Assignment 5: Naive Bayes Learning

1.About the model

Naïve Bayes algorithm is a supervised learning algorithm, it is based on Bayes theorem. We use this algorithm mostly in classification problems such as text classification where we have high dimensional dataset. Here the prediction is based on probability of an object.

P(A|B) = P(B|A).P(A)/P(B)

Where,

P(A): Prior probability i.e, probability of hypothesis before observing the evidence

P(B): Marginal probability i.e., probability of evidence

P(A|B): Evidence probability when hypothesis is true.

P(B|A): Hypothesis probability evidence is true.

Spam filtration, Sentimental analysis, and classifying articles are some popular examples of Naive Bayes Algorithm.

2.Dataset

This is a collection of customer reviews from six of the review topics used in the paper by Blitzer et al., (2007) mentioned above. Here we are formatting given data to make sure there is one review per line, and texts are split into lowercased separate words ("tokens").

url= http://www.cse.chalmers.se/~richajo/dit862/data/all_sentiment_shuffled.txt

Here necessary packages are imported along with requests package which allows you to send HTTP/1.1 requests easily.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
import requests
```

Here we are obtaining data from URL and storing it in txt_data and then we are printing the data.

```
#obtaining data using requests package from URL
rqst_url = requests.get ('http://www.cse.chalmers.se/~richajo/dit862/data/all_sentiment_shuff
txt_data = rqst_url.text
txt_data
```

```
'music neg 241.txt i bought this album because i loved the ti
tle song . it \'s such a great song , how bad can the rest of
the album be , right ? well , the rest of the songs are just
```

3.Data pre-processing

Here we are organizing this into columns

• 0: topic category label (books, camera, dvd, health, music, or software)

• 1: sentiment polarity label (pos or neg)

• 2: document identifier

• 3 and on: the words in the document

```
#defining the columns
label = []
category= []
document_id_no = []
review_of_words = []
```

Here we are splitting the data into the columns defined above

```
for line in txt_data.splitlines():
  label.append(line.split()[1])
  category.append(line.split()[0])
  document_id_no.append(line.split()[2])
  review_of_words.append(' '.join(token for token in line.split()[3:]))
```

Here with the columns defined we are creating a data frame

```
#Dataframe with columns ('label','category','document_id_no','review_of_words')
df_txt = pd.DataFrame(zip(label,category,document_id_no,review_of_words ), columns = ['label'

# Here we are obtaining first 5 rows of the dataframe
df_txt.head()
```

	label	category	document_id_no	review_of_words
0	neg	music	241.txt	i bought this album because i loved the title ...
1	neg	music	544.txt	i was misled and thought i was buying the enti...
				i have introduced many of my

Here we are checking if there are any null values in the data

```
df_txt.isnull().sum()
```

```
label              0
category           0
document_id_no     0
review_of_words    0
dtype: int64
```

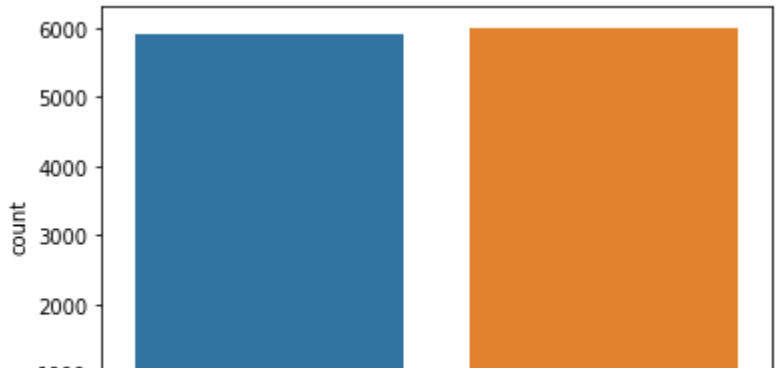
4.Data visualization

```
#Here we are counting values in label column and printing it
count_label = df_txt['label'].value_counts()
count_label
```

```
pos    6000
neg    5914
Name: label, dtype: int64
```

Here for the negative and positive values in the column we are creating seaborn plot

```
sns.countplot(data = df_txt, x = 'label')
plt.show();
```



Here we are replacing negative value with 0 and positive value with 1 and then we are converting str into int

```
df_txt['label'] = np.where(df_txt['label']=='pos', 1, 0)
df_txt
```

	label	category	document_id_no	review_of_words
0	0	music	241.txt	i bought this album because i loved the title ...
1	0	music	544.txt	i was misled and thought i was buying the enti...
2	0	books	729.txt	i have introduced many of my ell , high school...
3	1	books	278.txt	anything you purchase in the left behind serie...
4	1	dvd	840.txt	i loved these movies , and i cant wiat for the...

5.Splitting the data

Here we are cleaning the data

```
!pip install stop_words
```

```
Collecting stop_words
```

```
  Downloading stop-words-2018.7.23.tar.gz (31 kB)
Building wheels for collected packages: stop-words
  Building wheel for stop-words (setup.py) ... done
  Created wheel for stop-words: filename=stop_words-2018.7.23-py3-none-any.whl size=3291
  Stored in directory: /root/.cache/pip/wheels/fb/86/b2/277b10b1ce9f73ce15059bf6975d4547
Successfully built stop-words
Installing collected packages: stop-words
Successfully installed stop-words-2018.7.23
```

```
from stop_words import get_stop_words
#here we are removing 'en' words like i,were...etc
stop_words = get_stop_words('en')
# here we are Removing stopwords and numerics
df_txt['review_of_words'] = df_txt['review_of_words'].apply( lambda x: ' '.join([x for x in s
# here we are removing punctuations and non-letter tokens
df_txt['review_of_words']=df_txt['review_of_words'].str.replace('[^\w\s]','')
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:7: FutureWarning: The defau
  import sys
```

```
df_txt['review_of_words'].head()
```

```
0    bought album loved title song  s great song  b...
1    misled thought buying entire cd contains one song
2    introduced many ell  high school students lois...
3    anything purchase left behind series excellent...
4    loved movies  cant wiat third one  funny  suit...
Name: review_of_words, dtype: object
```

Here we are importing the train_test_split module from sklearn package

```
from sklearn.model_selection import train_test_split
```

Here we are assigning words to x and labels to y

```
X = df_txt['review_of_words']
y = df_txt['label']
```

splitting the data with test size as 20% and random state as 42

```
# Here we are creating test and train data using train_test_split model with train size 80% &
X_train, X_test, y_train, y_test = train_test_split (X, y, train_size = 0.8, random_state = 4
# Here we are obtaining train and test set shape
```

```
print ('The Shapes of X_train, y_train: ', X_train.shape, y_train.shape)
print ('The Shapes of X_test, y_test: ', X_test.shape, y_test.shape)
```

```
The Shapes of X_train, y_train:  (9531,) (9531,)
The Shapes of X_test, y_test:  (2383,) (2383,)
```

6.Applying the model

Here we creating token count matrix from text document

```
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer(lowercase = False)
```

Here we are fitting the vectorizer model for training data

```
X_train_tfrm = vectorizer.fit_transform(X_train)
#Here we are creating dense matrix for train data
X_train_tfrm_dnse_mtrx = X_train_tfrm.toarray()
print ('The shape of transformed X_train: ', X_train_tfrm_dnse_mtrx.shape)
print ('The shape of transformed y_train: ', y_train.shape)
```

```
The shape of transformed X_train:  (9531, 47692)
The shape of transformed y_train:  (9531,)
```

Here we are fitting the vectorizer model for test data

```
X_test_tfrm = vectorizer.transform(X_test)
#Here we are creating dense matrix for test data
X_test_tfrm_dnse_mtrx = X_test_tfrm.toarray()
print ('The shape of transformed X_test: ', X_test_tfrm_dnse_mtrx.shape)
print ('The shape of transformed y_test: ', y_test.shape)
```

```
The shape of transformed X_test:  (2383, 47692)
The shape of transformed y_test:  (2383,)
```

```
#Here we are printing the length of the vectorizer feature names
print('The total no. of features: ', len(vectorizer.get_feature_names()))
```

```
The total no. of features:  47692
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: F
  warnings.warn(msg, category=FutureWarning)
```

Here we are creating a Naive Bayes model for text classification

```
# Here we are importing the multinomial naive bayes module from sklearn package
from sklearn.naive_bayes import MultinomialNB
Multi_navie_bayes = MultinomialNB()
# Here we are fitting the model into training set
Multi_navie_bayes.fit(X_train_tfrm_dnse_mtrx, y_train)
```

```
MultinomialNB()
```

```
# Here we are training and validating the model using k-fold cross validation
from sklearn.model_selection import cross_validate
cross_validte = cross_validate (Multi_navie_bayes, X_train_tfrm_dnse_mtrx, y_train, cv = 5)
```

7.Prediction results and test scores

Here we are printing accuracy score for cv=5 and mean

```
print("the accuracy score of 5-fold cross validation:", cross_validte['test_score'])
print("the cross validation accuracy mean score: ", cross_validte['test_score'].mean())
```

```
the accuracy score of 8-fold cross validation: [0.81227058 0.80325289 0.77859391 0.7953
the cross validation accuracy mean score:  0.79708160854333
```

```
y_pred = Multi_navie_bayes.predict(X_test_tfrm_dnse_mtrx)
# Here we are evaluating the model on the test set
print('the Accuracy of model in the test set : {:.3f}'.format(Multi_navie_bayes.score(X_test_
```

```
the Accuracy of model in the test set : 0.822
```

8.Metrics

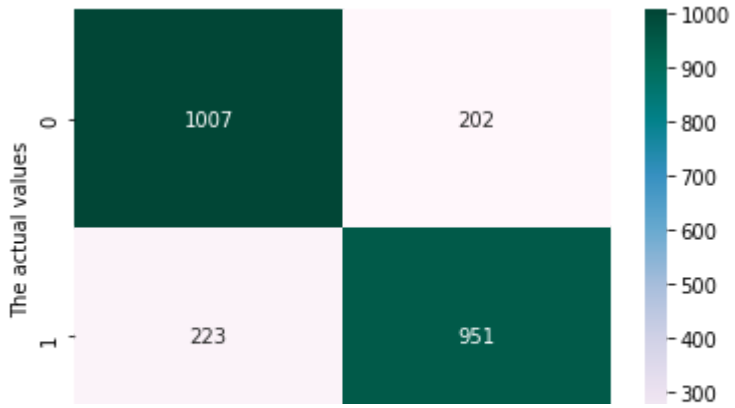
Here we are creating confusion matrix for y_test & y_train

```
from sklearn.metrics import confusion_matrix
cnfn_mtrx = confusion_matrix(y_test, y_pred)
```

Here we are creating a image for confusion matrix heat map by using seaborn package

```
sns.heatmap(cnfn_mtrx, cmap="PuBuGn", annot = True, fmt = '.0f')
plt.xlabel ('The predicted values')
```

```
plt.ylabel ('The actual values')
plt.show();
```



Here by observing we can say that model is correct about 1007 negative reviews, 951 positive reviews but the model is incorrect about 194 positive reviews as negative and 229 negative reviews as positive.

9. Conclusion

Here we are using Receiver Operating Characteristic curve(ROC) it is a graph showing the performance of a classification model at all classification thresholds. The value of AUC is expected to be higher i.e., greater than 0.5 for the model to perform better.

TPR and FPR are the probability that an actual positive will test positive and mistaken prediction of positive class

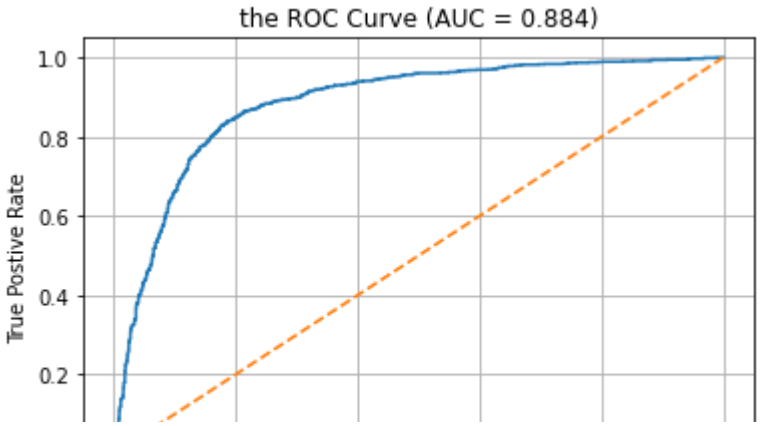
i.e., TPR = TP/P = TP/(TP+FN) and FPR = FP/N = FP/ (FP+TN)

```
from sklearn.metrics import roc_curve, roc_auc_score
```

```
y_pred_prb_1  = Multi_navie_bayes.predict_proba(X_test_tfrm_dnse_mtrx)[:,1]
flse_ptve_rte, true_ptve_rte, threshold = roc_curve (y_test, y_pred_prb_1, pos_label = 1)
roc_auc_score = roc_auc_score (y_test, y_pred_prb_1)
print('The ROC AUC: ', roc_auc_score)
```

```
The ROC AUC:  0.8840799342805167
```

```
# Here we are visualizing the ROC Curve
plt.plot (flse_ptve_rte, true_ptve_rte)
plt.plot([0,1], '--')
plt.xlabel ('False Postive Rate')
plt.ylabel('True Postive Rate')
plt.title ("the ROC Curve (AUC = {:.3f})".format(roc_auc_score))
plt.grid()
plt.show();
```



Here we are plotting between true positive rate and false positive rate at different thresholds.

The ROC curve area is 0.884.

✓ 0s completed at 9:04 PM