**Machine Learning Engineer Nanodegree**

**Capstone Project**

Zilvinas Marozas  
July 14th, 2018

**I. Definition**

**Project Overview**

Predicting financial instruments prices using machine learning has been attempted by many researchers with varying degree of success. Predicting Commodity Futures market is very important not only for speculators, money managers, etc. It is vital for many companies (food producers, oil refineries, farmers) to estimate these prices with certain degree of accuracy since profitability and sometimes very survival of the company and many jobs could depend on correct estimation of commodity prices. Commodities Futures market is very important to US economy and is highly regulated by US government. Various US government agencies like **Commodity Futures Trading Commission** (CFTC) collect lots of data and makes it freely available to the public. The Commitment of Traders (COT) Report is conducted by the Commodity Futures Trading Commission (CFTC) detailing the open interest in each futures and options on commodities markets containing 20 or more traders holding position sizes large enough to meet the CFTC’s reporting level. The purpose of this report is to provide traders with transparency in regards to the open interest in various futures markets and the sizes of those positions for different groups of traders. Therefore, this additional data could be used to supplement pricing information (closing price, and volume) while predicting commodities prices.

This project seeks to utilize Deep Learning models, Long-Short Term Memory (LSTM)

Neural Network algorithm, to predict prices of Corn Commodity Futures. I will use Time Series data (Closing Price and Volume) and data from COT report. Classical neural networks called Multilayer Perceptrons, or MLPs for short, can be applied to

sequence prediction problems i.e. MLPs do offer great capability for sequence prediction but still suffer from key limitation of having to specify the scope of temporal dependence between observations explicitly upfront in the design of the model. Recurrent Neural Networks, or RNNs for short, are a special type of neural network designed for sequence problems. There are a number of RNNs, but it is the LSTM that delivers on the promise of RNNs for sequence prediction. LSTMs have internal state, they are explicitly aware of the temporal structure in the inputs, are able to model multiple parallel input series separately. Given all these advantages of LSTM for sequence predictions , I will use Keras and LSTM for this project.

**Problem Statement**

In this project we will try to predict settling weekly price of Corn Commodity Futures given weekly corn price at the prior time step. In order to perform this prediction, we will create a dataset that includes weekly Corn Futures settling prices as well as Total Open Interest, Long Open Interest and Short Open Interest of Processors/Users (sometimes they are called Commercials) from COT reports. Why do we need to complicate our model and add additional data from COT report instead of just using pricing information? I do believe that using only historical prices alone is not sufficient to predict prices of commodities and especially it is very hard to predict turning points, when commodity reverses the previous trend and starts moving to the opposite direction. Therefore, we need to consider “Fundamentals data” in order to predict these prices with some degree of consistency. However, obtaining “Grains Fundamentals Data” (area planted, area harvested, exports, etc.) is difficult and expensive and it will make machine learning model much more complicated. My hypothesis is that we can substitute “Fundamentals Data” with data from COT report by tracking open interest (long open interest and short open interest) of Processors/Users. This group of traders represents well-funded enterprises that have “deep pockets” in order to conduct research and also have inside information since they are typically large users of grains. For this project I will use a Long Short Term Memory networks (LSTM).

**Metrics**

With forecasts and actual values in their original scale, we can then calculate an error score for the model. In this case, we calculate the Root Mean Squared Error (RMSE) that gives error in the same units as the variable itself.

Benchmark model we will evaluate on the validation and tests datasets. We do this using the walk-forward validation method. In essence, we step through the validation dataset time step by time step and get predictions. Once predictions are made for each time step in the validation dataset, they are compared to the expected values and a Root Mean Squared Error (RMSE) score is calculated. Once we have RMSE for both models we will compare them and decide whether or not LSTM model is good choice for our problem.

