PROJECT TITAN-X

AI ASSISTANT FOR STOCK INVESTING

Final Report

AIDI-2005-01 - CAPSTONE TERM II
Course Facilitator: Marcos Bittencourt
Prepared by Team*Ace*

April 2020

TABLE OF CONTENTS

Our Team

The people behind the project.

Team*Ace*

"We always aim to conquer any difficulties and challenges. 'Just do it, make it happen, and ACE it!' is our team's core spirit."

Keng Hin Cheong

100711777

AI Specialist

Au Quang Loc Nguyen

100741710

AI Specialist

Chutu Li

100765238

AI Specialist

Executive Summary

Stock trading has always been a task done by the very trading professionals in finance, since it can be very difficult, risky, time consuming and emotionally affected. Some people would even argue that it is similar to gambling in a casino. Those who really want to invest in the stock market are always concerned for a plethora of reasons such as having too little time to understand the stock market and all of its options. And when the people actually go ahead and invest their hard-earned money into stock markets, it is most likely that they will lose it all, due to the lack of experience with the scenarios and even the most common situations of stock markets. In the end, more often than not, they become frustrated and decide for other types of investments.

Therefore, our project aims to address the problems mentioned above regarding stock trading and attempts to solve them in a way that brings everyone to invest in stocks a peace of mind- with a highly capable, machine-learning-driven expert system doing most of the dirty work. The expert system will be designed to solve the difficulties of investing, which consists of removing the irrationality that leads to bad decisions because of human emotions and the large amount of time required to gain understanding of how stock market works and how to invest stocks with profits. The risks involved will be reduced since the expert system will analyze and recommend the best stocks in terms of future returns. No previous understanding of the stock market is necessary for users with this system developed. All the information needed from users will only be their text or voice inputs for the kind of stocks they would like to invest and how long they would like to hold their investments, then the expert system will come out with a list of stocks and pick the best ones for the users based on the future returns.

The ultimate goal of our project is not only to help new stock investors make money but to expand the community and get more people involved in this big industry that has no plans of stopping any time soon. However, due to the limits of time and resources, our project at this stage will only cover the SP500 IT sector of the US stock market.

Rationale Statement

The main goal of our project will solve the difficulties of investing, which consists of removing the irrationality that leads to bad decisions because of human emotions and the large amount of time required to gain understanding of how stock market works and how to invest stocks with profits. The risks involved will be reduced since the AI system will analyze future price and returns of the stock for both long term and short term. It will also recommend stocks to invest based on the user's input. No previous understanding of the stock market is necessary for users with this AI system developed by the project. All the information needed from users will only be their text inputs for the stocks they would like to invest and how long they would like to hold their investments, then the AI system will predict the future price and returns of the stock, and recommend stocks accordingly

Our goal is to make the AI engine’s accuracy rate to be over 80% in prediction. We also expect user satisfaction to the engine to be higher than 90%.

Clients/users will continuously benefit from our project by being able to trade with ease and have less worry, more time to spend on their daily lives, and, most importantly, maximize their investment returns in stock trading.

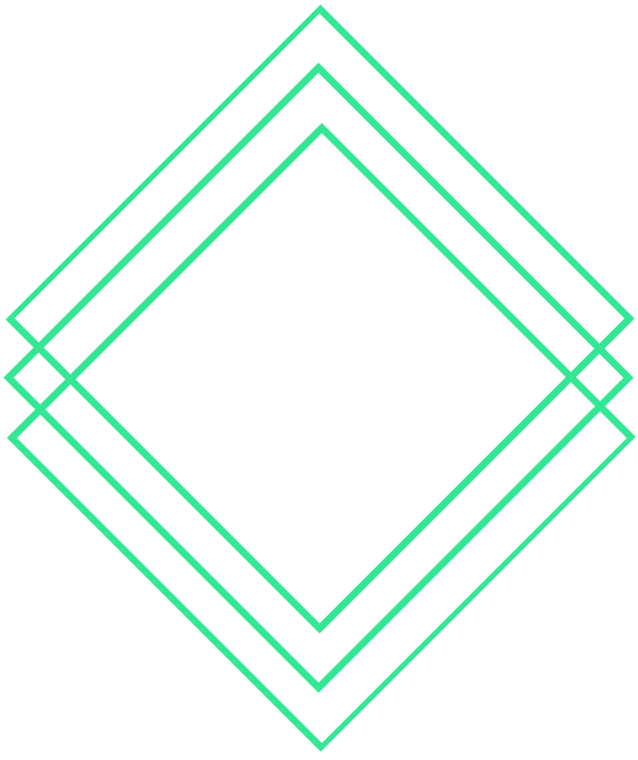
Problem Statements

1. Investing in the stock market is hard and not for beginners. 53% of people think that they don't have enough money to invest, 21% don't know about stocks, 9% don't trust the stockbrokers and 7% think that stocks are too risky. The 10% left are afraid of higher fees and some other reasons*.
    a. Our project will make investing easy and care-free, since the user will only have to enter the symbol of the stock he wants to invest in, and the system will take care of the rest.
2. Stock investors make irrational decisions based on emotions, accounting for almost 90% of losses in the stock market*.
    a. Our project will base its decisions on ML algorithms which learn from data and tendencies. It will improve the chances of winning and reduce that of loss.
3. Stock markets are different due to political and economic conditions. And because of the limit of resources and time, our project will focus on the US stock market only.

**See References*

Data Requirement

1. All data needs to be timely, accurate, consistent with the US Stock market.
2. All data needs to be relevant to our project objectives.
3. For training, validating, and testing our machine learning algorithm properly, the SP500 IT sector data should at least have columns of “Date”, “Close”, where “Date” is the record date; “Close”, is the price of the stock when the US stock market closes.
4. All the above stock dataset should have at least 3 years of data to be enough for effective machine learning.
5. The information dataset of each individual company should have company’s description, sector, Industry.



Data Requirement

Assumptions:

1. The stock/index data sources are trustable and accurate with the information provided.
2. The descriptions data of each company in the SP500 IT sector is accurate and up-to-date.

Constraints:

1. The dataset are all historical data, not real-time. Therefore, it may not reflect the most up-to-date info about the index/stock prices of our interest.
2. Due to the fact that all our stocks/index data only reflects daily changes, our project will produce better outcomes for medium to long term investors of the users, since it would be less dependent on the analyzing the immediate fluctuations on stock prices than short term investors.
3. More columns in the stock/index dataset are preferred, such as P/E ratio, intra-day high, and intra-low, for better predictive results.
4. The pre-market and after-market prices/index are not included in the datasets. The information could be helpful for price prediction for the machine learning algorithms.
5. Other factors could also influence stock prices, such as political, social factors. The data for those could be added to the project in the future if needed to provide better predictive results.

Data for the Project

The stock prices data and description of each company in SP500 IT Sector are needed for our project. Therefore, we are using an API called 'pandas_datareader.data' to get the stock data *and scrapping the information of the description of each company in SP500 IT sector on from Yahoo Finance:*

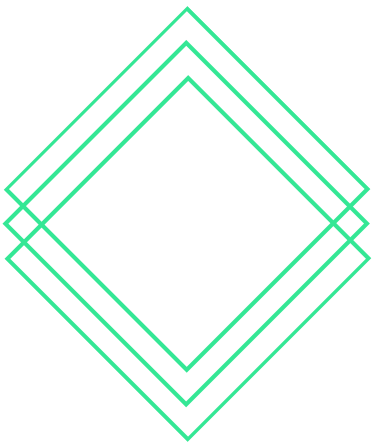
1. Category of Data: Open data
   For example, for the stock price of the Company of Apple:
   The sample of the Stock Dataset:

```
import pandas_datareader.data as web
```

```
In [91]: #get the stock data for Apple Inc using pandas_datareader
         stock_symbol = 'AAPL'
         data = web.get_data_yahoo(stock_symbol, '1/1/2016', '12/6/2019',)
         data.reset_index(inplace=True,drop=False)
```

```
In [96]: data.head(10)
```

