A NEURAL NET FOR STOCK TREND PREDICTOR

CS 297 Report

Presented to

Dr. Chris Pollett

Department of Computer Science

San Jose State University

In Partial Fulfillment

of the Requirements for the Class

CS 297

By

Sonal Kabra

Dec 2016

TABLE OF CONTENTS

INTRODUCTION

The “Neural Net Stock Trend Predictor” is an approach to developing a stock trend predictor. This report describes preliminary work towards my CS297 project. Stock prediction using computers is also known as algorithmic trading (AT) or automated trading. Since the mid-2000s, nearly all the financial trades are executed via computers. Much of that trading occurs algorithmically, with computers executing purchases and sales in response to a market. About 66 percent of all equity transactions in the United States now are done via high frequency or algorithmic trading. [1]

Algorithmic trading is the process of using programmed computers to follow a defined set of instructions for placing a trade in order. This helps to generate profits at a speed and frequency that is impossible for a human trader. These instructions can be based on volume, timing, price, or any mathematical model. Algorithmic trading helps to take out the human emotional impact on trading. Thus, increasing the profit opportunities for the trader. [1]

For example, if a computer is provided with all the historical stock data, it can learn and analyze the patterns in the data and can predict the future value for a stock based on its findings.

I am using Quandl for pooling all the historical stock data, and Yahoo-Finance for real time stock data. Both websites provide financial and economical data. The stock data pooled from there contains 5 features as shown in the following figure:

Date	Open	High	Low	Close	Volume
2004-08-19	100.000168	104.060182	95.960165	100.340176	44871300.0
2004-08-20	101.010175	109.080187	100.500174	108.310183	22942800.0
2004-08-23	110.750191	113.480193	109.050183	109.400185	18342800.0
2004-08-24	111.240189	111.600192	103.570177	104.870176	15319700.0
2004-08-25	104.960181	108.000187	103.880180	106.000184	9232100.0
2004-08-26	104.950180	107.950188	104.660179	107.910182	7128600.0
2004-08-27	108.100185	108.620186	105.690180	106.150181	6241200.0
2004-08-30	105.280178	105.490184	102.010172	102.010172	5221400.0
2004-08-31	102.300173	103.710180	102.160177	102.370175	4941200.0

Therefore, the objective of this project is to analyze the data and generate features that help predict stock prices. Besides the generated features the aim is to build a computer programmable model using a neural net and various machine-learning techniques. Neural

net is an information-processing paradigm. The concept is inspired from the way biological nervous systems, such as the brain, process information.

The following are the deliverables that I have implemented this semester to understand the essence and the working of algorithmic trading system.

In Deliverable 1, I created a presentation covering the basics of an algorithmic trading system and calculated some basic technical indicators for data analysis. In deliverable 2, I developed a neural network to predict the starting month for given data series. More details on this will be discussed under the section titled 'Deliverable 2.' In the next deliverable, i.e. Deliverable 3, I developed a program to predict the closing price for a stock using the Decision tree regression analysis and K nearest neighbor regression analysis. In Deliverable 4, I developed a program that predicts the returns of a portfolio of stock.

DELIVERABLE 1

In order to move towards implementation, it will be useful to understand the basic of algorithmic trading. The process of algorithmic trading is as shown in figure1. [2]

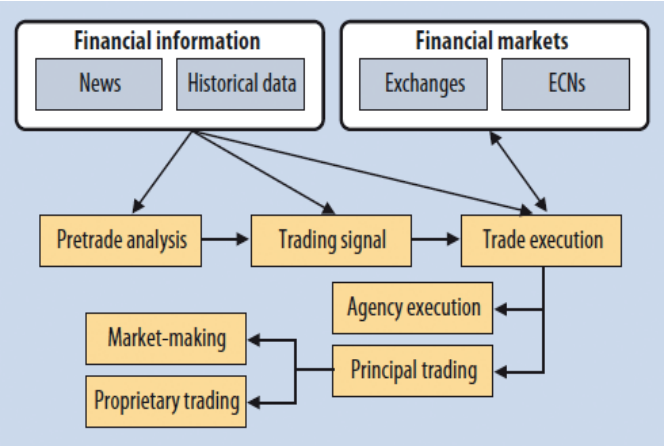


Figure 1: Algorithmic Trading Process [2]

The trading process is divided into five stages as follows:

1. Data accessing and cleaning:

Algorithmic trading obtains data from various sources and cleans the data according to the algorithmic requirements. The data can be raw data, cleaned data or analyzed data.

