GENDER CLASSIFIER FROM TWEET :)

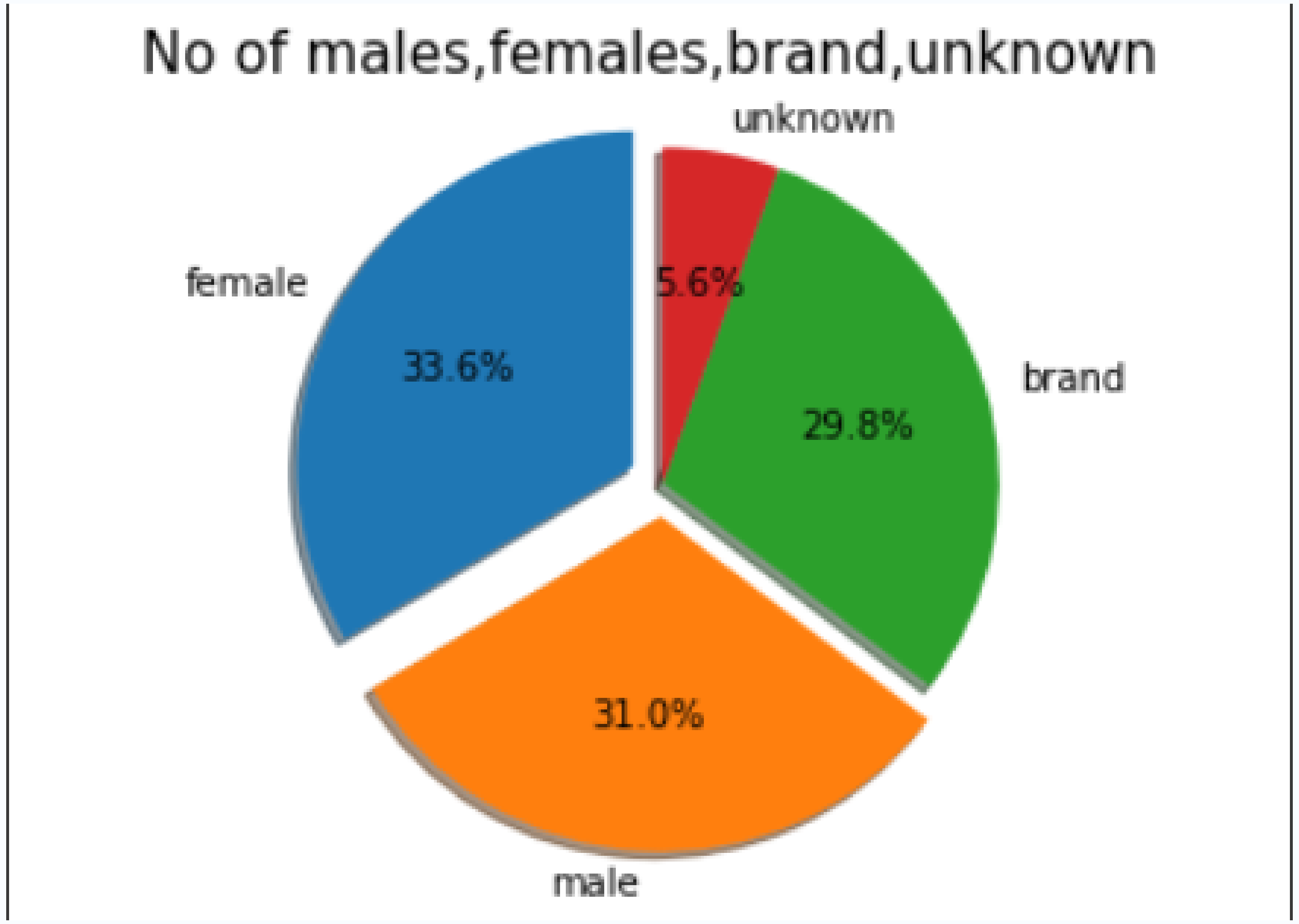
A presentation by DAGGERS

DATA-SET

A BRIEF HISTORY

The dataset consists of 20050 rows and 26 columns. Among 26 columns there are 25 predictor variables and 1 target variable which is gender in this case.

Gender and brand distribution(Pie-chart)



Focus Points

`description`	has a high cardinality: 11032 distinct values	High cardinalit
`link_color`	has a high cardinality: 2302 distinct values	High cardinalit
`sidebar_color`	has a high cardinality: 479 distinct values	High cardinalit
`text`	has a high cardinality: 12656 distinct values	High cardinalit
`retweet_count`	is highly skewed ($\gamma1$ = 92.37416976)	Skewed
`df_index`	has unique values	Unique
`fav_number`	has 1833 (13.3%) zeros	Zeros
`retweet_count`	has 13337 (96.6%) zeros	Zeros

Cleaning Dataset:

- Unnecessary columns like, '_unit_id', '_last_judgment_at', 'user_timezone', 'tweet_coord', 'tweet_count', 'tweet_created', 'tweet_id', 'tweet_location', 'profileimage', 'created' were dropped.
- Rows with unknown gender and no gender were removed.
- Profile attributes- 'profile_yn', 'profile_yn:confidence', 'profile_yn_gold' were removed as they were unavailable.
- Rows with confidence of labeling gender<100% were removed.

Manipulating Text Data:

• Text was normalized-(everything was converted to lower case, and URLs, special characters and double spaces were removed.
• The most common words which were meaningless in terms of sentiment (called stopwords) were removed.

Lemmatization:

• Words which expressed same positivity were reduced to their roots using Porter algorithm.
• Two tokenizers, a regular one and one that performs steaming, were used to break down the tweets into individual words.

