Weight Log Analysis for IBM Machine Learning Class 6

Thomas Arnold, Senior Data Scientist

The purpose of these analyses was to determine whether it is possible to predict weight in advance. A one year weight log was used, and a Long Short Term Memory (LSTM) deep learning keras model was used to predict future weight from the previous few day's weight. Various model parameters were tried to see which produced the best test model without overfitting.

Dataset Description

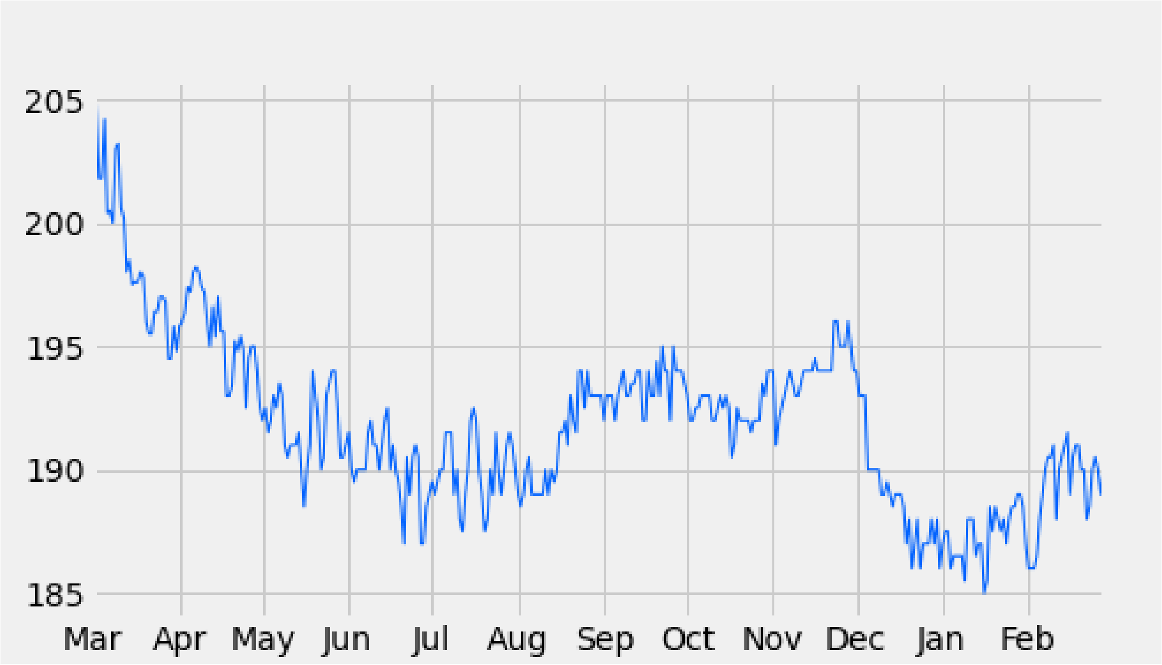
The dataset contained weight data from March 1st 2015 to February 28th 2016. The mean weight was 191.9 and the weight ranged from 185 to 204.8.

Description

Measure	Value
count	365
mean	191.9
std	3.4
min	185.0
25%	189.5
50%	192.0
75%	194.0
max	204.8

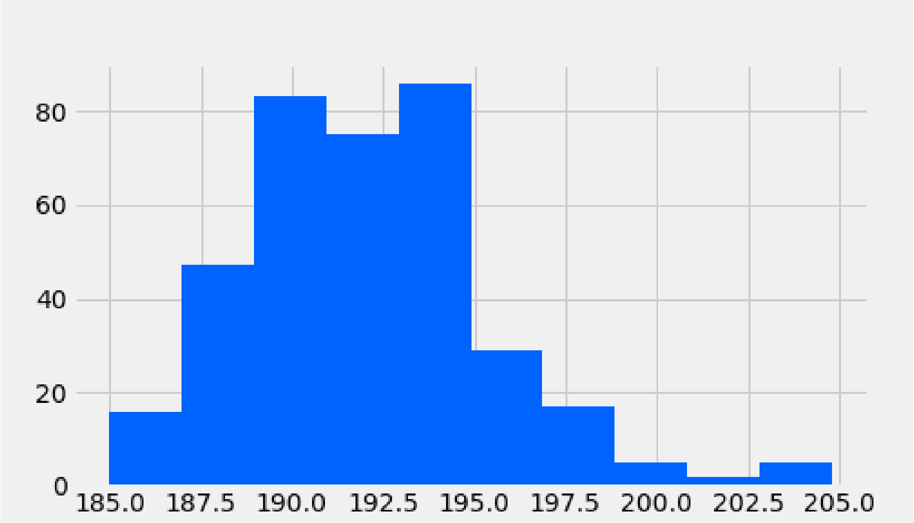
The time series is shown below. The data appeared to be trending downward.

One Year Weight Log



The data appeared to be somewhat normal, but was skewed slightly to the right.

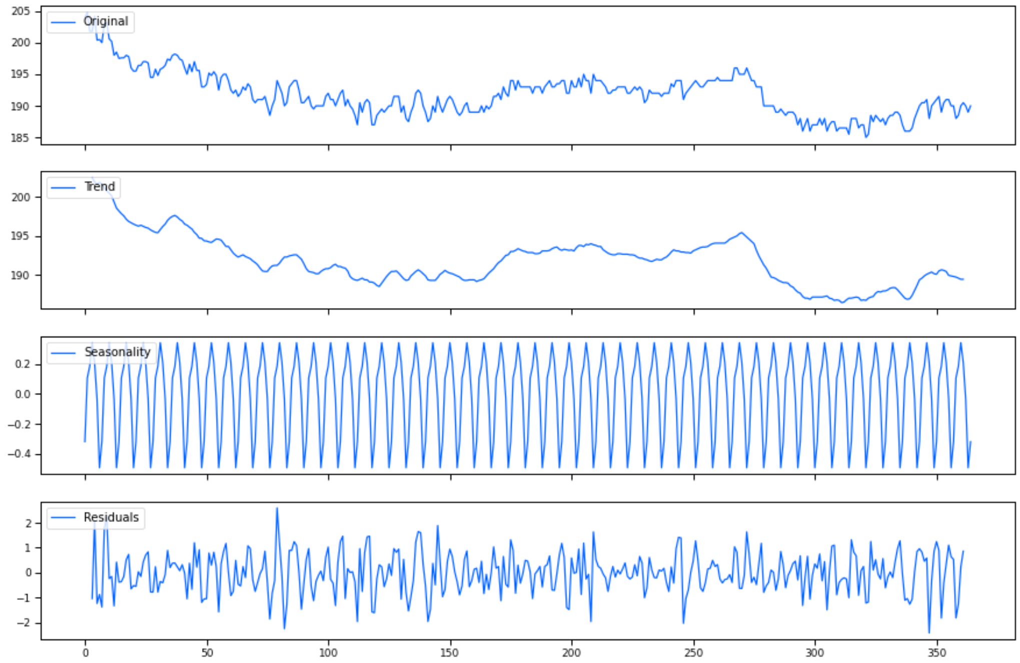
Histogram



Data Exploration

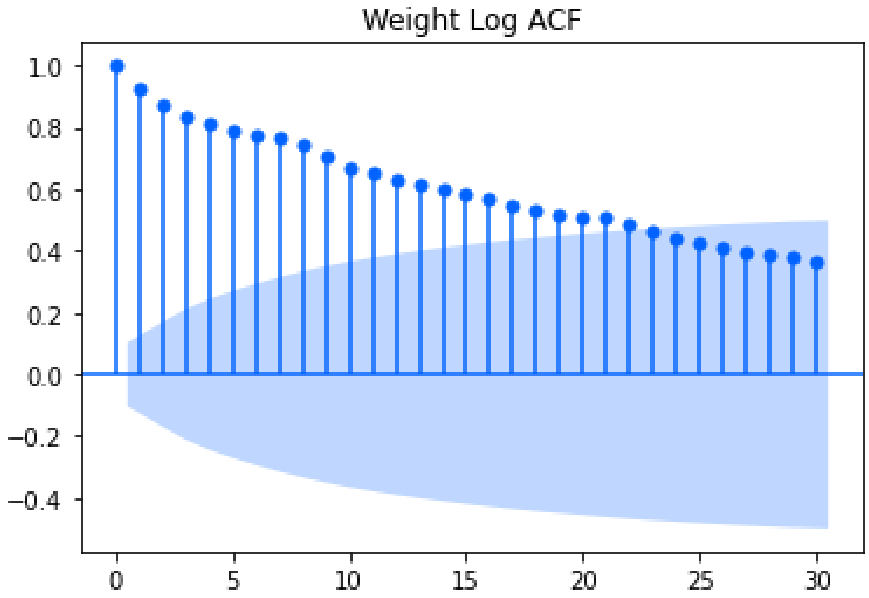
The first steps in the data analysis were to decompose the weight log data to see if there were and trend or seasonality. The weight was trending downward. It appeared that there was a 7 day seasonality pattern. The residuals looked like they were stationary.

Decomposition



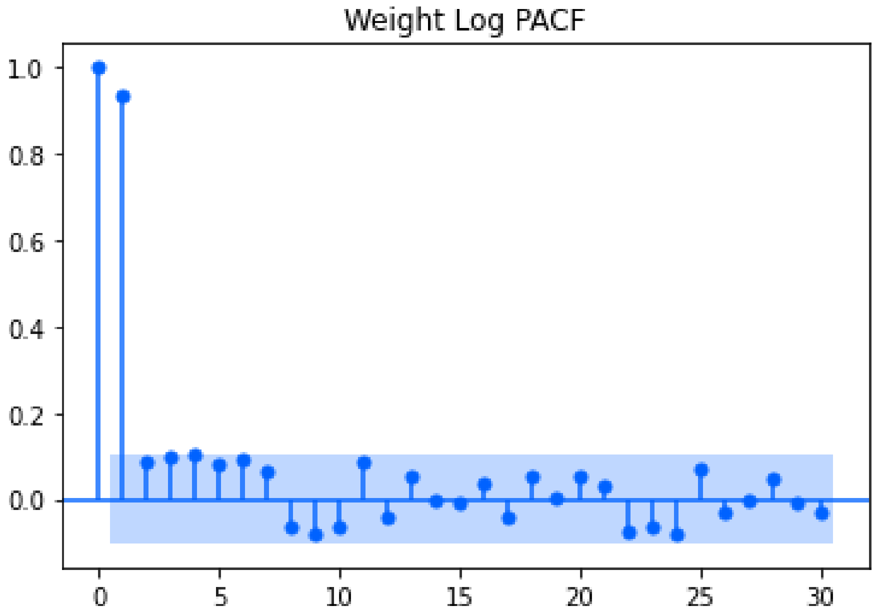
The autocorrelation plot suggested that there was a high degree of autocorrelation lasting out to about three weeks.

Autocorrelation Plot



The partial autocorrelation plot suggested that only the first two days were significantly correlated.

Partial Autocorrelation Plot



The Augmented Dickey Fuller test was used to determine whether the data was stationary. The ADF value was -3.15 and the p-value was .02 which is less than p = .05. Therefore, it appears that the data is stationary with a 5% confidence interval. The data would not be stationary with a 1% confidence interval.

Augmented Dickey Fuller Test Results

- Observations: 359
- ADF: -3.157995358696888
- p-value: 0.022548494733969433
- Critical Values: {'1%': -3.4486972813047574, '5%': -2.8696246923288418, '10%': -2.571077032068342}

Training the LSTM Deep Learning Model

An LSTM model was chosen because the goal was prediction and there were a large number of data points. The data was scaled using the min-max scaler to put all values on the same scale.

The simplest model was a LSTM with a one day lookback period. I used a 67% train set with a 33% test set. I tried varying the lookback period to see how that affected the RMSE for the train and test sets. I kept the model set at 100 epochs to start. I increased the lookback period from 1-8 days.

My base model was as follows.

- model = Sequential()
- model.add(LSTM(4, input_shape=(1, look_back)))
- model.add(Dense(1))
- model.compile(loss='mean_squared_error', optimizer='adam')
- model.fit(trainX, trainY, epochs=100, batch_size=1, verbose=2)

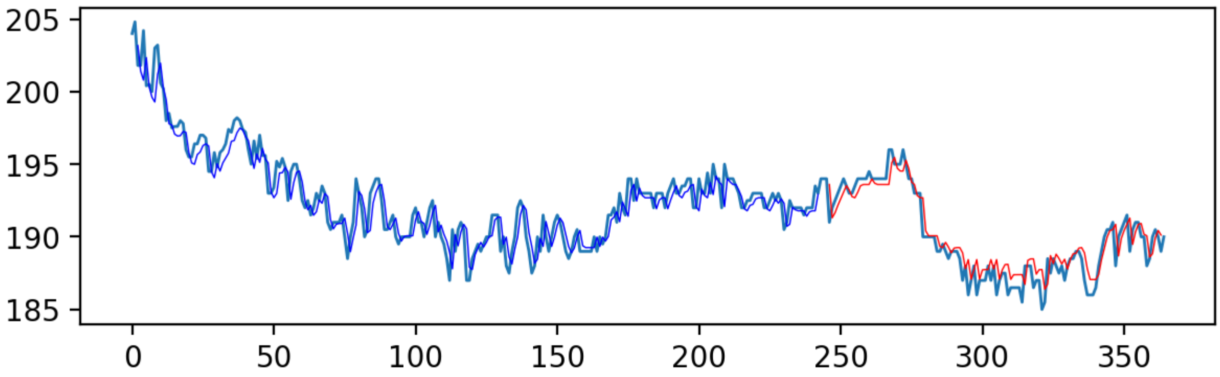
The RMSE for the train and test sets are shown below.

Lookback Days	Train RMSE	Test RMSE
1	1.14	1.04
2	1.12	1.16
3	1.1	1.1
4	1.1	1.18
5	1.07	1.11
6	1.02	1.09
7	1.02	1.12
8	1.02	1.09

It seemed that a one-day lookback period was optimal for predicting the next day's weight. Based on the high day to day correlation, this is perhaps not unexpected.

The plot for the one day lookback is shown below. The train predictions are dark blue and the test predictions are shown in red.

One Day Lookback for LSTM



Summary and Key Findings

The data had a high autocorrelation that was declining slightly over time. The partial autocorrelation was 1 to 2 days. There was a high degree of seasonality and a slight downward trend. The data appeared to be somewhat normal with a right skew. The Augmented Dickey Fuller test suggested that the data was stationary, but just barely.

The LSTM model seemed to perform best with a one day lookback period. This seems to be related to the high auto-correlation with the previous period.

Suggestions for Future Exploration

I would probably like to look at the seasonality a little closer. Is there a way to incorporate seasonality into the model? Should I be looking at the SARIMA model instead? What are the statistics for comparing SARIMA with LSTM?

I would also like to see if the future trend can be predicted. Predicting the next day seems to be a little simplistic. I am interested in how the downward trends over the space of several months reverses itself. Is there some way to use this model to predict whether slow weight loss is better than fast weight loss?

I tried adding another Dense layer but that seemed to make things worse. Are there other deep learning configurations that would improve predictions? It would be interesting to find out.