2. Pre-trade analysis:

In this stage, based on inputted data, the AT analyzes the various properties of the financial instruments in order to identify trading opportunities. The properties can be value of a company or sentiment analysis for a company. The analysis can be:

Fundamental Analysis: In this the factors, which might affect the instrument's value such as the economic state of a country, or price to earnings ratio of a company are analyzed.

Technical Analysis: It tries to predict future price movements of an instrument. This is based on its price history, various trading histories like volume traded previously. Thus, trying to identify the trading patterns.

Quantitative Analysis: This is a mathematical and statistical analysis of an instrument. It describes the randomness of the price of an instrument. This is mainly useful in performing risk management.

3. Trading signal generation:

In this stage, it generates the portfolio of instruments based on the previous stage analysis resulting in what to trade, how much to trade and when to trade.

4. Trade execution:

In this stage, the trade may or may not be executed via automation because of the constraining factors such as liquidity of market and size of order. Thus, it tries to execute the trade in such a way that impact on market and risk of timing is minimized.

5. Post-trade analysis:

In this stage, it analyzes the results of the trade. It calculates the difference between the price when a buy/sell decision was made and the final execution price. This is important because based on this analysis and results the above models are readjusted for generating more profit.

After learning AT process, I understood that technical analysis is crucial in analyzing how the stock market or a particular stock is moving. For the 297 part of the project, I have implemented the following technical indicators:

Daily Returns: This is the percentage difference of Adjusted Close Price of i-th day compared to (i-1)-th day:

$$Return_i = \frac{AdjClose_i - AdjClose_{i-1}}{AdjClose_{i-1}}$$

I have calculated the daily returns on the adjusted closing price of Apple's (Stock symbol: AAPL) stock data. I have pooled historical stock data from 01-01-2014 to 09-12-2016 from Yahoo Finance. To calculate this, I have used the pct_change() function of Pandas, a Python library module. Pandas is a data analysis library. It offers data structures and operations for manipulating numerical tables and time series. pct_change() is a percent change over a period of time t.

This gives us the percent change over a given number of periods. The output can be seen in the following screen shot. The s_returns indicate the stock returns based on adjacent close (s_adjclose) value of the Apple stock.

```
                m_adjclose  s_adjclose  s_returns  m_returns
Date
2016-09-06  2186.479980  107.149865        NaN        NaN
2016-09-07  2186.159912  107.806498   0.006128  -0.000146
2016-09-08  2181.300049  104.981001  -0.026209  -0.002223
2016-09-09  2127.810059  102.603209  -0.022650  -0.024522
2016-09-12  2159.040039  104.901415   0.022399   0.014677
2016-09-13  2127.020020  107.398588   0.023805  -0.014831
2016-09-14  2125.770020  111.199076   0.035387  -0.000588
```

Volatility: Volatility is the standard deviation of the returns of the stock. It gives us a sense of how much the stock returns fluctuate and how risky it is in comparison to the overall market return.

I am calculating volatility by using the covariance matrix. The first element of a matrix will give us the volatility. Thus, it gives a sense of how risky the stock is compared to market return.

Covariance matrix: Covariance indicates the level to which two variables vary together. The formula to calculate covariance matrix is:

$$\text{Covariance} = \frac{\sum (\text{Return}_{ABC} - \text{Average}_{ABC}) * (\text{Return}_{XYZ} - \text{Average}_{XYZ})}{(\text{Sample Size}) - 1}$$

I have used cov() function of Numpy library. Numpy is a fundamental package for scientific computing.

