Investment Strategy for the Stock Market

Brian Egolf, Milena Brixey, Will Stief, Yuhua Di

Miami University ECE449

Dr. Cheng

5/6/2021

Introduction

Professional ethics are rules that govern the actions of a person or group in the workplace in order to ensure that people behave in a morally appropriate and respectful manner toward one another. That being said, the capstone experience must be performed in a professional manner that resembles a real-world business setting, with us, the students, acting as engineers on a multidisciplinary team, and with a Professor serving as an advisor to oversee and guide our project. Therefore, establishing rules for behavior —such as learning about and understanding professional ethics— sends a message that universal compliance is expected, and provides a baseline for decency, fairness, integrity, and respect.

The objective of our senior project was to "develop an investment software for the stock market." We decided to develop a program that employs machine learning to learn from past stock trends and then use this information to trade stocks and benefit from the stock market. This report covers investment strategies, machine learning and its various counterparts, and how we developed a machine learning program to help us make money in the stock market. The report begins with a clear description along with some background information about the problem we were assigned. This includes a description of what stocks and technical indicators are. Next, we summarize our progress from the first semester and outline the project goals we created for this semester. Following that is a thorough explanation of algorithmic trading, as well as the Python programming language and its libraries. This leads into the project research, where we provide an in-depth look into the solution process. Lastly, we discuss our conclusion and future plans for the project.

We were given the task of developing an investment strategy for the stock market. More precisely, we needed to develop investment software that makes use of historical stock market and economic data to construct a picking method, a buying and selling plan, or something comparable. The first requirement of the project is that the program must use data from the US stock market to develop a profitable investment strategy. The second requirement is that the program cannot use data from the future to assist in decision-making, as this would negate the project's intent. Finally, the program is prohibited from allowing a user to interfere.

Given these project constraints, we decided to develop a computer program that would provide insight into which stocks to invest in and when to buy and sell specific stocks in order to make a profit. This software uses machine learning to recognize patterns from historical stock data. The software then applies what it has learned to buy and sell stocks in a profitable manner. During our first semester of this course, we were pleased to have developed a program that was successful at meeting these project constraints. We created a machine learning program that uses a recurrent neural network to make predictions of future stock value with a reasonable accuracy. The program developed in the first semester used historical stock data from the Wharton Database which is split into two data sets; training and test. The program then made a training model from the training data set, which could then be applied to the test data.

Goals

One of the goals we set for the project was to create a program that could make stock trading decisions and give insight to the user to be able to reliably make a profit. Another goal we had set for the project, specifically in the second semester, was to make our model more generalized. Making a model more generalized means training it on the data of many different stocks. In concept, this should make the machine learning program smarter, and more applicable to a wide range of stocks. The third goal we had set for our project this semester was to increase the accuracy of the model. Having an accurate model is one of the most important aspects of any machine learning program, and this is especially true when trading stocks. The more accurate the model is, the better the chance of profit.

Project Background

Stocks and the Stock Market

The stock market is a marketplace where investors can buy and sell stocks, which reflect a piece of a company's ownership. Any stock in the market changes in price all of the time, with no apparent predictability. This is what makes stock trading so difficult; it's always a gamble because there is no exact science to predicting whether a stock will rise or fall. Technical indicators are one way that traders try to grasp stocks and their patterns. Technical indicators are stock-specific mathematical equations that provide more accurate details. Some important technical indicators that we would later use are the price, volume, simple moving average, exponential moving average, and moving average convergence-divergence (MACD).

Technical Analysis and Trading Strategies

In an attempt to benefit from stocks in the stock market, several trading strategies have been created over the years. The majority of well-known trading strategies are focused on technical analysis, which is a trading method that evaluates certain trades by analyzing a variety of statistical trends. These statistical trends can be analyzed by looking at certain variables such as volume and price. Technical indicators are signals produced by a stocks' price and volume that analysts may use when doing technical analysis. In general, technical analysis is the process of recognizing trends and patterns that can help investors decide whether to purchase, sell, or keep a stock. Algorithmic trading is a trading process that follows a set of rules or algorithms, and is executed by a computer program. Algorithmic trading is beneficial because computers can make decisions and execute trades much faster than humans.

Project Research

Python and Libraries

We decided to write the code for our software in a commonly known programming language called Python. Python is a high-level programming language that is object-oriented and is dynamically-written which sets it apart from many other programming languages. Python also has a user-friendly syntax which makes learning the language much easier and quicker, especially for those that have previous coding experience from any language. Python also supports the use of packages and libraries which contain various useful functions. Currently, there are over 137,000 different

Python libraries all with unique functions, mainly which are used for machine learning and data visualization.

Google Colab

This semester we decided to move our code from Jupyter Notebook to Google Colab. These are both integrated development environments that allow the user to write and test Python code, but we switched over to Google Colab for a few main reasons. Google Colab, short for Google Colaboratory, allows multiple users to collaborate on shared code very easily, by taking advantage of Google Drive. Users can write and test Python code all online, without needing to install any external applications. The code is automatically saved periodically which is instantly updated on Google Drive. This made our progress much quicker and more efficient than it was during the first semester. Another benefit of Google Colab is the ability to access additional files that are posted on Google Drive.

