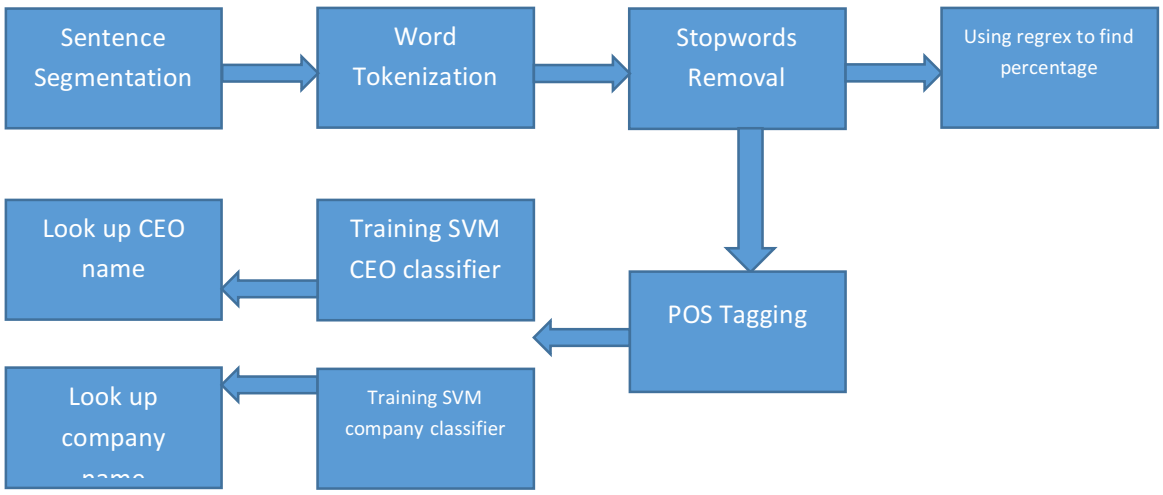
Zhaocheng yu netid zyr531

Target: extracting CEO names , company names and percentages.

Pipeline :



1.Preprocess

1. Sentence Segmentation:
   In this step, I just add all raw materials up into one file and then use Python Nltk package to finish the sentence segmentation. All relevant codes are in the sentence_seg.py.
2. Word Tokenization & Stopwords Removal:
   Strategy adopted here is to tokenize words by space and set up a set of stopword with the help of nltk.corpus. After this step, all words are tokenized and stopwords are removed.
3. Extracting percentage:
   Since the percentages in the raw materials can be easily defined by regular expression. All details could be find in percentage.py. And the pattern about how to extract percentage could also be found there. Output of percentage could be find in output_of_percent.txt.
4. POS tagging:
   In order to train the binary classifier to pick out all words which might be candidates of CEO names and companies' name. I have to finish the POS tagging first. Because all names are identified by tag = NNP. Nltk packages are used here to do the tagging job and we get a relatively nice result.
5. Training SVM classifier:
   The reason why I choose SVM over Logistic regression is that I am not sure weather we can used a linear hyper plane to distinguish candidates from non-candidates. Considering the relatively slow convergence speed of Neural Network, SVM is the ideal supervised learning algorithm .

2. Attribution of binary classifier.

a). I used 1/3 observations of raw data set as training set. It is not hard to find out patterns in ceo.csv and companies.csv. All CEO first names start with a uppercase letter and finished with a lowercase letter. The attributions of First Upper Letter and First Lower Letter are reasonable to build. Also, the family name of CEO also follow the pattern. And hence Second Upper Letter and Second Lower Letter are also chosen as attributes. Also the length of name is also an important attribute. And POS attribute equals one if POS == NNP and 0 otherwise. I code every bigram up as a single observation, if the first word start with uppercase letter , column FirstUpperLetter is 1 , otherwise 0. Column SecondUpperLetter , FirstLowLetter,FirstUpperLetter follow same rules. Count the length of bigram as an another attribute. With respect to label of the training set. I set up a dictionary of CEO name according to ceo.csv. Observation which has same name in that dictionary will be labels as 1. Otherwise 0.

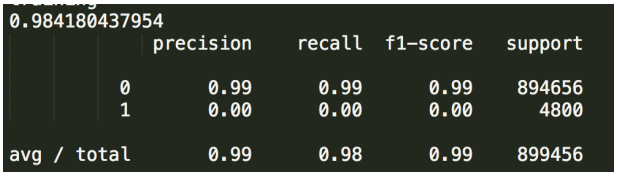
b). Similar rules of Company name classifier are also adopted. But since there might be three of four words, I added ThirdUpper Letter and FourthUpperLetter.

3. Dictionary trick.

Well, I review the results coming from the classifier. Unfortunately, my two classifiers failed to distinguish CEO name from Company name. Both classifiers just wrap them up. Therefore, I came up with an idea to solve this problem. I set up two dictionaries , one is about CEO name and another one is about company. And every time when I examine results from classifiers, I look up in those dictionaries. If it is the key of CEO dictionary, I assigned this observation to CEO names, otherwise companies' names. And the results are really good, you can check it at output_of_ceo.txt and output_of_company.txt.

4. Conclusion

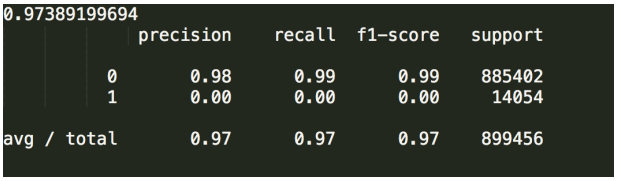
Although two classifiers share relatively good accuracy. But they still failed to distinguish CEO names from Company names, which means I failed to find attributions which can tell me big difference between those two different kinds of names. However, I still get a fairly food result by looking up a dictionary. I think I should spend more time in finding good attribution. It is very important when it comes to data science. The following figures show the results



```
0.984180437954
             precision    recall  f1-score   support

          0       0.99      0.99      0.99    894656
          1       0.00      0.00      0.00      4800

avg / total       0.99      0.98      0.99    899456
```



```
0.97389199694
             precision    recall  f1-score   support

          0       0.98      0.99      0.99    885402
          1       0.00      0.00      0.00     14054

avg / total       0.97      0.97      0.97    899456
```