Text Mining for corporate information in Business Insider articles

Yahia El Bsat
IEMS 308

Overview

Our goal is to extract company names, CEO names, and percentages from a corpus of Business Insider articles in 2013 and 2014. This report aims to outline the methodology and tools used for the analysis.

The processing and analysis makes use of the following python libraries: nltk, sklearn, geotext, and enchant.

Preprocessing

The corpus data provided as text files for each day of the week was in an undetermined encoding with non UTF-8 hex values. We first proceed with cleaning the corpus from all non printable characters and hex values then saving it again in txt file. The corpus was then read into python using nltk's PlaintextCorpusReader which allows to easily tokenize all the corpus and access it either by file ids, sentences or words with the appropriate loops. For the purpose of our analysis, we focus on the sentence and word levels (disregarding article level aggregation of information). The total number of sentences obtained in the corpus was 696194 sentences.

Given that those sentence are already tokenized using the function described above, we then proceed to eliminating common stop words both to shorten the processed words and to make the words analyzed more relevant especially that we only care about percentages and proper names which both are not stop words. "The" before other capitalized words were not eliminated as an observation of the training data provided for company names shows that some companies have "The" prefix and we would like to preserve the full name of the companies in our output.

Methodology

CEO and Companies

Given that both CEO and company names are proper names, we use the same process for both. Our aim is to build an NER classifier that can find for all proper names in the corpus a classification of CEO, company name, or other. For that, we first proceed by splitting our data into a test and a validation sample of sentences containing capitalized words. The test sample was taken as the first 200,000 sentences in the corpus (~28%) and we loop through these to find all sentences containing the names provided in the training labels for the CEO and company names. Those sentences were placed in a standalone data frame for further processing. We also take other sentences containing a sequence of capitalized words which don't necessarily contain the training data as those would be relevant in the validation phase of our training model. Regex with the expression "([A-Z][\w-]*(\s+[A-Z][\w-]*)+)" was used for this last part.

For each pair of (sentence, proper name) taken, we create a vector of the following features which we would like to feed into our classifier and that were deemed relevant for the context of the analysis:

- The relative position of the proper name in the sentence
- The number of words in the proper name
- Part of Speech of the token before the proper name
- Part of Speech of the token after the proper name

- Whether the sentence contains the following keywords:
    - For CEOs: CEO, chief, executive, officer, company
    - For companies: Co, Corp, Corporation, Company, Group, Inc, Ltd, Capital, Financial, Management
- Whether the proper name is all English words
- Whether the proper name is the name of a location (country or city using geotext tagging)
- The count of capital letters in the proper name
- Whether the proper name occurs in the beginning of the sentence
- Whether the proper name occurs at the end of the sentence
- Character length of proper names

For the company NER, we use as negative samples CEO names as well as more sentences from the unidentified entities which seem to contain none company information. The reverse was done for the CEO problem.

Once we had those features, we then proceed to splitting that data into further into training and testing (with a 75:25 split). We apply the following regressions models: logistic, gaussian naïve bayes, and random forest. Results are provided under the analysis section but Random Forest performed best.

With our NER classifier ready, we move with mining the remaining sentences in the corpus with the same regex expression provided previously to find all remaining proper names and vectorize the features to classify the entities using our just built NER.

We then proceed to exporting the identified companies and CEO names.

Percentages

To extract numbers involving percentages, we proceed with using regular expressions only without supervised learning as those follow a more or less highly structured form that we can identify with different regex expressions especially after analyzing the different occurrences provided in the training labels.

There are several classes of strings that we want to match. “D” in the below refers to a string of digits (e.g. 0,1,15, 06…):

- D% / D.D%
- D percent / D.D percent
- D percentage points / D.D percentage points
- One percent / one-third percent / twenty percent …

For the first 3 types, we need to match either D or D.D. A digit sequence is \d+ and the second set is optional giving “\d+(?:\.\d+)?”.
We also need a space and a % character: “\d+(?:\.\d+)? %”.
For “percent” or “percentage points” we can do the same matching “percent(?:age points)?.” With the regex pipe we can choose between the % sign or the literal string yielding the following overall expression: “\d+(?:\.\d+)?(?:%| percent(?:age points)?).

For English strings, there are more casework involved. Our string classes are:

- One/two/…/nine percent
- Ten / eleven / .. / nineteen percent
- Twenty /thirty / … / ninety percent
- (any of third) + (any of first) percent (e.g. thirty-six, twenty-nine)

We thus define regex strings for digits, teens, and tens and then taken the four options from above followed by the string “percent” or “percentage point” giving:

digits = “(?:one|two|three|four|five|six|seven|eight|nine)”
teens = “(?:eleven|twelve|thirteen|fourteen|fifteen|sixteen|seventeen|eighteen|nineteen)”
tens = “(?:twenty|thirty|forty|fifty|sixty|seventy|eighty|ninety)”

exp = f”(?:{digits}|{teens}|{tens}|(?:{tens}-{digits})) percent(?age points)?”

With those expressions, we then proceed to applying them to the entirety of our corpora and extracting the relevant percentages.

We thus get about 3600 unique percentage expressions.

Analysis

Company and CEO NER Classifier

The positive/negative split is 76:24 thus our baseline proability is 76%

Model	Accuracy	Precision	Recall	fbeta
Logistic	80%	71%	61%	0.63
Gaussian NB	28%	56%	52%	0.26
Random Forest	91%	86%	84%	0.85

Thus, we selected the random forest model for the NER classifier.

Applying the classifier we get 37000 unique CEO names and 65000 unique company names. Of course, there are false positives in the results and further features can be devised to make the model better but by looking at the results, a lot of the results make sense and the majority are in the appropriate category.