Methodology

STEP 1.

```
#INSTALL ALL LIBRARIES
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import accuracy_score, confusion_matrix, classification_repo
import nltk
nltk.download('stopwords')

%matplotlib inline
```

Step 2- Read data from CSV file

```
#reading the dataset
df = pd.read_csv("gender-classifier-DFE-791531.csv",encoding="latin1") #Reading da
df.shape
```

Step 3- Information of data

```
[151] df['gender'].value_counts()

female      6700
male        6194
brand       5942
unknown     1117
Name: gender, dtype: int64

#we should remove the rows with unknown gender
drop_items_idx = df[df['gender'] == 'unknown'].index
df.drop (index = drop_items_idx, inplace = True)
df['gender'].value_counts()

female     6700
male       6194
brand      5942
Name: gender, dtype: int64

[153] print ('profile_yn information:\n',df['profile_yn'].value_counts())

df[df['profile_yn'] == 'no']['gender']
```

Null values in the dataset:

```
data.isnull().sum()
```

```
_unit_id                    0
_golden                     0
_unit_state                 0
_trusted_judgments          0
_last_judgment_at          50
gender                     97
gender:confidence          26
profile_yn                  0
profile_yn:confidence       0
created                     0
description              3744
fav_number                  0
gender_gold             20000
link_color                  0
name                        0
profile_yn_gold         20000
profileimage                0
retweet_count               0
sidebar_color               0
text                        0
tweet_coord             19891
tweet_count                 0
tweet_created               0
tweet_id                    0
tweet_location           7484
user_timezone            7798
dtype: int64
```

Data Cleaning:

Drop Unnessery Data-Colum

```python
# Drop unnecessary columns/features
df.drop (columns = ['_unit_id',
                    '_last_judgment_at',
                    'user_timezone',
                    'tweet_coord',

                    'tweet_created',
                    'tweet_id',
                    'tweet_location',
                    'profileimage',
                    'created'], inplace = True)

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20050 entries, 0 to 20049
Data columns (total 17 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   _golden                20050 non-null  bool
 1   _unit_state            20050 non-null  object
 2   _trusted_judgments     20050 non-null  int64
 3   gender                 19953 non-null  object
 4   gender:confidence      20024 non-null  float64
 5   profile_yn             20050 non-null  object
 6   profile_yn:confidence  20050 non-null  float64
 7   description            16306 non-null  object
 8   fav_number             20050 non-null  int64
 9   gender_gold            50 non-null     object
 10  link_color             20050 non-null  object
```

Getting rid of gender type — “unknown”:

```python
#we should remove the rows with unknown gender
drop_items_idx = df[df['gender'] == 'unknown'].index
df.drop (index = drop_items_idx, inplace = True)
df['gender'].value_counts()
```

```
female    6700
male      6194
brand     5942
Name: gender, dtype: int64
```

Getting rid of rows where column “profile_yn” is no:

```
[20] drop_items_idx = data[data['profile_yn'] == 'no'].index
     data.drop (index = drop_items_idx, inplace = True)
     data.drop (columns = ['profile_yn','profile_yn:confidence','profile_yn_gold'], inplace = True)
```

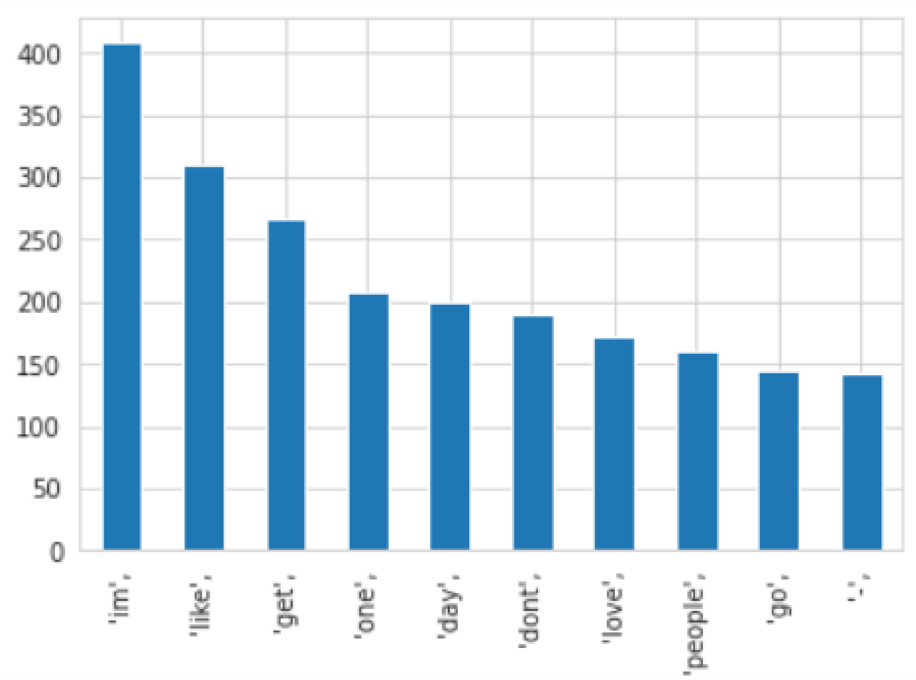
Removing Stopwords and cleanining text

```
[ ] import re
    def normalize_text(s):
        s = str(s)
        s = s.lower()
        s = re.sub('[^\x00-\x7F]+',' ',s)

        # Remove URLs
        s= re.sub('https?:\/\/.*[\r\n]*', ' ',s)

        # Remove special chars.
        s= re.sub('[?!+%{}:;.,"\'()\[\]_]', '',s)

        # Remove double spaces.
        s= re.sub('\s+',' ',s)



        return s

    df['text_norm'] = [normalize_text(s) for s in df['text']]
    #df['description_norm'] = [normalize_text(s) for s in df['description']]
```

Here we are using functions preprocessor(), remove_dup_whitespace(), tokenizer_porter(), clean_tweet, has_nan to clean, stem and tokenize the text

