Risk Analysis and Default Prediction for Taiwan Companies

Yuan-Hsi Lai
Dept. of Electrical Engineering
Columbia University
yl4305@columbia.edu

Pei-Ling Tsai
Dept. of Computer Science
Columbia University
pt2534@columbia.edu

Abstract

Nowadays, the world becomes more complicated than before. Many companies that perform well may still default on their commitments to pay the money back. The reason is: the status of a company is not only affected by its financial statements, but also the social events and the confidence its investors have. For example, during the period affected by COVID-19, Boeing's stock price dropped significantly than ever. Therefore, instead of using traditional financial ratios (e.g., Altman Z-score), we are going to use machine learning techniques to do the default prediction. Here we use the Taiwanese companies as a case study. Why we choose them as the dataset is because the financial statements are available on the "Market Observation Post System"[1] and we are more familiar with the social media in Taiwan.

Before training, we collect data from financial statements, news and social media, and company relationships in the industry. After that, we use sentiment analysis to analyze the public opinions, transferring them into concrete scores. Then we put all the numbers into vectors. Finally, we use XGBoost [2], a distributed gradient boosting library, to do the default prediction.

After training, we do the evaluation and get R2 score [3] equals to 0.9, which means that the prediction's direction is highly consistent with the true results. Thus, we use the model to do the default prediction and represent the result on our Webpage [4]

1. Introduction

In this project, we aim at obtaining information from many facets. It is important because the more useful information we get, the more precise the model can achieve. The first question here is: Which data should we get? And which key words should we use for searching?

In financial statements, they contain many numbers and we can find many ratios which use these number to represent different facets. Since we need to do default prediction, we choose ratios that can show a company's cash flow and ability to pay money back. For news and social media, we target at four categories: company names, products, industry and CEO names. With these categories, we can know how people think about them. For example, we can read from news that SSD is more popular than disks these years. Thus, we can expect companies that produce SSD would be more lucrative than companies produce disks. However, this method only works given the presumption that this company is famous enough and we can find it mentioned in many discussions. For those that are not well-known, we need to find another way to evaluate them. Thus, we decided to use the relationship among companies to complement the shortage of public posts. For example, for a laptop firm, if the company that sales memory to it performs well this year, then we can reasonably assume that the laptop firm performs well this year since it means that there is a large demand of laptops.

After having these data, there is another problem: how to transfer the data into concrete

numbers to compare? With this in mind, we adopt the sentiment analysis, which gives us scores regarding how positive or how negative a post is.

2. Dataset

As mentioned in previous section, our data is composed of accounting data, news and PTT [5] posts, and relationship between companies.

2.1. Accounting Data

We used nine financial ratios to represent the ability a company can pay money back. ***Table 1*** below shows these ratios and directions that show if the numbers mean good or not. After counting these ratios, we used JSON format to save company name with its vector into a file (***Figure 1***.). Each JSON file saves all the vectors of one quarter.

Table 1.

Category	Variable	Ratios	Direction (Good)
Firm size	Z1	Total asset value	↑
	Z4	Book-to-market value	↑
Financial leverage	Z5	long-term debts/ total invested capital	↓
	Z7	Total debt/ total capital	↓
Profitability	Z11	Operating income/ received capitals	↑
	Z13	Net income before tax/ received capitals	↑
	Z15	Gross profit margin	↑
	Z17	Earnings per share (EPS)	↑
Liquidity	Z22	Quick ratio	↑

Table 1. Financial ratios in four categories.


```
{
    "台苯": {
        "logz1": 16.146835671487644,
        "logz4": 0.4078992340818276,
        "z5": 0.06178000711539095,
        "z7": 0.2966012308399384,
        "z22": 1.9322107106722126,
        "z11": 0.15255992564827958,
        "z13": 0.13624510194424302,
        "z15": 0.15255992564827958,
        "z17": 0.63
    },
    "首利": {
        "logz1": 14.194495473379014,
        "logz4": -0.47736144087210264,
        "z5": 0.03379461421604965,
        "z7": 0.5653692299284504,
        "z22": 1.1787234935084003,
        "z11": -0.03983189732106638,
        "z13": -0.2423209852504503,
        "z15": -0.03983189732106638,
        "z17": -0.51
    },
    "士電": {
        "logz1": 17.512075726165968,
        "logz4": 1.5357664665443689,
        "z5": 0.6893738283923787,
        "z7": 0.665744559303456,
        "z22": 1.3169747825857039,
        "z11": 0.18635659640399627,
        "z13": 0.10058516220901659,
        "z15": 0.18635659640399627,
        "z17": 0.97
    },
```


Figure 1. Ratios saved in dictionary format in ratios2018Q4.json.