Beta: Beta of a stock is a measure of the relative risk of the stock with respect to the market. It indicates whether the stock is more or less volatile than the market.

If the beta value is greater than 1 means that the stock returns amplify the market returns on both the upside and downside. When the beta value is less, then 1 means that the stock returns are subdued in comparison to the market returns. I have used the following formula to calculate the value of Beta.

$$\beta = \frac{\text{Cov}(r_a, r_b)}{\text{Var}(r_b)}$$

Momentum: Momentum is a measure of the past returns over a certain period. For example, the 1-year momentum will be the 1-year return of the stock. It measures the rate of the rise or fall in stock prices. It is a very useful indicator of the strengths and weaknesses of a stock. The formula for calculating momentum is as follows.

$$momentum = close_{today} - close_{N\ days\ ago}$$

Alpha: Alpha of a stock gives you a measure of the excess return with respect to the market. Positive alphas for a stock or portfolio gives you a sense of how well your stock outperformed a market. Alpha is a historical measure of the stock returns on investment compared to the risk adjusted expected return.

RESULT:

The output of Deliverable 1 for all the above-mentioned technical indicators is as follows.

```
The volaility of a stock is : 0.044698
The beta value of a stock is : 0.746625
The alpha value of a stock is : 0.007397
The momentum of a stock is : 0.019231
```

DELIVERABLE 2

For this deliverable, I have built a neural network, which can tell the starting month of a given labeled data series. For example, given a series of sales number of iPhone labeled each with month for analyzing, one should be able to tell what could be the month of the first sales point in a given similar dataset (size=10) without labels.

My neural network in this deliverable will be doing the same thing. For this problem, I used an employment dataset of goods producing industry. I downloaded this dataset from the Quandl website. The dataset contains two columns of dates and employment values. The neural net will tell us the month of the first employment data value of given test set.

A neural network is a programming paradigm that enables a computer to learn from observational data. It can recognize complex patterns very easily. Usually, a neural network gets a large number of input and expected output for that input. It then searches for the relations between input and output. Once these patterns are identified then the neural network can produce output on any additional given dataset.

In stock market, a lot of information is produced in a short period of time. The analysis and understanding of such a vast data in limited time is not feasible for humans. Thus, neural network can be a suitable approach for it. To test this, I am building a convolutional neural network for a small dataset.

Convolutional neural network is a feed forward neural network. It is inspired by how the brain processes visual input by dividing the signals into smaller areas called receptive fields. Similarly, Convolutional layers exploit the spatial relationship between groups of data points and generate the feature map based on them [6]. These layers take the small size of input, multiply it with weights and add biases. This is called as an activation function. The Convolutional neural network uses it from raw input data to output class values.

For developing such a convolutional neural net, I am using Keras Python library, which is built on top of the Tensorflow library. Tensorflow is an open source software library for numerical computation using data flow graphs. Keras is a high-level neural network library that facilitates fast and easy prototyping for building a neural network.

The employment dataset is a one-dimensional input vector. There are total of 924 data points in a dataset. I preprocessed the data to reduce the values to range from 0 to 1 and divided it into a test set and a train set. As I am predicting the month, my target variable

is a value from 1 to 12. To transform it into a binary matrix, I used one hot encoding on the target variable. The basic architecture for my convolutional neural net is as shown in the following figure:

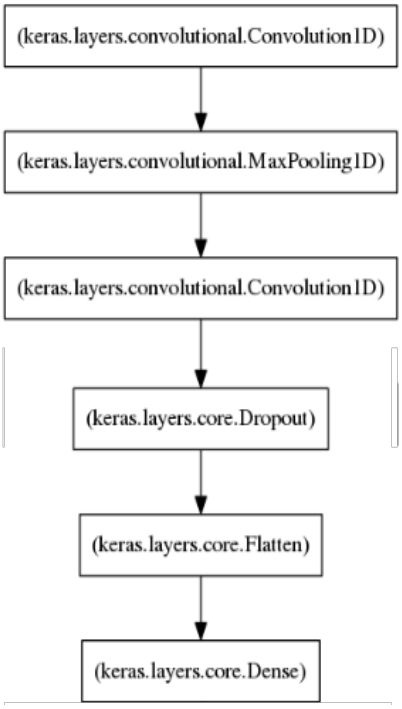


Figure 2: CNN architecture

My first layer of neural net is a convolutional layer. This layer takes the input in the shape of (12, 1). From this it generates 18 feature maps of stride 1. The feature length is of size 3. I used rectifier activation function. After this, I used a maxpool layer with the pool size of 2 and following this, I again used a convolutional layer. This generates 6-feature map of size 2. Finally, I used a dropout layer followed by a dense layer to generate the output. The output layer contains 12 classes. The dense layer used a softmax activation function to output probability-like predictions for each class.

RESULT:

The output for this convolutional neural network is as follows.

```
Epoch 299/300
0s - loss: 0.0215 - acc: 0.8333 - val_loss: 0.0599 - val_acc: 0.3778
Epoch 300/300
0s - loss: 0.0237 - acc: 0.8049 - val_loss: 0.0598 - val_acc: 0.3778
12/12 [==============================] - 0s
[0.03948664665222168, 0.75]
12/12 [==============================] - 0s
[ 8  7  6  5  4  3  2  1  1 12 12  9]
```

For this problem, I trained my neural network for 300 epochs. In machine-learning parlance, an epoch is a complete pass through a given dataset. That is, by the end of one epoch, my neural network, in this case - a convolutional net, will have been exposed to every record to example within the dataset once. After that I achieved 75% accuracy. The numbers in the last array indicate the output classification for a starting month. The expected output was [8, 7, 6, 5, 4, 3, 2, 1, 12, 11, 10, 9].

To evaluate whether my neural net is performing well in this problem, I decided to test the same problem with 3 people. The result is compiled in the following table.

	Predicted Result	Actual Result
Person 1	[10,9]	[9,8]
Person 2	[7,8]	[5,4]
Person 3	[10,9]	[12,11]

Table 1 : Human Experiment

As shown in the Table 1, the closest answer given has a difference of one month in actual results and predicted results. Thus, the results given by the neural net are certainly more accurate than humans.

DELIVERABLE 3

The third deliverable was to develop a program to predict the closing price of a stock using the historical data of the same stock. For example, stock data for CISCO Systems is given from the year 2010 till today. Based on this data, a program has to predict what will be the closing price of CISCO today.

In stock market prediction, we predict the price of a particular stock. As price is a continuous variable, the regression method works better. Because if we compare to stock trends, the exact increment in stock index may provide more information for building a prediction model.

And similarly, in general humans tend to do similar trading. They try to buy or sell the stock based on the past performance of the stock. People change their strategy only when there is some major current news that influences the stock market.

For this problem, I used the Decision Tree and K Nearest Neighbors (KNN) machine learning techniques, because they can capture the past data and make a reliable prediction based on it. Thus, decision trees and KNN regression techniques are good choices.

Decision trees split the data into small groups based on maximizing information gain. Thus, these groups capture the past data and precisely predict data that are similar to the past data. K-nearest neighbors predict the result using k-nearest data points from the past data. Hence, it also can capture the past data and assume similar data points will have similar results.

For implementing the above approach, I used Scikit-learn Python library. Scikit-learn is a special library for machine learning. It provides simple and efficient tools for data mining.

Initially, I queried the stock data from Quandl based on the read ticker symbol and user inputted dates. I used only 'Open,' 'High,' and 'Volume' features of the retrieved dataset. I also asked the user to input the prediction date for which they want the prediction.

If there are no given prediction dates, then query the most recent data from Yahoo Finance. I divided the above retrieved dataset using cross-validation to create a train and test data set by splitting the data in 75:25 ratio.