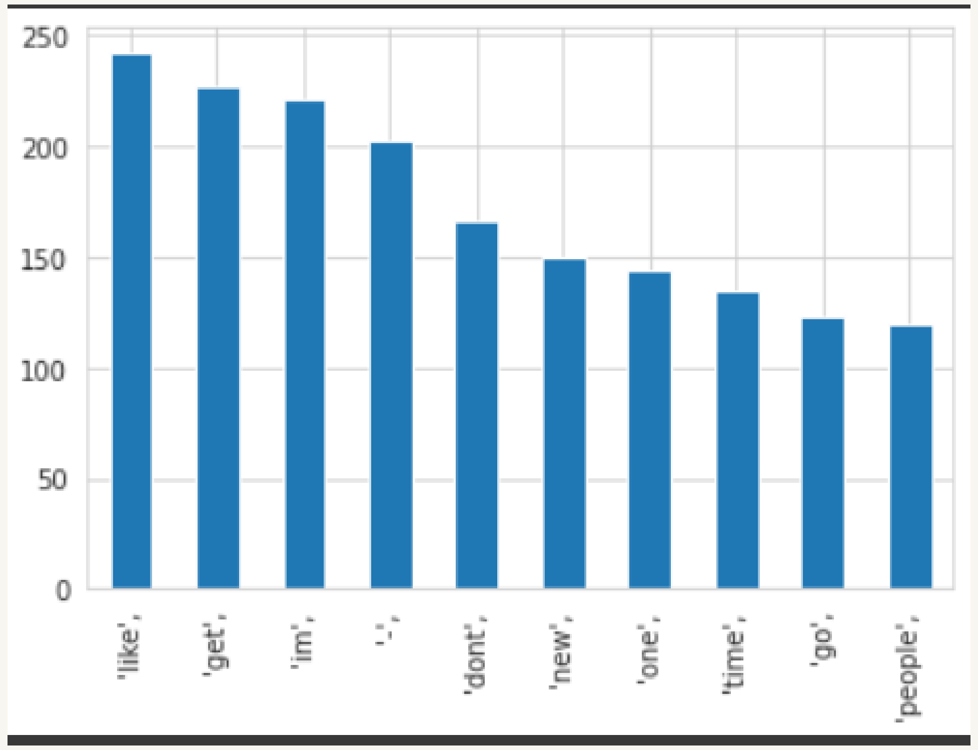
Counting the frequency of words used by female

```
'im',       408
'like',     309
'get',      267
'one',      208
'day',      200
dtype: int64
```



Counting the frequency of words used by male

```
'like',    242
'get',     227
'im',      221
'-',       202
'dont',    166
dtype: int64
```



Most common words without removing stopwords

```
[('the', 8370),
 ('and', 7964),
 ('to', 4196),
 ('I', 3229),
 ('a', 3064),
 ('of', 2741),
 ('in', 2270),
 ('you', 2173),
 ('for', 2157),
 ('The', 2018),
 ('is', 1878),
 ('on', 1621),
 ('my', 1362),
 ('it', 1205),
 ('', 1184),
 ('with', 1156),
 ('Weather', 1074),
 ('that', 1032),
 ('from', 1022),
 ('me', 1001)]
```

Most common words After removing stopwords

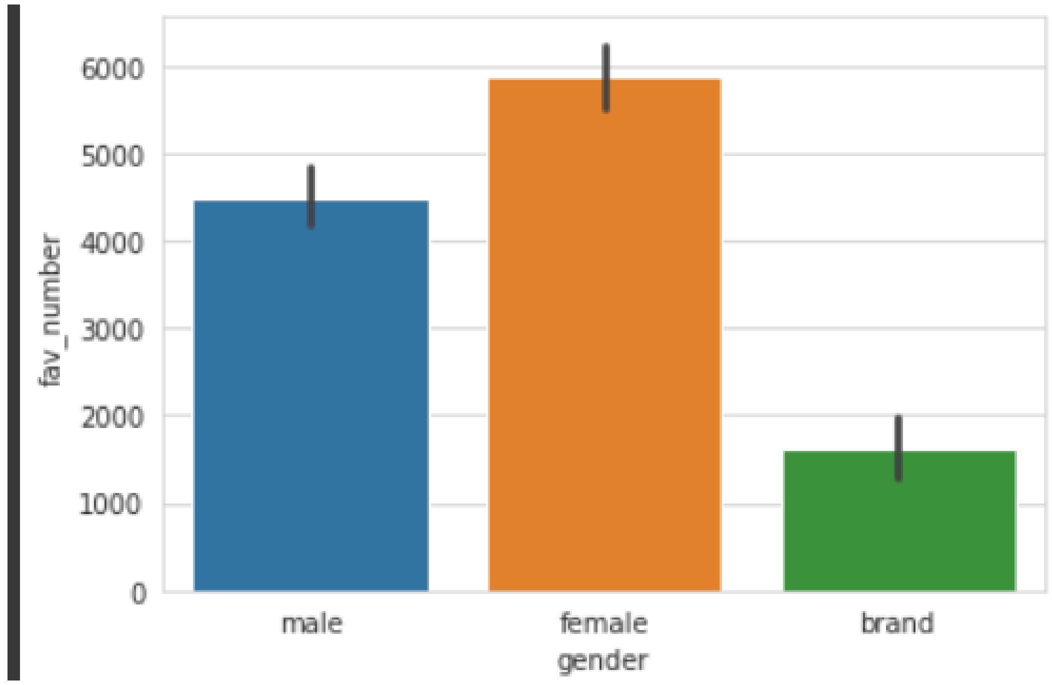
```
[('I', 3229),
 ('The', 2018),
 ('', 1184),
 ('Weather', 1074),
 ('-', 767),
 ("I'm", 651),
 ('like', 628),
 ('Get', 627),
 ('get', 570),
 ('Updates', 538),
 ('Channel.', 537),
 ('And', 487),
 ('one', 416),
 ('&amp;', 348),
 ('new', 343),
 ('love', 340),
 ('people', 315),
 ('time', 301),
 ('go', 290),
 ('know', 288)]
```

