Using Various Machine Learning Classifiers to Predict Smoking Status From Doctor's Notes

Michael Kwan

July 6, 2017

Abstract

The clinical diagnosis of a single doctor's visit can be usually either underused or incorrect in determining a patient's status, and a false reading may lead to further consequences. With advances in machine learning, certain algorithms can take in these doctor's notes and interpret them to return a diagnosis of the problem, using feature extraction methods such as n-gram and methods such as SVM or decision trees. In this paper, we look into these methods in classifying patients into four groups of smokers, through textual analysis of the doctor's notes of the visit. While each method has its strengths and weaknesses, we found that the Naive Bayes algorithm was the most accurate at determining the smoking status of the patients, with a categorization accuracy of 81%. This method can then be applied to further classification problems, such as problems arising from obesity, or classification of certain intensities of cancer.

1 Introduction

Every year, the biomedical computing group i2b2 (Informatics for Integrating Biology and the Bedside) release a dataset as well as a corresponding challenge that usually revolves around two topics: de-identification and classification. While the de-identification process revolves around identifying proper nouns that contain sensitive information such as addresses, names, and other identifiable fields, the classification process requires a machine learning algorithm to determine a certain status of a patient with the highest accuracy. In 2006, i2b2 released their smoking dataset, where the challenge required teams to utilize a training set of 398 patient visit notes and from there correctly identify the current smoking status of patients on a training set of 101 separate, unique visits. While some teams resorted to identify proper nouns related to medicine, and then use Bayesian networks and decision trees to determine the state of the patient, it misses the other relevant details of the visit and the underlying context of the sentences written. For example, the word **ibuprofen** was the only word tagged by the aforementioned classifier in the following sentences and diagnosed the same status for both patients, but it fails to analyze the context of the context:

Because of his mild pain problems and fever, I chose to prescribe **ibuprofen** to the patient. Since his gas and bloating resulted from doses of **ibuprofen**, I recommended that he stop taking it.

Thus, we introduce a new approach at the problem, taking up a different usage of machine learning to solve the problem, through other feature extraction methods as well as classification algorithms. In order to solve the context problem mentioned before, we use n-gram feature extraction to take segments of every word in the review and segment them as a feature, as well as utilizing regular expressions to cover any other potential features that might have been missed. For classification algorithms, we look towards the Stanford classifier, which takes these two feature extraction methods and uses a maximum entropy (maxent) Hidden Markov Model (HMM), and others such as Naive Bayes, Support Vector Machines, and Random Decision Forests.

2 Methodology

2.1 Data Cleaning

When the dataset is acquired, the patient notes are structured inside an mark-up language tree, and our end result would be to store each category that the record is classified to (one from `CURRENT SMOKER, NON SMOKER, PAST SMOKER, UNKNOWN`) as well as the record itself to an array in Python. We accomplished this with a simple Python script that would parse this tree, extracting the relevant text that would contain all of our features to go into the classifier. While we were able to accomplish this task, the Stanford classifier required a bit more complex configuration, and thus separated it from the rest of the algorithms.

2.2 Feature Extraction

Using the training dataset from the i2b2 site, we were able to train each classification algorithm through both the Stanford Classifier and the Python package `sci-kit learn`. However, we first needed to be able to extract the individual features from the record. This was done through the n-gram process, to find all n-character segments of a record incrementally. This resulted in $\sim 45,000$ features, which as a reasonable amount of features for the computer to process. However, some of them were superfluous, such as the tagline at the beginning and end of every note/visit would be extracted, and its frequency would result in a significant weight in the maxent classifier, and thus its removal cut down the number of features by around five thousand, leaving the resulting dataset at 40,000 features.

2.3 Classifier Validation

After training our datasets with the mentioned algorithms, we tested the dataset, returning a macro-averaged accuracy of the returned result of the classifier compared with the results given in the training set. The results comparing the different algorithms in the next section. For the `sk-learn` algorithms, we were able to build a pipeline that would be able to tokenize features, eliminate irrelevant/infrequent features, train a classifier, and calculate accuracy versus the test set using the various

3 Results

After we calculated the accuracy of each classifier against the true data set, their accuracies were compared to each other, resulting in the table below:

	Maxent	Maxent-kfold	Naive Bayes	SVM	Decision Tree	Random Forest
Accuracy	69 %	72 %	81 %	78 %	75 %	79 %
Package	`stan-nlp`	`sk-learn`	`sk-learn`	`sk-learn`	`sk-learn`	`sk-learn`

The Naive Bayes classifier had the best categorization accuracy, topping the chart at 81%, due to the training datasets' small size and relatively large sample bias. This problem with the database can be solved with correct sample weighting, and a feature that I'm looking to implement for the next iteration of the project. However, the performance of the highest classifier does not mean that it is the one that would deliver the most optimal results, as others trail by a trivial amount, and with some optimization could also produce results similar to the Naive Bayes classifier.

4 Discussion

Even though each method were able to reach similar results, each classification method has their pros and cons, such as the way they handle the amount of data given to the algorithm, the way optimization is supported and executed, and how new test data is mathematically computed result in different conclusions about the potential of the method. Foe example, the Maxent (Stanford Classifier) methods are able to perform more admirably if the training data set could be expanded, comparing the sub 70% result with the 89% of the "20 Newsgroups" forecast, which consisted of a training dataset that was 50 times the i2b2 smoking dataset. The Naive Bayes classifier is a different case, as our class label of unknown is not only affected by sample bias, as well as unspecific weights from other class labels.

This rough proof-of-concept of the use of natural language processing techniques in the field of medicine prove that it is possible for machine learning and artificial intelligence to be a good assistant to the doctor, aiding the author of these notes as they diagnose the patient through his or her notes. Being able to resolve events that happened in the past would be ground breaking in bringing technology in the clinic, and signals a pathway into predicting the future. With this use of classification in this smoking dataset, but can be used in foreshadowing the disease that may come from obesity, as well as categorize the intensity of a disease, such as the stages between mild and severe Parkinson's or the levels of cancer in a patient, just from the text that a doctor writes on his pad.