News and Market Sentiment Analytics Exam

(Fall 2021)

University of Southern Denmark

Practical Information

This is an individual exam. The final output should be a report with the answers to the problem statements below. Together with the report, you should upload your code. Readability in your code does count such that I can easily see what you've done.

Code fragments in the report is not a necessity unless you deem it so, but plots and other descriptives that are relevant to the given question should be included.

If you use code from the web (which is allowed), such as Kaggle kernels, remember to reference the source.

On the front page of your report, please state your exam number.

Overview

For this exam, you will be utilising the trump-tweets-corpora from Kaggle together with returns based on the SP500 throughout the same timeperiod which are already merged into the tweet dataset. The returns are calculated as the percentage change from today to tomorrow and should therefore be used directly as an outcome variable without lagging the variables further.

You will write programs to thoroughly investigate this corpus, by utilising various techniques covered in the course. The ultimate goal is to develop a stock market classifier, capable of predicting the directional outcome of the stock market based on tweets by Donald Trump.

While high accuracy/performance of your developed techniques is a plus, it is far more important that you discuss the reasoning behind your decisions. The reasons for a low

performing classifier could easily be fully credited to noisy data, as opposed to your decisions throughout development.

You cannot use other third-party NLP libraries if not explicitly stated in the problem description. Each question contains a percentage weight which is proportional to the effort required and/or difficulty.

Problem 1, String Manipulation (10%)

Question 1.1 (10%) Load the tweet dataset and write a program that combines all the tweets (the 'content' column in the dataframe) into one single text object. Be sure to make all words lower case.

```
tweet_1 = 'This is the first tweet'
tweet_2 = 'This is the second tweet'

# Your program
combine(data)
>>> ['this', 'is', 'the', 'first', 'tweet',
            'this', 'is', 'the', 'second', 'tweet']
```

Problem 2, Lexical Diversity (10%)

Question 2.1 (5%) Report the number of words in the combined text as well as in the vocubalary, how lexical diverse are the tweets?

Question 2.2 (5%) Are the tweets from Trump more or less lexical diverse than e.g. the news category from the brown corpus?

Problem 3, Comparing Vocabularies (15%)

Question 3.1 (5%) Find the collocation of Trump's tweets and report them are you surprised by any of these?

Question 3.2 (5%) List the most frequent bigrams, are these the same as the collocations, why?

Question 3.3 (5%) Finally, stem the words from the tweet corpus using the Porter stemmer, why does this reduce the vocabulary and by how much is the vocabulary reduced?

Problem 4, Frequency Distributions (15%)

Question 4.1 (5%) Write a program that takes as input a cleaned text and outputs a frequency distribution of tags in a given text.

Question 4.2 (5%) Use the tagged words of the "news" from the Brown corpus to solve the following problem. Train a unigram tagger using the universal tagset and plot the distribution of tags of the tweet corpus.

Question 4.3 (5%) Why is a large part of the tweet corpus tagged as 'None'? Use the most common tag of the 'News' corpus as a backoff tagger and plot the distribution of tags from the tweet corpus again.

Problem 5, Classification (30%)

In the following exercise, we are testing a classification model for predicting the direction of the stock market. To do so, we first need a feature selector. The following feature selector is our benchmark, which test whether a word is contained in a text.

```
all_words = nltk.FreqDist(w.lower() for w in combined_text)
word_features = list(all_words)[:100]

def document_features(document,word_features):
    document_words = set(document)
    features = {}
    for word in word_features:
        features['contains({})'.format(word)] = (word in document_words)
    return features
```

Before using our classifier, we should define a training, development test dataset, and a final test dataset.

- the training dataset should be defined as anything before January the 1'st 2016

- the development test dataset should be defined as (including) January the 1'st 2016 until January the 1'st 2017
- the test dataset should be defined as anything after (including) January the 1'st 2017.

Question 5.1 (5%) Create a binary indicator for whether the upcoming return is going to be positive or negative

Question 5.2 (5%) Create the training, development test dataset and test dataset

Question 5.3 (5%) Test the feature selector and use the nltk.NaiveBayesClassifier() to extract the top-10 most informative features and comment on your findings.

Question 5.4 (5%) How well is your model performing on the SP500 binary return? Be sure to report a confusion matrix and discuss your results.

```
from nltk.metrics import ConfusionMatrix
print(ConfusionMatrix(preds,y_dev))
```

Question 5.5 (10%) Discuss methods for improving your model. Adopt one of these methods and test if it outperforms your previous model. Keep on only testing on the development test dataset until Problem 6.

Problem 6, Trading Strategy (20%)

Note; as Problem 6 is built on top on Problem 5 you should only solve either Question 6.1 (a) **or** Question 6.1 (b), dependent on whether you managed to solve Problem 5 or not. If you managed to build a classifiation model, you should try to answer Question 6.1 (a). If you did not solve Problem 5 you should try to answer Question 6.1 (b).

You can use the cumulative sum of the returns as an approximation to define whether the overall return is positive. Assume that you can only hold the stocks or not hold them (i.e. we do not take into account shorting stocks). Hence, whenever you predict that the stock will

go down, you choose to not hold the stock (since that would yield a negative return if you are correct). Whenever you predict that the stock will go up, for a given day, you decide to hold the stock, and can therefore include the return in your cumulative sum of the returns.

Question 6.1 (a) (20%) Calculate the return of your investment strategy. If we want to adopt a trading strategy based on your findings, would your model have been profitable (we are assuming that there are no transaction costs)? Be sure to take into account that certain tweets are from the same date, and the dates are therefore not necessarily unique in your dataset.

Question 6.1 (b) (20%) Calculate the return of your investment strategy. If you did not manage to solve Problem 5, feel free to adopt a random trading strategy and discuss whether that strategy would be profitable or not, remember to incorporate into your model that multiple tweets appear on the same date.