**Analysis**

**Data Exploration**

The data used in this project is as follows:

1. Historical Futures Prices: Corn Futures, Continuous Contract #1. Non-adjusted price based on spot-month continuous contract calculations. Raw data from CME: https://www.quandl.com/data/CHRIS/CME\_C1-Corn-Futures-Continuous-Contract-1-C1-Front-Month
2. Commitment of Traders - CORN (CBT) - Futures Only (002602) <https://www.quandl.com/data/CFTC/002602_F_ALL-Commitment-of-Traders-CORN-CBT-Futures-Only-002602>

**The first data set (Corn Futures Prices) has the following format**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Date** | **Open** | **High** | **Low** | **Last** | **Change** | **Settle** | **Volume** | **Previous Day Open Interest** |
| 7/10/2018 | 344.25 | 344.75 | 336.25 | 339.5 | 6 | 339.75 | 2668 | 2186 |
| 7/9/2018 | 346 | 348.5 | 342.5 | 346 | 6 | 345.75 | 3190 | 2969 |
| 7/6/2018 | 342 | 352.25 | 342 | 350.75 | 8.25 | 351.75 | 3068 | 3959 |
| 7/5/2018 | 345.5 | 348.75 | 341.5 | 342.5 | 0.75 | 343.5 | 3302 | 4812 |
| 7/3/2018 | 340.25 | 345.25 | 339.25 | 343.25 | 5.25 | 342.75 | 3048 | 5687 |

Table 1

Columns like Open, High, Low, Last, Change and Settle are highly correlated and most important column is Settle (Futures Settling Price) which we will try to predict. Volume is also quite important. Previous Day Open Interest column will not be used, since we are going to use another data set (The Commitments of Traders Report) which provides much more detailed Open Interest information.

Summary statistics of the Historical Futures Prices data set is presented in the table below:

|  |  |  |
| --- | --- | --- |
|  | **Settle** | **Volume** |
| **count** | 3034 | 3034 |
| **mean** | 456.9793 | 103905.2 |
| **std** | 140.2046 | 73993.22 |
| **min** | 219 | 0 |
| **25%** | 359.75 | 40172.75 |
| **50%** | 389 | 102567 |
| **75%** | 564.625 | 152391.3 |
| **max** | 831.25 | 538170 |

Table 2

There is small reason to be concerned with validity of data, i.e. minimum volume is zero.

I performed some additional analysis on these records with zero volume, and my findings are presented in the table below:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Open** | **High** | **Low** | **Last** | **Change** | **Settle** | **Volume** | **Previous Day Open Interest** |
| **Date** |  |  |  |  |  |  |  |  |
| **4/5/2007** | 359.75 | 367.5 | 357.25 | 366 | NaN | 366 | 0 | 354349 |
| **4/6/2012** | 658.25 | 658.25 | 658.25 | 658.25 | NaN | 658.25 | 0 | 401521 |
| **4/3/2015** | 386.5 | 386.5 | 386.5 | 386.5 | NaN | 386.5 | 0 | 470964 |

Table 3

Seems all instance happened during different years, and since I am planning to resample this daily data to weekly data (reasons are explained in data preparation section ), I will remove these records from the final data set, and this removal should ne play a big impact on final resampled dataset .

**The second data set (****Commitments of Traders Report) has following columns:**

*Date, Open Interest, Producer Merchant Processor User Longs, Producer Merchant Processor User Shorts, Swap Dealer Longs, Swap Dealer Shorts, Swap Dealer Spreads, Money Manager Longs, Money Manager Shorts, Money Manager Spreads, Other Reportable Longs, Other Reportable Shorts, Other Reportable Spreads, Total Reportable Longs, Total Reportable Shorts, Non-Reportable Longs, Non-Reportable Shorts.*

We will use this data set as supplementary data set in order to supplement our primary data set i.e. we will use the following columns: *Date, Open Interest, Producer Merchant Processor User Longs, Producer Merchant Processor User Shorts.*

This is a sample of data presented below:

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Open Interest | Producer Merchant Processor User Longs | Producer Merchant Processor User Shorts |
| 7/10/2018 | 1818055 | 500172 | 750062 |
| 7/3/2018 | 1830330 | 484257 | 773851 |
| 6/26/2018 | 1885804 | 513100 | 840177 |
| 6/19/2018 | 1992169 | 525197 | 920764 |
| 6/12/2018 | 1963233 | 488666 | 917204 |

