POLYTECHNIC UNIVERSITY OF THE PHILIPPINES

SENTIMENT ANALYSIS IN PYTHON

In Partial Fulfillment of the Requirements for the
Bachelor of Science in Computer Engineering

By:
ESCOBAL, Anne Margaret O.

To:
Engr. Simon Salvador E. Tidon

2022

TABLE OF CONTENTS

INTRODUCTION

Sentiment analysis is the process of classifying whether a block of text is positive, negative, or, neutral. Sentiment analysis is contextual mining of words which indicates the social sentiment of a brand and also helps the business to determine whether the product which they are manufacturing is going to make a demand in the market or not. The goal which Sentiment analysis tries to gain is to analyze people's opinion in a way that it can help the businesses expand.

Python sentiment analysis is a methodology for analyzing a piece of text to discover the sentiment hidden within it. It accomplishes this by combining machine learning and natural language processing (NLP). Sentiment analysis allows you to examine the feelings expressed in a piece of text.

LIBRARIES USED

- **Plotly**

  The Plotly Python library is an interactive open-source library. This can be a very helpful tool for data visualization and understanding the data simply and easily. plotly graph objects are a high-level interface to plotly which are easy to use. It can plot various types of graphs and charts like scatter plots, line charts, bar charts, box plots, histograms, pie charts, etc.

- **Pandas**

Python Pandas is defined as an open-source library that provides high-performance data manipulation in Python.

- **Matplotlib**

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack.

- **Seaborn**

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

- **NLTK (National Language Toolkit)**

NLTK is a leading platform for building Python programs to work with human language data.

- **WORDCLOUD**

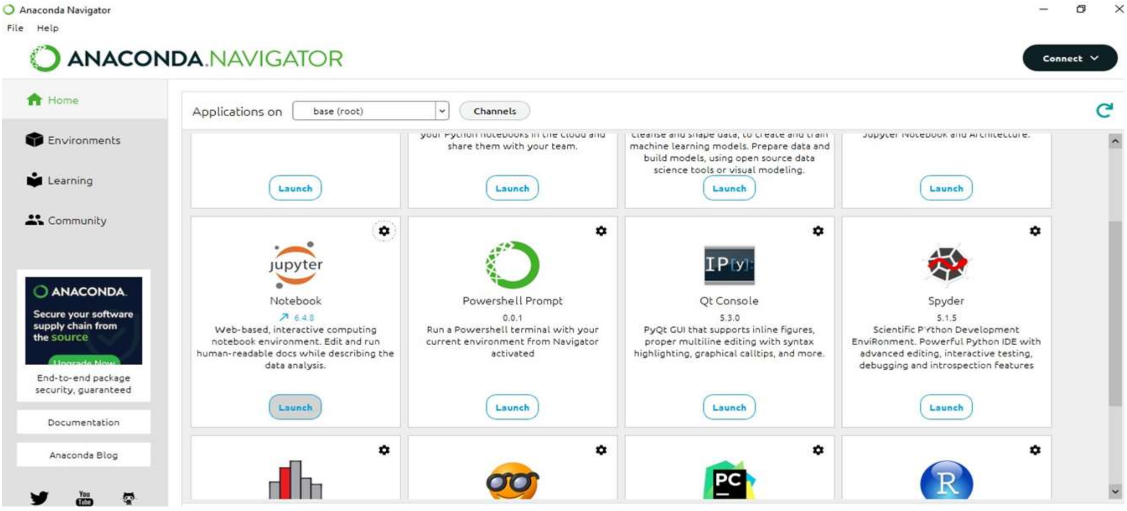
Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance. Significant textual data points can be highlighted using a word cloud. Word clouds are widely used for analyzing data from social network websites.

- **SKLEARN**

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python.