	Date	High	Low	Open	Close	Volume	Adj Close
0	2016-01-04	105.370003	102.000000	102.610001	105.349998	67649400.0	98.446655
1	2016-01-05	105.849998	102.410004	105.750000	102.709999	55791000.0	95.979675
2	2016-01-06	102.370003	99.870003	100.559998	100.699997	68457400.0	94.101387
3	2016-01-07	100.129997	96.430000	98.680000	96.449997	81094400.0	90.129868
4	2016-01-08	99.110001	96.760002	98.550003	96.959999	70798000.0	90.606438
5	2016-01-11	99.059998	97.339996	98.970001	98.529999	49739400.0	92.073563
6	2016-01-12	100.690002	98.839996	100.550003	99.959999	49154200.0	93.409874
7	2016-01-13	101.190002	97.300003	100.320000	97.389999	62439600.0	91.008270
8	2016-01-14	100.480003	95.739998	97.959999	99.519997	63170100.0	92.998695
9	2016-01-15	97.709999	95.360001	96.199997	97.129997	79833900.0	90.765305

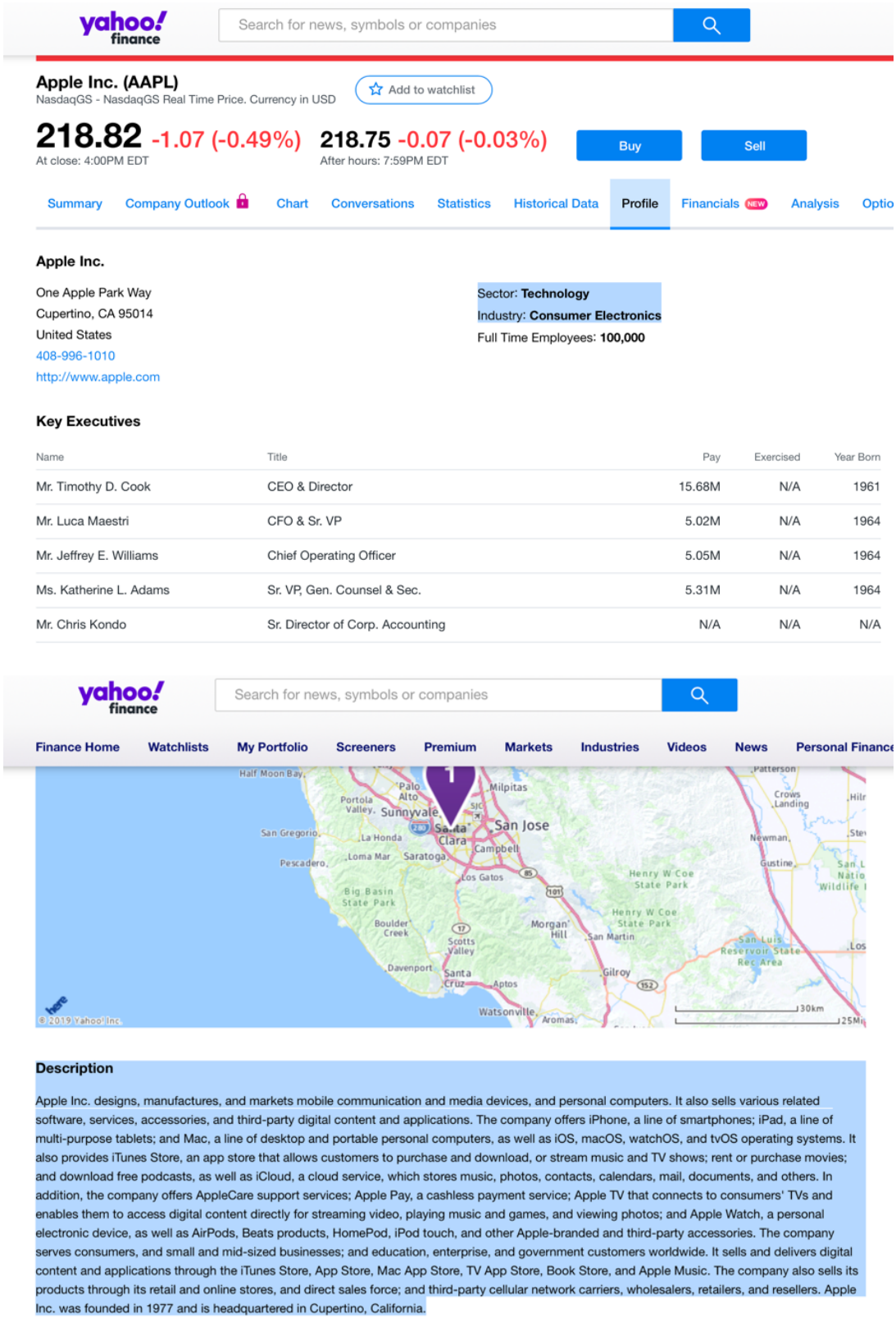


Data for the Project

2. Category of Data: Open data

Source Link: https://finance.yahoo.com/quote/AAPL/profile?p=AAPL

The Sample of the Description of the Company:



yahoo! finance

Search for news, symbols or companies

Apple Inc. (AAPL)

NasdaqGS - NasdaqGS Real Time Price. Currency in USD

Add to watchlist

218.82 -1.07 (-0.49%)
At close: 4:00PM EDT

218.75 -0.07 (-0.03%)
After hours: 7:59PM EDT

Buy | Sell

Summary | Company Outlook | Chart | Conversations | Statistics | Historical Data | Profile | Financials | Analysis | Optio

Apple Inc.

One Apple Park Way
Cupertino, CA 95014
United States
408-996-1010
http://www.apple.com

Sector: **Technology**
Industry: **Consumer Electronics**
Full Time Employees: **100,000**

Key Executives

Name	Title	Pay	Exercised	Year Born
Mr. Timothy D. Cook	CEO & Director	15.68M	N/A	1961
Mr. Luca Maestri	CFO & Sr. VP	5.02M	N/A	1964
Mr. Jeffrey E. Williams	Chief Operating Officer	5.05M	N/A	1964
Ms. Katherine L. Adams	Sr. VP, Gen. Counsel & Sec.	5.31M	N/A	1964
Mr. Chris Kondo	Sr. Director of Corp. Accounting	N/A	N/A	N/A

yahoo! finance

Search for news, symbols or companies

Finance Home | Watchlists | My Portfolio | Screeners | Premium | Markets | Industries | Videos | News | Personal Finance

Description

Apple Inc. designs, manufactures, and markets mobile communication and media devices, and personal computers. It also sells various related software, services, accessories, and third-party digital content and applications. The company offers iPhone, a line of smartphones; iPad, a line of multi-purpose tablets; and Mac, a line of desktop and portable personal computers, as well as iOS, macOS, watchOS, and tvOS operating systems. It also provides iTunes Store, an app store that allows customers to purchase and download, or stream music and TV shows; rent or purchase movies; and download free podcasts, as well as iCloud, a cloud service, which stores music, photos, contacts, calendars, mail, documents, and others. In addition, the company offers AppleCare support services; Apple Pay, a cashless payment service; Apple TV that connects to consumers' TVs and enables them to access digital content directly for streaming video, playing music and games, and viewing photos; and Apple Watch, a personal electronic device, as well as AirPods, Beats products, HomePod, iPod touch, and other Apple-branded and third-party accessories. The company serves consumers, and small and mid-sized businesses; and education, enterprise, and government customers worldwide. It sells and delivers digital content and applications through the iTunes Store, App Store, Mac App Store, TV App Store, Book Store, and Apple Music. The company also sells its products through its retail and online stores, and direct sales force; and third-party cellular network carriers, wholesalers, retailers, and resellers. Apple Inc. was founded in 1977 and is headquartered in Cupertino, California.

Architecture Approach

The pipeline for the project is described as follows:

Long Term Prediction:
The API ‘pandas_datareader.data’ will be called in Python to get the stock data given a stock symbol. Then the year, month, and day will be extracted from ‘Date’ column for preparing the machine learning training. After splitting the data into training set and test set, several algorithms will be used for the training and the results of them are compared in regression evaluations and the real-time actual stock price, to determine the models with the best accuracy to produce stock price prediction. The algorithms tested are Neural Network, Linear Regression, and Random Forest Regression.Lastly, by comparing the performance of each model, the combined best two models are chosen to be the final model to predict long term stock price.