I used ensemble bagging regressor method for both decision tree and KNN regressor. A bagging regressor is a method of combining multiple predictors and then using an averaging or voting method to get the value.

Therefore, to find the best parameters, I used a grid search method. Using those parameters train the data and calculate the root mean squared error with the test data and predict the value for the test data.

RESULTS:

1. Using Decision Tree:

Following is the output for the Facebook stock price using Decision Tree. Here prediction dates were not provided. Data is trained for data from Jan 2012 till Dec 2016. The actual closing prices are FB: 120.57, CSCO: 30.63, AAPL: 115.82

```
Please enter start date for data collection in yyyy-mm-dd format : 2012-01-02
Please enter end date for data collection in yyyy-mm-dd format : 2016-12-14
Please enter prediction dates in yyyy-mm-dd format separated by spaces :
Please enter 1 for DecisionTreeRegressor or 2 for KNeighborsRegressor to predict ticker value: 1

Training for FB...
BaggingRegressor(base_estimator=DecisionTreeRegressor(criterion='mse', max_depth=20, max_features=None,
          max_leaf_nodes=None, min_impurity_split=1e-07,
          min_samples_leaf=1, min_samples_split=2,
          min_weight_fraction_leaf=0.0, presort=False, random_state=None,
          splitter='best'),
         bootstrap=True, bootstrap_features=False, max_features=1.0,
         max_samples=1.0, n_estimators=50, n_jobs=1, oob_score=False,
         random_state=0, verbose=0, warm_start=False)
AVG score is  0.999371402343
This is the result for FB:
Predicted adjusted close value for today is 121.03

Training for CSCO...
BaggingRegressor(base_estimator=DecisionTreeRegressor(criterion='mse', max_depth=5, max_features=None,
          max_leaf_nodes=None, min_impurity_split=1e-07,
          min_samples_leaf=1, min_samples_split=2,
          min_weight_fraction_leaf=0.0, presort=False, random_state=None,
          splitter='best'),
         bootstrap=True, bootstrap_features=False, max_features=1.0,
         max_samples=1.0, n_estimators=50, n_jobs=1, oob_score=False,
         random_state=0, verbose=0, warm_start=False)
AVG score is  0.989417136863
This is the result for CSCO:
Predicted adjusted close value for today is 30.50

Training for AAPL...
BaggingRegressor(base_estimator=DecisionTreeRegressor(criterion='mse', max_depth=10, max_features=None,
          max_leaf_nodes=None, min_impurity_split=1e-07,
          min_samples_leaf=1, min_samples_split=2,
          min_weight_fraction_leaf=0.0, presort=False, random_state=None,
          splitter='best'),
         bootstrap=True, bootstrap_features=False, max_features=1.0,
         max_samples=1.0, n_estimators=50, n_jobs=1, oob_score=False,
         random_state=0, verbose=0, warm_start=False)
AVG score is  0.995627852153
This is the result for AAPL:
Predicted adjusted close value for today is 112.65
```

