AMRITA VISHWA VIDYAPEETHAM

B Tech -AIE

21MAT301 MATHEMATICS FOR INTELLIGENT SYSTEMS - 5

TEMPERATURE FORECASTING

USING DYNAMIC MODE DECOMPOSITION

- Dr. Gopalakrishnan

Submitted on

18 - 01 – 2023, Wednesday

Submitted by

Team - 06

Menta Sai Akshay - CB.EN. U4AIE20040

Penaka Vishnu Reddy - CB.EN. U4AIE20048

Sajja Bala Karthikeya - CB.EN. U4AIE20065

Dharavathu Rohith - CB.EN. U4AIE20060

Rachure Charith Sai - CB.EN. U4AIE20055

Abstract

Forecasting temperature has many implications since temperature variation in one location might have a significant impact on that particular area. This essay seeks to project.

Utilizing Dynamic Mode Decomposition, temperature (DMD). The DMD method is entirely data-driven. The future behaviour of the system can be predicted using the DMD modes that capture its dynamic activity. The ARIMA model, RNNs, and other machine learning algorithms, among others, can be used to predict temperatures. DMD doesn't need any training, in contrast to currently utilized machine learning methods. The use of DMD for this purpose has not been extensively studied in the literature.

This project used daily temperature data to forecast the weather for the following few days. A different sampling window was used for the prediction. The Root Mean Squared Error was used to calculate the error.

Introduction

Temperature forecasting is a crucial task in many fields, including meteorology, agriculture, and energy management. Accurate temperature forecasts can help individuals and organizations plan and prepare for extreme weather events, such as heat waves or cold snaps. One method for forecasting temperature is through the use of dynamic mode decomposition (DMD). DMD is a data-driven method that can extract the underlying dynamics of a system from a time series of measurements. It has been used to study a wide range of phenomena, including fluid dynamics, signal processing, and control systems.

In this paper, we will introduce the use of DMD for temperature forecasting. We will provide an overview of the DMD method and its mathematical foundations, as well as its application to temperature forecasting. We will demonstrate the effectiveness of DMD for temperature forecasting through case studies and comparisons to other forecasting methods. Additionally, we will discuss the potential benefits and limitations of using DMD for temperature forecasting and provide recommendations for future research in this area.

Background:

DMD is a data-driven method for identifying the dominant modes of a system and forecasting its future behavior. The method was first introduced by Schmid and Sesterhenn in 2010, and has since been applied to a wide range of fields, including fluid dynamics and control systems. The basic idea behind DMD is to represent the system as a linear combination of its dominant modes. The dominant modes are found by decomposing the data into a set of spatiotemporal patterns, which can then be used to predict the future behavior of the system.

Methodology:

Data Description

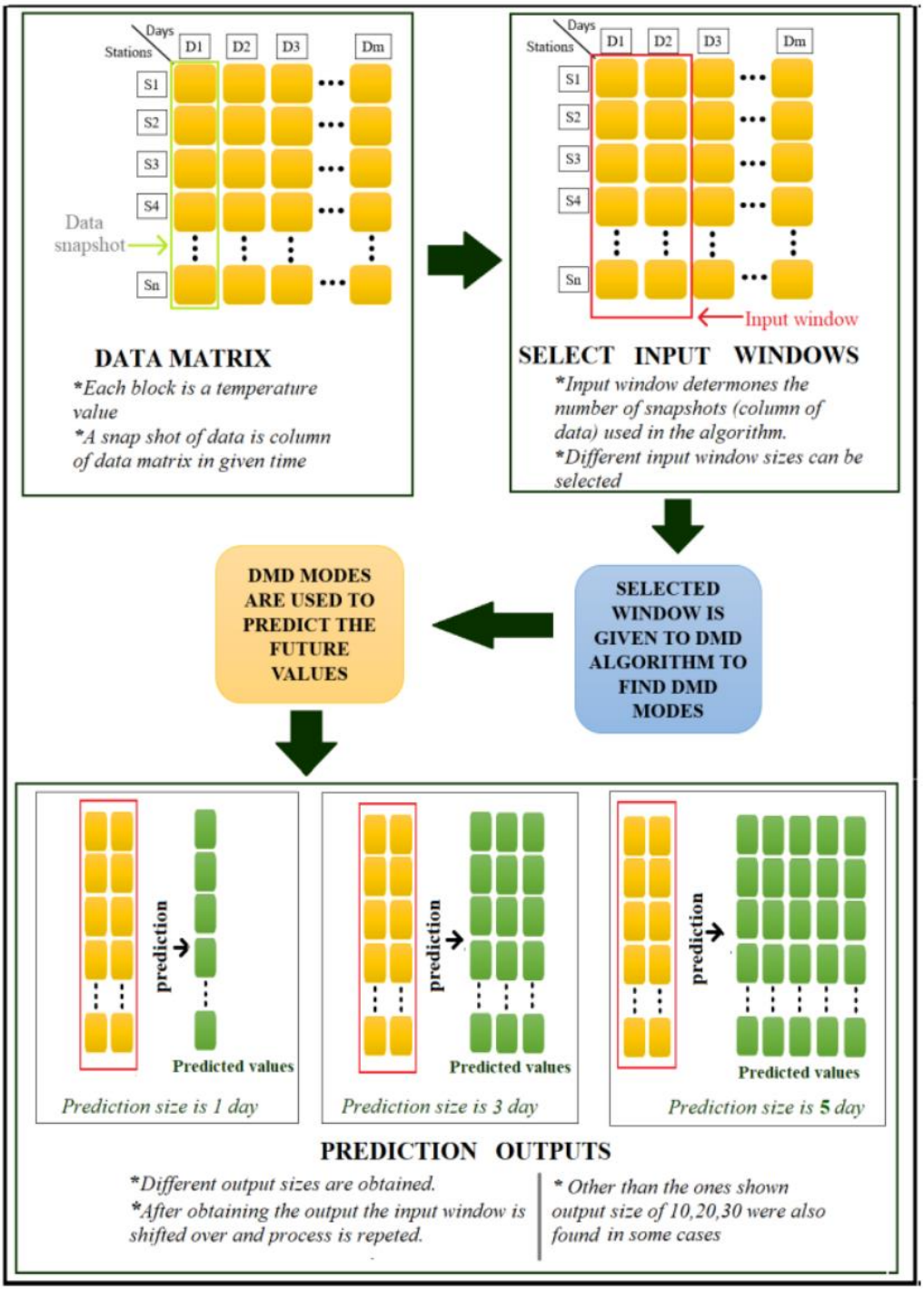
The data set used for the work is the Global Summary of the Day (GSOD) provided by NOAA and it can be obtained from the NOAA website. Data has been obtained for the year 2019. Data was pre-processed using python. The final data is having 190 stations and its temperature for 100 days temperature data.

