Avocado Price Prediction

Akshay Gudi
gudi@uni-potsdam.de

Md Rakibul Islam
md.rakibul.islam@uni-potsdam.de

Pratik Agrawal
agrawal1@uni-potsdam.de

Timofei Kornev
kornev@uni-potsdam.de

Jurate Vaistaraite
vaistaraite@uni-potsdam.de

Abstract

In this paper, we present the results of time series forecasting for avocado prices. The avocado dataset is available on the Hass Avocado Board website. For the benchmark against the deep learning models, machine learning models like Autoregression and ARIMA are used. We get the baseline models and accuracy for the dataset using these machine learning models. We apply deep learning models, namely a recurrent neural network (RNN) and a long short-term memory (LSTM) neural network. The aim is to find out if the deep learning methods outperform the traditional ML algorithms mentioned above. Also, our goal was to identify regions on which the deep learning-based models perform better. We also established which deep learning architecture performed better on this dataset.

1 Introduction

Time series forecasting is an established field of predicting future values based on past data. The techniques of the time series forecasting have been employed for a long time in domains like finance, supply chain, etc. Over a period of time, algorithms have evolved and improved. Recent advances in cheap computing hardware and its ease of availability has given rise to deep learning models, which are being applied to various applications like computer vision, natural language processing, time series forecasting, etc.

In this paper, we present our results of application of deep learning as well as traditional machine learning models.

This paper is structured as follows. Section 2 provides an overview of the literature related to time series forecasting based on deep learning models. Section 3 provides some insights into the dataset using exploratory data analysis. In Section 4, we talk about the architecture and training of machine learning- and deep learning-based models. In Section 5, we present the evaluation metrics and discuss the results. Section 6 discusses model performance, challenges. Finally, we conclude before describing the scope of future work.

2 Related Work

Looking at the available literature for deep learning-based time series forecasting (which is primarily focused on the financial data) [10], we can list the deep learning architectures used most often: Deep Multi-Layer Perceptron (DMLP), Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Convolutional Neural Networks (CNN), Gated Recurrent Unit (GRU), Deep Belief

Networks (DBN), Deep Reinforcement Learning (DRL), Autoencoders (AEs), Restricted Boltzmann Machines (RBM), etc. Some papers combined various architectures to improve the accuracy of the predictions. RNN-based models (in particular LSTM) are the most commonly used models. Meanwhile, CNN and DMLP have been used extensively in classification-type implementations (like trend classification) as long as appropriate data processing is applied to the raw data. [10]

In the majority of the studies, DL models were better than ML. However, there were also many cases where their performances were comparable. There were even two particular studies [12,4] where ML models performed better than DL models. Advances in computing power, availability of big data, superior performance, implicit feature learning capabilities, and user-friendly model development environment for DL models are among the main reasons for this migration.

The studies can also be clustered into the two main groups based on their expected outputs: price prediction and price movement (trend) prediction. Even though price forecasting is a regression problem, in most of the financial time series forecasting applications, the correct prediction of the price is not perceived as important as correctly identifying the directional movement. As a result, researchers consider trend prediction, i.e. forecasting which way the price will change, a more crucial study area compared with exact price prediction. In that sense, trend prediction becomes a classification problem. In some studies, only up or down movements are taken into consideration (2-class problem), whereas up, down or neutral movements (3-class problem) also exist.[10]

Regardless of the underlying forecasting problem, the raw time series data was almost always embedded directly or indirectly within the feature vector, which is particularly valid for the RNN-based models.

3 Dataset Description

The data was downloaded from the Hass Avocado Board website in July of 2020, pre-processed and compiled into a single CSV. It represents weekly actual retail sales of the Hass avocados from 2015 to 2020. The Average Price (of avocados) in the table reflects a per unit (per avocado) cost, even when multiple units (avocados) are sold in bags.

Some relevant columns in the dataset:

- Date - The date of the observation
- AveragePrice - The average price of a single avocado
- Type - The type of avocados, can be conventional or organic
- Year - The year
- Region - The region, sub-region of the observation
- Total Volume - Total number of avocados sold
- 4046 - Total number of avocados with PLU 4046 sold
- 4225 - Total number of avocados with PLU 4225 sold
- 4770 - Total number of avocados with PLU 4770 sold
- Total Bags - Total number of bags sold
- Small Bags - Total number of small bags sold
- Large Bags - Total number of large bags sold
- XLarge Bags - Total number of extra large bags sold

The shape of the dataset is (29483, 13)
The columns used for the model training: Date, type, region
Output: AveragePrice

3.1 Preprocessing Steps

We created a dataframe from the Avocado dataset, converted the Date column to the appropriate Date type object, set it as the index of the dataframe, and sorted the dataframe by the new index

