APPLYING SENTIMENT ANALYSIS TO STEEL MARKET FORECASTING

ABSTRACT

It's been shown that natural language recognition, particularly sentiment analysis, can aid in the prediction of commodity values, e.g. stock market prices. Here we examine the benefit of sentiment analysis with respect to hot rolled coil (steel) prices in Europe; our experiments show that prediction accuracy can easily be reduced when employing sentiment analysis, but that with some practical feature engineering, a marked improvement results. To facilitate this work, a custom software framework was developed.

1 The author

While prohibited from releasing details by Innocentive policies, the author's company specializes in custom software and deep learning.

2 The software framework

The software was written in the Python language and consists of an API that can be incorporated into other projects, as well as a command line interface that presents a unified whole to the user. The API encompasses stand-alone modules for data retrieval and storage, as well as various statistical model implementations which generated the results seen herein. All data is kept on disk in generic CSV (Comma Separated Values) files after processing, and can be used independently of the software package. This design provides the end user with the option of foregoing the API and model implementations, and using the framework solely as a data aggregation and storage tool.

2.1 Data retrieval and storage module

This stand-alone module includes all the necessary functionality to retrieve and cache data from Twitter and the AGMetalMiner.com site. Where a predefined data source does not exist, the API exposes functions that allow the user to make use of additional and custom data sources: all that is required is a usable hyperlink and a network connection to retrieve the data from said hyperlink (this method is RSS compatible). For additional information on this, please see the "RawData" class in "data.py", or peruse the HTML documentation by opening "index.html" located in the "doc/html" subdirectory.

The predefined data sources are:

1. Twitter posts; note that no access credentials are required, and there is no limit on the number of tweets returned by a search.
2. AGMetalMiner.com article categories (also without the need for credentials or tokens):
   (a) Industry News.
   (b) Steel Price.
   (c) Steel.
   (d) Steel Price Index.
   (e) Global Trade.
   (f) Ferro Alloys.
   (g) Metal Prices.
   (h) Supply Demand.

Twitter was found to yield the greatest benefit to forecasting accuracy.

2.1.1 Data storage format

The software, after downloading data from a given data source (note that each category from agmetalminer.com is considered its own data source), saves any downloaded HTML files on disk. It also saves a CSV format file that consists only of the text data parsed out of the downloaded HTML code, the date of publication, and two additional columns for the results of sentiment analysis ("Label" and "Score"). Sentiment analysis is performed as part of rebuilding the local cache file.

The software will store these files under the "data" directory, which will be created at the location of the "data.py" file. Each data source will be within a unique subdirectory.

2.1.2 (Re)building the data files

The software will download and rebuild the cache whenever it can't locate the necessary cache files it needs to perform a requested function (e.g. the first time the program is run). The command line interface also allows for manually rebuilding the data files at any time, and this should be done prior to predicting future values of the SBB index to ensure that the latest data is available for the prediction model. Note that no effort is made to keep the local cache synchronized with remote data sources, and rebuilding will be necessary from time to time.

2.2 The modeling module

Includes an implementation of a deep fully-connected network, as well as a convolutional network. Both architectures are implemented in Tensorflow. The mean absolute error scores for ARIMA and the convolutional network when making use of the Greed/Fear Index values – as harvested from Twitter – can be seen in table 1. Refer to table 2 to see a side-by-side comparison of ARIMA without exogenous data, as well as its score with a poor Greed/Fear Index time series.

While the contest description stipulates that Pyramid ARIMA be used, an illustration of the improved results when using other statistical models seemed appropriate.

One of the reasons that an alternative model can yield superior results, when compared to an ARIMA variant, is that a deep learning model intrinsically learns the best time lag (which will vary appropriately from sample to sample, no less). This is because the weights which interconnect the model layers (i.e. the neurons of the model) will amplify certain time steps in the input time series, while diminishing or eliminating other time steps. The net result is that the model will focus on the optimal input period automatically.

2.2.1 Sentiment analysis model

The sentiment analysis model is a state-of-the-art implementation, and is a derivation of the famous BERT[1] model with considerably greater efficiency. Please refer to the DistilBERT paper[2] for an in-depth review.

Table 1: Comparison of models when using exogenous data (i.e. Greed/Fear Index)

Model	Mean Absolute Error
Pyramid ARIMA	30.88259
Tensorflow Convolutional	**19.53523**

Scores are on the holdout set of Nov 2019 to Nov 2020; both models were trained using data from the period of Jan 2014 to Oct 2019.

Table 2: The ARIMA model with a poorly designed Greed/Fear Index time series

Model	Mean Absolute Error
Pyramid ARIMA (without exogenous data)	31.26869
Pyramid ARIMA (with bad exogenous data)	33.79030

Scores are on the holdout set of Nov 2019 to Nov 2020; both models were trained using data from the period of Jan 2014 to Oct 2019.

3 Data

Included in the submission .zip file are the processed .csv files which were generated by the software framework described above. These all include a $Score$ column and a $Label$ column, where $Score \in [-1, 1]$ and $Label \in POSITIVE, NEGATIVE$.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[2] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019.