CIS 434 Final Project Report

Xue Wang
05/03/2021

File that used for analyzing and predicting & the task

Three text files were given for analysis. They are “noncompliant.txt”, “compliant.txt”, that contains samples of customer tweets that are (not) considered by former TA, and “customertweets.csv” that contains samples of customer tweets that may or may not be complaints. Task is to find out as many noncompliant as I can from “customertweets.csv” file.

Steps for solving the task of this project

I use Spyder (Python 3.8) to finish this project.

1. Import the packages for later analyzing usage.

```
import re
import os
import sys
import pandas as pd
import numpy as np
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import CountVectorizer
from sklearn import svm
from sklearn.model_selection import train_test_split, cross_val_score, StratifiedKFold, GridSea
from sklearn.metrics import accuracy_score, f1_score, confusion_matrix
from sklearn.metrics import roc_auc_score, recall_score, precision_score
```

2. Set the path to read those file that have been downloaded in my local computer.

```
#%% Set the path

print("current directory is : " + os.getcwd())
os.chdir(f'{os.getenv("HOME")}/documents/spring/b/social media/final project')
print("current directory is : " + os.getcwd())
```

3. A. create the vector call “mystop” to store all the possible stop word in English, Spanish, Portuguese, French and German from the package stopwords.

   B. Read “noncompliant.txt” and “compliant.txt”, rename their columns names and add different types for each txt file, combine two txt files together as a whole data set for the training (and testing) data set.

C. Convert every tweet in the full data set into lower case and replace every non-verbal word （symbol, notation, etc.） in each tweet with space; join all tweets together with space; create the set that contains the unique word from split all tweets into words and delete the stopwords that we created before, store it in a list assign it to “vocab” as our vocabulary; use “CountVectorizer” by set the vocabulary equal to “vocab”; then apply “vectorizer.fit_transform” to the tweets (the list that contains tweets that converted to lower case and replaced all non-verbal words to space) and transform to an array. Assign this array as X; create a pandas series “y” that contain Boolean value of type of tweets is noncompliant.

C. Use the “train_test_split” to split the X and y into training and testing data set with proportion of 80% is training data set in order to build a text classifier model based on noncompliant.txt” and “compliant.txt”

```
mystop = set(stopwords.words("english"))|set(stopwords.words("spanish"))|set(stopwords.words("p

data1 = pd.read_csv("noncompliant.txt", header= None)
data1.columns = ['tweets']
data1["type"] = 1

data2 = pd.read_csv("compliant.txt", header= None)
data2.columns = ['tweets']
data2["type"] = 0

data = data1.append(pd.DataFrame(data = data2), ignore_index=True)

tweets = [re.sub("[^a-zA-Z]", " ", x.lower()) for x in data['tweets'].tolist()]
s = ' '.join( tweets )
vocab = list( set( s.split() ) - mystop )
vectorizer = CountVectorizer( vocabulary=vocab  )
X = vectorizer.fit_transform( tweets ).toarray()

y = data['type']== 1
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=23)
```

4. A. Use cross-validation and Grid- search to find the best model with the highest ROC accuracy score. (for SVM model), that is, to tuning the parameter to find the model with best performance. (parameter of “kernel” and “C”)
   B. use this model to predict X_test data set as “y_pred”; predict the probability of X_test data set as “y_prob”; create confusion matrix to show the prediction result; print the accuracy score, f1_score, precision, recall and the roc_auc_score given the prediction of X_test (y_pred) and the real y_test.

```
### Use cross-validation and Grid- search to find the best model with the highest accuracy score

kfolds = StratifiedKFold(n_splits=4, shuffle=True, random_state=1)
parameters = {'kernel':('rbf', 'sigmoid'), 'C':[2.5]}
grid_svm = GridSearchCV( svm.SVC(probability=True, class_weight="balanced"), parameters, cv=kfolds, sco
grid_svm.fit(X_train, y_train)
grid_svm.best_params_
## {'C': 2.5, 'kernel': 'rbf'}

grid_svm.best_score_
## 0.7974813633329633

grid_svm.score(X_test, y_test)
## 0.832945358063628

y_pred = grid_svm.predict( X_test )
y_prob = grid_svm.predict_proba( X_test )[:, 1]
roc_auc_score( y_test, y_prob )
## 0.832870266075943

accuracy_score( y_test, y_pred )
## 0.77

f1_score( y_test, y_pred )
## 0.7809523809523811

precision_score( y_test, y_pred )
## 0.7699530516431925

recall_score( y_test, y_pred )
## 0.7922705314009661
```

5. Apply this trained model to the “customertweets.csv”
   A. Import the csv file as “df” with “header = None” to create the data frame that contains the index and all customers’ tweets from this csv file.
   B. Rename its column’s name with “index” and “tweets”.
   C. As what we did early, convert every tweet in the data frame into lower case and replace every non-verbal word （symbol, notation, etc.） in each tweet with space.
   D. This time, apply “vectorizer.transform” (note: not fit_transform in the prediction because we don’t need to learn from the data anymore) to those converted tweets to get the array of the data frame.
   E. Predict the probability instead of Boolean value use the trained model for all tweets in the data frame.
   F. Create other column called “Probability” to store the probability of each tweet is noncompliant.
   G. Extract the tweets with probability higher than 96.8% from data frame and store those into the new data frame called “data”
   H. Extract the first two columns of all observations of “data” and assign it in “data1”.
   I. Finally export “data1 as csv file named “Noncompliant.csv” with index= False to export the data set without default index. (has 852 observations, two features in total)

```
#%% Apply the trained model to the csv file

## import the data
df = pd.read_csv("customertweets.csv", header= None)
df.columns = ['index','tweets']

## convert the format of the tweets
tweets = [re.sub("[^a-zA-Z]", " ", x.lower()) for x in df['tweets'].tolist()]
content = vectorizer.transform( tweets ).toarray()

clf = svm.SVC(probability=True, kernel='rbf', class_weight="balanced", C=2.5)
y_prob = clf.predict_proba( content )[:, 1]

df["Probability"] = y_prob

data = df.loc[df.iloc[:,2] > 0.968]
data1 = data.iloc[:,0:2]

data1.to_csv('Noncompliant.csv', index=False)
```

6. A. Open the “Noncompliant.csv”, manually check each tweet.
   B. Delete the tweet expresses anger, frustration, or disappointment towards an airline (because our model may predict the false positive result, that is, some tweets in our final csv file may be compliant but also included as noncompliant); and mention multiple airlines, lashing out at one while praising or swearing to fly with the other. (it’s relatively hard for model to predict for those tweets, model may predict some of them as noncompliant because it contains some features of noncompliant)
   C. Finalize the csv file and rename is as “Xue_Wang.csv” with 757 noncompliant tweets and two columns. (original index and the actual tweets)