Short Term Prediction:
Fuzzy inference systems are used and the forecasting prediction steps are as follows:
The API ‘pandas_datareader.data’ is called in Python to get the stock data given a stock symbol. A library called pyFTS - Fuzzy Time Series for Python-used. Then input value is converted into fuzzy values of the linguistic variable. Next, by defuzzyfication the compatible rules are found and finally the result are converted to a numeric value for forecasting.

Architecture Approach

Recommender for Stocks:
For this part, we pre-process the company's description to prepare for the NLP recommendation system. we first use those available libraries for text preprocessing (only clean and use the information in the description) to remove punctuation marks, special characters and numbers, etc. The steps are as follows:

1. Remove punctuation
2. Convert to lower case
3. Remove tags
4. Delete special characters and numbers
5. Convert to string list
6. Stemming
7. Lemmatisation

In this way, the text is cleared from all punctuation marks, special characters, and labels, and capital letters are standardized to lower case. Therefore, the description of each stock company is standardized as a set of lowercase words separated by a single space, so the system can easily handle it later. Next, the pre-processed data is converted into a bag of words ("BoWB"), which is a vector of words, and we will convert these BoWBs into vectors as elements for future algorithm processing for the recommendation. Then we do the same processing for customer input, and compare the formed user input vector with the vectors of description for each company. This is how the program processes stock recommendation for the keyword search of the users. When the users enter any keywords (for example, keywords in technical terms or certain types of products), the program will display a list of recommended companies for stocks according to the degree of matching with the records.

Throughout the whole project development, the programming language chosen is Python, for its simplicity and for having a plethora of libraries available. It’s also considered one of the best programming languages for Machine Learning. For deployment of the program, we have used Streamlit, which is an open-source app framework for Machine Learning and Data Science. It works with Python and is compatible with major libraries & frameworks of ML. Then we use make use of AWS for the cloud deployment of the program.

EDA & Data Preprocessing Pipeline

Stock Price Prediction-Long Term

```
In [4]: #get the stock data for stock symbol entry using pandas_datareader

        stock_symbol = input("Enter a stock symbol (for example, AAPL): ")
        data = web.get_data_yahoo(stock_symbol, '3/3/2017', '10/29/2020',)
        data.reset_index(inplace=True,drop=False)

        Enter a stock symbol (for example, AAPL): AAPL
```

```
In [5]: data.describe()
```

Out[5]:

	High	Low	Open	Close	Volume	Adj Close
count	778.000000	778.000000	778.000000	778.000000	7.780000e+02	778.000000
mean	196.473483	192.732008	194.504165	194.699650	3.132854e+07	191.109774
std	44.285336	42.914716	43.444063	43.714894	1.472784e+07	45.236476
min	138.789993	137.050003	138.740005	138.679993	1.136200e+07	132.735291
25%	163.907501	160.169994	162.612499	162.417507	2.149445e+07	157.392536
50%	186.275002	183.165001	184.755005	185.014999	2.711620e+07	180.984177
75%	217.155003	212.462498	214.617496	214.892494	3.643810e+07	211.266235
max	327.850006	323.350006	324.739990	327.200012	1.067212e+08	327.200012

```
In [7]: data.head(10)
```

	Date	High	Low	Open	Close	Volume	Adj Close
0	2017-03-03	139.830002	138.589996	138.779999	139.779999	21108100.0	133.788147
1	2017-03-06	139.770004	138.600006	139.369995	139.339996	21750000.0	133.367004
2	2017-03-07	139.979996	138.789993	139.059998	139.520004	17446300.0	133.539307
3	2017-03-08	139.800003	138.820007	138.949997	139.000000	18707200.0	133.041580
4	2017-03-09	138.789993	137.050003	138.740005	138.679993	22155900.0	132.735291
5	2017-03-10	139.360001	138.639999	139.250000	139.139999	19612800.0	133.175598
6	2017-03-13	139.429993	138.820007	138.850006	139.199997	17421700.0	133.233002
7	2017-03-14	139.649994	138.839996	139.300003	138.990005	15309100.0	133.032028
8	2017-03-15	140.750000	139.029999	139.410004	140.460007	25691800.0	134.439026
9	2017-03-16	141.020004	140.259995	140.720001	140.690002	19232000.0	134.659134

```
In [8]: data.tail(10)
```

Out[8]:

	Date	High	Low	Open	Close	Volume	Adj Close
768	2020-03-23	228.500000	212.610001	228.080002	224.369995	84188200.0	224.369995
769	2020-03-24	247.690002	234.300003	236.360001	246.880005	71882800.0	246.880005
770	2020-03-25	258.250000	244.300003	250.750000	245.520004	75900500.0	245.520004
771	2020-03-26	258.679993	246.360001	246.520004	258.440002	63021800.0	258.440002
772	2020-03-27	255.869995	247.050003	252.750000	247.740005	51054200.0	247.740005
773	2020-03-30	255.520004	249.399994	250.740005	254.809998	41994100.0	254.809998
774	2020-03-31	262.489990	252.000000	255.600006	254.289993	49250500.0	254.289993
775	2020-04-01	248.720001	239.130005	246.500000	240.910004	44054600.0	240.910004
776	2020-04-02	245.149994	236.899994	240.339996	244.929993	41419200.0	244.929993
777	2020-04-03	245.699997	240.651993	242.800003	241.827499	16656627.0	241.827499

EDA & Data Preprocessing Pipeline

Stock Price Prediction-Long Term

In [9]:
```
#extracting the year, month, and the day from the column 'Date'
data['Year'] = pd.DatetimeIndex(data['Date']).year
data['Month'] = pd.DatetimeIndex(data['Date']).month
data['Day'] = pd.DatetimeIndex(data['Date']).day
```

In [10]:
```
data.head()
```

Out[10]:

	Date	High	Low	Open	Close	Volume	Adj Close	Year	Month	Day
0	2017-03-03	139.830002	138.589996	138.779999	139.779999	21108100.0	133.788147	2017	3	3
1	2017-03-06	139.770004	138.600006	139.369995	139.339996	21750000.0	133.367004	2017	3	6
2	2017-03-07	139.979996	138.789993	139.059998	139.520004	17446300.0	133.539307	2017	3	7
3	2017-03-08	139.800003	138.820007	138.949997	139.000000	18707200.0	133.041580	2017	3	8
4	2017-03-09	138.789993	137.050003	138.740005	138.679993	22155900.0	132.735291	2017	3	9

In [18]:
```
# Split the given train dataset into training set and test set
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x, y, test_size=0.35, shuffle=False)
x_train.head()
```

Out[18]:

	Year	Month	Day
0	2017	3	3
1	2017	3	6
2	2017	3	7
3	2017	3	8
4	2017	3	9

EDA & Data Preprocessing Pipeline

Stock Price Prediction-Short Term

In [2]:
```
from yahoo_fin import stock_info as si
import pandas_datareader.data as web
import pandas as pd

data = web.get_data_yahoo('aapl', '3/3/2018', '4/15/2020',)
data.reset_index(inplace=True,drop=False)
```

In [3]:
```
data
```

Out[3]:

	Date	High	Low	Open	Close	Volume	Adj Close
0	2018-03-05	177.740005	174.520004	175.210007	176.820007	28401400.0	171.917709
1	2018-03-06	178.250000	176.130005	177.910004	176.669998	23788500.0	171.771851
2	2018-03-07	175.850006	174.270004	174.940002	175.029999	31703500.0	170.177322
3	2018-03-08	177.119995	175.070007	175.479996	176.940002	23774100.0	172.034363
4	2018-03-09	180.000000	177.389999	177.960007	179.979996	32185200.0	174.990051
...	...	...	...	...	...	...	...
528	2020-04-08	267.369995	261.230011	262.739990	266.070007	42223800.0	266.070007
529	2020-04-09	270.070007	264.700012	268.700012	267.989990	40529100.0	267.989990
530	2020-04-13	273.700012	265.829987	268.309998	273.250000	32755700.0	273.250000
531	2020-04-14	288.250000	278.049988	280.000000	287.049988	48748700.0	287.049988
532	2020-04-15	286.329987	280.630005	282.399994	284.429993	32731400.0	284.429993

Defining the number of days based on the number of rows

In [5]:
```
data['days'] = range(len(data))
```

In [6]:
```
data = data.drop(columns = ['Date', 'Close', 'High', 'Low', 'Open', 'Volume'])
```

Using only the ADJ close

In [7]:
```
data
```

Out[7]:

	Adj Close	days
0	171.917709	0
1	171.771851	1
2	170.177322	2
3	172.034363	3
4	174.990051	4
...	...	...
527	259.429993	527
528	266.070007	528
529	267.989990	529
530	273.250000	530
531	287.049988	531

532 rows × 2 columns

EDA & Data Preprocessing Pipeline

Stock Price Prediction-Short Term

1. Definition of the Universe of Discourse U

First we need to know the universe of discourse U from the training data, such as U = [min(X), max(X)]

2. Create the Linguistic Variable Ã (Universe of Discourse Partitioning)

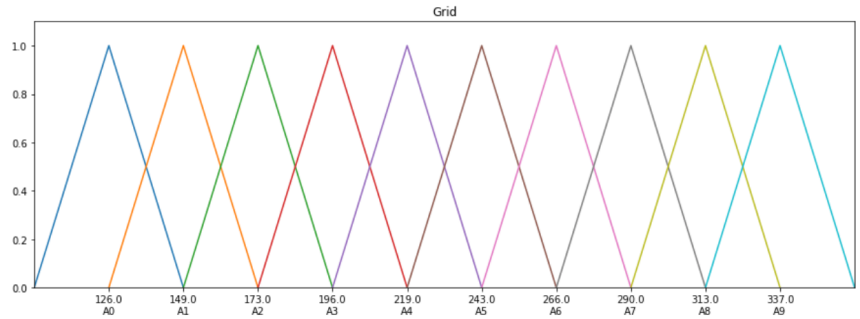
Now we need to split U on several overlapping intervals (a.k.a partitions) and create a fuzzy set for each one of them

Visualization of the fuzzy sets of the linguistic variable Adj Close

For this training I am using a 10 partitions scheme

linguistic variable be Ã = { A0, A1, ..., A9 }.

In [9]:
```
from pyFTS.partitioners import Grid

fs = Grid.GridPartitioner(data=data,npart=10)

fig, ax = plt.subplots(nrows=1, ncols=1, figsize=[15,5])

fs.plot(ax)
```



3. Fuzzyfication

Now we can convert the numerical values of X(t) into fuzzy values of the linguistic variable Ã,

In [10]:
```
# Using the method maximum, just the maximum membership fuzzy set is choosen.
#However in other FTS methods all fuzzyfied values are considered.
fuzzyfied = fs.fuzzyfy(data, method='maximum', mode='sets')
fuzzyfied
```

Out[10]:
```
['A2',
 'A2',
 'A2',
 'A2',
 'A2',
 'A2',
 'A2',
 'A2',
 'A2',
 'A2',
 'A2',
 'A2',
 'A2',
 'A2',
 'A2',
 'A2',
 'A1',
 'A2',
 'A2',
 'A2',
 'A2',
 'A2',
 'A2',
```

4. Creating the temporal patterns

A temporal patterns indicates two fuzzy sets that appear sequentially on fuzzy time series F(t) and have the format Precedent →Consequent

In [11]:
```
#This entire process runs automatically within the fit function, which trains the model

from pyFTS.common import FLR

#Using the library to show the patterns
patterns = FLR.generate_non_recurrent_flrs(fuzzyfied)

print([str(k) for k in patterns])
```

```
['A2 -> A2', 'A2 -> A1', 'A1 -> A2', 'A1 -> A1', 'A2 -> A3', 'A3 -> A2', 'A3 -> A3', '
A3 -> A4', 'A4 -> A4', 'A4 -> A3', 'A4 -> A5', 'A5 -> A5', 'A5 -> A6', 'A6 -> A6', 'A6
-> A7', 'A7 -> A7', 'A7 -> A8', 'A8 -> A8', 'A8 -> A9', 'A9 -> A9', 'A9 -> A8', 'A8 ->
A7', 'A7 -> A6', 'A6 -> A5', 'A5 -> A4']
```

EDA & Data Preprocessing Pipeline

Recommender for Stocks

```
In [7]: dataset[['Symbol','Name','Description']].head()
```

Out[7]:

	Symbol	Name	Description
0	AAPL	Apple Inc	Apple Inc. designs, manufactures, and markets ...
1	ACN	Accenture Plc	Accenture plc provides consulting, technology,...
2	ADBE	Adobe Systems Inc	Adobe Inc. operates as a diversified software ...
3	ADI	Analog Devices	Analog Devices, Inc. designs, manufactures, an...
4	ADP	Automatic Data Procs	Automatic Data Processing, Inc. provides cloud...

```
In [8]: dataset_Org = pd.DataFrame(dataset)
```

```
In [9]: stop_words=set(stopwords.words("english"))
```

```
In [16]: dataset['Processed'] = corpus
```

```
In [17]: dataset[['Description','Processed']].head()
```

Out[17]:

	Description	Processed
0	Apple Inc. designs, manufactures, and markets ...	apple inc design manufacture market smartphone...
1	Accenture plc provides consulting, technology,...	accenture plc provides consulting technology o...
2	Adobe Inc. operates as a diversified software ...	adobe inc operates diversified software compan...
3	Analog Devices, Inc. designs, manufactures, an...	analog device inc design manufacture market in...
4	Automatic Data Processing, Inc. provides cloud...	automatic data processing inc provides cloud b...

The "Proccessed" column will be converted into bag of words (the "BoWB" column) which is a vector of the words as elements

```
In [18]: BoWB =[]
for i in range(0, len(dataset['Processed'])):
    #Remove punctuations
    text_BoW = re.sub('[^a-zA-Z]', ' ', dataset['Processed'][i])
    text_BoW = text_BoW.split()
    BoWB.append((text_BoW))
```

```
In [19]: dataset['BoWB'] = BoWB
dataset[['Description','Processed','BoWB']].head()
```

Out[19]:

	Description	Processed	BoWB
0	Apple Inc. designs, manufactures, and markets ...	apple inc design manufacture market smartphone...	[apple, inc, design, manufacture, market, smar...
1	Accenture plc provides consulting, technology,...	accenture plc provides consulting technology o...	[accenture, plc, provides, consulting, technol...
2	Adobe Inc. operates as a diversified software ...	adobe inc operates diversified software compan...	[adobe, inc, operates, diversified, software, ...
3	Analog Devices, Inc. designs, manufactures, an...	analog device inc design manufacture market in...	[analog, device, inc, design, manufacture, mar...
4	Automatic Data Processing, Inc. provides cloud...	automatic data processing inc provides cloud b...	[automatic, data, processing, inc, provides, c...

```
In [20]: from sklearn.feature_extraction.text import CountVectorizer
import re
cv=CountVectorizer(max_df=0.8,stop_words=stop_words, max_features=10000, ngram_range=(1,1))
X=cv.fit_transform(dataset['Processed'])
```

We turned these BoWB into vector for later use of recommendation part

```
In [21]: dataset['dicB'] = X
dataset[['Description','Processed','BoWB','dicB']].head()
```

Out[21]:

	Description	Processed	BoWB	dicB
0	Apple Inc. designs, manufactures, and markets ...	apple inc design manufacture market smartphone...	[apple, inc, design, manufacture, market, smar...	(0, 106)\t9\n (0, 878)\t2\n (0, 512)\t1\n ...
1	Accenture plc provides consulting, technology,...	accenture plc provides consulting technology o...	[accenture, plc, provides, consulting, technol...	(0, 106)\t9\n (0, 878)\t2\n (0, 512)\t1\n ...
2	Adobe Inc. operates as a diversified software ...	adobe inc operates diversified software compan...	[adobe, inc, operates, diversified, software, ...	(0, 106)\t9\n (0, 878)\t2\n (0, 512)\t1\n ...
3	Analog Devices, Inc. designs, manufactures, an...	analog device inc design manufacture market in...	[analog, device, inc, design, manufacture, mar...	(0, 106)\t9\n (0, 878)\t2\n (0, 512)\t1\n ...
4	Automatic Data Processing, Inc. provides cloud...	automatic data processing inc provides cloud b...	[automatic, data, processing, inc, provides, c...	(0, 106)\t9\n (0, 878)\t2\n (0, 512)\t1\n ...

Algorithm Evaluation

Stock Price Prediction-Long Term

In [20]:
```
# Linear Regression

lr = LinearRegression(n_jobs= -1)

lr.fit(x_train, y_train)

y_pred = lr.predict(x_train)

print('-------Linear Regression(on Training Set)--------')
printScore(y_train, y_pred)

y_pred = lr.predict(x_test)

print('')
print('-------Linear Regression(on Test Set)------')
printScore(y_test, y_pred)
```

```
-------Linear Regression(on Training Set)--------
Evaluation Metrics
MAE          : 0.0699
MAPE         : 1.3487
RMSLE        : 0.0145

-------Linear Regression(on Test Set)------
Evaluation Metrics
MAE          : 0.1167
MAPE         : 2.0822
RMSLE        : 0.0260
```

In [21]:
```
# LR_Lasso
model_lasso = Lasso(random_state=42,alpha=0.00035)
lr_lasso = make_pipeline(RobustScaler(), model_lasso)
lr_lasso.fit(x_train,y_train)



y_pred = lr_lasso.predict(x_train)

print('-------LR_lasso Regression(on Training Set)--------')
printScore(y_train, y_pred)

y_pred = lr_lasso.predict(x_test)

print('')
print('-------LR_lasso Regression(on Test Set)------')
printScore(y_test, y_pred)
```

```
-------LR_lasso Regression(on Training Set)--------
Evaluation Metrics
MAE          : 0.0700
MAPE         : 1.3510
RMSLE        : 0.0145

-------LR_lasso Regression(on Test Set)------
Evaluation Metrics
MAE          : 0.1174
MAPE         : 2.0941
RMSLE        : 0.0261
```

Algorithm Evaluation

Stock Price Prediction-Long Term

In [22]:
```
# LGBM Regression
lgb = lgb.LGBMRegressor()
lgb.fit(x_train, y_train)

y_pred = lgb.predict(x_train)

print('-------LGBM Regression(on Training Set)--------')
printScore(y_train, y_pred)

y_pred = lgb.predict(x_test)

print('')
print('-------LGBM Regression(on Test Set)------')
printScore(y_test, y_pred)
```

```
-------LGBM Regression(on Training Set)--------
Evaluation Metrics
MAE         : 0.0121
MAPE        : 0.2347
RMSLE       : 0.0027

-------LGBM Regression(on Test Set)------
Evaluation Metrics
MAE         : 0.2342
MAPE        : 4.1757
RMSLE       : 0.0518
```

In [23]:
```
# RandomForest Regression
forest_reg = RandomForestRegressor(n_estimators=1, min_samples_leaf=1, max_features=0.5
forest_reg.fit(x_train, y_train)

y_pred = forest_reg.predict(x_train)

print('-------RandomForest Regression(on Training Set)--------')
printScore(y_train, y_pred)

y_pred = forest_reg.predict(x_test)

print('')
print('-------RandomForest Regression(on Test Set)------')
printScore(y_test, y_pred)
```

```
-------RandomForest Regression(on Training Set)--------
Evaluation Metrics
MAE         : 0.0099
MAPE        : 0.1914
RMSLE       : 0.0055

-------RandomForest Regression(on Test Set)------
Evaluation Metrics
MAE         : 0.3034
MAPE        : 5.4590
RMSLE       : 0.0567
```

In [24]:
```
# GBoost Regression
model_GBoost = GradientBoostingRegressor(n_estimators=3000, learning_rate=0.0156,
                                         max_depth=4, max_features='sqrt',
                                         min_samples_leaf=15, min_samples_split=10,
                                         loss='huber', random_state=42)

model_GBoost.fit(x_train, y_train)

y_pred = model_GBoost.predict(x_train)

print('-------GradientBoosting Regression(on Training Set)--------')
printScore(y_train, y_pred)

y_pred = model_GBoost.predict(x_test)

print('')
print('------- GradientBoosting Regression(on Test Set)------')
printScore(y_test, y_pred)
```

```
-------GradientBoosting Regression(on Training Set)--------
Evaluation Metrics
MAE         : 0.0099
MAPE        : 0.1926
RMSLE       : 0.0023

------- GradientBoosting Regression(on Test Set)------
Evaluation Metrics
MAE         : 0.2398
MAPE        : 4.2802
RMSLE       : 0.0520
```

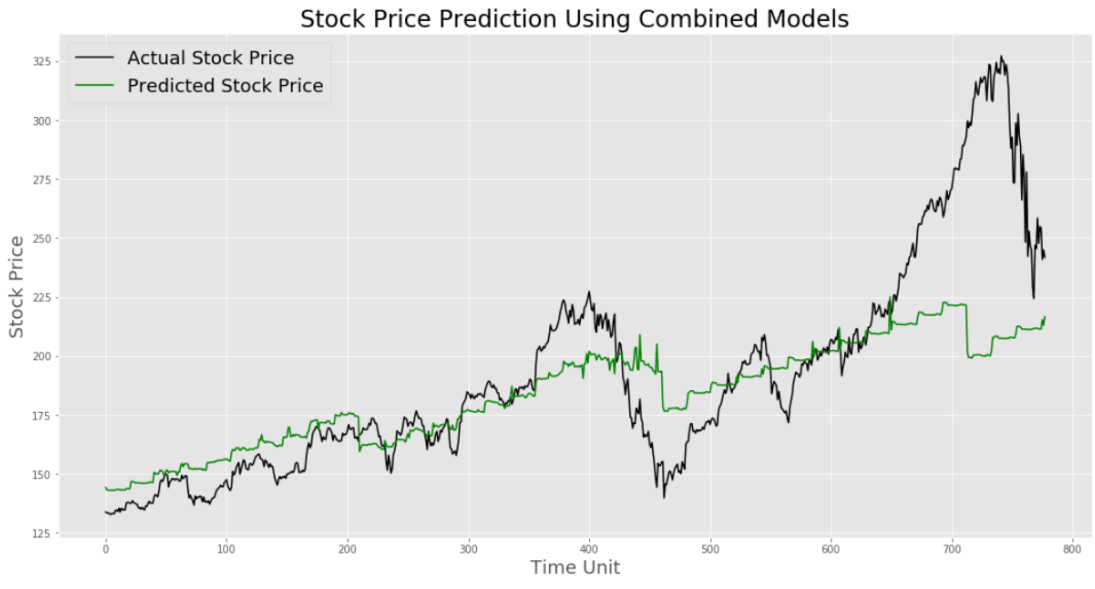
Algorithm Evaluation

Stock Price Prediction-Long Term

```
In [25]: # XGBoost Regression
XGB_regr = xgb.XGBRegressor(learning_rate=0.01,n_estimators=3460,
                            max_depth=3, min_child_weight=0,
                            gamma=0, subsample=0.7,
                            colsample_bytree=0.7,
                            objective='reg:linear', nthread=-1,
                            scale_pos_weight=1, seed=27,
                            reg_alpha=0.00006)

XGB_regr.fit(x_train, y_train)

y_pred = XGB_regr.predict(x_train)

print('-------XGB Regression(on Training Set)--------')
printScore(y_train, y_pred)

y_pred = XGB_regr.predict(x_test)

print('')
print('------- XGB Regression(on Test Set)------')
printScore(y_test, y_pred)
```

```
[13:13:54] WARNING: src/objective/regression_obj.cu:152: reg:linear is now deprecated
in favor of reg:squarederror.
-------XGB Regression(on Training Set)--------
Evaluation Metrics
MAE          : 0.0155
MAPE         : 0.3008
RMSLE        : 0.0033

------- XGB Regression(on Test Set)------
Evaluation Metrics
MAE          : 0.2471
MAPE         : 4.4120
RMSLE        : 0.0532
```

```
In [37]: #linear regression+RFR, weights: 0.8, 0.2
VizPricePredictComb(lr, forest_reg, 0.8, 0.2)
```



Overall, the Linear Regression model and RFR model perform the best. So the combined model of them is chosen to be the final output model for stock price prediction for long term.