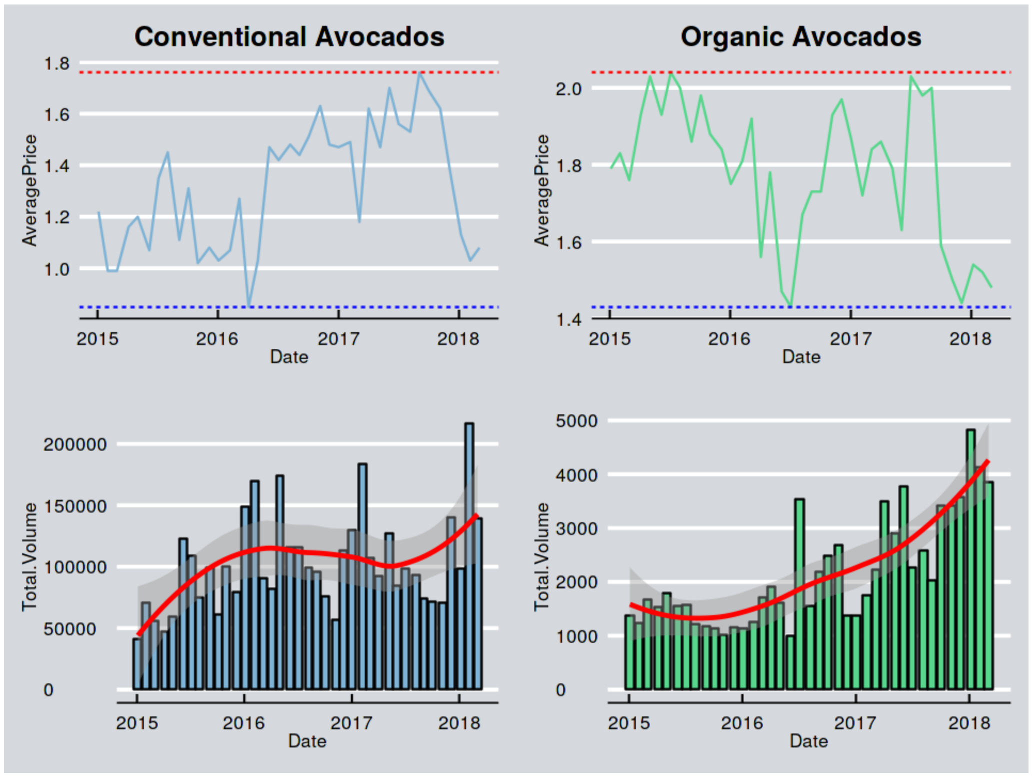

Figure 1: Average price and Total volumn of avocados in USA[9]

(DatetimeIndex). We dropped any unnecessary columns [which?]. After inspecting the categorical variables, we found two avocado types contained in the ”type” column of the dataset: conventional and organic. Then, we applied one-hot encoding to this ”type” columns so that later we could easily feed the dataset into our machine learning models. As for the regions of the USA represented in the dataset, we have 5 regions (”Midsouth”, ”Northeast”, ”SouthCentral”, ”Southeast”, ”West”), 48 sub-regions (i.e., cities and towns), and finally a special label ”TotalUS” which represents the weekly price averaged across all regions. We also created 3 subsets from the original dataset: one containing the data for just the ”TotalUS” region, another one for the regional data, and the last one for the sub-regions. After checking the correlation between columns, we found out that there is almost no correlation between AveragePrice and all other variables.

We draw an autocorrelation plot between data and AveragePrice and noticed that there’s some significant correlation between t=1 and t=15 (roughly) with a significant decline in correlation after that time frame. Then we performed a stationary test on the time series. A series is said to be stationary if the mean, variance, and covariance are constant over a period of time or time-invariant. To check this we performed Rolling statistics check, autocorrelation check and Dicky-Fuller test. From these 3 tests, we found that the time series is not stationary and requires differencing. Differencing is a method by which a time series can be made stationary. After differencing once, the series is called as integrated of order 1 and denoted by l(1). Finally, We split the dataframes into training and test data. Our dataframe contains time series for 273 weeks and we selected 95% of the data as training data and the remaining 5% of the data for the test data. That means, 259 weeks for training data and the rest 14 weeks for test data.

After plotting the time series based on “TotalUS”, regions and sub-regions, we had observations. These time series plots look pretty similar for US, regions and sub-regions. Most of the prices in the year of 2015 were in the range of $1.0 for conventional avocados. While from 2016 and onwards the density of the prices was a little bit higher. Most price peaks occur for both conventional and organic avocados between the months of September and October. At the end of the year, there is a major price drop for both types of avocados. During the year of 2017 and 2019, the avocado market

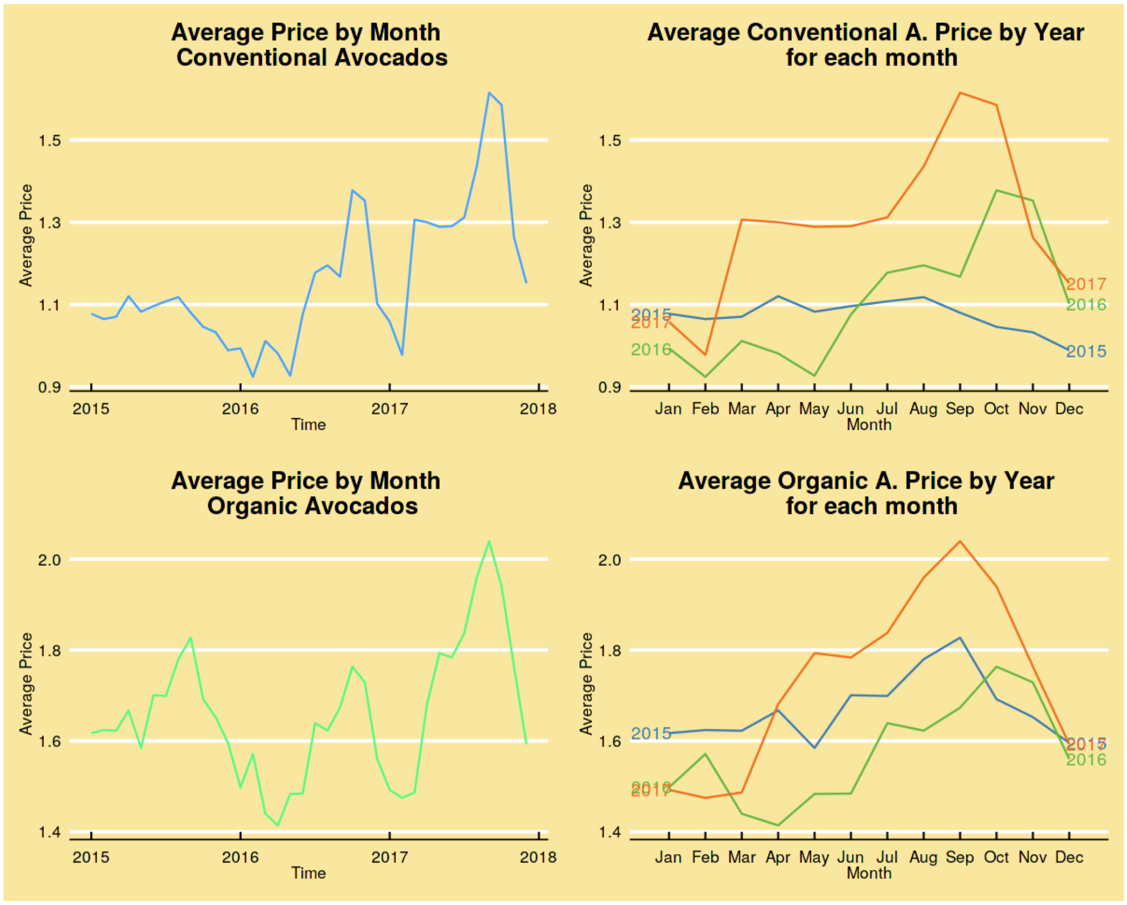

Figure 2: Average Monthly price of avocados in USA[9]

experienced the highest volatility for both conventional and organic avocados. For both types of avocados, we can see that there is always a price drop in the first four/five months of each year, but then it picked up the price in the next few months before dropping again at the end of each year.