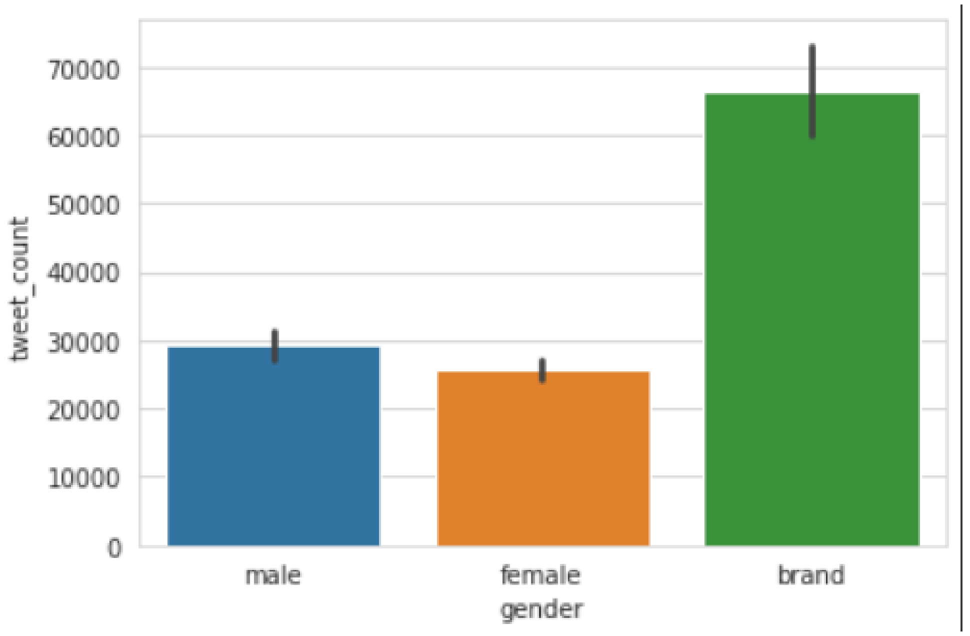
Insights From Data

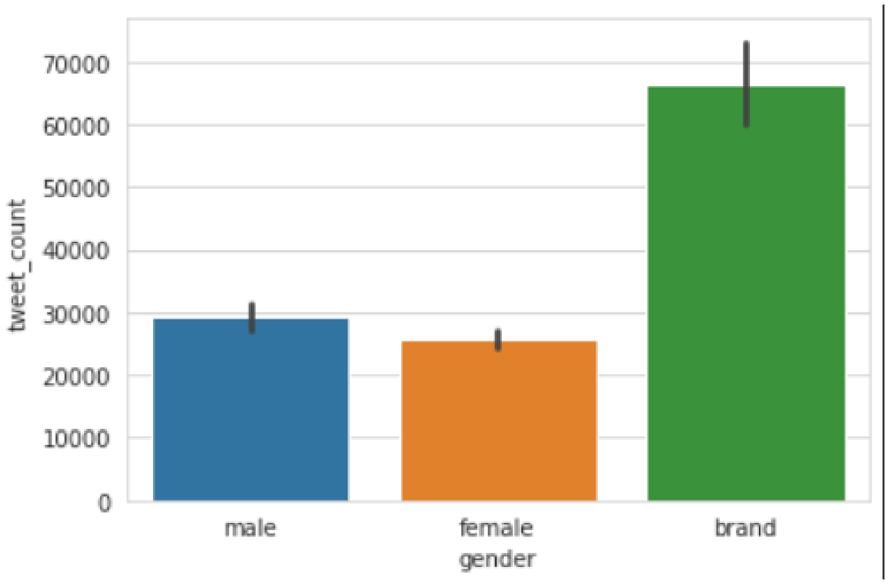
Subplots for fav_number, retweet_count, tweet_count for all ‘gender’ types:

From the figure above we notice that the retweet_count is higher for male and tweet_count for brands and fav_number for women.

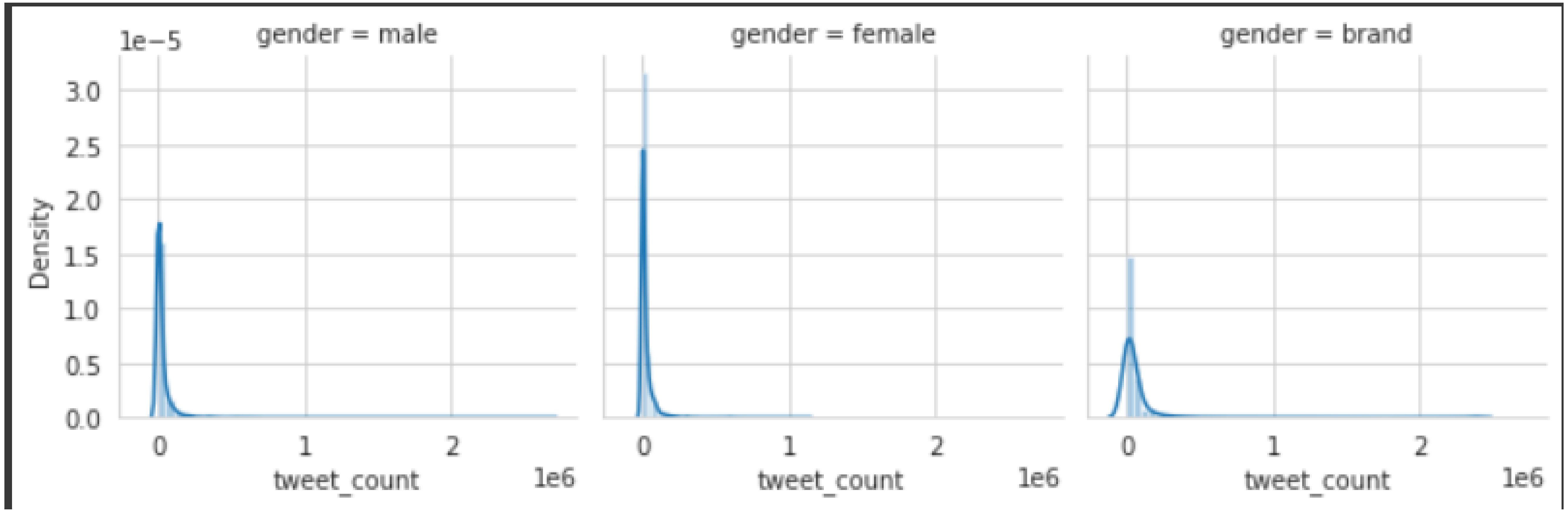
We Can notice that tweet count for female is less than the retweet they do.Unlike in brand







Density Graph of Tweet count vs Gender



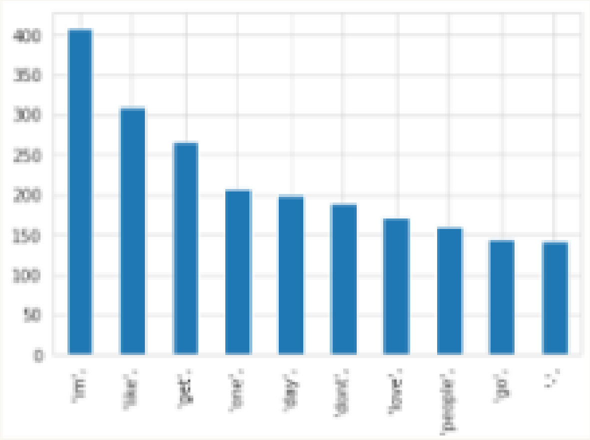
The above image shows the density for tweet count for genders male, female, and brand. The female gender has a high tweet count density in the dataset.

