**Script Programming/Python**

**PROJECT REPORT**

**Project Name: TradingWiZ**

**Date: 11-29-2017**



**Group Members:**

**David Guerrero**

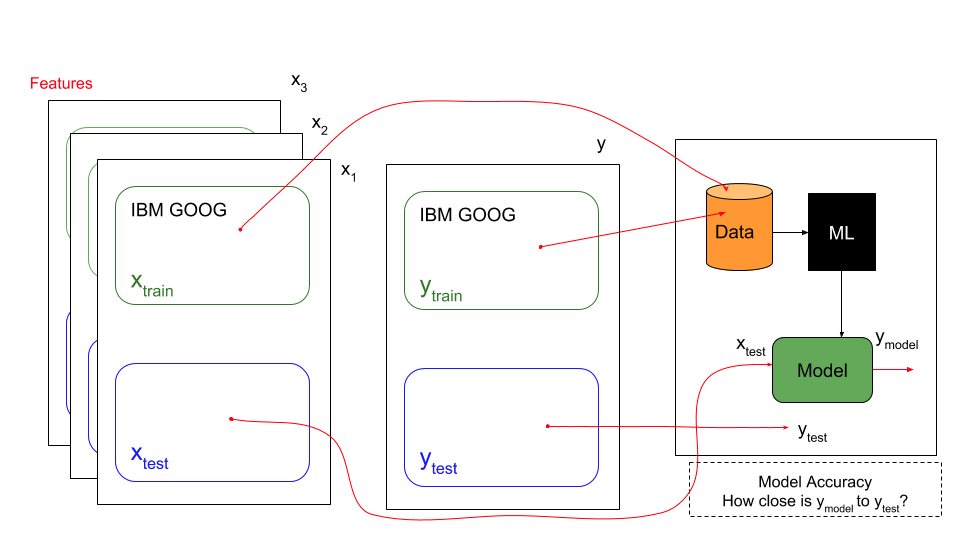
**Shubham Jaiswal**

**Submitted to:** Frank G Breitinger

**TradingWiz**

**Problem Statement and Solution:**

Trading algorithms are an efficient method of computing technical analysis of financial stock data within a time series. The purpose of Trading Wiz is to implement a Machine Learning (ML) trading algorithm that trains and tests a probabilistic mathematical model using historical stock data.



**Fig1.** Trading Wiz Architecture

The overall application takes a set of historical stock data xtrain and ytrain ,and runs the data through a Standard Average and Exponential Moving Average algorithm. This generates a model that can be analysed to verify the accuracy of the algorithms. Variable xtrain is a multi-dimensional representation of different features of certain ticker symbols (AAPL, GOOG, AMZN). These features can be a price change, momentum, or a median price. ytrain is a set of stock price data for a given point of time.

The accuracy of the model can be tested by inputting a set of xtest data that produces a ymodel. The closer this ymodel is to ytest the more accurate the model is. The data being used is based on linear time. Older data is utilized to train and generate a model, the more recent data is used to test the model.

**Architecture of Application**

1. TradingWiz

2. DataImport

3. LinRegLeaner

4. PlotResults

**1. TradingWiz**

TradingWiz is the main module. It calls the Learning module to predict prices of stocks using a Standard Average algorithm and an Exponential Moving Average. It feeds data to the Plotting module to display the graphs for each of the training algorithms overlaid with the medium data.

**2. DataImport**

This module is implemented for importing stock data using the Pandas Data Reader which utilizes the Yahoo Finance historical stock database. Specific features can be imported as well as a full set of data. Uses pandas\_datareader to pull a feature for a ticker (ie. GOOG, AAPL, GE) from a start date to an end date. A feature can be Open Price, High Price, Low Price, Closing Price, Adjusted Close Price, and Volume. This will return a Pandas DataFrame.

**3. LinRegLearner**

This module computes the mid\_price, then divides it into training and testing data. It scales and fits the train and test data accordingly. Also, smooths out train data to remove noise. This returns a tuple of scaled/smoothed trainData & scaled testData. This trainData and testData is combined. Dates are extracted from a dataFrame of stock data; it is used to compute a Standard Average and an Exponential Moving Average.

Calculates a standard average of historical stock data that was previously observed. This calculated average is the predicted future stock market prices within a fixed window size. Window Size represents a window of 100 points of data. The prediction of a day ahead is the average of all the stock prices 100 days previous to the current day.

Calculates an Exponential Moving Average (EMA) maintained over time of historical stock data. Decay decides how relevant the most current prediction is towards the EMA. In our case, it will be 0.5.

**4. PlotResults**

This module generates 3 plots. The first plot is the raw stock medium price for the full range of start to end. The second generated plot overlays the actual stock price and the predicted stock price using a standard mean algorithm. The third plot overlays the actual stock price and the predicted stock price when calculating the exponential moving average

**Difficulties faced:**

1. Learning to work with GitHub. Watched video tutorials to learn the process.

2. Issues with installing the python libraries like Numpy and Pandas but later we downloaded Anaconda which comes with the built-in libraries.

3. Modularizing the code. We established the required classes to deploy a working application. Then, organized the functions that work together and are dependent on each other. Finally, grouped them within the same class. Developed a main module that runs the application.

4. Finding an algorithm that was accurate enough for the application while having an easy to understand mathematical expression