CS677 Final Project Report

-Alfred Zane Rajan Velladurai

Reneel Sagar Pamarthi

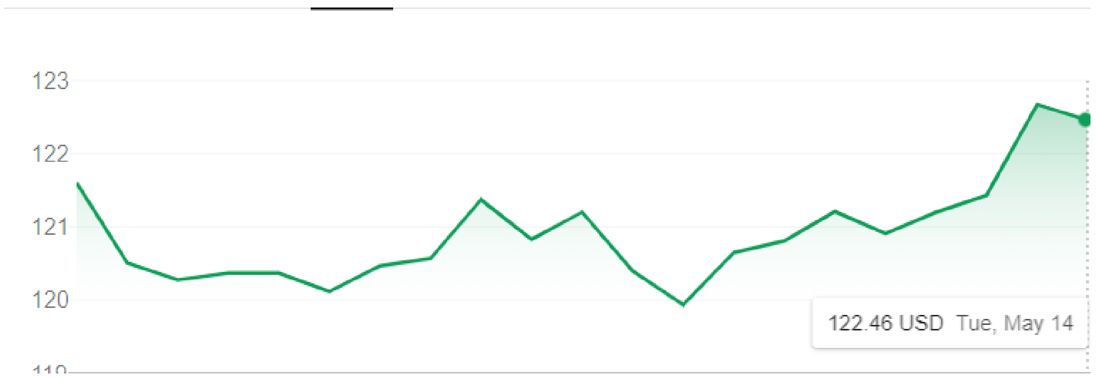
GLD: SPDR Gold Trust

Commodity market

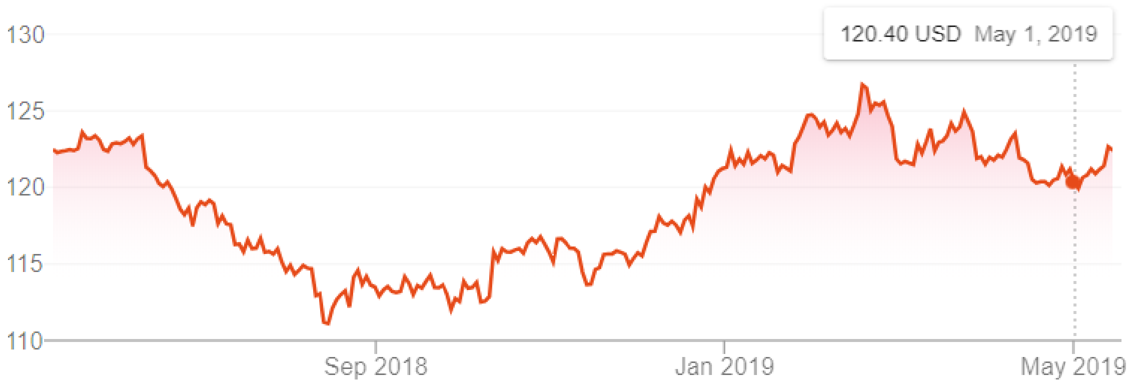
GLD is the largest ETF to invest directly in physical gold. NAV is determined using the LBMA PM Gold Price (formerly the London PM Gold Fix), so GLD has an extremely close relationship with spot prices. Its structure as a grantor trust protects investors; trustees cannot lend the gold bars. However, taxes on long-term gains can be steep, as GLD is deemed a collectible by the IRS. It is extremely liquid, trading at miniscule spreads. Although more expensive to hold than competitor IAU, GLD is typically cheaper to trade. Also, GLD's NAV has a larger handle, which corresponds to more gold exposure per share. Investors paying per-share commissions should find this fact appealing.

Market trend:

Past month:



Past Year:



Deep Learning Network Architecture:

- Input (10x4)
- Dense Layer (100)
- LSTM (100)
- Dense Layer (100)
- Dense Layer (4)
- ReLU Activation
- MSE
- Adam Optimizer
- 20 Epochs

Dense Layer:
A dense layer is just a regular layer of neurons in a neural network. Each neuron recieves input from all the neurons in the previous layer, thus densely connected. The layer has a weight matrix W, a bias vector b, and the activations of previous layer a. The following is te docstring of class Dense from the keras documentation:

output = activation(dot(input, kernel) + bias)where activation is the element-wise activation function passed as the activation argument, kernel is a weights matrix created by the layer, and bias is a bias vector created by the layer.

Our dense layer contains 100 units. ReLU is used as the activation function.

