

LP 2.

Assignment 4

Date of Completion:-
24.9.2020

Date of Submission:-
29.9.2020

Title:- Text analysis.

Problem Statement:- Consider a suitable text dataset. Remove stop words, apply stemming and feature selection techniques to represent documents as vectors. Classify documents and evaluate precision recall.

Learning Objectives:- To understand the process of stemming, calculating precision and recall.

Learning Outcomes:- Students will be able to understand stop words, stemming, feature selection techniques and calculate precision and recall.

Software / Hardware Requirements:- ~~NLP~~ ~~NLP~~ NLP data / packages, python, Anaconda IDE.

Theory:-

Stop Words

- 1) The most commonly used words in a language.
- 2) These words are filtered out before or after the

- natural language data (txt) is processed.
- 3) They do not add much meaning to a sentence.
eg. as, the, at, a etc.

Stemming:-

- 1) A process of producing morphological variant of the root/base word.
- 2) A word is reduced to its root form.
eg loved \rightarrow love
played \rightarrow play
eating \rightarrow eat.

Precision and Recall:-

- 1) precision = $\frac{\text{Number of correct triples}}{\text{Number of triples retrieved}} = \frac{Tp}{Tp + Fp}$
 $Tp \Rightarrow$ true positive
 $Fp \Rightarrow$ false positive
- 2) recall = $\frac{\text{Number of correct triples}}{\text{Number of triples in gold set}} = \frac{Tp}{Tp + Fn}$
 $Fn \Rightarrow$ false negative
- 3) A measure of success of prediction when classes are very imbalanced.
- 4) Precision is measure of result relevancy
- 5) Recall is measure of how many truly relevant results are returned.

Algorithm:-

1. Import python different packages numpy, pandas, nltk (natural language toolkit), re (regEx)
2. read the dataset. and perform stemming on the words
3. remove the stopwords from the dataset and form a

- corpus (dataset with no stop word and stemmed words)
4. Prepare or vectorize the words. dataset. (feature extraction)
 5. Split the data in training and test set.
 6. Fit classifier model on the dataset (Naive Bayes)
 7. Predict value on the test data.
 8. Build a confusion matrix
 9. Calculate precision and recall.

Conclusion:-

Dataset Used:- Restaurant Reviews.

Conclusion:- Thus I have completed this assignment and understood how to calculate precision recall and analyze text.

CODE

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

# Importing the dataset
dataset = pd.read_csv('Restaurant_Reviews.tsv', delimiter = '\t', quoting = 3)

# Cleaning the texts
import re
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
corpus = []
for i in range(0, 1000):
    review = re.sub('[^a-zA-Z]', ' ', dataset['Review'][i])
    review = review.lower()
    review = review.split()
    ps = PorterStemmer()
    review = [ps.stem(word) for word in review if not word in set(stopwords.words('english'))]
    review = ' '.join(review)
    corpus.append(review)

# Creating the Bag of Words model
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features = 1500)
X = cv.fit_transform(corpus).toarray()
y = dataset.iloc[:, 1].values

# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0)

# Fitting Naive Bayes to the Training set
from sklearn.naive_bayes import GaussianNB
classifier = GaussianNB()
classifier.fit(X_train, y_train)
```

```
# Predicting the Test set results
```

```
y_pred = classifier.predict(X_test)
```

```
# Making the Confusion Matrix
```

```
from sklearn.metrics import confusion_matrix
```

```
cm = confusion_matrix(y_test, y_pred)
```

```
recall = np.diag(cm) / np.sum(cm, axis = 1)
```

```
precision = np.diag(cm) / np.sum(cm, axis = 0)
```

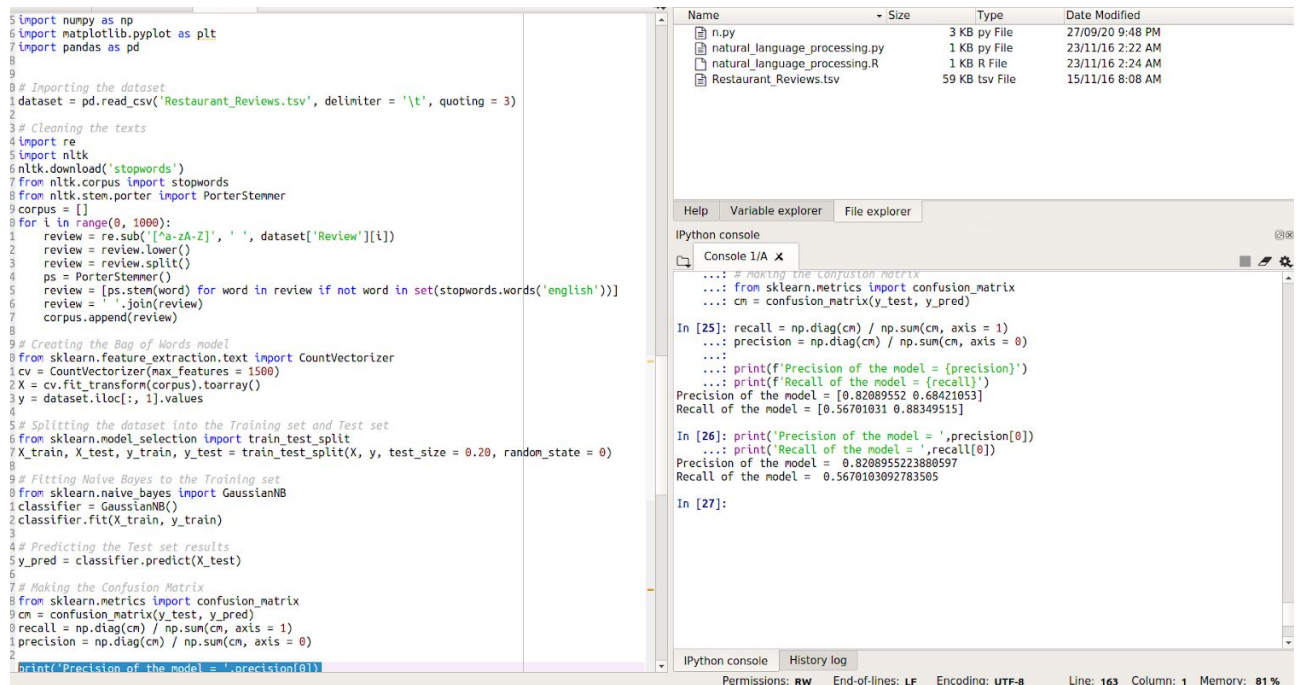
```
print('Precision of the model = ',precision[0])
```

```
print('Recall of the model = ',recall[0])
```

OUTPUT

Precision of the model = 0.8208955223880597

Recall of the model = 0.5670103092783505



```
5 import numpy as np
6 import matplotlib.pyplot as plt
7 import pandas as pd
8
9
10 # Importing the dataset
11 dataset = pd.read_csv('Restaurant_Reviews.tsv', delimiter = '\t', quoting = 3)
12
13 # Cleaning the texts
14 import re
15 import nltk
16 nltk.download('stopwords')
17 from nltk.corpus import stopwords
18 from nltk.stem.porter import PorterStemmer
19 corpus = []
20 for i in range(0, 1000):
21     review = re.sub('[^a-zA-Z]', ' ', dataset['Review'][i])
22     review = review.lower()
23     review = review.split()
24     ps = PorterStemmer()
25     review = [ps.stem(word) for word in review if not word in set(stopwords.words('english'))]
26     review = ' '.join(review)
27     corpus.append(review)
28
29 # Creating the Bag of Words model
30 from sklearn.feature_extraction.text import CountVectorizer
31 cv = CountVectorizer(max_features = 1500)
32 X = cv.fit_transform(corpus).toarray()
33 y = dataset.iloc[:, 1].values
34
35 # Splitting the dataset into the Training set and Test set
36 from sklearn.model_selection import train_test_split
37 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20, random_state = 0)
38
39 # Fitting Naive Bayes to the Training set
40 from sklearn.naive_bayes import GaussianNB
41 classifier = GaussianNB()
42 classifier.fit(X_train, y_train)
43
44 # Predicting the Test set results
45 y_pred = classifier.predict(X_test)
46
47 # Making the Confusion Matrix
48 from sklearn.metrics import confusion_matrix
49 cm = confusion_matrix(y_test, y_pred)
50 recall = np.diag(cm) / np.sum(cm, axis = 1)
51 precision = np.diag(cm) / np.sum(cm, axis = 0)
52
53 print('Precision of the model = ',precision[0])
```

Name	Size	Type	Date Modified
n.py	3 KB	py File	27/09/20 9:48 PM
natural_language_processing.py	1 KB	py File	23/11/16 2:22 AM
natural_language_processing.R	1 KB	R File	23/11/16 2:24 AM
Restaurant_Reviews.tsv	59 KB	tsv File	15/11/16 8:08 AM

```
Help  Variable explorer  File explorer
IPython console
Console I/O x
...: # Making the Confusion Matrix
...: from sklearn.metrics import confusion_matrix
...: cm = confusion_matrix(y_test, y_pred)
In [25]: recall = np.diag(cm) / np.sum(cm, axis = 1)
...: precision = np.diag(cm) / np.sum(cm, axis = 0)
...:
...: print(f'Precision of the model = {precision}')
...: print(f'Recall of the model = {recall}')
Precision of the model = 0.82089552 0.68421053
Recall of the model = 0.56701031 0.88349515
In [26]: print('Precision of the model = ',precision[0])
...: print('Recall of the model = ',recall[0])
Precision of the model = 0.8208955223880597
Recall of the model = 0.5670103092783505
In [27]:
IPython console  History log
Permissions: RW  End-of-lines: LF  Encoding: UTF-8  Line: 163  Column: 1  Memory: 81 %
```