2.2. News and PTT Posts

For news, we obtained from the Google search engine: for social media, we tried to use Twitter. However, we found that only premium member of Twitter can get posts over 7 days. Thus, we decided to change the social media to PTT.

2.2.1 Searching key words

We divided them into four categories as the following, which can reveal the status of one company:

- Company names
- Products
- Industry
- CEO names

2.2.2 Searching period

To align to the duration of financial statements, we use one quarter (3 months) as our searching duration. And we will put the posts of one year (4 quarters) together via the dictionary format in one JSON file. The format is: {"{DURATION}": "……"} as ***Figure 2*** below.

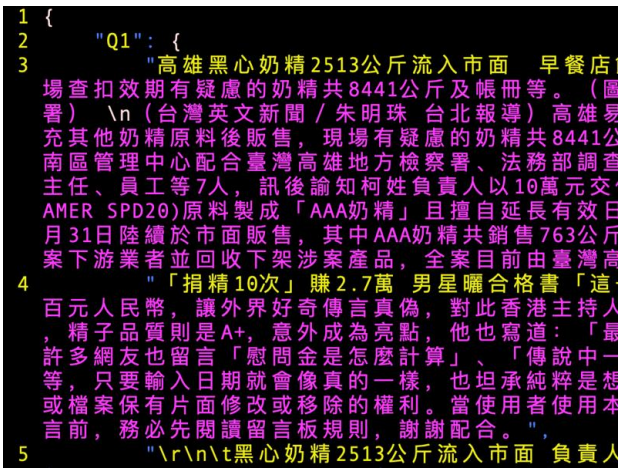

Figure 2. The dictionary format for news posts

2.2.3 Preprocessing for news posts

In news searching, some small companies that are not famous, so the Google search engine would fill in unrelated articles to the search result, especially when the date is far from now. Thus, we used the filter to eliminate those articles. Although it would make the result list to be empty sometimes, it can prevent those unrelated articles from affecting the sentiment score.

2.2.4 Preprocessing for the PTT posts

We managed to crawl posts by keyword and save the result posts by their dates. For each post, we also keep the comments and the like or dislike they have left. ***Figure 2*** shows an example of a PTT post with keyword "三星"(Samsung).

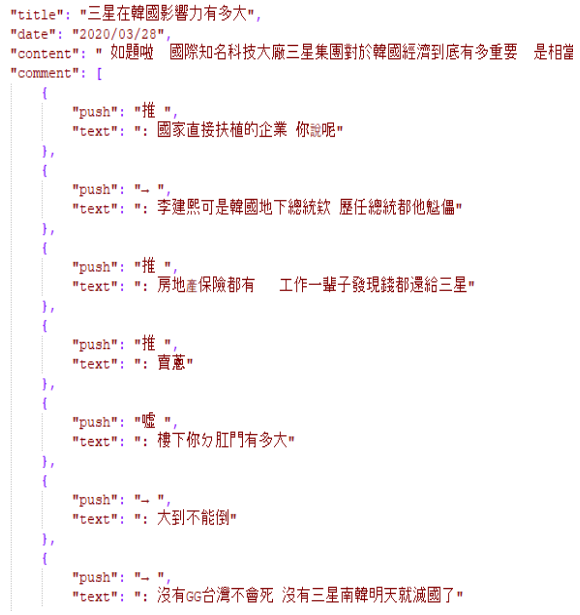

Figure 3. The format saved for PTT posts.

2.2.5 Relationship between Companies

In this part, we used Ardi [6] to build the graph database. We obtained "distances" among companies by referring to the following facets:

- Upstream
- Downstream
- Competitors.

As mentioned before, these three numbers can help us evaluate companies that are not well-known by referring to the scores their "neighbors" get.

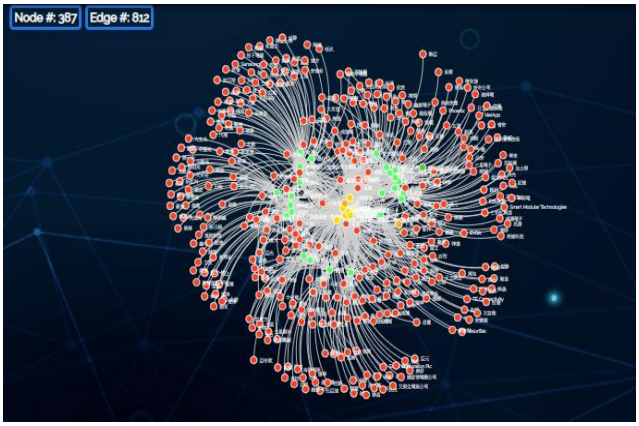

Figure 4. Relationship between companies

3. Methods

In this section, we are going to introduce the data preprocessing, the machine learning model, problems happened during training and the evaluation of the model. ***Figure 5*** below shows the pipeline of the prediction model.

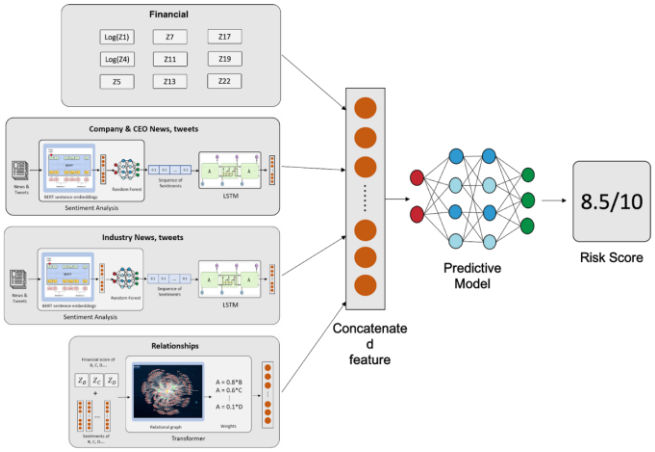

Figure 5. The overview of the prediction model

3.1. Data Preprocessing

For financial ratios, we simply put these nine ratios into the vector. For news and PTT posts, we use sentiment analysis to convert the public opinions into sentiment scores to represent the positive or negative reaction toward companies. For relationships in graph database, we find the closest upstream, downstream, and competitor to the company. We than utilize the three related company's financial score as a relation vector.

3.1.1 Sentiment Analysis

3.1.1.1 SnowNLP

It is a Python package that used to get the sentiment score of one sentence. First, it needs to cut a sentence into pieces (words or phrases). Then it would use the "part-of-speech" concept, giving each piece a "tag" (e.g., noun, verb, adverb, … etc). After that, it counts the probability of the piece given the tag. Later, it uses the Naïve Bayes classifier to count the overall probability of the sentence. In the end, it will know the extent of positive or negative of the sentence.

3.1.1.2 Jieba

It is a Python package that is used for Chinese text segmentation. The can cut a sentence into words or phrases. In this model, we used Jieba to cut sentences before the sentiment analysis

With the help of SnowNLP, we got 81.8 % accuracy. The result is not bad; however, we found it failed to resolve the semantic and polysemy problem in Chinese. Thus, we used BERT instead.

3.1.1.3 Bidirectional Encoder Representations from Transformers (BERT)

BERT [8] is a language model proposed by Google. It is trained using Wikipedia data and has pre-trained model for multiple language. The original BERT outputs a 512-dimension vector for each word. Therefore, for a sentence or an article, this feature size might be too huge. Therefore, we found a package called BERT sentence transformer, which solved this problem. It used word embedding in the begin, then it did the pooling regarding the results. BERT sentence embedding only allow us to get a reasonable representation of each sentence. In order to utilize this into the sentiment analysis, we still need to train a model that performs classification of these embeddings into positive and negative. We therefore trained a random forest [7] model using the dataset provided from SnowNLP and got 92.4% accuracy in the end.

3.1.2 The relation feature from graph database

We used the Ardi platform for our graph database. Our original thought was that for a target company, which we want to get the relation feature of, we separate the feature into three parts: competitor, upstream, and downstream. For each feature, we find the companies with certain distance, which could be either pure BFS distance,

or some metrics such as number of common neighbors. However, since the relation we parsed from internet is not as explicit that we can get the exact upstream and downstream companies of each company, at the end we just used the BFS distance of each company and aggregate all the companies' financial data that has distance to 2.

3.2. Model

During the training, we first chose the LSTM model, since it takes the timestep information into consideration. However, it suffered an issue, so we adopted XGBoost in the end. We will have more discussions in the following part.

3.2.1 Long short-term memory (LSTM)

Long short-term memory (LSTM) is an artificial recurrent neural network (RNN) architecture used in the field of deep learning. It is developed to deal with the exploding and vanishing gradient problems that can be encountered when training traditional RNNs.

LSTM networks are well-suited to classifying, processing and making predictions based on time series data, since there can be lags of unknown duration between important events in a time series. LSTMs were developed to deal with the exploding and vanishing gradient problems that can be encountered when training traditional RNNs.

In this project, we used ["Q1", "Q2"] together for training data and ["Q3", "Q4"] as testing data. Since our labels are not binary or classes; instead, they are numbers that are proportion to the numbers in the vector, we used ReLu [8] as the activation function. We got accuracy of 94% in evaluation. However, we found that LSTM is more suitable to data with many timesteps. For example, the trend of the everyday stock price. In our case, we only have 1 timestep (2020Q1) in our data for prediction because we use quarter as the unit of one timestep. Thus, we changed to XGBoost model.

3.2.2 XGBoost

XGBoost model evolves from decision tree. It is a decision-tree-based ensemble machine learning algorithm that uses a gradient boosting framework. It is a scalable and accurate implementation of gradient boosting machines and it has proven to push the limits of computing power for boosted trees algorithms, as it was built and developed for the sole purpose of model performance and computational speed.

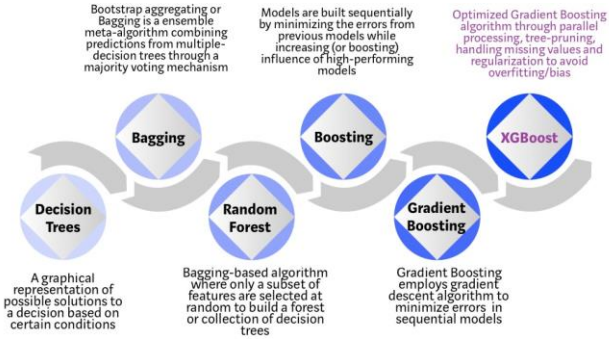

Figure 6: Evolution of XGBoost Algorithms

4. Experiment

In this section, we will demonstrate how we train and evaluate the XGBoost model.

4.1. Prepare Training Data and Labels

In order to train our model using machine learning methods, we will need labels of each piece of data. As we showed in figure 5, we combined the result of sentiment analysis on social media posts, the relation factors, and financial ratios into a vector that is the training data of our model. For the label, we originally wanted to get risk scores predicted by credit scoring companies, however for Taiwanese companies, the score data needs to be paid. Thus, we decided to use an approximation as the label by combining stock prices, debt-to-equity ratios using the equation below to map the scores into 0-10 scale, which is our desired risk score that 0 being no risk and 10 being high risk. The equation is shown below.