The density graph of female is higher than other implies when they engage on twitter they are more active than others

Counting the frequency of words used by Different Genders,

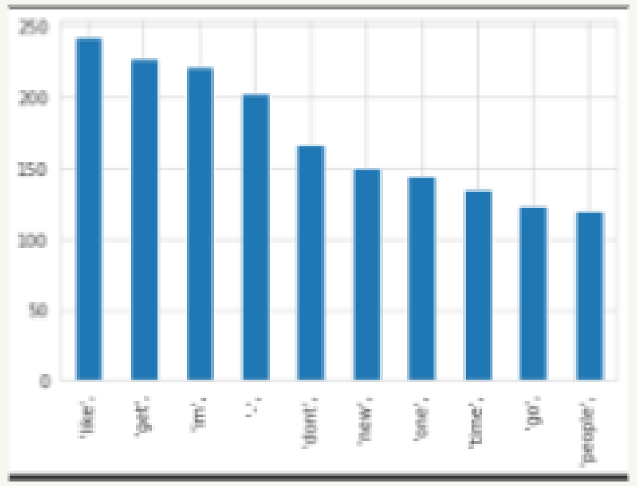
Counting the frequency of words used by female

```
'im',      408
'like',    309
'get',     267
'one',     208
'day',     200
dtype: int64
```



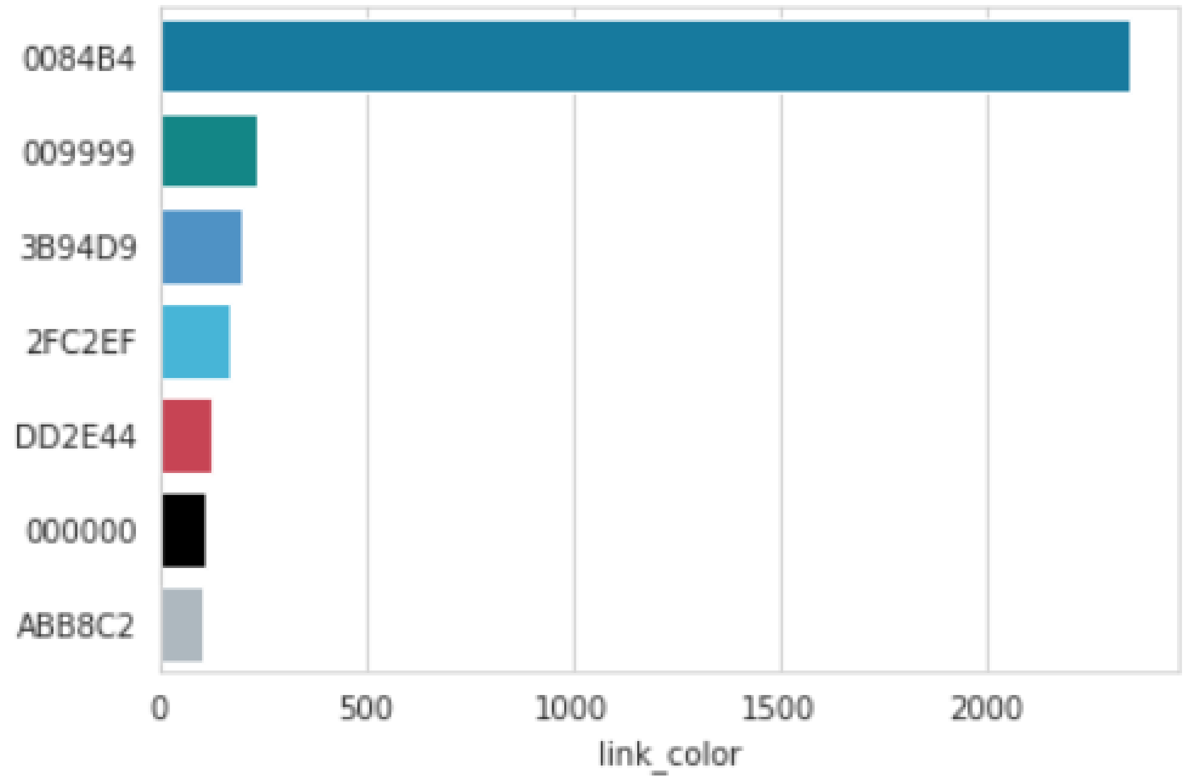
Counting the frequency of words used by male

```
'like',    242
'get',     227
'im',      221
'-',       202
'dont',    166
dtype: int64
```

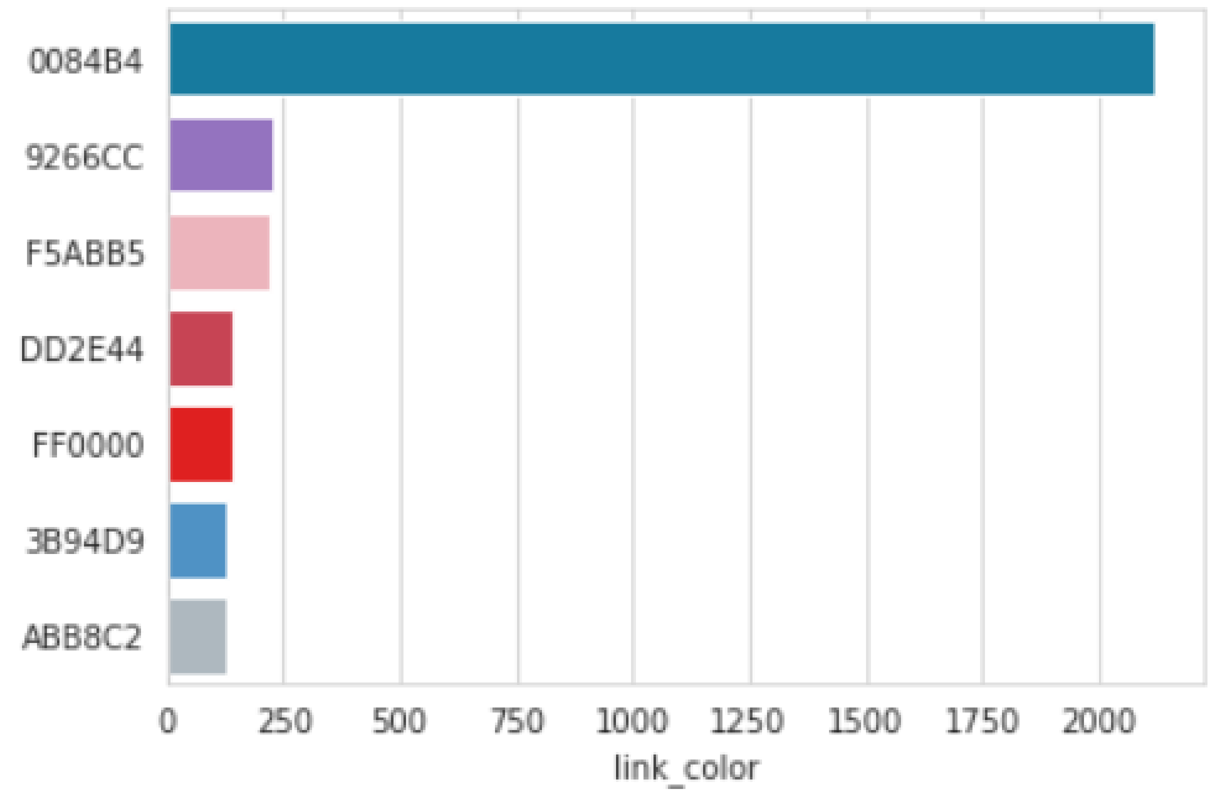


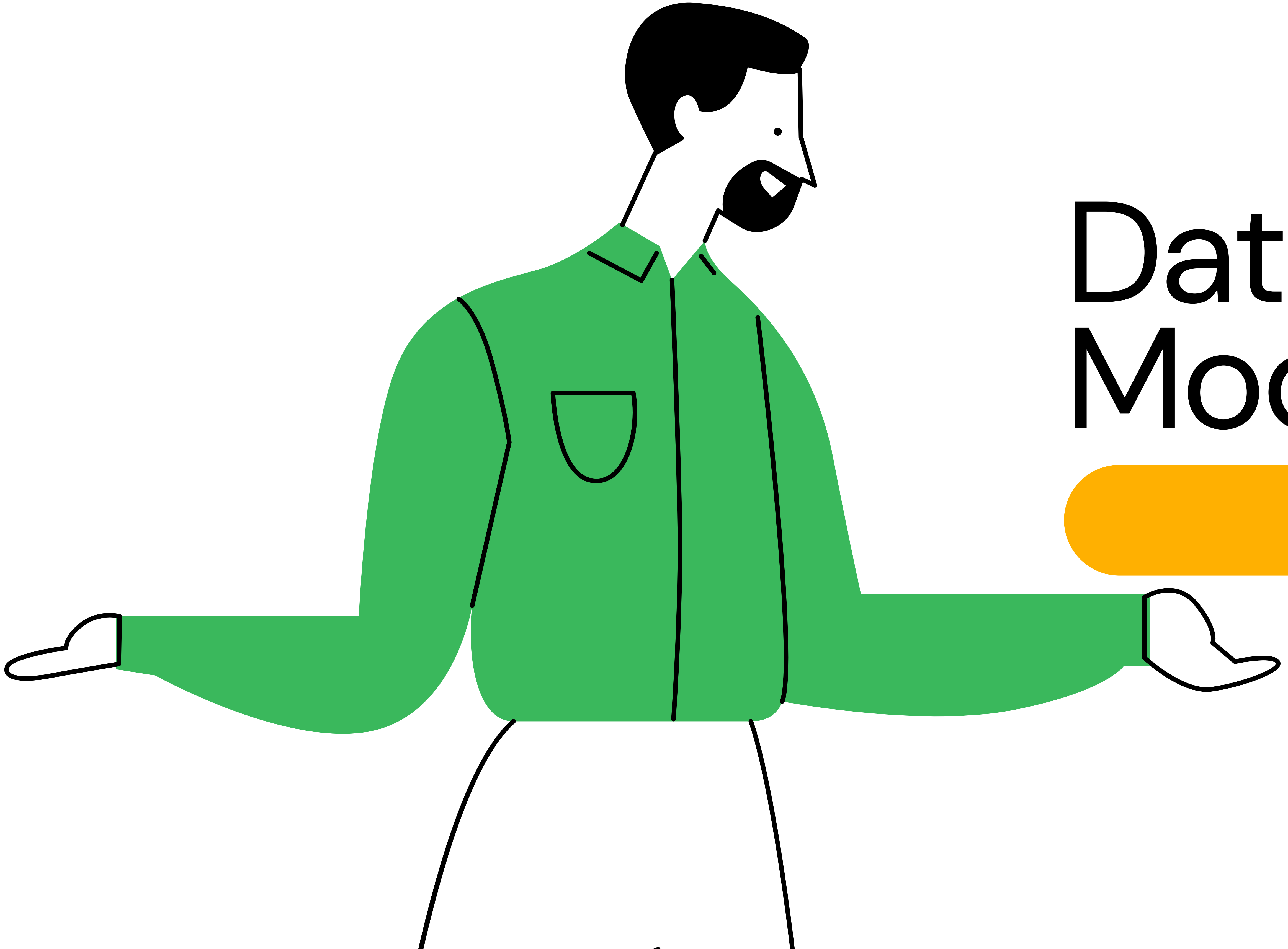
Color-Link to Gender

Male



Female





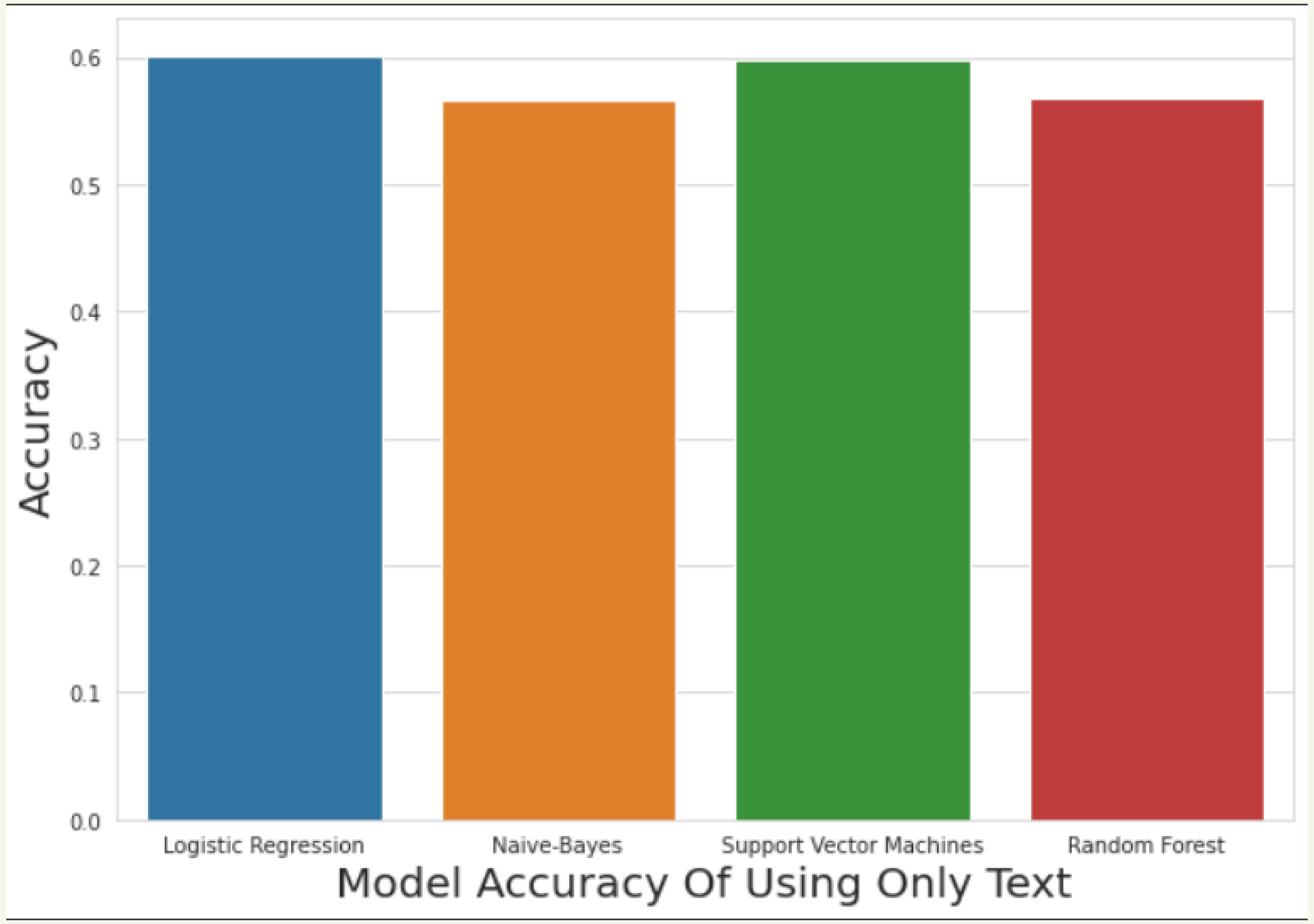
Data-Modeling

