DATA EXPLORATION REPORT

Importing Pandas library for data analysis. Importing matplotlib and seaborn for visualization and model verification.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
color = sns.color_palette()
%matplotlib inline
```

Reading the dataset (CSV) as a pandas dataframe called as df. The dataframe now consists of the Amazon Customer Reviews dataset.

```
df = pd.read_csv("amazon_reviews.txt", delimiter = "\t")
```

The data set contains a series of other features for each review (rating, verified purchase, product category, product ID, product title, review title). The corpus is made up of 21,000 reviews, equally distributed across product categories, which have been identified as 'non-compliant' with respect to Amazon policies.

```
df.columns
```

```
Index(['DOC_ID', 'LABEL', 'RATING', 'VERIFIED_PURCHASE', 'PRODUCT_CATEGORY',
       'PRODUCT_ID', 'PRODUCT_TITLE', 'REVIEW_TITLE', 'REVIEW_TEXT'],
      dtype='object')
```

```
len(df)
```

```
21000
```

To understand the meta data, displaying the datatypes of all the columns

```
df.dtypes
```

```
DOC_ID               int64
LABEL               object
RATING               int64
VERIFIED_PURCHASE   object
PRODUCT_CATEGORY    object
PRODUCT_ID          object
PRODUCT_TITLE       object
REVIEW_TITLE        object
REVIEW_TEXT         object
dtype: object
```

DataFrame.describe() has been used to generate the descriptive statistics of the dataset. Count, mean, std, min and the quartiles. This to understand to understand the dataset better.

```
df.describe()
```

	DOC_ID	RATING
count	21000.000000	21000.000000
mean	10500.500000	4.127952
std	6062.322162	1.278333
min	1.000000	1.000000
25%	5250.750000	4.000000
50%	10500.500000	5.000000
75%	15750.250000	5.000000
max	21000.000000	5.000000

To check for null values and NaN entries, we've used insull(). This dataset is clean and has no null values.

```
df.isnull().sum()
```

```
DOC_ID              0
LABEL               0
RATING              0
VERIFIED_PURCHASE   0
PRODUCT_CATEGORY    0
PRODUCT_ID          0
PRODUCT_TITLE       0
REVIEW_TITLE        0
REVIEW_TEXT         0
dtype: int64
```

The reviews are labelled as fake or real (in the dataset they're mapped fake (__label1__) or real (__label2__)).

```
df.head()
```

	DOC_ID	LABEL	RATING	VERIFIED_PURCHASE	PRODUCT_CATEGORY	PRODUCT_ID	PRODUCT_TITLE	REVIEW_TITLE	REVIEW_TEXT
0	1	__label1__	4	N	PC	B00008NG7N	Targus PAUK10U Ultra Mini USB Keypad, Black	useful	When least you think so, this product will sav...
1	2	__label1__	4	Y	Wireless	B00LH0Y3NM	Note 3 Battery : Stalion Strength Replacement ...	New era for batteries	Lithium batteries are something new introduced...
2	3	__label1__	3	N	Baby	B000I5UZ1Q	Fisher-Price Papasan Cradle Swing, Starlight	doesn't swing very well.	I purchased this swing for my baby. She is 6 m...
3	4	__label1__	4	N	Office Products	B003822IRA	Casio MS-80B Standard Function Desktop Calculator	Great computing!	I was looking for an inexpensive desk calcolat...
4	5	__label1__	4	N	Beauty	B00PWSAXAM	Shine Whitening - Zero Peroxide Teeth Whitenin...	Only use twice a week	I only use it twice a week and the results are...

We are changing the labels from (__label1__) or (__label2__) to 0 or 1.

