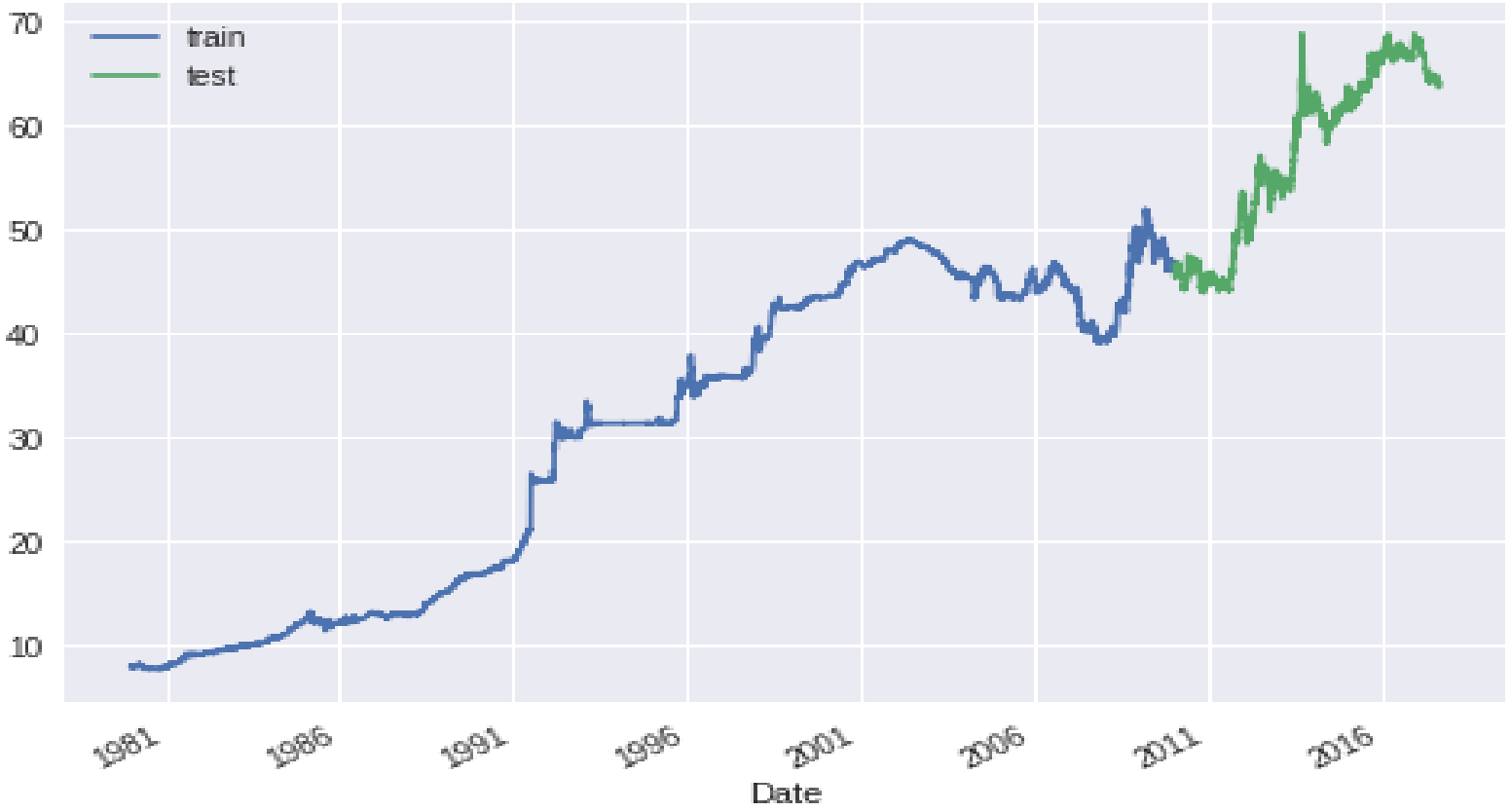
Predicting India - USA Exchange Rate using LSTM Time series forecasting

Why will this work when other economic datasets did not?

