a02

May 21, 2022

[107]:

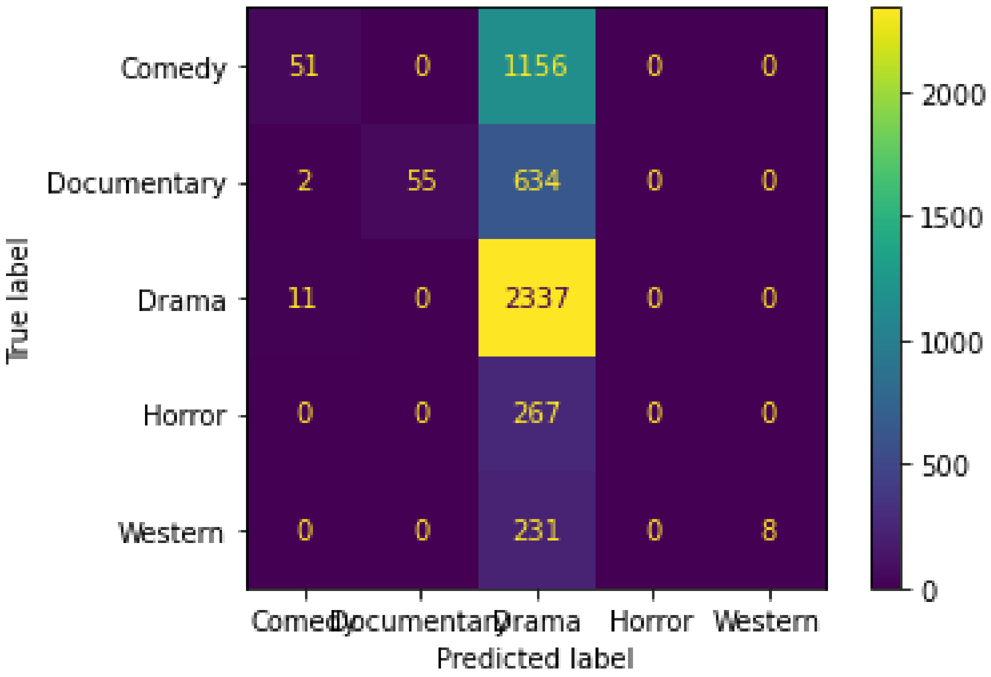
```
# A02 Assignment
# Author : Muhammad iftikhar
# Batch  : 18 Batch
# Course : Information Retrieval
import pandas as pd
import numpy as np
import re

# nltk
from nltk.tokenize import word_tokenize
import nltk
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
porter_stemmer = PorterStemmer()


# ML libs
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import plot_confusion_matrix
import matplotlib.pyplot as plt
def ReadingDataset(path:str):

    df = pd.read_csv(path)
    print(df.head())
    pass

def ConvertingTSV_To_CSV(tsvfilePath:str ,csvfilePath:str):

    moviesDic = {
        'title': [],
        'story': []
    }
    with open(tsvfilePath) as tsvfile:

        # iterate over the tsv file for subtitue the tab to comma
        for line in tsvfile:
            # split at tab
            contentfile = line.split('\t')

            # append to dictionary
            moviesDic['title'].append(contentfile[0])
            moviesDic['story'].append(contentfile[1])


    # print("Converted Successfully")
    return pd.DataFrame.from_dict(moviesDic).to_csv(csvfilePath,index=False)

def PreProcessing(Text:str):
    # remove links
    Text = re.sub(r'http\S+', '', Text)

    # remove alphanumeric character
    Text=re.sub(r'[\W_]+', ' ', Text)



    # Removing Stopwords from the TEXT
    stop_words = set(stopwords.words('english'))

    word_tokens = word_tokenize(Text)

    filtered_sentence = [w for w in word_tokens if not w.lower() in stop_words]

    filtered_sentence = []

    for w in word_tokens:
        if w not in stop_words:

            # apply Stemming at the same time for saving the time of further␣
 ↪loop
            # print(w , porter_stemmer.stem(w))
            filtered_sentence.append(porter_stemmer.stem(w))
    st = ' '
    # print('filtered sentance \n',filtered_sentence)
    return st.join(filtered_sentence)

def VectorizationData(DfPath:str):

    df = pd.read_csv(DfPath)
    for index, count in enumerate(range(0,len(df['story']))):
        df['story'][index] = PreProcessing( df['story'][index])
    print("Successfully PreProcessing.")
```

```
#     multi_class = {
#         'Drama' : 1,
#         'Comedy':2,
#         'Documentary':3,
#         'Western':4,
#         'Horror':5
#         }
#     for index, value in enumerate(df['title']):

#         if value=='Drama':
#             df['title'][index]=multi_class['Drama']
#             # print(df['title'][index])

#         elif value == 'Comedy':
#             df['title'][index]=multi_class['Comedy']

#         elif value == 'Documentary':
#             df['title'][index]=multi_class['Documentary']

#         elif value == 'Western':
#             df['title'][index]=multi_class['Western']

#         elif value == 'Horror':
#             df['title'][index]=multi_class['Horror']



    X = df['story']
    y = df['title']

    return X,y
def model(X_train, X_test, y_train, y_test):

    gnb = GaussianNB()
    from sklearn.pipeline import Pipeline
    from sklearn.feature_extraction.text import TfidfTransformer,CountVectorizer

    lr = Pipeline([('vect', CountVectorizer()),
                   ('tfidf', TfidfTransformer()),
                   ('gnb', MultinomialNB()),
                  ])

    lr.fit(X_train,y_train)
    y_pred = lr.predict(X_test)
```

```python
    # print(f"Accuracy is : {accuracy_score(y_pred,y_test)}")
    # print(confusion_matrix(y_test, y_pred))
    plot_confusion_matrix(lr, X_test, y_test)
    plt.show()


if __name__=="__main__":
    # PreProcessing
    # for training Datasets
    tsvpathtraining = "Datasets/film-genres-train.tsv"
    csvpathtraining = "Datasets/film-genres-train.csv"

    # for testing Dataset
    tsvpathtesting = "Datasets/film-genres-test.tsv"
    csvpathtesting = "Datasets/film-genres-test.csv"



    # conveting the dataset to csv
    # ConvertingTSV_To_CSV(tsvpathtesting ,csvpathtesting )

    # PreProcessing


    # Vectorization  for training data

    X_train,y_train = VectorizationData(csvpathtraining)

    # vectorization for testing datasets
    X_test,y_test = VectorizationData(csvpathtesting)


    # Gussain Machine learning model
    model(X_train, X_test, y_train, y_test)
```

```
Successfully PreProcessing.
Successfully PreProcessing.
```

```
/home/ifti/anaconda3/lib/python3.9/site-
packages/sklearn/utils/deprecation.py:87: FutureWarning: Function
plot_confusion_matrix is deprecated; Function `plot_confusion_matrix` is
deprecated in 1.0 and will be removed in 1.2. Use one of the class methods:
ConfusionMatrixDisplay.from_predictions or
ConfusionMatrixDisplay.from_estimator.
  warnings.warn(msg, category=FutureWarning)
```



