05/21

05/10/2021

Please use this template. Retain the gray text. Your new materials—in black 12-point Times New Roman—should not exceed 5 additional pages excluding references and figures. Use the Appendix for bulky material that will be read on an as-needed basis only. Note the evaluation criteria, and leave plenty of time for editing.

# ASSIGNMENT 2P: PROJECT PROPOSAL PLUS

Aidan Duffy

## 2.0 WHAT’S CHANGED (if applicable)

Provide no more than a page of 12-point type explaining what has been changed or added since assignment 1P (if you selected it). Include in this whether and how the material in module 2 influenced this, or refer to reading that you did in working on this assignment (#2).

N/A – did Assignment 1T.

## 2.1 SUMMARY DESCRIPTION, VERSION 2 (as applicable)

One- or two-paragraph overall description of your proposed term project.

I want to develop a machine learning program, called CRYPTOFORECAST, that will help develop price predictions for the cryptocurrency market. Many programs are already tailored to predicting fluctuations in the stock market, and I did not want to reinvent the wheel. Additionally, given the rise of the popularity of investing cryptocurrencies like Bitcoin and Ethereum (and yes, even Dogecoin!), CRYPTOFORECAST will use the same fundamental financial indicators and technical analysis but rely on learning from a dataset that analyzes the largest tokens currently in circulation. This program may help future investors in DCA, or dollar cost averaging, to lower their cost bases, as well as realizing when the end of a “bull run” may be approaching.

## 2.2 I/O EXAMPLES, VERSION 2

At least two specific examples of projected output for designated input. You will not be held to this exactly.

Here is the I/O demonstrating the price history of a given input token:

Graphical user interface, text, application

Description automatically generated

## 2.3 FUNCTIONAL REQUIREMENTS, VERSION 2

Separate your requirements into two approximately even categories (select modest “definite” requirements, otherwise “nice-to-do”). This organization allows you to first attain readily do-able goals without getting bogged down, and then move on to other goals as you are able. State requirements in declarative language as advised in assignment 1. Giving each requirement a label (e.g., “(Recognize 0-9):”) helps with clarity and readability.

Your response replaces this.

### 2.3.1 Definite Functional Requirements (first priority)

### 2.3.1.1 Definite #1: Read Datasets

CRYPTOFORECAST shall read in and format all of the datasets.

### 2.3.1.2 Definite #2: Filter and Normalize Datasets

CRYPTOFORECAST shall sort data, apply necessary filters, and normalize the filtered datasets.

### 2.3.2 Nice-to-do Functional Requirements (second priority)

### 2.3.2.1 Nice-to-do #1: Integrate Visualizer for Datasets

CRYPTOFORECAST shall provide visualizations and graphs for the price history of cryptocurrencies in the datasets.

## 2.4 V2: HOW SUCCESS WILL BE ASSESSED

Explain, as specifically as possible (quantification is ideal) how success of the project should be assessed.

Luckily, given the short feedback cycle of financial markets, success can be assessed almost immediately. I will be able to cross reference my model’s price predictions daily with actual market prices.

## 2.5 V2 TECHNOLOGY EXPLANATION

Explain what two specific machine learning technologies you intend to use--and exactly why you feel they apply to your particular project. One of the two may be emphasized as the implementation and the other as an alternative or as a complement—discussed but not implemented if need be. Include a discussion of neural nets, whether you select them or not.

prices. After doing research on similar projects that have already been completed, I will attempt to utilize a recurrent neural network, or RNN. More specifically, I will attempt to implement an LSTM, or long short term memory, setup. This type of neural network was selected most often because of its ability to store past information for sequence prediction problems, making it a most powerful tool in a machine learning developer’s toolbox. Since the dataset will be sequential, the data from one hour or day is directly related to the data point from the previous unit of time, again reaffirming the importance of this style of neural network. As a proposed alternative, in my research I found an algorithm called “XGBoost”(Extreme Gradient Boosting), which a tree based supervised learning algorithm. It is often used as well as it is very efficient. The main reason it is an alternative is because all of the data would first need to be preprocessed and transformed into a supervised learning problem first.

## 2.6 CODE EXAMPLES

Show fragments of code execution. For example, if you are using TensorFlow, show that you have run some relevant code. The more relevant to your project, the better. This does not need to complete at this stage.

This is the code addressing functional requirement number one:

Table

Description automatically generated

Here is code snippet 2 that addresses the graphing functionality of price histories:

Graphical user interface, text, application, email

Description automatically generated

## 2.7 DATA SOURCES V2

Explain whether or not your project requires data. If so, describe were you will obtain it. Be careful about this because you won’t have a project if it needs data and you have to spend too much time hunting and gathering it.

This project most definitely requires a dataset, and as of writing this, I am using one acquired from Kaggle [here](https://www.kaggle.com/sudalairajkumar/cryptocurrencypricehistory). If I determine this data is insufficient, there are ample other sources of data available on Kaggle that I will be able to utilize.

## 2.8 REFERENCES FOR PROPOSAL V2

Fill in, and also cite each of the following (e.g., “[2]”) within the text. References can include specific places in the notes and textbook. You are free to include references used in the prior assignment version. Keep in mind that this “use of resources” is a whole evaluation criterion.

[1] <https://towardsdatascience.com/lstm-time-series-forecasting-predicting-stock-prices-using-an-lstm-model-6223e9644a2f>

[2] <https://cs230.stanford.edu/projects_winter_2020/reports/32066186.pdf>

[3] <https://medium.com/swlh/a-technical-guide-on-rnn-lstm-gru-for-stock-price-prediction-bce2f7f30346>

## 2.9 Evaluation of Assignment 2



2.10 Appendix

# Assignment 4: Project Design + Implementation Preliminaries

Please limit this to 6 pages of 12-point text excluding figures, references, and appendices. This revision is your final view of the design prior to implementation (though you may still change it when you implement).

## 4.0 What’s New

Summarize what is new about your project since the last submission. This should reflect 15-20% of the total project effort.

Since my previous submission, I have continued designing the overall structure of the product and implementation changes as well as some new steps towards implementation. With regards to changes to the previous implementation, I refined the data normalization completed after everything is read-in and processed. In terms of new implementation, I integrated some averaging models for the prediction forecasting. This includes calculating the MSE, or mean squared error, for the standard average and exponential moving average, or EMA, which are both common metrics used in financial asset prediction models. I will discuss the actual design decisions in the 4.2 section.

## 4.1 Relevance of RNNs and GANs

Explain whether or not RNN’s and GAN’s could or should apply to your problem, and why/not.

RNNs apply to my project, as I will be implementing an LSTM, or long short-term memory, model. I found LSTM to be a useful form of the RNN in my research about past implementations of stock price prediction programs. It is primarily useful because of the ability to retain information about data over long stretches of time.

## 4.2 Design and Theory: Final Version

Describe the final pre-implementation version of the design of your proposed system. Use annotated diagrams. Explain the theory behind your design. Explain how the two technologies will interface or compare. The reader should understand how you plan to fit the pieces together. Show this at a high level, as well as providing as much relevant detail as you can. Include at least one (meaningful) figure.

Once the data is uploaded, the program will work as follows. Pandas will store all of the csv files’ data into pandas data frames. From there, some preliminary graphs will demonstrate the price history of selected cryptocurrencies. After this, pandas will split data into training and testing datasets. Once the datasets are created, CRYPTOFORECAST will normalize the data using a Min Max Scaler and utilize TensorFlow to implement an LSTM RNN.

**Note**: As I fell a bit behind in past weeks, I have not solidified all of the details for the specific parameters below (i.e. how many layers the neural network will have, but will be sure to include that information in the coming submissions).

The LSTM RNN will use several hidden layers. The RNN will need hyperparameters, like the number of hidden neurons for each cell or information related to the backpropagation through time for optimization. The BPTT hyperparameter will assist in determining how many continuous steps CRYPTOFORECAST will take into account for one optimization step. Additionally, CRYPTOFORECAST’s RNN will require standard inputs and outputs as well as parameters for the layers of the LSTM and a regression layer which would calculate the final prediction for the program.

Further, the program will need a tool to determine loss and an optimizer for the RNN. I will use the MSE for the calculation of loss, and I researched that “AdamOptimizer” is a recently developed optimizer that generally has been deemed as performing well. It is a stochastic gradient descent optimizer.

Once all of the data is reshaped by numpy, the model will create predictions using the testing dataset. After this, CRYPTOFORECAST will output a visualization to illustrate the predictions generated.

## 4.3 Tools: Final Version

Describe the tool(s) (e.g., TensorFlow) you will definitely use, or explain why you will build from scratch. Support the fact that you have reasonably investigated and tried out tools. Explain your choice. Show samples that make you and your reader reasonably confident of your choices.

TensorFlow will be one of the most important, along with pandas, numpy, and sklearn, tools that will be imported to complete the project. Pandas, numpy, and sklearn will be used for the standard data management, storage, and manipulation. I will be using pandas data frames to store the data, numpy for relevant operations, and sklearn for normalizing and preprocessing all of the cryptocurrency pricing data.

TensorFlow, specifically, is important has an API called “RNN API” that CRYPTOFORECAST will utilize to implement a time series model. As mentioned in my HW2 submission, one of the other tools that I discovered could be used was XGBoost. While I noticed that it processed the test and training datasets more efficiently than a TensorFlow implementation, upon more extensive research, I realized that XGBoost performs better when given additional features. While I mention in later sections below my intent to integrate a possible sentiment analysis aspect for the model, given my relative lack of experience in the field, I want to focus primarily on historical data points first and have a functioning program. Perhaps if I were to have done a project such as this one a few years down the line with more knowledge and experience, I would have leaned more towards XGBoost. However, the TensorFlow implementation of the RNN for CRYPTOFORECAST was easier for me to comprehend and ultimately take steps towards implementation. Additionally, given that the size of my current dataset is smaller, I wanted to prioritize accuracy and performance over efficiency. Further down the line in the project, however, I plan to revisit this decision for further analysis.

## 4.4 Challenges

Describe the challenges that your particular project faces. Avoid generic statements—focus on *your* project objectives. Explain how you intend to overcome them.

One of the largest challenges that faces this project, one that is separate from a typical stock price prediction machine learning program, is that the cryptocurrency market is both significantly more volatile than the stock market and can be swayed very easily by sentiment. As an example, the price of 1 BTC, which is meant to be the overall market indicator, fluctuates several hundred percent in a given year whereas the typical year for the S&P 500 rarely reaches double digits. I believe that since the whole market is volatile, that should be factored in properly from the LSTM RNN as the market has always behaved this way. I will need to ensure that even the direction of the price is correct in terms of increases and decreases, the scaling of the data is correct. If not, CRYPTOFORECAST will need to complete some post processing.

Further, single influential actors, such as Elon Musk, have the power to greatly influence prices of not only individual tokens but the whole market. For example, while Musk’s influence on Dogecoin is similar to his influence over his own Tesla stock, when his company stopped accepting BTC as payment for cars, the whole market fell 10% over the course of a few hours. These factors will need to factored in to make a more robust and accurate program. Therefore, adding a sentiment analysis stage would help improve the overall accuracy of the program. I will use Twitter as my main source of sentiment data as well as Google searches. CRYPTOFORECAST should be able to overcome the influencers with that.

One other issue that is more so specific to the cryptocurrency space is that individual asset owners, known as “whales” can own tens to hundreds of millions of USD worth of cryptocurrencies, and when these specific folks decide to sell (without the fanfare of someone like Elon Musk posting about it on Twitter) a massive amount of their assets, this can instigate a large dump or “flash crash” in the asset’s value. I hope to be able to factor in volume of transactions as well as the size of large purchases or sales, which can instigate panic sales. Also, given that sometimes these flash crashes can lead to a huge drop in price than immediately recover, CRYPTOFORECAST will eventually need to utilize a larger dataset as these massive fluctuations can occur over the course of ten minutes, whereas the current dataset gives daily price points.

## 4.5 Trade-offs

Describe trade-offs that you are making or may need to make, and their consequences. These can be divided into trade-offs given the duration of the term vs. those that would apply to a longer-term development period.

Short-term Trade-offs:

The most glaring trade-off for CRYPTOFORECAST is that, given the processing power of a single laptop and the shorter time given for the project, the dataset that illustrates pricing must be smaller so that all of the data can properly be processed and inform a meaningful forecast in a reasonable amount of time. If each new test run took hours to run, it would be next to impossible to make meaningful changes over a month of development. This means that the accuracy of pricing models may be smaller than what is ideal. Luckily, in the long-term, this is an easier fix provided a larger dataset can be found. If the smaller dataset is less accurate than random (<0.50), than I will need to search for larger datasets.

Additionally, given the sheer magnitude of the cryptocurrency market and ease of access to create new tokens, I have decided to use datasets that limit to larger, more popular tokens. This will, however, hurt the programs ability to determine if a token’s “moonshot” potential. A moonshot is a small market cap token that shoots up by thousands or tens of thousands of percent.

Long-term Trade-offs:

Once CRYPTOFORECAST enters into a larger, longer development lifecycle, when the dataset size increases, the program will be sacrificing efficiency of a quick decision for performance and accuracy. At times, however, the program will need to sacrifice some performance to deliver a decision in a timely manner.

## 4.5 Implementation Fragments

Show enough *significant parts* of your implementation code—or a simplified form of it—to convince the reader that you will have the implementation of the definite requirements completed on time. Precede each part with a sentence or two explaining its context within the whole application.

## 4.6 References

Add to your references. Instructions as above.

[1] https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/ [2] <https://towardsdatascience.com/lstm-time-series-forecasting-predicting-stock-prices-using-an-lstm-model-6223e9644a2f>

[3] https://www.kaggle.com/sudalairajkumar/cryptocurrencypricehistory

## 4.7 Evaluation of Assignment 4



# Implementation version 1

## 5.1 Summary v1 (next week’s will be your final version)

In a paragraph or two, summarize the current outcome of your project functionally and learning-wise. Avoid duplication with Section 5.3 below, so exclude details here.

The first alteration was that I replaced my dataset with a larger one that simply focuses entirely on the fluctuation of Bitcoin over a nine-year period. Moreover, I have actually implemented the first large components of the CRYPTOFORECAST program. Specifically, that is predictions using standard averages and EMAs, creating a data generator, and much of the initial setup for the LSTMs. I found that the EMA was significantly more successful at predicting the price of Bitcoin than simply using a standard average, though both had fairly low loss averages. The training set was up to and just after the big run up and crash in 2017, when Bitcoin first hit $19,000. Additionally, I did a lot of additional research, as listed below, into similar projects done in the past for inspiration and help in actual implementation.

## 5.2. Report on Requirements v1

Please copy each definite requirement—"DX" (X = F or L)—from Assignment 4 and explain the extent to which you accomplished each definite requirement *so far*. For each, include 1-4 sentences and screenshot of the relevant IO. Your effectiveness depends largely on how much you demonstrate that you learned.

These were not listed in assignment 4, but I will list some of the definite requirements below:

### 5.2.1 Definite Functional Requirements (first priority)

### 5.2.1.1 Definite #1: Implement Preliminary Prediction Programs

CRYPTOFORECAST shall provide basic predictions using averages.

### 5.2.1.2 Definite #2: Setup LSTM

CRYPTOFORECAST shall configure all of the settings, inputs, and parameters for the LSTM.

For the first requirement, I implemented a way to create and visualize basic predictions using an EMA, or exponential moving average. CRYPTOFORECAST uses this EMA to predict the future price of Bitcoin. Here is the screenshot below of both the actual implementation as well as the visualization graph:

Text

Description automatically generatedGraphical user interface, chart

Description automatically generated

While this was very accurate for older data, once we extend it into the future, the EMA model quickly begins to fail, even though this was highly accurate. For reference, the price chart is over 9 years and from 0 to 60k dollars.

For the second requirement, I began utilizing TensorFlow to implement the initial setups for the LSTM RNN. I had to use both the current version as well as TF version 1 due to the deprecation of functions I found that others had used in the past. In the future, I plan to update these to the most up to date methods used. In the code, tf1 denotes TensorFlow version 1. Here are some snippets below, along with the output noting the deprecated code, which will be updated in future implementations. Specifically, the sample is for the parameters needed for the LSTM and the regression layer that the LSTM output will be piped into in the end:

Text

Description automatically generated

As you can see, I use the MultiRNNCell in TensorFlow for the three LSTM cells that I had created earlier in my implementation. I also use dropout in the future to prevent any potential overfitting, as I know that could be a major issue in the future for CRYPTOFORECAST.

## 5.3 Report on Design v1

Describe the design that you have used so far. Indicate how, where, and why it has differed (thus far) from your planned design. Describe its advantages and its shortcomings. Include a description of how the technologies you explored (not the tools—those are described below) have worked out so far. Include at least one diagram.

I have not changed on the design front so much of the details below come from assignment 4.

Once the data is uploaded, the program will work as follows. Pandas will store all of the csv files’ data into pandas data frames. From there, some preliminary graphs will demonstrate the price history of selected cryptocurrencies. After this, pandas will split data into training and testing datasets. Once the datasets are created, CRYPTOFORECAST will normalize the data using a Min Max Scaler and utilize TensorFlow to implement an LSTM RNN. Before anything is done on the LSTM front, I run basic predictions with a standard average and EMA prediction, which both were accurate for historical predictions but failed in the future. The visualization for the superior EMA model is provided above.

The LSTM RNN will use several hidden layers. The RNN will need hyperparameters, like the number of hidden neurons for each cell or information related to the backpropagation through time for optimization. The BPTT hyperparameter will assist in determining how many continuous steps CRYPTOFORECAST will take into account for one optimization step. CRYPTOFORECAST utilized 50 unrolling for BPTT, the batch size is 5000 and a few thousand hidden neurons in each layer as there are well over four million data points to search through. Additionally, CRYPTOFORECAST’s RNN will require standard inputs and outputs as well as parameters for the layers of the LSTM and a regression layer which would calculate the final prediction for the program.

Everything below has yet to be implemented:

Further, the program will need a tool to determine loss and an optimizer for the RNN. I will use the MSE for the calculation of loss, and I researched that “AdamOptimizer” is a recently developed optimizer that generally has been deemed as performing well. It is a stochastic gradient descent optimizer.

Once all of the data is reshaped by numpy, the model will create predictions using the testing dataset. After this, CRYPTOFORECAST will output a visualization to illustrate the predictions generated.

## 5.4 Tools v1

Describe the tool(s) that you are using. Show samples. Describe their advantages and their shortcomings. Limit: 1 page of 12-point Times New Roman.

As mentioned in assignment 4, the most important tool for CRYPTOFORECAST was TensorFlow. Although I used numpy and pandas as well, TensorFlow is the most relevant tool. One of the most obvious disadvantages for me is my lack of familiarity with it as a background. When compared to XGBoost, it is a big slower and less efficient, so it takes some time to run, though I find it’s accuracy is more than acceptable.

The code provided below illustrates when the LSTM completes its output calculations and then pipes those outputs into the regression layer. The c and h variables are the ones that will contain lists for the cell states and the hidden states for respective LSTMCells. This gets transformed into a new shape that will end up being all of the inputs for the training inputs and help calculate the outputs for the RNN.

Text

Description automatically generated

Also, in the end, I utilize TensorFlow gets the dynamic RNN outputs and then I split the results that I got to a list of tensors as well as the loss between the predictions and the actual price of Bitcoin. In my final submission, CRYPTOFORECAST will implement a loss calculator and optimizer, using AdamOptimizer as I mentioned in my design section.

# References

1. <https://www.kaggle.com/mczielinski/bitcoin-historical-data>
2. <https://www.kaggle.com/taniaj/cryptocurrency-price-forecasting>
3. <https://towardsdatascience.com/predicting-prices-of-bitcoin-with-machine-learning-3e83bb4dd35f>
4. <https://towardsdatascience.com/cryptocurrency-prediction-with-lstm-4cc369c43d1b>
5. <https://www.analyticsvidhya.com/blog/2021/05/bitcoin-price-prediction-using-recurrent-neural-networks-and-lstm/>
6. <https://www.kdnuggets.com/2018/11/keras-long-short-term-memory-lstm-model-predict-stock-prices.html>

# Instructor’s Evaluation



6.10.2021

# Implementation, final version

Excluding appendices and figures, this response should not exceed 6 pages.

## 6.1 Summary, final version

In a paragraph or two, summarize the outcome of your project functionally and learning-wise but avoid duplication with Section 6.3 below (which is detailed). Underline edited sentences and additions from v1, if any.

For this finalized version, CRYPTOFORECAST has achieved a foundational functionality and is able to complete its goal. It is able to provide users with a forecast for future Bitcoin prices.

In terms of actual implementation, I developed a loss calculator, an optimizer, the actual script to run the entire project, and finally, CRYPTOFORECAST outputs a chart illustrating the calculated predictions for future prices. Beyond that, I have reached out to like minded friends and laid out a framework and plan for future iterations of the project, as CRYPTOFORECAST is something I am ecstatic about and cannot wait to continuing developing it.

## 6.2. Report on Requirements, final version

Re-list and explain the extent to which you accomplished each definite requirement "DiX" (X = F or L), as well as any other fulfilled requirements. For each, include 1-4 sentences and screenshot(s) (including of command-line text). Underline edited sentences and additions from v1, if any.

These were not listed in assignment 5, but I will list some of the definite requirements below:

### 6.2.1 Definite Functional Requirements (first priority)

### 6.2.1.1 Definite #1: Implement Loss Calculator and Optimizer

CRYPTOFORECAST shall provide basic predictions using averages.

### 6.2.1.2 Definite #2: Run the LSTM

CRYPTOFORECAST shall properly run the LSTM to calculate price predictions for Bitcoin.

### 6.2.1.3 Definite #3: Illustrate Model Predictions

CRYPTOFORECAST shall graph the predictions in order to visualize them for users.

For the first requirement, I utilized TensorFlow to implement the loss and optimization. Here is the implementation:

Graphical user interface, text, application

Description automatically generated

For the second requirement, most of the last parts of my code are the running of the LSTM, and it is mostly implemented using various functions from TensorFlow. In my screenshot, I demonstrate the definitions for the TensorFlow functions:

Text

Description automatically generated

For the third and final requirement, I implemented a visualizer that is much the same as the previous implementations for the best epoch and show how the predictions evolved over time, or how they improved over the course of multiple epochs. Here is a screenshot:

Graphical user interface, text, application

Description automatically generated

## 6.3 Report on Design, final version

Describe the design that you actually used. Indicate how, where, and why it differed from your planned design (we expect that these evolved over time). Describe the advantages and shortcomings of your design. Include at least one diagram to refer to in your narrative. Underline edited sentences and additions from v1, if any.

I have not changed on the design front so much of the details below come from assignment 5.

Once the data is uploaded, the program will work as follows. Pandas will store all of the csv files’ data into pandas data frames. From there, some preliminary graphs will demonstrate the price history of selected cryptocurrencies. After this, pandas will split data into training and testing datasets. Once the datasets are created, CRYPTOFORECAST will normalize the data using a Min Max Scaler and utilize TensorFlow to implement an LSTM RNN. Before anything is done on the LSTM front, I run basic predictions with a standard average and EMA prediction, which both were accurate for historical predictions but failed in the future. The visualization for the superior EMA model is provided above.

The LSTM RNN will use several hidden layers. The RNN will need hyperparameters, like the number of hidden neurons for each cell or information related to the backpropagation through time for optimization. The BPTT hyperparameter will assist in determining how many continuous steps CRYPTOFORECAST will take into account for one optimization step. CRYPTOFORECAST utilized 50 unrolling for BPTT, the batch size is 5000 and a few thousand hidden neurons in each layer as there are well over four million data points to search through. Additionally, CRYPTOFORECAST’s RNN will require standard inputs and outputs as well as parameters for the layers of the LSTM and a regression layer which would calculate the final prediction for the program.

Everything below was implemented for this submission:

Further, the program will need a tool to determine loss and an optimizer for the RNN. I will use the MSE for the calculation of loss, and I researched that “AdamOptimizer” is a recently developed optimizer that generally has been deemed as performing well. It is a stochastic gradient descent optimizer.

Once all of the data is reshaped by numpy, the model will create predictions using the testing dataset. After this, CRYPTOFORECAST will output a visualization to illustrate the predictions generated.

## 6.4 Tools, final version

Describe the tool(s) that you are using. Describe their advantages and their shortcomings. Underline edited sentences and additions from v1, if any.

As mentioned in assignment 4, the most important tool for CRYPTOFORECAST was TensorFlow. Although I used numpy and pandas as well, TensorFlow is the most relevant tool. One of the most obvious disadvantages for me is my lack of familiarity with it as a background. When compared to XGBoost, it is a big slower and less efficient, so it takes some time to run, though I find it’s accuracy is more than acceptable.

The code provided below illustrates when the LSTM completes its output calculations and then pipes those outputs into the regression layer. The c and h variables are the ones that will contain lists for the cell states and the hidden states for respective LSTMCells. This gets transformed into a new shape that will end up being all of the inputs for the training inputs and help calculate the outputs for the RNN.

Text

Description automatically generated

Also, in the end, I utilize TensorFlow gets the dynamic RNN outputs and then I split the results that I got to a list of tensors as well as the loss between the predictions and the actual price of Bitcoin. In my final submission, CRYPTOFORECAST will implement a loss calculator and optimizer, using AdamOptimizer as I mentioned in my design section.

## \*6.5 Contrast between approaches

You were to include two technologies or approaches to your problem, and implement at least one. Contrast the two technologies (not the tools—those are described above) as they specifically relate to your project.

As was noted in every assignment, the biggest choice I had to make for CRYPTOFORECAST in terms of design and approach occurred the first week I worked on the assignment. This was the choice between two underlying foundational choices in how I decided to implement the forecasting technology. The choice was between a RNN, specifically a LSTM, and XGBoost. While I ultimately decided to utilize an LSTM using TensorFlow’s RNN API, I recognize the many advantages to the XGBoost approach. I mostly leaned towards a recurrent neural network due to two factors. Firstly, I noticed the prevalence of our course time that was spent reviewing neural networks and RNNs. As a newcomer to the ML field, I assumed that this would make the actual implementation process a bit more streamlined and easier to comprehend. Secondly, I noticed that in the history of projects in similar veins, (those that predicted stock pricing, mostly), most projects utilized LSTM implementations.

On a more technical front, I became more confident in my ultimate choice. While I would like to eventually add a sentiment analysis function as well as some other primary factors in addition to the historical price of Bitcoin, I knew that the timeline and scope for this term project, CRYPTOFORECAST ought to stick to simply the historical data. XGBoost operates best in the scenarios when the developer provides it with additional inputs outside of the historical observations of the price of the target which it is attempting to forecast. LSTM approaches, on the other hand, still operate well without additional inputs and features.

Moreover, XGBoost, in its more recent iterations and releases, has outperformed LSTM models in terms of actual performance as well as efficiency, winning many ML competitions. Therefore, in my future implementations, CRYPTOFORECAST will eventually transition to an XGBoost approach to optimize itself for sentiment analysis and other inputs. Even at this point, CRYPTOFORECAST takes excess time on a smaller dataset to perform optimally and yield predictions, which is something I would like to remove. Despite this though, as I mentioned, an LSTM model was the optimal approach for a six week version of CRYPTOFORECAST.

## \*6.6 What did *not* work well

We want to see that you understand limitations, not just benefits. Explain the most significant aspects of your project that fell short of your plans or desires.

There were two shortcomings of the project overall that I was anticipating as much. First, my implementation of the LSTM ran much longer than I had initially anticipated. This means that even for the initial setup of CRYPTOFORECAST in the colab notebook, it took quite some time. Because of this, I had to make major changes to the project. First and foremost, CRYPTOFORECAST altered into more of a Bitcoin forecaster. I had to totally change my dataset. I had to remove much of the data I initially planned to use, then switched the dataset for Bitcoin entirely. I had planned to utilize a wide array of cryptocurrencies to model predictions on, such as Ethereum, Litecoin, Dogecoin, Aave, and beyond. While I will be able to do this in the future, I had to eliminate these from the CRYPTOFORECAST dataset for now. On top of that, I expanded from a basic Bitcoin historical dataset to a larger, decade long one as it became the sole focus.

Further, the program was incredibly resource intensive. I needed to use colab pro as well as hardware accelerators to ensure colab did not crash in the final processes on my end. In the future, I would hope to either use a platform without such restrictions or be able to optimize CRYPTOFORECAST further such that it would not be so intensive that it crashes regularly.

## 6.7 What *did* work well

In paragraph form, explain the most significant aspects of your project that met or exceeded your plans or desires.

First and foremost, I felt that the project exceeded my expectations on the learning front; I was skeptical how much I would be able to fully understand as time progressed, but I learned a lot about neural networks and how to implement them with TensorFlow. Additionally, although I see that the performance was not as ideal I saw other models performing, I am glad to see that it does perform better than purely random “coin tosses”, which means the program is worth using. I had researched ML models predictions for both stock price changes as well as similar cryptocurrency prediction models, and while many crypto projects outperformed their stock counterparts (In my opinion, this is mostly due to the fact that the crypto market tends to follow more of the typical mathematical patterns that occur in the stock market), CRYPTOFORECAST ultimately fell somewhat short in terms of actual accuracy when compared to larger scale projects, though I am quite pleased with the final results. I hope to improve the outcomes with further implementations and development on the project. If either my TA Franklyn or Professor Braude have suggestions on this front, I would love to hear them!

## 6.8 Sample Source

Supply key excerpts from your source code—or what comes closest to “source code.” Limit: 2 pages of 12-point Times New Roman equivalent. Include an explanation of where the excerpts fit in your implementation. These are counted as figures, and do not count towards the total page limit.

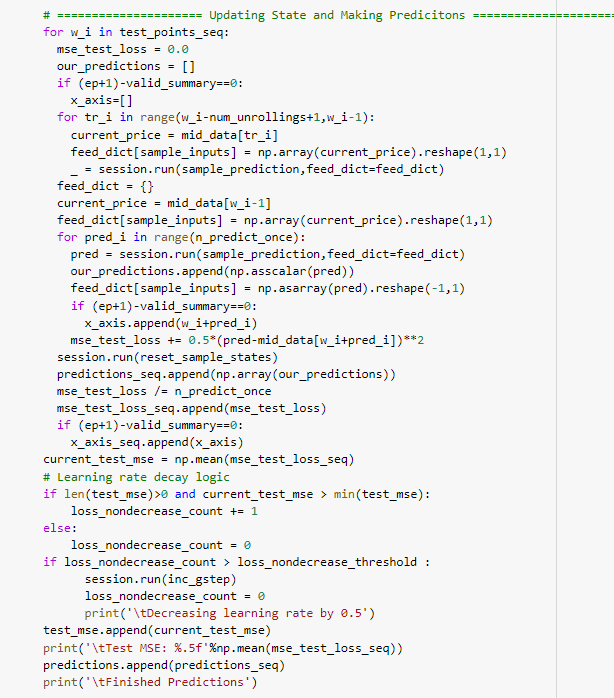
The key sample source provided will be from the actually running of the LSTM for generating the predictions of future Bitcoin prices for CRYPTOFORECAST. Below, I have shown the initial setup and the actual training of the RNN. This is under the training header. As mentioned, my program has had issues being very resource intensive on my machine, so I have set it for 3 epochs for the time being, but that may increase in future releases. The second sample is my actual implementation of CRYPTOFORECAST making predictions for the LSTM.

Basically, what is occurring in this for every epoch is as follows: for a full sequence length of the training data, it unrolls a batch set and trains the RNN with the unrolled batches, calculates loss, and for each of the test set starting points, it updates the state for the LSTM by iterating through previous data points from before the selected starting point, it makes predictions for 1000 steps with the previous prediction as the input, and it calculates the MSE loss.

Here are key source excerpts:

Text

Description automatically generated



## 6.9 Source

Refer the reader to your source code (or what comes closest to it) and input where possible.

My colab source code can be found [here](https://colab.research.google.com/drive/1yNw_wE4E0vv0VxQ2oU-q6oOD3tIQHKdD?usp=sharing). My data set can be found [here](https://www.kaggle.com/mczielinski/bitcoin-historical-data).

## 6.10 Presentation

Make a 3 minute video (5 minute maximum if absolutely needed) presentation of your results, including a demonstration, and point your facilitator to the location of the video.

My presentation can be found here[LINK NEEDED].

# Evaluation



## Appendix 1

For voluminous material, as needed—to be read on an as-needed basis only. (References in at least one place within the paper.)

# References

* <https://www.kaggle.com/mczielinski/bitcoin-historical-data>
* <https://www.kaggle.com/taniaj/cryptocurrency-price-forecasting>
* <https://towardsdatascience.com/predicting-prices-of-bitcoin-with-machine-learning-3e83bb4dd35f>
* <https://towardsdatascience.com/cryptocurrency-prediction-with-lstm-4cc369c43d1b>
* <https://www.analyticsvidhya.com/blog/2021/05/bitcoin-price-prediction-using-recurrent-neural-networks-and-lstm/>
* <https://www.kdnuggets.com/2018/11/keras-long-short-term-memory-lstm-model-predict-stock-prices.html>