- Stock price movement
  - Maximum increase: 150%
  - Maximum decrease: 66%
- Debt/Equity Ratio
  - Maximum: 4.956
  - Minimum: 0.032
- Estimated risk score:
  Debt/Equity + 3 + -2 * (stock movement)

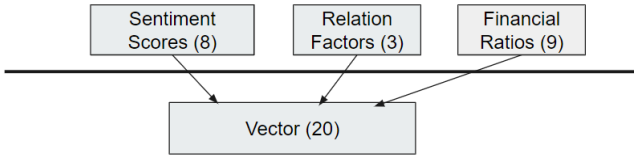

Figure 7: The training data of our model

4.2. Training

For the XGBoost model, the settings are as the following:

- training data: 0.67 of ["Q1", "Q2", "Q3"]
- testing data: 0.33 of ["Q1", "Q2", "Q3"]
- regression function: linear
- learning rate: 0.6

During the training, we found that some companies didn't have financial ratios. The reason is: these companies register in other countries or do not register in the open market.

The workaround we used is: Fill in the average of each factor (the average of values in one column) and time a penalty factor of 1.2. We put the penalty here is because we want to convey the idea that we want investors to be more conservative to those companies which don't provide sufficient information.

4.3. Evaluation

We used the training model to predict our testing data. After that, we use R2 score to judge the performance. The score we got is 0.9. This means that the prediction is highly align to labels.

4.4. Prediction

We use the same model to predict ["Q4"] data. ***Figure 8.*** shows the comparison between the change rate of stock prices from Q3 to Q4 and our predictions. In this figure, we can find that the shape is basically the same. Thus, we can say the result is reasonable.

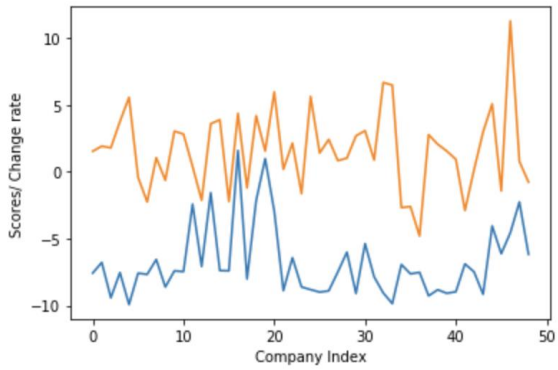

Figure 8. The comparison between the change rate of the stock price and the prediction

5. System Overview

The backend of our risk score system is shown in figure 5. We also designed a webpage using D3js for showing the analysis result and several features. First is the risk score. As shown in the figure below, the risk score has colors that shows the degree of risk and there's also description below.

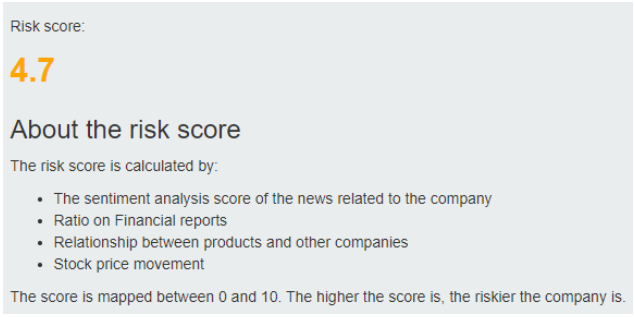

Figure 9. Risk score on webpage

The second part is the sentiment analysis. This graph clearly shows that how a company's social media posts has been. There are 8 sections being the four sections we mentioned above and each

with Google News and PTT data. The length is the number of posts in that section.

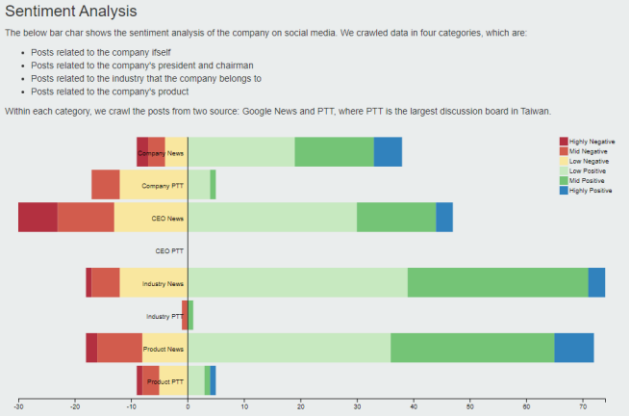

Figure 10. Sentiment Analysis on webpage

The third part is the relation graph of that company. We queried the egonet of that company from Ardi and showed it on the webpage. This gives a view of relations between the company and its competitors, products.