2. Using KNN:

The following screen shot shows the predicted and actual values for stocks of Facebook, Cisco and Apple. Here prediction date is provided. The dataset is from Jan. 2016 till Dec. 2016.

```
Training for FB...
BaggingRegressor(base_estimator=KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkowski',
          metric_params=None, n_jobs=1, n_neighbors=15, p=2,
          weights='uniform'),
         bootstrap=True, bootstrap_features=False, max_features=1.0,
         max_samples=1.0, n_estimators=50, n_jobs=1, oob_score=False,
         random_state=0, verbose=0, warm_start=False)
AVG score is  0.320832437726
This is the result for FB:
Training data are from 2016-01-01 to 2016-12-08
Predicted adjusted close value for 2016-12-09 is 119.7534
Actual adjusted close value for 2016-12-09 is 119.6800

Training for CSCO...
BaggingRegressor(base_estimator=KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkowski',
          metric_params=None, n_jobs=1, n_neighbors=15, p=2,
          weights='uniform'),
         bootstrap=True, bootstrap_features=False, max_features=1.0,
         max_samples=1.0, n_estimators=50, n_jobs=1, oob_score=False,
         random_state=0, verbose=0, warm_start=False)
AVG score is  0.271023825754
This is the result for CSCO:
Training data are from 2016-01-01 to 2016-12-08
Predicted adjusted close value for 2016-12-09 is 29.1343
Actual adjusted close value for 2016-12-09 is 30.0600

Training for AAPL...
BaggingRegressor(base_estimator=KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkowski',
          metric_params=None, n_jobs=1, n_neighbors=15, p=2,
          weights='uniform'),
         bootstrap=True, bootstrap_features=False, max_features=1.0,
         max_samples=1.0, n_estimators=50, n_jobs=1, oob_score=False,
         random_state=0, verbose=0, warm_start=False)
AVG score is  0.0454476803948
This is the result for AAPL:
Training data are from 2016-01-01 to 2016-12-08
Predicted adjusted close value for 2016-12-09 is 102.3748
Actual adjusted close value for 2016-12-09 is 113.9500
```

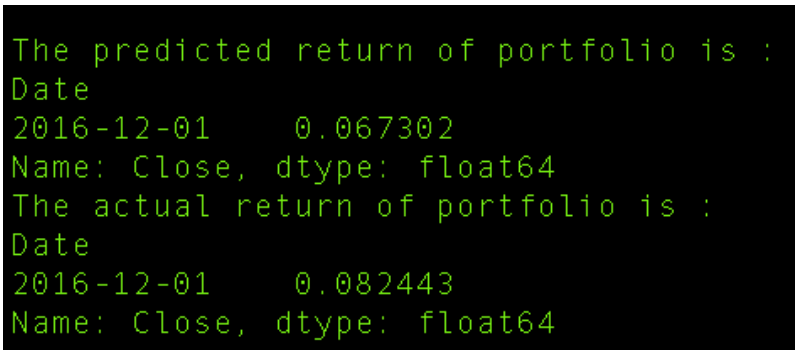
DELIVERABLE 4

For this deliverable, I predicted return portfolio of a stock. When a person invests in more than one stock then it creates the portfolio of a stock. People buy, sell, or hold the portfolio based on profit or loss generated from it. In this deliverable, I calculated profit or loss of a portfolio. This deliverable is an extension of Deliverable 3.

I calculated return of stock based on predicted close price for individual stock, and later, summed up all the returns of a stock. If the resulting value is positive, then portfolio is generating profit.

RESULT:

The portfolio of a stock contains Facebook, Cisco and Apple shares. Here, suppose I am investing on Dec. 01, 2016, then my predicted return of portfolio for Dec. 12 is around 7% as follows.



CONCLUSION

I have completed the basic implementation part of my project. I am now aware of what parts I need to emphasize more for CS 298.

During the span of CS 297, I started by gaining an understanding of how the stock prediction system works, and what are the important features that affect the trending of a stock. Stock prediction is not a straight problem. There are various patterns in the data and all the features of a prediction model depend on each other.

In CS 297, I explored all these dependencies of data by using different types of approaches like neural net, decision tree and k-nearest neighbors. I also performed technical analyses on data set in order to measure the return, risk of investing, and performance of stock factors.

In CS 298, I will merge these features with other approaches to improve their prediction capabilities. I will also add the fundamental analysis module, which will perform the analysis of a company for which we want to predict the stock price. For that, I need to study in depth the various economic indicators. I will also explore the other machine learning techniques for stock prediction.

REFERENCES

1. Read more: Basics of Algorithmic Trading: Concepts and Examples | Investopedia http://www.investopedia.com/articles/active- trading/101014/basics-algorithmic-trading-concepts-and- examples.asp#ixzz4PA8nb8Kq
2. Philip, T., et all, algorithmic Trading Review
3. Barry Johnson - Algorithmic Trading & DMA
4. Perry Kaufman - Trading Systems and Methods (5th ed) (2013)