Sentimental Analysis For Marketing

phase-3

We started building our project by loading the dataset, performing text preprocessing, and conducting analysis in Colab Notebook.

Sentimental Analysis with Python:

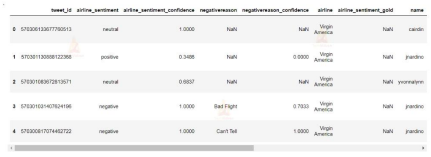
To build a machine learning model to accurately classify whether customers are saying positive or negative.

Steps to build Sentiment Analysis Text Classifier in Python

1. Data Preprocessing:

As we are dealing with the text data, we need to preprocess it using word embeddings.

```
import pandas as pd
df =
pd.read_csv("./DesktopDataFlair/Sentime
nt-Analysis/Tweets.csv")
```



	tweet_id	airline_sentiment	airline_sentiment_confidence	negativereason	negativereason_confidence	airline	airline_sentiment_gold	name
0	570306133677760513	neutral	1.0000	NaN	NaN	Virgin America	NaN	cairdin
1	570301130888122368	positive	0.3486	NaN	0.0000	Virgin America	NaN	jnardino
2	570301083672813571	neutral	0.6837	NaN	NaN	Virgin America	NaN	yvonnalynn
3	570301031407624196	negative	1.0000	Bad Flight	0.7033	Virgin America	NaN	jnardino
4	570300817074462722	negative	1.0000	Can't Tell	1.0000	Virgin America	NaN	jnardino

We only need the text and sentiment column.

```
review_df =
df[['text','airline_sentiment']]

print(review_df.shape)
review_df.head(5)
```

(14640, 2)

Out[4]:

	text	airline_sentiment
0	@VirginAmerica What @dhepburn said.	neutral
1	@VirginAmerica plus you've added commercials t...	positive
2	@VirginAmerica I didn't today... Must mean I n...	neutral
3	@VirginAmerica it's really aggressive to blast...	negative
4	@VirginAmerica and it's a really big bad thing...	negative

```
df.columns
```

```
Out[23]: Index(['tweet_id', 'airline_sentiment', 'airline_sentiment_confidence',
       'negativereason', 'negativereason_confidence', 'airline',
       'airline_sentiment_gold', 'name', 'negativereason_gold',
       'retweet_count', 'text', 'tweet_coord', 'tweet_created',
       'tweet_location', 'user_timezone'],
      dtype='object')
```

```
review_df =
review_df[review_df['airline_sentiment'
] != 'neutral']

print(review_df.shape)
review_df.head(5)
```

```
(11541, 2)
Out[22]:
```

	text	airline_sentiment
1	@VirginAmerica plus you've added commercials t...	positive
3	@VirginAmerica it's really aggressive to blast...	negative
4	@VirginAmerica and it's a really big bad thing...	negative
5	@VirginAmerica seriously would pay $30 a flight...	negative
6	@VirginAmerica yes, nearly every time I fly VX...	positive

The labels for this dataset are categorical. Machines understand only numeric data. So, convert the categorical values to numeric using the factorize() method. This returns an array of numeric values and an Index of categories.

```
sentiment_label =
review_df.airline_sentiment.factorize()
sentiment_label
```

```
In [6]: sentiment_label = tweet_df.airline_sentiment.factorize()
        sentiment_label
Out[6]: (array([0, 1, 1, ..., 0, 1, 1], dtype=int64),
         Index(['positive', 'negative'], dtype='object'))
```

2. Build the Text Classifier:

For sentiment analysis project, we use LSTM layers in the machine learning model. The architecture of our model consists of an embedding layer, an LSTM layer, and a Dense layer at the end. To avoid overfitting, we introduced the Dropout mechanism in-between the LSTM layers.

```
from tensorflow.keras.models import
Sequential
from tensorflow.keras.layers import
LSTM,Dense, Dropout, SpatialDropout1D
from tensorflow.keras.layers import
Embedding


embedding_vector_length = 32
model = Sequential()
model.add(Embedding(vocab_size,
embedding_vector_length,
input_length=200))
model.add(SpatialDropout1D(0.25))
model.add(LSTM(50, dropout=0.5,
recurrent_dropout=0.5))
model.add(Dropout(0.2))
model.add(Dense(1,
activation='sigmoid'))
model.compile(loss='binary_crossentropy
',optimizer='adam', metrics=
['accuracy'])

print(model.summary())
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 200, 32)           423488
_________________________________________________________________
spatial_dropout1d (SpatialDr (None, 200, 32)           0
_________________________________________________________________
lstm (LSTM)                  (None, 50)                16600
_________________________________________________________________
dropout (Dropout)            (None, 50)                0
_________________________________________________________________
dense (Dense)                (None, 1)                 51
=================================================================
Total params: 440,139
Trainable params: 440,139
Non-trainable params: 0
_________________________________________________________________
None
```

3. Train the sentiment analysis model:

Train the sentiment analysis model for 5 epochs on the whole dataset with a batch size of 32 and a validation split of 20%.

```
history =
model.fit(padded_sequence,sentiment_lab
el[0],validation_split=0.2, epochs=5,
batch_size=32)
```

The output while training looks like below:

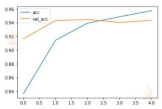
```
Epoch 1/5
249/249 [==============================] - 471s 2s/step - loss: 0.4926 - accuracy: 0.7980 - val_loss: 0.2133 - val_accuracy: 0.
9164
Epoch 2/5
249/249 [==============================] - 457s 2s/step - loss: 0.2282 - accuracy: 0.9118 - val_loss: 0.1624 - val_accuracy: 0.
9428
Epoch 3/5
249/249 [==============================] - 423s 1s/step - loss: 0.1755 - accuracy: 0.9340 - val_loss: 0.1667 - val_accuracy: 0.
9466
Epoch 4/5
249/249 [==============================] - 420s 1s/step - loss: 0.1292 - accuracy: 0.9519 - val_loss: 0.1678 - val_accuracy: 0.
9402
Epoch 5/5
249/249 [==============================] - 420s 1s/step - loss: 0.1117 - accuracy: 0.9600 - val_loss: 0.1818 - val_accuracy: 0.
9453
```

```
import matplotlib.pyplot as plt

plt.plot(history.history['accuracy'],
label='acc')
plt.plot(history.history['val_accuracy'
], label='val_acc')
plt.legend()
plt.show()

plt.savefig("Accuracy plot.jpg")
```

Output:



Let’s execute sentiment analysis model
Define a function that takes a text as input and outputs its prediction label.

```
def predict_sentiment(text):
    tw =
tokenizer.texts_to_sequences([text])
    tw = pad_sequences(tw,maxlen=200)
    prediction =
int(model.predict(tw).round().item())
    print("Predicted label: ",
sentiment_label[1][prediction])


test_sentence1 = "I enjoyed my journey
on this flight."
predict_sentiment(test_sentence1)

test_sentence2 = "This is the worst
flight experience of my life!"
predict_sentiment(test_sentence2)
```

Python Sentiment Analysis Output

Team Members:

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