Andre Schweizer
Prof. Diego Klabjan
IEMS 308 – Data Science & Analytics
February 27th, 2018

HOMEWORK ASSIGNMENT III

Text Analytics

The analysis procedure for this assignment, described in the sections below, aimed at extracting three different types of information from the corpus provided: company names, CEO names, and percentages. Before starting on any particular category, I first split the text contained in the corpus it into sentence using the traditional method (i.e. split on ".", "?", "!"), and stored those strings as members of a list.

I | COMPANY NAMES

1) *Preprocess*
I looped through the initial list of sentences using the *pos_tag* function in the *re* library, and only kept those that contained words tagged as proper nouns (NNP or NNPS).

2) *Create list of features and corresponding regular expressions*
Looking at provided company names, I was able to observe common characteristics that differentiated them from other words, and used that to compile regular expressions to catch features that pertained to the patterns of characters within the words. The complete list of features and regular expressions can be found in *Appendix A*.

3) *Create train & test data, and negative samples*
I first split the provided training labeled data into training and test – 75% as training and 25% as test. I then split the CEO names training data in the same proportions and appended them to the appropriate set (i.e. 75% to training and 25% to test) to be used as negative samples. The rationale behind this is the fact that companies are very often named after people, and, if not, their names frequently contain some of the same features as CEO names. Hence, CEO names are quite similar to those of companies in terms of orthographical features, and are thus effective negative samples.

4) *Analyze train and test data*
I filled out feature frames for each of the data sets (e.g. if the sample contained "LLC", a 1 was inserted under the appropriate column for the corresponding row) that were later used by the binary classifier.

5) *Extract and analyze data from corpus*
To find the pieces of the original corpus that had the greatest probability of being company names, I made a list of all the proper nouns found in the sentences selected in step (1). I also applied *re.search* the regular expressions described in (2) to each of the sentences to catch possible company names that didn't show up as proper nouns (saved as *possiblecomp*), like bigrams. I then repeated step (4), this time with the corpus data.

6) *Use classifier to make predictions and obtain results*
I fitted a logistic regression object on the training data. I then used that to make predictions for the test and corpus data sets. I used the corpus predictions to filter through the list of candidate samples, and only keep the ones that were actually predicted to be company names. I then further processed it to remove names that occurred more than once and obtain a final list of company names, available in *SchweizerAndre_CompanyNames.csv*.

7) *Model performance*
The predictions made for the test data set, in which samples were known to be company names or not, yielded a true positive rate of 57% and a false positive rate of 2%. This reveals that the NER classifier is maybe a little too rigid at the token level: it is pretty good at not classifying negative instances as positive ones at the price of underestimating the number of actual company name samples. Increasing the flexibility of the model would yield a higher true positive rate (i.e. more company names would be found), but more negative samples and features would likely be needed to ensure that the false positive rate doesn't increase significantly. The ROC plot (see *Appendix* A) shows that this would only be relatively effective for an increase in true positive rates of up to around 80%. After that, the true vs. false positive rate trade-off becomes quite shallow (i.e. a slight increase in true positive rates require a significant increase in false positive rates), meaning that the benefit of acquiring extra accuracy would likely not be worth it. Furthermore, a closer look at the company names obtained from the corpus quickly shows that the false positive rate on the corpus data was much higher than 2%, and reveals a disparity between the performance of the model on the training and test data and that on the corpus. The training and test samples were standalone tokens for the most part – none of them were inserted in the context of a sentence. Hence, the features in the model were geared towards those instances, and did little in analyzing the full sentences. Consequently, when evaluating corpus samples, it did poorly, as it didn't capitalize on the richness of information provided by the words that surrounded the sample.

II | CEO NAMES

The procedure used for extracting CEO names was quite similar to that used for company names, except for the following alterations in the indicated steps:

1) The preprocessing stage consisted of finding the sentences in the corpus that contained either a company name found in (I) or the word "CEO". The reason for this is the fact that, at least for the first time the name of a CEO is mentioned in an article, it will very likely be contextualized with his/her company name or his/her designation. I then further analyzed the sentences with a regular expression (([A-Z][a-z]+?\s[A-Z][a-z]+?\s)) to find the candidate samples: bigrams composed of two words starting with a capital letter followed by lowercase letters. That's because, similar to the previous case, at least for the first time a CEO name is mentioned, it will very likely contain the person's full name.