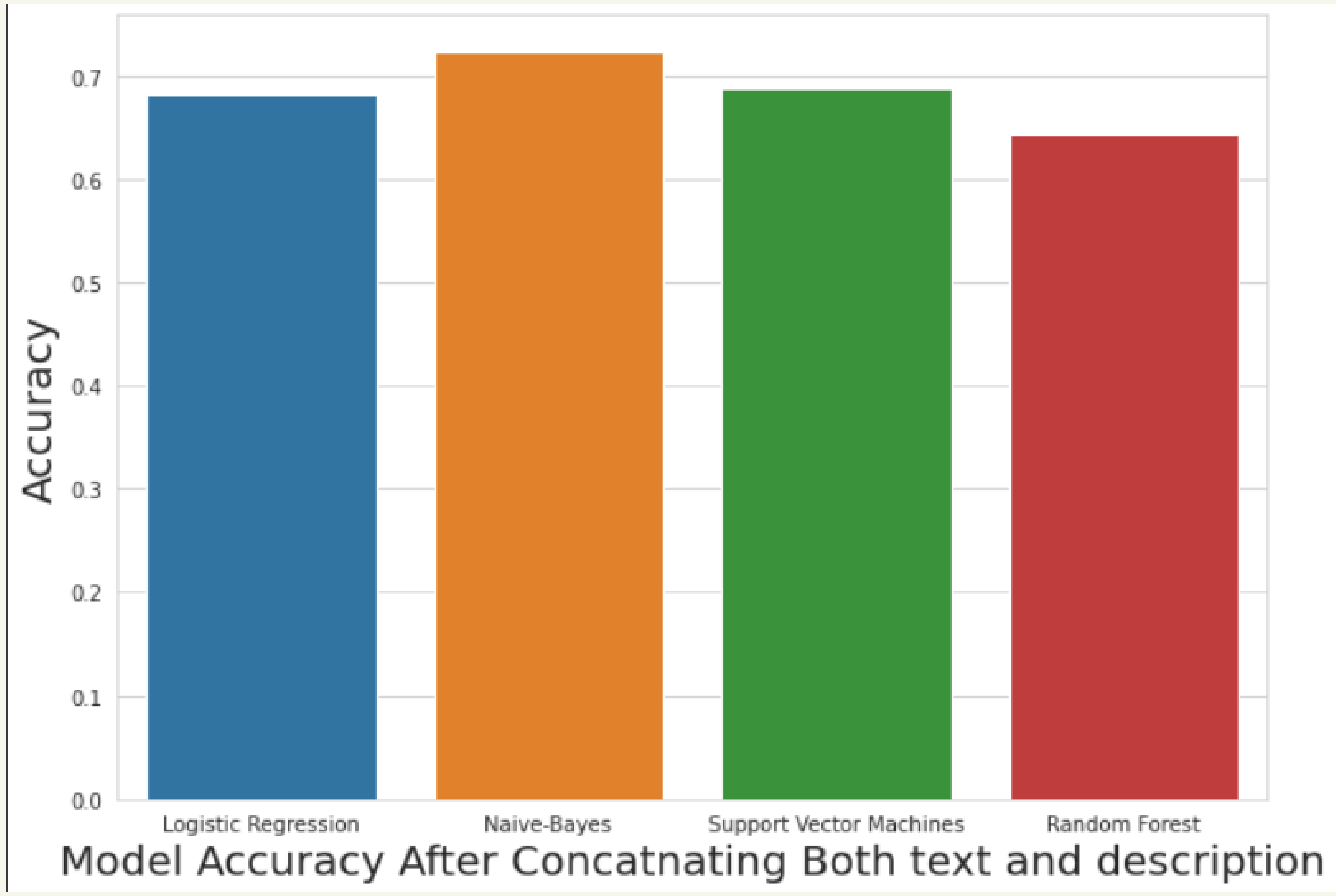
Firstly, the categorical labels were converted into numerical ones and it was encoded using LabelEncoder. The data was split into train and test.