Table 4

Summary statistics of the Commitments of Traders Report data set is presented in the table below:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Open Interest** | **Producer Merchant Processor User Longs** | **Producer Merchant Processor User Shorts** |
| **count** | 6.31E+02 | 631 | 6.31E+02 |
| **mean** | 1.29E+06 | 270795.049 | 6.27E+05 |
| **std** | 2.10E+05 | 68976.2216 | 1.55E+05 |
| **min** | 7.48E+05 | 102373 | 2.97E+05 |
| **25%** | 1.19E+06 | 226595 | 5.24E+05 |
| **50%** | 1.30E+06 | 262823 | 6.11E+05 |
| **75%** | 1.40E+06 | 314224 | 7.06E+05 |
| **max** | 1.99E+06 | 525197 | 1.00E+06 |

Table 5

**Exploratory Visualization**

Let’s look at visual representation of data:

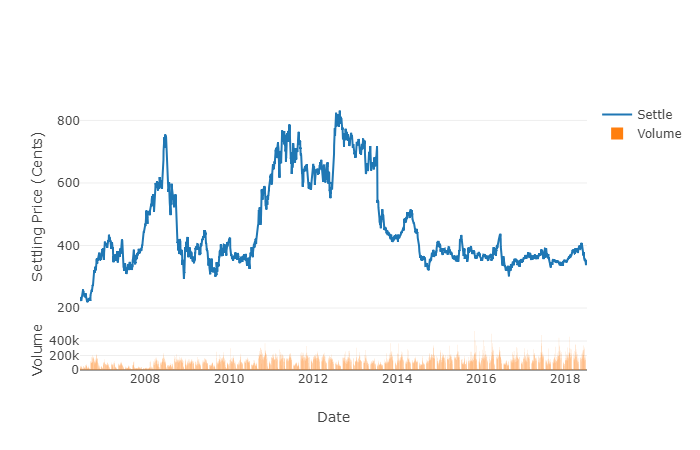


Figure 1

We can see that Corn Futures during 12-year history traded in a range with peak prices in 2008, 2012 and 2013.

Let’s combine all information from both datasets and put them on one chart. Also, Futures Pricing information has been resampled to weekly data, since we can only obtain COT weekly data. We can see that for the most part Open Interest (total, long and short) also tend to fluctuate over time with slight upward bias.

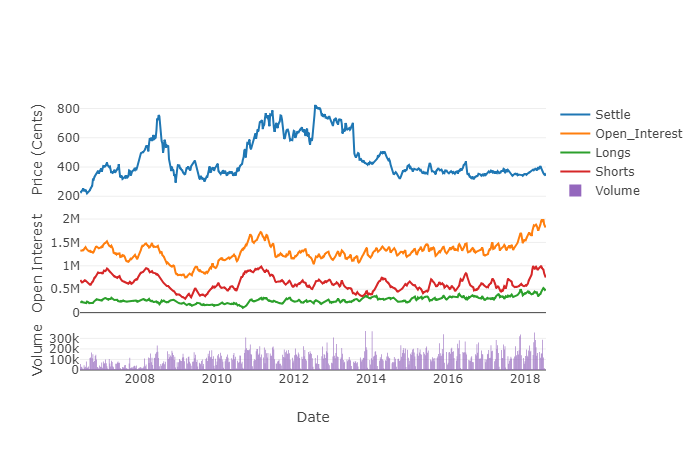


Figure 2

Let’s look at another visualization. It is similar to chart presented above, however on the X axis we have trading weeks starting a zero instead of dates:



Figure 3

It can be helpful to compare line plots for the same interval, such as weekly Settle price,

and year-to-year in order to determine certain seasonality patterns:

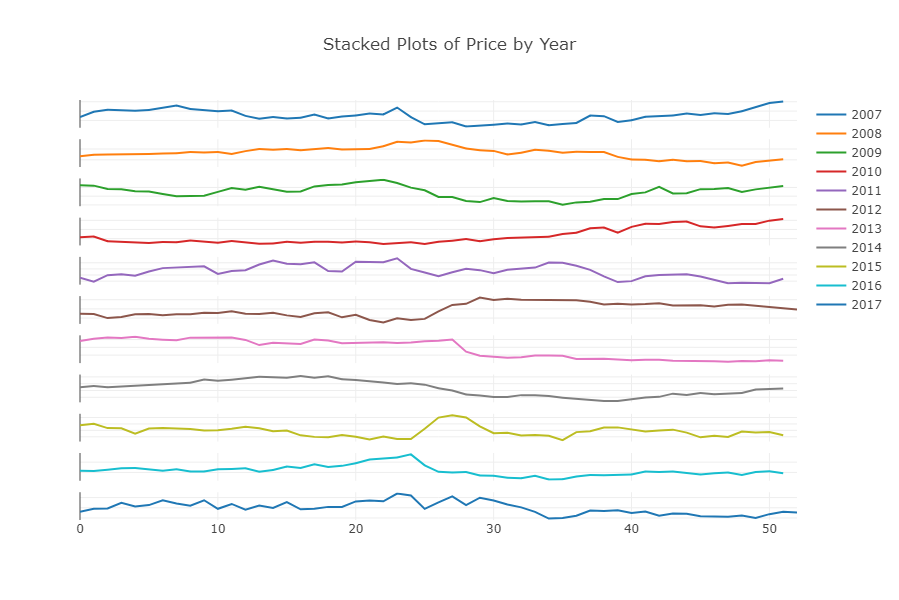


Figure 4

It is really hard to see any significant seasonality patterns, from the chart presented above.

Time series modeling assumes a relationship between an observation and the previous observation. Previous observations in a time series are called lags, with the observation at the previous time step called lag1, the observation at two-time steps ago lag=2, and so on. A useful type of plot to explore the relationship between each observation and a lag of that observation is called the scatter plot. Let’s plot Lag Scatter Plots of Corn Futures Settle Price. We can see from the chart presented below the strongest relationship between an observation with its lag=1 value, but generally a good positive correlation with each value in the last month (4 weeks).

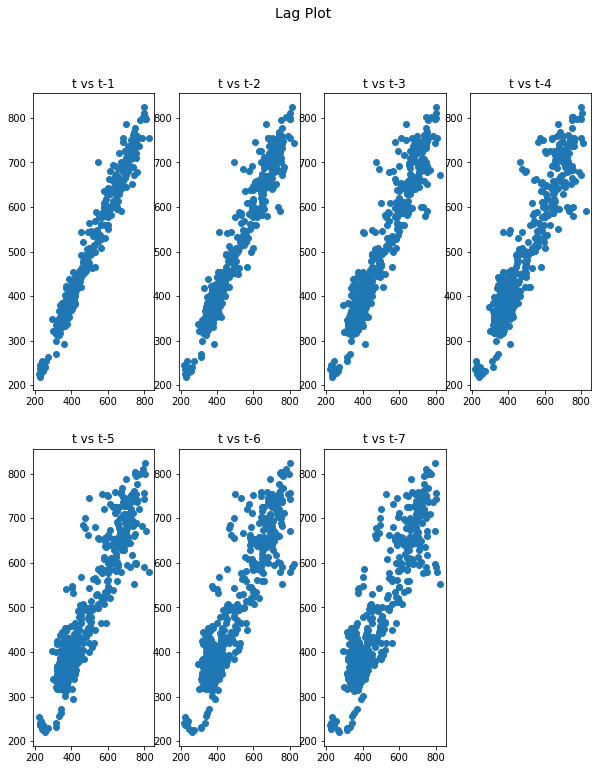


Figure 5 Multiple Lag scatter plots of the Corn Futures Settle Price.

**Algorithms and Techniques**

The main goal of this project is to apply LSTM (Long Short-Term Memory) to predict the Corn Futures Settling price. The Long Short-Term Memory, or LSTM, network is a type of Recurrent Neural Network. Recurrent Neural Networks, or RNNs for short, are a special type of neural network designed for sequence problems. LSTMs are very different to other deep learning techniques, such as Multilayer Perceptrons (MLPs) and Convolutional Neural Networks (CNNs), in that they are designed specifically for sequence prediction problems. Sequence prediction is different to other types of supervised learning problems. The sequence imposes an order on the observations that must be preserved when training models and making predictions.

In this project we will use a relatively simple Neural Network Architecture based on LSTM. Our network will contain:

1. Input layer.
2. Fully connected LSTM hidden layer.
3. Fully connected output layer.

This Neural Network is presented in the figure bellow.