```
df.loc[df["LABEL"] == "__label1__", "LABEL"] = '1'
df.loc[df["LABEL"] == "__label2__", "LABEL"] = '0'
```

```
df.head()
```

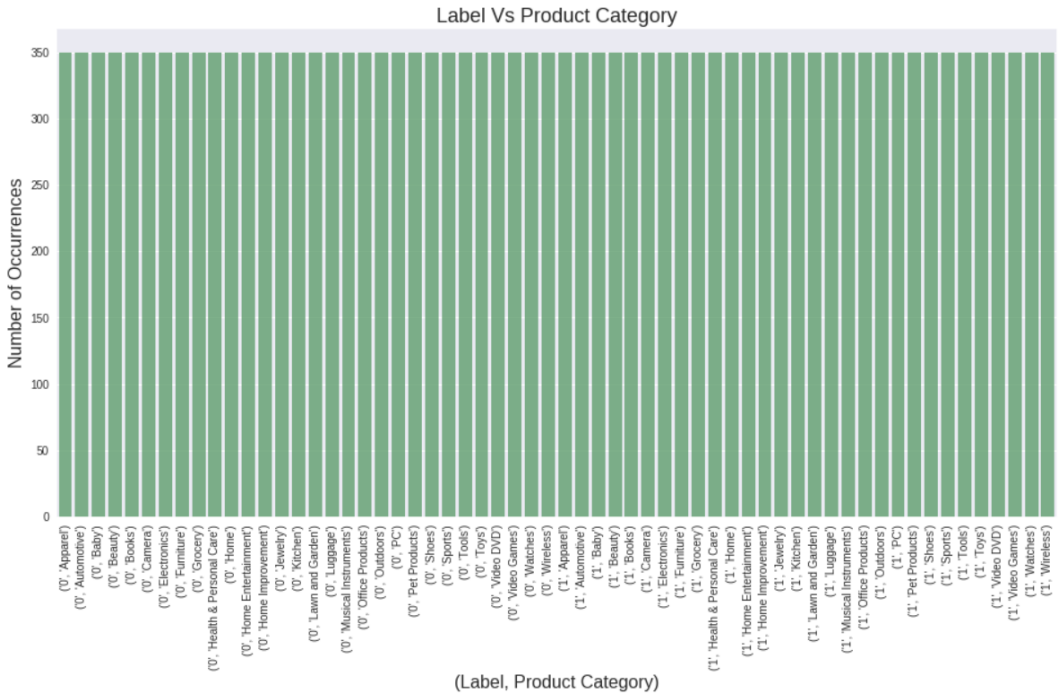
	DOC_ID	LABEL	RATING	VERIFIED_PURCHASE	PRODUCT_CATEGORY	PRODUCT_ID	PRODUCT_TITLE	REVIEW_TITLE	REVIEW_TEXT
0	1	1	4	N	PC	B00008NG7N	Targus PAUK10U Ultra Mini USB Keypad, Black	useful	When least you think so, this product will sav...
1	2	1	4	Y	Wireless	B00LH0Y3NM	Note 3 Battery : Stalion Strength Replacement ...	New era for batteries	Lithium batteries are something new introduced...
2	3	1	3	N	Baby	B000I5UZ1Q	Fisher-Price Papasan Cradle Swing, Starlight	doesn't swing very well.	I purchased this swing for my baby. She is 6 m...
3	4	1	4	N	Office Products	B003822IRA	Casio MS-80B Standard Function Desktop Calculator	Great computing!	I was looking for an inexpensive desk calcolat...
4	5	1	4	N	Beauty	B00PWSAXAM	Shine Whitening - Zero Peroxide Teeth Whitenin...	Only use twice a week	I only use it twice a week and the results are...

We are now studying the distribution of the reviews based on the Product Category

```
cnt_srs=df.groupby(df["LABEL"]).PRODUCT_CATEGORY.value_counts()
cnt_srs
```

```
LABEL  PRODUCT_CATEGORY
0      Apparel                  350
       Automotive               350
       Baby                     350
       Beauty                   350
       Books                    350
       Camera                   350
       Electronics              350
       Furniture                350
       Grocery                  350
       Health & Personal Care   350
       Home                     350
       Home Entertainment       350
       Home Improvement         350
       Jewelry                  350
       Kitchen                  350
       Lawn and Garden          350
       Luggage                  350
       Musical Instruments      350
       Office Products          350
       Outdoors                 350
       PC                       350
       Pet Products             350
       Shoes                    350
       Sports                   350
       Tools                    350
       Toys                     350
       Video DVD                350
       Video Games              350
       Watches                  350
       Wireless                 350