Steps to implement Sentiments in Python

1. Launch Jupyter from Anaconda Navigator.



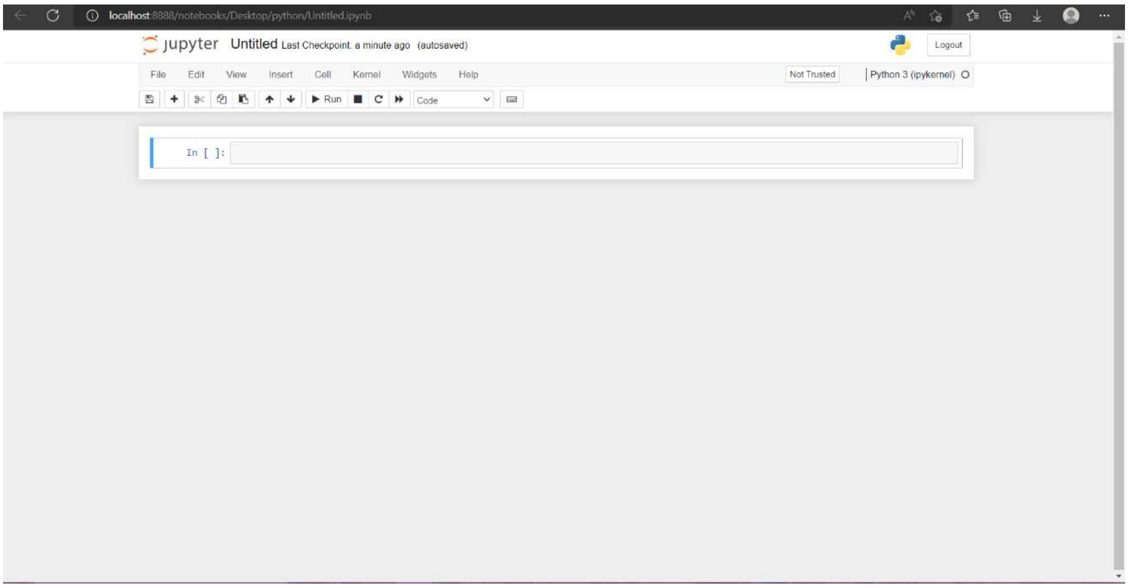
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Run a Powershell terminal with your current environment from Navigator activated

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

2. Create a new Python Notebook.

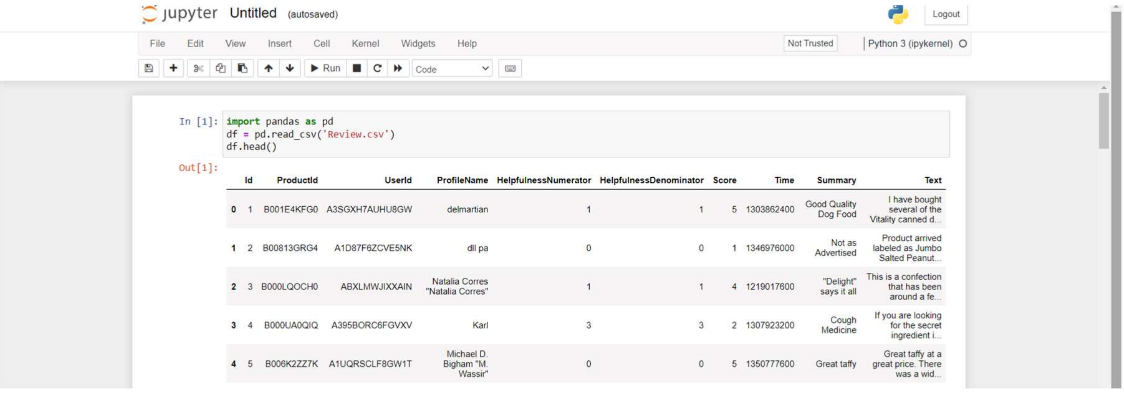

localhost:8888/notebooks/Desktop/python/Untitled.ipynb
jupyter Untitled Last Checkpoint: a minute ago (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [ ]:

3. Import modules/libraries needed.

- pip install plotly==5.8.0
- pip install pandas
- pip install matplotlib
- pip install seaborn
- pip install nltk
- pip install wordcloud
- pip install sklearn

4. Analysis

I used Review.csv file from Kaggle's Amazon Fine Food Reviews dataset to perform the analysis.


jupyter Untitled (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code

```
In [1]: import pandas as pd
        df = pd.read_csv('Review.csv')
        df.head()
```

Out[1]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d...
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"Delight" says it all	This is a confection that has been around a fe...
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine	If you are looking for the secret ingredient i...
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Great taffy	Great taffy at a great price. There was a wid...

