Predictive Capability of Text Mining in Yelp Reviews

Analyzing the text mining techniques and several machine learning algorithms in predict business and reviews attributes

Giuliano Sposito

November, 2015

This is the report for the Capstone Project of the Coursera Data Science Specialization by Johns Hopkins Bloomberg School of Public Health.

Introduction

In this study we will study the ability of the texts, in the Yelp's Reviews dataset (Yelp Challenge, 2015), to be used to predict things business characteristics and attributes. For example, is it possible, through them, to predict the evaluation stars, if the establishment has free Wi-Fi, or have parking? There in the text comments, information about the price, if it is good for kids or groups? Comments about how noisy the environment is? In addition, we will try to understand what would be the best approach for text prediction and what are the limitations of this technique in the Yelp's context? Has a model, trying to predict this business attributes enough accuracy to be considered useful?

Methods

General Text Mining Approach

The classic textbook approach to Texting Mining is to create a *Corpus* of text to be preprocessed. In the pre process the text if *normalized* and *cleaned* transforming all words to lowercase, removing punctuation, extra whitespaces and even numbers, removing non-significant words, as *a, and, also, the, etc.* (it's known as *stop words*). In some case, the common word endings (e.g., "ing", "es", "s") are removed also. This is referred to as "stemming" documents. We stem the documents so that a word will be recognizable to the computer, despite whether or not it may have a variety of possible endings in the original text.

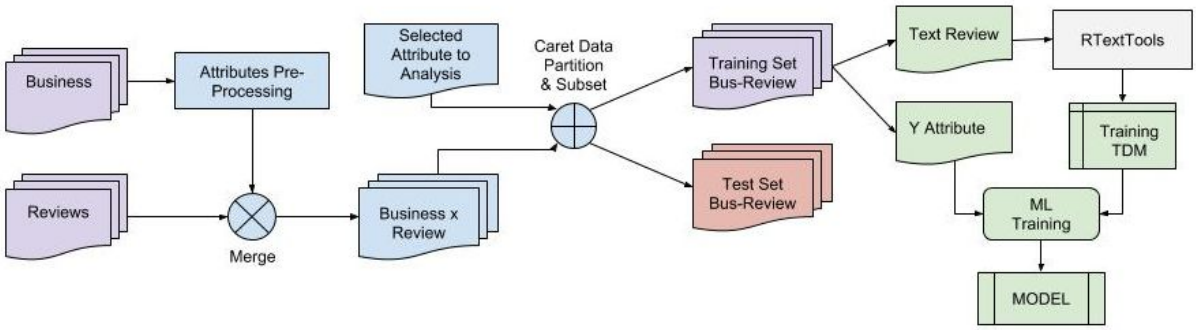
After this process is created a Term Document Matrix, or TDM (Feinerer, 2005), that is a sparse matrix that counts the occurrence of all unique words found in text corpus for each document in corpus. Therefore, the TDM characterizes each document, so it can be used as features to be analyzed or as input in machine learning algorithms trying to predict aspects to the object associated to the text or the text itself. A great description of the process above, using R, can be seen in this online tutorial.

Business Attributes Predictive Analysis Workflow

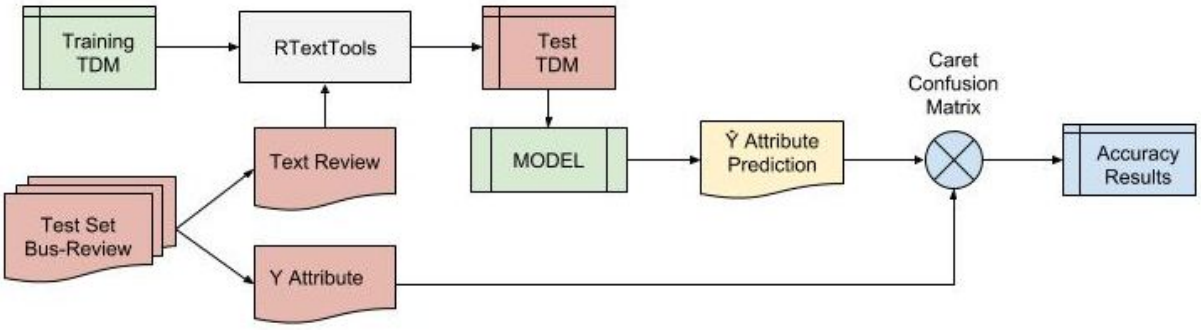
The method to study the predictive capability of a text from Yelp `Review` tells us things about an `Business` attributes, so basically what we did was to test several machine learning algorithms accuracy in use the TDM of text `Reviews` to predic several `Business` attribute.