1      Apparel                  350
       Automotive               350
       Baby                     350
       Beauty                   350
       Books                    350
       Camera                   350
       Electronics              350
       Furniture                350
       Grocery                  350
       Health & Personal Care   350
       Home                     350
       Home Entertainment       350
       Home Improvement         350
       Jewelry                  350
       Kitchen                  350
       Lawn and Garden          350
       Luggage                  350
       Musical Instruments      350
       Office Products          350
       Outdoors                 350
       PC                       350
       Pet Products             350
       Shoes                    350
       Sports                   350
       Tools                    350
       Toys                     350
       Video DVD                350
       Video Games              350
       Watches                  350
       Wireless                 350
Name: PRODUCT_CATEGORY, dtype: int64
```

```
plt.figure(figsize=(16,8))
sns.barplot(cnt_srs.index, cnt_srs.values, alpha=0.8, color=color[1])
plt.ylabel('Number of Occurrences', fontsize=16)
plt.xlabel('(Label, Product Category)', fontsize=16)
plt.title('Label Vs Product Category', fontsize=18)
plt.xticks(rotation='vertical')
plt.show()
```

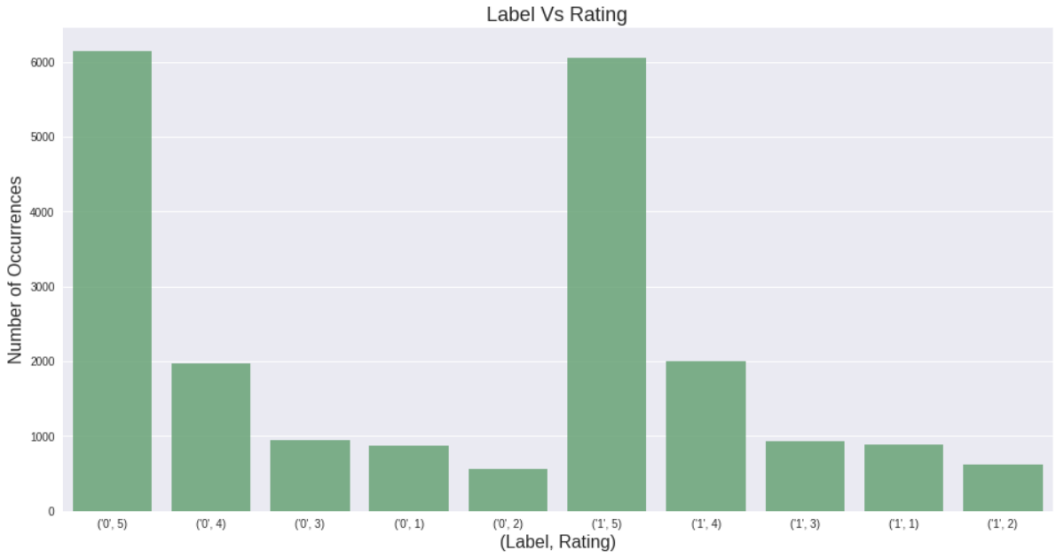


We are now studying the distribution of the reviews based on the Rating

```
cnt_srs = df.groupby(df["LABEL"]).RATING.value_counts()
cnt_srs
```

```
LABEL  RATING
0      5         6151
       4         1974
       3          942
       1          868
       2          565
1      5         6059
       4         1999
       3          926
       1          889
       2          627
Name: RATING, dtype: int64
```

```
plt.figure(figsize=(16,8))
sns.barplot(cnt_srs.index, cnt_srs.values, alpha=0.8, color=color[1])
plt.ylabel('Number of Occurrences', fontsize=16)
plt.xlabel('(Label, Rating)', fontsize=16)
plt.title('Label Vs Rating', fontsize=18)
plt.xticks(rotation='horizontal')
plt.show()
```



We are now studying the distribution of the reviews based on Verified Purchases