4 Architecture and Training

For time series analysis with a machine learning approach, few of the best-suited models are AutoRegression (AR), ARIMA, Facebook Prophet, SARIMA, SARIMAX, etc.
In this paper, we have implemented an AR and an ARIMA model.

4.1 Auto Regression (AR)

An AR (Autoregression) model is a linear predictive modeling technique that attempts to predict the dependent variable based on the past values of the variable. It implies that previously observed steps are useful to predict future values.

Equation of the AR model of order p is described by the formula (1), where $X_{t-1}, ..., X_{t-p}$ are the past series values (lags) and ϵ_t is white noise.

$$X_t = \gamma + \phi_1 X_{t-1} + \phi_2 X_{t-2} + ... + \phi_p X_{t-p} + \epsilon_t \tag{1}$$

The disadvantage of the autoregressive model is that it expects the fundamental forces that drive the variable not to change over time and if the change happens, the model cannot predict the outcomes accurately anymore. [13]

Training: For training, the AutoReg model was used. It trains automatically on the given data and uses the Ordinary least squares method to calculate weights for the specified number of lags (i.e. values of prices in the past X weeks and use for predicting a future value). After trying multiple different values for the lag size, setting the lags to 53 weeks showed the most favorable result for the total USA data. Hyperparameters (in this case one hyperparameter which is the number of lags) were selected manually using an autocorrelation plot.

The same experiment has been run for various subregions (i.e. cities like Atlanta, Albany, etc.) and also for the total data of the USA. The time series for a certain city were extracted and the AR model was trained on it. The accuracy of the predictions was evaluated with the Mean Squared Errors method and plotted along with the ground-truth data. Before applying the AR model, the data was made stationary using the diff method from the Pandas library. It performs automatic differencing by subtracting the value at the previous timestep $t-1$ from the value at the current timestamp t for all time steps.

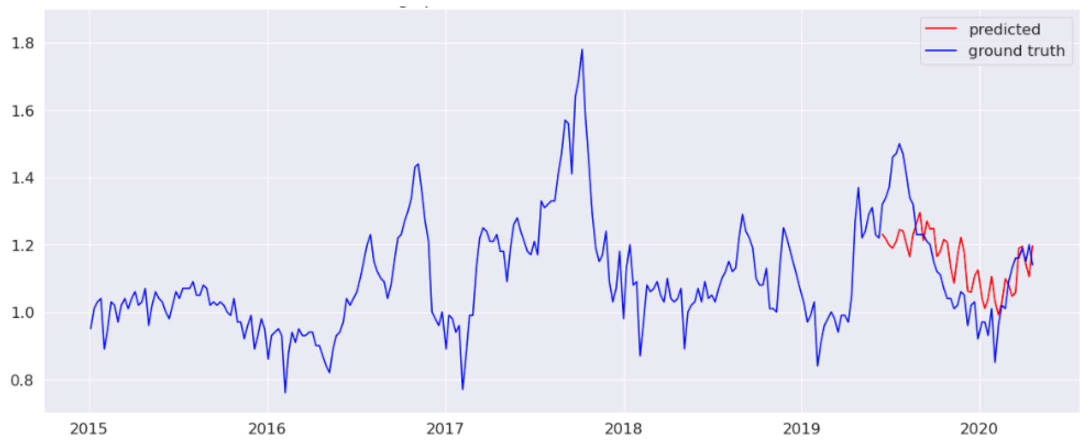

Figure 3: Predicted price vs true price (AR)

4.2 ARIMA

ARIMA is a statistical data model that is used to analyze and predict time series data. ARIMA stands for AutoRegressive Integrated Moving Average. It extrapolates patterns from the past into the future and assumes that the future will remain similar to what it has seen in the past. This model consists of three components: autoregression (AR), integration (I) and moving average (MA). The ARIMA model takes three parameters – p, d, and q. p is the number of autoregressive terms (also called the lag order). d is the number of nonseasonal differences (also known as the degree of differencing), and it tells how many times to perform the differencing. For example, if we take lag-1 to remove the trend from the data, then $d=0$ means that there is no differencing applied, $d=1$ means that differencing is performed once, $d=2$ means double differencing, and so on. q is the parameter describing the size of the moving average window (another term for this parameter is the order of moving average). [1]

An autoregressive model (Y_t depends only on its lags – it includes predictions that are lag versions of the series) can be described by the following equation:

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + ... + \beta_p Y_{t-p} + \epsilon_1 \tag{2}$$

A moving average model (Y_t depends only on the lagged forecast errors):

$$Y_t = \alpha + \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + ... + \phi_p \epsilon_{t-p} \tag{3}$$

The ARIMA model (Y_t is equal to a constant plus linear combination of lags of Y plus a linear combination of lagged forecast errors. The idea of the ARIMA model is to capture all forms of autocorrelation by including lags of the series and the forecast errors):

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + ... + \beta_p Y_{t-p} \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + ... + \phi_q \epsilon_{t-q} \tag{4}$$

Training: The most important part of the ARIMA model is to select the p, d, q values. Trying different combinations for these 3 values and figuring out for which combination the model performs better is a very lengthy and time-consuming process. That's why grid search on p, d, q was used. We used Akaike Information Criteria (AIC) as a measurement factor for the grid search. AIC is a widely used measure of a statistical model. It basically quantifies the goodness of fit, and the simplicity of the model into a single statistic without over-fitting the data. When comparing the two models, the one with the lower AIC is generally "better". We applied these powerful criteria in comparing various ARIMA models with different combinations of p, d, q and took the values for which the AIC was lower.

We filtered the time series dataframe based on the US and all 5 regions for both conventional and organic Avocados and run the ARIMA model. From the grid search, we got different parameter values p, d, q for each model. We split these dataframes into train and test data. We selected 95% (273 weeks) data as training data and the remaining 5% (14 weeks) as test data. The mean squared error method was used to measure the accuracy of the models and finally plotted along with the test data.