2) The following features were searched in the candidate bigrams: number of capital letters, number of words, average word length. Although counting the number of words in bigrams seems trivial, I found that adding this feature to the model made it more robust when it differentiated them from other proper nouns (e.g. company names) – it made a strong case that CEO names followed a "FirstName LastName" structure.

3) The training and test data were the same, except that now CEO names were classified as positive samples and company names as negative ones.

5) This step wasn't necessary, as the candidate samples had already been extracted in step (1).

6) The method was used on the CEO names rather than on company names. The final list is available in *SchweizerAndre_CEONames.csv.*

7) The predictions made for the test data yielded a true positive rate of 99% and a false positive rate of 2%. This reveals that the NER classifier for extracting CEO names performed considerably better than that for company names, at least on the token level – it was able to correctly classify the vast majority of actual CEO names and only made a few mistakes when classifying other words. The ROC plot (*see Appendix B*) gives us insight as to why this happens: the steep curve towards the right reveals a true vs. false positive rate trade-off that benefits the accuracy of the model. In other words, it is possible to achieve quite a high level of true positive rates with only marginal increases in the false positive rates. If we look at the CEO names extracted from the corpus, we see that there are a lot of samples (2,195). This indicates that the model found around 3 CEO names per article, which would make it reasonable to assume that most of the names were caught. Nevertheless, we can quickly see this isn't completely true, as false positive rates were actually higher than 2%. Although not as high as that of the company name model, a considerable part of the extracted names isn't accurate – the model extracted bigrams following the pattern described in step (1) that weren't actually CEO names. Once again, this is likely due to the fact that the model was trained on data that didn't contain the samples' sentence contexts. Although I attempted to take that type of information into account when extracting corpus data through preprocessing as described in step (1), results would likely have been better if the model could also have been trained using with sentence context.

III | PERCENTAGES

The procedure used for extracting percentages from the corpus was similar to that used to extract CEO names, with the following alterations in the indicated steps:

1) I looked for sentences containing "%" or "percent" in the corpus, rather than company names or "CEO". For finding "percent" I used the following regular expression: [Pp][Ee][Rr][Cc][Ee][Nn][Tt].

2) Features used were: whether the sample contains a decimal point (binary), number of digits, number digits after decimal point, number digits before decimal point. To find decimal points in the samples, count the number of digits after the decimal point, and count the numbers of digits before the decimal point, I used the following regular expressions: \.[0-9a-zA-Z], ([0-9\-]+?)\., and ([0-9\-]+?)\., respectively.

3) The negative samples used in the training and testing data were random numbers rounded to 1-4 decimal places rather than company names.

5) Unlike the case with CEO names, this step was necessary, as I had to further process the sentences extracted in step (1) to find the tokens that were the best candidates to actually be percentages. I did that by extracting numbers or words that preceded either a percent sign or the word "percent" itself using the following regular expressions: ([\.0-9]+?\s?\%) and ([\-A-Za-z_]+?)\s[Pp][Ee][Rr][Cc][Ee][Nn][Tt], respectively. I then filled out a feature frame for this data like I had done with the training and test data, as described in step (4) of section I.

6) The method was used on the candidate percentage samples rather than on CEO names. The final list is available in *SchweizerAndre_Percentages.csv.*

7) The true positive rate on the test data was 88% and the false positive rate, 100%. This shows that, at the token level, the model is good at finding numbers, but is too flexible when classifying them as percentages. A great part of this flaw is explained by the fact that the training/test samples have no other context – as with company and CEO names, we are limited to the individual tokens. This is particularly significant with percentages because without the "%" sign or the word "percent", there isn't really a way to tell them apart from regular numbers. This is further corroborated by the superior performance of the model with the corpus data – the false positive rate is far lower than 100%. The reason for this is how the preprocessing phase in steps (1) and (5) only selected samples that were followed by "%" or "percent" – by taking into consideration the context of the samples, it was able to better classify them. Thus, it is possible to say that, for the corpus data, the model was more accurate than what it was for the test data. One flaw, however, that is revealed from a quick visual inspection of the results, is that the model sometimes missed spelled-out numbers that had more than one word (e.g. point nine percent). Nevertheless, the frequency of these mistakes, when compared to the sizes of the result list and of the corpus, renders them trivial.