Tensorflow and Keras

On the topic of Python and libraries, the main library we used for the machine learning project is called Tensorflow. Tensorflow is an open source library that contains a variety of functions all which center around machine learning. These are the functions that we used to develop and train our machine learning models. Tensorflow is also integrated with Keras which is a deep learning application programming interface, or API. Keras is a high level API, which simply means that the functions are more generic and simple to implement. Using both Tensorflow and Keras, the user can experiment with a variety of machine learning models such as feedforward neural networks, recurrent neural networks and the binary classification.

Machine Learning

Machine learning is a way of programming computers similar to the human brain, so they are able to learn for themselves. In other words, the program is able to learn without explicit directions. We chose to use machine learning for this project because the stock market is a perfect application of this type of program. We wanted a program that could use a large amount of data as an input to be able to learn from, and the stock market is perfect for this task. In our case, we developed a machine learning program for stocks, which is able to recognize patterns in past stock data and use this information to make future stock market predictions. There are many different types of machine learning models, all of which work in slightly different ways and have different benefits and drawbacks. For our project we decided to create a binary classification model.

Artificial Neural Networks

An artificial neural network, or ANN, is a computing system that resembles the human brain. The human brain contains about one hundred billion neurons, and through connections of these neurons, we are able to learn. In a way similar to the human brain, the artificial neural network is able to simulate the human brain and learn in the same way. Each neuron in an artificial neural network has one or multiple binary inputs with specific weights which are then passed through a function to give a singular binary

output. A neuron is only used when a certain number of the input neurons are triggered, which is how the model “learns”. One specific artificial neural network that we studied was the feedforward neural network. A feedforward neural network is a neural network where the connections between the neurons only travel one direction, which is forward. In figure one we can see an example of the output of each layer traveling to the next layer.

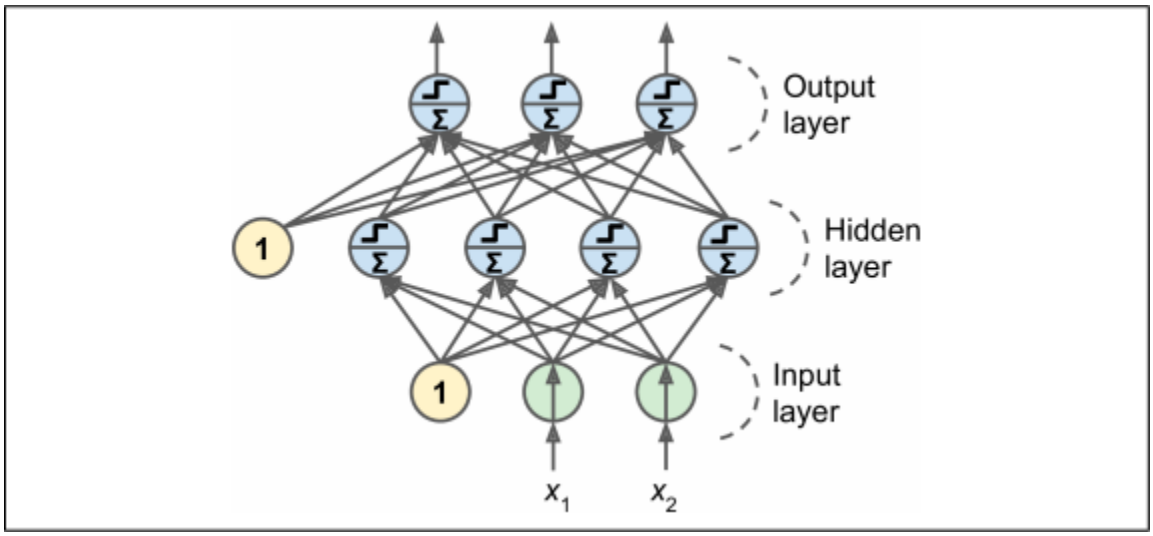


Fig. 1 (Feedforward neural network from Hands-on-Machine-Learning)

Another type of neural network that we studied is called a recurrent neural network. The recurrent neural network is derived from the feedforward neural network but has one main difference which alters the way the network behaves. A recurrent neural network has feedback loops and backpropagation, hence the name “recurrent neural network”. As seen in figure 2, a feedback loop essentially allows the output of a certain layer to be sent backwards in the network, which is more similar to the way the human brain works. The other differentiating feature from a feedforward neural network is the use of backpropagation. Backpropagation is used to help the model identify which weights are causing the error. This allows the weights to be adjusted to get closer to their ideal values for maximized accuracy.

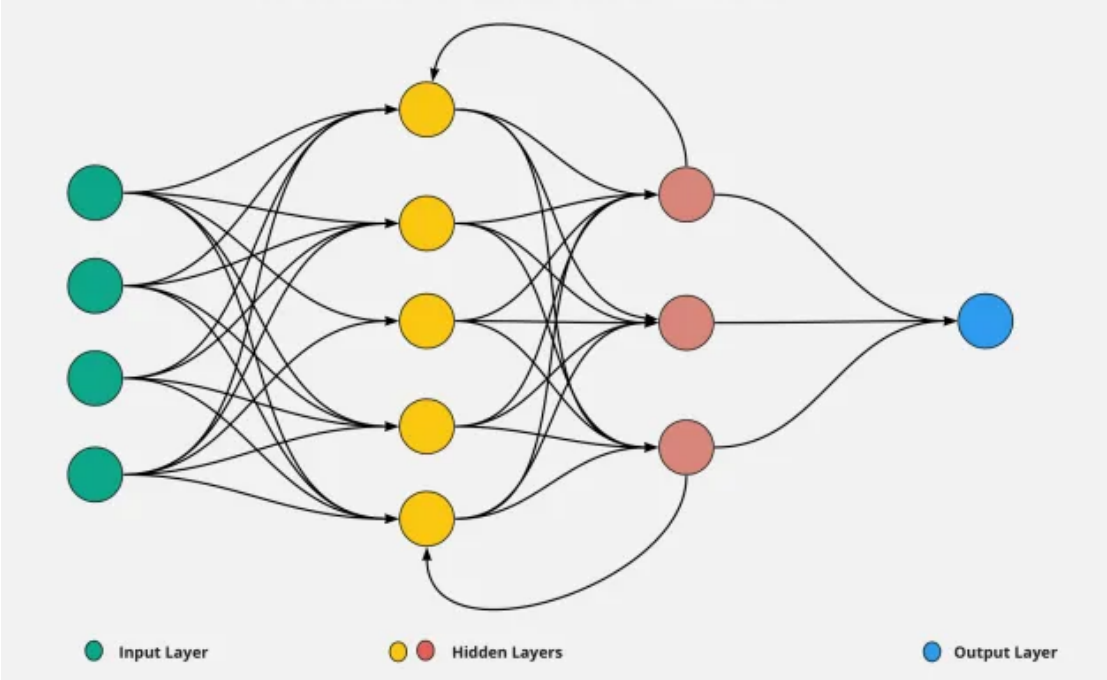


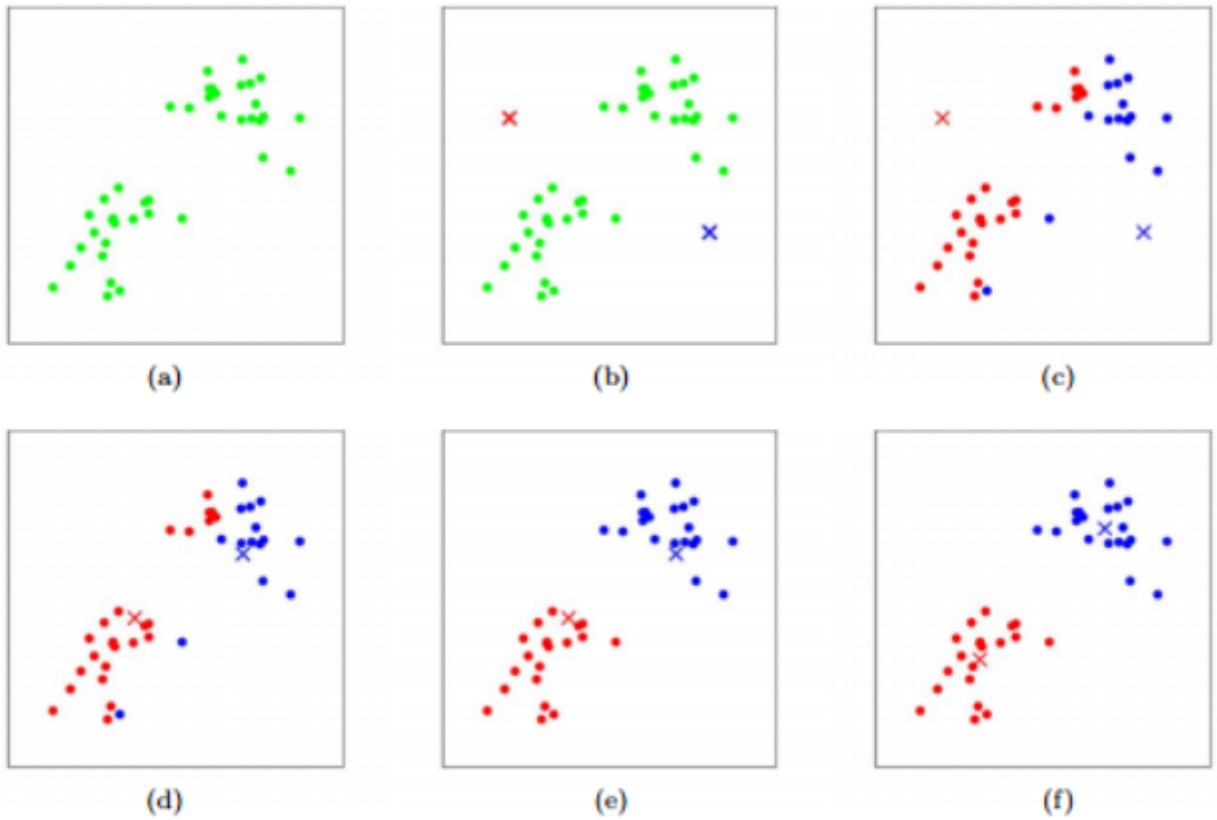
Fig. 2 (Recurrent neural network from Dataaspirant)

K-means Clustering

K-means clustering belongs to the clustering algorithm class of unsupervised machine learning. Clustering is the task of identifying similar instances and assigning them to groups of similar instances known as clusters. In short, clustering deals with finding a structure in a collection of unlabelled data. It is a great tool for data analysis, semi-supervised learning, dimensionality reduction, and more. Unsupervised learning is a machine learning technique in which the model does not need to be supervised, instead, the model works on its own to discover patterns and information, primarily dealing with unlabelled data. Essentially, the system tries to learn without a teacher.

The basic idea behind k-means consists of defining k clusters such that total within-cluster variation —the error— is minimum. The k-means algorithm searches for a predetermined number of clusters within an unlabelled multidimensional dataset. Without being given neither the labels nor the centroids, centroids are assigned randomly by picking k clusters at random and using their locations as centroids. Then, the clusters are labelled, centroids are updated, and so on and so forth until the centroids stop moving. The algorithm is guaranteed to converge to a finite number of steps, however it is not guaranteed to converge to the right solution. Furthermore, different runs of k-means —each with randomized k clusters— will result in slightly different clustering assignments. Therefore it is sometimes necessary to run the algorithm several times to avoid sub-optimal solutions and find a run that has the best separation, or the lowest total sum of within-cluster variations across all k clusters. Instances do not always have to be picked at random, however, as there are algorithms to aid in finding the optimal number of clusters for the given data.

The computational complexity of the k-means algorithm is generally linear with regards to the number of instances, the number of clusters, and the number of dimensions. However, this is only true when the data has a clustering structure. If it does not, then the complexity can increase exponentially with the number of instances. However, this rarely happens in practice, and k-means is considered to be one of the fastest clustering algorithms.



(Fig. 3 from Stanford CS221)

K-means algorithm. Training examples are shown as dots, and cluster centroids are shown as crosses. (a) Original dataset. (b) Random initial cluster centroids. (c-f) Illustration of running two iterations of k-means.

This method is used when the datasets that are being used are not very large and not labeled. Finding or building large, labeled datasets can be time consuming and expensive. This method relies on a few ways to organize the data but one of the most popular is clustering. In this method, the algorithm creates clusters of similar data points and when the algorithm receives new data points, it can predict which cluster that point will belong to. The K-Means algorithm is pretty simple and takes in an array that has six features. That means that the clustering algorithm will build clusters in a six-dimensional space which may sound complicated but the K-Means algorithm is well suited for this task. Once the algorithm builds the clusters, we can try and predict new data points. The key here is to feed the model data that it has not seen before. This way we can

obtain the true accuracy for our model. An important aspect of K-Means is that the user has to define the number of clusters for the algorithm and the algorithm will determine where those are located within the given dimensions of the data. Our model is looking for two main clusters, which are increase in returns and decrease in returns. Thus, a good starting point for us was to use two clusters in our model. Unfortunately, the accuracy of this model was not great. About 50% of cluster one and two both had positive returns. This means that the algorithm is guessing about where the points are located, or that the clusters are not separated according to return on investment. Thus we increased the number of clusters. However, we obtained similar results for the number of clusters equal to three and four. We have faith in this type of model but there is more work to be done here. This will be discussed in the future work and conclusion sections of this paper.

Literature Review

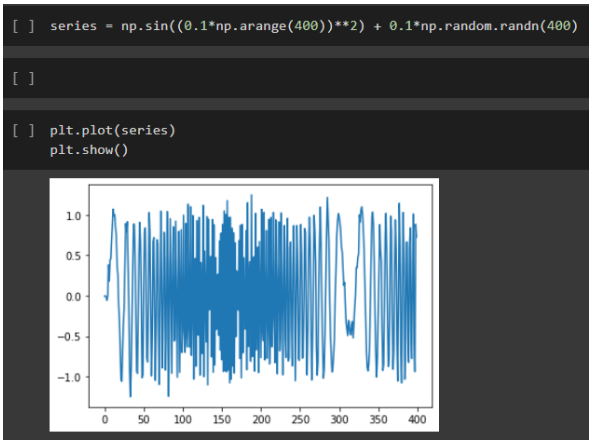
We utilized a handful of books by publisher O'Reilly as sources of information for our extensive research on machine learning algorithms.

1. *Hands-On Unsupervised Learning Using Python: How to Build Applied Machine Learning Solutions* by Ankur A. Patel (Patel).
2. *Hands-On Machine Learning with SciKit-Learn, Keras & TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems* by Géron Aurélien (Géron).

LSTM

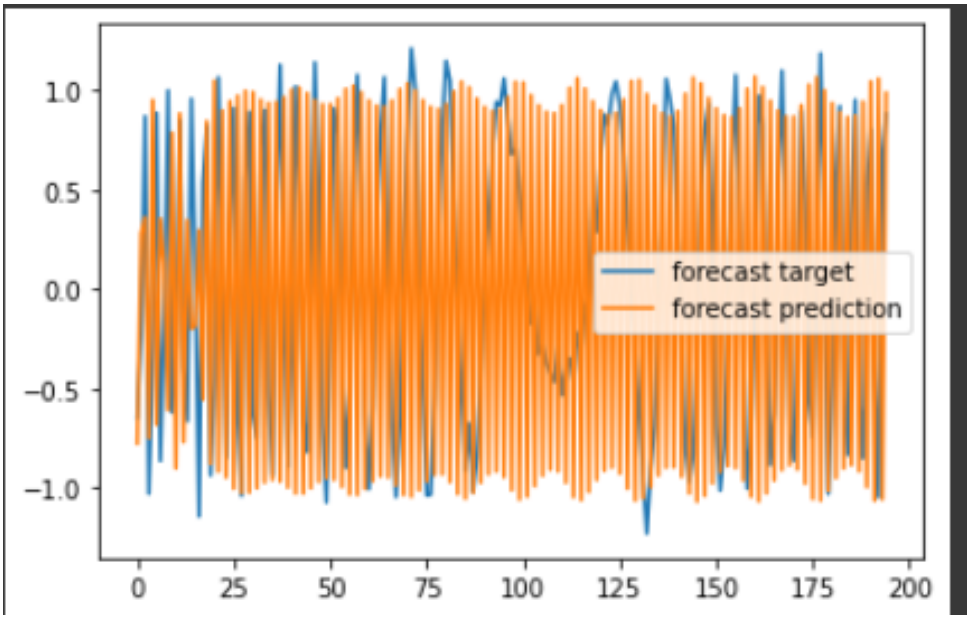
Long Short-Term Memory models seemed to be a fantastic approach to predicting stock data. It is a recurrent model that works well with time-series data. However upon further analysis this was decided to not be a good approach. This was determined by evaluating an LSTM model's accuracy in predicting the value of a variable frequency cosine wave with noise. This should be a relatively easy task because this is a known function. The only difference here is that there is noise added. It can be shown that an LSTM model performs very well on simple function predictions such as regular cosine but once frequency is altered, noise is added, and multi-step forecasting is required, the best the LSTM model can do is memorize the data, and it cannot predict. Figure A is an image of the function that we want to be able to predict while figure B is an image of the LSTM predictions for multi-step forecasting. These results led us to take a different approach this spring.

A.



(Fig. 4 Output of code constructed in Google Colab)

B.



(Fig. 5 Output of code constructed in Google Colab)

Binary Classification

Binary classification is one of the simplest uses of an artificial neural network. This type of neural network takes an input and predicts which group that data point belongs to. As the name suggests, this type of classification is binary so it only has two groups. Since predicting price seems to be difficult for our models, we decided to just try and predict an increase or decrease in a given time frame. We calculated the return on investment over the course of a given time and set a threshold to determine how much we wanted it to move. This means that if we set the time frame to 5 days and the threshold to 5%, if the stock only went up 2%, we added a 0 to the end of that data point in the dataset.

Examples of binary classifications include Email spam detection (spam or not), churn prediction (churn or not), conversion prediction (buy or not). Typically, binary classification tasks involve one class that is the normal state and another class that is

the abnormal state. For the Email example, “not spam” is the normal state and “spam” is the abnormal state. The class for the normal state is assigned the class label 0 and the class with the abnormal state is assigned the class label 1. At this time, we realized that the changes in the stock market can also be expressed by classifiers. In the project, 0 represents a bear market, and 1 represents a bull market. Using historical data for training and learning to predict future stock market fluctuations is theoretically possible. In order to enable it to successfully predict the stock market, we need to go through the following steps.

1. Prepare company stock data.

In order to successfully predict the stock market, we first need to prepare company stock data. Like other model training, in order for the algorithm to run efficiently, we need a lot of data.We have selected some representative stocks of mainstream companies such as Tesla, Google, and Microsoft. The stock data includes stock price and trading volume every day from the time the stock was listed to the near future. In addition to the stock price, we need other price-related eigenvalues. What is an eigenvalue? For example, if we want to predict the survival probability of passengers on the Titanic, what kind of data characteristics should we prepare? The answer is passenger A's age, height, weight, seat number, number of accompanying persons, etc.These can all help the machine to learn. Obviously, the more dataset is better, the larger the scope, the better. In our project, the following data is required.

Variable:	Definition:
Price	Price of Bid/Ask Average
Volume	Amount of an asset or security that is traded over the period of a day
Simple Moving Average (SMA)	
Exponential Moving Average (EMA)	
Trading	Volume/Shares outstanding
Bear or Bull Market	Bull market is 1, Bear Market is 0

After preparing the data set, we can upload it to colab for use in the following steps.The use of Colab Notebook is very convenient. We can write instructions directly in the code part（Figure 6）to directly access the data set of the cloud drive.

```
from google.colab import drive
drive.mount('/content/drive', force_remount=True)
```

```
Mounted at /content/drive
```

```
#set path to the data we want to train
path = '/content/drive/Shareddrives/ECE 448 FA 2020-SP2021/449 (SP21)/Data/data (total shares)/amzn_edited.xlsx'
```

(Fig. 6 Code constructed in Google Colab)

2. Data processing and classification

Next, in order for the machine to recognize these data sets, a series of works are required.For example, binary classification models are often used to distinguish image content.Pictures and number's dataset are not the same. The computer language only has numbers, so we need to convert the text to a type that the computer can recognize.It is well understood in our project.These contents are all suitable data sets prepared for machine learning.This has to go back to our broad classification of machine learning.

Classification tasks belong to supervised learning.The characteristic of supervised learning is to be marked.For example, give you 1,000 pictures of cats and 1,000 pictures of dogs, throw them together without marking them. This way you can't do classification.Although you can let the machine learn the characteristics of different pictures, let it distinguish the pictures.But this is called clustering, which is unsupervised learning.It is unknown what characteristics the machine separates the pictures.The result you want is to put cats in one category and dogs in the other category.But when the machine grabs the features, it may prefer to distinguish them by color.As a result, white cats and white dogs were placed in one category, and yellow cats and yellow dogs were placed in another category. The result is very different from what you want. Therefore, to classify, you must have a label. Labeling is a professional and complicated labor.Fortunately, our data does not require marking. The reason is that we can judge a bull market or a bear market by comparing daily prices.This is different from the example just mentioned in the picture classification.The data sets we prepare are all in Excel table format, so the first step we need to extract the columns we need.Then replace them with new names.

```
def  make_data(prepped_data):
        charecter  =  prepped_data.iloc[:,1:7]
        charecter.columns=['one','two','three','four','five','six']
        new_charecter  =  charecter.copy()
        new_charecter.columns=['before_one','before_two','before_three','before_four','before_five','before_six']
```

(Fig. 7 Output of code constructed in Google Colab)

Create a new dataframe called df, first set all the first row to 0 (representing the first day of the data).
df_final is adding the previous new_charecter
df_final is the data of the previous day, so after merging with the character dataframe, you can calculate the difference in one row.

```
df  =  pd.DataFrame(columns  =  ['before_one','before_two','before_three','before_four','before_five','before_six'])
df.loc[0]  =  [0,0,0,0,0,0]
df_final  =  df.append(new_charecter)
df_final  =  df_final.reset_index(drop=True)
```

(Fig. 8 Output of code constructed in Google Colab)

The last part is the labeling part. final_data['check one'] These checks are used to calculate (current period-the previous day), so the calculated value has two distributions greater than 0 or less than 0
final_data['P/N check one'] These P/N writes are used to further process the above final_data['check one'], if the value of final_data['check one'] is greater than 0, it will be assigned to It is 1, otherwise it is assigned 0

3. Training model and evaluation

Next, we need to build a model and use the data we just processed for training.Before learning, you need to divide all the data you have at hand into 3 categories: training set, validation set, and test set.The training set allows your model to learn how to use the current hyperparameters (such as the number of layers of the neural network, the number of neurons in each layer, etc.) combination, and label the results as much as possible. The verification set exists to allow you to compare different hyperparameter choices, which set is more suitable for the current task. It must use unused data in the training set. How does this model perform?

Of course you need other data to judge. This is why you have to divide up another test set. So in our project, train_test_split will be used to split the data set into an 80% training set, 10% test set, and 10% validation set. (Figure 9)

```
from  sklearn.model_selection  import  train_test_split

def  train_test_val_split(df,ratio_train,ratio_test,ratio_val):
        train,middle  =  train_test_split(df,test_size=1-ratio_train)
        ratio=ratio_val/(1-ratio_train)
        test,validation  =train_test_split(middle,test_size=ratio)
        return  train,test,validation
```

```
x_train,  x_test,  x_val  =  train_test_val_split(X,0.8,0.1,0.1)
y_train,  y_test,  y_val  =  train_test_val_split(y,0.8,0.1,0.1)
```

(Fig. 9 Code constructed in Google Colab)

If you understand the overall process of the binary classification problem and make a good choice of model, then the actual machine learning process is very simple.

For most common machine learning problems, you can use Scikit-learn to call the model.Note that in fact, every model has the need for parameter setting. But for many problems, you can use the initial parameters, which can bring very good results. Among the models we built, keras is used here, and the activation functions used in the model are relu and sigmoid.The number of layers and parameters of the model can be adjusted and changed. After the model is established, an optimizer step is added, called optimizers, whose main function is to optimize the model.(Figure 10)

```
import numpy as np

from keras import models
from keras import layers

model = models.Sequential()
model.add(layers.Dense(100,activation='relu',input_shape=(6,)))
model.add(layers.Dropout(0.3, name='lstm_dropout_0'))
model.add(layers.Dense(50,activation='relu'))
model.add(layers.Activation('sigmoid', name='sigmoid_0'))
model.add(layers.Dense(1,activation='sigmoid'))


from keras import optimizers

model.compile(optimizer=optimizers.RMSprop(lr=0.001), loss='binary_crossentropy', metrics=['accuracy'])
```

(Fig. 10 Code constructed in Google Colab)

For the Keras algorithm, if you need deep learning, we can need some GPU hardware support, so we will choose to run on Google Colab, so that we can use cloud computing for free.

4. Evaluation model.

Train the model for 10 epochs in mini-batches of 8 samples. This is 10 iterations over all samples in the x_train and y_train tensors. While training, monitor the model's loss and accuracy on the samples from the validation set（Figure 11）

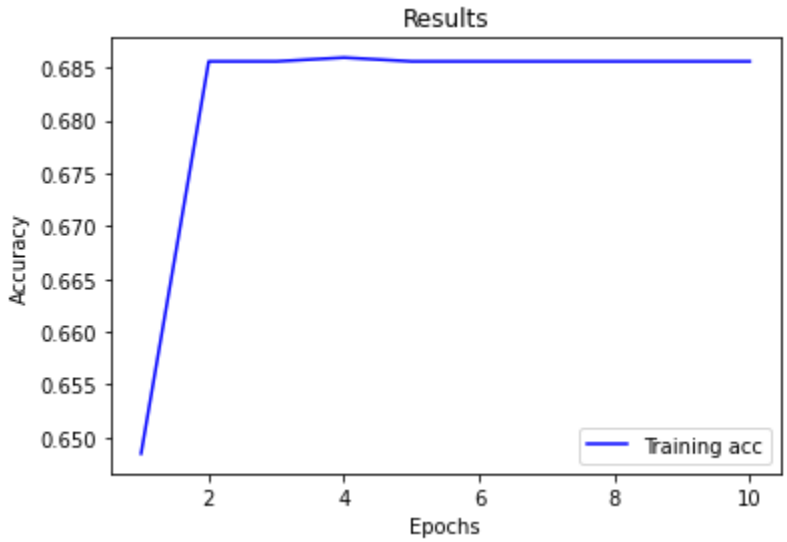
```
history = model.fit(x_train,y_train,epochs=10,batch_size=8)
history_dict = history.history
```

```
Epoch 1/10
350/350 [==============================] - 1s 1ms/step - loss: 0.7050 - accuracy: 0.5702
Epoch 2/10
350/350 [==============================] - 0s 1ms/step - loss: 0.6181 - accuracy: 0.6926
Epoch 3/10
350/350 [==============================] - 0s 1ms/step - loss: 0.6305 - accuracy: 0.6814
Epoch 4/10
350/350 [==============================] - 0s 1ms/step - loss: 0.6358 - accuracy: 0.6781
Epoch 5/10
350/350 [==============================] - 0s 1ms/step - loss: 0.6415 - accuracy: 0.6716
Epoch 6/10
350/350 [==============================] - 0s 1ms/step - loss: 0.6337 - accuracy: 0.6827
Epoch 7/10
350/350 [==============================] - 0s 1ms/step - loss: 0.6271 - accuracy: 0.6900
Epoch 8/10
350/350 [==============================] - 0s 1ms/step - loss: 0.6342 - accuracy: 0.6804
Epoch 9/10
350/350 [==============================] - 0s 1ms/step - loss: 0.6303 - accuracy: 0.6862
Epoch 10/10
350/350 [==============================] - 0s 1ms/step - loss: 0.6327 - accuracy: 0.6735
```

(Fig. 11 Output of code constructed in Google Colab)

The loss part is the loss value calculated by our pre-set loss function; the accuracy is the evaluation result of the model based on the given label on the data set. There is a relationship between loss and accuracy, but they are not equal; from the perspective of the model, loss is the measurement standard; but a good model must eventually be

brought to accuracy. Here we can plot a graph of the accuracy of the model to observe the training results. (Figure 12)



(Fig. 12 Output of code constructed in Google Colab)

We can directly observe that the accuracy of the model can reach about 0.68. Here, the training samples we use come from the listed company Amazon. If we try to use other companies as training samples, the results we get are also close.

5. Predict target data

Ultimately, what we need to achieve by the algorithm is to predict the stocks of other companies. At this time, we need to prepare the corresponding prediction data set, and the most important thing is that our prediction data set needs to have the same Excel format as the training data set, so that we can use the model to make predictions. In other words, we need to use the data structure of company A's stock to predict company B's stock.As we mentioned above, after training with 1,000 photos of cats and dogs, we took out two or three new photos for the machine to identify and verify the model.In our project, we need to upload the predicted stock file to colab.Then make predictions after the same data processing. (Figure 13)

```
goog_y,goog_edited = getFeature(5,0.05,1,'goog_edited.xlsx')
goog_edited = goog_edited[goog_edited.index >= goog_y.index[0]]
goog_df = pd.read_excel('goog_edited.xlsx', index_col=0,parse_dates=True)
pred_goog = model.predict(goog_edited).round()
print ("Accuracy goog: ")
print(accuracy_score(aapl_y['N/X'],pred_aapl)*100)
```

(Fig. 13 Constructed in Google Colab)

And we can upload multiple stock data sets at the same time to make predictions at the same time. In the project, we predicted the stocks of 5 companies at the same time and successfully obtained the correct rate of prediction based on historical data.

Companies:	Tesla	Nvidia	Microsoft	Apple	Google
Accuracy	72.1480	71.0774	68.99114	68.9339	68.5634

Future Work

Given the limited success of our models and our literary research, there are many opportunities for future research. Two main areas for research are different models and combinations of models. The different models that are of interest are transformer models and convolutional neural network models. Transformer models are relatively new and are generally used for natural language translation. This model relies on encoders and decoders to code in context to the data. This could be a game changer in prediction algorithms as it might allow for the algorithm to apply context to some change in data. Convolutional neural networks (CNNs) are mainly used in image processing. Before, we used numerical data to try and predict the next price, however, with CNNs we can look at the image created by the numerical data and try and find patterns. This would require a bit more time and effort because there is no pre-labeled database. So someone would need to grad the data, create the images and then label the data in order to apply this method. The interesting thing with CNNs and financial data is that we can adjust the resolution of the data to obtain different graphs. This means that the graph with a 1-day resolution will look completely different from a graph that has a 1-minute resolution. The final area of future work is the combination of models. [12] and [13] have had limited success in predicting stock prices with combining ANNs with more simple SVMs. Basically, what they did was they put an SVM at the output of a simple ANN to further manipulate the data. This could be an area of more research combined with the methods mentioned above.

Conclusions

At the end of the day, we have merely skimmed the surface of all the possibilities that this field has to offer. We began by researching machine learning and how to implement machine learning to solve simple problems. Next we built a database that we could then use to train our own machine learning algorithms. After that we each set out to try and build our own models so that we could compare results. We have built feedforward models, recurrent models (LSTM), clustering models and binary classifiers, all in an effort to produce a model that has an acceptable accuracy. In the end, Yuhua's binary classification model had the best results. In hindsight, it may have been better for all of us to work on a similar model together so that we could do more hyper parameter tuning or spend more time implementing new model ideas such as the combination of models discussed in the future work section. We also believe that the unsupervised learning approach may be a better option however, more time is needed to research that model. Overall, this was a successful semester and also a great learning experience for the entire group.

Engineering Ethics

Engineering is an important and learned profession, and as members of this profession, engineers are expected to exhibit high standards of integrity and honesty.

Engineers should strive to serve the public interest, as engineering has a direct and vital impact on the quality of life for all people. In addition, engineers are encouraged to follow the principles of sustainable development in order to preserve and protect the environment for future generations. Therefore, engineering requires equity, fairness, honesty, impartiality, and must be dedicated to the protection of public health, safety, and welfare. Engineers should accept personal responsibility for their professional service and activities, give credit for work that is not their own, and overall avoid all conduct or practice that deceives the public. All in all, engineers must perform under a standard of professional behavior that requires adherence to the highest principles of ethical conduct.

The principles of ethical conduct are as follows:

1. Prioritize the health, safety, and welfare of the public.
2. Perform services only in areas of their competence.
3. Issue statements to the public in a forward and objective manner.
4. Interact with each employer and client as faithful and trustworthy agents.
5. Avoid deceptive acts.
6. Conduct honorably, ethically, lawfully, and responsibly, so as to enhance the honor, reputation, and usefulness of the profession.

Having worked on a capstone project that focuses on finance —specifically, a model to predict stock market price based on historical data— it is of paramount importance to understand and follow engineering ethics principles, as we are dealing with a representation of ownership claims on business. In some cases, a trader's whole livelihood revolves around the stock market, therefore there is no room for error regarding the businesses and trader's finances.

The Need For Engaging In Lifelong Learning

The STEM workforce accounts for more than 50 percent of the nation's sustained economic growth. Engineering, by nature, requires that its practitioners continue learning new things, even after their formal education has ended. With the world rapidly advancing scientifically and technologically, the half-life of an engineer's knowledge is decreasing. Even engineers with extensive experience are vulnerable to becoming outdated. Moreover, new fields such as nanotechnology, biotechnology, information technology, and genetics are constantly emerging, and many problems require engineers to work—and therefore learn—across boundaries of engineering disciplines. Therefore, stimulating lifelong learning improves the knowledge base of engineers and our capacity for competition and innovation.

The Need For A Broad Education

In today's rapidly evolving engineering landscape, we have an increased obligation to transform the undergraduate educational experience from the traditional curriculum to a broader foundational experience for life-long success. Mastery of the technical aspects of engineering must remain at the curriculum's core, adding new dimensions —such as our capstone project— will better prepare students for the world of the future. A broad education across many fields of study can heighten the

professional awareness and performance of a practicing engineer. An engineer who has had a narrow education and only has technical knowledge learned from engineering classes may create a design that looks perfect on paper, but without any practical experience, the design may not be feasible to execute. Being a STEM leader this century requires a broad education that goes beyond the classroom and the laboratory, and continued professional development is imperative to ensure public safety, a competitive national economy, a sustainable environment, a respected profession, and a fulfilling career.

The capstone project has sharpened our technical skills regarding Python programming and logical thinking, as the project required for us to do extensive research on and to write numerous algorithms for our machine learning model. Additionally, working on the capstone further enforced the principles of engineering ethics and broadened our understanding of what it means to be a professional, as the project provided us with practical experience by resembling a real business environment where we were considered to be engineers working on a multidisciplinary team with an advisor to oversee our work.

Works Cited

[1] “Code of Ethics.” *Code of Ethics | National Society of Professional Engineers*, www.nspe.org/resources/ethics/code-ethics.

[2] Géron Aurélien. *Hands-on Machine Learning with Scikit-Learn and TensorFlow Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly, 2019.

[3] “Important Announcement.” *Engineering Education for the 21st Century | College of Engineering*, www.bu.edu/eng/about/dean-lutchen/engineering-education-for-the-21st-century/ .

[4] Model, Slide. “Stock Market PowerPoint Template.” *SlideModel*, slidemodel.com/templates/stock-market-powerpoint-template/.

[5] Patel, Ankur A. *Hands-on Unsupervised Learning Using Python: How to Build Applied Machine Learning Solutions from Unlabeled Data*. O'Reilly, 2019.

[6] Piech, Chris. *CS221*, stanford.edu/~cpiech/cs221/handouts/kmeans.html, 2013.

[7] “Read ‘Lifelong Learning Imperative in Engineering: Sustaining American Competitiveness in the 21st Century’ at NAP.edu.” *National Academies Press: OpenBook*, www.nap.edu/read/13503/chapter/2#4.

[8] Valkov, Venelin. “Making a Predictive Keyboard Using Recurrent Neural Networks - TensorFlow for Hackers (Part V).” *Medium*, Medium, 16 May 2019, medium.com/@curiousily/making-a-predictive-keyboard-using-recurrent-neural-networks-tensorflow-for-hackers-part-v-3f238d824218.

[9] Digication EPortfolio :: James Matthews :: 9 - Broad Education, alaska.digication.com/james_matthews/9_-_Broad_Education.

[10] Das, K. (2020, December 15). How recurrent neural network (rnn) works. Retrieved April 25, 2021, from https://dataaspirant.com/how-recurrent-neural-network-rnn-works/

[11] Brownlee, J. (2020, August 19). 4 types of classification tasks in machine learning. Retrieved April 25, 2021, from https://machinelearningmastery.com/types-of-classification-in-machine-learning/

[12] Madge, Saahil, and Swati Bhatt. “Predicting Stock Price Direction Using Support Vector Machines.” 2015.

[13] Pahwa, Nirbhey, et al. “Stock Prediction using Machine Learning a Review Paper.” *International Journal of Computer Applications,* vol. 163, no. 5, 2017 pp.36-43