The Final Report

Wang Mingxi

Jilin University
College of Computer Science and Technology

July 12, 2021

Overview

Research motivation and context

Project Objectives

- Determine whether the disaster reflected in the tweets is real

Project background I

A tweet that says there's a fire ~~which is in aba~~in ABA, ablaze means fire, it's the key word in the whole tweet, it means it's a disaster tweet, ~~and~~ so it's marked 1. It could also be a key word in "getting people angry," which is disaster_neutral Twitter, and it should be flagged zero.

A	B	C	D	E
38			Was in NY(	0
39			Love my g	0
40			Cooool :)	0
41			Do you like	0
44			The end!	0
48	ablaze	Birminghar	@bbcmtd	1
49	ablaze	Est. Septer	We always	0
50	ablaze	AFRICA	#AFRICAN	1

Project background II

So our task was to build a model that could pick out tweets that were actually disasters and mark them as ~~1.~~ 1, and distinguish between tweets that were not disasters and tweets that were not disasters and mark them as 0. The test.csv file has 3,263 such tweets waiting to be tagged.

Research contents and methods

Import the file AND Discard the same data


```
train=pd.read_csv('./nlp_getting_started/train.csv')
test=pd.read_csv('./nlp_getting_started/test.csv')

df = data
df = data.drop_duplicates().
```

Solve the category imbalance

- The imbalance of categories (i.e., labels, 0,1) causes the classifier to bias the test set prediction. Too many positive examples in the training set will lead to the model's tendency to predict the test set as positive examples and vice versa.
- Target = 0 has 4,322 tweets and target = 1 has 3,239 tweets. It's about 4 3. It's OK, don't change it.

Data cleaning


```
def clean_text(text):
temp = text.lower()
temp = re.sub('n', ' ' , temp)
temp = re.sub("́, ", temp)
temp = re.sub(", ' ', temp)
temp = re.sub(r'(http|https|pic.)S', ' ', temp)
temp = re.sub(r'[s]', ' ', temp)
return temp
```

Delete stop word


```
def remove_stopwords(text):
temp = [text for text in text.split() if len(text) > 3]
tokenized_words = word_tokenize(text)
temp = [word for word in tokenized_words if word not in stop_words]
temp = ' '.join(temp)
return temp
```

Data cleaning

- There is an artificial addition to the dataset of "clean" : the Twitter text after it has been cleaned.

train['clean'] = train['text'].apply(clean_text)
train['clean'] = train['clean'].apply(remove_stopwords)
train['clean'] = train['text'].apply(clean_text)
train['clean'] = train['clean'].apply(remove_stopwords)

Add artificial features

- In order to analyze the needs, we should first establish a new combination attribute~~.~~: that is, the cleaned feature and the keyword are combined with blank space as the new feature.

```
def combine_attributes(text, keyword):
var_list = [text, keyword]
combined = ' '.join(x for x in var_list if x)
return combined
train.fillna('', inplace = True)
train['combine'] = train.apply(lambda x:
combine_attributes(x['clean'],x['keyword']), axis = 1)
test.fillna('', inplace = True)
test['combine'] = test.apply(lambda x:
combine_attributes(x['clean'],x['keyword']), axis = 1)
```

Data set segmentation

- The training data were divided according to the ratio of 8 2, with 0.8 of the data training model and 0.2 of the data testing the training effect of the model.

```
X = train['combine']
y = train['target']
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.8)
```

Word frequency inverse text frequency (TF IDF) processing

- The word frequency_inverse text frequency (TF_IDF) processing is used to add weight to words in the text.

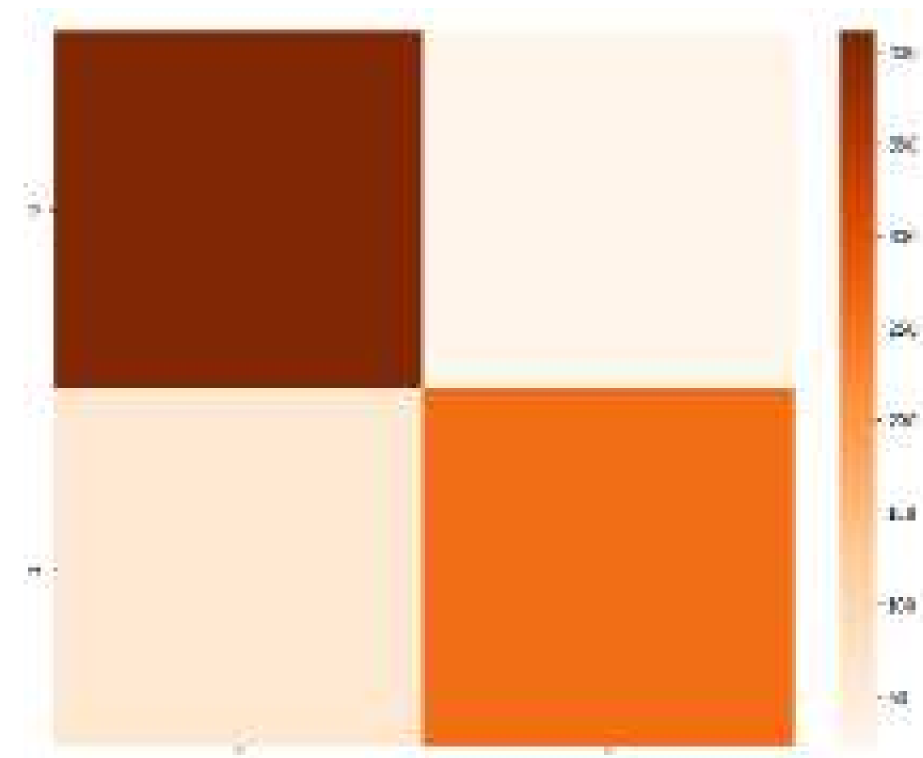
```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer()

X_train_vect = vectorizer.fit_transform(X_train)
X_train_vect_all = vectorizer.transform(train['clean'])
X_test_vect = vectorizer.transform(X_test)
X_test_vect_all = vectorizer.transform(test['clean'])
```

Visualization heat map I

import seaborn as sns from sklearn.metrics import accuracy_score, confusion_matrix import matplotlib.pyplot as plt conf_mat = confusion_matrix(y_test, y_pred) fig, ax = plt.subplots(figsize=(10,8)) sns.heatmap(conf_mat,cmap="Oranges") plt.ylabel('actually',fontsize=18) plt.xlabel('predict',fontsize=18)



Visualization heat map II

As you can see, there are fewer categories of 1 (target = 1) and more categories of 0. And the prediction was wrong more often when category 0 was classified as category 1 than when 1 was classified as category 0.

The model was created by "support vector machine classification" SVC


```
from sklearn.svm import SVC
from sklearn.svm import LinearSVC

clf = SVC(kernel = 'linear')
clf.fit(X_train_vect, y_train)

y_pred = clf.predict(X_test_vect)
```

Conclusion

Evaluation model


```
from sklearn.metrics import accuracy_score
accuracy_score(y_test, y_pred)
```

RESULT : 0.8006

Contact Information

Wang Mingxi

College of Computer Science and Technology

Jilin University, China

mxwang@tulip.academy

Team for Universal Learning and Intelligent Processing