Sentimental Analysis for Marketing

Phase-4

We further building our project by loading the Data Set and Describing that and Cleaning the data and Visualising the distributions and Evaluation in Google colab Notebook.

Let's Import the necessary Modules and take a look at the data:

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import seaborn as sns
import math
import warnings
warnings.filterwarnings('ignore') # Hides warning
warnings.filterwarnings("ignore", category=DeprecationWarning)
warnings.filterwarnings("ignore",category=UserWarning)
sns.set_style("whitegrid") # Plotting style
np.random.seed(7) # seeding random number generator

df = pd.read_csv('amazon.csv')
print(df.head())
```

Describing the Dataset:

Overall description about the dataset should be contain in this

The purpose of this to done a overall relationship between the data and the future predictions based upon that.

```
                     id  ... reviews.username
0  AVqkIhwDv8e3D1O-lebb  ...          Adapter
1  AVqkIhwDv8e3D1O-lebb  ...           truman
2  AVqkIhwDv8e3D1O-lebb  ...            DaveZ
3  AVqkIhwDv8e3D1O-lebb  ...           Shacks
4  AVqkIhwDv8e3D1O-lebb  ...       explore42

[5 rows x 21 columns]
```

Describing the Dataset

```
1 data = df.copy()
2 data.describe()
```

	reviews.id	reviews.numHelpful	reviews.rating	reviews.userCity	reviews.userProvince
count	1.0	34131.000000	34627.000000	0.0	0.0
mean	111372787.0	0.630248	4.584573	NaN	NaN
std	NaN	13.215775	0.735653	NaN	NaN
min	111372787.0	0.000000	1.000000	NaN	NaN
25%	111372787.0	0.000000	4.000000	NaN	NaN
50%	111372787.0	0.000000	5.000000	NaN	NaN

We need to clean up the name column by referencing asins (unique products) since we have 7000 missing values:

Describing the Dataset

```
1 data = df.copy()
2 data.describe()
```

	reviews.id	reviews.numHelpful	reviews.rating	reviews.userCity	reviews.userProvince
count	1.0	34131.000000	34627.000000	0.0	0.0
mean	111372787.0	0.630248	4.584573	NaN	NaN
std	NaN	13.215775	0.735653	NaN	NaN
min	111372787.0	0.000000	1.000000	NaN	NaN
25%	111372787.0	0.000000	4.000000	NaN	NaN
50%	111372787.0	0.000000	5.000000	NaN	NaN
75%	111372787.0	0.000000	5.000000	NaN	NaN
max	111372787.0	814.000000	5.000000	NaN	NaN

```
1 data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
1 data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34660 entries, 0 to 34659
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   id                    34660 non-null  object
 1   name                  27900 non-null  object
 2   asins                 34658 non-null  object
 3   brand                 34660 non-null  object
 4   categories            34660 non-null  object
 5   keys                  34660 non-null  object
 6   manufacturer          34660 non-null  object
 7   reviews.date          34621 non-null  object
 8   reviews.dateAdded     24039 non-null  object
 9   reviews.dateSeen      34660 non-null  object
 10  reviews.didPurchase   1 non-null      object
 11  reviews.doRecommend   34066 non-null  object
 12  reviews.id            1 non-null      float64
 13  reviews.numHelpful    34131 non-null  float64
 14  reviews.rating        34627 non-null  float64
 15  reviews.sourceURLs    34660 non-null  object
 16  reviews.text          34659 non-null  object
 17  reviews.title         34655 non-null  object
 18  reviews.userCity      0 non-null      float64
 19  reviews.userProvince  0 non-null      float64
 20  reviews.username      34658 non-null  object
dtypes: float64(5), object(16)
memory usage: 5.6+ MB
```

```
data["asins"].unique()
```

```
array(['B01AHB9CN2', 'B00VINDBJK', 'B005PB2T0S', 'B002Y27P3M',
       'B01AHB9CYG', 'B01AHB9C1E', 'B01J2G4VBG', 'B00ZV9PXP2',
       'B0083Q04TA', 'B018Y229OU', 'B00REQKWGA', 'B00IOYAM4I',
       'B018T075DC', nan, 'B00DU15MU4', 'B018Y225IA', 'B005PB2T2Q',
       'B018Y23MNM', 'B00OQVZDJM', 'B00IOY8XWQ', 'B00LO29KXQ',
       'B00QJDU3KY', 'B018Y22C2Y', 'B01BFIBRIE', 'B01J4ORNHU',
       'B018SZT3BK', 'B00UH4D8G2', 'B018Y22BI4', 'B00TSUGXKE',
       'B00L9EPT8O,B01E6AO69U', 'B018Y23P7K', 'B00X4WHP5E', 'B00QFQRELG',
       'B00LW9XOJM', 'B00QL1ZN3G', 'B0189XYY0Q', 'B01BH83OOM',
       'B00BFJAHF8', 'B00U3FPN4U', 'B002Y27P6Y', 'B006GWO5NE',
       'B006GWO5WK'], dtype=object)
```

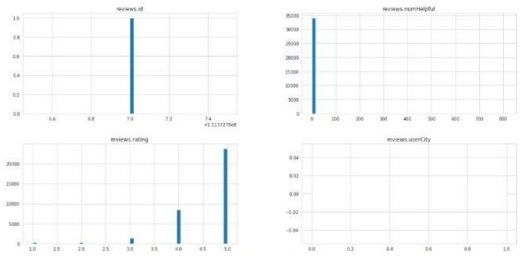
```
asins_unique = len(data["asins"].unique())
print("Number of Unique ASINs: " + str(asins_unique))
```

#Output– Number of Unique ASINs: 42

Visualizing the distributions of numerical variables:

Visualizing the distributions of numerical variables:

```
data.hist(bins=50, figsize=(20,15))
plt.show()
```



```
from sklearn.model_selection import StratifiedShuffleSplit
print("Before {}".format(len(data)))
dataAfter = data.dropna(subset=["reviews.rating"])
# Removes all NAN in reviews.rating
print("After {}".format(len(dataAfter)))
dataAfter["reviews.rating"] = dataAfter["reviews.rating"].astype(int)

split = StratifiedShuffleSplit(n_splits=5, test_size=0.2)
for train_index, test_index in split.split(dataAfter,
                                           dataAfter["reviews.rating"]):
    strat_train = dataAfter.reindex(train_index)
    strat_test = dataAfter.reindex(test_index)
```

#Output-

Before 34660
After 34627

Outliers in this case are valuable, so we may want to weight reviews that had more than 50+ people who find them helpful.

Majority of examples were rated highly (looking at rating distribution). There is twice amount of 5 star ratings than the others ratings combined.

Split the data into Train and Test:

Lets see all the different names for this product that have 2 ASINs:

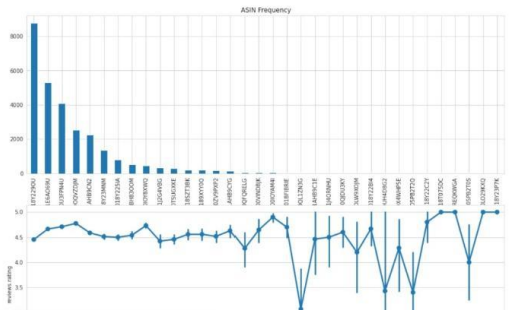
```
different_names = reviews[reviews["asins"] ==
                          "B00L9EPT8O,B01E6AO69U"]["name"].unique()
for name in different_names:
    print(name)
print(reviews[reviews["asins"] == "B00L9EPT8O,B01E6AO69U"]["name"].value_co
```

```
#Output
Echo (White),,,
Echo (White),,,
Amazon Fire Tv,,,
Amazon Fire Tv,,,
nan
Amazon - Amazon Tap Portable Bluetooth and Wi-Fi Speaker - Black,,,
Amazon - Amazon Tap Portable Bluetooth and Wi-Fi Speaker - Black,,,
Amazon Fire Hd 10 Tablet, Wi-Fi, 16 Gb, Special Offers - Silver Aluminum,,,
Amazon Fire Hd 10 Tablet, Wi-Fi, 16 Gb, Special Offers - Silver Aluminum,,,
Amazon 9W PowerFast Official OEM USB Charger and Power Adapter for Fire Tab
```

Data Exploration (Training Set):

We will use regular expressions to clean out any unfavorable characters in the dataset, and then preview what the data looks like after cleaning.

```
plt.show()
```



Sentimental Analysis:

Using the features in place, we will build a classifier that can determine a review's sentiment.

```
def sentiments(rating):
    if (rating == 5) or (rating == 4):
        return "Positive"
    elif rating == 3:
        return "Neutral"
    elif (rating == 2) or (rating == 1):
        return "Negative"
# Add sentiments to the data
strat_train["Sentiment"] = strat_train["reviews.rating"].apply(sentiments)
strat_test["Sentiment"] = strat_test["reviews.rating"].apply(sentiments)
print(strat_train["Sentiment"][:20])
```

#Output-

Output:

#Output-

```
4349     Positive
30776    Positive
28775     Neutral
1136     Positive
17803    Positive
7336     Positive
32638    Positive
13995    Positive
6728     Negative
22009    Positive
11047    Positive
22754    Positive
5578     Positive
11673    Positive
19168    Positive
14903    Positive
30843    Positive
5440     Positive
28940    Positive
31258    Positive
```


Team Members

I.JaharaAsmi
M.Muthulakshmi
K.Sandhiya
M.Shajitha Barveen
M.SathiyaJothi

JP College Of Engineering