Yusiqi Wang

Professor Diego Klabjan

Data Science and Analytics

March 6, 2020

# Executive Summary

**Process Taken:**

This study aims to extract all company names, CEO names and percentages from daily business news for 2013 and 2014.

All articles are read and broke into list and dataframe of sentences. Word2Vec is used to vectorize words, in order to transfer text to numerical feature. Regular Expression is used to find continuous capitalized words in business news, which are potential CEO and Company names. POS Tag is used to distinguish different parts of speech. Then, positive data set and negative data set are defined. Finally, the Naïve Bayes classifier is tuned on the training data set and make prediction.

Percentage outcome is extracted by Regular Expression from the articles and verified by the Classifier.

**Performance of the classifier:**

Accuracy for Company Classifier: 0.8875

Accuracy for CEO Classifier: 0.8680

**Report for Text Analysis**

# Data Preprocessing

The first step is data preprocessing.

Labeled data of CEO, company and percentage was processed from CSV and put into list.

All articles in TXT files were read. Each article was tokenized into sentences. Punctuations were removed and error data was ignored. Since it is inefficient to process and analyze more than 700 documents separately, all of the sentences were stored in just one list and dataframe.

# Extract Percentage

Because the format of percentage is quite fixed and all of the number are similar, it is not necessary to imply supervised learning model for extracting percentage. A number of **Regular Expression** are enough to match nearly all of the percentage from the articles.

Here are some **typical percentages formats**:

“1%”, “one%”, “1 percent”, “one percent”, “1 percentage points”, “one percentage points”

**Regex** used as follows:

图片包含 屏幕截图

描述已自动生成

All of the percentages in these formats are extracted and saved in CSV.

# Extract Company & CEO\_Part 1

* + **Word2vec**

In order to extract features for company name and CEO name, Word2vec is imported here.

**Word2vec** vectorizes words, and by doing so it makes natural language read by the computer. Parameter “word vector dimensionality” is set as 100 in this assignment, thus each word in this text is represented by a unique array which contains 100 float numbers. The array stands for the position of a word in multidimensional space, namely **each of the 100 numbers could be treated as a feature of certain word**.

More features make the classifier more precise. The **reason** I’m using Word2Vec here is that Word2Vec could provide 100 features, which would be more precise than several features extracted by the rules like “if the article contains a keyword like CEO.”

But there is a drawback when a neural word embedding represents a word with numbers. Although it is in a simple way, it is unlikely to be translated, (thus it is hard to give a list of features used.)

All words in txt file “all news” are converted to vector and used to train the Word2vec model.

图片包含 屏幕截图

描述已自动生成

* + **Numerical feature embedding**

Company and CEO name in CSV is converted to vector. Since I use data in TXT to train the Word2vec model, a problem emerges. If there is a word in CSV but not in TXT, an exception will be reported to terminate the loop. Then a solution is to use “try: except”. Ignore it if an error is reported, for the error proves that these names did not appear in the article.

图片包含 屏幕截图

描述已自动生成

Regular Expression is used here to find **potential Company and CEO** name in articles. I suppose continuous capitalized vocabulary could be Company or CEO name. Vocabulary less than 4 words are taken into consideration. Then the potential words are vectorized. Further step is going to be taken later in classification model part.

图片包含 屏幕截图

描述已自动生成

After dealing with the CEO and company data in this way, we have already had data of **positive label.** The next step is to find data of **negative label**. If both positive label and negative label are found, the classification model could be trained with these data.

* + **Find negative labels**

Regular Expression is used here to remove punctuations and find capitalized words. **Pos Tag** is used here to find **Adjective, Verb and Determiner**, which are most unlikely to become Company or CEO word, then the capitalized Adjective, Verb and Determine is treat as part of negative label. Only taking 8201 lines instead of the whole dataset is to balance positive and negative data set.

* + **Positive and Negative data for Company**

Label data from the given Company CSV can be taken as **positive data**.

When looking for company name, all data except company name can be taken as **negative data**. Thus, capitalized adjective and CEO name could be taken as negative labeled data.

Assign “1” to positive data. Assign “0” to negative data. Since there is a large amount of data, shuffle data and split data for 50% train data and 50% test data.

* + **Positive and Negative data for CEO**

Similarly, label data from the given CEO CSV can be taken as **positive data**.

Company name could be taken as **negative data**.

Assign “1” to positive data. Assign “0” to negative data. Shuffle data and split data for 70% train data and 30% test data.

# Extract Company & CEO\_Part 2

* + **Native Bayes Classifier**

Naïve Byes classifier has proven to not only be simple but also **fast, accurate, and reliable**. It has been successfully used for many purposes, but it **works particularly well with natural language processing problems**. When Naive Bayes predict the label of a text, it calculates the probability of each label for a given text and then output the label with the highest one. Thus, it is appropriate to choose Native Bayes as classification model here.

* + **Predict Company Data & Predict CEO Data**

Tune the classifier on the training data set and make prediction.

**Performance of the classifier:**

Accuracy for Company Classifier: 0.8875

Accuracy for CEO Classifier: 0.8680

# Additional Information

In the process of doing the assignment, I have tried different methods. Here are some additional information I didn’t use in the final version source code.

**Random Forest Classifier:** At first, I selected the Random Forest classifier since it can handle data with high dimensions (many features.) Although the accuracy is higher than the final one I got by Naïve Bayes, the actual data extracted is not as accurate as Bayes did.

**Using Bayes Classifier to verify percentages outcome:** After extracting percentages by Regular Expression, I also tried to use Bayes Classifier to do verification. Percentage labeled data is defined as positive value, CEO and company csv labeled data as negative value. “Match\_percent” is percentage outcome I got by using Regular Expression. I use classifier to verify “match\_percent”, and got negative value for 1,876 out of 78,953, which shows the good performance of the classifier.