Language Analysis for Securities Price Prediction

RESEARCH PROPOSAL

Hart Traveller | Psychology of Language | Feb 2019

ABSTRACT

This research project seeks to use NLP (natural language processing) with news and financial reports to try and predict the price of securities. News data, as well as quantitative data relating to the DJIA will be processed, before being fed into a machine learning algorithm – specifically a logistic regression algorithm. After a functional model is produced, a train-test split is performed on the data, and the model will be implemented on historical data to gauge its accuracy. Future research will potentially involve a more comprehensive dive into the different machine learning models available for this task, as well as the construction of more complex NLP models for text processing and classification.

NATURAL LANGUAGE PROCESSING

Natural language processing is defined as "a tract of Artificial Intelligence and Linguistics, devoted to make computers understand the statements or words written in human languages."[1] There are many applications of NLP, including text classification, artificial intelligence assistants, document summarization, speech recognition, and as we will be researching – stock market prediction. For our purposes here, we will implement a proof of concept in python. This will be combined with pandas for data cleaning and scikit-learn for implementation of machine learning models. The data used is from Kaggle and is provided by user Aaron7sun.[2]

EXISTING RESEARCH

There has been a significant amount of research done in the field of natural language processing and stock market prediction. In this research proposal, I will be referencing a paper titled: "Using NLP on news headlines to predict index trends" by Marc Velay and Fabrice Daniel for information pertaining to the subject, as well as using my own coding expertise to do a short proof of concept project.

RESEARCH QUESTION

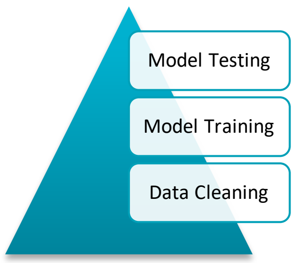
Is it possible to predict the DJIA using news headline text data?

[1] Diksha Khurana, Aditya Koli, Kiran Khatter, Sukhdev Singh. *Natural Language Processing: State of The Art, Current Trends and Challenges*
[2] https://www.kaggle.com/aaron7sun/stocknews

HYPOTHESIS

It will be possible to gain "better than random" prediction using historical data. Given the dissemination of prediction models however, the model applied to historical data may not be as effective when applied to current data.

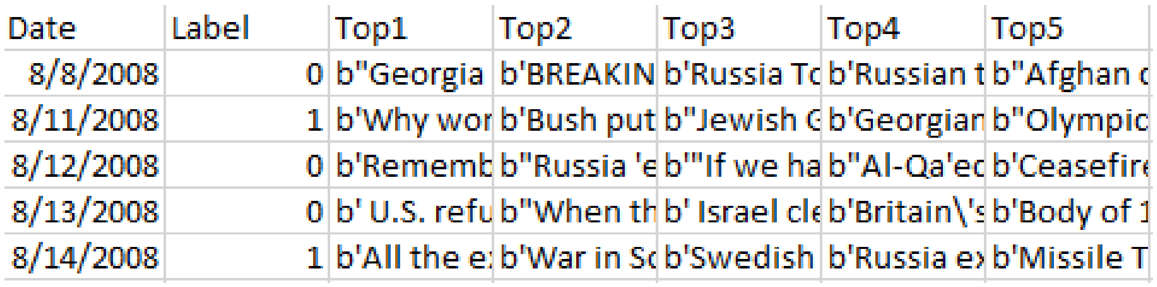
PROOF OF CONCEPT[3]



There are three main components to the proof of concept. First – taking the data and cleaning it so it can be useful in a logistic regression test as well as splitting it into a train portion and a test portion. We will be using logistic regression specifically because in the paper referenced[4], their tests showed logistic regression to be the most effective. Once this is done, a logistic regression model can be trained on the train portion of the data. Once the trained model is finished, it can be tested on the test portion of the data, and the effectiveness of the model can be measured. I have done all this in python, and the results of the test are shown below.

PROOF OF CONCEPT IMPLEMENTATION

First the data was loaded in to python using the pandas library. The data has a date column that specifies the day the news headlines were run, as well as a label column for whether the DJIA made gains or losses that day. There are then 25 columns to the right of this, representing the 25 top news headlines of that day on the news subreddit.