Algorithm Evaluation

Stock Price Prediction-Short Term

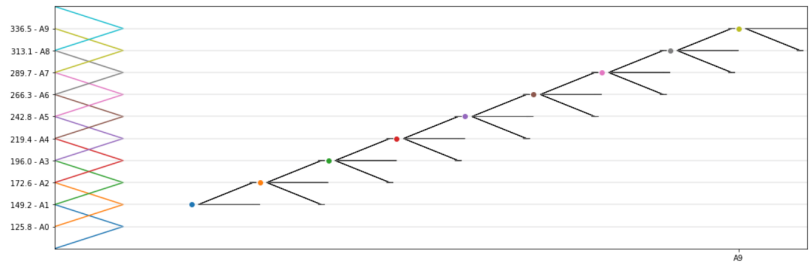
```
#Chen Fuzzy Time Series method
model = chen.ConventionalFTS(partitioner=fs)
model.fit(data)
print(model)
```

```
Conventional FTS:
A1 -> A1,A2
A2 -> A1,A2,A3
A3 -> A2,A3,A4
A4 -> A3,A4,A5
A5 -> A4,A5,A6
A6 -> A5,A6,A7
A7 -> A6,A7,A8
A8 -> A7,A8,A9
A9 -> A8,A9
```

Plot of the rules

The rule set describes how our time series behaves and, if it was stationary enough (well behaved)

In [13]:
```
from pyFTS.common import Util

Util.plot_rules(model, size=[15,5] , rules_by_axis=10)
```

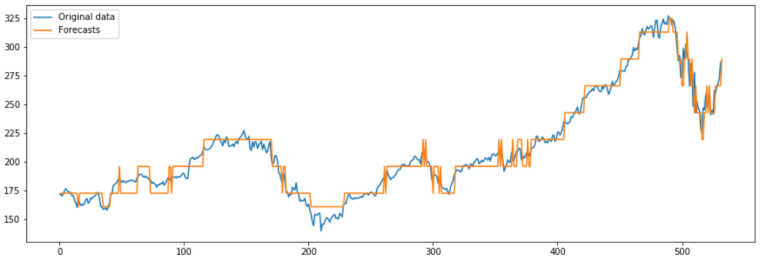


6. Forecasting

We will predict the next day Adj Close

In [14]:
```
fig, ax = plt.subplots(nrows=1, ncols=1, figsize=[15,5])

#Prediction of the existing data agains the fuzzy model
forecasts = model.predict(data)
forecasts.insert(0,None)

orig, = plt.plot(data, label="Original data")
pred, = plt.plot(forecasts, label="Forecasts")

plt.legend(handles=[orig, pred])
```

Out[14]:
```
<matplotlib.legend.Legend at 0x7f36382b9810>
```



Get the last day Adj Close to predict the following day(tomorrow)

In [15]:
```
model.predict([data[len(data)-1]])
```

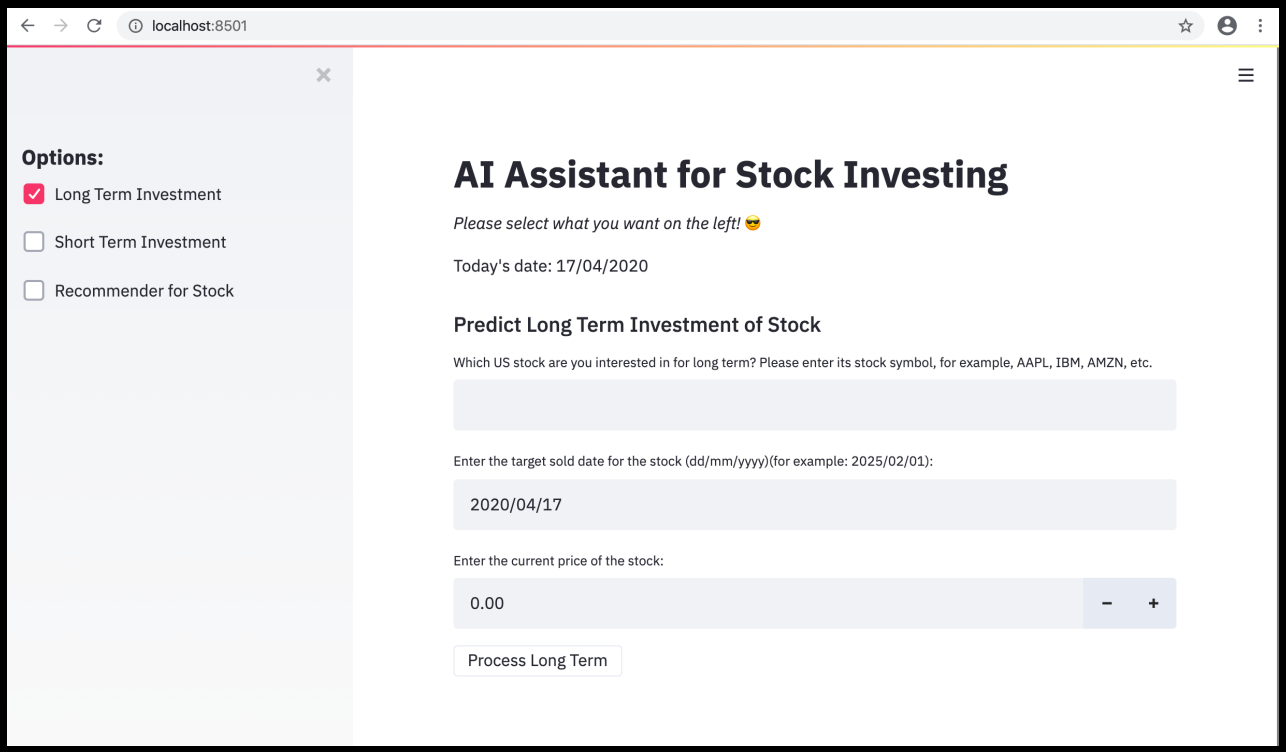
Out[15]:
```
[289.6774652099609]
```

The prediction on the stock price of AAPL and the actual price today are 289 and 284 respectively. Therefore, the accuracy rate is over 95%, which suggests the model works very well.

System Deployment

Local

```
cena2098 — open ◂ streamlit run ~/Desktop/AIDIDurham/Winter2020/AI_IN_E...
Last login: Fri Apr 17 14:14:16 on ttys001

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) CenAs-MacBook-Pro:~ cena2098$ streamlit run /Users/cena2098/Desktop/AIDID
urham/Winter2020/AI_IN_ENTERPRISE_SYSTEMS/finalProject/deployment2/app.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://192.168.2.18:8501

/Users/cena2098/anaconda3/lib/python3.7/site-packages/pandas_datareader/compat/_
_init__.py:7: FutureWarning: pandas.util.testing is deprecated. Use the function
s in the public API at pandas.testing instead.
  from pandas.util.testing import assert_frame_equal
Using TensorFlow backend.
From /Users/cena2098/anaconda3/lib/python3.7/site-packages/keras/backend/tensorf
low_backend.py:95: The name tf.reset_default_graph is deprecated. Please use tf.
compat.v1.reset_default_graph instead.
```



System Deployment

System Trial with Apple Stock

Options:

- [x] Long Term Investment
- [ ] Short Term Investment
- [ ] Recommender for Stock

AI Assistant for Stock Investing

Please select what you want on the left! 😎

Today's date: 17/04/2020

Predict Long Term Investment of Stock

Which US stock are you interested in for long term? Please enter its stock symbol, for example, AAPL, IBM, AMZN, etc.