We can see that the data frame contains some product, user and review information.

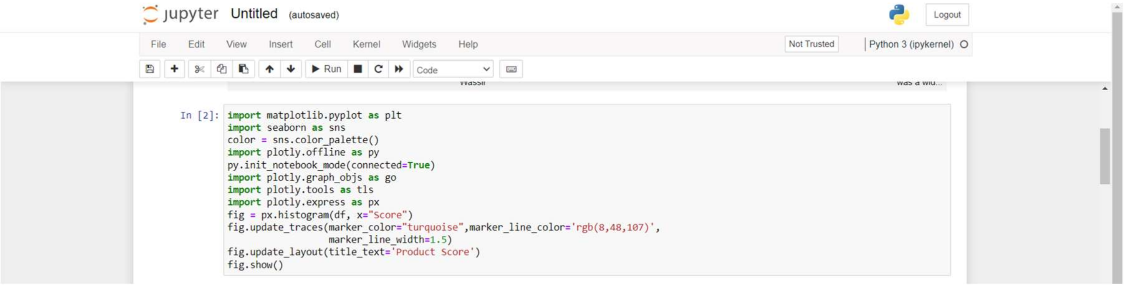
The data that we will be using most for this analysis is "*Summary", "Text",* and "*Score."*

Text — This variable contains the complete product review information.

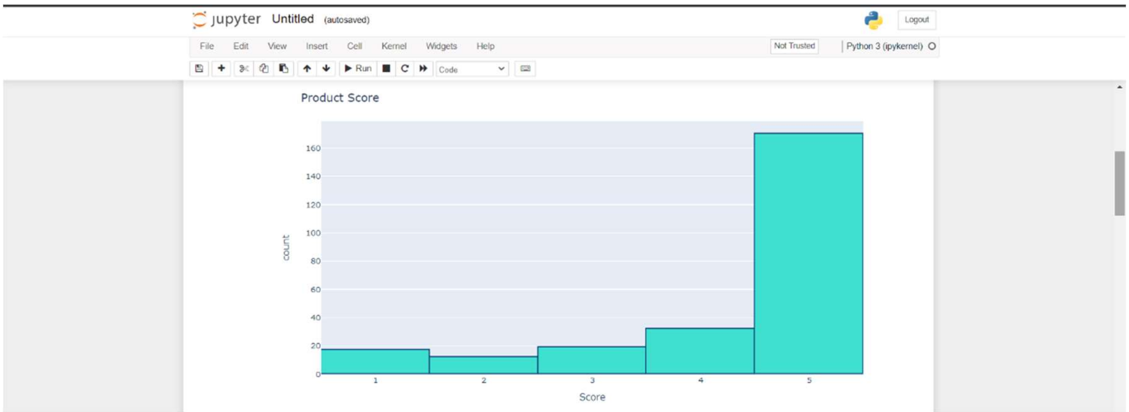
Summary — This is a summary of the entire review.

