Text Mining Assignment

**Part 1**

In part 1 I was expected to extract all the names of companies in the corpora.

*Preprocessing*

First the format of the corpora was as converted from .csv to .ascii in order to clean the data from unreadable characters. Then, all the text documents were read in and merged as a single corpora. The text was tokenized by sentences first and then further tokenized into words. Each of these sentences and words were then saved into a list. Next stop words using a dictionary were removed from the lists in order to reduce the size and consequently the speech tagger package was applied to label very word in the corpora.

*Features:*

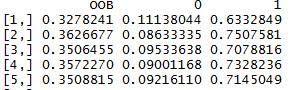
Here several features were designed to determine the characteristics of the sentence in order to determine the likelihood of a sentence containing the name of a company. The value of a feature was calculated twice each time, for the sentence with company names and once for a sentence with a non company proper noun.

Features and rationale:

1. Check for words such as ltd, inc, corp, company etc. in the sentence and classify sentence as 1. It was observed that these words will often accompany the name of a company in that same sentence.
2. Check for performance words such as sales , revenue, profit and business terms such as stakeholder etc. Again these words refer to a company and hence it is likely that a company is named and assign a value of 1
3. Using Regular expressions look for more than 10 capitalized words in a sentence. Again this often means there is high likelihood that a company’s full form is being named since each word will have a capital letter.
4. Check for 3 proper nouns in a sentence side by side. This is often a big indication of a company being named and assign 1.

Data analysis:

Next random forests were used on R to fit a classifier on the training data. Random forests regression performs several iterations on a set to give a model based on the training data. The response variable is the binary 1,0 if the sentence has the company name and 0 if not. The model performance as shown on r is below:



The false negative rate rate is quite high at 0.6 and false positive was low around 0.1 suggesting a good fit, The OOB errors seem quite large here and may be a cause of concern.

Now using python we calculate the feature values for all the sentences and create a features matrix that contains the values of each feature for all sentences. The resulting response of company names present of not as 1 and 0 are then calculated from the features. Then the sentences with 1 are saved in a list for further analysis.

Finally analyzing the list of sentences labeled as 1, we use regular expressions as follows to extract capitalized words not at the beginning of the sentence

List2.append(re.findall('([A-Z][\w-]\*(?:\s+[A-Z][\w-]\*)+)', i))

*Analysis of results and improvements*

Looking at the results we see the effect of having a very high false positive rate as many non company names are included, especially proper nouns. Firstly, it was noted than searching for 10 capitalized words as a feature was trivial and perhaps this number should be analyzed further. Secondly, we might need to pay more attention to the performance and business terms since in many cases a word like profit was not accompanied by a company name. Lastly more analysis should be done on checking the type of proper noun used. This will help distinguish companies from other nouns like countries etc.

**Part 2**

In part 2, I was required to extract all the names of CEO’s from the collection of texts as the corpora. A very similar approach to task 1 was used.

*Preprocessing:*

Continuing to use the processed corpora from task 1, the training set data on CEO was analyzed to select the sentences where CEO’s were present. These sentences were saved in a list. Applying the tagged part of speech words, 500 proper nouns were randomly selected from the corpus. This was to create a list of sentences that appear to have a name of a CEO in them but actually do not. The proper nouns in each sentence was matched to sentences in the corpora. Now sentences from the sample containing CEO names were labeled as 1 while those containing non CEO proper nouns were labelled 0. This was the training set and was used to train the model and classify the results.

*Features:*

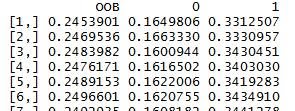
In order to determine how likely that each sentence saved in the previous part would contain the name of a CEO, several features were designed to determine the characteristics of the sentence. The feature values were calculated twice each time: one for the sentences with known CEO names and the other with the non-CEO proper nouns. This allows to then run a classifier model on the entire corpora using the values of the features. Note that this is the same method as in part 1.

Features and rationale:

1. Check if CEO, ceo, Chief Executive, CTO, CFO etc in sentence and if yes assign 1, 0 otherwise. The logic behind having this feature was that it was observed that such words were good precursors to having the name of a CEO in the sentence.
2. Check using regular expressions that there are more than 7 capitalized words in the sentence and assign 1. This is because it is likely that a CEO is named when there are many letter capitalizations including their standing “CEO” for example.
3. Check whether head, department, chair are contained in a sentence and give a value of 1. This words are often good indicators of a CEO name being present in the same sentence since they are used together.

*Data analysis*

Random forests were used in a similar way as before.



This suggests that there seems to be high accuracy as shown by a small OOB error and there appears to be a relatively high true classification rate (false negative of 0.33125). However, the false positive rate is 0.164 suggesting there are few instances where non CEO names were considered CEO’s.

Python was used to run the features to fill the response variables like before.

Now I parse through this list of selected sentences using RegEx as follows:

List2.append(re.findall('([A-Z][\w-]\*(?:\s+[A-Z][\w-]\*)+)', i))

Then spaces were removed:

List2 = [[x for x in list2 if x != []]

The resulting list was then exported to excel to remove all duplicates.

*Analysis of results and improvements*

Since the false positive misclassification rate was quite high, there were many non CEO names that were posted. About 70% of the CEO names supplied were included in the training set which suggests that most were picked by the program. However since only a random sample was taken for the features analysis, this could be a potential source of not suing more of the actual CEO names. The non CEO names analysis suggest that 80% were classified correctly.

Several improvements could be made in the future. Firstly, extracting capitalized words from sentences was not an ideal way since non CEO names can also be capitalized. Secondly, CEO names not always had the first and last name meaning it was often not necessary to extract two consecutive words from a sentence. Lastly the random forest algorithm was a compromise I had to make in order to save computing time. Better algorithms for fitting should be explored next time.

**Part 3**

In part 3 I was expected to extract all the percentage symbols with their associated value. Here a relatively simple approach was taken focusing on the use of regular expressions. Using r, the following expression was used

percent <- regexpr("([0-9]\*.?[0-9]\*%)|([0-9]+.?[0-9]\*( percent))|(((third)|(quarter)|(half)|(one)|(two)|(three)|(four)|(five)|(six)|(seven)|(eight)|(nine)|(ten))( percent))|(((third)|(quarter)|(half)|(one)|(two)|(three)|(four)|(five)|(six)|(seven)|(eight)|(nine)|(ten))(%))", textdm$dimnames$Terms)

The model performed well as can be seen in the results except for numbers that were tagged as strings. No false positives returned suggesting the model was able to classify well. To improve this model in the future we must amend the regular expression to also take into account strings that show up before or after a percentage symbol.