Classify Pubmed Data and Create Train &Test CSVs

November 16, 2017

```
In [ ]: # -*- coding: utf-8 -*-
        """
        Created on Mon Nov 13 12:54:29 2017
        @author: Gina
        """
        #===============================================================================
        #Create train and test sentiment CSVs from PUBMED data
        #===============================================================================
        import csv
        from textblob import TextBlob
        import pandas as pd
        import numpy as np

        #===============================================================================
        #Read Pubmed CSV to Pandas data frame
        #===============================================================================
        df1=pd.read_csv("pubmedAbstractOnly.csv", encoding = "ISO-8859-1")

        #===============================================================================
        #Send Abstract to Textblob for Sentiment analysis
        #===============================================================================
        df1 = sentiment(df1)

        def sentiment(df):
            #Process Data
            #Add new df columns with sentiment, polarity and nouns. Start with 1st row
            df['sentiment'] = df.apply(lambda x: TextBlob(x['Abstract']).sentiment,axis=1)
            df['polarity'] = df.apply(lambda x: TextBlob(x['Abstract']).sentiment.polarity,axis
            df['noun'] = df.apply(lambda x: TextBlob(x['Abstract']).noun_phrases,axis=1)
            df['subjectivity'] = df.apply(lambda x:
                                    TextBlob(x['Abstract']).sentiment.subjectivity,axis=1)

            #Add new df column to determine if polarity is positve, negative, or neutral.
            df['classify'] = np.where(df['polarity'] > 0,'positive',
                                  np.where(df['polarity'] == 0, 'neutral',
                                                               'negative'))
            return df
```

```
#==============================================================================
#convert dataframe to list
#==============================================================================
df1list = df1[['Abstract','polarity','subjectivity','classify']].values.tolist()

#==============================================================================
# 60% train data
#==============================================================================
L = len(df1list)
train_index = int(.60 * L)

#==============================================================================
# Create 60% Train data and 40% test data
#==============================================================================
df_train, df_test = df1list[:train_index],df1list[train_index:]

#==============================================================================
#Write TRAIN data to a CSV file path
#Add header names
#==============================================================================
with open("trainpubmed2016_17.csv", "w", newline='',encoding = "ISO-8859-1") as f:
        writer = csv.writer(f)
        for line in df_train:
            writer.writerow(line)
f.close()

dfheader = pd.read_csv('trainpubmed2016_17.csv', encoding = "ISO-8859-1")
dfheader.columns = ["Abstract", "polarity","subjectivity","classify"]
dfheader.to_csv('trainpubmed2016_17.csv')


#==============================================================================
#Write TEST data to a CSV file path
#Add header names
#==============================================================================


with open("testpubmed2016_17.csv", "w", newline='',encoding = "ISO-8859-1") as f:
        writer = csv.writer(f)
        for line in df_test:
            writer.writerow(line)
f.close()

dftheader = pd.read_csv('testpubmed2016_17.csv', encoding = "ISO-8859-1")
dftheader.columns = ["Abstract", "polarity","subjectivity","classify"]
dftheader.to_csv('testpubmed2016_17.csv')
```