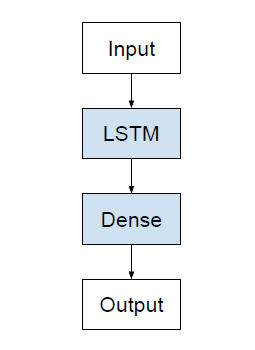


Figure 6 LSTM Network Architecture

In the figure presented below we define simplified implementation of this network in Keras, with ellipsis for the specific configuration of the number of neurons in each layer.

|  |
| --- |
| *model = Sequential()*  *model.add(LSTM(..., input\_shape=(...)))*  *model.add(Dense(...))* |

Figure 7 Defining Simple LSTM model in Keras.

**Benchmark**

A benchmark in forecast performance provides a point of comparison. Three properties of a good technique for making a benchmark forecast are:

* Simple: A method that requires little or no training or intelligence.
* Fast: A method that is fast to implement and computationally trivial to make a prediction.
* Repeatable: A method that is deterministic, meaning that it produces an expected output given the same input.

A common algorithm used in establishing a baseline performance for time series forecasting is the persistence algorithm. The persistence algorithm uses the value at the previous time step (t-1) to predict the expected outcome at the next time step (t). This satisfies the three above conditions for a baseline forecast. Due to simplicity of benchmark model it will consider only the price and will disregard other feature like Open Interest from COT report.

|  |
| --- |
| *def model\_persistence(x):*  *return x* |

Figure 8 Persistence Model implemented in Python

**III. Methodology**

**Data Preprocessing**

**Data Cleaning Resampling and Merging**

The data used in the project has been described in section Data Exploration. The data is not ready to use as is. Here are the steps we will have to do:

* Delete suspicions data records from Corn Futures Prices dataset. This has been discussed in Data Exploration section.
* Drop unnecessary columns from Corn Futures Prices dataset and COT report dataset
* Resample Corn Futures Prices dataset from daily time frame to weekly time frame, since COT report data is provided only for the weekly time frame and we need to use both data sets at the same time frame.
* Merge both data sets into combined one dataset.

Here is how final dataset looks like:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Settle** | **Volume** | **Open Interest** | **Longs** | **Shorts** |
| **Date** |  |  |  |  |  |
| **6/18/2006** | 235.5 | 56486 | 1320155 | 209662 | 699163 |
| **6/25/2006** | 228.25 | 28361 | 1321520 | 224476 | 666688 |
| **7/2/2006** | 235.5 | 30519 | 1329400 | 234769 | 645735 |
| **7/9/2006** | 241 | 13057 | 1327482 | 220552 | 648405 |
| **7/16/2006** | 253.5 | 2460 | 1333225 | 216968 | 673110 |

Table 6

The columns Settle and Volume came from Corn Futures Prices dataset and columns Open Interest, Longs, Shorts came COT report dataset. All data preparation code is located in grains\_futures\_prices\_prognosticator Jupyter notebook.

**LSTM Data Preparation**

LSTMs are sensitive to the scale of the input data, specifically when the sigmoid (default) or tanh activation functions are used. It can be a good practice to rescale the data to the range of 0-to-1, also called normalizing. We can easily normalize the dataset using the **MinMaxScaler** preprocessing class from the scikit-learn library.

Next, we need to frame the dataset as a supervised learning problem. We will frame the supervised learning problem as predicting the Corn Futures Weekly Settlement price at the current week (t) given the settlement price, volume and open interest at the prior time step.

The python code for this transformation is located in file *data\_preparer.py* ( function: *series\_to\_supervised* )

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **var1(t-1)** | **var2(t-1)** | **var3(t-1)** | **var4(t-1)** | **var5(t-1)** | **var1(t)** |
| **1** | 0.026044 | 0.15256 | 0.45976 | 0.253744 | 0.570655 | 0.014055 |
| **2** | 0.014055 | 0.076421 | 0.460857 | 0.28878 | 0.52454 | 0.026044 |
| **3** | 0.026044 | 0.082263 | 0.467192 | 0.313123 | 0.494786 | 0.035138 |
| **4** | 0.035138 | 0.03499 | 0.46565 | 0.279499 | 0.498578 | 0.055808 |
| **5** | 0.055808 | 0.006302 | 0.470267 | 0.271023 | 0.533659 | 0.028938 |

Table 7

**Splitting the data**

We will split data into three datasets

Training data: 06/16/2006- 12/31/2016

Validation data: 01/01/2017-12/31/2017

Testing data: 01/01/2018 -07/10/2018

Finally, the inputs (X) are reshaped into the 3D format expected by LSTMs, namely [samples, timesteps, features].