	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	...	D91	D92	D93	D94	D95	D96	D97	D98	D99	D100
1001099999	24.0	36.2	35.5	35.8	35.8	28.1	23.5	24.0	36.7	22.2	...	25.2	21.6	24.5	32.6	31.6	26.0	28.9	30.9	32.1	33.2
1001499999	39.7	36.4	36.5	45.6	42.5	38.9	40.6	35.9	38.4	39.7	...	51.4	51.4	60.1	66.4	62.5	60.5	58.7	55.3	50.2	57.5
1002099999	-0.1	5.1	19.7	14.3	16.1	20.0	18.8	16.1	19.9	20.1	...	-0.7	4.1	10.3	12.1	18.2	14.0	9.3	5.9	21.3	14.5
1003099999	3.9	11.6	27.8	23.6	27.9	29.0	25.6	24.9	22.4	25.6	...	14.6	16.0	20.1	23.4	25.9	21.5	15.4	13.8	27.2	29.8
1006099999	5.2	4.3	14.2	16.0	18.8	18.9	21.2	16.5	21.6	22.0	...	4.8	5.0	7.9	12.7	22.2	14.2	13.5	8.3	20.0	20.7
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1374099999	33.0	26.9	30.5	39.4	33.7	31.7	25.1	29.2	24.4	23.7	...	30.1	27.9	29.9	38.8	33.6	33.6	32.4	31.7	29.8	26.7
1375099999	25.5	21.6	30.0	36.1	34.1	32.8	23.6	22.9	21.0	26.5	...	24.4	22.9	28.3	34.6	28.7	29.1	25.8	22.4	20.8	18.8
1375599999	40.0	40.2	44.3	48.1	46.6	44.5	44.1	41.0	40.2	48.1	...	41.3	41.7	44.5	46.4	49.2	44.9	44.2	42.9	43.2	41.9
1376099999	40.6	31.9	28.4	42.1	29.0	26.1	25.9	35.2	29.5	23.1	...	33.9	36.2	33.9	43.3	39.7	40.0	39.2	39.3	35.4	34.8
1378099999	33.8	26.9	24.7	30.7	33.3	32.2	27.3	33.1	28.3	23.2	...	30.3	33.6	32.6	40.1	35.5	34.6	35.3	35.0	33.2	30.3

190 rows × 100 columns

Approach

The purpose of this work is to create a forecast for temperature values of the future. The data was fed to the DMD algorithm in an input window size, which makes the matrix size as (190 x input) and prediction was done to different length. The input windows experimented with were 5, 10, 20, 30, 40, 50 and each of this window size were used as input size to predict different output size like one day prediction, three day prediction, five day prediction and so on.



Dynamic Mode Decomposition (DMD):

DMD is a data driven and equation free algorithm it mainly relies on measurements, like experimental data and numerical simulations. Assumptions about the under lying system is not required for DMD, this make it a popular data driven technique.

The measurements are set in two matrices, one of them has a time shift, in order that the DMD is enabled to approximate the dynamics matrix, A

$$\mathbf{X_1} = \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_{m-1} \\ | & | & & | \end{bmatrix} \quad \mathbf{X_2} = \begin{bmatrix} | & | & & | \\ \mathbf{x}_2 & \mathbf{x}_3 & \cdots & \mathbf{x}_{m} \\ | & | & & | \end{bmatrix}$$

$$\mathbf{X_{aug,1}} = \begin{bmatrix} x_1 & x_2 & \cdots & x_{m-s} \\ x_2 & x_3 & \cdots & x_{m-s+1} \\ \vdots & \vdots & \ddots & \vdots \\ x_s & x_{s+1} & \cdots & x_{m-1} \end{bmatrix} \quad \mathbf{X_{aug,2}} = \begin{bmatrix} x_2 & x_3 & \cdots & x_{m-s+1} \\ x_3 & x_4 & \cdots & x_{m-s+2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{s+1} & x_{s+2} & \cdots & x_m \end{bmatrix}$$

From our system we are considering two-time snapshot matrices X and X' where X' is shifted forward by one step from X. An operator matrix A is defined as,

$$X' = AX \qquad \text{.... (i)}$$

Equation (i) can be rewritten as,

$$A = X'X^{+}$$

Where $X^{+}$is the Moore-Penrose pseudo-inverse. Let p be the number of measurement points then the dimension of the matrix X will be p x m (which could possibly be very huge matrix), To avoid direct computation of matrix a low rank SVD of X is used. The SVD decomposition of X can be written as,

$$X = U\Sigma V^{H} \qquad \text{…(iii)}$$

Where $U \in C^{pxr}$, $\Sigma \in C^{rxr}$ and $V \in C^{mxr}$, The A matrix could be approximated to a low rank equivalent matrix $\tilde{A}$. $\tilde{A}$ can be re written as,

$$\tilde{A} = U^{H}AU = X'V\Sigma^{-1}U^{H} \qquad \text{…(iv)}$$

The eigenvalues and eigenvectors of $\tilde{A}$can be calculated as W and D and the DMD modes can be found as,

$$\phi = X'V\Sigma^{-1}W \qquad \ldots(v)$$

Here ϕ is the DMD modes. The continues time frequencies can be obtained as,

$$w_i = \frac{\ln\wedge}{dt} \qquad \ldots(vi)$$

Where is the DMD eigenvalue and dt is the interval.

$$x_{DMD}(t) = \sum_{i=1}^{n} e^{wit} b_i \phi_i \qquad \ldots(vii)$$

Here $x_{DMD}(t)$ represent the system at time t, ϕ is the DMD modes, w is the eigenvalues of matrix A and b is constant which can be obtained by finding $x_{DMD}(0)$. This can be used to provide the state of the system at any time t and can be used to predict future state of the system.

Root Mean Square Error (RMSE):

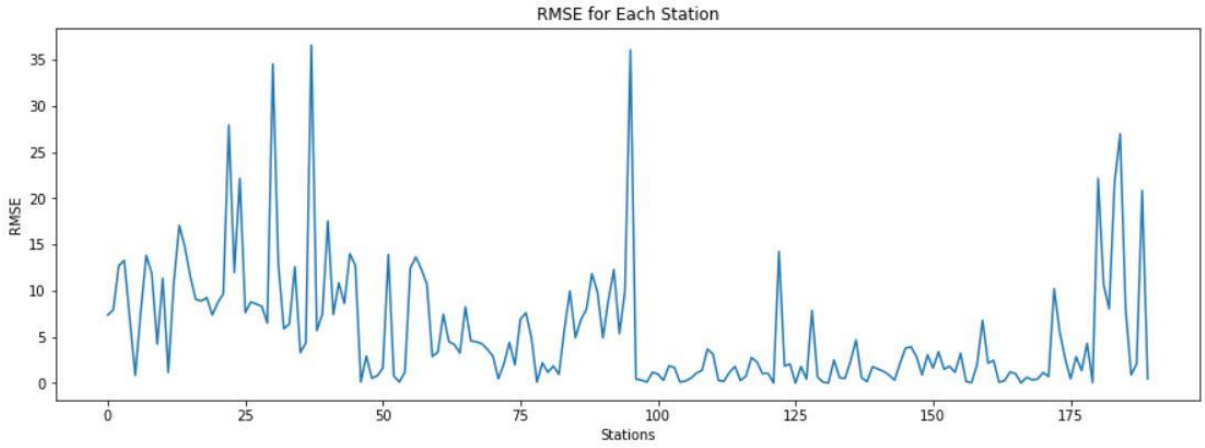
Root mean square error is used to calculate how spread out the data points are, from a line of best fit or a regression line. It can be said that it is the standard deviation of the distance between the data points and regression line thus it is regularly used in time series analysis. The RMSE can be calculated as,

