Appendix 1.2

Loading Python Libraries

'Pandas' library will help in importing, exporting and manipulating the dataset. 'textblob' and 'vaderSentiment' will allow us to calculate the Textblob and Vader sentiment scores respectively.

In [ ]:
```
import pandas as pd
from textblob import TextBlob
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
```

Loading Data

In [ ]:
```
df1 = pd.read_csv('sentiment140_clean_tweet.csv', encoding = "latin-1")
df2 = pd.read_csv('usairline_clean_tweet.csv', encoding = 'latin-1')
df3 = pd.read_csv('apple_clean_tweet.csv', encoding = 'latin-1')
```

Inspect the pandas dataframe

In [ ]:
```
df1.info()
df2.info()
df3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1490224 entries, 0 to 1490223
Data columns (total 3 columns):
 #   Column       Non-Null Count    Dtype
---  ------       --------------    -----
 0   target       1490224 non-null  int64
 1   tweet        1490224 non-null  object
 2   clean_tweet  1490224 non-null  object
dtypes: int64(1), object(2)
memory usage: 34.1+ MB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14017 entries, 0 to 14016
Data columns (total 3 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   airline_sentiment  14017 non-null  object
 1   text               14017 non-null  object
 2   clean_tweet        14017 non-null  object
dtypes: object(3)
memory usage: 328.6+ KB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3671 entries, 0 to 3670
Data columns (total 3 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   sentiment    3671 non-null   int64
 1   text         3671 non-null   object
 2   clean_tweet  3671 non-null   object
dtypes: int64(1), object(2)
memory usage: 86.2+ KB
```

Sentiment Scoring (TEXTBLOB)

In [ ]:
```
def sentiment_calc(text):
    try:
        return TextBlob(text).polarity
    except:
        return None

df1['textblob'] = df1['clean_tweet'].apply(sentiment_calc)
df2['textblob'] = df2['clean_tweet'].apply(sentiment_calc)
df3['textblob'] = df3['clean_tweet'].apply(sentiment_calc)
```

Sentiment Scoring (VADER)

In [ ]:
```
analyzer = SentimentIntensityAnalyzer()
```

In [ ]:
```
df1['vader'] = df1['clean_tweet'].apply(lambda tweet: analyzer.polarity_scores(tweet))
df2['vader'] = df2['clean_tweet'].apply(lambda tweet: analyzer.polarity_scores(tweet))
df3['vader'] = df3['clean_tweet'].apply(lambda tweet: analyzer.polarity_scores(tweet))
```

In [ ]:
```
df1.head()
```

Out[ ]:

	target	tweet	clean_tweet	textblob	vader
0	0	@switchfoot http://twitpic.com/2y1zl - Awww, t...	- Awww, that is a bummer. You shoulda got Davi...	0.216667	-0.0173
1	0	is upset that he can't update his Facebook by ...	is upset that he can not update his Facebook b...	0.000000	-0.7500
2	0	@Kenichan I dived many times for the ball. Man...	I dived many times for the ball. Managed to sa...	0.500000	0.4939
3	0	my whole body feels itchy and like its on fire	my whole body feels itchy and like its on fire	0.200000	-0.2500
4	0	@nationwideclass no, it's not behaving at all....	no, it is not behaving at all. I am mad. why a...	-0.625000	-0.4939

In [ ]:
```
df2.head()
```

Out[ ]:

	airline_sentiment	text	clean_tweet	textblob	vader
0	positive	@VirginAmerica plus you've added commercials t...	plus you have added commercials to the experie...	0.000000	0.0000
1	neutral	@VirginAmerica I didn't today... Must mean I n...	I did not today... Must mean I need to take an...	-0.390625	0.0000
2	negative	@VirginAmerica it's really aggressive to blast...	it is really aggressive to blast obnoxious "en...	0.006250	-0.2716
3	negative	@VirginAmerica and it's a really big bad thing...	and it is a really big bad thing about it	-0.350000	-0.5829
4	negative	@VirginAmerica seriously would pay $30 a fligh...	seriously would pay $ a flight for seats that ...	-0.208333	-0.5945

In [ ]:
```
df3.head()
```

Out[ ]:

	sentiment	text	clean_tweet	textblob	vader
0	3	#AAPL:The 10 best Steve Jobs emails ever...htt...	:The best Steve Jobs emails ever...	1.00	0.6369
1	3	RT @JPDesloges: Why AAPL Stock Had a Mini-Flas...	RT Why AAPL Stock Had a Mini-Flash Crash Today...	0.00	-0.4019
2	3	My cat only chews @apple cords. Such an #Apple...	My cat only chews cords. Such an .	0.00	0.0000
3	3	I agree with @jimcramer that the #IndividualIn...	I agree with that the should own not trade , i...	0.65	0.6597
4	3	Nobody expects the Spanish Inquisition #AAPL	Nobody expects the Spanish Inquisition	0.00	-0.2960

In [ ]:
```
df1.to_csv("sentiment140_scores.csv", index=False)
df2.to_csv("usairline_scores.csv", index=False)
df3.to_csv("apple_scores.csv", index = False)
```

Testing the existence of Vader and Textblob sentiment scores on the machine learning text classification model's peformance

Without Any Sentiment Scores

In [ ]:
```
import pandas as pd
import numpy as np
df = pd.read_csv('usairline_ml_text.csv')
np.random.seed(100)
```

In [ ]:
```
df.head()
```

Out[ ]:

	airline_sentiment	clean_tweet
0	positive	plus you have added commercials to the experie...
1	neutral	I did not today... Must mean I need to take an...
2	negative	it is really aggressive to blast obnoxious "en...
3	negative	and it is a really big bad thing about it
4	negative	seriously would pay $ a flight for seats that ...