Date	Label	Top1	Top2	Top3	Top4	Top5
8/8/2008	0	b"Georgia	b'BREAKIN	b'Russia Tc	b'Russian t	b"Afghan c
8/11/2008	1	b'Why wor	b'Bush put	b"Jewish G	b'Georgian	b"Olympic
8/12/2008	0	b'Rememb	b"Russia 'e	b'"If we ha	b"Al-Qa'ec	b'Ceasefire
8/13/2008	0	b' U.S. refu	b"When th	b' Israel cle	b'Britain\'s	b'Body of 1
8/14/2008	1	b'All the e:	b'War in Sc	b'Swedish	b'Russia ex	b'Missile T

[3] For a copy of the technical documentation and code used in this project – please refer to my github page, and specifically the github repository ML-LR-DJIA
Link: https://github.com/harttraveller/ML-LR-DJIA

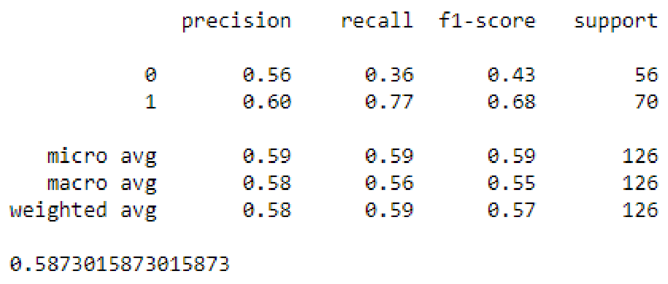
[4] Velay, Marc and Fabrice Daniel. *Using NLP on news headlines to predict index trends.* Artificial Intelligence Department of Lusis, Paris, France. June 2018

There are a total of 27 columns and around 2000 rows of news in this dataset, making up around ~55,000 news headlines. After loading in the data, I performed a train test split. This allows us to train the machine learning model on one section of the data, and then test its accuracy on another. The training data consisted of all news headlines before June 1st, 2015, and the testing data contained all the text after that date.

Through several lines of code[5], I was able to clean the data, removing all punctuation, as well as joining the separate text columns into one text cell for each date. After this all I needed to do was load a logistic regression model from scikit-learn into the program, fit it to my training data, and test it on my testing data.

RESULTS

The classification report and accuracy score of the testing done are shown below with a logistic regression ML model are shown below.



```
              precision    recall  f1-score   support

           0       0.56      0.36      0.43        56
           1       0.60      0.77      0.68        70

   micro avg       0.59      0.59      0.59       126
   macro avg       0.58      0.56      0.55       126
weighted avg       0.58      0.59      0.57       126

0.5873015873015873
```