$$RMSE = \sqrt{\frac{1}{N}\sum(P-A)^2} \qquad \ldots(viii)$$

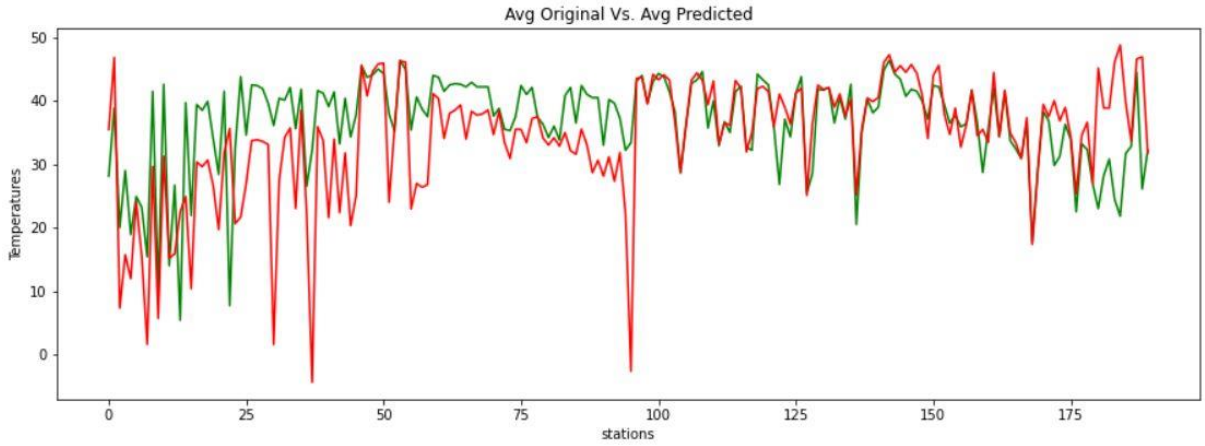
Where N is the total number of data points, P is the predicted value and A is the actual value

Results And Discussions

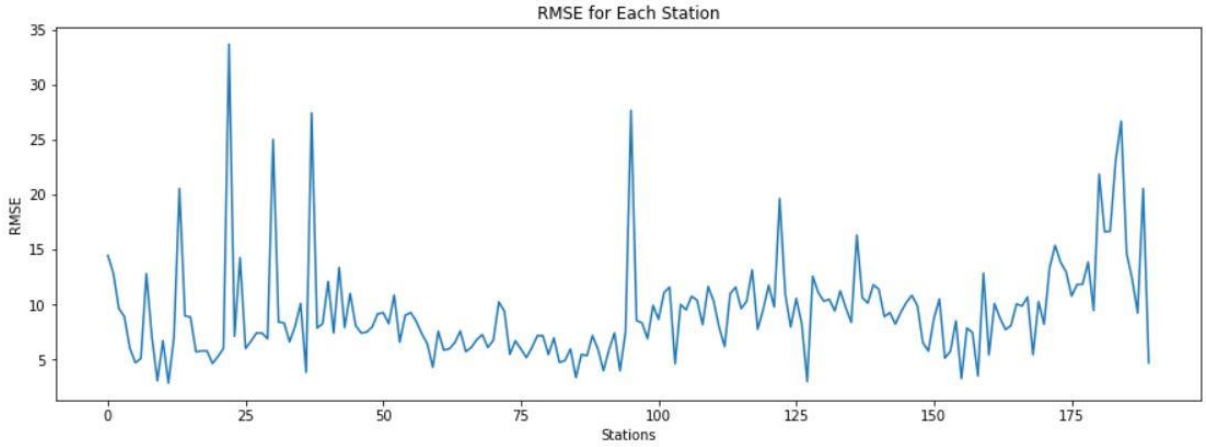
For Input window size of 5



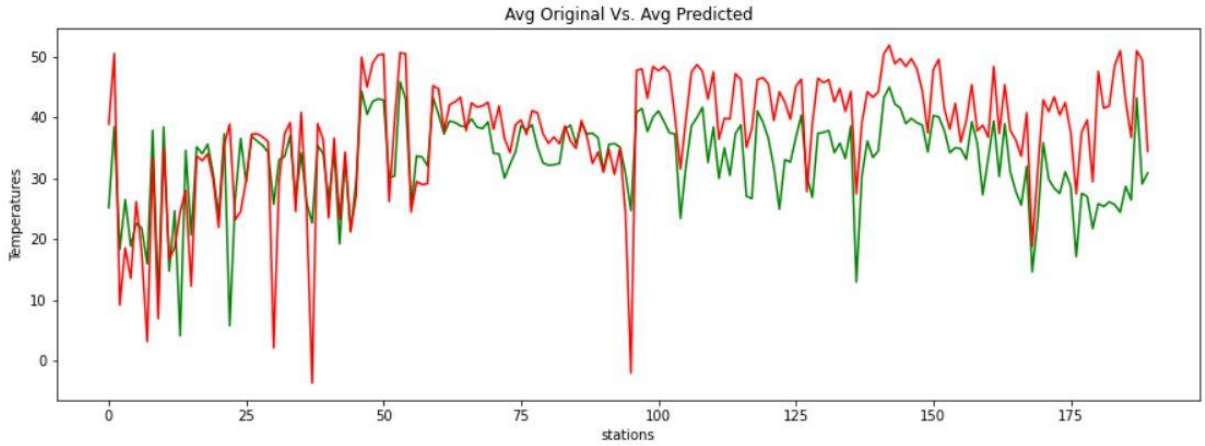
The graph infers the RMSE value of each station for input window of 5 and predicting the temperature of one day