1 Calculation for Accuracy

```
TP -- = Is the diagonal valus with respect to the valaue
FN -- = Sum of Row except TP
FP -- = Sum of column except TP
TN -- = Sum of all column and rows except column ,rows of respective values

* Comedy :
    formula , Accuracy = tp + tn / tp + tn + fn + fp
    Precision = 51 + 3,532 / (51 + 3,532  + 13 +1156) = 0.75

* Documentory :
    formula , Accuracy = tp + tn / tp + tn + fn + fp
    Precision = 55 + 4061 / (55 + 4061 + 0 + 636) = 0.86

 * Drama :
    formula , Accuracy = tp + tn / tp + tn + fn + fp
    Precision = 2337 + 750 / ( 2337 + 750 + 2288 +11 )= 0.57

 * Horror :
    formula , Accuracy = tp + tn / tp + tn + fn + fp
    Precision = 0 + 4485 / 0 + 4485 + 0 + 267 = 0.94

 * Western :
    formula , Accuracy = tp + tn / tp + tn + fn + fp
    Precision = 8 + 4513/ 8 + 4513 + 0 + 231 = 0.95
```

2 Calculation for Precision

```
TP -- = Is the diagonal valus with respect to the valaue
FN -- = Sum of Row except TP
FP -- = Sum of column except TP
TN -- = Sum of all column and rows except column ,rows of respective values

* Comedy :
    formula , precision = TP/TP + FP
    Precision = 51 / 51 + 13 = 0.79

* Documentory :
    formula , precision = TP / TP + FP
    Precision = 55 / 55 + 0 = 1

 * Drama :
    formula , precision = TP / TP + FP
    Precision = 2337 / 2337 + 2288 = 2337/4625 = 0.50

 * Horror :
    formula , precision = TP / TP + FP
    Precision = 0 / 0 + 0 =  = 0

 * Western :
    formula , precision = TP / TP + FP
    Precision = 8 / 8 + 0 = 1
```

3 Calculation for Recall

```
TP -- = the  diagonal valus correspoding to the value
FN -- = Sum of Row except TP
FP -- = Sum of column except TP
TN -- = Sum of all column and rows except column ,rows of respective values

* Comedy :
    formula , Recall = TP/TP + FN
    Recall = 51 / 51 + 1156 = 0.004

* Documentory :
    formula , Recall = TP / TP + FN
    Recall = 55 / 55 + 636 = 0.079

 * Drama :
    formula , Recall = TP / TP + FN
```

```
   Recall = 2337 / 2337 + 11 = 2337/2348 = 0.99

* Horror :
   formula , Recall = TP / TP + FN
   Recall = 0 / 0 + 267 = 0

* Western :
   formula , Recall = TP / TP + FN
   Recall = 8 / 8 + 231  = 0.033
```

```
[ ]:
```

```
[ ]:
```

```
[ ]:
```