Matthew Shipley

IEMS 308

Professor Klabjan

Homework Assignment 3

# Text Mining and Analytics

## General Preprocessing

To scrape the articles for CEO names, company names, and percentage data, the numpy, pandas, csv, sklearn, nltk, and re (regular expression) packages were used. The articles were loaded into memory using the myfile.read() function, and each article was a string entry in a list entitled either *texts2013* or *texts2014* for the 2013 and 2014 articles respectively.

The training datasets were first cleaned up by removing any duplicates in the lists of CEOs, companies, and percentage points and removing single names from the list of CEOs. From the company training dataset, a list of the 50 most common company suffixes (i.e. Group, Ltd, Inc, Corp, etc.) were extracted by counting through the list of unique companies. These four training datasets were imported from Excel *.csv* into pandas dataframes.

## CEO Names

To trim the corpus into a more manageable size, both *texts2013* and *texts2014* were first tokenized by sentence. Then, each sentence was parsed for the regular expression r*"[A-Z][a-z]* [A-Z][a-z]*"*, which only selects sentences where the "Firstname Lastname" pattern is found. Each sentence with this pattern was tokenized into words, then the nltk package was used to remove stop words and stem. This created a list of lists *firstLastSent*, in which each entry is a sentence consisting of a list of words used in that sentence. From this list of lists, a training sample entitled *trainingSample* was created by selecting all sentences in which a real CEO name (from the CEO name training set) was found, or a random 10% selection of sentences in which a CEO name was not found. To collect data, the following variables were used:

*Token* – string value of the current token
*Coupling* – string value of the current token + the next token (i.e. *Firstname Lastname*)
*AlphaNum* – is the coupling alphanumeric?
*PrevCap* – is the word before the coupling capitalized?
*Title* – is the coupling a "title" (i.e. *Firstname Lastname* instead of *firstname lastname*)
*NextCap* – is the word after the coupling capitalized?
*Position* – index of the Token's position in the sentence
*FirstLength* – length of the first word in the coupling
*SecondLength* – length of the second word in the coupling
*FirstNoun* – is the first word in the coupling a noun or a proper noun?
*SecondNoun* – is the second word in the coupling a noun or a proper noun?
*CEOinSent* – is either the string "CEO" or the string "Chief Executive Officer" in the sentence?

*IsCEO* – is this coupling actually a CEO, found by checking if the coupling is in the CEO training dataset

Each of these values was found for each word in the sentence for the training set, and stored in a list. The training dataset was compiled into a numpy array *train*, and a decision tree classifier was constructed using *PrevCap*, *Title, Position, FirstLength, SecondLength, NextCap, FirstNoun, SecondNoun,* and *CEOinSent* and using *IsCEO* as the training labels. This same information was then collected over the entire *firstLastSent* dataset in order to extract the overall list of CEOs.

**Percentages**

A similar process was taken to preprocess the data for extracting percentages, except the regular expression searches *r”\%”* and *r”percent”* were used to select sentences that had either the percent symbol or the word percent. A training sample was created by taking a random selection of 20% of sentences that matched this regular expression search. To collect data, the following variables were used:

*Token* – the token currently being processed
*Coupling* – this token + the next token
*FirstNoun* – is the first token a noun
*FirstAdj* – is the first token an adjective
*FirstCD* – is the first token a cardinal number?
*SecondNoun* – is the second token a noun?
*SecondPerc* – is the second token the string “percent”?
*SecondPercSymb* – is the second token the percent symbol “%”?
*IsPercent* – is this coupling actually a percentage, found by checking the percentage training dataset

This data was collected for each word in the training corpus, and the training dataset *ptrain* was compiled using the *FirstNoun, FirstAdj, FirstCD, SecondNoun, SecondPerc, SecondPercSymb* features. This training dataset and the *IsPercent* training labels were entered into a decision tree classifier algorithm and the model was constructed. The same data was collected over the entire *percentSent* corpus (a list of tokenized sentences that contain the string “percent” or the percentage symbol “%”), and the decision tree model was used to classify each *Coupling* as either a percentage or not a percentage.

**Company Names**

Again, the total corpus was trimmed down to include only sentences that contained the most common company suffixes. This was done using the following lines of code:

```
suffixes = re.compile('| '.join(companySuffixes['Suffixes']))
m = re.search(suffixes, thisSentence)
```

Where *suffixes* was the regular expression search term, containing each suffix separated by “|”. The training dataset was again selected by taking a random selection of 20% of the sentences that matched this regular expression search. For each word in the training dataset, the following variables were extracted:

*Token* – the token currently being processed
*Grouping* – the grouping that corresponds to a potential company name. For each token, a bigram, trigram, and four-gram were processed and evaluated as a potential name
*Title* – is this grouping a title? (i.e. all words capitalized)
*EndSuffix* – is the last word in this grouping one of the most common company suffixes?
*PrevCap* – is the word before this grouping capitalized?
*IsAlpha* – is this grouping alphanumeric?
*TitleSubset* – is this grouping a subset of another potential company name that is also a title?
*IsCompany* – is this grouping actually a company, found by checking if the grouping is in the training set

The training dataset was created using the *Title, EndSuffix, PrevCap, IsAlpha, and TitleSubset* features and the *IsCompany* array for training labels. A Bernoulli naïve Bayes classifier was used to create the classification model on the training dataset as each of the features are Booleans variables. This data was then collected on the entire corpus of sentences that matched common company suffixes, and the test dataset was run through the Bayes classifier to output the list of company names.

**Model Evaluation and Improvement**

*CEO Names:*

```
30889 false positives
8838 false negatives
9178 true positives
1251935 true negatives
accuracy is 0.9694605024445743
precision is 0.229066313924177
recall is 0.5094360568383659
```

*Percentages:*

```
23541 false positives
625 false negatives
51636 true positives
916098 true negatives
accuracy is 0.9756366569210606
precision is 0.6868590127299573
recall is 0.9880407952392798
```

*Company Names:*

```
34914 false positives
12072 false negatives
5305 true positives
3952951 true negatives
accuracy is 0.9882688736410934
precision is 0.1319028319948283
recall is 0.30528859987339585
```

All three classification models achieved very high accuracy. Because of the nature of text analytics when trying to identify specific entities, most candidates are negatives. This increases the amount of true negatives classified. Precision was quite high at 67% for percentages, but low for both CEO and company names at 23% and 13% respectively. Many of the features used were related to parts of speech and capitalization, so this led to many candidates being labeled as false positives, such as names that are not CEO names or picking up company suffixes in places that are not within a company name (such as the word "Group" which is used very often outside of company names). Furthermore, because only bigrams were analyzed in the CEO name classifier, CEO names that are written with a middle initially may have been classified as negatives. For CEO names, the classifier may have missed company names that are commonly written with no company suffix (Ltd, Corp, Inc, etc.) if they are written independently in the news articles.

To improve the classifier, several steps could be taken. By tokenizing into sentences, none of the classifiers used information gained from elsewhere in the article, and only used information from that sentence. For example, if an article is about a certain company and a name is stated within the article, it may be more likely that the name is the CEO of that company. In the company name classifier, it may be beneficial to include as a feature the number of times that grouping of words is repeated, as an article about say, the Boston Consulting Group, may repeat "Boston Consulting Group" several times. Another way to improve the classifier might be to include more features pertaining to parts-of-speech in both the CEO and company classifier, as the placement of a text grouping within the context of various parts of speech might change the probability that a certain grouping is or is not a CEO name or company name.