07_BagOfWords_v1.0-PauloBraga

July 20, 2020

```
[8]: '''
         Paulo Simplício Braga
         20.07.2020
     '''
     import pandas as pd
     import nltk
     import re
     from bs4 import BeautifulSoup
     from nltk.corpus import stopwords
```

```
[9]: train_df = pd.read_csv("Data/labeledTrainData.tsv", header=0,\
                             delimiter="\t", quoting=3)
```

```
[10]: train_df.head()
```

```
[10]:          id  sentiment                                             review
      0  "5814_8"          1  "With all this stuff going down at the moment …
      1  "2381_9"          1  "\"The Classic War of the Worlds\" by Timothy …
      2  "7759_3"          0  "The film starts with a manager (Nicholas Bell…
      3  "3630_4"          0  "It must be assumed that those who praised thi…
      4  "9495_8"          1  "Superbly trashy and wondrously unpretentious …
```

```
[11]: train_df.shape
```

```
[11]: (25000, 3)
```

```
[12]: # Função para converter o review em uma string de palavras.
      # Input: review string
      # Output: string pré-processada do review
      def review_to_words( raw_review ):
          # 1. Remove HTML
          review_text = BeautifulSoup(raw_review).get_text()
          #
          # 2. Remove non-letters
          letters_only = re.sub("[^a-zA-Z]", " ", review_text)
          #
          # 3. Converte para minúsculo e separa as palavras
```

```
    words = letters_only.lower().split()
    #
    # 4. Converte a 'stop words' para um conjunto, com o intuíto de
    # ganhar velocidade no processamento
    stops = set(stopwords.words("english"))
    #
    # 5. Remove as 'stop words'
    meaningful_words = [w for w in words if not w in stops]
    #
    # 6. Retorna uma string de palavras tratadas
    return( " ".join( meaningful_words ))
```

[14]:
```
clean_review = review_to_words(train_df['review'][1])
print(clean_review)
```

```
classic war worlds timothy hines entertaining film obviously goes great effort
lengths faithfully recreate h g wells classic book mr hines succeeds watched
film appreciated fact standard predictable hollywood fare comes every year e g
spielberg version tom cruise slightest resemblance book obviously everyone looks
different things movie envision amateur critics look criticize everything others
rate movie important bases like entertained people never agree critics enjoyed
effort mr hines put faithful h g wells classic novel found entertaining made
easy overlook critics perceive shortcomings
```

[15]:
```
# Pega a quantidade exata de reviews
num_reviews = train_df['review'].size

# Lista para guardar os reviews limpos
clean_train_reviews = []

# Rotina para passar por todos os reviews do data frame e
# adicioná-los a lista 'clean_train_reviews'
for i in range(0, num_reviews):
    if( (i+1)%1000 == 0 ):
        print("*", end='')
    clean_train_reviews.append(review_to_words(train_df['review'][i]))
```

```
*************************
```

[16]:
```
from sklearn.feature_extraction.text import CountVectorizer

# Cria um objeto do CountVectorizer
vectorizer = CountVectorizer(analyzer = "word",   \
                             tokenizer = None,    \
                             preprocessor = None, \
                             stop_words = None,   \
                             max_features = 5000)
```

```
# O fit_transform() irá aprender o vocabulário e transformar
# os dados de treinamento em vetores
train_data_features = vectorizer.fit_transform(clean_train_reviews)

# Converte para uma matriz para trabalhar com numpy
train_data_features = train_data_features.toarray()
```

[17]:
```
train_data_features.shape
```

[17]:
```
(25000, 5000)
```

[20]:
```
import numpy as np

vocab = vectorizer.get_feature_names()
# Transforma em uma matriz numpy
dist = np.sum(train_data_features, axis = 0)

# O laço a seguir mostra a quantidade de ocorrência das
# palavras (exemplo com as 5 primeiras)
cnt = 0
for tag, count in zip(vocab, dist):
    if cnt>4:
        break
    print(count, tag)
    cnt+=1
```

```
187 abandoned
125 abc
108 abilities
454 ability
1259 able
```

[154]:
```
from sklearn.ensemble import RandomForestClassifier

# Cria um objeto do tipo Random Forest para fazer a classificação
forest = RandomForestClassifier(n_estimators = 100)

# Treina o conjunto de treinamento, utilizando o saco de palavras
# como feature e 'sentiment' como target
forest = forest.fit(train_data_features, train_df["sentiment"])
```

[155]:
```
# Lendo o dataset de teste
test_df = pd.read_csv("Data/testData.tsv", header=0, delimiter="\t", \
                   quoting=3 )
test_df.shape
```

[155]:
```
(25000, 2)
```

[157]:
```
# Rotina para fazer a limpeza do data set, utilizando a função
# review_to_words, declarada anteriormente
num_reviews = len(test_df["review"])
clean_test_reviews = []
for i in range(0,num_reviews):
    if( (i+1) % 1000 == 0 ):
        print("*", end='')
    clean_review = review_to_words(test_df["review"][i])
    clean_test_reviews.append( clean_review )
```

```
*************************
```

[158]:
```
# Cria um bag of words para o set de teste e transforma em matriz
test_data_features = vectorizer.transform(clean_test_reviews)
test_data_features = test_data_features.toarray()
```

[159]:
```
# Utilizando o modelo treinado anteriormente, faz a previsão na
# base de teste
result = random_forest.predict(test_data_features)
```

[163]:
```
# Cria um data frame para guardar os resultados da predição na
# base de teste
result_df = pd.DataFrame( data={"id":test_df["id"],\
                    'review':test_df['review'],"sentiment":result} )
```

[164]:
```
# Utiliza a função a sample() do pandas para selecionar 5
# comentários aleatórios e mostra o resultado da aplicação do
# classificador
result_df.sample(5)
```

[164]:
```
                  id                                             review  \
2476     "4538_1"  "This was an awful short film that tries to be…
6127     "2276_8"  "I saw this movie at the 2005 Toronto Internat…
1726   "11370_10"  "I saw this mini-series when I was in high sch…
21486   "11601_2"  "I only voted it 2/10 mainly because Hitchcock…
7508    "5536_10"  "Married To The Mob was one of the first VHS t…

       sentiment
2476           0
6127           0
1726           1
21486          0
7508           1
```