Software Requirements Specification

for

Stock Predictor

Version 1.0 Approved

Prepared By: -

Mohammad Abedin [ 1722014 ]

Manipal Academy of Higher Education, Dubai

5/17/2020

Table of Contents

Table of Contents ii

1. Introduction 3

1.1 Purpose 3

1.2 Document Conventions 3

1.3 Intended Audience and Reading Suggestions 3

1.4 Project Scope 4

1.5 References 4

2. Overall Description 4

2.1 Product Perspective 4

2.2 Product Features 5

2.3 User Classes and Characteristics 5

2.4 Operating Environment 5

2.5 Design and Implementation Constraints 6

2.6 User Documentation 7

2.7 Assumptions and Dependencies 7

3. System Features 8

3.1 functional requirements **Error! Bookmark not defined.**

4. External Interface Requirements 13

4.1 User Interfaces 13

4.2 Hardware Interfaces 13

4.3 Software Interfaces 14

4.4 Communications Interfaces 14

5. Other Nonfunctional Requirements 14

5.1 Performance Requirements 14

5.2 Safety Requirements 15

5.3 Security Requirements 15

5.4 Software Quality Attributes 15

# Introduction

## Purpose

Predicting the Stock Market has been the bane and goal of investors since its existence. Everyday billions of dollars are traded on the exchange, and behind each dollar is an investor hoping to profit in one way or another. Entire companies rise and fall daily based on the behavior of the market. Should an investor be able to accurately predict market movements, it offers a tantalizing promise of wealth and influence. It is no wonder then that the Stock Market and its associated challenges find their way into the public imagination every time it misbehaves. The 2008 financial crisis was no different, as evidenced by the flood of films and documentaries based on the crash. If there was a common theme among those productions, it was that few people knew how the market worked or reacted. Perhaps a better understanding of stock market prediction might help in the case of similar events in the future.

## Document Conventions

This document uses the following conventions:

|  |  |
| --- | --- |
| DB | Database |
| DDB | Distributed Database |
| OS | Operating System |
| RBF | Radial Basis Function Kernel |

## Intended Audience and Reading Suggestions

**This Document is intended for Developers, Project Managers and Testers**.

## Project Scope

A chief goal of this project is to add to the academic understanding of stock market prediction. The hope is that with a greater understanding of how the market moves, investors will be better equipped to prevent another financial crisis. The project will evaluate some existing strategies from a rigorous scientific perspective and provide a quantitative evaluation of new strategies.

It is important here to define the scope of the project. Although vital to any investor operating in the real world, no attempt is made in this project at portfolio management. Portfolio management is largely an extra step done after an investor has made a prediction on which direction any stock will move. The investor may choose to allocate funds across a range of stocks in such a way to minimize his or her risk

## References

* Basic and fundamentals of Python: <https://www.geeksforgeeks.org/python-programming-language/>
* Linear Regression: <https://towardsdatascience.com/a-beginners-guide-to-linear-regression-in-python-with-scikit-learn-83a8f7ae2b4f>
* Support Vector Machines: <https://scikit-learn.org/stable/modules/svm.html>
* Basics of Pandas DataFrames: <https://www.learnpython.org/en/Pandas_Basics>
* Basics of NumPy Library: <https://www.tutorialspoint.com/numpy/index.htm>
* PyPlot Library from Matplotlib: <https://matplotlib.org/tutorials/introductory/pyplot.html>

# Overall Description

## Product Perspective

In the past decades, there is an increasing interest in predicting markets among economists, policymakers, academics and market makers. The objective of the proposed work is to study and improve the supervised learning algorithms to predict the stock price.

Technical Objective:

The technical objectives will be implemented in Python. The system must be able to access a list of historical prices. It must calculate the estimated price of stock based on the historical data. It must also provide an instantaneous visualization of the market index.

Experimental Objective:

Prediction system will be implemented using Support Vector Machines. The experimental objective will be to compare the forecasting ability of SVM and Linear Regression Models. We will test and evaluate both the systems with same test data to find their prediction accuracy.