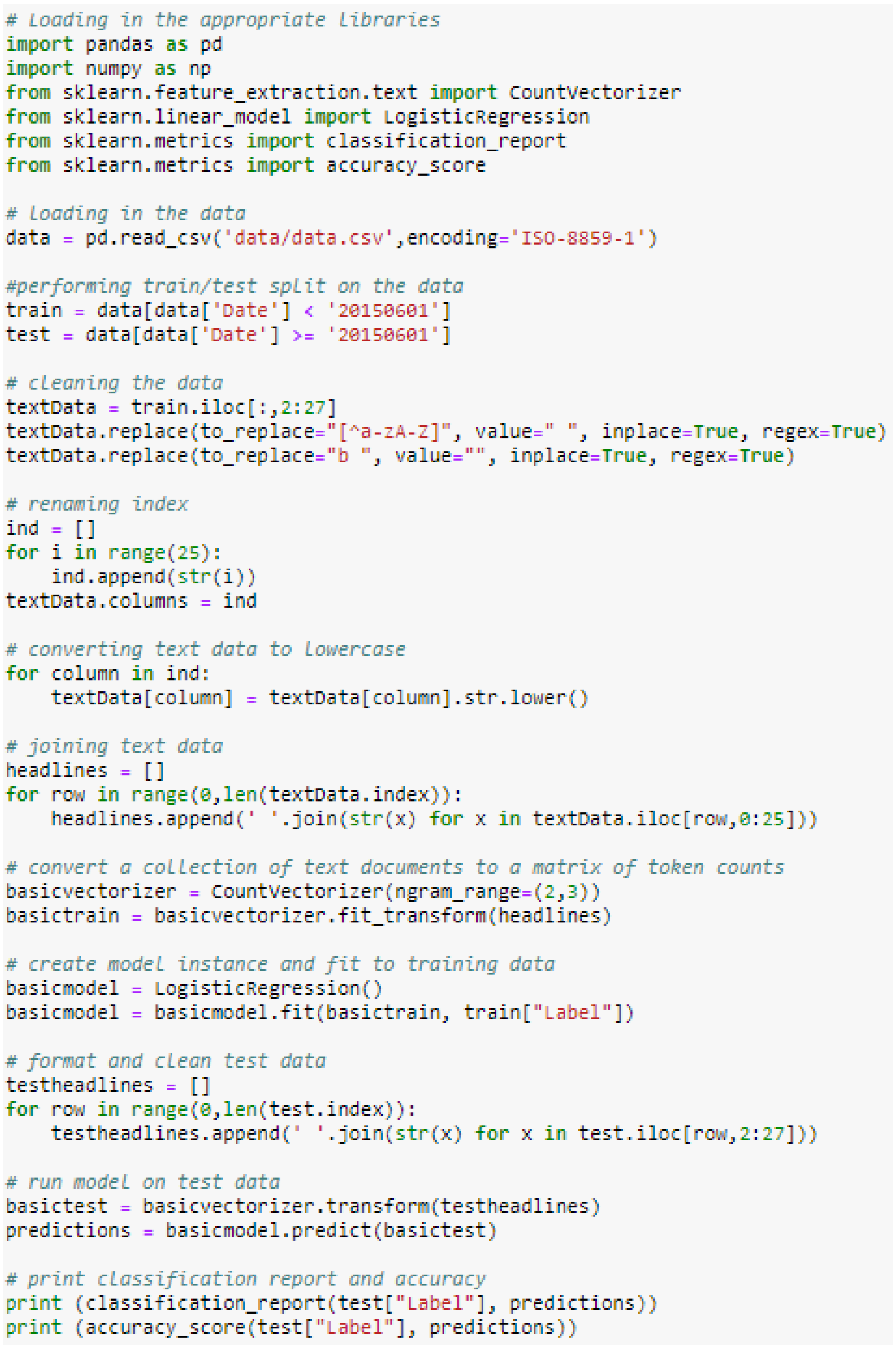
The model was around 59% accurate at predicting whether the market would go up or down in a given day. This does not prove the hypothesis, however it shows that there is definite potential in this field of research.

CONTINUING RESEARCH

This short research proposal by no means represents totally original research. Lots of work has been done in the field of natural language processing and stock market prediction. In the future however, I would see the proof of concept project done in this paper as forming the groundwork for more extensive research into different kinds of models that might be used. Adaptive models that improve over time, obfuscated models that have different levels to filter out noise and focus on signal, models that attempt to implement quantitative data and technical analysis side by side with NLP, as well as models that have conditional logic built into them by humans – as it's difficult to replicate good old-fashioned human intuition in machines.

[5] Program code is included at bottom of report for reference.

CODE DOCUMENTATION



```python
# Loading in the appropriate libraries
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score

# Loading in the data
data = pd.read_csv('data/data.csv',encoding='ISO-8859-1')

#performing train/test split on the data
train = data[data['Date'] < '20150601']
test = data[data['Date'] >= '20150601']

# cleaning the data
textData = train.iloc[:,2:27]
textData.replace(to_replace="[^a-zA-Z]", value=" ", inplace=True, regex=True)
textData.replace(to_replace="b ", value="", inplace=True, regex=True)

# renaming index
ind = []
for i in range(25):
    ind.append(str(i))
textData.columns = ind

# converting text data to lowercase
for column in ind:
    textData[column] = textData[column].str.lower()

# joining text data
headlines = []
for row in range(0,len(textData.index)):
    headlines.append(' '.join(str(x) for x in textData.iloc[row,0:25]))

# convert a collection of text documents to a matrix of token counts
basicvectorizer = CountVectorizer(ngram_range=(2,3))
basictrain = basicvectorizer.fit_transform(headlines)

# create model instance and fit to training data
basicmodel = LogisticRegression()
basicmodel = basicmodel.fit(basictrain, train["Label"])

# format and clean test data
testheadlines = []
for row in range(0,len(test.index)):
    testheadlines.append(' '.join(str(x) for x in test.iloc[row,2:27]))

# run model on test data
basictest = basicvectorizer.transform(testheadlines)
predictions = basicmodel.predict(basictest)

# print classification report and accuracy
print (classification_report(test["Label"], predictions))
print (accuracy_score(test["Label"], predictions))
```