***Score* —** The product rating provided by the customer.

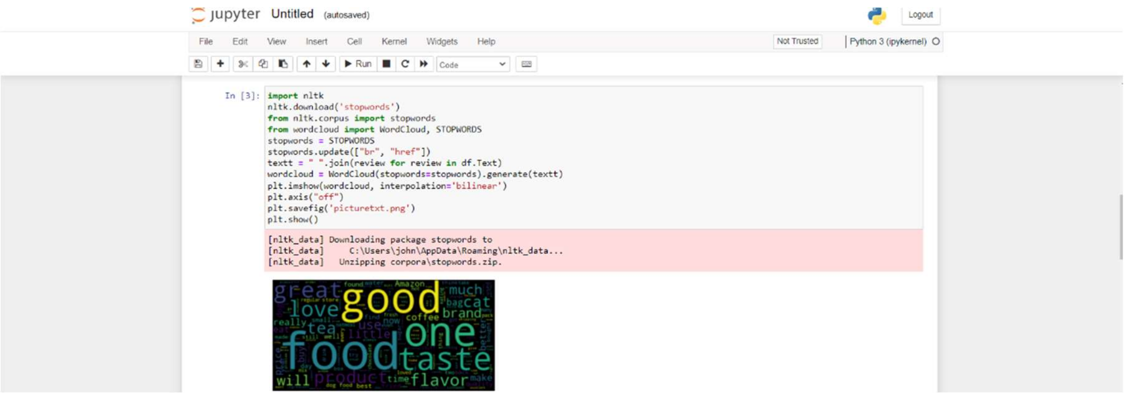
5. Data Analysis


jupyter Untitled (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code

```
In [2]: import matplotlib.pyplot as plt
        import seaborn as sns
        color = sns.color_palette()
        import plotly.offline as py
        py.init_notebook_mode(connected=True)
        import plotly.graph_objs as go
        import plotly.tools as tls
        import plotly.express as px
        fig = px.histogram(df, x="Score")
        fig.update_traces(marker_color="turquoise",marker_line_color='rgb(8,48,107)',
                          marker_line_width=1.5)
        fig.update_layout(title_text='Product Score')
        fig.show()
```

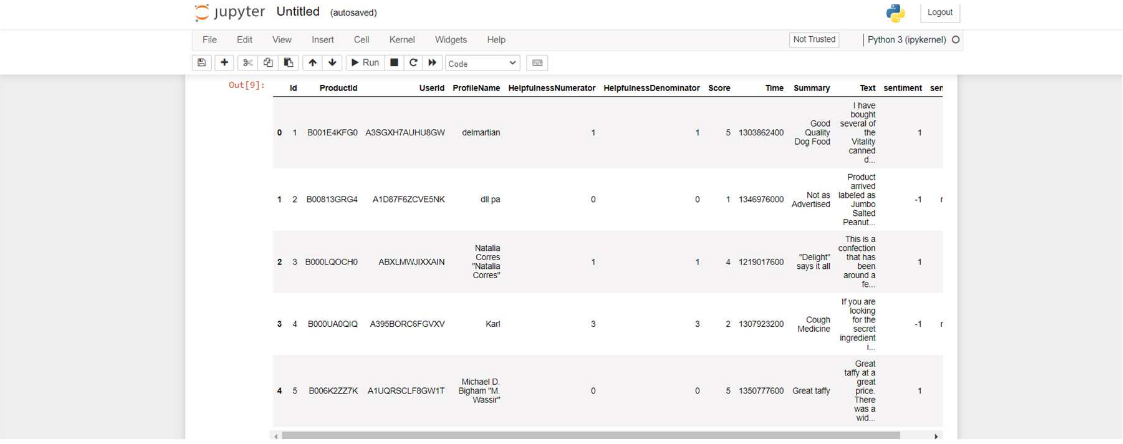


Now, we can create some ***wordclouds*** to see the most frequently used words in the reviews.



```
In [3]: import nltk
        nltk.download('stopwords')
        from nltk.corpus import stopwords
        from wordcloud import WordCloud, STOPWORDS
        stopwords = STOPWORDS
        stopwords.update(["br", "href"])
        textt = " ".join(review for review in df.Text)
        wordcloud = WordCloud(stopwords=stopwords).generate(textt)
        plt.imshow(wordcloud, interpolation='bilinear')
        plt.axis("off")
        plt.savefig('picturetxt.png')
        plt.show()

        [nltk_data] Downloading package stopwords to
        [nltk_data]     C:\Users\john\AppData\Roaming\nltk_data...
        [nltk_data]   Unzipping corpora\stopwords.zip.
```