AAPL

Enter the target sold date for the stock (dd/mm/yyyy)(for example: 2025/02/01):

2025/02/18

Enter the current price of the stock:

282.00 − +

Process Long Term

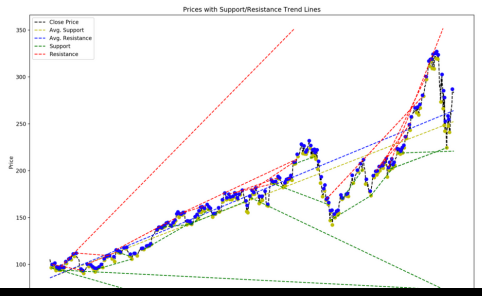
Options:

- [x] Long Term Investment
- [ ] Short Term Investment
- [ ] Recommender for Stock

Data of the Stock of the Latest 5 Trading Days

	High	Low	Open	Close	Volume	Adj Close
Apr 13, 2020	273.7000	265.8300	268.3100	273.2500	32755700	273.2500
Apr 14, 2020	288.2500	278.0500	280	287.0500	48748700	287.0500
Apr 15, 2020	286.3300	280.6300	282.4000	284.4300	32788600	284.4300
Apr 16, 2020	288.2000	282.3500	287.3800	286.6900	38845800	286.6900
Apr 17, 2020	286.9450	276.8600	284.6900	282.8000	52176318	282.8000

Long-Term Technical Analysis of the Stock (4 Years to Date)

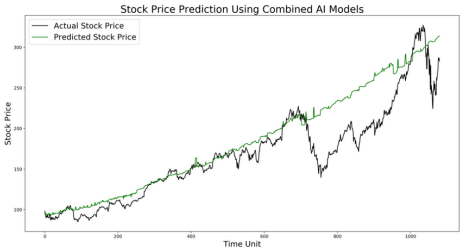


localhost:8501

Options:

- [x] Long Term Investment
- [ ] Short Term Investment
- [ ] Recommender for Stock

Actual Price vs AI Predicted Price of the Stock (4 Years to Date)



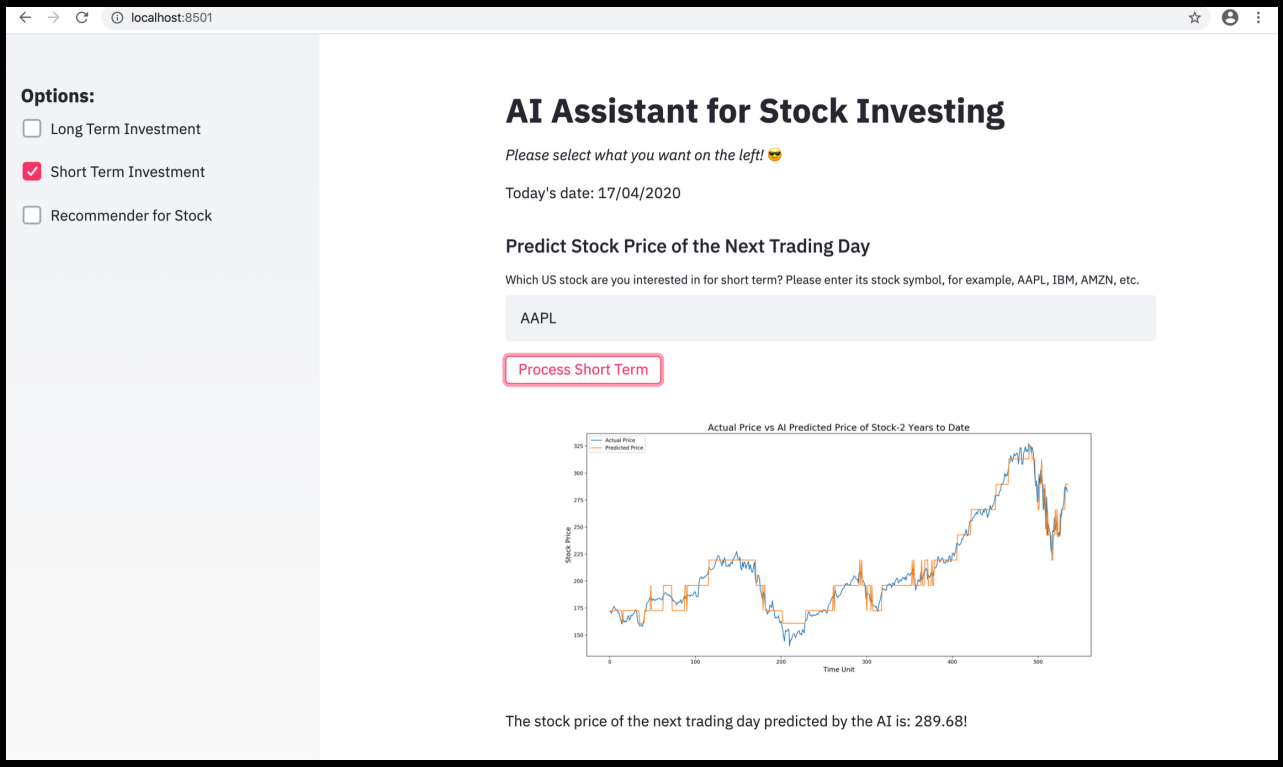
AI Prediction for Investment on the Stock

The future price on the target date is prediected as: 779.56

If you invest today, the future returns on the chosen stock till the target date is: 176.44%!