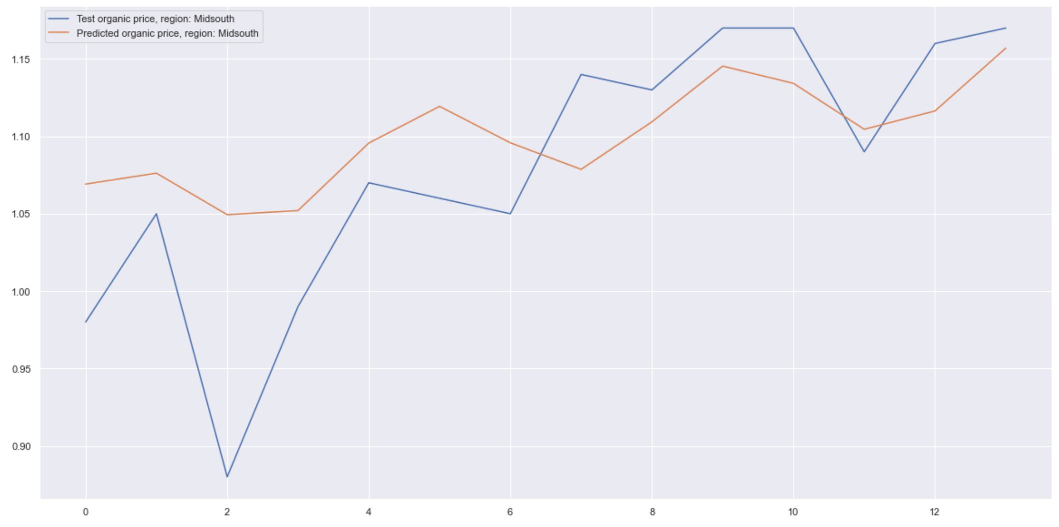

Figure 4: Predicted price vs original price (ARIMA)

4.3 RNN

A recurrent neural network (RNN) is a generalization of a feedforward neural network that has internal memory. An RNN is recurrent in nature as it performs the same function for every input of data while the output of the current input depends on the past one computation. After producing the output, it is copied and sent back into the recurrent network. For making a decision, it considers the current input and the output that it has learned from the previous input. RNNs can use their internal state (memory) to process sequences of inputs. In other neural networks, all the inputs are independent of each other. But in an RNN, all the inputs are related to each other.[10]

$$h_t = Wf(h_{t-1}) + W^{(hx)}x_{[t]} \tag{5}$$

$$y_t = W^{(S)} + f(h_t) \tag{6}$$

$$\frac{\partial E}{\partial W} = \sum_{t=1}^{T} \frac{\partial E_t}{\partial W} \tag{7}$$

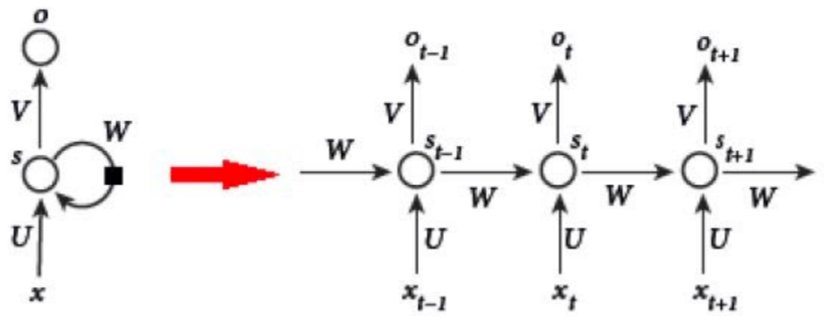

Figure 5: RNN architecture

4.3.1 Hyperparameter Optimization

One way to improve a model's performance is to optimize its hyperparameters by trying various combinations of hyperparameters' values and checking which one leads to the best model performance. For the RNN model, we have the following hyperpameters: the number of RNN layers, the number of hidden units in the RNN layers (we keep this number the same for each layer) and the activation function of the output of [the second layer / both layers?]. Table 1 lists all combinations of hyperparameters used in the experimentation and the corresponding values of the two evaluation metrics: MSE and RMSE.

Table 1: Combinations of the hyperparameters and their evaluation for the RNN model

Hidden Units	**Layers**	**Activation**	**MSE**	**RMSE**
40	1	ReLu	0.0031	0.055
30	1	ReLu	0.0035	0.059
30	1	tanh	0.0036	0.060
40	1	tanh	0.0028	0.053
30	2	tanh	0.0024	0.049
40	2	tanh	0.0036	0.060

As could be seen from the Table 1, the best performing model is the one with 2 stacked RNN layers, 30 hidden units in each layer and the tanh activation function. As an output layer, we use a single Dense (fully connected) layer with 1 unit.

4.3.2 Learning Rate Optimization

After finding the best hyperparameters for the model, we need to find the best learning rate for training it. In order to accomplish this, the RNN is initially trained with a learning rate scheduler starting with the learning rate of 1e-5 (0.00001) and ending with a value of ..., The Figure 6 shows the plot for Loss vs Learning rate, from which it can be inferred that learning rate of 1e-2 is optimal for our model.

4.3.3 Training

Now we train the model with best hyperparameters and learning rate determined by experimentation in previous steps. To find the best fit model during training checkpoints are used to keep track of model with best parameters. At the end of training the model with best parameters is extracted from checkpoints to make predictions.

The Adam algorithm is a better choice for optimizer since it contains in-built exponential decay rate and we need specify the decay rate.

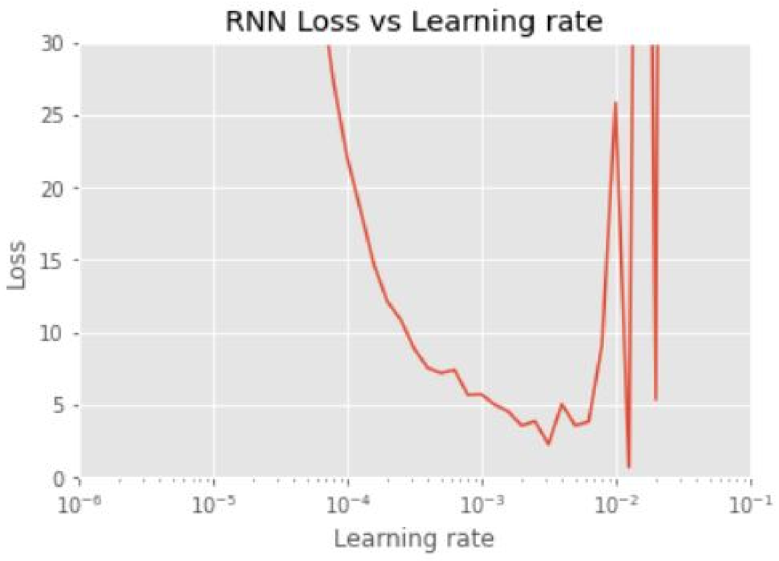

Figure 6: Loss vs learning rate (RNN)

The Figure 7 shows variation in Training and validation loss through the epochs.

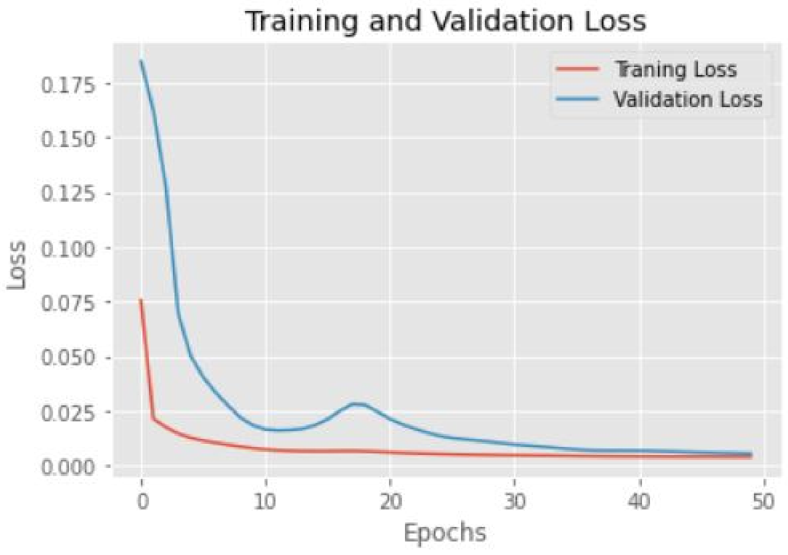

Figure 7: Epoch vs learning rate (RNN)

4.3.4 Prediction

Table 2 provides metrics for performance of the model for all regions.

It is evident that the model has best performance for Mid-South region, least performance for North-east region and it has performed moderately for other regions. Figure 8 and Figure 9 shows predictions for few regions.

Table 2: Evaluation of the model - RNN

Region	MSE	RMSE
West	0.0055	0.0745
Northeast	0.0070	0.0840
Southeast	0.0065	0.0810
SouthCentral	0.0053	0.0729
MidSouth	0.0024	0.049

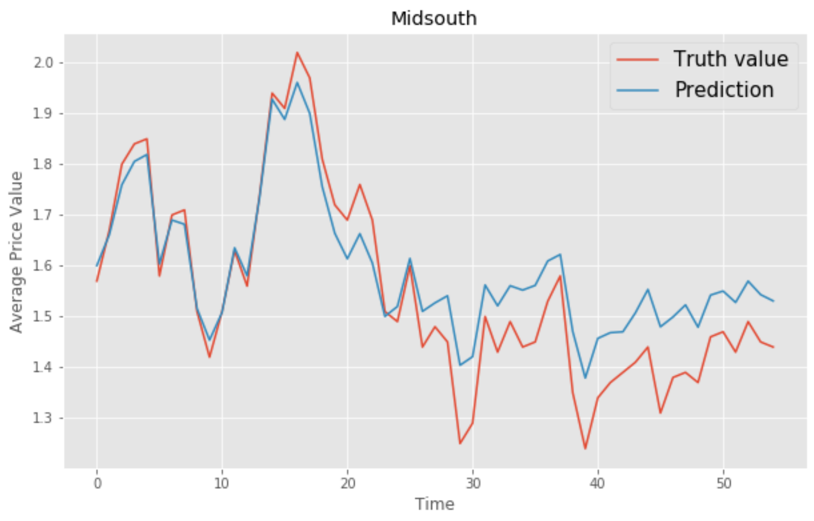

Figure 8: RNN - MidSouth Prediction

4.4 LSTM

LSTM is a type of RNN where the network can remember both short term and long term values.LSTM networks consist of LSTM units Each LSTM unit merges to form an LSTM layer. An LSTM unit is composed of cells having an input gate, output gate and forget gate. Three gates regulate the information flow. With these features, each cell remembers the desired values over arbitrary time intervals. [10]

The equations below show the form of the forward pass of the LSTM unit. (xt: input vector to the LSTM unit, ft: forget gate's activation vector, it: input gate's activation vector, ot: output gate's activation vector, ht: output vector of the LSTM unit, ct cell state vector, σ_g - sigmoid function,σ_c ,σ_h: hyperbolic tangent function, *: element-wise (Hadamard) product, W , U: weight matrices that need to be learned, b: bias vector parameters that need to be learned).

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \tag{8}$$

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \tag{9}$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \tag{10}$$

$$c_t = f_t * c_{t-1} + i_t * \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \tag{11}$$

