**Introduction/Problem Description**

With the rise of social media and the internet, the opinions of thousands can be accessed with just a keystroke. Never have the opinions of multitudes of people concerning a platitude of topics been so readily available. Websites such as Facebook, Twitter, and Yelp, offer platforms where people can voice their opinions and express their approval or disapproval for an issue or product. This information has been useful for the those connected to the internet and many would relate to trying to pick a good restaurant for dinner or finding the next movie to watch using these platforms as a means to access information.

However, consumers are not the only ones taking advantage of this new abundance of data. Businesses have been using these services as well to retrieve valuable feedback about their products or services. In turn, these businesses have used this feedback to either enhance their product, or to find the niche where their product will be successful. A challenge many of these businesses have run into is the sheer abundance of data and how to consolidate this data to understand the general consensus among consumers towards the product or service.

It would be unreasonable to physically sieve through each and every comment or post and we hope to solve this problem using machine learning algorithms. The goal of this project is to develop an application to process a comment and analyze the author’s opinions and sentiments based solely on word choice and sentence structure. Using the Naïve Bayes and J48 tree models the provided application will be able to determine whether a provided comment is negative or positive. Based on the accuracy of each model we will determine which is best suited for use in sentiment analysis of text.

Previously, most sentiment prediction problems have been addressed using Naïve Bayes classifiers and SVM (support vector machines). J48 trees have been used for sentiment analyses on many platforms including Facebook and Twitter. In these studies, researches were primarily categorizing sentiment data in two categories: positive/negative. For our project, we hope to use Yelp restaurant reviews to better determine the emotional state of the user. Since Yelp utilizes a 5-star rating system for its clients, our goal is to more specifically determine the author’s emotional state by increasing the number of categories.

**Description of the Data Used in the Project**

Movie Review sites Like IMDB allow users to rate Movies on a scale of 0 to 5. Along with the rating users can also leave a review explaining what they liked about the movie. This combination of data points fits really well with sentiment analyses due to the reviews being already pre rated by humans.

The data that we are using in this project was compiled by a research group at Cornell University. We will be examining multiple data sets and will choose the one that is the most suited for our project’s scope. From the many sets the group has compiled, our project has chosen to focus on sentiment analyses. The sentiment polarity dataset classifies each review left by an IMDB visitor as either positive or negative.

The datasets were collected by extracting specific features from text. Since this project is text classification based, we will be using algorithms That are better suited for text classification.

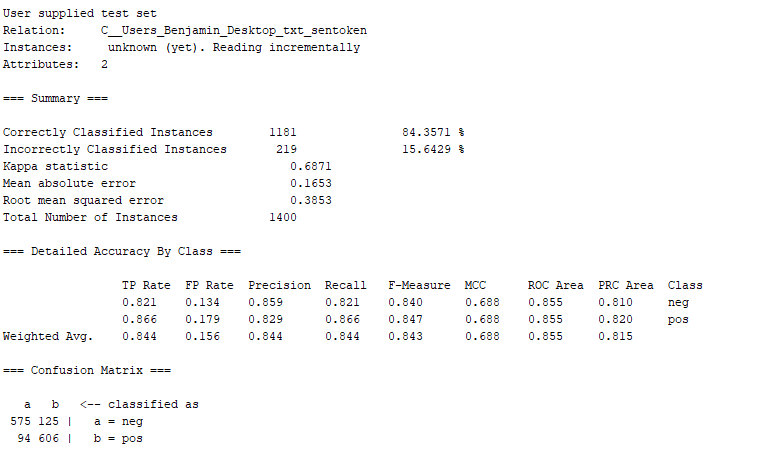
**What have you done so far**

So far, our group has selected our dataset. As stated prior, the data that we are using in this project was compiled by a research group at Cornell University. This set consists of 1000 positive and 1000 negative processed reviews from the IMDB movie review website. The data and “Star” rating system where pulled from the website. The rating was used to categorize the review as positive or negative, with 5 to 3 ½ stars being positive and 2 ½ stars and below being negative. The Data itself comes in the form of 2000 individual text documents that have been categorized in two folders labeled positive and negative. Each folder holds and classifies 1000 movie reviews each. The major task here has been formatting the provided data to be used easily with the WEKA API.

A large part of our time with this project has been spent learning the WEKA API and how to manipulate datasets with it. WEKA is a java based collection of machine learning algorithms for data mining tasks. The collection also contains tool for data pre-Processing, classification, regression and visualization. The algorithms and tools provided in this collection are being called from our java code and are then used to process, classify analyze the data.

Since our project required us to use WEKA we had to reformat our dataset to fit the API. Though pre-classified, our data was placed int 2000 separate files and therefore not formatted to work with the WEKA API. In order for us to use our review data it has to be formatted in a Attribute-Relation File Format or “ARFF” file. An ARFF file is an ASCII text file that describes a list of instances sharing a set of attributes and is used in WEKA to hold and analyze datasets. In order to reformat our data we utilized WEKA’s “TextDirectoryLoader”. By using The TextDirectoryloader, we were able to loading the 1000 positive reviews and the 1000 negative reviews into a WEKA .arff File all at once.

Once our data had been correctly formatted we filter our data. First, we converted our string attributes into a set of numeric attributes that represented the word occurrences from the text contained in the strings. This was accomplished using WEKA’s StringToWordVector class and trigrams. Through the use of the StringToWordVector class we were able to narrow the scope of instances we would be testing. Next, using WEKA’s InfoGainAttribueEval class we evaluated the worth of the given attributes by measuring the information gain they provided. Using info gain has allowed us to determine which words are most useful in determining a reviews sentiment polarity.

Once our data was formatted and filtered, we generated a test model in the form of a J48 decision tree. The J48 Decision tree classifier, in order to classify a new item, first needs to create a decision tree based on the attribute values of the available training data. So we felt that this algorithm was the best fit for the task of evaluating text filtered to favor high information gain. With a batch size of 100, binary splits disabled, a confidence factor of .25 and a minimum number of Objects of 2, our model was able to correctly predict the polarity of a provided review with about 84% accuracy(fig.1).

**What remains to be done**

Though testing on our J48 model has been started we still need to run more tests. From our current results it seems there is still some discrepancy with our model. As you can see in (fig.1) confusion matrix classification errors where a little high but overall acceptable. By modifying the filters, we hope to see an increase in our accuracy. To insure a higher accuracy rating are going to test different filter configurations to see which gives us the best fit. We hope to achieve this this by altering the batch sizes and modifying the N-grams we are using.

We also still must run tests on the Naive Bayes model. Our plan is to use a multinomial Naïve Bayes model and determine if it is a better fit for our data set. As with the J48 model, In order to see if this model is a good fit for our data we are going to test different filter configurations to see which gives us the best fit.

Lastly, once all of our models have been made, our final task will to run experiments on the models using instances which are completely different from the training set. This will consist of us providing our models sentences that it has never seen before. The models will then be evaluated on how closely they are able to determine whether or not a sentence is either positive or negative.