Data and Implementation

- We have a total of 13,730 records starting from January 2, 1980 to August 10, 2017

Train - Test (split date - 1-1-2010)



First : A Multi layer perceptron Network

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 12)                24
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 13
=================================================================
Total params: 37
Trainable params: 37
Non-trainable params: 0
_________________________________________________________________
```

First : A Multi layer perceptron Network

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 12)                24
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 13
=================================================================
Total params: 37
Trainable params: 37
Non-trainable params: 0
_________________________________________________________________
```

```
Epoch 1/200
7712/7712 [==============================] - 6s 822us/step - loss: 1.7679e-05
Epoch 2/200
7712/7712 [==============================] - 6s 723us/step - loss: 2.4265e-04
Epoch 3/200
7712/7712 [==============================] - 6s 722us/step - loss: 8.0999e-05
Epoch 00003: early stopping



 The R2 score on the Train set is:       0.924
 The Adjusted R2 score on the Train set is:      0.924

 The R2 score on the Test set is:        0.900
 The Adjusted R2 score on the Test set is:       0.900
```

Sliding window (n = 50) LSTM

Layer (type)	Output Shape	Param #
lstm_8 (LSTM)	(None, 50)	12600
dropout_7 (Dropout)	(None, 50)	0
dense_8 (Dense)	(None, 1)	51
activation_3 (Activation)	(None, 1)	0

Total params: 12,651
Trainable params: 12,651
Non-trainable params: 0

```
Train on 6887 samples, validate on 766 samples
Epoch 1/5
6887/6887 [==============================] - 21s 3ms/step - loss: 0.0034 - val_loss: 2.8487e-04
Epoch 2/5
6887/6887 [==============================] - 21s 3ms/step - loss: 0.0032 - val_loss: 3.2886e-04
Epoch 3/5
6887/6887 [==============================] - 21s 3ms/step - loss: 0.0031 - val_loss: 1.9685e-04
Epoch 4/5
6887/6887 [==============================] - 21s 3ms/step - loss: 0.0030 - val_loss: 1.8710e-04
Epoch 5/5
6887/6887 [==============================] - 21s 3ms/step - loss: 0.0029 - val_loss: 5.5229e-04
> Compilation Time :  106.50379467010498
```

```
model_ann = load_model('ANN')
model_lstm = load_model('LSTM')
score_ann= model_ann.evaluate(X_test_ann, y_test_ann, batch_size=1)
score_lstm= model_lstm.evaluate(x_test, y_test, batch_size=1)
```

```
1984/1984 [==============================] - 1s 636us/step
1935/1935 [==============================] - 19s 10ms/step
```

```
print('ANN: %f'%score_ann)
print('LSTM: %f'%score_lstm)
```

```
ANN: 0.003334
LSTM: 1.134302
```

Simple LSTM

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 7)                 252
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 8
=================================================================
Total params: 260
Trainable params: 260
Non-trainable params: 0
```

Simple LSTM

```
model_ann = load_model('ANN')
model_rnn = load_model('LSTM2')
score_ann= model_ann.evaluate(X_test_ann, y_test_ann, batch_size=1)
score_lstm= model_rnn.evaluate(X_tst_t, y_test_ann, batch_size=1)
```

```
1984/1984 [==============================] - 1s 440us/step
1984/1984 [==============================] - 1s 680us/step
```