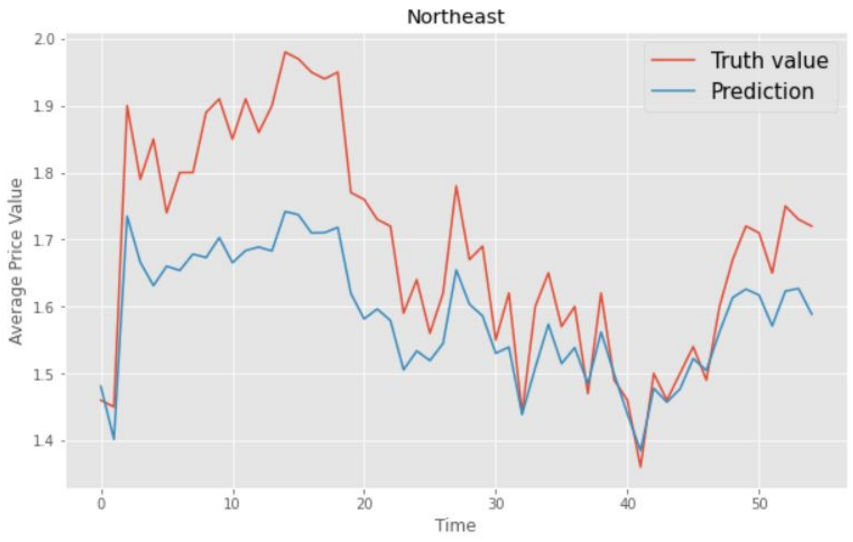

Figure 9: RNN - NorthEast Prediction

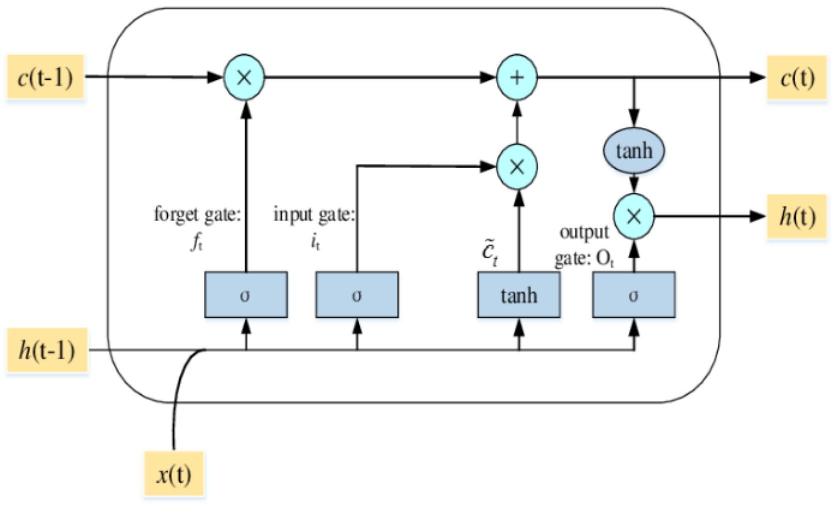

Figure 10: LSTM cell[5]

$$h_t = o_t * \sigma_h(c_t) \tag{12}$$

4.4.1 Naive Hyperparameter Optimization

Our final LSTM model consists of 1 LSTM layer with single Dense layer. We perform the same experimentation as done for RNN to find the best hyperparameters like number of hidden units, number of layers, optimizer, activation function and learning rate.

The Table 3 provides all combinations of hyperparameters used during experimentation and their corresponding metrics.

From the performance metrics provided in the table 3, it is evident that the relatively best parameters to construct our model are 30 hidden units, with 1 LSTM layer, tanh activation function, and adam optimizer.

Table 3: Hyperparameters of the model - LSTM

Hidden Units	Layers	Activation	MSE	RMSE
40	1	ReLu	0.0036	0.060
30	1	ReLu	0.0035	0.059
30	1	tanh	0.0030	0.054
30	2	tanh	0.0021	0.046
40	2	tanh	0.0075	0.087
50	2	tanh	0.0073	0.085

4.4.2 Learning Rate Optimization

We perform training of the LSTM model with best hyperparameters and a learning rate scheduler to find the best learning rate for training our model.

The diagram below shows the plot for Loss vs Learning rate, from which it can be inferred that learning rate of 1e-2 is optimal for our model.

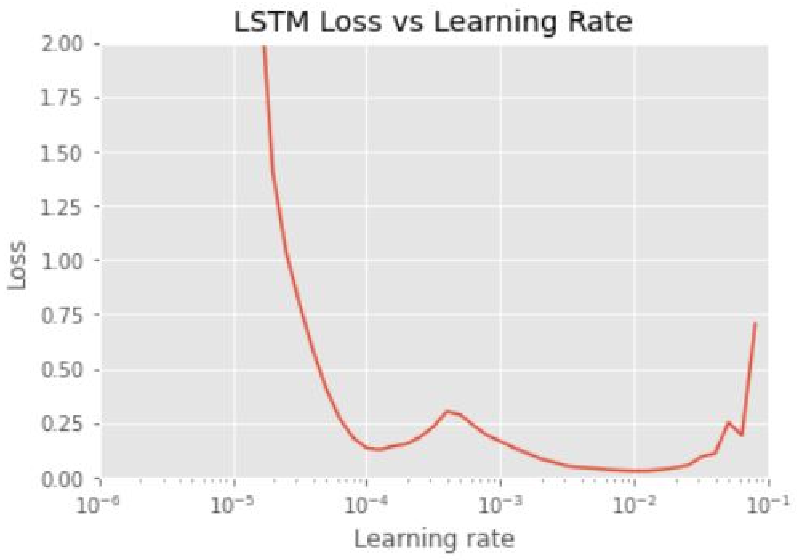

Figure 11: Loss vs learning rate (LSTM)

4.4.3 Training

Now we train the model with best hyperparameters and learning rate determined by experimentation in previous steps. To find the best fit model during training checkpoints are used to keep track of the model with best parameters. At the end of training the model with best parameters is extracted from checkpoints to make predictions. The Figure 12 shows variation in Training and validation loss through the epochs.

4.4.4 Prediction

Table 4 provides metrics for performance of the model for all regions.

It is evident that the model has best performance for Mid-south region and South-central region, least performance for North-east and South-east region and it has performed moderately for West region.

The Figure 14 and 15 shows predictions for few regions,

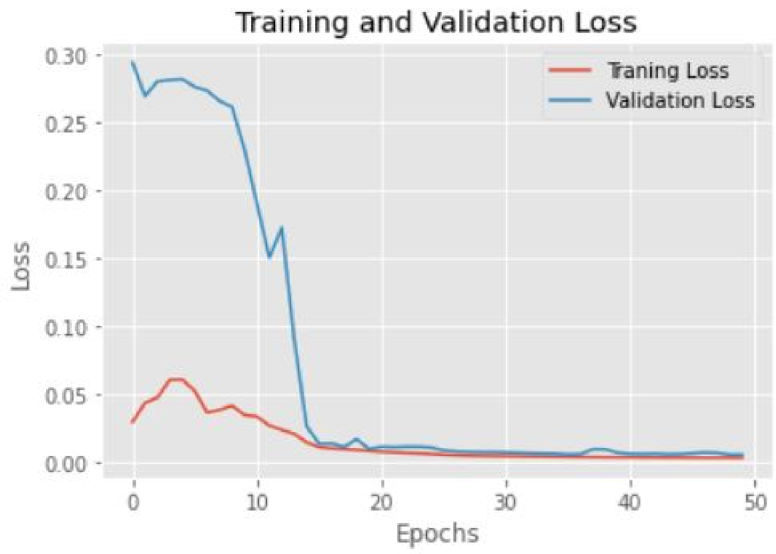

Figure 12: Epoch vs learning rate (LSTM)

Table 4: Evaluation of the model - LSTM

Region	MSE	RMSE
West	0.0063	0.0799
Northeast	0.0091	0.0955
Southeast	0.0089	0.0946
SouthCentral	0.0035	0.0597
Midsouth	0.0024	0.0497

5 Evaluation

To evaluate the performance of the model 3 metrics are used in our approach.