APPENDIX

A | COMPANY NAMES

Feature		Regular expression
Word contains…	"LLC"	([A-Z].+?\s?)(?:\W\s?[Ll]\.?[Ll]\.?[Cc])
	"Co"	([A-Z].+?\s?)(?:\W\s?[C][Oo])
	"Ltd"	([A-Z].+?\s?)(?:\W\s?[Ll]\.?[Tt]\.?[Dd])
	"Company"	([A-Z].+?\s?)(?:\W\s?[C][Oo][Mm][Pp][Aa][Nn][Yy])
	"Corporation"	([A-Z].+?\s?)(?:\W\s?[C][Oo][Rr][Pp][Oo][Rr][Aa][Tt][Ii][Oo][Nn])
	"Enterprise"	([A-Z].+?\s?)(?:\W\s?[Ee][Nn][Tt][Ee][Rr][Pp][Rr][Ii][Ss][Ee])
	"Limited"	([A-Z].+?\s?)(?:\W\s?[Ll][Ii][Mm][Ii][Tt][Ee][Dd])
	"Inc"	([A-Z].+?\s?)(?:\W\s?[I][Nn][Cc])
	"Corp"	([A-Z].+?\s?)(?:\W\s?[C][Oo][Rr][Pp])
	"Incorporated"	([A-Z].+?\s?)(?:\W\s?[I][Nn][Cc][Oo][Rr][Pp][Oo][Rr][Aa][Tt][Ee][Dd])

	“International”	([A-Z].+?\s?)(?:\W\s?[I][Nn][Tt][Ee][Rr][Nn][Aa][Tt][Ii][Oo][Nn][Aa][Ll])
	“Intl”	([A-Z].+?\s?)(?:\W\s?[I][Nn][Tt][Ll])
	“Venture”	([A-Z].+?\s?)(?:\W\s?[V][Ee][Nn][Tt][Uu][Rr][Ee])
	“Group”	([A-Z].+?\s?)(?:\W\s?[G][Rr][Oo][Uu][Pp])
	“Association”	([A-Z].+?\s?)(?:\W\s?[A][Ss][Ss][Oo][Cc][Ii][Aa][Tt][Ii][Oo][Nn])
	“Brand”	([A-Z].+?\s?)(?:\W\s?[B][Rr][Aa][Nn][Dd])
	“Technolog(y)(ies)”	([A-Z].+?\s?)(?:\W\s?[T][Ee][Cc][Hh][Nn][Oo][Ll][Oo][Gg])
	“Management”	([A-Z].+?\s?)(?:\W\s?[M][Aa][Nn][Aa][Gg][Ee][Mm][Ee][Nn][Tt])
Number of capital letters		---

Table A.1: List of features and corresponding regular expressions for extraction of company names