6. Classifying Tweets

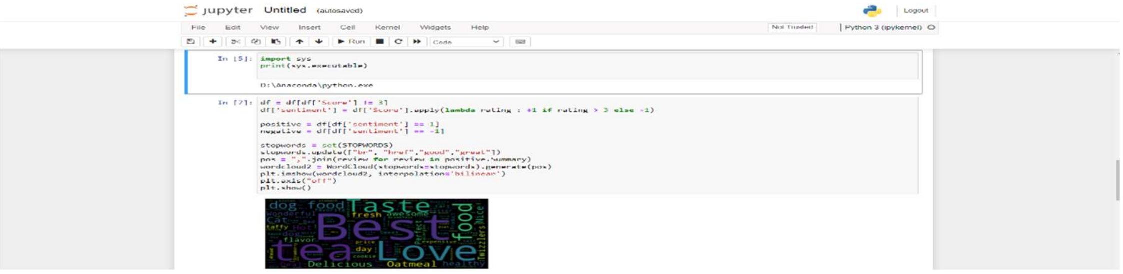


Out[9]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text	sentiment	ser
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d...	1	
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...	-1	r
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"Delight" says it all	This is a confection that has been around a fe...	1	
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine	If you are looking for the secret ingredient i...	-1	r
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Great taffy	Great taffy at a great price. There was a wid...	1	

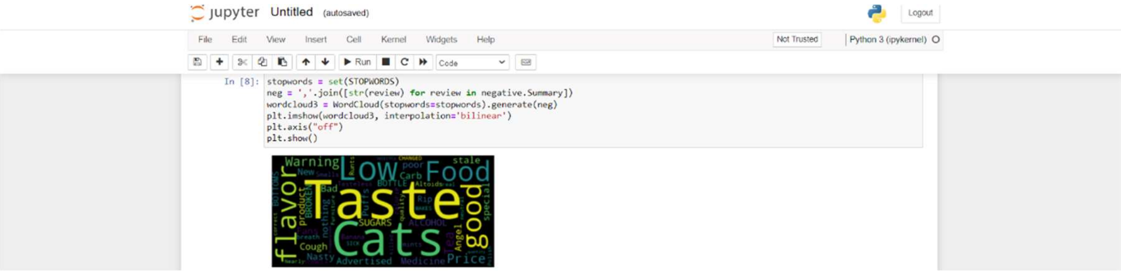
7. More Data Analysis

Wordcloud Positive Sentiment:



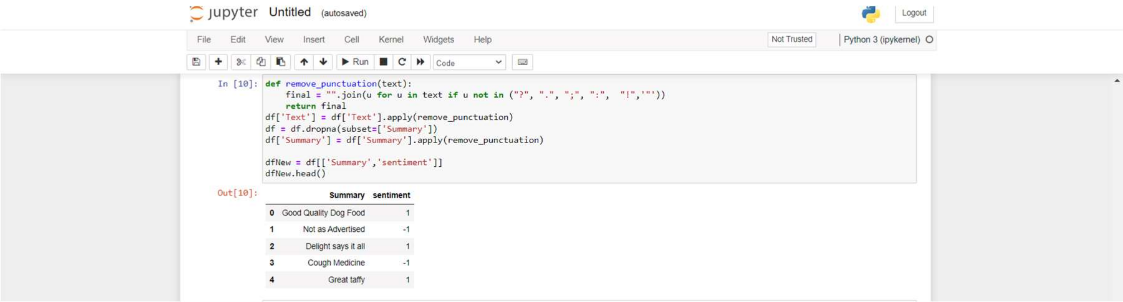
```
In [5]: import sys
        print(sys.executable)

        D:\Anaconda\python.exe

In [7]: df = df[df['Score'] != 3]
        df['sentiment'] = df['Score'].apply(lambda rating : +1 if rating > 3 else -1)

        positive = df[df['sentiment'] == 1]
        negative = df[df['sentiment'] == -1]

        stopwords = set(STOPWORDS)
        stopwords.update(["br", "href","good","great"])
        pos = ",".join(review for review in positive.Summary)
        wordcloud2 = WordCloud(stopwords=stopwords).generate(pos)
        plt.imshow(wordcloud2, interpolation='bilinear')
        plt.axis("off")
        plt.show()
```

Wordcloud Negative Sentiment



```
In [8]: stopwords = set(STOPWORDS)
        neg = ','.join([str(review) for review in negative.Summary])
        wordcloud3 = WordCloud(stopwords=stopwords).generate(neg)
        plt.imshow(wordcloud3, interpolation='bilinear')
        plt.axis("off")
        plt.show()
```