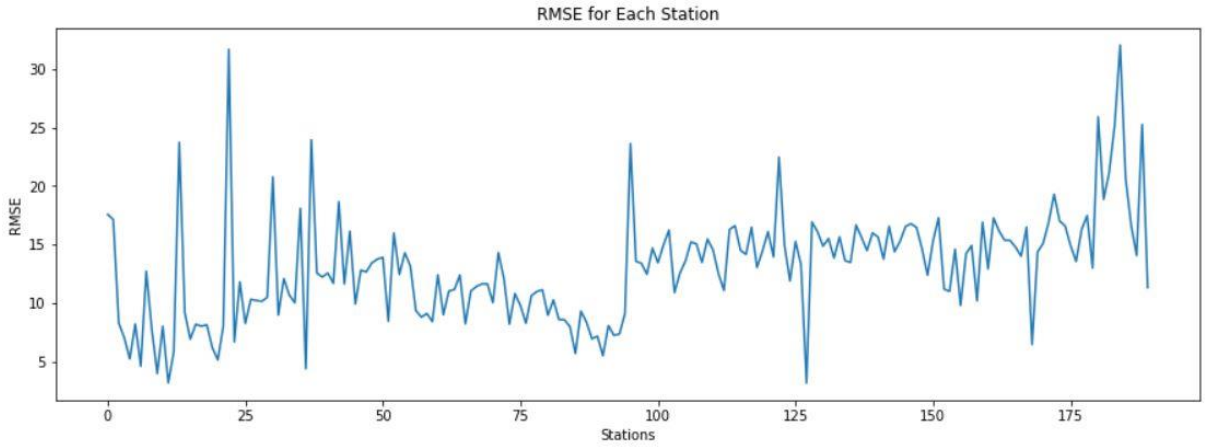
Avg of Original Temperature Vs Avg of Predicted Temperature



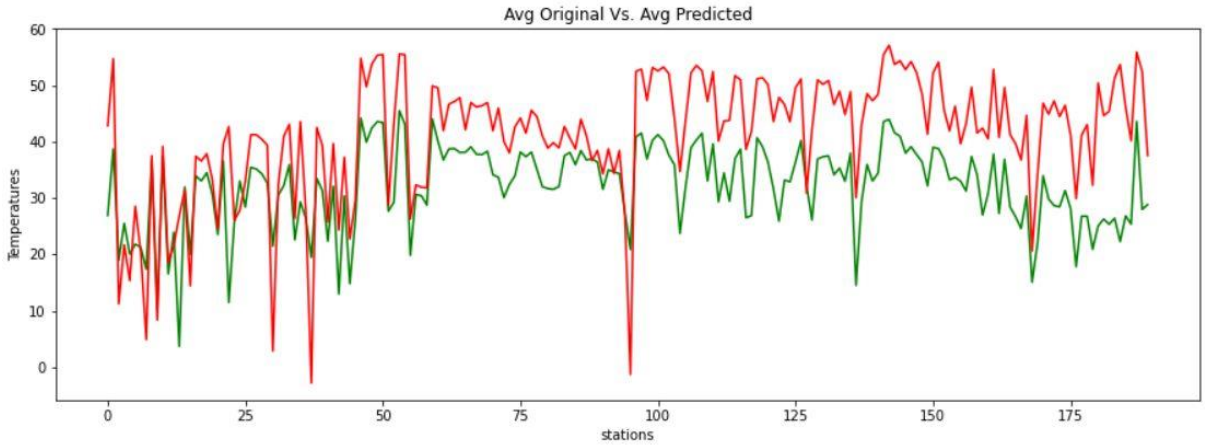
The graph infers the RMSE value of each station for input window of 5 and predicting the temperature of three days