ReLU activation function:
In the context of artificial neural networks, the rectifier is an activation function defined as the positive part of its argument:

{\displaystyle f(x)=x^{+}=\max(0,x)} {\displaystyle f(x)=x^{+}=\max(0,x)},

where x is the input to a neuron. This is also known as a ramp function and is analogous to half-wave rectification in electrical engineering. This activation function was first introduced to a dynamical network by Hahnloser et al. in 2000 with strong biological motivations and mathematical justifications. It has been demonstrated for the first time in 2011 to enable better training of deeper networks, compared to the widely-used activation functions prior to 2011, e.g., the logistic sigmoid (which is inspired by probability theory; see logistic regression) and its more practical counterpart, the hyperbolic tangent. The rectifier is, as of 2017, the most popular activation function for deep neural networks.

A unit employing the rectifier is also called a rectified linear unit (ReLU).

LSTM:

LSTM is special type of neuron to deal with problem of RNN which is basically the extend the memory of RNN. Meaning Sequent Marker tag to size of the context.

Specifically designed to handle long term dependency. Meaning remember the input long sequence of time to preserve the context of input.

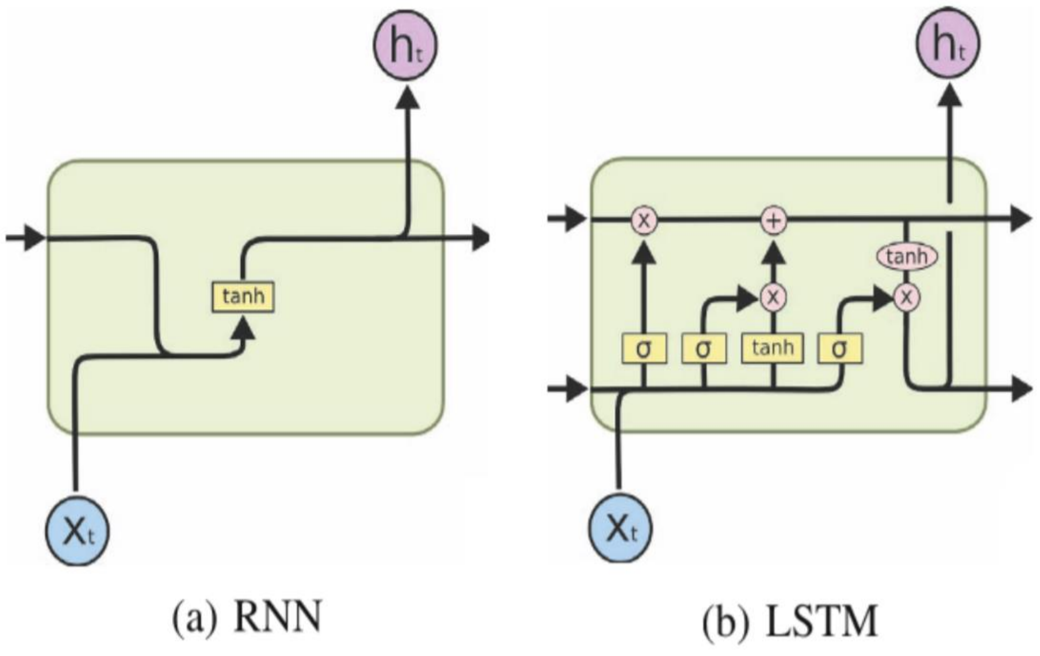
Example with window size is 5 : "Color of apple is green"

Here we don't need any information from past to understand this sentence.

Now, Take look at this sentence.