```
cnt_srs = data.groupby("VERIFIED_PURCHASE").LABEL.value_counts()
cnt_srs
```

```
VERIFIED_PURCHASE  LABEL
N                  1        7623
                   0        1679
Y                  0        8821
                   1        2877
Name: LABEL, dtype: int64
```

```
plt.figure(figsize=(16,8))
sns.barplot(cnt_srs.index, cnt_srs.values, alpha=0.8, color=color[1])
plt.ylabel('Number of Occurrences', fontsize=16)
plt.xlabel('(Verified Purchase (Y/N), Label)', fontsize=16)
plt.title('Verified_Purchase Vs Labels', fontsize=18)
plt.xticks(rotation='horizontal')
plt.show()
```

We are now studying the distribution of the reviews based on Text Length

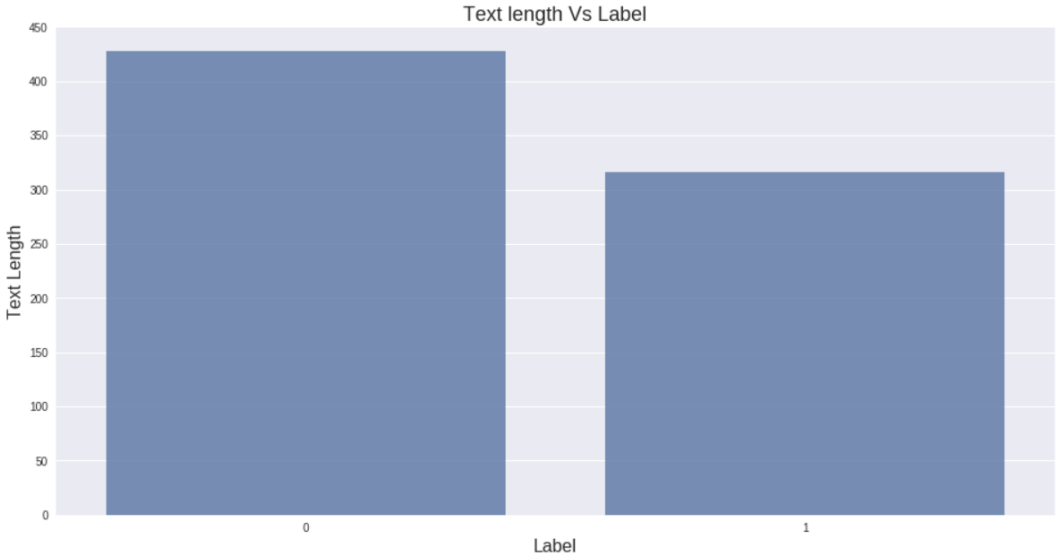
```
df1 = data.groupby("LABEL").REVIEW_TEXT
```

```
data['TEXT_LENGTH'] = data['REVIEW_TEXT'].apply(len)
```

```
cnt_srs = data.groupby(["LABEL"]).TEXT_LENGTH.agg(lambda x: sum(x)/len(x))
cnt_srs
```

```
LABEL
0    428.102857
1    316.550000
Name: TEXT_LENGTH, dtype: float64
```

```
plt.figure(figsize=(16,8))
sns.barplot(cnt_srs.index, cnt_srs.values, alpha=0.8, color=color[0])
plt.ylabel('Text Length', fontsize=16)
plt.xlabel('Label', fontsize=16)
plt.title('Text length Vs Label', fontsize=18)
plt.xticks(rotation='horizontal')
plt.show()
```



Word cloud is a technique for visualising frequent words in a text where the size of the words represents their frequency.

```
from wordcloud import WordCloud, STOPWORDS
```

```
import numpy as np
import pandas as pd
from os import path
from PIL import Image
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
```

```
rawData = df["REVIEW_TEXT"]
text=''
for x in rawData:
    text+=x
text=text.replace('br','')
```

```
wordcloud = WordCloud(width = 3000, height = 2000, random_state=1, background_color='black',
                      colormap='Set2', stopwords = STOPWORDS).generate(text)
plt.figure()
plt.imshow(wordcloud, interpolation="bilinear")
plt.axis("off")
plt.show(text)
```

Textstat is an easy-to-use library to calculate statistics from text. It helps determine readability, complexity, and grade level.