8. Building the Model.

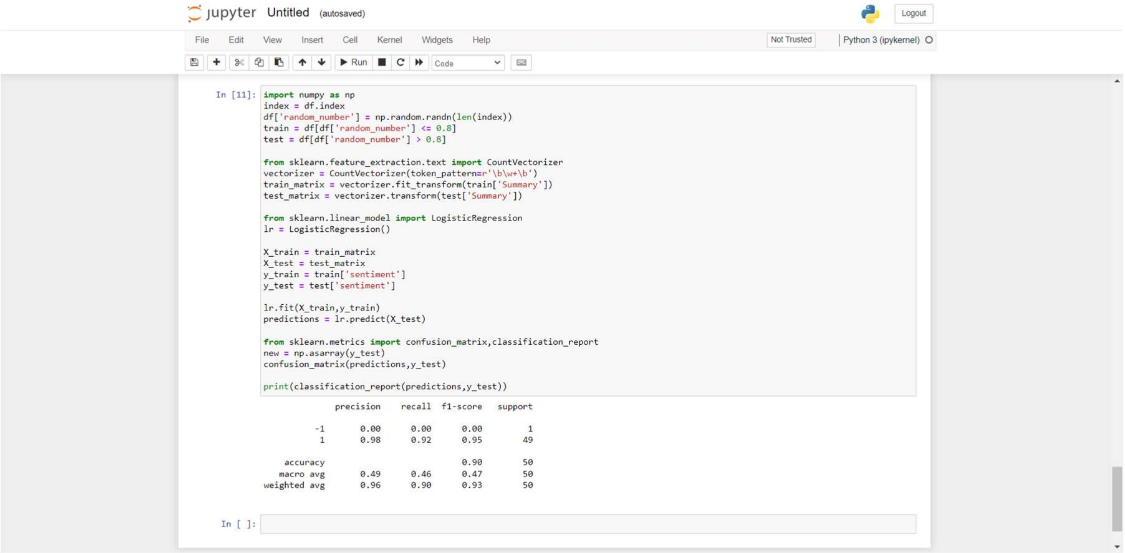

jupyter Untitled (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code

```
In [10]: def remove_punctuation(text):
             final = "".join(u for u in text if u not in ("?", ".", ";", ":",  "!",'"'))
             return final
         df['Text'] = df['Text'].apply(remove_punctuation)
         df = df.dropna(subset=['Summary'])
         df['Summary'] = df['Summary'].apply(remove_punctuation)

         dfNew = df[['Summary','sentiment']]
         dfNew.head()
```

Out[10]:

	Summary	sentiment
0	Good Quality Dog Food	1
1	Not as Advertised	-1
2	Delight says it all	1
3	Cough Medicine	-1
4	Great taffy	1

9. Testing


jupyter Untitled (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code

```
In [11]: import numpy as np
         index = df.index
         df['random_number'] = np.random.randn(len(index))
         train = df[df['random_number'] <= 0.8]
         test = df[df['random_number'] > 0.8]

         from sklearn.feature_extraction.text import CountVectorizer
         vectorizer = CountVectorizer(token_pattern=r'\b\w+\b')
         train_matrix = vectorizer.fit_transform(train['Summary'])
         test_matrix = vectorizer.transform(test['Summary'])

         from sklearn.linear_model import LogisticRegression
         lr = LogisticRegression()

         X_train = train_matrix
         X_test = test_matrix
         y_train = train['sentiment']
         y_test = test['sentiment']

         lr.fit(X_train,y_train)
         predictions = lr.predict(X_test)

         from sklearn.metrics import confusion_matrix,classification_report
         new = np.asarray(y_test)
         confusion_matrix(predictions,y_test)

         print(classification_report(predictions,y_test))
```

```
              precision    recall  f1-score   support

          -1       0.00      0.00      0.00         1
           1       0.98      0.92      0.95        49

    accuracy                           0.90        50
   macro avg       0.49      0.46      0.47        50
weighted avg       0.96      0.90      0.93        50
```

```
In [ ]:
```