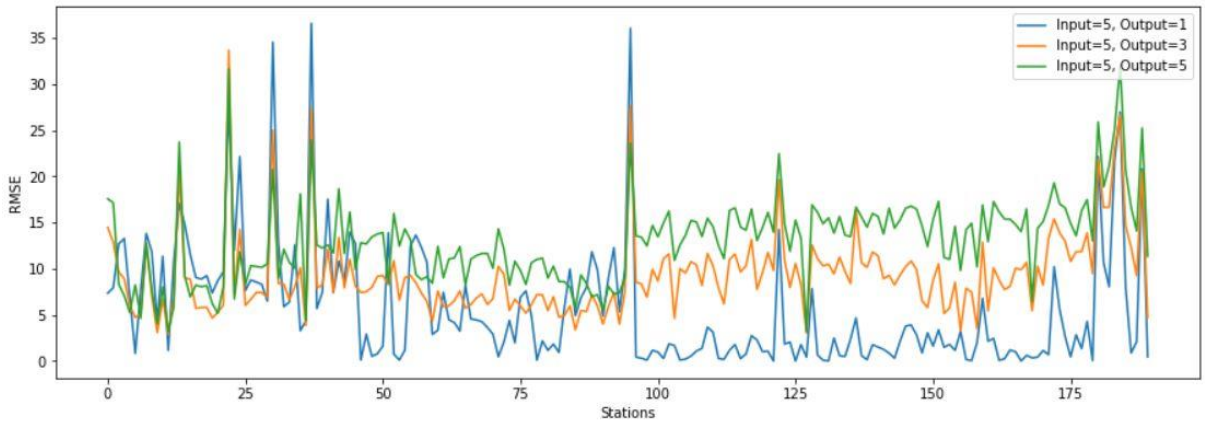
Avg of Original Temperature Vs Avg of Predicted Temperature



The graph infers the RMSE value of each station for input window of 5 and predicting the temperature of five days



Avg of Original Temperature Vs Avg of Predicted Temperature

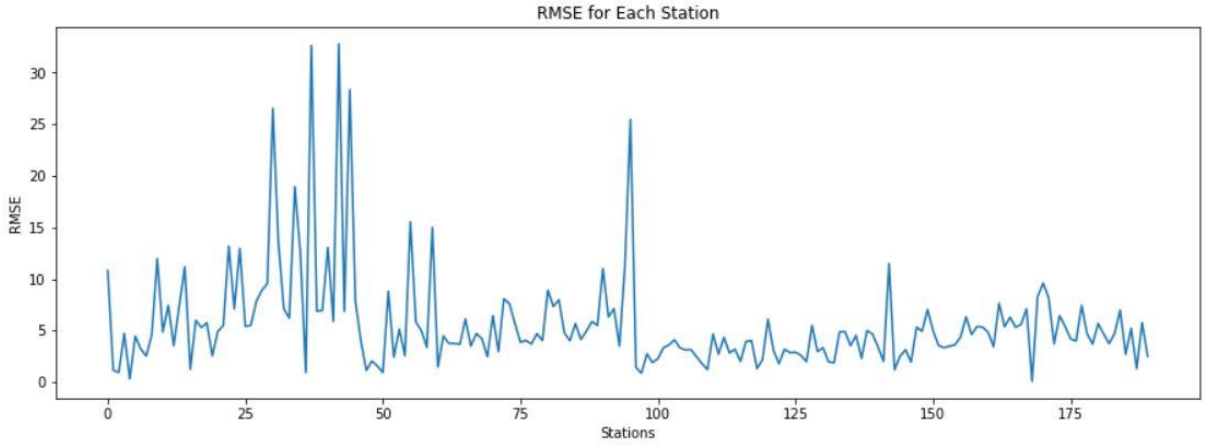


Finally this graph presents the RMSE values of each station for input size of 5 days and predicting one day (blue line) three day (Orange line) five days (Green line) and from the graph we can infer that the error is more while predicting more no of days with the same input of 5 days

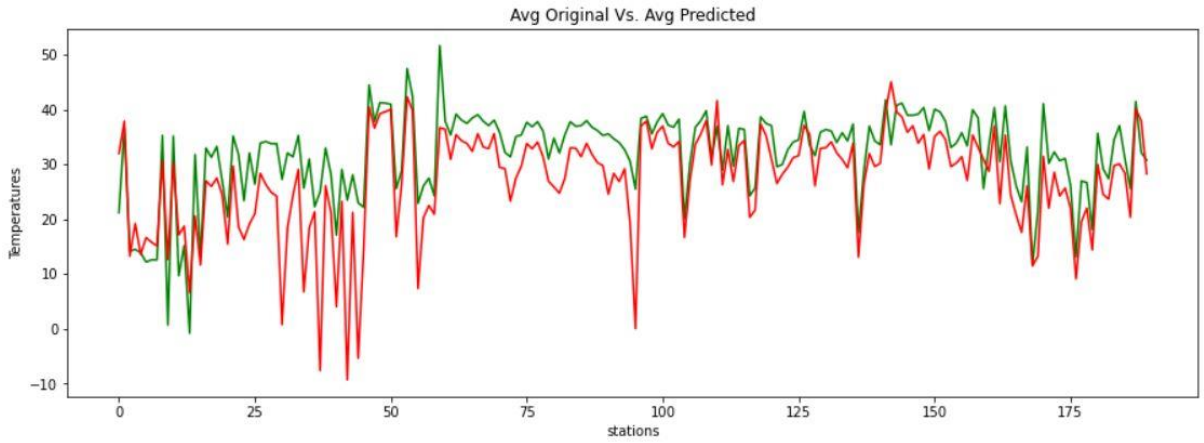
	RMSE
Out=1	5.665343
Out=3	9.360380
Out=5	13.004502