The python code for splitting the data is in file data\_preparer.py ( function: split\_data)

**Implementation**

We are now ready to design and fit our LSTM network for our problem. We will define the LSTM with 1 neurons in the first hidden layer and 1 neuron in the output layer for predicting pollution. The input shape will be 1 time step with 5 features. We will use the Mean Absolute Error (MAE) loss function and the efficient Adam version of stochastic gradient descent. The model will be fit for 500 training epochs with a batch size of 64. We will keep track of both the training and test loss during training by setting the validation\_data argument in the fit() function. At the end of the run both the training and test loss are plotted.

In this section, the process for which metrics, algorithms, and techniques that you implemented for the given data will need to be clearly documented. It should be abundantly clear how the implementation was carried out, and discussion should be made regarding any complications that occurred during this process. Questions to ask yourself when writing this section:

* *Is it made clear how the algorithms and techniques were implemented with the given datasets or input data?*
* *Were there any complications with the original metrics or techniques that required changing prior to acquiring a solution?*
* *Was there any part of the coding process (e.g., writing complicated functions) that should be documented?*

**Refinement**

In this section, you will need to discuss the process of improvement you made upon the algorithms and techniques you used in your implementation. For example, adjusting parameters for certain models to acquire improved solutions would fall under the refinement category. Your initial and final solutions should be reported, as well as any significant intermediate results as necessary. Questions to ask yourself when writing this section:

* *Has an initial solution been found and clearly reported?*

**IV. Results**

*(approx. 2-3 pages)*

**Model Evaluation and Validation**

In this section, the final model and any supporting qualities should be evaluated in detail. It should be clear how the final model was derived and why this model was chosen. In addition, some type of analysis should be used to validate the robustness of this model and its solution, such as manipulating the input data or environment to see how the model’s solution is affected (this is called sensitivity analysis). Questions to ask yourself when writing this section:

* *Is the final model reasonable and aligning with solution expectations? Are the final parameters of the model appropriate?*
* *Has the final model been tested with various inputs to evaluate whether the model generalizes well to unseen data?*
* *Is the model robust enough for the problem? Do small perturbations (changes) in training data or the input space greatly affect the results?*
* *Can results found from the model be trusted?*

**Justification**

In this section, your model’s final solution and its results should be compared to the benchmark you established earlier in the project using some type of statistical analysis. You should also justify whether these results and the solution are significant enough to have solved the problem posed in the project. Questions to ask yourself when writing this section:

* *Are the final results found stronger than the benchmark result reported earlier?*
* *Have you thoroughly analyzed and discussed the final solution?*
* *Is the final solution significant enough to have solved the problem?*

**V. Conclusion**

*(approx. 1-2 pages)*

**Free-Form Visualization**

In this section, you will need to provide some form of visualization that emphasizes an important quality about the project. It is much more free-form, but should reasonably support a significant result or characteristic about the problem that you want to discuss. Questions to ask yourself when writing this section:

* *Have you visualized a relevant or important quality about the problem, dataset, input data, or results?*
* *Is the visualization thoroughly analyzed and discussed?*
* *If a plot is provided, are the axes, title, and datum clearly defined?*

**Reflection**

In this section, you will summarize the entire end-to-end problem solution and discuss one or two particular aspects of the project you found interesting or difficult. You are expected to reflect on the project as a whole to show that you have a firm understanding of the entire process employed in your work. Questions to ask yourself when writing this section:

* *Have you thoroughly summarized the entire process you used for this project?*
* *Were there any interesting aspects of the project?*
* *Were there any difficult aspects of the project?*
* *Does the final model and solution fit your expectations for the problem, and should it be used in a general setting to solve these types of problems?*

**Improvement**

In this section, you will need to provide discussion as to how one aspect of the implementation you designed could be improved. As an example, consider ways your implementation can be made more general, and what would need to be modified. You do not need to make this improvement, but the potential solutions resulting from these changes are considered and compared/contrasted to your current solution. Questions to ask yourself when writing this section:

* *Are there further improvements that could be made on the algorithms or techniques you used in this project?*
* *Were there algorithms or techniques you researched that you did not know how to implement, but would consider using if you knew how?*
* *If you used your final solution as the new benchmark, do you think an even better solution exists?*