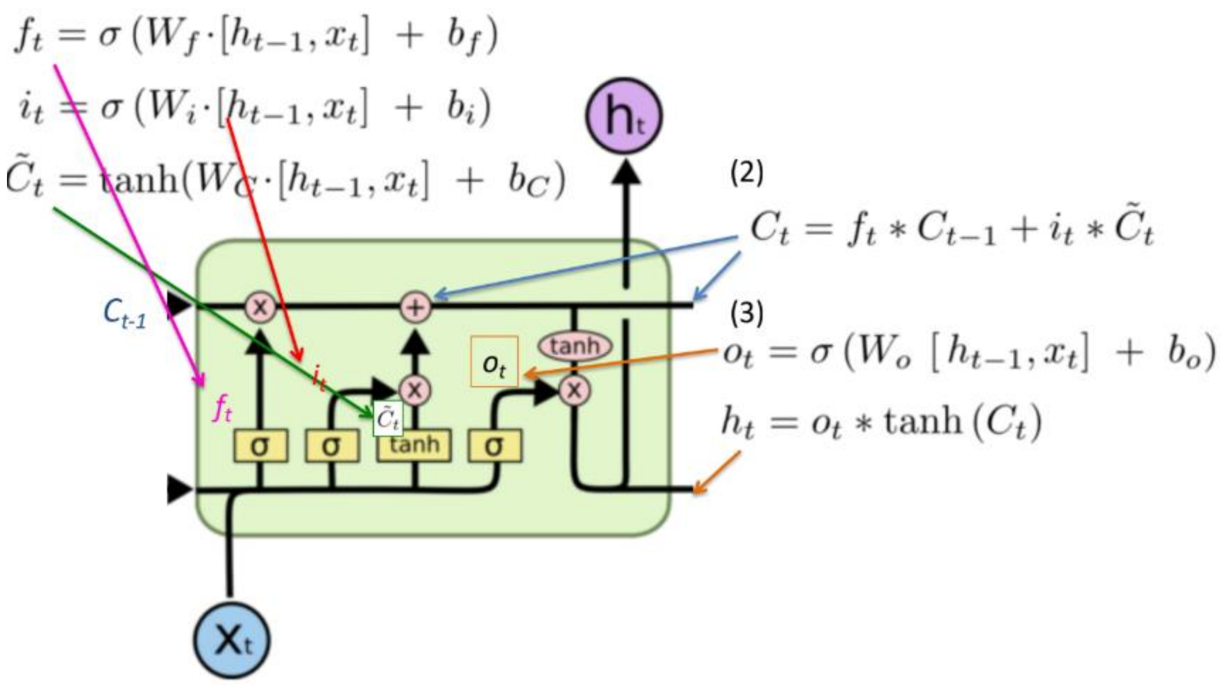
"Rajan likes working in Data Science . He wants to perceive carrier towards in his interested area."

Difference between RNN and LSTM:



What information thrown away from Cell state using Sigmoid layer called forget gate. This layer will check Cell state value from from previous timestamp using previous timestamp output and current input and decide to keep the previous cell state value or not. Output of this layer between 0 and 1. 0 means completely remove cell state and 1 means keep the state.

To overcome the vanishing gradient problem, we need a function whose second derivative can sustain for a long range before going to zero. tanh is a suitable function with the above property.


$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right)$
$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right)$
$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$
(2)
$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$
(3)
$o_t = \sigma\left(W_o \, [h_{t-1}, x_t] + b_o\right)$
$h_t = o_t * \tanh\left(C_t\right)$
h_t
C_{t-1}
f_t
i_t
$\tilde{C}_t$
o_t
σ
σ
tanh
σ
tanh
X_t

We added another two Dense Hidden layers as specified above which yield a 1x4 output. We predict all four features in one go.

Optimizer:

The Adam optimization algorithm is an extension to stochastic gradient descent that has recently seen broader adoption for deep learning applications in computer vision and natural language processing.

the authors list the attractive benefits of using Adam on non-convex optimization problems, as follows:

- Straightforward to implement.
- Computationally efficient.
- Little memory requirements.
- Invariant to diagonal rescale of the gradients.
- Well suited for problems that are large in terms of data and/or parameters.
- Appropriate for non-stationary objectives.
- Appropriate for problems with very noisy/or sparse gradients.
- Hyper-parameters have intuitive interpretation and typically require little tuning.

Adam is different to classical stochastic gradient descent.

Stochastic gradient descent maintains a single learning rate (termed alpha) for all weight updates and the learning rate does not change during training.

A learning rate is maintained for each network weight (parameter) and separately adapted as learning unfolds.

The method computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients.

The authors describe Adam as combining the advantages of two other extensions of stochastic gradient descent. Specifically:

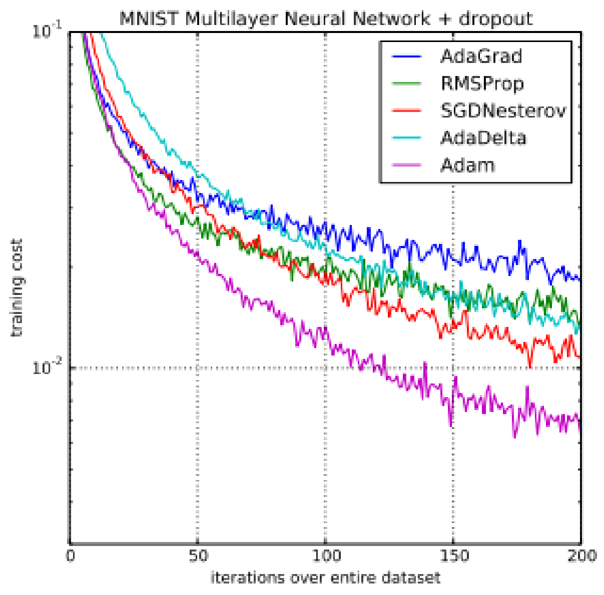
Adaptive Gradient Algorithm (AdaGrad) that maintains a per-parameter learning rate that improves performance on problems with sparse gradients (e.g. natural language and computer vision problems).
Root Mean Square Propagation (RMSProp) that also maintains per-parameter learning rates that are adapted based on the average of recent magnitudes of the gradients for the weight (e.g. how quickly it is changing). This means the algorithm does well on online and non-stationary problems (e.g. noisy).
Adam realizes the benefits of both AdaGrad and RMSProp.

Instead of adapting the parameter learning rates based on the average first moment (the mean) as in RMSProp, Adam also makes use of the average of the second moments of the gradients (the uncentered variance).

Specifically, the algorithm calculates an exponential moving average of the gradient and the squared gradient, and the parameters beta1 and beta2 control the decay rates of these moving averages.

The initial value of the moving averages and beta1 and beta2 values close to 1.0 (recommended) result in a bias of moment estimates towards zero. This bias is overcome by first calculating the biased estimates before then calculating bias-corrected estimates.

The paper is quite readable and I would encourage you to read it if you are interested in the specific implementation details.

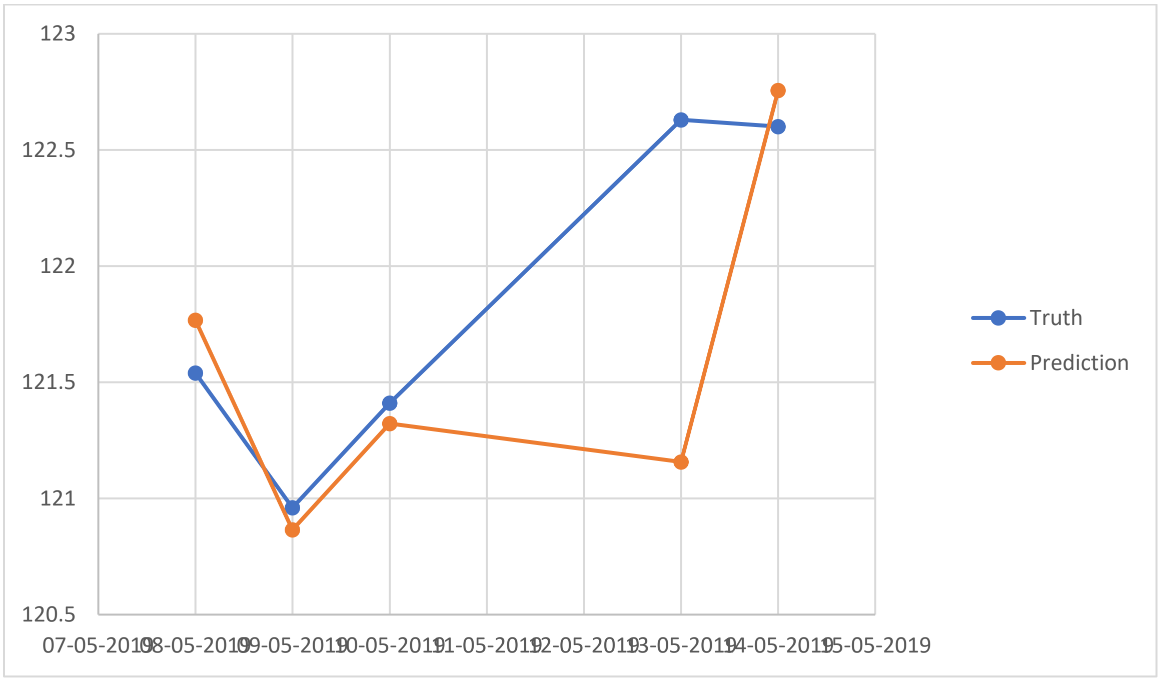


20 epopchs gave a balanced variance
MSE was used for Loss function.

