Midterm Progress Report

Table of Contents

Executive Summary:

To make predictions on tweets, testing proved that using a hybrid approach between a Naive Bayes Classifier and a Neural Network Produced the best results. This hybrid approach is then used in order to determine sentiment on a selected tweet from our database to the user. A configuration file approach was implemented to cache the results of past queries to make future queries on the same keywords return a result in faster time.

Results to Date:

Description of Beta Prototype:

Our beta prototype incorporates a hybrid approach to prediction. First a tweet is fed to Naive Bayes Classifier. If this classifier is below a certainty threshold for a given tweet, the tweet is then fed to a trained neural network. Neural networks work by simulating a brain through neurons, and improve using a genetic algorithm. To begin the training of a neural network, a network is randomly generated, and tested by predicting known sentiment values. The best performing networks will pass their traits to the next generation. These traits come in the form of weights attached to each neuron, so successive generations will calculate the sentiment using mutated weights. This process will continue until significant improvements stop being made, where mutations can no longer make the prediction more accurate. This increase in accuracy typically comes logarithmically, so at some point it isn’t worth it to generate more networks.

When an Amibroker user wants sentiment data on tweets containing a specific keyword, the user will fill out a config file that mentions the types of tweets being requested. This config file will then be uploaded via a winforms app to a database that contains a config ID as well as the contents of the config file. When the user is ready to retrieve the sentiment on the keywords within a certain config file, they will open a winforms app that asks for a config ID and an optional earliest date. The app checks if the config ID and date are valid before it grabs the keywords from the config file and runs a query on the database to retrieve the sentiment associated with them. A CSV file is returned to the users specified save location with the columns dateTime, and sentiment. Amibroker is able to read data from any CSV file so the returned data will be interpreted by Amibroker. The database for the config files as well as the database of collected tweets is managed by the web hosting service Amazon Web Services(AWS).

Test Results:

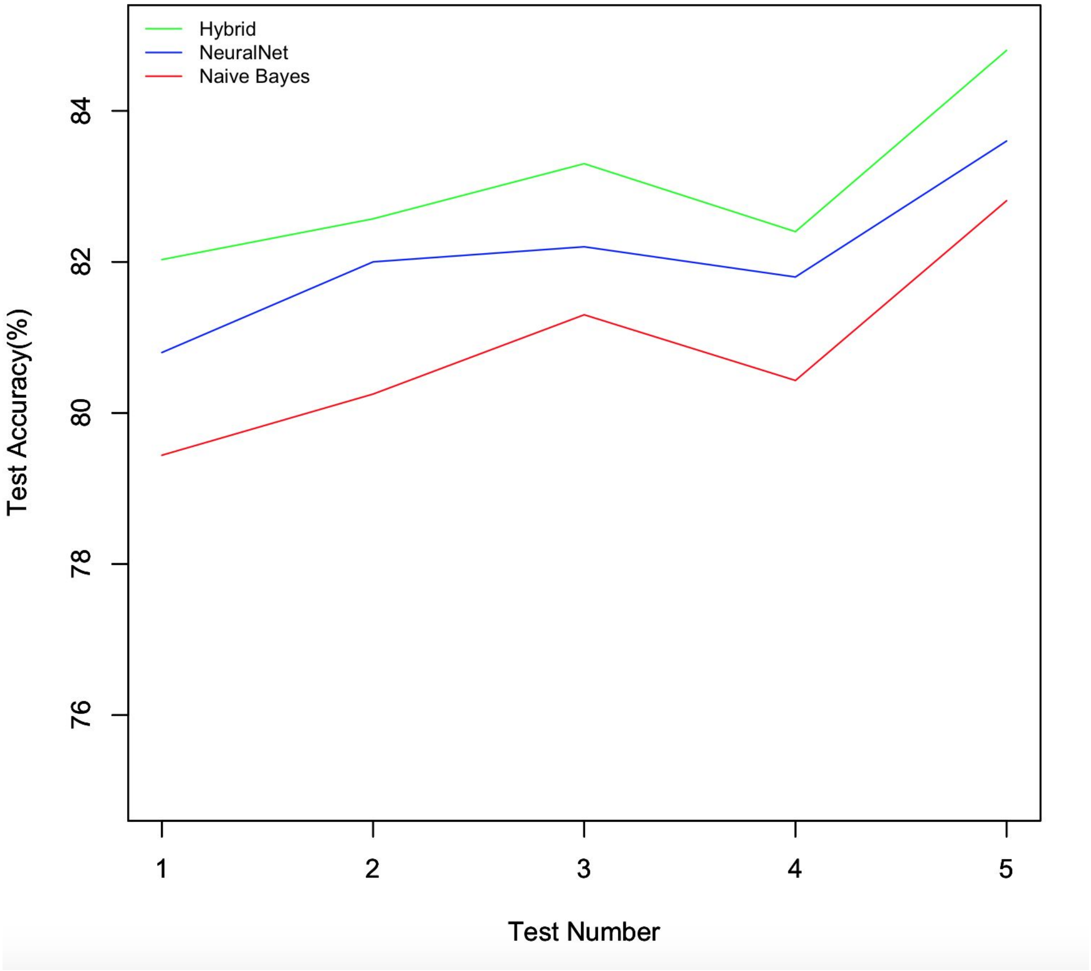

Image 1.1

Image 1.1 displays the test results of running Naive Bayes Classifier, the Neural Network, and the hybrid approach over five different sets of 10,000 tweets. Over the five tests, the Naive Bayes Classifier had an average of 80.85% accuracy, the Neural Network had an average of 82.08 accuracy, and the hybrid approach had an average of 83.02% accuracy.

Validation Results:

Analysis, Modeling and Simulation Results:

Broader Impacts Considerations:

The idea to use the sentiment to predict stock prices is not a new one, but the rise of machine learning, cloud computing, and the ability to process mass amounts of social media data has recently made this idea feasible. Because this area of computer science is actively developing, our project could potentially have major influence on future works in this area. This assumes that our project is successful, however. If our project is not successful, we have at least created a unique implementation of a sentiment analysis system. Although we have followed many previous works, it is unlikely that our project is completely similar to other works in terms of implementation. Our implementation may provide a good starting point for future projects in this area.

In the end, we will have provided Amibroker with a functioning sentiment analyzer that is capable of using real time or historical data to backtest on stock prices. This will allow clients to estimate stock trends before they occur, which can lead to financial gains. Of course, this is assuming that the sentiment retrieved from tweets correlates with stock prices.

Summary of Work Remaining This Semester:

We still have a decent amount of work to do this semester. The remaining work can be broken down into several parts. First, we need to finalize the sentiment analysis algorithm. The algorithm is already working, but some tweaks may need to be made to improve it's accuracy. Some tweaks we might make could be extracting different features from the text to use as input to the neural network, adjusting the size and shape of the neural network, adjusting the accept/reject threshold after the naive bayes phase of the algorithm, or integrating new technologies such as Facebook's Fastword into the algorithm. However, since our time is limited, we will probably not make any major changes to this part of the project.

Next, we need to finish the code implementation of our project. Our project is going to be implemented as a RESTful ASP.NET web API in C#. Most of the implementation is already complete, however, some of the implementation is not done. One thing that still needs to be implemented is the configuration file system. The configuration files provide a way for the user to specify what tweets to match and feed into the sentiment analysis program. We currently have a simple version working, but we plan to develop the system to allow more complex configurations.

The client side executable that is responsible for grabbing sentiment based on config files and returning the data in a CSV format still has some work to be completed. As of now, config files are uploaded to the database through a web API endpoint. The goal is to allow the client to upload config files from the same winforms app used to retrieve the sentiment. The config files include keywords that can be separated by ‘and’ and ‘or’ which will act as expected. For example, the config file containing ‘microsoft & surface’ will query all tweets that contain both ‘microsoft’ and ‘surface’ in the text and run the sentiment analyzer on those tweets. The next step is allowing the user to include parentheses inside of the config files, opening up a ton of new opportunities for config file customization.

Once the code is implemented, we need to move the project to Amazon Web Services so our mentor can use our work after we are done with this semester. Moving to AWS is probably going to involve some minor code changes. None of us have much experience with AWS, so this part of the project may take some time. We already have parts of the project running on AWS, such as our database of sample tweets, and our file storage system for the configuration files. We have also had success running the web API on AWS, but there are still some major bugs that need to be worked out before the product is usable.

Finally, once we have everything in a working state, we need to test the project on real stock market data. For this part of the project, our mentor will likely be heavily involved and guide us through his process. Our mentor has a lot of experience analysing this type of data and will be able to evaluate how well our sentiment algorithm works, and if we should make any changes to it. We will be using software called Amibroker to perform this analysis. Assuming the everything works, this part of the project will tell us whether or not our sentiment algorithm works for identifying trends in the stock market.