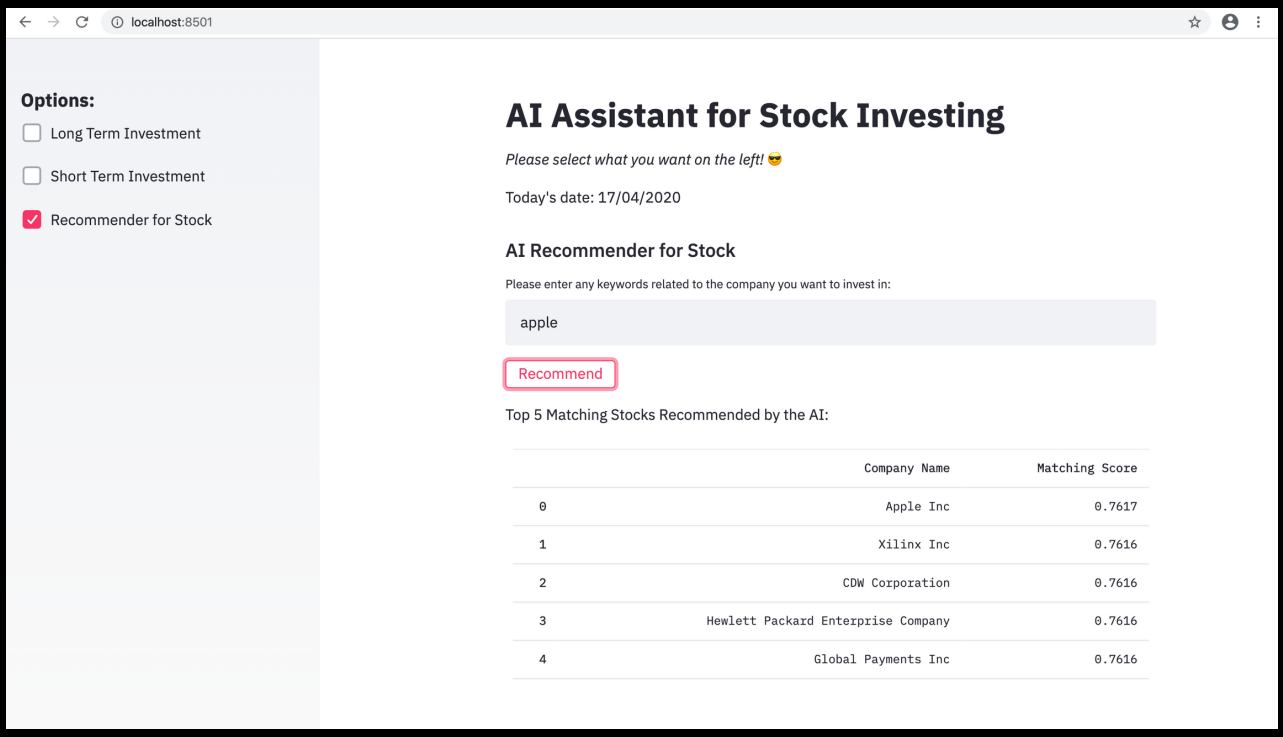
System Deployment

System Trial with Apple Stock



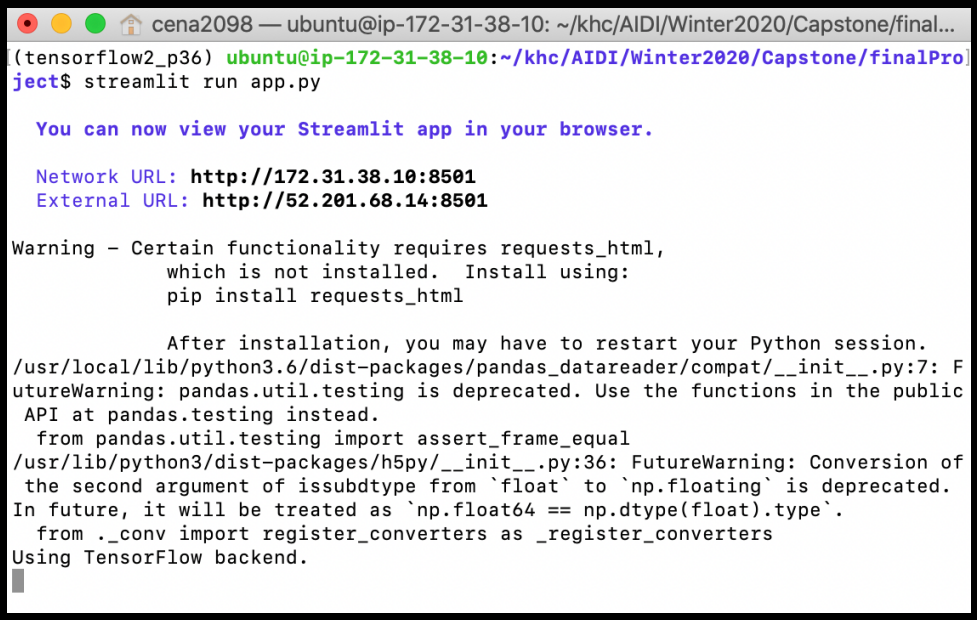
System Deployment

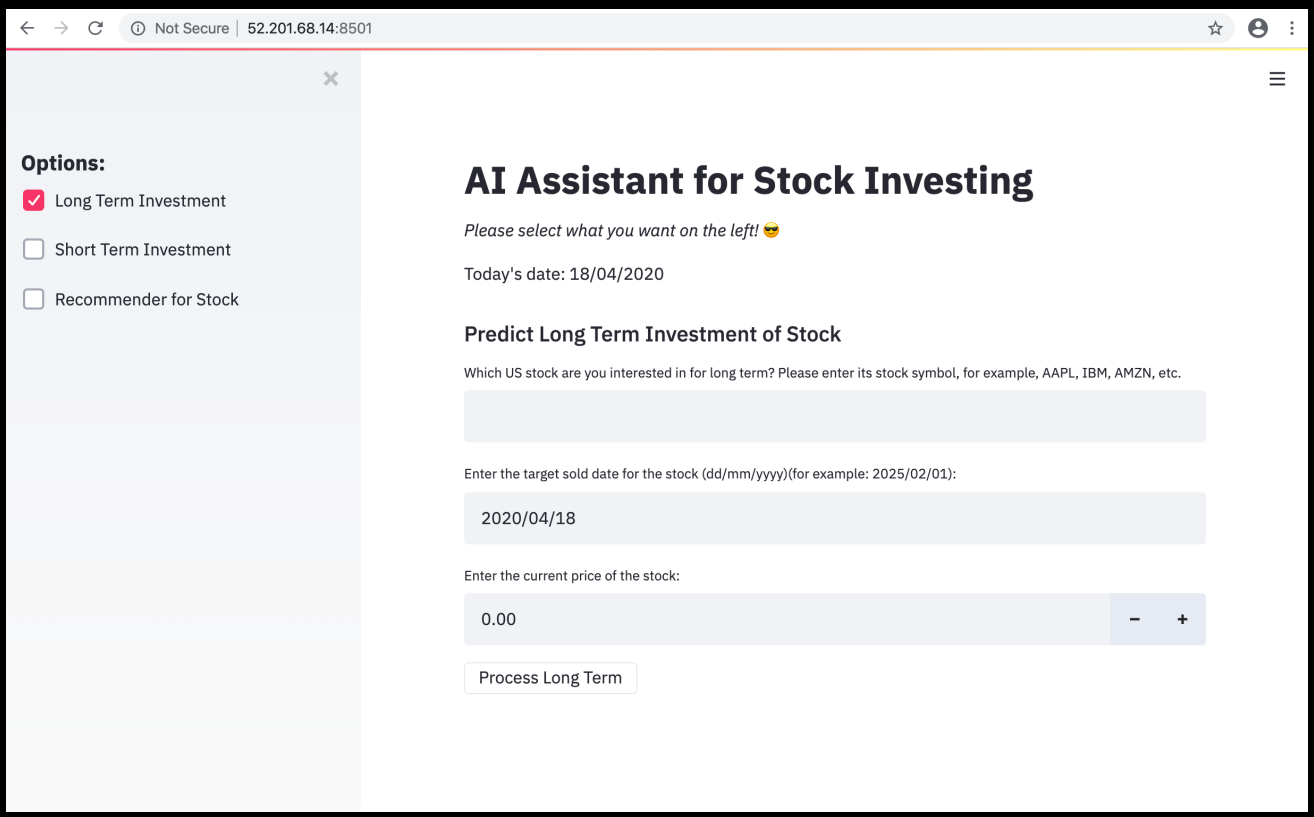
System Trial with Apple Stock



System Deployment

Cloud



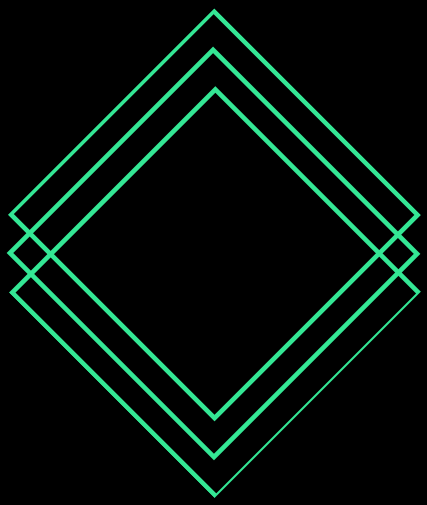


Conclusion

Stock prices are highly unpredictable and volatile in nature. This means that there are no consistent patterns in the data that allow you to model stock prices over time near-perfectly.

The result of the ML models used in the AI system gives us a good starting point for trend analysis, but there is still space for further improvements in the future. For this purpose, we can use wider datasets in addition to stock prices and the NN/ML models to perform trend analysis and combine it with the Fuzzy Model to create a Hybrid predicting AI system that will be more precise and accurate in predictions.

From the viewpoint of traders and investors, this project might help them for a better understanding of the day-to-day stock price performance and let them aware that, with AI, stock trading can become much easier. Therefore, our future works will be aimed at that direction and push the limits further.



References

https://www.quora.com/Why-do-people-not-invest-in-stock-market-when-we-have-seen-tremendous-results-in-long-term-investments

https://www.investing-arena.com/why-do-90-of-people-lose-money-on-the-stock-market/

https://www.investopedia.com/articles/basics/03/062003.asphttps://chartyourtrade.com/7-common-trading-problems-and-their-solutions/

https://www.desiretotrade.com/5-difficulties-in-trading-and-how-to-overcome-them/

https://medium.com/datadriveninvestor/machine-learning-for-stock-market-investing-f90ad3478b64

https://towardsdatascience.com/machine-learning-techniques-applied-to-stock-price-prediction-6c1994da8001

https://medium.com/recombee-blog/machine-learning-for-recommender-systems-part-1-algorithms-evaluation-and-cold-start-6f696683d0ed

https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a

https://www.toptal.com/machine-learning/supervised-machine-learning-algorithmshttps://medium.com/python-pandemonium/regression-for-understanding-machine-learning-ef89a62906a9

https://finance.yahoo.com

https://www.investing.com

https://www.ft.com

https://money.cnn.com/data/us_markets/

https://www.wsj.com

https://docs.streamlit.io

https://www.semanticscholar.org/paper/Interval-forecasting-with-Fuzzy-Time-Series-Silva-Sadaei/65e19278164559777d6c5fe3aa274e43171cadea