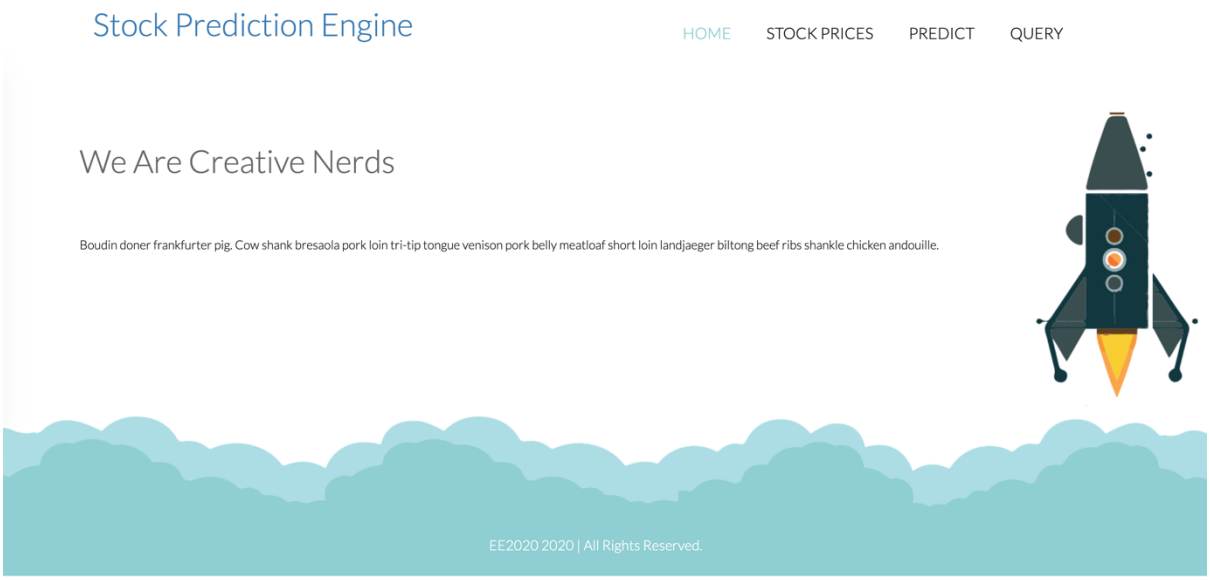
Web-Based Stock Forecaster

Course : Software Engineering Web Application II

Team 7 Group 4



Member Name	NETID	Contribution
Ningyuan Zhang	nz146	Web Application
Weikang Li	wl494	Database
Xueyu Wu	xw318	Algorithm

Table of Contents

1.Customer Statement of Requirements

1.1 Problem Statement

Stock market prediction is the act of trying to determine the future value of a company stock or other financial instrument traded on an exchange. The successful prediction of a stock's future price could yield significant profit. The efficient-market hypothesis suggests that stock prices reflect all currently available information and any price changes that are not based on newly revealed information thus are inherently unpredictable. Others disagree and those with this viewpoint possess myriad methods and technologies which purportedly allow them to gain future price information.

Stock market analysis came into sight because of its ability to analyze trends of stocks and make proper suggestions. Stock market analysis enables investors to, before investing, identify intrinsic value to a stock and make wise decisions. Stock analysis utilizes past and current data to study and evaluate securities and help investors make informed decisions. It's a must to research before making an investment. Combining with news and other source of information, stock market analysis can perform better and more likely to give useful advice.

For large securities companies, it is easier for them to get, analyze and integrate information for they have more resources more powerful machines to make predictions. For individual traders, they have to rely on more basic types of analysis. There are two types of stock analysis: fundamental analysis and technical analysis. Fundamental analysis rely more on data from financial records, company assets, and economic reports to determine if a company is in a health state and to ascertain how much revenues, expenses, and profits a company made. The second method is technica

l analysis. It focuses on the past market action to predict futu re price movement. Technical analysis primarily concerned with t he price and the volume, as well as the demand and supply factor s that move the market.

Our project mainly focuses on building a stock inquiry, stock st orage and stock prediction pipeline, and therefore focuses on te chnical analysis. The goal of our project is to benefit individu al investors by giving them tools and algorithms that they can u se to make better predictions.

2. Glossary of Terms

SVM: One of algorithms to make long term prediction on stock’s price by using Support Vector Regression (SVR) library.

Bayesian Curve Fitting: One of algorithms to make short term prediction on stock’s price by using Bayesian Ridge Regression library.

ANN: One of algorithms to make short term prediction on stock’s price by add four layers to train the model.

Long term prediction: In this project, we have long term and short term prediction. Long term prediction means we have to get predicted price in thirty days. For example, if we train data from 1st day to 160th day, the predicted price should from 30th day to 190th day.

Short term prediction: There is a big difference between long term prediction and short term prediction. Short term prediction means we get the predicted price in the next day. The predicted price should from 2nd day to 161th day.

Database: A database is an organized collection of data, generally stored and accessed electronically from a computer system. We use Mysql to as database in this project to store stock, which can let users to add, remove stock.

Technical Indicators: Technical indicators are mathematical calculations based on the price, volume, or open interest of a stock used by traders who follow technical analysis. Technical analysts can use indicators to predict future price movements by analyz

ing historical data. In this project, we use Moving Average (MA), Moving Average Convergence Divergence (MACD) and Relative Streng th Index (RSI) as indicators to predict price.

3. System Requirements

3.1. Enumerated Functional Requirements

Identifier	Priority	Requirement
REQ1	5	The system will use Alpha Vantage API to collect stock data.
REQ2	5	The system shall allow users to select certain stocks and dates and display indicators and historical data of the stocks on a graph.
REQ3	5	The system shall allow users to select certain stocks and dates and display today price of the stocks on a graph.
REQ4	5	Store stock data store in the database and conduct prediction based on that data.
REQ5	5	The system shall allow users to select certain stocks and display highest price and lowest price over the past one year.
REQ6	4	The system shall allow for users to search for a particular company and display the prediction result.
REQ7	4	The system shall allow users to select short term prediction and long term prediction.

Analysis: Functional Requirements

The main requirements of our project is to be able to obtain historical and live data from Alpha Vantage API and then store them in to database for future use (REQ1). In our project, we stored 10 most used stocks as a test sample. If a stock is not in the Alpha Vantage database, it will acquire nothing.

Also, users have other functionalities, such as visualize stock data and corresponding indicators , make predictions, add new st ocks and query historical data. (REQ2-7).

3.2. Enumerated Non-Functional Requirements

Identifie r	Requirement
REQ8	The system shall be able to run through all prediction models for every stock within 30 minutes.
REQ9	The system will be able to update stock values every m inutes.
REQ10	The system shall allow users to get prediction for a s tock within 2 mouse click.
REQ11	The prediction model will be updated every day using d ata acquired that day.
REQ12	The historical data will be shown in a table with cert ain number of rows.

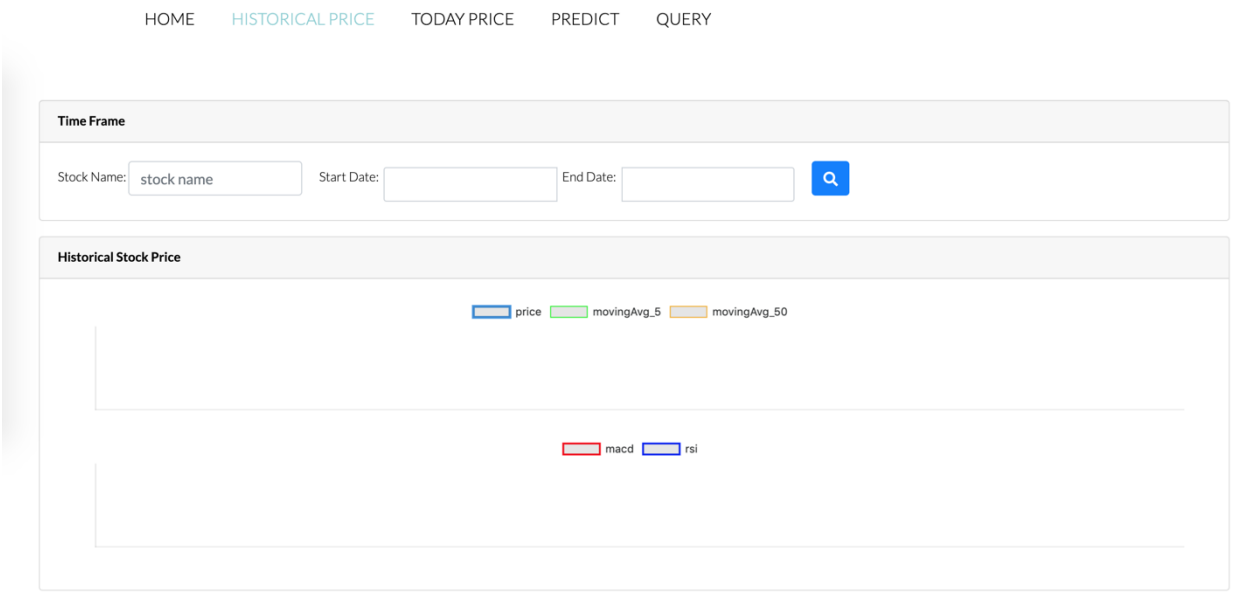
Analysis: Enumerated Non-Functional Requirements

The system will insure that latency will be low since the system will have numerous users and keeping predictions within reach an d within reasonable time are keys to users satisfaction (REQ8,RE Q10).

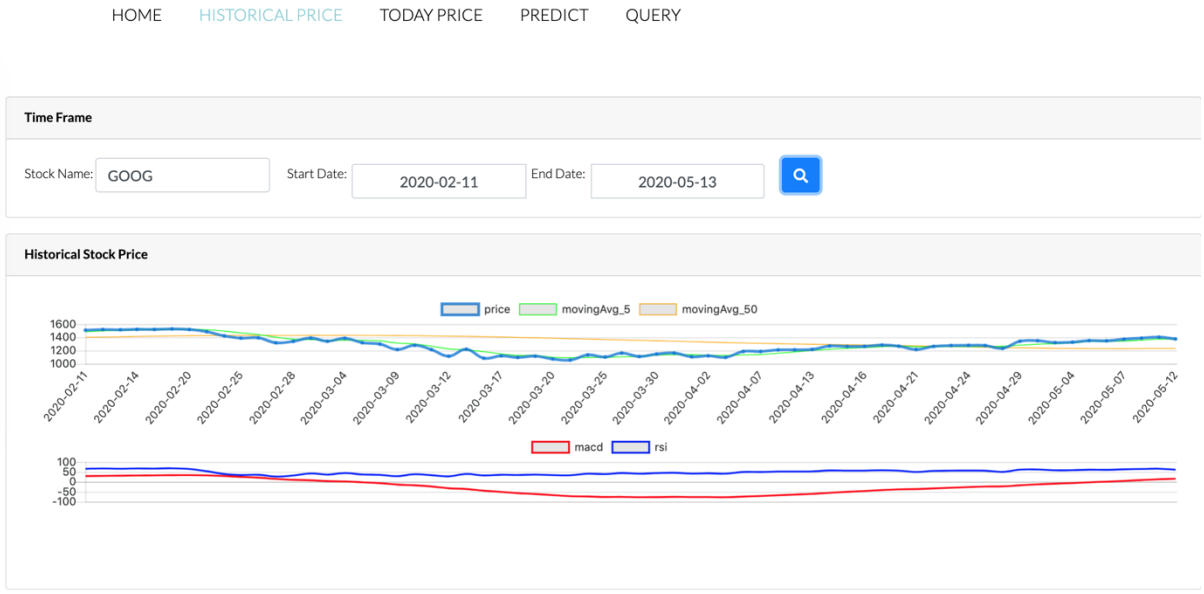
Also, the database and the model will be updated on a daily basi s to ensure that the prediction values are up to date (REQ9 REQ1 1). Finally, users can browse stock data in a table (REQ12).

3.3. On Screen Appearance Requirements

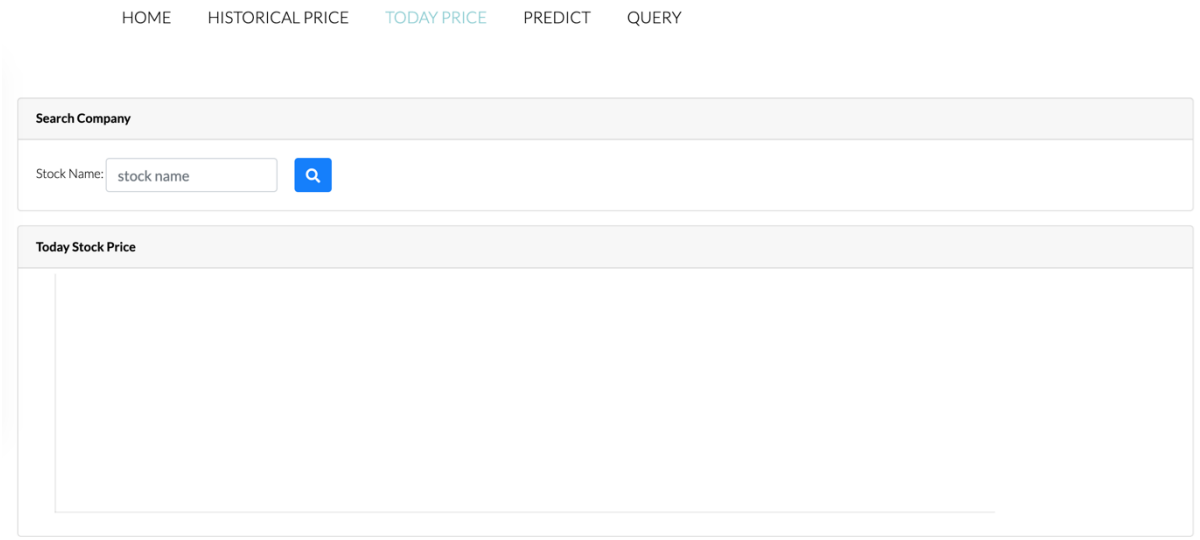
Here are screenshots of this stock prediction website.



Dashboard-Historical Stock Price

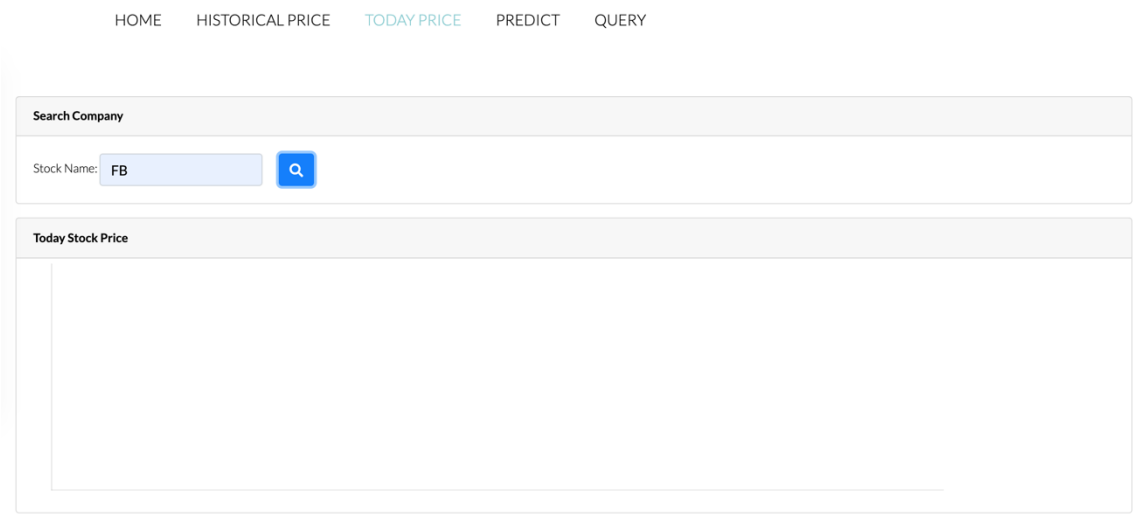


Search History Stock Data (include three indicators)

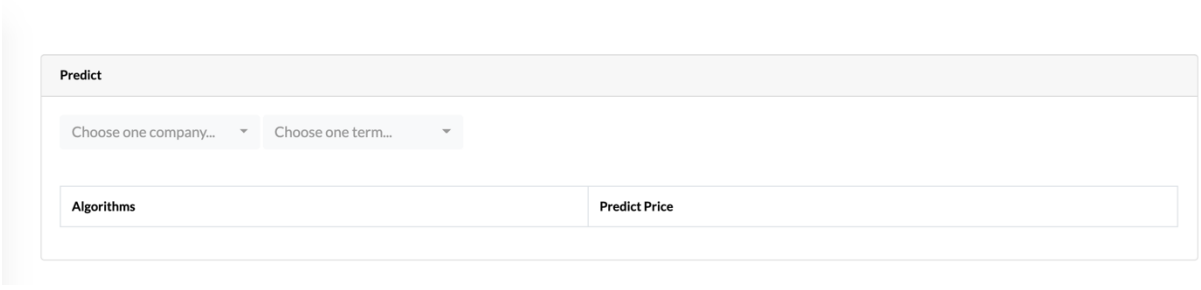


Realtime data-Today Stock Price

The time slice is 1 minute for real time data.

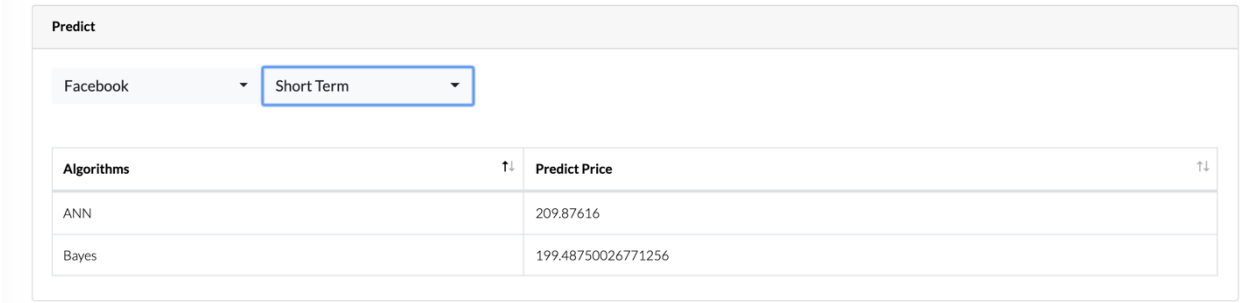


Search real time data



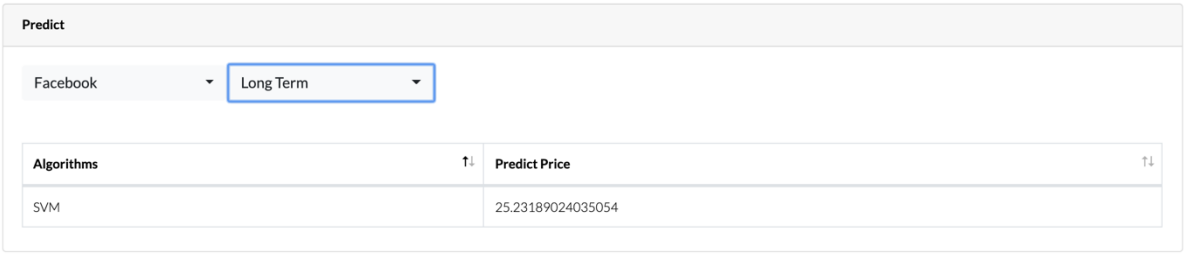
Prediction page

HOME HISTORICAL PRICE TODAY PRICE PREDICT QUERY



Prediction-Short Term (ANN, Bayes Curve Fitting)

HOME HISTORICAL PRICE TODAY PRICE PREDICT QUERY



Prediction-Long Term (SVM)

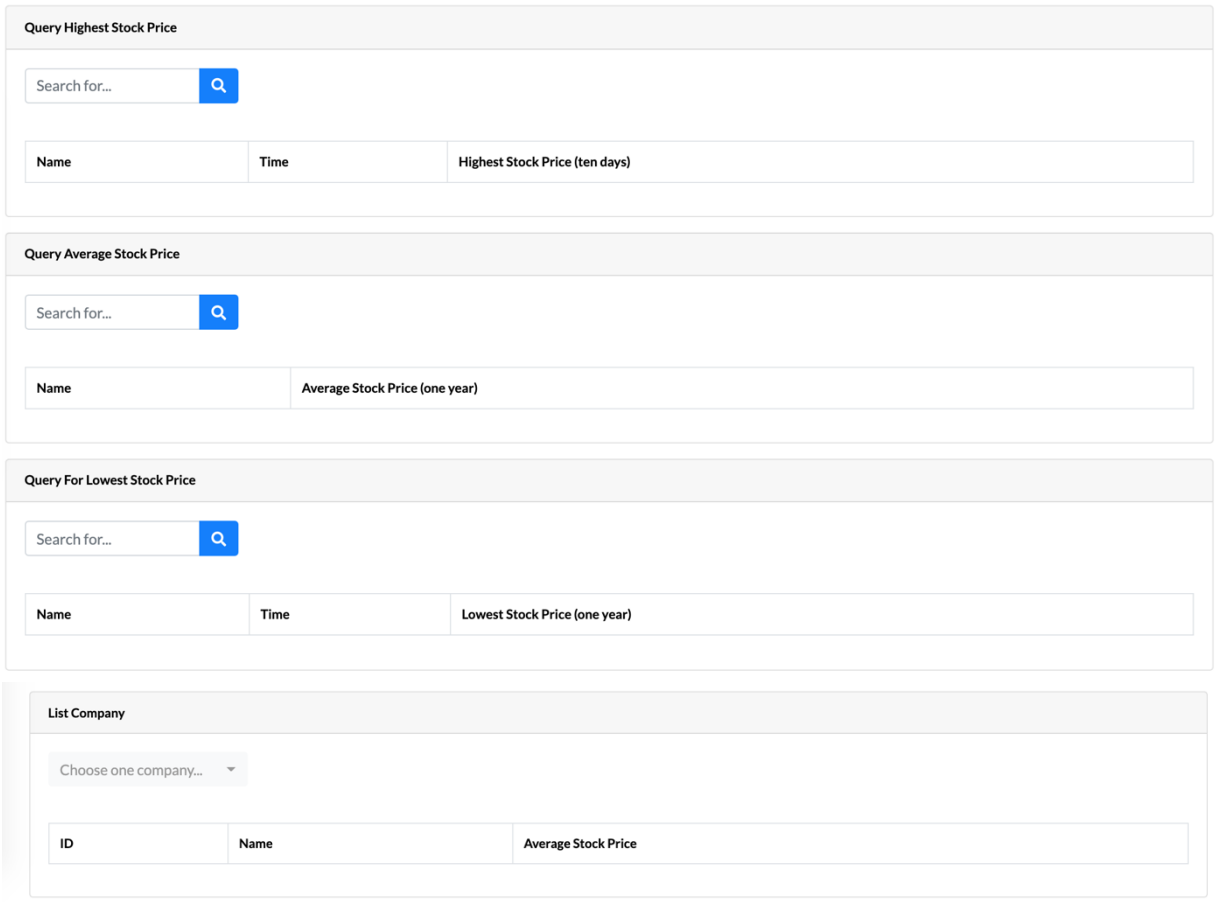
Latest Stock Price

Show 10 entries Search:

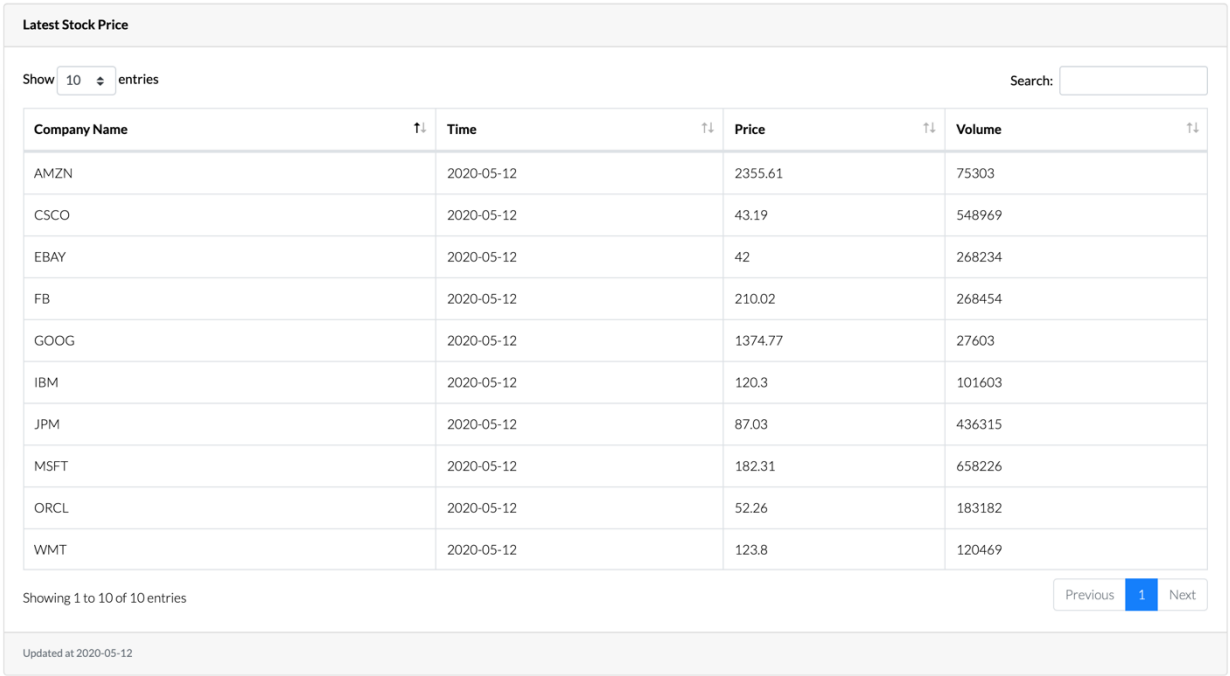
Company Name	Time	Price	Volume
AMZN	2020-05-12	2355.61	75303
CSCO	2020-05-12	43.19	548969
EBAY	2020-05-12	42	268234
FB	2020-05-12	210.02	268454
GOOG	2020-05-12	1374.77	27603
IBM	2020-05-12	120.3	101603
JPM	2020-05-12	87.03	436315
MSFT	2020-05-12	182.31	658226
ORCL	2020-05-12	52.26	183182
WMT	2020-05-12	123.8	120469

Showing 1 to 10 of 10 entries Previous 1 Next

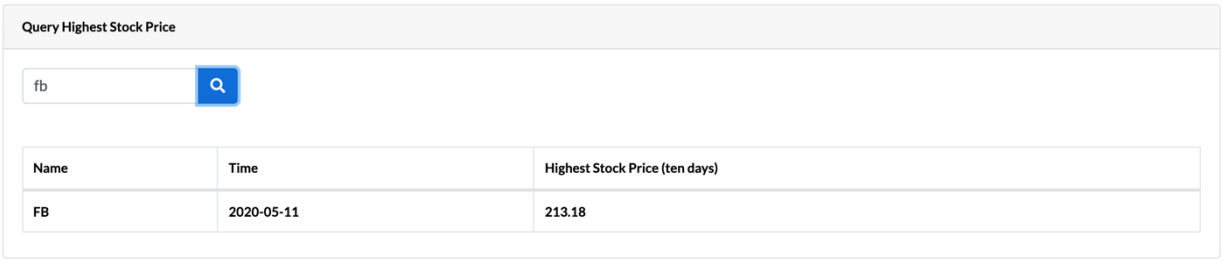
Updated at 2020-05-12



Query page

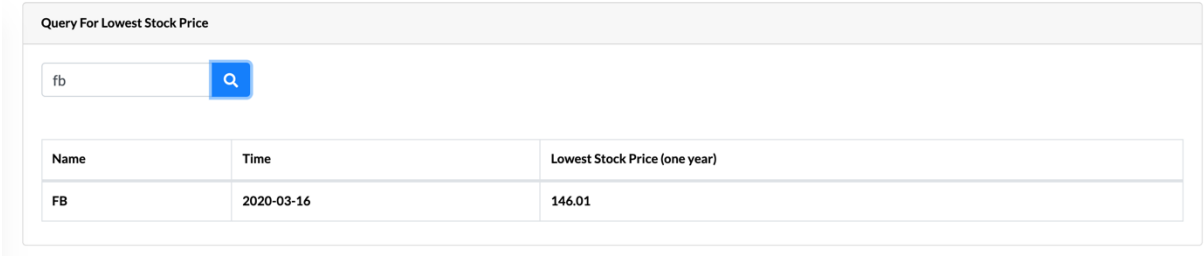


Query-Latest Stock Price

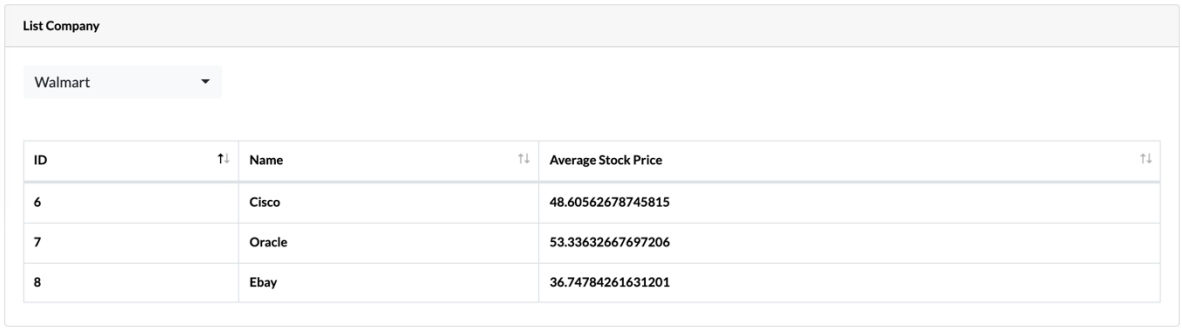


Query-Highest Stock Price in the latest ten days

Query-Average Stock Price in the latest year



Query-Lowest Stock Price in the latest year



Query-List company

4. Functional Requirements Specification

4.1 Stakeholders

Client: any user uses the webiste
Administrator: maintain service for users

4.2 Actors and Goals

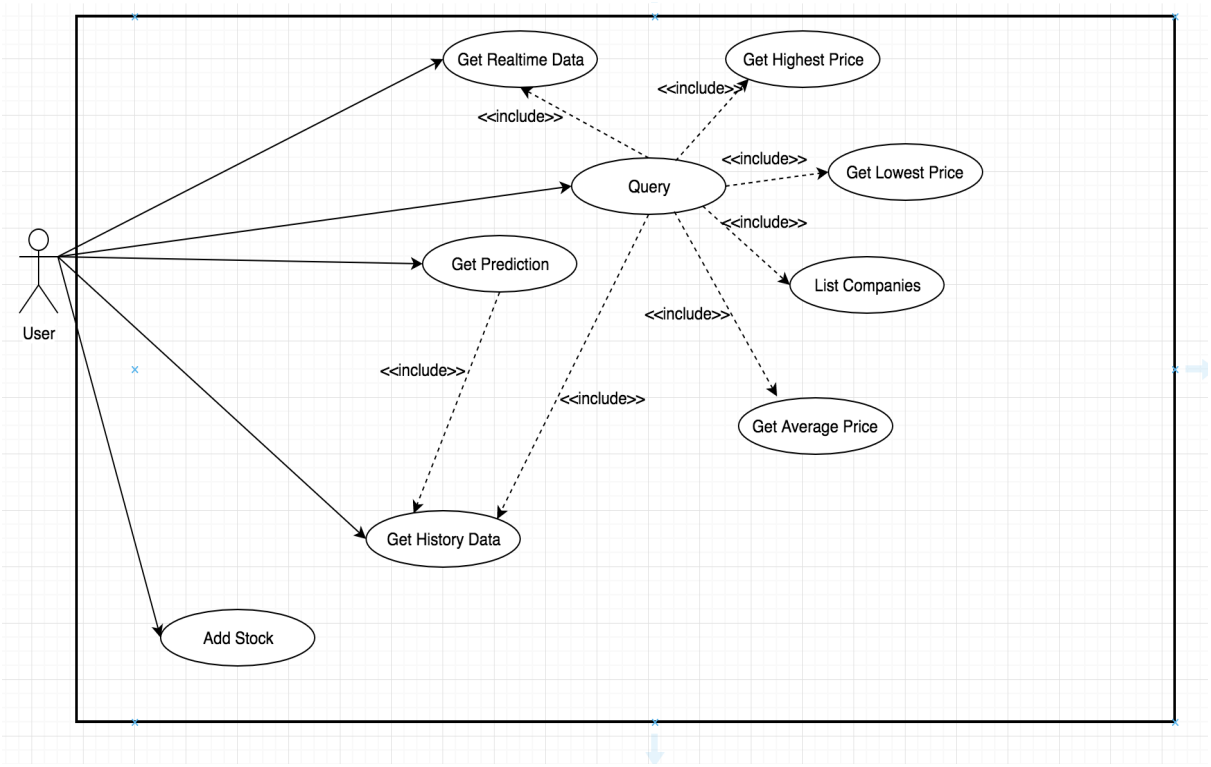
User: a user which visits the website
Database: records of real-time and historical stock information
Price Provider: Yahoo! Finance which provides stock information
Timer: a timer to update stock data periodically
Visualization: use chart.js to visualize stock data
Administrator: a special user maintains and updates website services

4.3 Use Cases

4.3.1 Casual Descritption

Get Real-time Data: allow users to get real-time data from MySQL and update stock data from Alpha Vantage API
Get Historical Data: allow users to get historical data from MySQL database
Get Prediction: Use historical data to predict next day and next 30 day stock pric e
Get Highest Price: get highest price of one stock
Get Lowest Price: get lowest price of one stock
Get Average Price: get average price of one stock
Add New Stock: download historical data of a stock from Alpha Vantage API

4.3.2. Use Case Diagram



4.3.3 Fully Dressed Description

Use Case: Add Stock
Initiating Actor: User **Actor' s Goal:** To add a new stock for user to track **Participating Actors:** Price Provider, Database **Preconditions:** None **Success End Condition:** The stock name is included in the Alpha Vantage API **Failed End Condition:** The stock name is not included in the Alpha Vantage or the database already exist the stock **Flow of Events for Main Success Scenario:** 0. User enters a stock name in the browser 1. System verifies the stock is in the database or not 2. System creates a request URL for historical prices 3. System creates a request URL for real-time prices 3. System requests historical prices from Alpha Vantage API 4. System requests real-time prices from Alpha Vantage API 7. Alpha Vantage API returns current prices for the given stock 8. Alpha Vantage API returns historical prices for the given stock 9. System stores the historical prices into database 9. System stores the real-time prices into database 10. System notifies user that the stock is added

Use Case: Get Real-time Stock Prices
Initiating Actor: User **Actor’ s Goal:** get real-time stock prices and show in the front end **Participating Actors:** Price Privider, Database **Preconditions**: Stock name is in the database and price privder both **Success End Condition:** The stock name is included in the Alpha Vantage API and the database **Failed End Condition:** The stock name is not included in the database and Alpha Vantage API **Flow of Events for Main Success Scenario:** 1. User enters a stock name for today’ s price 2. System verifies the stock is in the database or not 3. System updates real-time stock information from Alpha Vantage API to the database 4. System queries the database based on the stock name in the stock’ s real-time table 5. System updates a new price for the stock every one minute from Alpha Vantage API and the database stores the new data. Show the stock prices in the front end

Use Case: Get Historical Stock Prices
Initiating Actor: User **Actor’ s Goal:** get historical stock prices and show in the front end **Participating Actors:** Database **Preconditions**: Stock name is in the database **Success End Condition:** The stock name is included in the database **Failed End Condition:** The stock name is not included in the database **Flow of Events for Main Success Scenario:** 1. User enters a stock name and a time range in the front end 2. System verifies the stock is in the database or not 3. System queries the database based on the stock name and the range of dates that a user enters 4. Database returns prices and dates of the stock 5. Front end shows each day’ s price on the chart

Use Case: Get Highest Price

Initiating Actor: User
Actor’ s Goal: Get highest stock price in the latest ten days and show in the front end
Participating Actors: Price Provider, Database
Preconditions: Stock name, price and time in the database
Success End Condition: The stock name is included in the Alpha Vantage API and the database
Failed End Condition: The stock name is not included in the database
Flow of Events for Main Success Scenario:
1. User enters a stock name in the search box in the front end
2. System verifies the stock is in the database or not
3. System queries the database based on the stock name that the user enters
4. Database returns name, higest stock price and time of the stock
5. Front end shows the result in the datatable

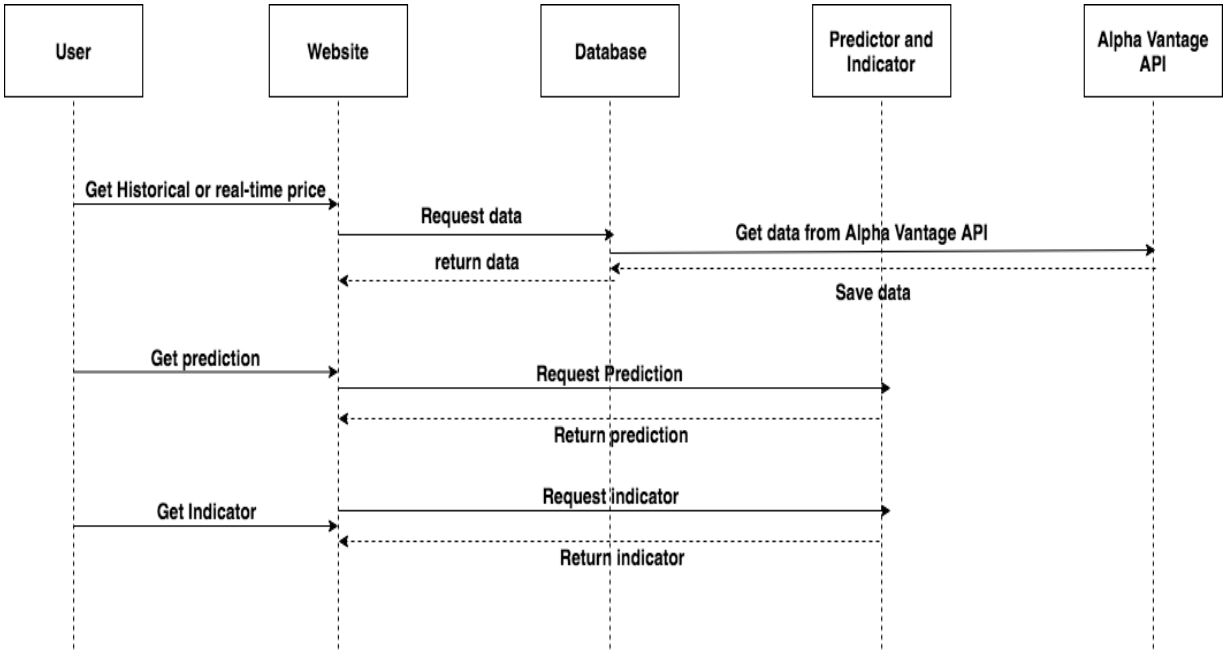
Use Case: Get Average Price

Initiating Actor: User
Actor’ s Goal: Get average price in the latest year and show in the front end
Participating Actors: Price Provider, Database
Preconditions: Stock name and price in the database
Success End Condition: The stock name is included in the Alpha Vantage API and the database
Failed End Condition: The stock name is not included in the database
Flow of Events for Main Success Scenario:
1. User enters a stock name in the search box in the front end
2. System verifies the stock is in the database or not
3. System queries the database based on the stock name that the user enters
4. Database returns name, average stock price of the stock
5. Front end shows the result in the datatable

Use Case: Get Lowest Price
Initiating Actor: User **Actor’ s Goal:** Get lowest price in the latest year and show in the front end **Participating Actors:** Price Provider, Database **Preconditions**: Stock name and price in the database **Success End Condition:** The stock name is included in the Alpha Vantage API and the database **Failed End Condition:** The stock name is not included in the database **Flow of Events for Main Success Scenario:** 1. User enters a stock name in the search box in the front end 2. System verifies the stock is in the database or not 3. System queries the database based on the stock name that the user enters 4. Database returns name, lowest stock price and time of the stock 5. Front end shows the result in the datatable

Use Case: Get Prediction
Initiating Actor: User **Actor’ s Goal:** Get prediction price and show in the front end **Participating Actors:** Price Provider, Database **Preconditions**: History stock data in the database **Success End Condition:** The stock name is included in the Alpha Vantage API and the database **Failed End Condition:** The stock name is not included in the database **Flow of Events for Main Success Scenario:** 1. User select a stock name and time period from two select boxes in the front end 2. System verifies the stock is in the database or not 3. System queries the database based on the stock name and the time period, long term or short term that the user chooses 4. Database returns the name of prediction algorithm, predicted stock price 5. Front end shows the result in the datatable

4.3.4 System Sequence Diagram



UCP = UUCP*TCF*ECF = 103*0.92*0.68 = 64.44

5.1. Concept Definitions

We will show the concepts and their definitions below

Website:

Definition: A website or Web site is a collection of related network web resources, such as web pages, multimedia content, which are typically identified with a common domain name, and published on at least one web server.[1]

Responsibilities:

- Shows HTML document that display the actor the current context(K) .
- Shows what actions can be taken through buttons(K)

Query:

Definition: search query.

Responsibilities:

- Do a unique search query (K)

Timekeeper:

Definition: keeps track of internal time of the system.

Responsibilities:

- Knows when to update real time stock data(K).
- Knows when to update history time stock data(K).

Controller:

Definition: Directs or regulates the requests made from user. Responsibilities:

- Retrieves information from Data Renderer and passes to Website (K)
- Coordinate decisions based on the specific use case (D)

[1] Website Definition https://en.wikipedia.org/wiki/Website

PageMaker:
Definition: Generates display inputs ultimately for website
Responsibilities:
• Must be able to display text, numbers, datatables and charts for website environment(D)

Predictor:
Definition: Generate stock predictions.
Responsibilities:
• Apply prediction algorithms to history stock data(D)

StockRetriever:
Definition: Collects data from Alpha Vantage API and get stock prices by querying the PriceProvider.
Responsibilities:
• Retains momentary stock data from Alpha Vantage API and passes to Data Handler(D)

StockExtractor:
Definition: Extracts stock data to be stored within the database. Responsibilities: • Extracts real time and history time stock data from a given file and stores it within the database(D)

DB:Connection:
Definition: An organized collection of stock data.
Responsibilities:
• Store company data (K)
• Store stock data (K)

Notifier:
Definition: Deals with the system for various messages.
Responsibilities:
. Notify of stock prediction system problems or errors(D)

StockHistoricalPriceDoc:
Definition: Stores historical prices for a given stock.
Responsibilities: • Stores historical prices for a given stock (K)

StockCurrentPriceDoc:
Definition: Stores real time stock price for a given stock.
Responsibilities: • Stores real time price for a given stock (K)

StockInformationDoc:
Definition: Stores stock information including stock name, comp any name and id for a given stock.
Responsibilities:
• Stores stock information for a given stock (K)

StockInfo:
Definition: Holds current stock information.
Responsibilities:
• Holds current stock information(K)

Searcher
Definition: Queries database for stocks.
Responsibilities:
• Get stocks price based on user input(D)
• Get stocks price based on prediction(D)

5.2. System Operation Contracts:
Operation Contract - Search Function
Name: Search
Responsibilities: Takes users inputted stock name, selected comp any name, time period including long term or short term and star

t date and end date and match it to the database, and retrieve t he data for that stock.
Exceptions: If the stock name does not exist.
Preconditions: You have to connect to the database and connect t o the Internet.
Post conditions:
- A stocks prediction was created and displayed.
- Stock data was displayed using datatables or charts.

Operation Contract - Changing Navigations

Name: Changing Navigations
Responsibilities: When users click one of the tab on the navigat ion bar, the web application changes to that page.
Exceptions: None.
Preconditions: You have to connect to database and also connect to the Internet.
Post conditions:
- The screen was changed to another one.

5.3. Mathematical Models

Moving Average (MA)

A moving average (MA) is a indicator in technical analysis that h elps smooth out price action by filtering out the “noise”, whi ch is based on past prices and a trend-following, or lagging, i ndicator.

The most two basic moving averages are the simple moving average (SMA) and and the exponential moving average (EMA). SMA is the s imple average of a security over a number of time periods. EMA g ives greater weight to more recent prices. Moving averages are u sually to identify the trend direction and to determine support a nd resistance levels.

Formulas For Moving Averages:

Simple Moving Average (SMA)

$(A1 + A2 + \ldots + An)$ / n

A=average in period n and n=number of time periods

Exponential Moving Average (EMA)

$$[\text{Value}_{\text{today}} \times (\frac{s}{1+d})] + EMA_{\text{yesterday}} \times [1 - (\frac{s}{1+d})]$$

s = smoothing and d = number of days

In order to calculate an EMA, you must compute the SMA over a time period. Next, you must calculate the multiplier for weighting the EMA , which is smoothing.

Moving Average Convergence Divergence (MACD)

Moving Average Convergence Divergence (MACD) is a trend-following momentum indicator to show the relationship between two moving averages of a price. The MACD is the 12-period EMA minus the 26-period Exponential Moving Average (EMA) , which is the MACD line. A nine-day EMA of the MACD is the "signal line," then plotted on top of the MACD line. The signal line can be as a trigger for buy and sell signals. Investors may buy the stock when the MACD crosses above its signal line and sell the stock when the MACD crosses below the signal line. Moving Average Convergence Divergence (MACD) indicators can be interpreted in crossovers, divergences, and rapid rises/falls.

The Formula for MACD is:

MACD= EMA(12) - EMA(26)

Relative Strength Index (RSI)

Relative Strength Index (RSI) is a indicator that measures the magnitude of recent price changes to identify overbought or oversold conditions in the a stock’s price, which is displayed as an oscillator that moves between two extremes. The values of 70 or above indicate that a stock is becoming overbought or overvalued and may be primed for a corrective pullback or a trend reversal in price; RSI reading of 30 or below means an oversold or undervalued condition.

The Formula for RSI is

$$RSI_{\text{step one}} = 100 - \left[\frac{100}{1+\frac{\text{Average gain}}{\text{Average loss}}}\right]$$

The average gain or loss is the average percentage gain or losses during the period, the average losses use positive values. The RSI is usually to use 14 periods to calculate the initial RSI value.

6. System Architecture and System Design

6.1. Architectural Styles

Client/Server Architecture

Client-server architecture is an application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients. A server host runs one or more server programs which share their resources with clients. A client does not share any of its resources, but requests a server's content or service function. Clients therefore initiate communication sessions with servers which await incoming requests. [2]

Database-centric Architecture

We use a relational database management system, the data a user need is in DBMS. The design is around the database, that is, the business logic will retrieve the data from the database, and show the result in the application interface.
designing everything else around the database.…Around the database you develop

6.2. Mapping Subsystems to Hardware

Our software can be seen as three subsystems, client, sever, and DBMS system. Client is run on the web browser, like Chrome. It can access the server through the network. The server can be run on Unix-like system, and a single server could provide all the services that a client needs.[3] The server side works with the DBMS, the server will query the data based on the business logic need and send the data.
The DBMS is deployed on AWS.

---

[2] https://en.wikipedia.org/wiki/Client%E2%80%93server_model
[3] https://www.ibm.com/support/knowledgecenter/en/SSAL2T_8.1.0/com.ibm.cics.tx.doc/concepts/c_clnt_sevr_model.html

6.3. Persistent Data Storage

The data is saved in the DBMS. For example, a real-time stock data will be saved separately based on the stock name in the system. For example, a row of each real-time stock data will cover the information like, a company name, a close price, a open price, a low price and a high price. The data will be stored using a relational database, MySQL. MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses. MySQL is written in C and C++, and can work on many different operating systems. [4]

6.4. Network Protocol

In general, a service is an abstraction of computer resources and a client does not have to be concerned with how the server performs while fulfilling the request and delivering the response. The client only has to understand the response based on the HTTP protocol. A client and a server establishes a connection using HTTP protocol. Once the connection is established, Client sends across the request to the server in the form of JSON which both client and server understand. After understanding the request server responds with appropriate data by sending back a response.

[4] https://en.wikipedia.org/wiki/MySQL

7. Algorithms and Data Structures

Support Vector Machine (SVM)

An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible(hyperplane).
The advantages of support vector machines are:

- Effective in high dimensional spaces.
- Effective in cases where number of dimensions is greater than the number of samples.
- Uses a subset of training points in the support vectors, which causes memory efficient.
- Versatile: different Kernels can be specified for the support vectors.

The support vector machines in scikit-learn support both dense and sparse sample vectors as input. However, if we want to use an SVM to make predictions for sparse data, it must have been fit on that data.

In our implementation, we use a method called Support Vector Regression. It is produced by support vector classification depends on a subset of the training data.

```python
def build_SVM_model():
    clf = SVR(gamma='scale', C=1.0, epsilon=0.2)
    return clf


def rmse(predict, truth):
    return np.sqrt(np.mean((predict - truth) ** 2))


def train_SVM_model(clf, train_data, target_data):
    clf.fit(train_data, target_data)


def test_SVM_model(clf, test_data, truth_data):
    predict_test = clf.predict(test_data)
    return rmse(predict_test, truth_data)
```

Bayesian Curve Fitting

In the curve fitting problem, we are given the training data x, t and with a new test point x, so our goal is to predict the value of t. We wish to evaluate the predictive distribution p(t|x, x, t). Here we shall assume that the parameters α and β are fixed and known in advance. A Bayesian treatment simply corresponds to a consistent application of the sum and product rules of probability, which allow the predictive distribution to be written in the form:

$$p(t|x,\mathbf{x},\mathbf{t}) = \int p(t|x,\mathbf{w})p(\mathbf{w}|\mathbf{x},\mathbf{t})\,\mathrm{d}\mathbf{w}.$$

Here p(w|x, t) is the posterior distribution over parameters. For example, this posterior distribution is a Gaussian and can be evaluated analytically. It can also be performed analytically with the result that the predictive distribution is given by a Gaussian of the form:

$$p(t|x,\mathbf{x},\mathbf{t}) = \mathcal{N}\left(t|m(x),s^2(x)\right)$$

Where mean and variance are written by:

$$
\begin{aligned}
m(x) &= \beta \boldsymbol{\phi}(x)^{\mathrm{T}} \mathbf{S} \sum_{n=1}^{N} \boldsymbol{\phi}(x_n) t_n \\
s^2(x) &= \beta^{-1} + \boldsymbol{\phi}(x)^{\mathrm{T}} \mathbf{S} \boldsymbol{\phi}(x).
\end{aligned}
$$

Matrix S is given by:

$$
\mathbf{S}^{-1} = \alpha \mathbf{I} + \beta \sum_{n=1}^{N} \boldsymbol{\phi}(x_n) \boldsymbol{\phi}(x)^{\mathrm{T}}
$$

where I is the unit matrix, and we have defined the vector Φ(x) with elements Φi(x) = xi for i = 0,...,M.

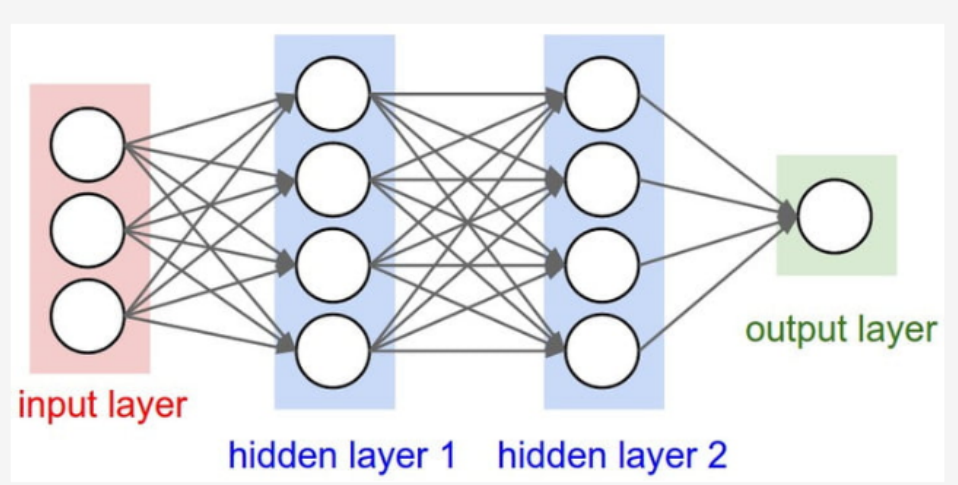
In the end we can calculate by using Bayesian Curve Fitting. The predictive distribution resulting is from a Bayesian treatment of polynomial curve fitting.

In our project, we use Bayesian Ridge Regression as library. Compared to the OLS (ordinary least squares) estimator, the coefficient weights are slightly shifted toward zeros, which stabilises them. As the prior on the weights is a Gaussian prior, the histogram of the estimated weights is Gaussian. The estimation of the model is done by iteratively maximizing the marginal log-likelihood of the observations.

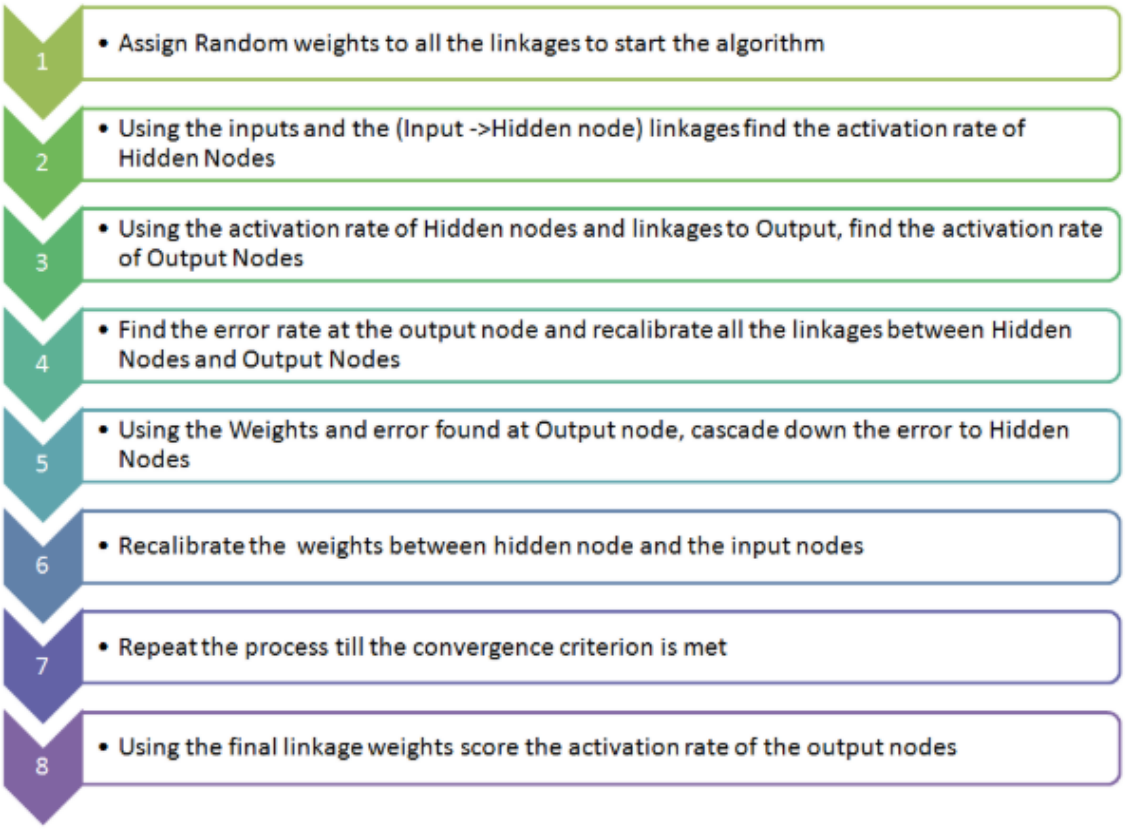
```
def build_Bayes_model():
    clf = BayesianRidge(compute_score=True)
    return clf


def rmse(predict, truth):
    return np.sqrt(np.mean((predict - truth) ** 2))


def train_Bayes_model(clf, train_data, target_data):
    clf.fit(train_data, target_data)


def test_Bayes_model(clf, test_data, truth_data):
    predict_test = clf.predict(test_data)
    return rmse(predict_test, truth_data)


def predict_Bayes_model(clf, predict_data):
    return clf.predict(predict_data)
```

Artificial Neural Network (ANN)

Artificial neural networks are main tools used in machine learning area. Neural networks consist of input, output layers, and a hidden layer consisting of units that transform the input into something where output layer can use. They are excellent tools for finding patterns and teach the machine to recognize. A technique called “backpropagation,” which allows networks to adjust their hidden layers of neurons in situations where the outcome doesn’t match what the creator is hoping for.



Following is how ANN works:



Artificial Neural Networks (ANN) have many different coefficients, so we can optimize. Hence, it can handle much more variability as compared to traditional models.

In our project, we use four layers to train our model so as to get a better model for predicted price.

```
def build_ANN_model():
    model = Sequential()
    model.add(Dense(30, input_shape=(5,), activation=None, use_bias=True, kernel_initializer='glorot_uniform',
                    bias_initializer='zeros', kernel_regularizer=None, bias_regularizer=None, activity_regularizer=None,
                    kernel_constraint=None, bias_constraint=None))
    model.add(Dense(17, activation=None, use_bias=True, kernel_initializer='glorot_uniform', bias_initializer='zeros',
                    kernel_regularizer=None, bias_regularizer=None, activity_regularizer=None, kernel_constraint=None,
                    bias_constraint=None))
    model.add(Dense(3, activation=None, use_bias=True, kernel_initializer='glorot_uniform', bias_initializer='zeros',
                    kernel_regularizer=None, bias_regularizer=None, activity_regularizer=None, kernel_constraint=None,
                    bias_constraint=None))
    model.add(Dense(1, activation=None, use_bias=None, kernel_initializer='glorot_uniform', bias_initializer='zeros',
                    kernel_regularizer=None, bias_regularizer=None, activity_regularizer=None, kernel_constraint=None,
                    bias_constraint=None))
    model.compile(loss='mse', optimizer='adam')
    return model


def rmse(predict, truth):
    return np.sqrt(np.mean((predict - truth) ** 2))


def train_ANN_model(model, train_data, target_data):
    model.fit(train_data, target_data, epochs=250, batch_size=20, verbose=0)


def test_ANN_model(model, test_data, truth_data):
    predict_test = model.predict(test_data)
```

Data structure

Pandas is a popular Python package, which offers powerful, expressive and flexible data structures to make data manipulation and analysis easily. The Pandas DataFrame is one data structure that used in our project.

DataFrames in Python are very similar and they come with the Pandas library. They are defined as two-dimensional labeled data structures with columns of potentially different types. Besides data, you can specify the index and column names for your DataFrame. The index indicates the difference in rows and the column names indicate the difference in columns.

In our project, we store data from csv file using DataFrames and also using DataFrames to deal with data by indicators (Moving Average (MA), Moving Average Convergence Divergence (MACD) as feature to train the model.

Performance

Here we can see the root mean square error (RMSE) of each algori thm. Compared with long term prediction, short term has a better performance and high accuracy. This might because we don’t have too much information to predict the price 30 days later and the s tock market is unpredictable for that long time.

```
For short term prediction:
The RMSE of Bayesian Curve Fitting: 2.9332142101224754
The RMSE of ANN: 3.01720214195715
For long term prediction:
The RMSE of SVM: 28.05918389369734
```

8. User Interface Design and Implementation

When the user enters this stock prediction website, the user wil l see the home page at first. And the user can also see a naviga tion bar on the right, showing different pages, historical price s, today price, predict and query. User can click these names to enter these pages.

The first page is historical stock price data page.The user can see three columns, time frame, historical stock price and add ne w stock. User can enter the stock name of the companies and sele ct start date and end date to see the history stock data. The le gends in these two charts show the price and the three indicator s.

The second page is for querying real time stock data. The user c an type the stock name in the search box. And the chart below th e search box will show the real time stock data. The time slice of real time data is 1 minute. You can see the real time data of whole day, from 9:30 am to 4:00pm.

The third page is about prediction. First, the user has to choos e a company and then select short term or long term. The website will show the result of prediction price of ANN and Bayes curve fitting for short term and SVM for long term.

The final page is query page. The first datatable shows the late st price of all companies in the database. It tells the user the name, price, time and volume of the stock data. The following th ree columns are used to search highest stock price in the latest ten days, lowest and average stock price in the latest one year for one stock. And the list company column tells the user the id s, name and average stock price of companies along with their na

me who have the average stock price lesser than the lowest of any of the selected company in the latest one year.

We use jQuery to handle button click events. When a certain button is clicked, we use AJAX to perform an asynchronous HTTP (Ajax) request to get corresponding result. We use Chart.js to display plots on the web page.

9.Design of Tests

User Interface Testing

Query for Stock

This is one of the important parts in this stock prediction system. Use can query stock data from the database. If the stock data is in the database, the user can also see the prediction result of the selected company for different time periods. The test for this use case is to make sure that the query result is correct and the query function is working correctly. Furthermore, this test also includes error handling in case the user has entered something wrong.

Stock Page

Predict Page

This test wants to make sure that the user can get predict result from the predict page. And it can also make sure that the user can select different time periods for different algorithms. And we also compute the RMSE error to make sure the algorithms we choose are suitable for making predictions. Furthermore, we will use previous data for testing which can help us decide how much data we choose for training and partition.

Chart Page

This test will make sure the user can get history data and real time for different stock after selecting various time span. There are two charts, one for history stock data and the other real time data.

Load Page

We will run test for all the stock pages to make sure these pages will load in a proper time, especially for the stock page. We will run test on adding new stock feature to make sure it will not cost too much to load.

10. History of Work

First stage: Create Database and Data Collection

We first find the tool to collect data, and then find Alpha Vantage API which is free and can collect stock data much easier without scraping the stock information from Yahoo! Finance directly. Through Alpha Vantage API, we can collect real-time data and historical data, and then save the stock information into MySQL database. Weikag Li finished this part.

Second Stage: Client and Sever Side Development

We choose Flask as our backend framework, which is a mirco framework. We add routes to each web page to get corresponding result and display it in front-end. We use HTML to construct web page, CSS to style HTML, JavaScript to add logic to web page. This part is done by Ningyuan Zhang.

In the function linked to each route, we connect to database and execute SQL code to retrieve data and return it to front end. This part is done by Weikang Li.

Third Stage: Prediction Algorithms and Indicators

We use three different algorithms (Support Vector Machine (SVM), Bayesian Curve Fitting and Artificial Neural Network (ANN)) and three different indicators (Moving Average, Moving Average Convergence Divergence (MACD) and Relative Strength Index (RSI)) to predict price in this project. Xueyu Wu is responsible for this part.

For long term prediction, we use Support Vector Machine (SVM) algorithm and Moving Average as feature to train the model to get predicted price in thirty days, 50 days average and 100 days average. For example, if we train data from 1st day to 160th day, we will get predicted price from 30th day to 190th day.

For short term prediction, we use Bayesian Curve Fitting and Artificial Neural Network(ANN) algorithms and Moving Average Convergence Divergence (MACD) and Relative Strength Index (RSI)) as feature to train the model to get the next day predicted price. If we train data from 1st day to 160th day, we will get predicted price from 2nd day to 161th day.

10.1. Future Work

About the prediction models, we plan to add more algorithms for the stock prediction, and then evaluate each algorithms one by one. For the web system part, we plan to improve the user interface. We'll make usage of the chart more flexible. In addition, we'll add more features like allow users to add new stock data to database.

11. References

[1] https://www.google.com/search?q=stock+prediction+picture&source=lnms&tbm=isch&sa=X&ved=0ahUKEwiK-v2OrIXiAhVQnuAKHfE-DXYQ_AUIDigB&biw=1680&bih=851#imgrc=5VGZoF7sPY8ekM:

[2]
https://en.wikipedia.org/wiki/Website

[3]
https://en.wikipedia.org/wiki/MySQL

[4]
https://www.ibm.com/support/knowledgecenter/en/SSAL2T_8.1.0/com.ibm.cics.tx.doc/concepts/c_clnt_sevr_model.html

[5]
https://en.wikipedia.org/wiki/Client%E2%80%93server_model

[6]
http://flask.pocoo.org/

[7]
https://jquery.com/

[8]
https://www.w3schools.com/html/

[9]
https://developer.mozilla.org/en-US/docs/Web/CSS

[10]
https://scikit-learn.org/stable/

[11]
https://realpython.com/python-requests/

[12]
https://www.google.com/search?q=chartjs&oq=chartjs&aqs=chrome..69i57j0l5.3735j0j7&sourceid=chrome&ie=UTF-8

[13]
https://stackoverflow.com/questions/13670075/how-does-ajax-work-with-python