```
!pip install textstat
```

```
Requirement already satisfied: textstat in c:\users\bioni\anaconda3\lib\site-packages (0.7.0)
Requirement already satisfied: pyphen in c:\users\bioni\anaconda3\lib\site-packages (from textstat) (0.10.0)
```

```
import textstat
```

```
from textstat.textstat import textstat
df["FK_Score"] = df["REVIEW_TEXT"].apply(textstat.flesch_kincaid_grade)
```

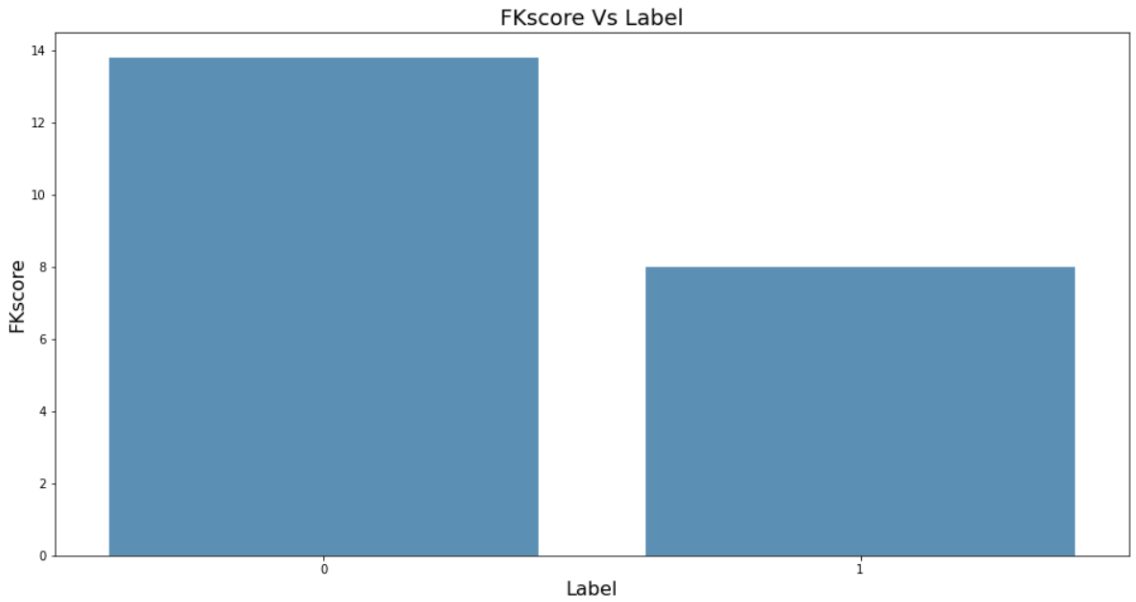
According to AWAI, FK or Flesch Kincaid is a statistical program that measures the simplicity of writing. It's a quick way to find out how easy or difficult it is to understand the writer.

```
cnt_srs = df.groupby(["LABEL"]).FK_Score.agg(lambda x: sum(x)/len(x))
cnt_srs
```

```
LABEL
0    13.803848
1     8.007886
Name: FK_Score, dtype: float64
```

```
plt.figure(figsize=(16,8))
sns.barplot(cnt_srs.index, cnt_srs.values, alpha=0.8, color=color[0])
plt.ylabel('FKscore', fontsize=16)
plt.xlabel('Label', fontsize=16)
plt.title('FKscore Vs Label', fontsize=18)
plt.xticks(rotation='horizontal')
plt.show()
```

```
C:\Users\bioni\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```



Stopwords are the English words which does not add much meaning to a sentence. They can safely be ignored without sacrificing the meaning of the sentence.

```
import nltk
wpt = nltk.WordPunctTokenizer()
stop_words = nltk.corpus.stopwords.words('english')
```