In [ ]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(df['clean_tweet'], df['airline_ser
```

In [ ]:
```
import string
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')

stopwords = set(stopwords.words('english'))
def preprocess(text):
    text = text.lower()
    text = ''.join([word for word in text if word not in string.punctuation])
    text = ' '.join([word for word in text.split() if word not in stopwords])
    return text

X_train = X_train.apply(preprocess)
X_test = X_test.apply(preprocess)
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Salman\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

In [ ]:
```
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer

vectorizer = TfidfVectorizer()
X_train = vectorizer.fit_transform(X_train)
X_test = vectorizer.transform(X_test)
```

In [ ]:
```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)
```

Out[ ]:
```
LogisticRegression(max_iter=1000)
```

In [ ]:
```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

y_pred = model.predict(X_test)
print('Accuracy:', accuracy_score(y_test, y_pred))
print('Precision:', precision_score(y_test, y_pred, average='weighted'))
print('Recall:', recall_score(y_test, y_pred, average='weighted'))
print('F1 Score:', f1_score(y_test, y_pred, average='weighted'))
```

```
Accuracy: 0.7703281027104137
Precision: 0.7610065854065596
Recall: 0.7703281027104137
F1 Score: 0.7509463972878909
```

With Textblob Scores

In [ ]:
```
import pandas as pd

# load the dataset with the new variable
df2 = pd.read_csv('usairline_ml_text_textblob.csv')
df2.head()
```

Out[ ]:

	airline_sentiment	clean_tweet	textblob
0	positive	plus you have added commercials to the experie...	0.000000
1	neutral	I did not today... Must mean I need to take an...	-0.390625
2	negative	it is really aggressive to blast obnoxious "en...	0.006250
3	negative	and it is a really big bad thing about it	-0.350000
4	negative	seriously would pay $ a flight for seats that ...	-0.208333

In [ ]:
```
# split the dataset into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(df2[['clean_tweet', 'textblob']],

# preprocess the text input
import string
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')

stopwords = set(stopwords.words('english'))
def preprocess(text):
    text = text.lower()
    text = ''.join([word for word in text if word not in string.punctuation])
    text = ' '.join([word for word in text.split() if word not in stopwords])
    return text
```

```
X_train['clean_tweet'] = X_train['clean_tweet'].apply(preprocess)
X_test['clean_tweet'] = X_test['clean_tweet'].apply(preprocess)

# vectorize the text input
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer

vectorizer = TfidfVectorizer()
X_train_text = vectorizer.fit_transform(X_train['clean_tweet'])
X_test_text = vectorizer.transform(X_test['clean_tweet'])

# combine the vectorized text input with the numeric input
import scipy.sparse as sp

X_train_num = sp.csr_matrix(X_train['textblob'].values.reshape(-1, 1))
X_test_num = sp.csr_matrix(X_test['textblob'].values.reshape(-1, 1))

X_train = sp.hstack([X_train_text, X_train_num])
X_test = sp.hstack([X_test_text, X_test_num])

# train the logistic regression model
from sklearn.linear_model import LogisticRegression

model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# evaluate the model
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

y_pred = model.predict(X_test)
print('Accuracy:', accuracy_score(y_test, y_pred))
print('Precision:', precision_score(y_test, y_pred, average='weighted'))
print('Recall:', recall_score(y_test, y_pred, average='weighted'))
print('F1 Score:', f1_score(y_test, y_pred, average='weighted'))
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Salman\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Accuracy: 0.7781740370898717
Precision: 0.7680282985788505
Recall: 0.7781740370898717
F1 Score: 0.7614631813035634
```

With Vader Scores

In [ ]:

```
import pandas as pd

# load the dataset with the new variable
df3 = pd.read_csv('usairline_ml_text_vader.csv')
df3.head()
```

Out[ ]:

	airline_sentiment	clean_tweet	vader
0	positive	plus you have added commercials to the experie...	0.0000
1	neutral	I did not today... Must mean I need to take an...	0.0000
2	negative	it is really aggressive to blast obnoxious "en...	-0.2716
3	negative	and it is a really big bad thing about it	-0.5829
4	negative	seriously would pay $ a flight for seats that ...	-0.5945