```
print('ANN: %f'%score_ann)
print('LSTM: %f'%score_lstm)
```

```
ANN: 0.003334
LSTM: 0.000672
```

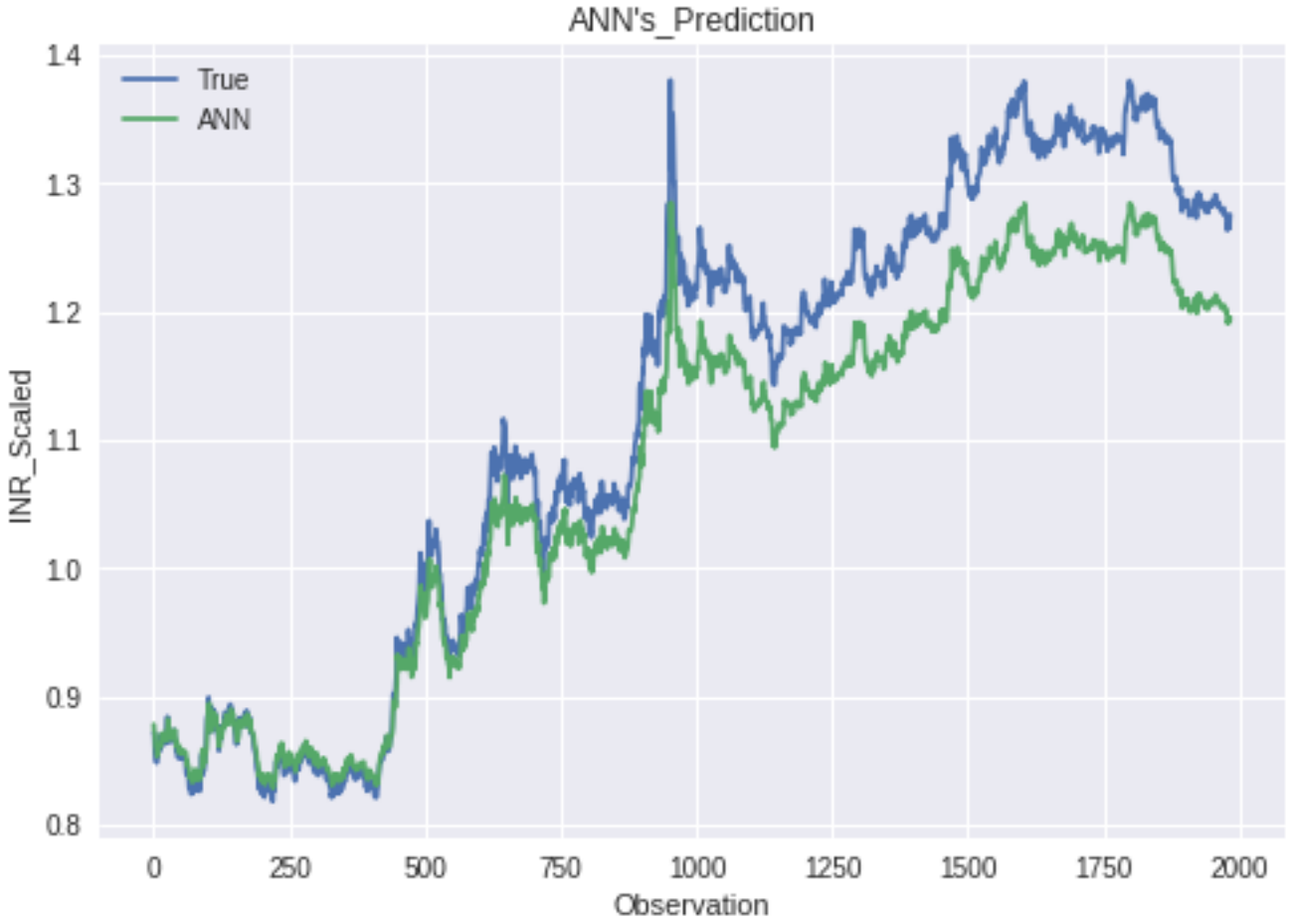
Simple LSTM

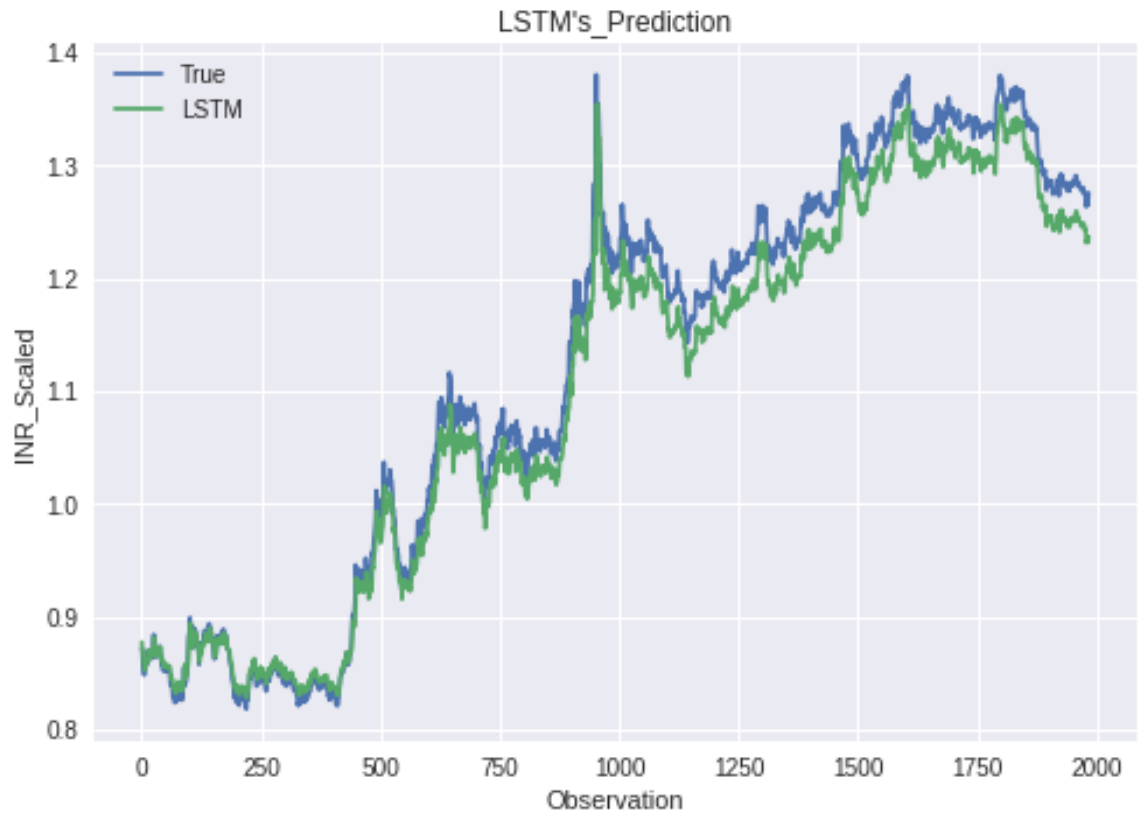
```
model_ann = load_model('ANN')
model_rnn = load_model('LSTM2')
score_ann= model_ann.evaluate(X_test_ann, y_test_ann, batch_size=1)
score_lstm= model_rnn.evaluate(X_tst_t, y_test_ann, batch_size=1)
```

```
1984/1984 [==============================] - 1s 440us/step
1984/1984 [==============================] - 1s 680us/step
```

```
print('ANN: %f'%score_ann)
print('LSTM: %f'%score_lstm)
```

```
ANN: 0.003334
LSTM: 0.000672
```


ANN's_Prediction
True
ANN
INR_Scaled
Observation
14
13
12
11
10
09
08
0
250
500
750
1000
1250
1500
1750
2000


LSTM's_Prediction
True
LSTM
INR_Scaled
Observation
14
13
12
11
10
09
08
0
250
500
750
1000
1250
1500
1750
2000