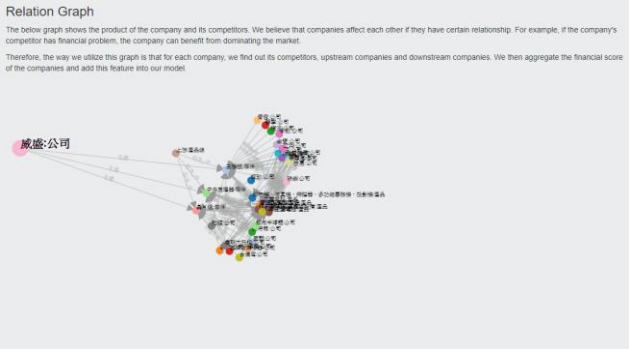

Figure 11. Relation graph on webpage

We also showed the stock price of the company since that we utilized the stock price as label calculation.

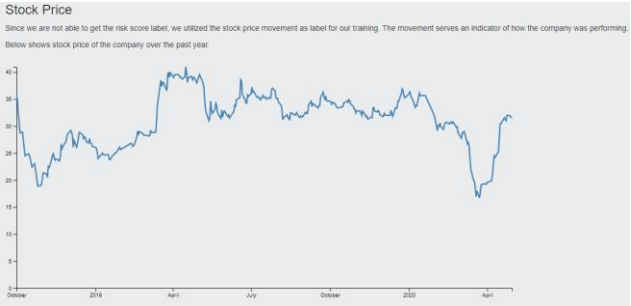

Figure 12. Stock price on webpage

The last part is the Financial data. We show the 9 ratios that we used to perform prediction and the length of the bar chart is the ranking among the companies that we have financial data. The higher the bar is, the better the company is performing in that section.

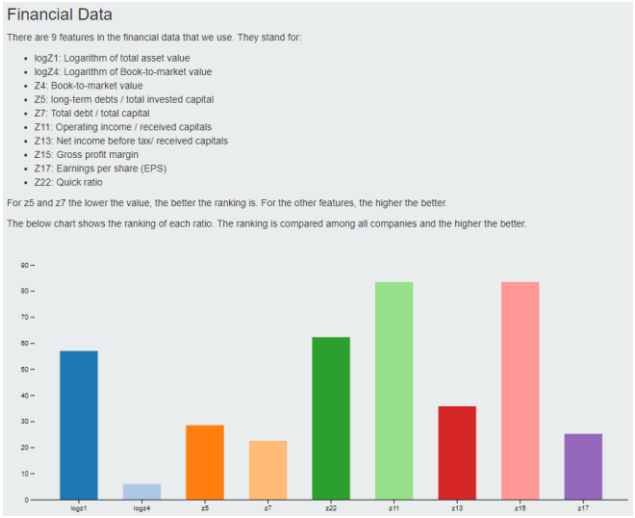

Figure 13. Financial data on webpage

6. Conclusion

In this project, we tried to design a model that can react to the change of the real world. We used the financial ratios as our foundation, adding factors of sentiment scores and "distances" among companies. Though the idea is reasonable, it's hard to collect sufficient data since not all the companies are well-known or reveal their financial statements. The only way we can do is predict those companies without sufficient data more conservatively. The R2 score we got shows that the overall prediction is align to the label. Having the result in hand, we put the predicted score with the everyday stock price, news posts and graph database together on the Webpage. Investors can use the information on the Webpage to review if the predicted score is reasonable.

References

[1] Market Observation Post System: https://emops.twse.com.tw/server-java/t58query

[2] XGBoost official website: https://xgboost.readthedocs.io/en/latest/

[3] R2 score WIKI: https://en.wikipedia.org/wiki/Coefficient_of_determination

[4] Our project Webpage: http://146.148.63.155:5001/index

[5] PTT: https://en.wikipedia.org/wiki/PTT_Bulletin_Board_System

[6] Ardi: https://www.graphen.ai

[7] Random forest: https://en.wikipedia.org/wiki/Random_forest

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” arXiv preprint rXiv:1810.04805, 2018.