Model-	Accuracy obtained-
Logistic Regression Model:	59.995171414775 47%
Random Forest:	56.76001931434089 %
SVM:	59.82617093191694 %
Naive Bayes	58.995171414775 47 %
.	

Best Accuracy Obtained By-Logistic Regression

Accuracy BY Model





Comparison analysis of models

Logistic regression

	precision	recall	f1-score	support
0	0.75	0.84	0.79	1136
1	0.66	0.74	0.70	1610
2	0.64	0.48	0.55	1396
accuracy			0.68	4142
macro avg	0.68	0.69	0.68	4142
eighted avg	0.68	0.68	0.67	4142

Random Forest

	precision	recall	f1-score	support
0	0.75	0.84	0.79	1136
1	0.66	0.74	0.70	1610
2	0.64	0.48	0.55	1396
accuracy			0.68	4142
macro avg	0.68	0.69	0.68	4142
weighted avg	0.68	0.68	0.67	4142

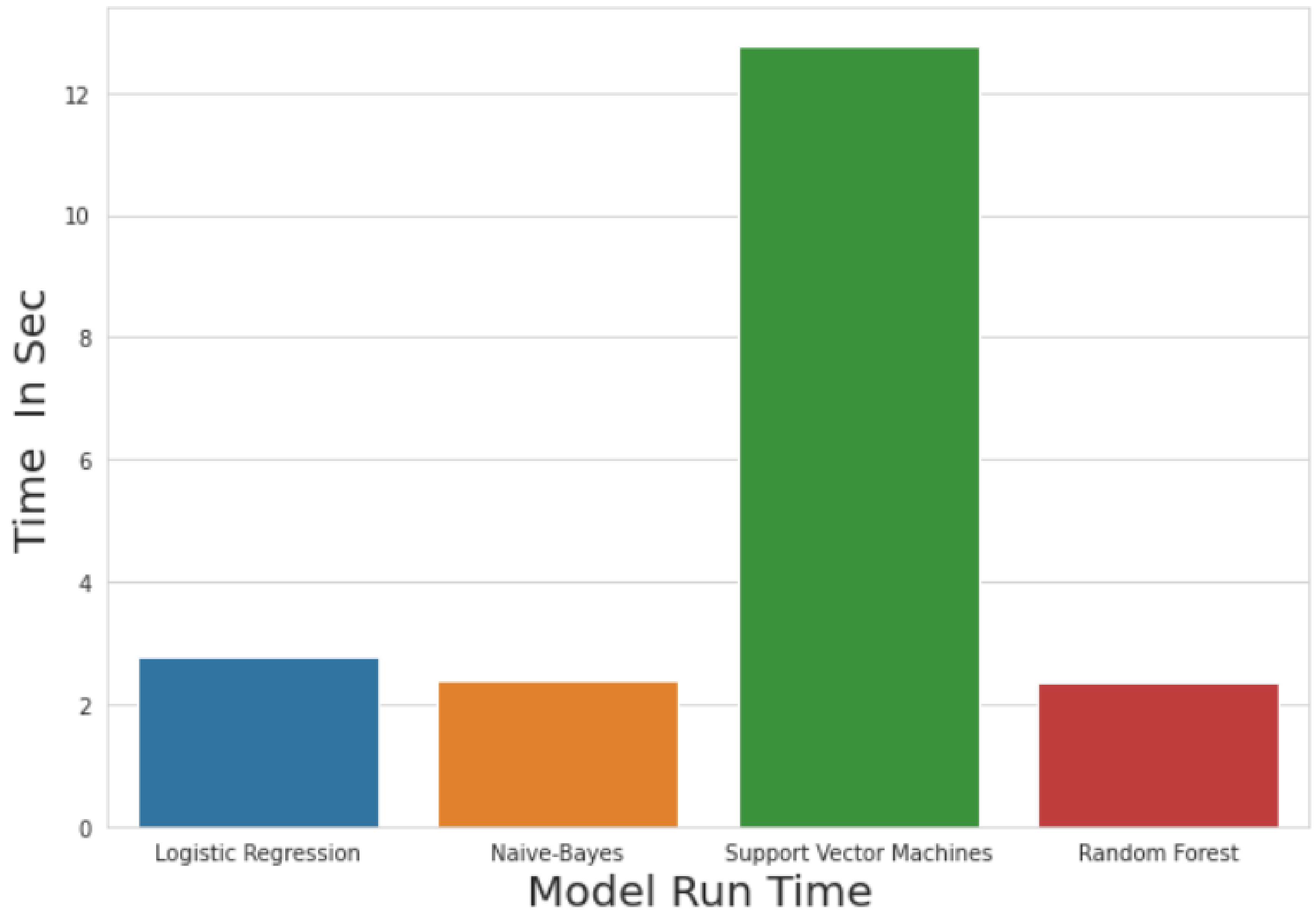
Naive-Bayes

precision	recall	f1-score	support
0.72	0.80	0.76	1077
0.73	0.63	0.68	927
		0.72	2004
0.72	0.72	0.72	2004
0.72	0.72	0.72	2004

SVM

	precision	recall	f1-score	support
0	0.76	0.84	0.80	1136
1	0.67	0.73	0.70	1610
2	0.63	0.52	0.57	1396
accuracy			0.69	4142
macro avg	0.69	0.69	0.69	4142
weighted avg	0.68	0.69	0.68	4142

Run Time



12
10
8
6
4
2
0
Time In Sec
Logistic Regression
Naive-Bayes
Support Vector Machines
Random Forest
Model Run Time

Result

SVM and Naive Bayes have approx same accuracy nearaly 68.86% accuracy scores have increased significantly when we are combining the text and the description column.

We have better prediction chances using the user's profile description and the text they are tweeting. There is no single prediction model which performs well in all the cases; however, we see that overall SVM has higher accuracy predicting the gender compared to other models.

Conclusion

Ensemble Modelling:
Ensemble technique was used to take advantage of all the three models.
Accuracy obtained: 69.87633993239981 %

Best performave is given by logistic regression considering all parameters itts accuracy is 69 and less run time comparing other ,