Project Proposal DS 1003 Machine Learning and Computational Statistics

Conor Howells, Shuyu Pan, Xinyi Yu

Problem Statement

Financial news is a valuable source of information that allows researchers and portfolio managers to update their knowledge of stock prices in real-time. Because interpreting news requires processing information, machine learning is an effective method to capture the main information contained in a news piece without the required human capital necessary to read everything. Hence, we propose a project to predict the sentiment score of news headlines using the RavenPack financial news analytics database.

Dataset Description

For this project, we have obtained an academic subscription to RavenPack Equity News Analytics. In this dataset, each news article is stored as a record in the database.

Each record includes information such as the headline of the article, the timestamp of the news, the company it is associated with. RavenPack also includes analytics, such as predictive scores that are derived by RavenPack researchers. Each event is marked by a unique event ID. To note, news observation that regard the same event share an ID.

Our analysis will be focused on RavenPack's 'Dow Jones Edition' data set with equity-related headlines as our independent variables. The sources of news articles for this dataset are: Dow Jones Financial Wires, Wall Street Journal, Barron's and MarketWatch. An archive of observations is available from January 1st, 2000 onwards.

Ravenpack does not provide information regarding the 'emotionally charged words and phrases' it focuses on, but it does give five aggregate indicators that it focuses its sentiment analysis on. Thus, we have a strong starting point to begin our research and add to our baseline algorithms.

Performance Metrics

The set-up of our problem will be to use the headlines to predict the Composite Sentiment Score (CSS) derived by RavenPack. The CSS includes two dimensions: the sentiment direction and sentiment strength. Thus, the score ranges from 0 to 100: 0-49 is negative impact, 50 is neutral and 51-100 is positive impact.

We can frame our problem in two different ways: with binary classification or with regression-related methods.

For **binary classification**, we transfer the CSS in 0-49 to "negative" and 51-100 to "positive", then use this binary class as a prediction target. For binary classes, possible performance evaluations are:

- Confusion Matrix
- AUC (Area under ROC curve)

Another approach is to use **regression-related methods**. We treat the original 0-100 score as real numbers. Possible performance evaluations for regression are:

- Square loss
- Other loss function(L-1 loss)

Under the regression-related method, we believe the square loss will be a better fit for our problem, since a big mis-prediction would cause a severe financial loss, which the square loss penalizes.

Baseline Algorithm and Potential Improvements

We use algorithms in two different states of our project: the feature engineering stage and the predictive modeling stage.

In the feature engineering stage:

1. We use the Bag-of-words assumption as well as Word2Vec to project the original words into a vector space.

2. To extract more features, we could use Term Frequency- Inverse Document Frequency (TF-IDF) which will penalize the frequent terms that appear in the headline.
3. We will analyze industry-standard papers and academic research results, as well as keep close discussion with our advisors, to get more ideas on feature engineering. Our goal is that for each feature we engineer, we will be able to provide a logical explanation for its existence.

In the predictive modeling stage:

1. In binary classification theme where we predict “positive” or “negative” sentiments, possible models would be SVM and decision trees.
2. In the regression theme where we predict the exact 0-100 sentiment score, we could frame it as a time-series regression problem, and use a Long-Short Term Memory neural network (LSTM). LSTM is a type of recurrent neural network with specifically defined blocks to capture the memory of data that occurred in the distant past. LSTM has implementations in various Python deep learning packages which are publicly-accessible.

Project Framework

We propose an iterative approach where we will first cycle through the data over a short period of time, and then repeat this process in more-depth to gain a deeper understanding of the data with the hopes of improving our prediction results.

In each life cycle we will be repeating the following five steps:

1. Data understanding.
2. Data cleaning (e.g. removing stop words, capturing important numbers)
3. Feature generation: First, we will apply classical feature generation methods to our dataset; Second, we will attempt to define topic specific features that are relevant to our focus.
4. Training the model: We will start with a simple classification model and also a simple regression model. In the later cycle we will migrate to non-linear models such as trees, and LSTM.
5. Evaluating prediction results (discussed in question 3 above)

Since this is mostly an NLP problem, most of our time will be spent in steps 2 and 3, namely data cleaning and feature engineering.

Beyond this framework, we will also be attempting to apply different algorithm models to the data set (see part 4). These applications will be incorporated into step 4 of the life cycle. The optimal performance of each model will be compared and discussed in detail for our analysis.