords]


# In[356]:


import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
mus_stop = stopwords.words('english')

#increasing the stopwords list to incoporate other stopwords
mus_stopwords = ['https', 'I', 'That','This','There','amp', 'It']
for i in mus_stopwords:
   mus_stop.append(i)
print(mus_stop)

#applying it into the Tweet column of fifa_1
fifa_1["Tweet"] = fifa_1["Tweet"].apply(lambda x: ' '.join([word for word in x.split() if word
not in (mus_stop)]))


# In[357]:


# removing texts less than 7 characters
# Defining function for removing shortwords from a text
def delete_shortwords(text):
   words = text.split()
   filtered_words = [word for word in words if len(word) > 7]
   filtered_text = " ".join(filtered_words)
   return filtered_text

fifa_1['Tweet'] = fifa_1['Tweet'].apply(delete_shortwords)
```

In[358]:

```
fifa_1['length_of_tweet'] = fifa_1['Tweet'].apply(len)
```

2. Wordcloud

In[359]:

```
get_ipython().system('pip install wordcloud')

from wordcloud import WordCloud
from nltk.sentiment import SentimentIntensityAnalyzer
wordcloud = WordCloud(width=600, height=400, random_state=2,
max_font_size=100).generate(' '.join(fifa_1['Tweet']))
```

In[360]:

```
import plotly.express as px
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 7))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.show()
```

3. Sentiment Analysis

In[361]:

```
get_ipython().system('pip install textblob')

nltk.download('wordnet')

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from textblob import TextBlob
```

In[362]:

```
# calculating and assigning polarity to the clean_Tweet in the fifa_1 data
fifa_1['polarity'] = fifa_1['Tweet'].apply(lambda x: TextBlob(x).sentiment.polarity)

#creating a sentiment column and assigning polarity values to positive > 0, negative < 0 and neutral=0 tweets
fifa_1['sentiment'] = fifa_1['polarity'].apply(lambda x: 'Positive' if x > 0 else ('Negative' if x < 0 else 'Neutral'))


# In[363]:


# Counting and creating the dataframe for the negative, neutral and positive sentiments
number_of_sentiment_counts = fifa_1.groupby(['sentiment'])['Tweet'].count().reset_index()
print(number_of_sentiment_counts)


# In[364]:


# Bar chart showing sentiments at the FIFA 18 World Cup
sns.set_style('whitegrid')
plt.bar(number_of_sentiment_counts['sentiment'], number_of_sentiment_counts['Tweet'], color=['magenta', 'black', 'blue'])
plt.title('Sentiment Analysis of FIFA 18 World Cup Tweets')
plt.xlabel('Sentiment')
plt.ylabel('Count')
plt.show()


# ### **Top Five Tweets**

# In[365]:


# Sort the DataFrame by number of likes and select the top 5 tweets
top_five_tweets = fifa_1.sort_values("Likes", ascending=False).head(5)

for i, tweet in top_five_tweets.iterrows():
    print(f"{tweet['Tweet']}\nLikes: {tweet['Likes']}\n")


# In[366]:


top_five_tweets['polarity'] = top_five_tweets['Tweet'].apply(lambda x: TextBlob(x).sentiment.polarity)
top_five_tweets['sentiment'] = top_five_tweets['polarity'].apply(lambda x: 'Positive' if x > 0 else ('Negative' if x < 0 else 'Neutral'))


# In[367]:
```

```
# Create a DataFrame with the counts of positive, negative, and neutral tweets
senti_Anal_count = top_five_tweets.groupby(['sentiment'])['Tweet'].count().reset_index()


# In[368]:


# plotting a bar chart for the outputted sentiment analysis
sns.set_style('whitegrid')
plt.bar(senti_Anal_count['sentiment'], senti_Anal_count['Tweet'], color=['magenta', 'black',
'blue'])
plt.title('Top Five Tweets With Most Likes')
plt.xlabel('Sentiment')
plt.ylabel('Count')
plt.show()


# In[369]:


from textblob import TextBlob

# creating the function that will calculate sentiment polarity of a text.
def mus_sentiment_score(text):
    blob = TextBlob(text)
    return blob.sentiment.polarity

# inputting the derived function to the fifa_1 dataframe
fifa_1['Sentiment_score'] = fifa_1['Tweet'].apply(mus_sentiment_score)


# In[370]:


display(fifa_1)


# # **MACHINE LEARNING MODELS**
#
# ### 1. Linear Regression

# In[371]:


# setting the seed so that it can be reproducable
import random
random.seed(1845)

import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error
```

```
input_variable =
fifa_1[['Url','RTs','polarity','Hour_of_day','length_of_tweet','Sentiment_score']]
target_variable = fifa_1[['Likes']]


# Split the data into training and testing sets
input_variable_train, input_variable_test, target_variable_train, target_variable_test =
train_test_split(input_variable, target_variable, test_size=0.20)


# Train the model
model = LinearRegression()
model.fit(input_variable, target_variable)


# Make predictions on the testing set
y_pred = model.predict(input_variable_test)

# Evaluate the model using mean squared error
mse = mean_squared_error(target_variable_test, y_pred)
print("Mean Squared Error:", mse)

# Mean Absolute Error
mae = mean_absolute_error(target_variable_test, y_pred)
print("Mean Absolute Error:", mae)

# Root Mean Squared Error
rmse = np.sqrt(mse)
print("Root Mean Squared Error:", rmse)


# ### 2. Decision Tree

# In[372]:


# # Decision Tree Model

import pandas as pd
from sklearn.tree import DecisionTreeRegressor

a = input_variable
b = target_variable
a_train, a_test, b_train, b_test = train_test_split(a, b, test_size=0.20, random_state=40)



# Train the decision tree model
decT_model = DecisionTreeRegressor(random_state=40)
decT_model.fit(a_train, b_train)


# Make predictions on the testing set
y_pred = decT_model.predict(a_test)
```

```
# Evaluate the model using mean squared error
mse = mean_squared_error(b_test, y_pred)
print("Mean Squared Error:", mse)

# Mean Absolute Error
mae = mean_absolute_error(target_variable_test, y_pred)
print("Mean Absolute Error:", mae)

# Root Mean Squared Error
rmse = np.sqrt(mse)
print("Root Mean Squared Error:", rmse)


# ### 3. Random Forest

# In[373]:


# # Random Forest Model

from sklearn.ensemble import RandomForestRegressor

# Split the data into training and testing sets
a = input_variable
b = target_variable
a_train, a_test, b_train, b_test = train_test_split(a, b, test_size=0.20, random_state=40)

# Train the Random Forest model
ranF_model = RandomForestRegressor(random_state=40)
ranF_model.fit(a_train, b_train)



# Make predictions on the testing set
y_pred = ranF_model.predict(a_test)


# Evaluate the model using Mean Squared Error
mse = mean_squared_error(b_test, y_pred)
print("Mean Squared Error:", mse)

# Mean Absolute Error
mae = mean_absolute_error(target_variable_test, y_pred)
print("Mean Absolute Error:", mae)

# Root Mean Squared Error
rmse = np.sqrt(mse)
print("Root Mean Squared Error:", rmse)
```

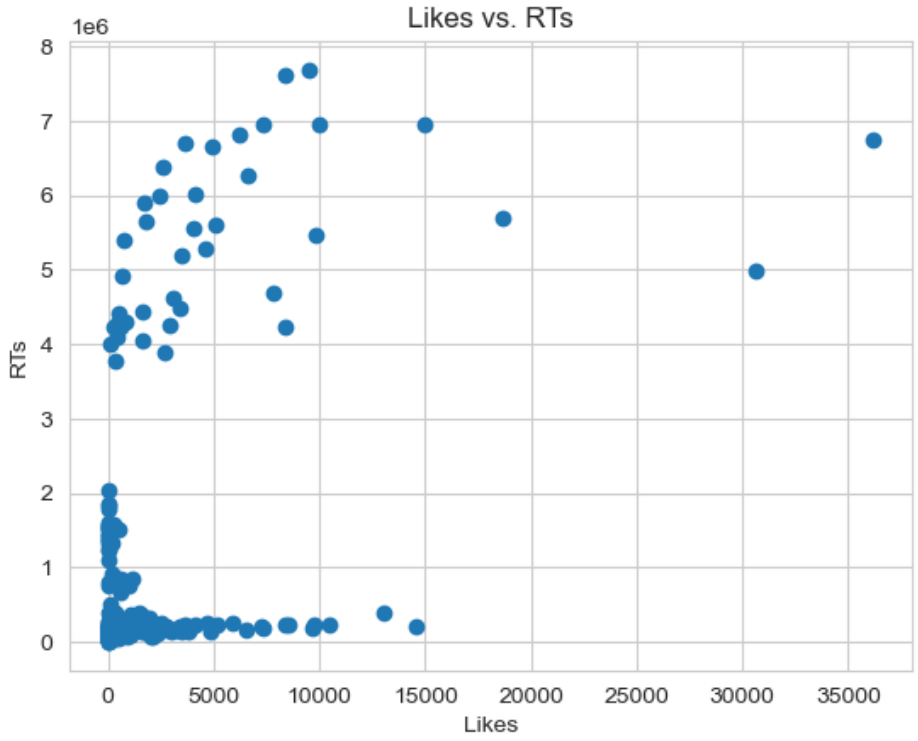
Appendix 2: Summary Statistics

```
In [333]: fifa.head
          fifa
```

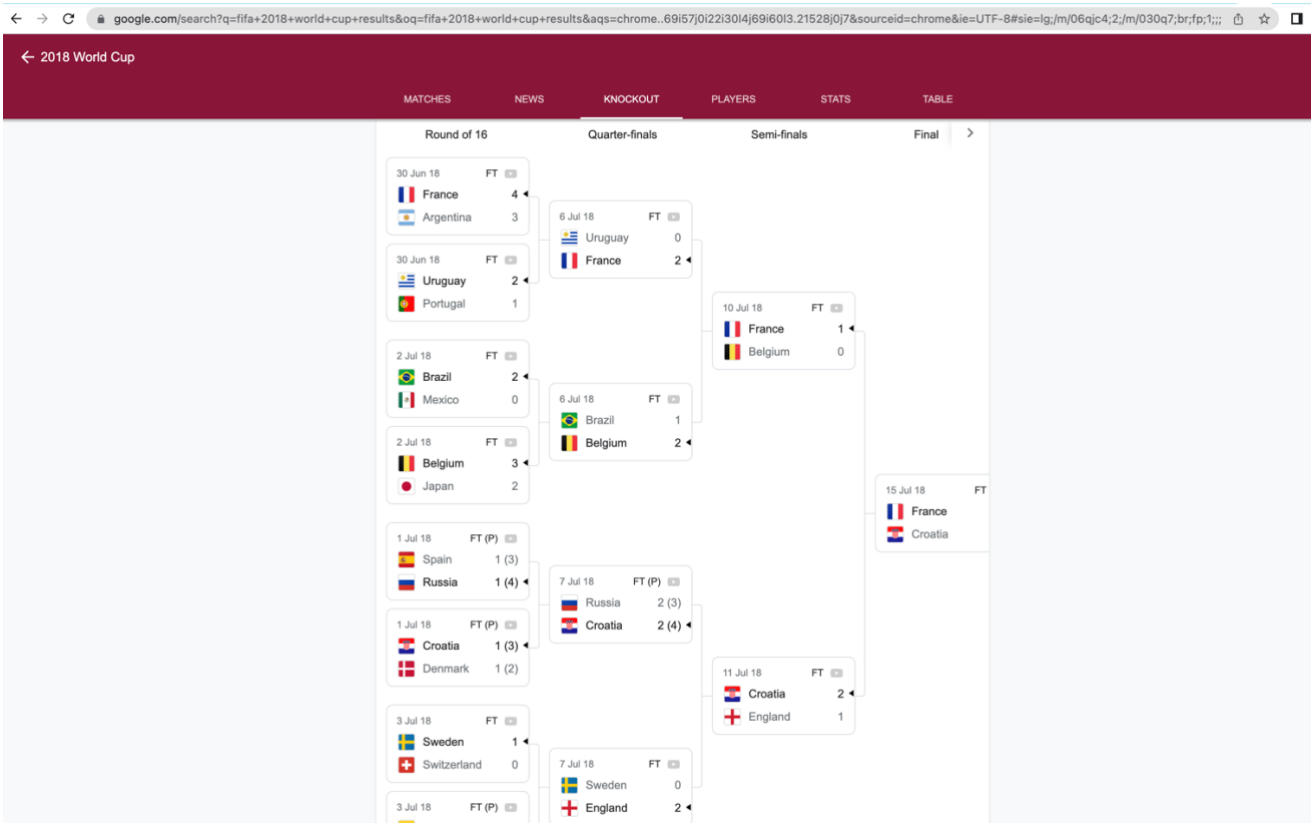
Out[333]:

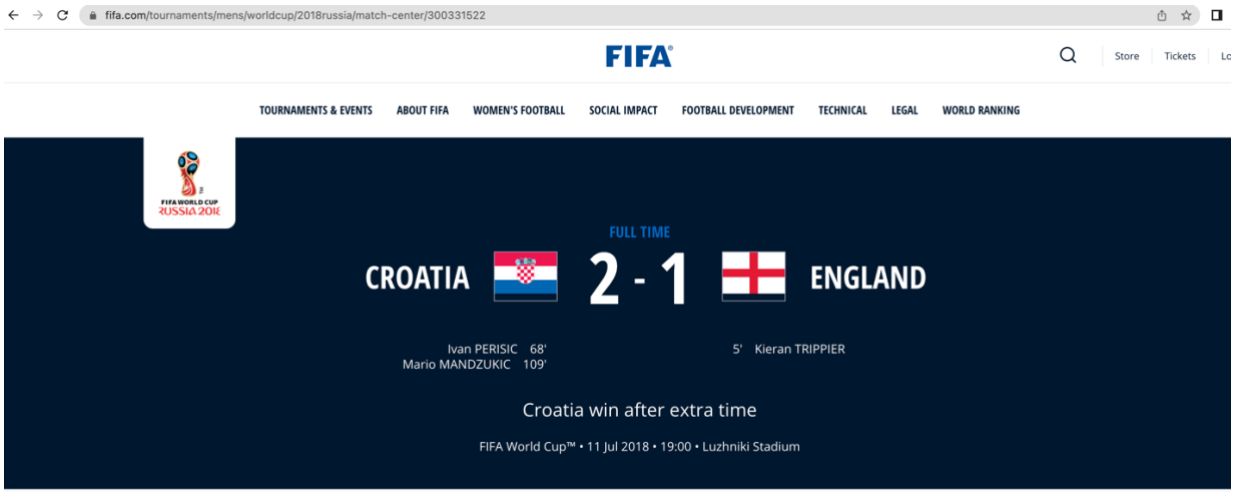
	Date	Tweet	Likes	RTs
0	02/07/2018 01:35	Only two goalkeepers have saved three penaltie...	0	477
1	02/07/2018 01:35	scores the winning penalty to send into the qu...	0	1031
2	02/07/2018 01:35	Tonight we have big game	0	488
3	02/07/2018 01:35	We get stronger Turn the music up now We got t...	0	0
4	02/07/2018 01:35	Only two goalkeepers have saved three penaltie...	0	477
...	...	...	...	...
529995	15/07/2018 22:49	France have won the FIFA in Moscow	0	63163
529996	15/07/2018 22:49	Beyonc JAY performed in blue jersey to celebra...	0	687
529997	15/07/2018 22:49	They don say immigrants are ruining France whe...	0	119
529998	15/07/2018 22:49	starts for in todays final	0	1013
529999	15/07/2018 22:49	Dear France Congratulations on winning the of ...	0	96746

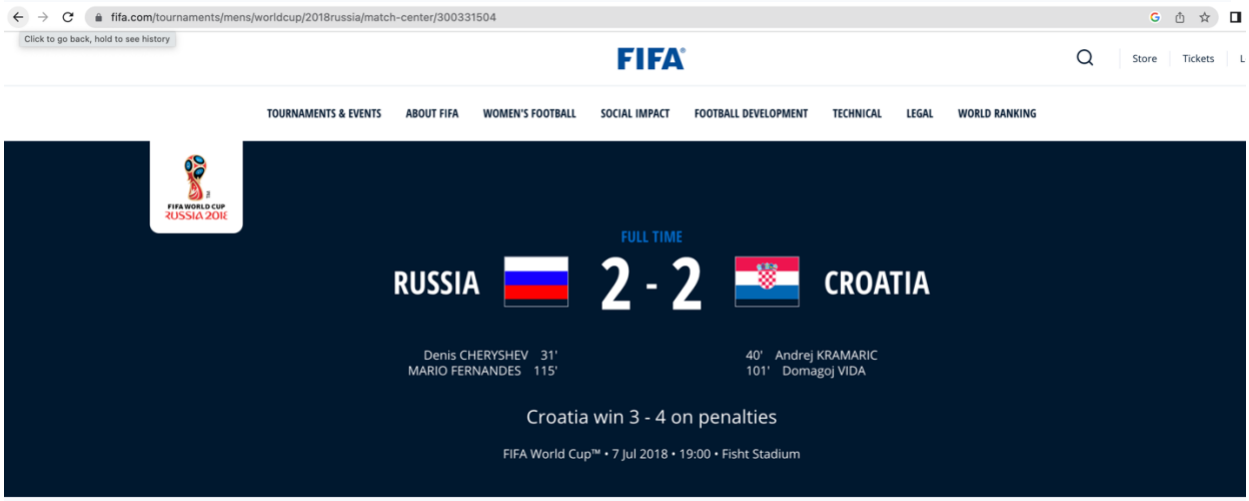
530000 rows × 4 columns



Appendix 3: FIFA 2018 World Cup Knock Out Phase Results







Appendix 4: Machine Learning Models

MACHINE LEARNING MODELS

1. Linear Regression

In [371]:
```
# setting the seed so that it can be reproducable
import random
random.seed(1845)

import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, mean_absolute_error


input_variable = fifa_1[['Url','RTs','polarity','Hour_of_day','length_of_tweet','Sentiment_score']]
target_variable = fifa_1[['Likes']]


# Split the data into training and testing sets
input_variable_train, input_variable_test, target_variable_train, target_variable_test = train_test_split(input_variabl


# Train the model
model = LinearRegression()
model.fit(input_variable, target_variable)


# Make predictions on the testing set
y_pred = model.predict(input_variable_test)

# Evaluate the model using mean squared error
mse = mean_squared_error(target_variable_test, y_pred)
print("Mean Squared Error:", mse)

# Mean Absolute Error
mae = mean_absolute_error(target_variable_test, y_pred)
print("Mean Absolute Error:", mae)

# Root Mean Squared Error
rmse = np.sqrt(mse)
print("Root Mean Squared Error:", rmse)
```

```
Mean Squared Error: 4659.047438899682
Mean Absolute Error: 4.050214120904778
Root Mean Squared Error: 68.25721528819999
```

2. Decision Tree

In [372]:
```
# # Decision Tree Model

import pandas as pd
from sklearn.tree import DecisionTreeRegressor

a = input_variable
b = target_variable
a_train, a_test, b_train, b_test = train_test_split(a, b, test_size=0.20, random_state=40)



# Train the decision tree model
decT_model = DecisionTreeRegressor(random_state=40)
decT_model.fit(a_train, b_train)


# Make predictions on the testing set
y_pred = decT_model.predict(a_test)

# Evaluate the model using mean squared error
mse = mean_squared_error(b_test, y_pred)
print("Mean Squared Error:", mse)

# Mean Absolute Error
mae = mean_absolute_error(target_variable_test, y_pred)
print("Mean Absolute Error:", mae)

# Root Mean Squared Error
rmse = np.sqrt(mse)
print("Root Mean Squared Error:", rmse)
```

```
Mean Squared Error: 3732.2526557126916
Mean Absolute Error: 3.9170764932822926
Root Mean Squared Error: 61.092165256378756
```

3. Random Forest

In [373]:
```
# # Random Forest Model

from sklearn.ensemble import RandomForestRegressor

# Split the data into training and testing sets
a = input_variable
b = target_variable
a_train, a_test, b_train, b_test = train_test_split(a, b, test_size=0.20, random_state=40)

# Train the Random Forest model
ranF_model = RandomForestRegressor(random_state=40)
ranF_model.fit(a_train, b_train)



# Make predictions on the testing set
y_pred = ranF_model.predict(a_test)


# Evaluate the model using Mean Squared Error
mse = mean_squared_error(b_test, y_pred)
print("Mean Squared Error:", mse)

# Mean Absolute Error
mae = mean_absolute_error(target_variable_test, y_pred)
print("Mean Absolute Error:", mae)

# Root Mean Squared Error
rmse = np.sqrt(mse)
print("Root Mean Squared Error:", rmse)
```

```
/var/folders/9s/tv63gxmn1r1dt_kjdy74g4wc0000gn/T/ipykernel_32259/1361925906.py:12: DataConversionWarning: A column-ve
ctor y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel
().
  ranF_model.fit(a_train, b_train)
```

```
Mean Squared Error: 2827.584242549061
Mean Absolute Error: 3.922298821542951
Root Mean Squared Error: 53.1750340154951
```

Appendix 5: Number of Twitter Users