These are the RMSE value for the respective output for input 5 days

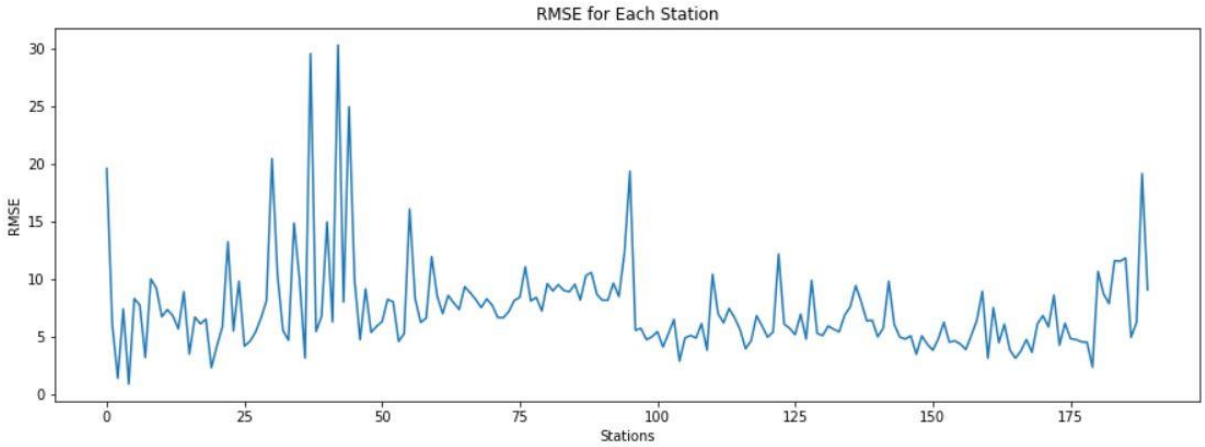
For Input window size of 10



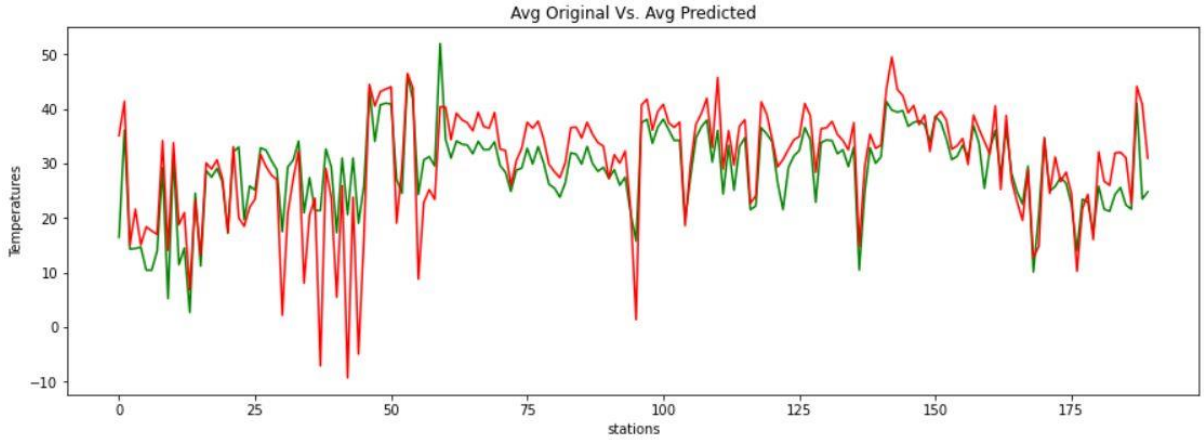
The graph says about the RMSE value of each station for input window of 10days and predicting one day's temperature