## Product Features

The main feature of the described software is to deliver the predicted stock prices of the user desired company. The user enters the company name and its symbol in the input region and initiates the process and software communicates to the yahoo finance server and gather the data of the previous years of the company and places as an input to the algorithm and the algorithm will initiates its process, once the values are obtained then it displays to the user.

## User Classes and Characteristics

* Stock Market Investors
* Stock Market Researchers

## Operating Environment

The current Operating Environment is Windows and due to the simplicity of the programs and resource management, the program may be available in other platforms like MacOS, iOS, Android, etc.

The scripting language is used in this program is purely restricted to python and no other language dependencies has been used in this project.

## Design and Implementation Constraints

***Throughout the project, there are a few ideas that warn us that we might not find a profitable way to predict market trends.***

Random Walk Hypothesis

The random walk hypothesis sets out the bleakest view of the predictability of the stock market. The hypothesis says that the market price of a stock is essentially random. The hypothesis implies that any attempt to predict the stock market will inevitably fail.

Efficient Market Hypothesis

Another concept to keep in mind while working on the project was the Efficient Market Hypothesis. Informally, the Efficient Market Hypothesis says that the market is efficient at finding the correct price for the stock market. It comes in three flavors. However, it is still a matter of debate which one, if any, is correct:

* Weak-form Efficient Market Hypothesis: The weak form of the hypothesis says that no one can profit from the stock market by looking at trends and patterns within the price of a product itself. It is important to note that this does not rule out profiting from predictions of the price of a product based on data external to the price.
* Semi-Strong Efficient Market Hypothesis: The semi-strong form rules out all methods of prediction, except for insider trading. This means that if we are only to use public domain information in our prediction attempt, the semi-strong form says that we will be unsuccessful.
* Strong form Efficient Market Hypothesis: The strong form says that no one can profit from predicting the market, not even insider traders.

Clearly, if we are to predict the stock market using only public information, we must hope that at most the weak form of the Efficient Market Hypothesis is true so that at least then we can use external data to predict the price of a product.

## User Documentation

Although the data gathered in the previous section is certainly a good start, it is admittedly far behind what any serious investor more than likely has access to. One obvious piece of missing data that is the intraday prices, i.e. the prices minute by minute. It is possible that this data could be used to guide and investors decisions on the interday level. However, intraday prices are not as freely available as interday prices and are considered a commodity in themselves. To get hold of such a dataset would incur a large cost, one that is not within the budget of a project such as this. Later in the project we will evaluate a strategy in which this limitation becomes significant. Another important piece of missing data is the order book. The order book is a record of live buy and sell orders for a stock. It consists of the amount of stock each trader is willing to buy or sell, as well as their price. Successful orders are matched off against the order book by the exchange. The price of a stock is usually considered to be halfway between the highest buying price and the lowest selling price. It is easy to imagine that the order book contains useful data. For instance, the weighted average of orders might be predictive of the price. However, access to this data is extremely costly and far beyond what most casual investors can afford, let alone the budget for this project. With no way around these limitations, we use the data provided by Yahoo Finance.

## Assumptions and Dependencies

Python and Associated Packages

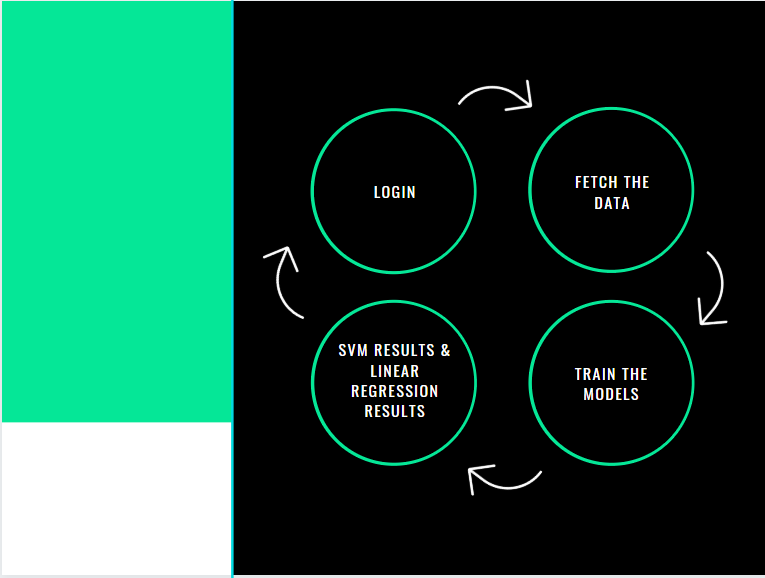
Python was the language of choice for this project. This was an easy decision for the multiple reasons:

* Python as a language has an enormous community behind it. Any problems that might be encountered can be easily solved with a trip to Stack Overflow. Python is among the most popular languages on the site which makes it very likely there will be a direct answer to any query.

# Python has an abundance of powerful tools ready for scientific computing. Packages such as NumPy, Pandas, and SciPy are freely available, performant, and well documented. Packages such as these can dramatically reduce, and simplify the code needed to write a given program. This makes iteration quick.

* Python as a language is forgiving and allows for programs that look like pseudo code. This is useful when pseudo code given in academic papers needs to be implemented and tested. Using Python, this step is usually reasonably trivial.

# System Features



## Login

### Description and Priority

Priority Level: High

First, Users need to login to use the system. Only the very basic user information like the username & password will be collected by the system to use the services of the program.

### Stimulus/Response Sequences

* Input the desired username
* Input the desired password following program standards.
* Enter the Date of Birth of the User.

## Fetch the Data

### Description and Priority

* Priority: High

Before training the models, we must prepare the data to feed them. Because no thresholds had to be defined, this step was relatively straight forward. The primary price database originally collected from Yahoo Finance was used for this experiment. For each company, its price history was iterated over the previous year’s prices.

The results of these calculations were stored in a temporary collection.

## Train the Models

### Description and Priority

* Priority: High

Four Python scripts are written to transform the raw stock prices (.csv files) into feature vectors, for training, predicting and testing, respectively. The scripts take the input options and the raw stock prices as inputs and produce the correct features by building the lookback arrays and the moving averages. It concatenates the features into the final feature vectors, which will be passed to the model for training or testing. The 4 scripts share common operations in building a dataset except the output size and the range of dates to build from, so common functions are written to centralize the logic instead of repeating the same index-calculation-intensive work across functions.

* + Radial Basis Function Kernel:
    - In machine learning, the radial basis function kernel, or RBF kernel, is a popular kernel function used in various kernelized learning algorithms. It is commonly used in support vector machine classification.
  + Linear Kernel:
    - Linear Kernel is used when the data is Linearly separable, that is, it can be separated using a single Line. It is one of the most common kernels to be used. It is mostly used when there are a Large number of Features in a Data Set. One of the examples where there are a lot of features, is Text Classification, as each alphabet is a new feature. So, we mostly use Linear Kernel in Text Classification.
  + Polynomial Kernel:
    - In machine learning, the polynomial kernel is a kernel function commonly used with support vector machines (SVMs) and other kernelized models, that represents the similarity of vectors in a feature space over polynomials of the original variables, allowing learning of non-linear models.
    - Intuitively, the polynomial kernel looks not only at the given features of input samples to determine their similarity, but also combinations of these. In the context of regression analysis, such combinations are known as interaction features. The (implicit) feature space of a polynomial kernel is equivalent to that of polynomial regression, but without the combinatorial blowup in the number of parameters to be learned. When the input features are binary-valued (Booleans), then the features correspond to logical conjunctions of input features.
  + Linear Regression:
    - Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output).

