Assignment 8

NLP Sentiment Analysis (IMDB Dataset)

Name : Athavra Ramgirkar
Registration Number: 19BCI0114
Submission Date : 15 July, 2021
Program : VIT-AI Industry Certification
Email : athavra.ramgirkar2019@vitstudent.ac.in

Other Assignments can be found in the link:
https://drive.google.com/drive/folders/1QGOLHyZykoj_CroTJu6-YkZWI32JZ-QH?usp=sharing

Table of Content

In [ ]:

In [ ]:

1. Importing Libraries

Back to Top

In [1]:
```
import pandas as pd
import numpy as np

# For NLP
import re
import nltk
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from sklearn.feature_extraction.text import CountVectorizer

# For Train Test Split
from sklearn.model_selection import train_test_split

# For Neural Network
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

# For handling Missing Values
import missingno as ms
import matplotlib.pyplot as plt

# Model Evaluation
from sklearn.metrics import accuracy_score
```

1.1 Initializing Objects

In [2]:
```
ps = PorterStemmer()
cv = CountVectorizer(max_features=4000)
```

In [ ]:

2. Reading Data

Back to Top

In [3]:
```
df = pd.read_csv("IMDB Dataset.csv")
```

In [ ]:

3. Understanding Data

Back to Top

In [4]:
```
df.head()
```

Out[4]:

	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. The...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative
4	Petter Mattei's "Love in the Time of Money" is...	positive

3.1 Checking For Null Values

In [5]:
```
df.isnull().sum()
```

Out[5]:
```
review       0
sentiment    0
dtype: int64
```

3.2 Checking For Duplicate Values

In [6]:
```
df.shape
```

Out[6]:
```
(50000, 2)
```

In [7]:
```
len(df['review'].unique())
```

Out[7]:
```
49582
```

3.2.1 Dropping Duplicate Rows

In [8]:
```
df.drop_duplicates(inplace=True)
```

In [9]:
```
df.shape
```

Out[9]:
```
(49582, 2)
```

In [10]:
```
len(df['review'].unique())
```

Out[10]:
```
49582
```

In [11]:
```
df.head()
```

Out[11]:

	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. The...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative
4	Petter Mattei's "Love in the Time of Money" is...	positive

3.3 Checking if Dataset is Balanced

In [12]:
```
df['sentiment'].value_counts()
```

Out[12]:
```
positive    24884
negative    24698
Name: sentiment, dtype: int64
```

4. Encoding Target Coulmn

Back to Top

In [13]:
```
df['sentiment'] = np.where(df['sentiment']=="positive",1,0)
```

In [14]:
```
df.head()
```

Out[14]:

	review	sentiment
0	One of the other reviewers has mentioned that ...	1
1	A wonderful little production. The...	1
2	I thought this was a wonderful way to spend ti...	1
3	Basically there's a family where a little boy ...	0
4	Petter Mattei's "Love in the Time of Money" is...	1

5. Re-ordering the Index

Back to Top

In [15]:
```
df.reset_index(drop=True,inplace=True)
```

In [ ]:

6. Separating the Features and the Target Variable

Back to Top

In [16]:
```
y = df['sentiment'].values
```

6.1 Storing the Target Variable

In [17]:
```
y
```

Out[17]:
```
array([1, 1, 1, ..., 0, 0, 0])
```

6.2 NLP on the "review" Column

In [18]:
```
data = []
for i in range(0,49582):
    rev = df['review'][i]

    # Removing Special Characters
    rev = re.sub('[^a-zA-Z]',' ',rev)

    # Converting to Lower Case
    rev = rev.lower()

    # Splitting Sentences to List of Words
    rev = rev.split()

    # Stemming and Stop Word Removal
    rev = [ps.stem(word) for word in rev if not word in set(stopwords.words('english'))]

    # Re-Forming Sentence
    rev = ' '.join(rev)

    # Appending to Corpus
    data.append(rev)
```

6.3 Vectorizing the Features

In [19]:
```
X = cv.fit_transform(data).toarray()
```

In [ ]:

7. Train Test Split

Back to Top

In [20]:
```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.3,random_state=114,st
```

In [ ]:

8. Model Building

Back to Top

8.1 Initialize the Model

In [21]:
```
model = Sequential()
```

8.2 Add Input Layer

In [22]:
```
model.add(Dense(units = 4000,
                kernel_initializer="random_uniform",
                activation="relu"))
```

8.3 Add Hidden Layers

In [23]:
```
model.add(Dense(units = 2000,
                kernel_initializer="random_uniform",
                activation="relu"))

model.add(Dense(units = 1000,
                kernel_initializer="normal",
                activation="relu"))
```

8.4 Add Output Layer

In [24]:
```
model.add(Dense(units = 1,
                kernel_initializer="random_uniform",
                activation="sigmoid"))
```

9. Compile the Model

Back to Top

In [25]:
```
model.compile(optimizer="adam",
              loss="binary_crossentropy",
              metrics=["accuracy"])
```

In [ ]:

10. Train the Model

Back to Top

In [26]:
```
model.fit(X_train,y_train,epochs=2)
```

```
Epoch 1/2
1085/1085 [==============================] - 199s 184ms/step - loss: 0.3416 - accurac
y: 0.8583
Epoch 2/2
1085/1085 [==============================] - 189s 174ms/step - loss: 0.1636 - accurac
y: 0.9359
```

Out[26]:
```
<tensorflow.python.keras.callbacks.History at 0x16e8b5dba90>
```

In [ ]:

11. Make prediction on Test Data

Back to Top

In [27]:
```
pred = model.predict(X_test)
pred = pred>0.44
```

In [28]:
```
accuracy_score(y_test, pred)
```

Out[28]:
```
0.8646722689075563
```

In [ ]:

12. Single Predictions

Back to Top

In [29]:
```
model.predict(cv.transform(["amazing good awesome movie"]))
```

Out[29]:
```
array([[0.66847323]], dtype=float32)
```

In [30]:
```
model.predict(cv.transform(["bad no good boring long movie"]))
```

Out[30]:
```
array([[0.42939442]], dtype=float32)
```

In [31]:
```
model.predict(cv.transform(["movie waste of time"]))
```

Out[31]:
```
array([[0.65661886]], dtype=float32)
```