Mean absolute error (MAE): Mean absolute error is the average of the absolute values of the deviation. This type of error measurement is useful when measuring prediction errors in the same unit as the original series.

Mean squared error (MSE): The mean squared error is the average of the square of the forecast error. As the square of the errors are taken, the effect is that larger errors have more weight on the score.

Root mean square error (RMSE): It is a metric calculated by taking the root of MSE. It advantage of being in the same unit as the forecast variable, and since it takes squares of errors it will have huge effect on outliers

6 Discussion on Model Performance

Looking at the metrics from Table 2 and 4, we can see the RNN and LSTM perform relatively well, However LSTM performs better for multiple regions (MidSouth, West, East and South-central) and RNN performs well only for 2 regions (MidSouth and SouthCentral) Both models have bad performance for predicting NorthEast region. On the other hand, AR performs well for 2 regions (Northeast and SouthCentral) and ARIMA performs well only for Midsouth region.

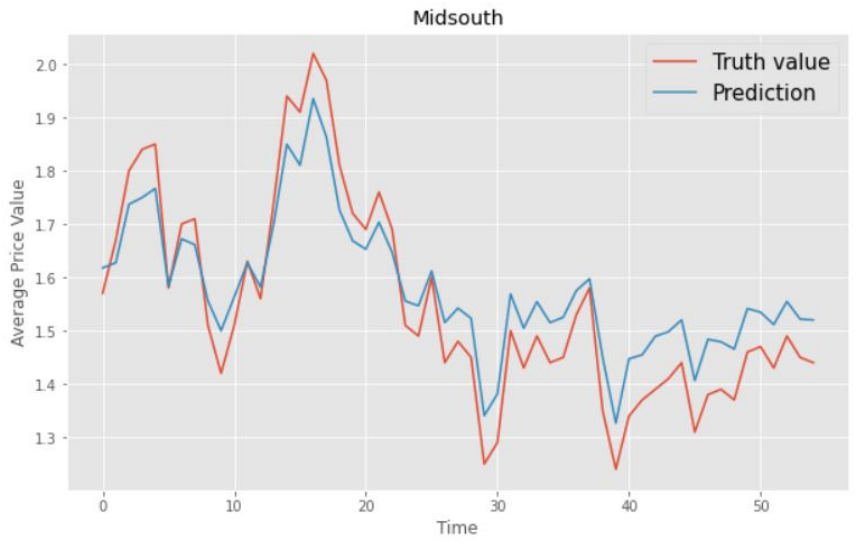

Figure 13: LSTM - MidSouth Prediction

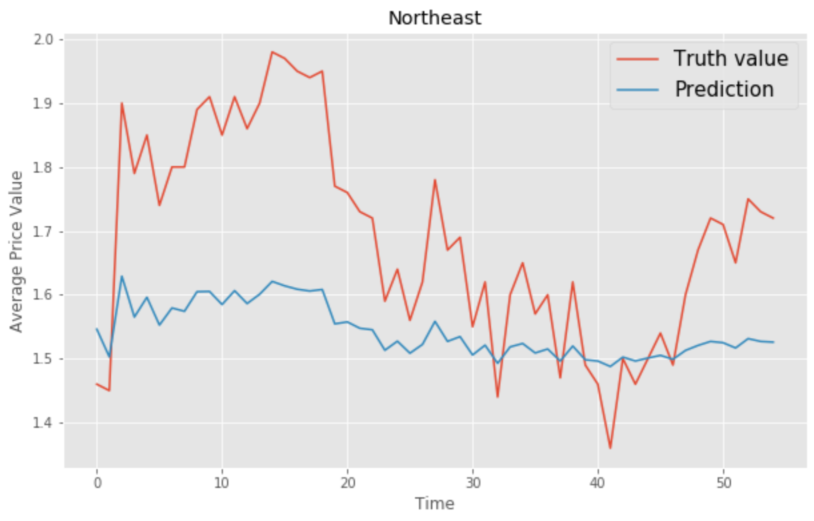

Figure 14: LSTM - NorthEast Prediction

Table 5: Evaluation of the models

Algorithm/Architecture	**MAE**	**MSE**	**RMSE**
AR	0.050	0.005	0.224
ARIMA	0.072	0.008	0.266
RNN	0.022	0.0024	0.049
LSTM	0.021	0.0024	0.049

Deep learning approach using RNN and LSTM have better performance when compared to shallow learning approach. Shallow learning approaches have better performance for the Northeast region however deep learning approach performs better for most of the regions.

6.1 Challenges and Experimentation with dataset

Initially the model used only Average Price over weeks to Train and Predict, hence each region had to be trained separately. However, in order to have a single model to predict values for all regions, We inputted Average Price as well as regions (in form of one-hot encoded values) to the model. This improves the performance of the model considerably since it learns Avocado price along with region information, and a single model can be used to predict prices of all the regions.

6.2 Improving the model

For Deep Learning approach more advanced architectures like CNN with LSTM and transformers can be used. For choosing better hyper-parameters, advanced hyper-parameter optimization techniques like Bayesian or Gradient Based optimization can be used. In order to get the best architecture for the model, advance techniques like Neural Architectural Search can be used.

it is however to be noted that some studies[8] have found that it is not necessarily the case that the Deep Learning models outperform traditional time series approaches always.

One study used a subset of 1045 monthly series from the 3003 of the M3 Competition [9] to calculate the performance of eight traditional statistical methods and eight popular ML ones, plus two more that have become popular during recent years.[8] Comparing the performance of all algorithmic models available for time series, it was found that the machine learning methods were all out-performed by simple classical methods, where ETS and ARIMA models performed the best overall.