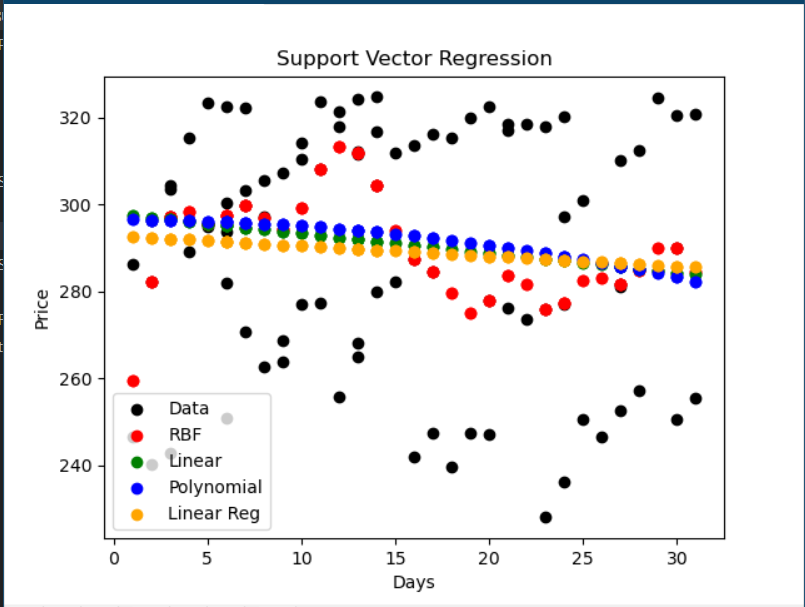
All these models are collectively used and trained according to the data collected by Yahoo Finance. The data is then fed into these models and are calculated based on its gamma value and Error constants while keeping the whole data in memory giving out the next day prediction prices.

## PyPlot: Plotting the Models

* Priority: Medium
  + matplotlib. pyplot is a collection of command style functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc.
  + In matplotlib. pyplot various states are preserved across function calls, so that it keeps track of things like the current figure and plotting area, and the plotting functions are directed to the current axes.

## Display the Models & Output

* Sample Chart Model is shown below:



As we can see, The plotted data of the RBF, Linear & Polynomial Kernels of the SVM trained models along with the Linear Regression Model.

The best result out of these 4 Machine Learning models will be displayed separately along with the values each model has given out.

# External Interface Requirements

## User Interfaces

### PyQt5

* + PyQt is a library that lets you use the Qt GUI framework from Python. Qt itself is written in C++. By using it from Python, you can build applications much more quickly while not sacrificing much of the speed of C++.
* The Graphical User Interface consists of:
  + Input Boxes
  + Dialogs
  + Alerts
  + Map Display
  + Login Panel
  + Chart Panel

## Hardware Interfaces

* Storage: 100MB or more
* RAM: 1GB or more
* Processor: Intel i3 5th Gen or higher
* Graphics: Intel Graphics or Nvidia

## Software Interfaces

* Python – Version 3.8.2
* Pandas – Version 1.0.3
* NumPy – Version 1.18.4
* Pandas-Datareader – Version 1.0.0
* pyQt5 – Version 5.3.1
* scikit\_learn – Version 0.22.2
* requests – Version 2.23.0
* matplotlib. pyplot – Version 3.2.1

## Communications Interfaces

There are relatively many communications taking place throughout the program:

* For starters, the Login Panel has its own database connection for manipulating data and validation of user accounts.
* The Connection between the program and Yahoo Finance establishes using the Request Library from python.

# Other Nonfunctional Requirements

## Performance Requirements

There are certain requirements for the software to run.

* The user must have a minimum of 1gb ram or more.
* The user must have an Intel i3 Processor & above
* The user must have a minimum 100 MB on the hard disk to avail for the software
* Required Graphics card: Intel HD Graphics or Nvidia GeForce

## Safety Requirements

Due to the simplicity of the software there will not be any data loss

## Security Requirements

Due to the simplicity of the software there will not be any security issues present.

## Software Quality Attributes

Adaptability:

* The program is well constructed and tuned in such a way that the database and the models can process bulk amount of data considering the system resources.

Reusability:

* SVM Models and the data collected from Yahoo Finance are reusable several times during the program and the users can see through the data too.

Reliability:

* The Support Vector Machines and Database used in this program are stable and updated thus eliminating any bugs causing any program crashes.

Correctness:

* The program shows predicted values and shows no correctness whatsoever, as it depends on the data history provided and thus does not provide exact values for price prediction.

Maintainability:

* The program uses all latest python language and libraries for compilation and usability.

The program availability is subject to the data available on Yahoo Finance. If the data is not found, the program is rendered nonfunctional since the models need to be trained based on the data fed into them. Availability: