

CSI 4999

**Senior Project**

**Stock Watch**

## 

## **Project Vision**

This documentation is designed to describe steps taken from us during the development of our software product “StockWatch”. This software is an implementation of a stock market prediction algorithm. This algorithm predicts stock moves and make and action by either long, short, stay a stock to maximize profit. Our goal through this project was to build a unified model using known strategies and provide guidance about investment return of a given stock based on various buy and sell points predicted by the system.

## **Project Execution and Planning**

**Team Information** - For this project called StockWatch, we are split into two groups. One will work with machine languages and another will work on our webpage. The webpage is the front end and the machine language group is the back end.

**Tools and technology**

* Python
* Flask Framework
* SQL
* Html & CSS
* Machine learning language
* Pythonanywhere server
* Github
* Trello
* Matplot
* Visual Studio

**Project Plan -** Build a software that can predict the price of the stock market through the help of historical data and algorithm calculation. The software will provide guidance to the users in the stock market business to help users with their investment.

**Best standards and Practices -**

Using configuration management on the source code. This is done so that everyone is updated and no one person is control of the entire project. Using trello as a scheduling tool so that everyone has a shared responsibility in the project. Setting up group meetings to assign tasks between group members. We also created a mitigan plan in the initial phase of the project.

**Mitigation Plan**

|  |  |  |
| --- | --- | --- |
| **Risk #** | **Risk** | **Mitigation** |
| **1** | **Inexperience with Python** | **Every group member must be prepared to put in 40 to 60 hours the first week in order to refresh themselves on this particular programming language.** |
| **2** | **Unfamiliar with the stock market** | **Set aside at least 10 hours to watch YouTube videos on how the stock market works and to look for examples of predictive software already in the marketplace** |
| **3** | **Scheduling Problem** | **Have group decisions the first week and make a concrete time to meet via in person or by skype** |
| **4** | **Some group members are not strong on the backend of the project** | **Group uses a collaborative effort to support and help each other in the areas that they are lacking** |

**System Requirement Analysis**

Function Requirements

* Front end (html based)
  + Outgoing links can direct to another webpage
  + Internal links can allow navigation from one page in a webpage to another
  + Have anchor links to make navigating easier
  + Make sure default values are being populated
  + Forms are formatted clearly and readable
  + Checking to make sure the right error message is shown

## **Functional Requirements Specification**

**User Stories -** Build a user login/authentication which stores password in a secure database and keeps personal lists of stocks

|  |  |  |  |
| --- | --- | --- | --- |
| User account creation | | | |
| US# | Role | Action | Goal |
| 1 | Unregistered user | As a first-time user I want to be able to register on the website | Secure My Account |
| 2 | Unregistered user | As a first-time user I want to have a unique username | Customize Account |
| 3 | Unregistered user | As a first-time user I want to have a secure password | Secure My Account |
| 4 | Registered User | As a user I want to be able to access my account at anytime | Availability |

|  |  |  |  |
| --- | --- | --- | --- |
| Role Management | | | |
| US# | Role | Action | Goal |
| 1 | Administration | As the admin I want to access the database so that I can view all current emails | Participate in communication for registered users |
| 2 | Administration | As the admin I want site wide access | I can manually resolve any issues |
| 3 | Administration | As the admin I want to access the database | To delete, update user’s information |
| 4 |  | As the admin I want to create other administrators | I can delegate task |

## **User Interface Specifications**

**User Interface Requirements -** Build a website to display graphs and provide stock market guidance in a user friendly manner.

* Build a drop-down list the user should be able to select which stock to run the experiment on.
* Build a drop-down list the user should be able to select which algorithm to run the experiment on.

**ML Requirements -** Implement a new prediction model for predicting the future price of a stock.

* We shall use Keras to build a neural network model.
* We shall build an algorithm to convert data sets to a data set acceptable by the model
* The model shall receive training, testing, and validation data set from the data directory.
* The model shall calculate reward if made a correct/wrong decision of buying, selling, or no action on a stock.
* The model shall save the last financial position (either long or short)
* The model shall output a graph for reward/punishment function.
* The model shall output a graph for action taken by model (buy, sell, hold).
* The model shall be dynamic in the number of steps taken to train/test itself.
* The model shall be dynamic in the data size taken to train/test itself.

## **Static Design**

**ML Model Design**

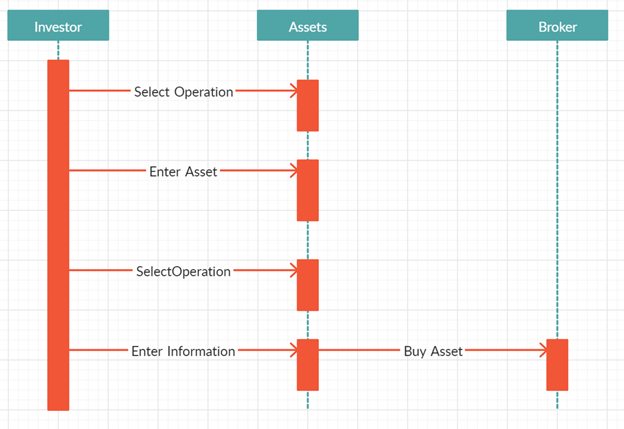
****

* Our data input will be represented by three different data sets
  + Training data set
  + Validation data set
  + Testing data set
* Neural network represents the model in our algorithm
  + We manually set the number of nodes (neurons), batch size, memory.
  + The model receives different data sets to process on (epoch).
  + We make the decision for the model if the data is test data or validation data or training data.
  + As well as the number of steps the program shall take to train/test.
  + For a single experiment we run our agent two times then we reset the variables:
    - Training/validation run: the agent is trained to make the right decisions on a subset of data.
    - Testing run: the agent ability to make the right decisions is tested during this step.
  + After each epoch or at the start of a process (train, test, validation) reset the variables.
* After data is processed an output data is displayed in a form of graph.
  + Reward graph
  + Agent Action graph
* Then the output is tested with real data and a margin of error is returned.
  + The model reward function depend on the error margin.
  + The model will be positively rewarded as the error margin shrinks
  + Error margin will be a new input to the model to improve the model prediction ability.
* The reward function of the agent. Based on his action calculate a pnl and a fee as a result Normalize the reward to a proper range

## **Dynamic Design**

**Sequence Diagram design**

* Users can pick whatever they like for their assets
* Users should be able to determine the price of each asset to see if they are willing to buy or sell

****

## 

## **System Architecture and System Design**

* Designed our project using Flask framework which is a python tool extension
* Used sqlalchemy for database
* Updated our file using file transfer protocol on pythonanywhere which is a hosting site for python projects
* Only hardware that we used were our personal laptops for programming and our phones for communication between group members
* Github and google drive is used for configuration management and storage of documents and code**.**

## **Algorithms and Data Structures**

* The core of this project is the machine learning algorithm which we derived from a research paper called “*Deep Reinforcement Learning for Financial Trading Using Price Trailing”*  The program is provided with a historical dataset, upon which a model is trained to act in an optimal way. In other words, indicating points where buying or selling a stock would likely yield the best results.
* We utilized Q-learning methods in our program, which is a type of Reinforcement Learning. Q-Learning is a model-free algorithm, meaning we don’t have to provide the Agent with a model.
* Instead, the agent builds a model itself based on its reward from previous cycles.
* Since there is so much variation in stock prices, the model-free nature of Q learning is the best way to generate a robust ML program.
* The program is provided with historical stock data from which a model is generated and trained to act in an optimal way
* As the agent progresses through each iteration, its goal is to adjust its position to follow the price trend. After each step, the agent will make a decision on whether to stay on the current trend, follow an upward trend, or follow a downward trend, based on a preset margin.
* The agent is given a positive or negative reward based on whether or not its position falls outside of the margin around the actual price. If the agent is within the margin, it will get a greater reward based on how close it remains to the actual stock price. However, if the agent falls outside of the margin, it will receive negative reward proportional to its distance from the actual price. By implementing rewards in this manner, we are able to generate a robust model that will ignore the noise caused by the small fluctuations in price while still following the general price trend. Additional measures are taken in the form of a small penalty which the agent incurs every time that it changes its trend in the opposite direction. This is intended to prevent the agent from chasing minor or insignificant price fluctuations.
* We generated models for most of our data using 100 episodes, each consisting of 150 steps. At the start of each episode, it starts from a random point in the stock price data and optimizes the model.
* Data is presented to the agent in the form of a numpy array which is generated from a csv containing historical stock market data from selected stocks. The data we used was gathered from Yahoo Finance.

## 

## **User Interface Design**

The design schema for StockWatch follows a typically standard format for all internet stock forecast website. The user will open the webpage. In the upper right corner, there will be a button to sign up or log in for current members. The person will fill in their username and password to create an account. Users can view the purpose of our website and our team members through the About button. Users can view the latest stock news and analysis via the StockNew button. The Buyandsell button allows users to see how the stock market is doing for individual companies and to see our recommended buying or selling times.

## **Testing**

* We implemented the authentication system, user interface & database and did quality assurance test for both
* Create a database schema to store stocks and use the received training data and test data to store it under its proper table
* We tested that the user should be able to select which stock to run and then display its graph
* Created an algorithm to turn the training data and test data received from backend team to numpy format
* Tested our BuyandSell tab to make sure it displays graph correctly

## **Project Management**

## 

## **Github link to Machine Learning Code**

<https://github.com/dosterhoff/StockMarketPrediction>

## 

## **References**

<https://link.springer.com/chapter/10.1007/978-981-13-0761-4_84>

<https://books.google.com/books?hl=en&lr=&id=99FwDwAAQBAJ&oi=fnd&pg=PP1&dq=%22reinforcement+learning%22++for+stock+market+python&ots=HRYRTlEw3n&sig=ju_K_E4d5CWQp0ax0DPOQJcnZhs#v=onepage&q&f=false>

<https://towardsdatascience.com/simple-stock-price-prediction-with-ml-in-python-learners-guide-to-ml-76896910e2ba>

<https://www.analyticsvidhya.com/blog/2018/10/predicting-stock-price-machine-learningnd-deep-learning-techniques-python/>  
  
<https://github.com/Kostis-S-Z/trading-rl/blob/master/README.md>

<https://ieeexplore.ieee.org/document/8683161/authors#authors>