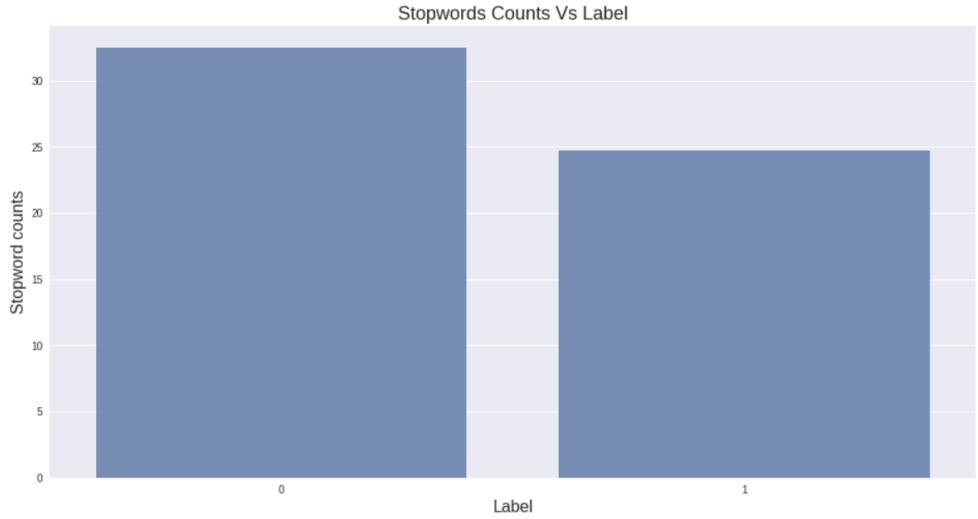
```
def stopCount(x):
    sum =0
    for char in x.split():
        sum+= char in stop_words
    return sum
df['stop_count'] = df['REVIEW_TEXT'].apply(stopCount)
```

```
cnt_srs = df.groupby(["LABEL"]).stop_count.agg(lambda x: sum(x)/len(x))
cnt_srs
```

```
LABEL
0    32.519048
1    24.696190
Name: stop_count, dtype: float64
```

```
plt.figure(figsize=(16,8))
sns.barplot(cnt_srs.index, cnt_srs.values, alpha=0.8, color=color[0])
plt.ylabel('Stopword counts', fontsize=16)
plt.xlabel('Label', fontsize=16)
plt.title('Stopwords Counts Vs Label', fontsize=18)
plt.xticks(rotation='horizontal')
plt.show()
```

```
C:\Users\bioni\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword ar
gs: x, y. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit
keyword will result in an error or misinterpretation.
  warnings.warn(
```

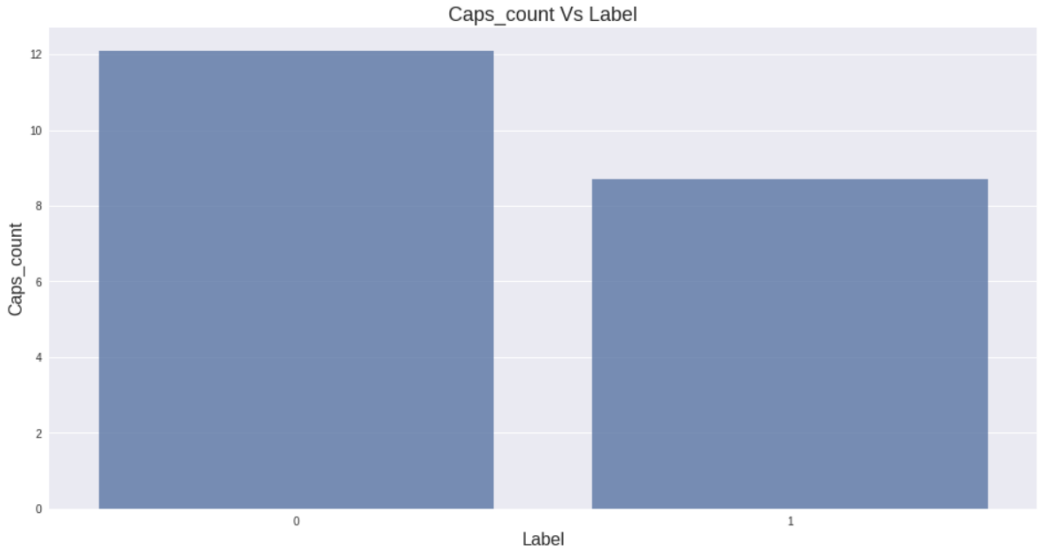


Exploring other attributes of the reviews such as Caps, Punctuation, Emoji count etc.

```
def capsCount(x):
    sum =0
    for char in x:
        sum+= char in "QWERTYUIOPASDFGHJKLZXCVBNM"
    return sum
df['caps_count'] = df['REVIEW_TEXT'].apply(capsCount)
```

```
cnt_srs = df.groupby(["LABEL"]).caps_count.agg(lambda x: sum(x)/len(x))
cnt_srs
```

```
plt.figure(figsize=(16,8))
sns.barplot(cnt_srs.index, cnt_srs.values, alpha=0.8, color=color[0])
plt.ylabel('Caps_count', fontsize=16)
plt.xlabel('Label', fontsize=16)
plt.title('Caps_count Vs Label', fontsize=18)
plt.xticks(rotation='horizontal')
plt.show()
```

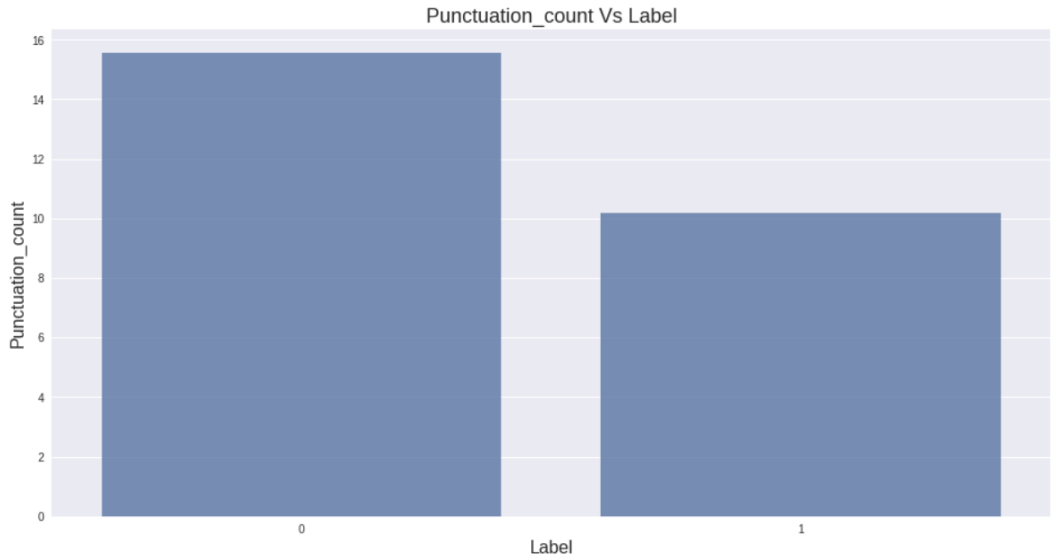


```
import string
count = lambda l1,l2: sum([1 for x in l1 if x in l2])
def punctCount(x):
    return count(x, set(string.punctuation))
df['punct_count'] = df['REVIEW_TEXT'].apply(punctCount)
```

```
cnt_srs = df.groupby(["LABEL"]).punct_count.agg(lambda x: sum(x)/len(x))
cnt_srs
```

```
LABEL
0    15.571524
1    10.182571
Name: punct_count, dtype: float64
```

```
plt.figure(figsize=(16,8))
sns.barplot(cnt_srs.index, cnt_srs.values, alpha=0.8, color=color[0])
plt.ylabel('Punctuation_count', fontsize=16)
plt.xlabel('Label', fontsize=16)
plt.title('Punctuation_count Vs Label', fontsize=18)
plt.xticks(rotation='horizontal')
plt.show()
```

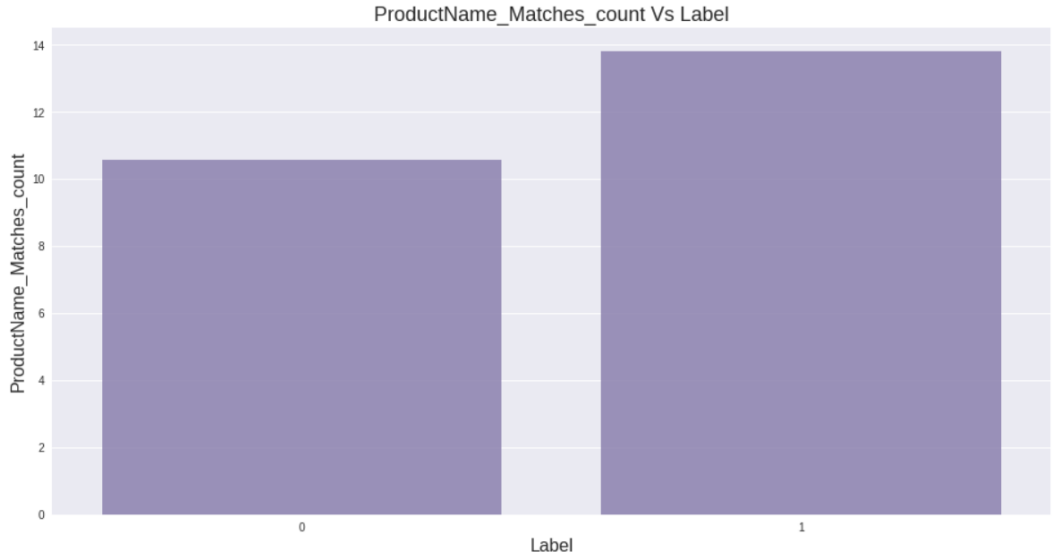



```
import re
import string
match_list = []

def checkName(title,text):
    matches = []
    for word in title.split():
        #removing punctuation
        word = "".join((char for char in word if char not in string.punctuation))
        #print(word)
        myreg = r'\b'+word+r'\b'
        r = re.compile(myreg, flags=re.I | re.X)
        matches.append(r.findall(text))
    return len(matches)


for a,b in zip(df.PRODUCT_TITLE, df.REVIEW_TEXT):
    number_of_matches = checkName(a,b)
    match_list.append(number_of_matches)

df["matchesDf"] = match_list
```

```
cnt_srs = df.groupby(["LABEL"]).matchesDf.agg(lambda x: sum(x)/len(x))
cnt_srs
```

```
LABEL
0    10.570667
1    13.823143
Name: matchesDf, dtype: float64
```

```
plt.figure(figsize=(16,8))
sns.barplot(cnt_srs.index, cnt_srs.values, alpha=0.8, color=color[3])
plt.ylabel('ProductName_Matches_count', fontsize=16)
plt.xlabel('Label', fontsize=16)
plt.title('ProductName_Matches_count Vs Label', fontsize=18)
plt.xticks(rotation='horizontal')
plt.show()
```




```
df["emojis"] = df["REVIEW_TEXT"].apply(lambda x: 1 if ";)" in x.split() or ":)" in x.split() or ":-)" in x.split() else 0)
```

```
cnt_srs = df.groupby(["LABEL"]).emojis.agg(lambda x: sum(x))
cnt_srs
```

```
LABEL
0    107
1     85
Name: emojis, dtype: int64
```

```
plt.figure(figsize=(16,8))
sns.barplot(cnt_srs.index, cnt_srs.values, alpha=0.8, color=color[3])
plt.ylabel('Emojis_count', fontsize=16)
plt.xlabel('Label', fontsize=16)
plt.title('Emojis_count Vs Label', fontsize=18)
plt.xticks(rotation='horizontal')
plt.show()
```