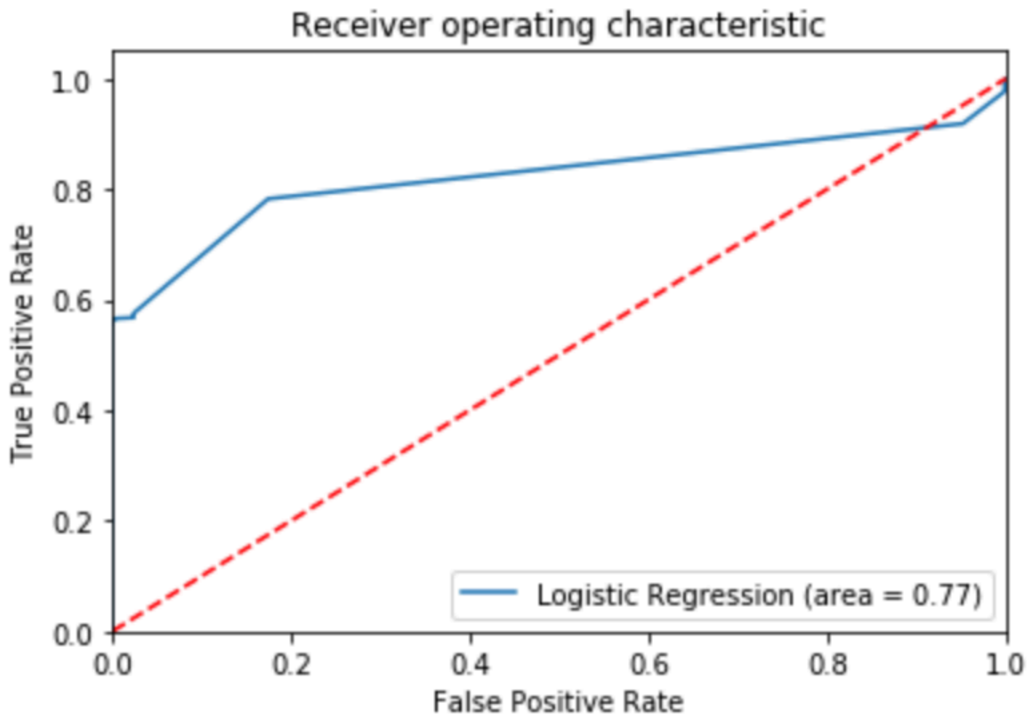

Figure A.1: ROC plot for logistic regression model used in extraction of company names

B | CEO NAMES

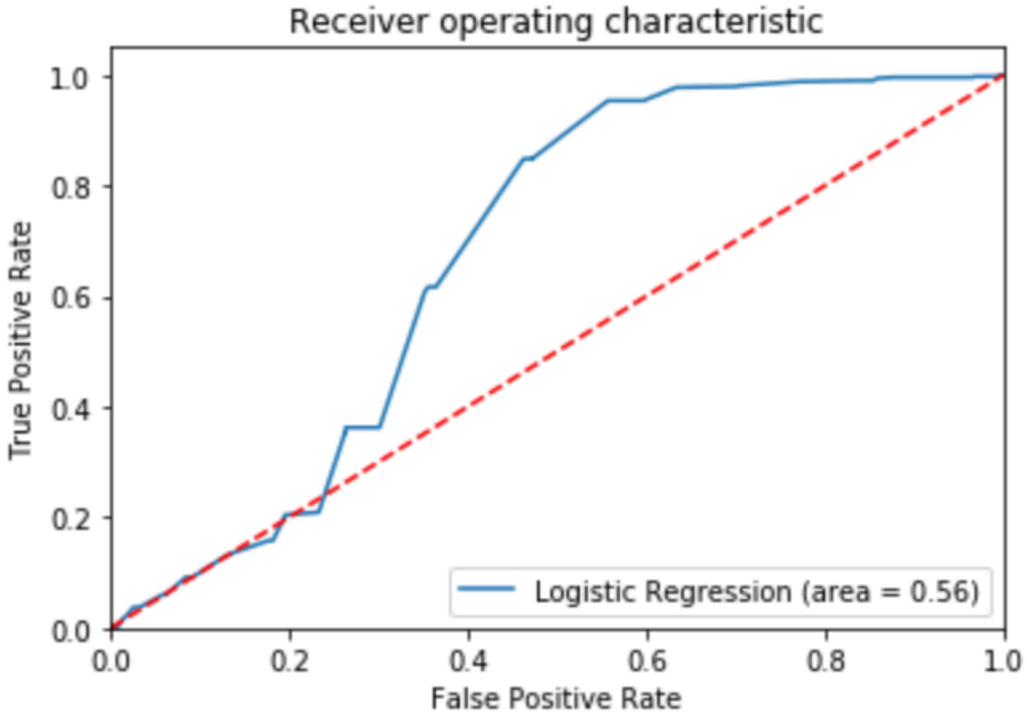

Figure B.1: ROC plot for logistic regression model used in extraction of CEO names