Project Report 22

Applications in Natural Language Processing

NAME: DHIVAKAR.R

COURSE: AI and ML

Batch : Aug 2020

Question:

Using NLP we can easily analyse any given text. The steps involved for such an analysis are tokenization, pre processing each word and then finally vectorising each of them. One of the most common and easy to implement vectorisation algorithm is BoW. Using BoW and NLTK for processing, implement a simple spam filter that marks all the spam texts as dangerous.

Prerequisites

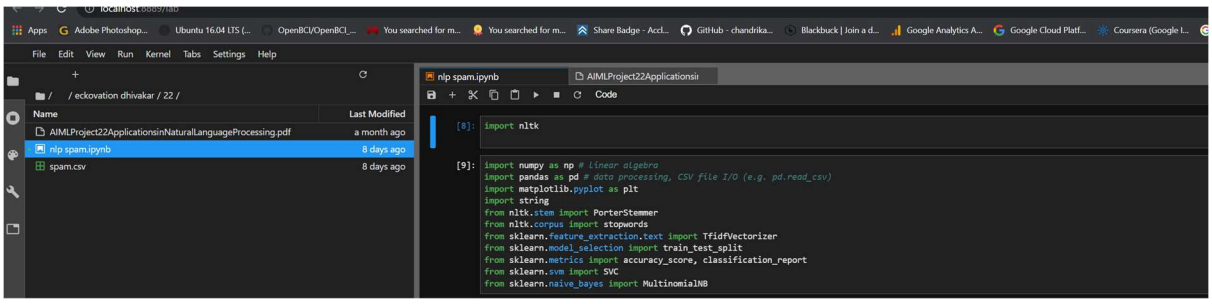
What things you need to install the software and how to install them:

Python 3.6 This setup requires that your machine has latest version of python. The following url https://www.python.org/downloads/ can be referred to download python. Once you have python downloaded and installed, you will need to setup PATH variables (if you want to run python program directly, detail instructions are below in how to run software section). To do that check this: https://www.pythoncentral.io/add-python-to-path-python-is-not-recognized-asan-internal-or-externalcommand/ . Setting up PATH variable is optional as you can also run program without it and more instruction are given below on this topic.

Second and easier option is to download anaconda and use its anaconda prompt to run the commands. To install anaconda check this url https://www.anaconda.com/download/ You will also need to download and install below 3 packages after you install either python or anaconda from the steps above Sklearn (scikit-learn) numpy scipy if you have chosen to install python 3.6

Implementation

Importing the libraries and dataset



Loading the dataset

```
[10]: nltk.download('stopwords')
      ps = PorterStemmer()

[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\dhiva\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!

[11]: data = pd.read_csv('spam.csv')
      data.head()
```

[11]:

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

Finding the length of the text from the dataset

```
[12]: data = data.drop(["Unnamed: 2", "Unnamed: 3", "Unnamed: 4"], axis=1)
      data = data.rename(columns={"v1":"class", "v2":"text"})
      data.head()
```

[12]:

	class	text
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

```
[13]: data['length'] = data['text'].apply(len)
      data.head()
```

[13]:

	class	text	length
0	ham	Go until jurong point, crazy.. Available only ...	111
1	ham	Ok lar... Joking wif u oni...	29
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	155
3	ham	U dun say so early hor... U c already then say...	49
4	ham	Nah I don't think he goes to usf, he lives aro...	61

```
[14]: data['class'].value_counts()

[14]: ham     4825
      spam     747
      Name: class, dtype: int64

[32]: stop_words = nltk.corpus.stopwords.words('english')
      stop_words[:5]

[32]: ['i', 'me', 'my', 'myself', 'we']

[33]: def pre_process(text):
          text = text.translate(str.maketrans('', '', string.punctuation))
          text = [word for word in text.split() if word.lower() not in stopwords.words('english')]
          words = ""
          for i in text:
                  words += (ps.stem(i))+" "
          return words
```

```
[39]: text = data['text'][0]
      text

[39]: 'Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...'

[40]: text = text.translate(str.maketrans('', '', string.punctuation))
      text = [word for word in text.split() if word.lower() not in stopwords.words('english')]
      text[:5]

[40]: ['Go', 'jurong', 'point', 'crazy', 'Available']

[41]: text = data['text'][0]
      text

[41]: 'Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...'

[42]: pre_process(text)

[42]: 'Go jurong point crazi avail bugi n great world la e buffet cine got amor wat '
```

Finding the text of the spam messages

```
[43]: data['text']

[43]: 0       Go until jurong point, crazy.. Available only ...
      1                           Ok lar... Joking wif u oni...
      2       Free entry in 2 a wkly comp to win FA Cup fina...
      3       U dun say so early hor... U c already then say...
      4       Nah I don't think he goes to usf, he lives aro...
                                          ...
      5567    This is the 2nd time we have tried 2 contact u...
      5568                Will Ì_ b going to esplanade fr home?
      5569    Pity, * was in mood for that. So...any other s...
      5570    The guy did some bitching but I acted like i'd...
      5571                           Rofl. Its true to its name
      Name: text, Length: 5572, dtype: object

[44]: textFeatures = data['text'].copy()
      textFeatures = textFeatures.apply(pre_process)
      textFeatures[:5]

[44]: 0    Go jurong point crazi avail bugi n great world...
      1                                Ok lar joke wif u oni
      2    free entri 2 wkli comp win FA cup final tkt 21...
      3                U dun say earli hor U c alreadi say
      4           nah dont think goe usf live around though
      Name: text, dtype: object

[45]: vectorizer = TfidfVectorizer(ngram_range=(1, 2))
      features = vectorizer.fit_transform(textFeatures)
      features.shape

[45]: (5572, 39213)

[47]: features_train, features_test, labels_train, labels_test = train_test_split(features, data['class'], test_size=0.3, random_state=111)

[48]: features_train.shape, features_test.shape, len(labels_train), len(labels_test)

[48]: ((3900, 39213), (1672, 39213), 3900, 1672)
```

Prediction using Support vector & Multinomial naïve bayes Model

```
[49]: # Prediction using Support Vector Machine
      svc = SVC(kernel='sigmoid', gamma=1.0)
      svc.fit(features_train, labels_train)
      prediction = svc.predict(features_test)
      # accuracy_score(labels_test,prediction)
      print(classification_report(labels_test, prediction, target_names = ['ham', 'spam']))
```

```
              precision    recall  f1-score   support

         ham       0.98      1.00      0.99      1440
        spam       0.99      0.85      0.91       232

    accuracy                           0.98      1672
   macro avg       0.98      0.93      0.95      1672
weighted avg       0.98      0.98      0.98      1672
```

```
[50]: # Prediction using Multinomial Naive Bayes Model
      mnb = MultinomialNB(alpha=0.2)
      mnb.fit(features_train, labels_train)
      prediction = mnb.predict(features_test)
      # accuracy_score(labels_test,prediction)
      print(classification_report(labels_test, prediction, target_names = ['ham', 'spam']))
```

```
              precision    recall  f1-score   support

         ham       0.99      0.99      0.99      1440
        spam       0.96      0.93      0.95       232

    accuracy                           0.99      1672
   macro avg       0.97      0.96      0.97      1672
weighted avg       0.98      0.99      0.98      1672
```

```
[ ]:
```