A diagram showind the first step (1) can be seen in the image below, the `Business` Yelp dataset is preprocessed to transform and prepare the attributes to be used in the fitting process (transform them in factors, converting the `NA` values in an `unknown` value, merging some scales to reduce the dimensionality, etc.). After this, the `Business` dataset is merged with the `Review` dataset, so for each text `Review` now has the attributes values to which the text refers. This `Business-Review` dataset is separated in **Training** and **Test** datasets using

Caret CreateDataPartition function. The **Training** dataset's text is processed using the RTextTools package to extract the TDM, and to fit a model to a specific `Business` attribute.



In the second step (2) the accuracy of the model generated is verified using the `Business-Review` Test dataset. This is done creating another *TDM* for the text, but containing the same words present in the *Training TDM*. This is necessary because the model was trained in the *training glossary* so the *Test TDM* must have the same words. The *RTextTool's create_matrix* function can take another TDM as input to keep the glossary the same. After that, the model fitted in the first step is used to predict the attribute value, and the output is confronted with the real value. The accuracy (and others performance value) is measured using the *Caret's confusionMatrix* function.



Source Code

As said, for the analysis we use the RTextTools package that provides text-mining infrastructure for machine learning analysis, also used the Caret package to sample and split the dataset and analyses the accuracy. For KNN and Naive Bayes algorithms, we use the e1071 package. Because the page number limitation we won't show any code in this article. You can see all the code and analysis at github, the repository tells the code structure and organization, also the basic instruction to run de code.

Algorithms and Business Attributes

For time and resources limitations (mainly process power and memory size), for this study we choose analyze the performance of the following machine learning algorithms: Support Vector Machine (SVM), Maximum Entropy Classifier (MAXENT), Decision Tree Learning (TREE), Random Forest, K-nearest neighbors (KNN) and Naive Bayes Classifier (NAIVEBAYES). Aurangzeb Khan (Khan, A. 2010) write an excellent article about these algorithms in the context of text classifying.

The `Business` attributes selected for the study can be seen in the table below. They was first preprocessed to transform its values in factors and be ready to be used as categories in the ML Classifiers. The `NA values`

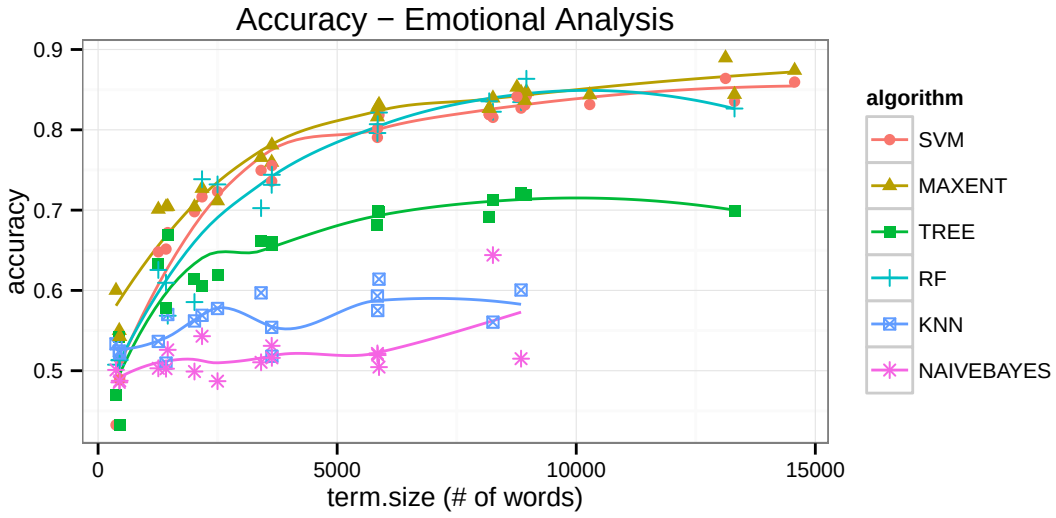
was reclassified as `unknown` to be processed. Also we do some exercises reducing the number of class of some attributes, for example, the for the `star review` we also create an `emotional attribute` reclassifying the reviews with 1 and 2 stars in "negative", and the reviews with 4 and 5 stars in "positive" (and removing reviews from 3 stars). Details of each attribute transformation can be seen at source code at GitHub.

This was made because text classifiers notoriously works better with fewer classes and with extreme values. Business atributes preprocessed (and its possible output levels) are: **hasWifi** (2: no | yes), **hasWifi** (3: no | unknown | yes), **hasParking** (2: no | yes), **hasParking** (3: no | unknown | yes), **noiseLevel** (4: average | loud | quiet | very_loud), **noiseLevel** (5: average | loud | quiet | unknown | very_loud), **noisy** (2: no | yes), **noisy** (3: no | unknown | yes), **priceRange** (4: 1 | 2 | 3 | 4), **priceRange** (5: 1 | 2 | 3 | 4 | unknown), **pricy** (2: no | yes), **pricy** (3: no | unknown | yes), **goodForKids** (2: no | yes), **goodForKids** (3: no | unknown | yes), **goodForGroups** (2: no | yes), **goodForGroups** (3: no | unknown | yes), **hasTV** (2: no | yes), **hasTV** (3: no | unknown | yes), **smoking** (4: no | outdoor | unknown | yes), **goodForSmoking** (2: no | yes), **goodForSmoking** (3: no | unknown | yes), **reviewEmotion** (2: negative | positive), **reviewEmotion** (3: negative | positive | unknown), **reviewStars** (5: 1 | 2 | 3 | 4 | 5).

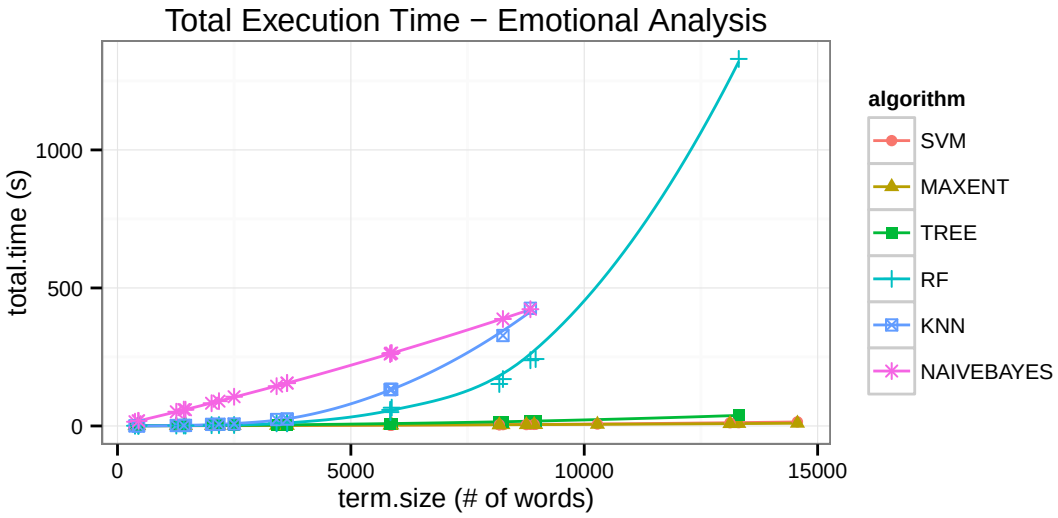
Results

Algorithm Analysis

To check the performance (accuracy and time-consuming resources), we confront the chosen algorithms in a simplified classification test. The task is to classify if a `Review` (*reviewEmotion* attribute) are `positive` or `negative` from `Restaurants` category only, here are the Accuracy Results by the number of terms in the Term Document Matrix.



We can see that, for this kind of problem, the SVM, MAXENT and RF algorithms has the best accuracy with values around 85% after 10000 terms in TDM. We also can note that the accuracy stabilizes next 15000 terms. We analyzes the total time to train and predict each test, see below.



The TREE, SVM and MAXENT algorithms have less time consuming, so taking accuracy and performance as selection criteria, we choose perform the predictive analysis of remaining attributes using only SVM and MAXENT algorithms.

Predictive Capability of Business Attributes

For each of the attribute select to the study, and for each of the algorithms we perform the accuracy study, first removing the `unknown` values and after with the `unknown` values. Beside this, each study was applying once to the *Restaurant* category only and after to *All* categories in the study dataset (We hypothesized that the `Review` text glossary for one category is different for other category). Also, for each attributed analyzed, we eliminate the prevalence effect in the samples, i.e., we balance to equal numbers of the output classes to a specific attribute.

After applying the Accuracy test, several times, trying to predict the attributes using SVM and MAXENT algorithms to a Term Document Matrix with more than 15000 terms, of balanced `Business-Review` samples, and take an average, we show bellow the results of the prediction that achieve **more than 70% of accuracy**, ordered decreasingly by the MAXENT accuracy.

	Attribute	Classes	Business Category	SVM	MAXENT
1	reviewEmotion	2	all	0.8685	0.8835
2	reviewEmotion	2	Restaurants	0.8645	0.8795
3	goodForSmoking	2	all	0.7800	0.7925
4	pricy	2	all	0.7725	0.7840
5	pricy	2	Restaurants	0.7765	0.7815
6	goodForKids	2	Restaurants	0.7390	0.7485
7	goodForSmoking	2	Restaurants	0.7328	0.7430
8	goodForKids	2	all	0.7285	0.7195
9	pricy	3	all	0.7065	0.7165
10	smoking	3	all	0.6930	0.7035

Discussion

Looking to the algorithm performance analysis made for the `reviewEmotion` attribute of `Restaurants`, we can verify:

1. The algorithms studied has different accuracy, and Random Forest, Support Vector Machine and Maximum Entropy has similar and better results. The performance of these algorithms is attached to the great number of features (the terms in the TDM) and sample size (around 1.1 million of `Review` text available to train them).
2. The accuracy increases with the number of terms used as algorithm input in the TDM, but stabilizes after 15000 terms, so the three algorithm has different processing times, so for this kind of problem is important to balance the number of terms, the precision desired and the computer processing power available. In your study we concluded that using MAXENT with enough samples that generate around 15000 to 20000 terms in the TDM, are good parameters for the study of predictive capability of text `Reviews`.

We can see, in the results table, the following interesting results:

1. The Maximum Entropy (MAXENT) algorithm has a slight better accuracy than SVM.
2. The text `Review` in Yelp database has a good signal (or information enough) to prediction *the emotion of a review* (see results #1 and #2 of reviewEmotion attribute) with accuracy from 0.86 to 0.88, independently of business category.
3. Users also seems to comment in the reviews information about the smoking status (result #3) and the business Price Range (results #4 and #5) because the models can predict the information with more than 0.75 of accuracy.
4. Surprisingly the information about `goodForKids` in `Restaurants` (study # 6) is present in the review texts, while information like hasParking and Wi-Fi (didn't appears in the results because has accuracy less than 0.7) aren't in the text reviews.
5. We hypothesized that the glossary in the reviews differs from a category to other (seems reasonable), but the category information affect the results in distinctive ways. Attributes like `reviewEmotion` and `goodForSmoking` are better predicting ignoring the business category, while `goodForKids` and `hasParking` seems better when only the category `Restaurant` is selected. Therefore, a Machine Learning Classifier has to take in account the Business Category as one of the feature or input, to do better predictions.
6. In general, when the model is trying to classify the text in two extreme classes (when it has to decide only between two values), it has better accuracy than when has to classify in three or more classes. See the `pricy` attribute (#4) with two classes, it was better than when it considers the `unknown` class (#9). The original attribute `noise level` with four levels (five levels with `NA`) has failed to appears in the table, i.e., has accuracy below the 0.70.

Further analysis

1. With more time and computational power others algorithms could be test. It would be interesting see the performance of Neural Networks (GLMNET and NNET), Gradient Boosting Machine, Bootstrap Aggregating and Latent Dirichlet Allocation (LDA/SLDA).
2. There are information about Emotion, Smoking, Price Range and *suitable* for kids in the text reviews, this study didn't try makes a predicting algorithm for this features, but the existence of the signal appears to indicate that is possible combine text and other characteristics to predict this business attributes.
3. We eliminate the prevalence effect because we are interested only in the Text Predictive Capability, but adding this information back it would be possible make an analysis of Machine Learning classifiers using True Table (True Positives, False Positives, etc.), to define if a business attribute can be, indeed, predicted by Yelp `Review` information.