Predictions:

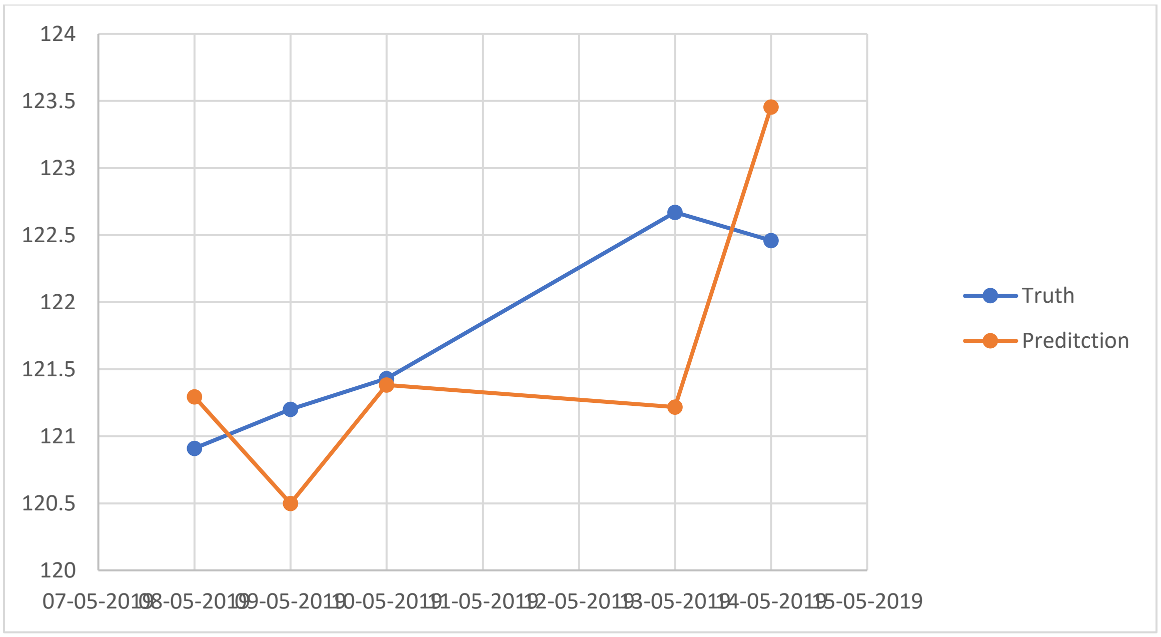
	Open	Close	High	Low
5/8	121.76718	121.29517	121.976654	121.35081
5/9	120.864914	120.49896	121.1847	120.418434
5/10	121.322525	121.38349	122.1955	121.28582
5/13	121.15658	121.216774	121.569	120.72179
5/14	122.75632	123.456245	124.19715	122.170944

Open



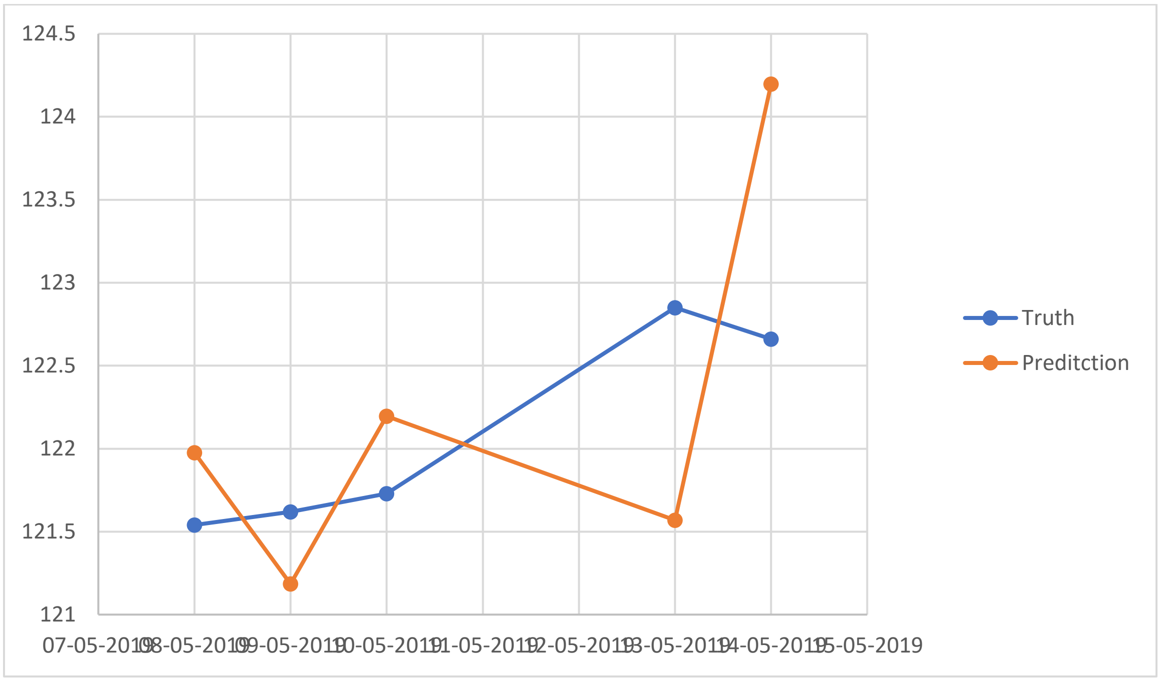
123
122.5
122
121.5
121
120.5
07-05-2019
08-05-2019
09-05-2019
10-05-2019
11-05-2019
12-05-2019
13-05-2019
14-05-2019
15-05-2019
Truth
Prediction