In [ ]:

```
# split the dataset into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(df3[['clean_tweet', 'vader']], df3

# preprocess the text input
import string
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')

stopwords = set(stopwords.words('english'))
def preprocess(text):
    text = text.lower()
    text = ''.join([word for word in text if word not in string.punctuation])
    text = ' '.join([word for word in text.split() if word not in stopwords])
    return text

X_train['clean_tweet'] = X_train['clean_tweet'].apply(preprocess)
X_test['clean_tweet'] = X_test['clean_tweet'].apply(preprocess)

# vectorize the text input
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer

vectorizer = TfidfVectorizer()
X_train_text = vectorizer.fit_transform(X_train['clean_tweet'])
X_test_text = vectorizer.transform(X_test['clean_tweet'])

# combine the vectorized text input with the numeric input
import scipy.sparse as sp

X_train_num = sp.csr_matrix(X_train['vader'].values.reshape(-1, 1))
X_test_num = sp.csr_matrix(X_test['vader'].values.reshape(-1, 1))

X_train = sp.hstack([X_train_text, X_train_num])
X_test = sp.hstack([X_test_text, X_test_num])

# train the logistic regression model
from sklearn.linear_model import LogisticRegression

model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# evaluate the model
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

y_pred = model.predict(X_test)
print('Accuracy:', accuracy_score(y_test, y_pred))
print('Precision:', precision_score(y_test, y_pred, average='weighted'))
```

```
print('Recall:', recall_score(y_test, y_pred, average='weighted'))
print('F1 Score:', f1_score(y_test, y_pred, average='weighted'))
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Salman\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Accuracy: 0.7885164051355207
Precision: 0.7777831580798289
Recall: 0.7885164051355207
F1 Score: 0.7741147898432414
```

With Vader + Textblob

In [ ]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from scipy import sparse as sp
import string
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')

# load the data
df4 = pd.read_csv('usairline_ml_text_vader_textblob.csv')
df4.head()
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Salman\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[ ]:

	airline_sentiment	clean_tweet	vader	textblob
0	positive	plus you have added commercials to the experie...	0.0000	0.000000
1	neutral	I did not today... Must mean I need to take an...	0.0000	-0.390625
2	negative	it is really aggressive to blast obnoxious "en...	-0.2716	0.006250
3	negative	and it is a really big bad thing about it	-0.5829	-0.350000
4	negative	seriously would pay $ a flight for seats that ...	-0.5945	-0.208333

In [ ]:
```
# split the data into train and test sets
X_train_text, X_test_text, y_train, y_test = train_test_split(df4['clean_tweet'], df4[
X_train_num = sp.csr_matrix(df4.loc[X_train_text.index, ['textblob', 'vader']].values)
X_test_num = sp.csr_matrix(df4.loc[X_test_text.index, ['textblob', 'vader']].values)

# preprocess the text data
stopwords = set(stopwords.words('english'))
def preprocess(text):
    text = text.lower()
    text = ''.join([word for word in text if word not in string.punctuation])
    text = ' '.join([word for word in text.split() if word not in stopwords])
    return text

X_train_text = X_train_text.apply(preprocess)
```

```
X_test_text = X_test_text.apply(preprocess)

# create tf-idf vectorizer
vectorizer = TfidfVectorizer()
X_train_text = vectorizer.fit_transform(X_train_text)
X_test_text = vectorizer.transform(X_test_text)

# combine the text and numeric features
X_train = sp.hstack([X_train_text, X_train_num])
X_test = sp.hstack([X_test_text, X_test_num])

# train the logistic regression model
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# make predictions on the test data and evaluate the performance
y_pred = model.predict(X_test)
print('Accuracy:', accuracy_score(y_test, y_pred))
print('Precision:', precision_score(y_test, y_pred, average='weighted'))
print('Recall:', recall_score(y_test, y_pred, average='weighted'))
print('F1 Score:', f1_score(y_test, y_pred, average='weighted'))
```

```
Accuracy: 0.7892296718972895
Precision: 0.777975159189456
Recall: 0.7892296718972895
F1 Score: 0.7750863723452115
```

Vader Sentiment Scoring of Climate Tweets

In [ ]:
```
df4 = pd.read_csv('climate_clean_tweet.csv')
df4.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9048 entries, 0 to 9047
Data columns (total 2 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   clean_tweet  9048 non-null   object
 1   Timestamp    9048 non-null   object
dtypes: object(2)
memory usage: 141.5+ KB
```

In [ ]:
```
df4['vader'] = df4['clean_tweet'].apply(lambda tweet: analyzer.polarity_scores(tweet))

df4.head()
```

Out[ ]:

	clean_tweet	Timestamp	vader
0	The only solution I have ever heard the Left p...	2022-01-17T23:32:38Z	0.5279
1	Climate change does not cause volcanic eruptions.	2022-01-17T22:54:02Z	0.0000
2	Vaccinated tennis ball boy collapses in the te...	2022-01-17T23:51:41Z	-0.2960
3	North America has experienced an average winte...	2022-01-17T21:42:04Z	0.2924
4	they are gonna do the same with Climate Change...	2022-01-17T21:10:40Z	-0.4939

In [ ]:
```
df4.to_csv('climate_vader_scores.csv', index = False)
```