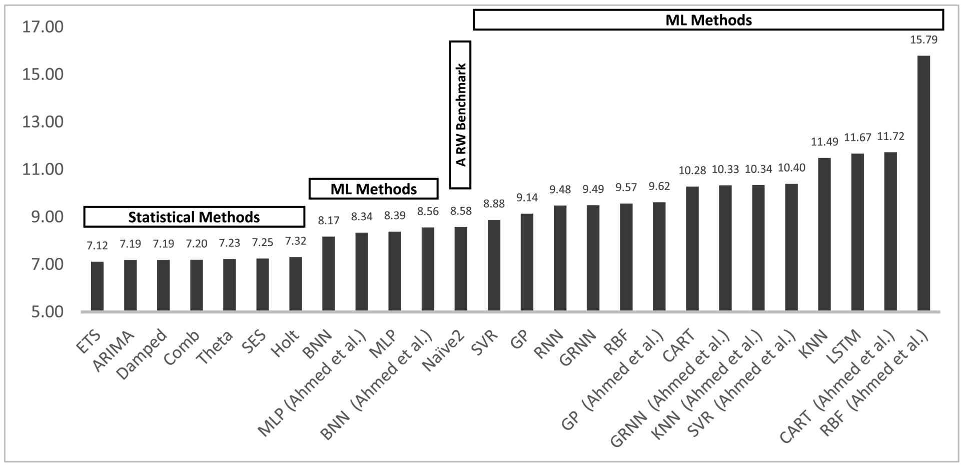

Figure 15: Models Comparison: M3 Competition [8]

7 Future Work

We intend to apply Convolutions Neural Network-based architectures on the dataset. We have observed the increasing interest in 2-D CNN implementations of financial forecasting problems through converting the time series into an image-like data type. This innovative methodology seems to work quite satisfactorily and provides promising opportunities. Some studies used the ensemble of methods (DL Traditional) [6][2] which proved to be providing better accuracy than stand-alone LSTMs or ML methods. we intend to explore these models in the future.

8 Conclusion

Price forecasting is one of the key applications of time-series prediction, In our paper, we experimented with shallow and deep learning approaches for avocado price forecasting. Although the

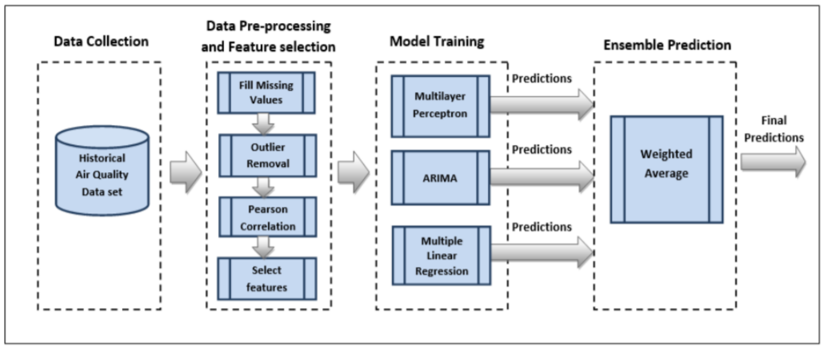

Figure 16: Ensemble Prediction Model[2]

shallow learning models perform well to a certain extent, the application of Deep learning can help us understand multiple patterns in data, Hence the usage of more complex Deep learning approaches in the future will help in predicting prices more accurately. However, It is advisable that the decision of what time series forecasting algorithm to choose should be taken on case by case bases depending on a dataset and bench-marking of different statistical, machine learning and deep learning methods.

9 References

[1] Adhikari, A. R., Agrawal, R. K. (2013). An Introductory Study on Time Series Modeling and Forecasting. Van Haren Publishing.

[2] Amrutha C, Dr. B G Prasad. Air Pollutant Concentration Prediction using Ensemble of Machine Learning Techniques *IJRECE VOL.6 ISSUE 3*(JULY -SEPTEMBER 2018)

[3] Chen, K., Zhou, Y., Dai, F. (2015). A LSTM-based method for stock returns prediction: A case study of China stock market. *2015 IEEE International Conference on Big Data (Big Data)* . doi:10.1109/bigdata.2015.7364089

[4] Dezsi, E., Nistor, I. A. (2016). CAN DEEP MACHINE LEARNING OUTSMART THE MARKET? A COMPARISON BETWEEN ECONOMETRIC MODELLING AND LONG- SHORT TERM MEMORY. *Romanian Economic and Business Review*, 11, 54-73.

[5] Fig. 2. The structure of the LSTM unit. (2019, June 20). *ResearchGate*. Retrieved July 29, 2020, from https://www.researchgate.net/figure/The-structure-of-the-LSTM-unit$_fig2_3$31421650

[6] Krstanovic S., Paulheim H. (2017) Ensembles of Recurrent Neural Networks for Robust Time Series Forecasting. In: *Bramer M., Petridis M. (eds) Artificial Intelligence XXXIV. SGAI 2017*. Lecture Notes in Computer Science, vol 10630. Springer, Cham

[7] M3-Competition. (2020, July 29). *International Institute of Forecasters*. Retrieved July 29, 2020, from https://forecasters.org/resources/time-series-data/m3-competition/

[8] Makridakis, S., Spiliotis, E., Assimakopoulos, V. (2018). Statistical and Machine Learning forecasting methods: Concerns and ways forward. PLOS ONE, 13(3), e0194889. doi:10.1371/journal.pone.0194889

[9] Price of Avocados —— Pattern Recognition Analysis. (2019, April 29). *Kaggle*. Retrieved July 29, 2020, from https://www.kaggle.com/janiobachmann/price-of-avocados-pattern-recognition-analysis

[10] Sezer, O. B., Gudelek, M. U., Ozbayoglu, A. M. (2020). Financial time series forecasting with deep learning : A systematic literature review: 2005–2019. *Applied Soft Computing, 90*, 106181. doi:10.1016/j.asoc.2020.106181

[11] Selvin, S., Vinayakumar, R., Gopalakrishnan, E. A., Menon, V. K., Soman, K. P. (2017). Stock price prediction using LSTM, RNN and CNN-sliding window model. *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. doi:10.1109/icacci.2017.8126078

[12] Sermpinis, G., Stasinakis, C., Dunis, C. (2014). Stochastic and genetic neural network combinations in trading and hybrid time-varying leverage effects. *Journal of International Financial Markets, Institutions and Money, 30, 21–54* . doi:10.1016/j.intfin.2014.01.006

[13] What Does Autoregressive Mean? (2019, October 3). *Investopedia*. Retrieved July 29, 2020, from https://www.investopedia.com/terms/a/autoregressive.asp