Close

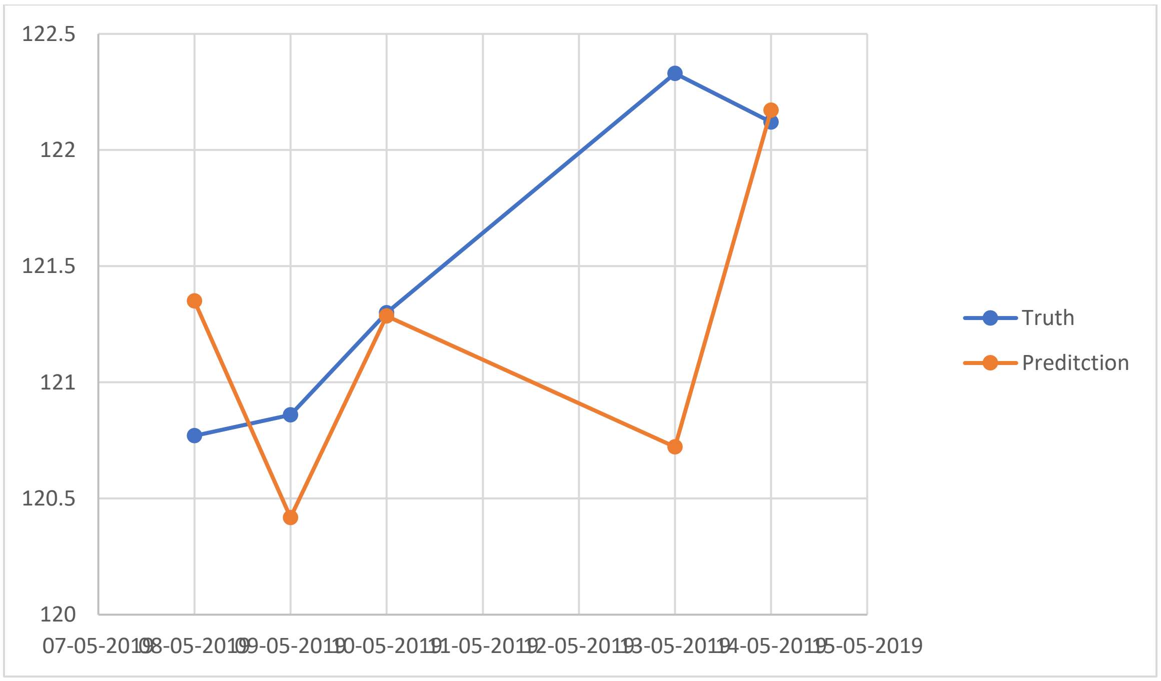


124
123.5
123
122.5
122
121.5
121
120.5
120
07-05-2019
08-05-2019
09-05-2019
10-05-2019
11-05-2019
12-05-2019
13-05-2019
14-05-2019
15-05-2019
Truth
Preditction

High


124.5
124
123.5
123
122.5
122
121.5
121
07-05-2019
08-05-2019
09-05-2019
10-05-2019
11-05-2019
12-05-2019
13-05-2019
14-05-2019
15-05-2019
Truth
Preditction

Low


122.5
122
121.5
121
120.5
120
07-05-2019
08-05-2019
09-05-2019
10-05-2019
11-05-2019
12-05-2019
13-05-2019
14-05-2019
15-05-2019
Truth
Preditction

Model Analysis:

- Worked better at predicting Opening price than other factors.
- Particularly weakest at predicting the highest price point.
- Did not take weekend gap into factor.
- Market was smoother over this week than expected.