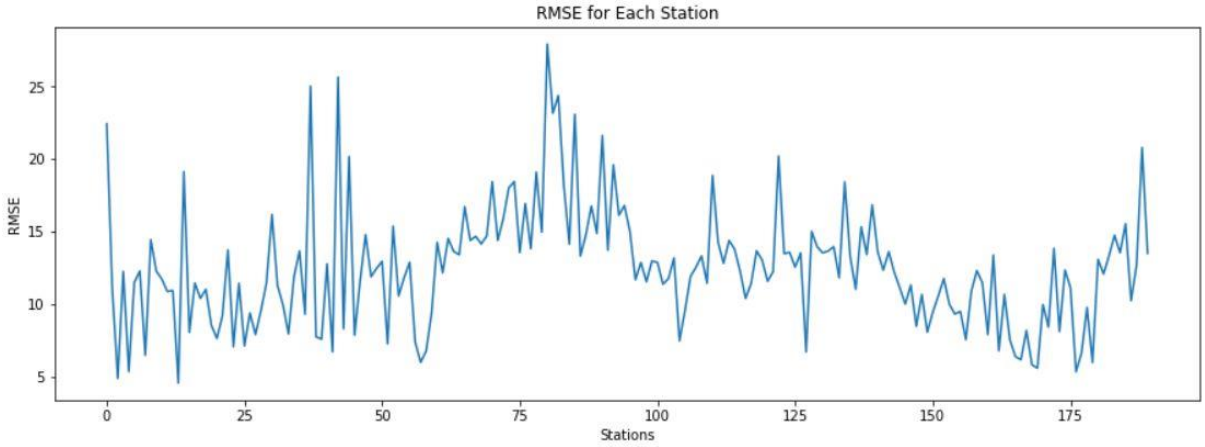
Avg of Original Temperature Vs Avg of Predicted Temperature



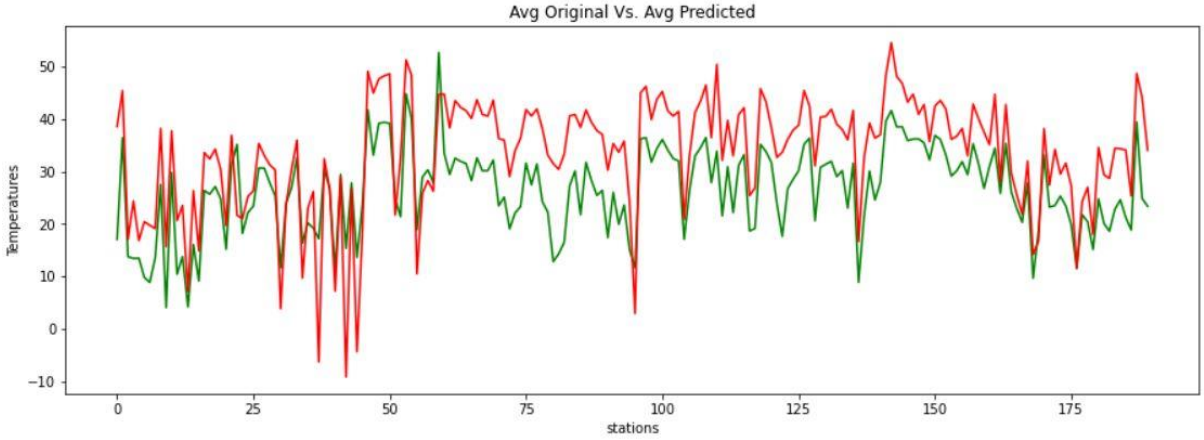
The graph says about the RMSE value of each station for input window of 10days and predicting three days temperature



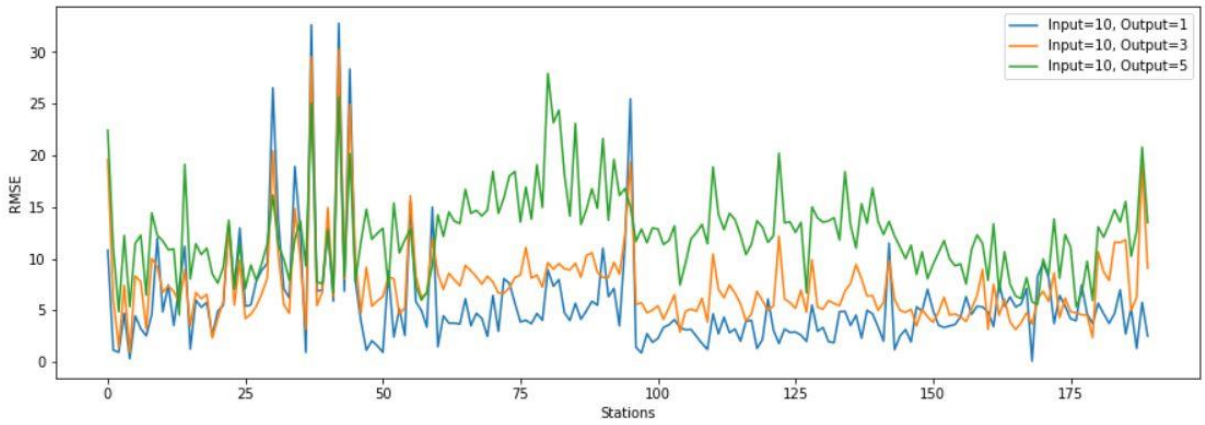
Avg of Original Temperature Vs Avg of Predicted Temperature



The graph says about the RMSE value of each station for input window of 10days and predicting five days temperature



Avg of Original Temperature Vs Avg of Predicted Temperature

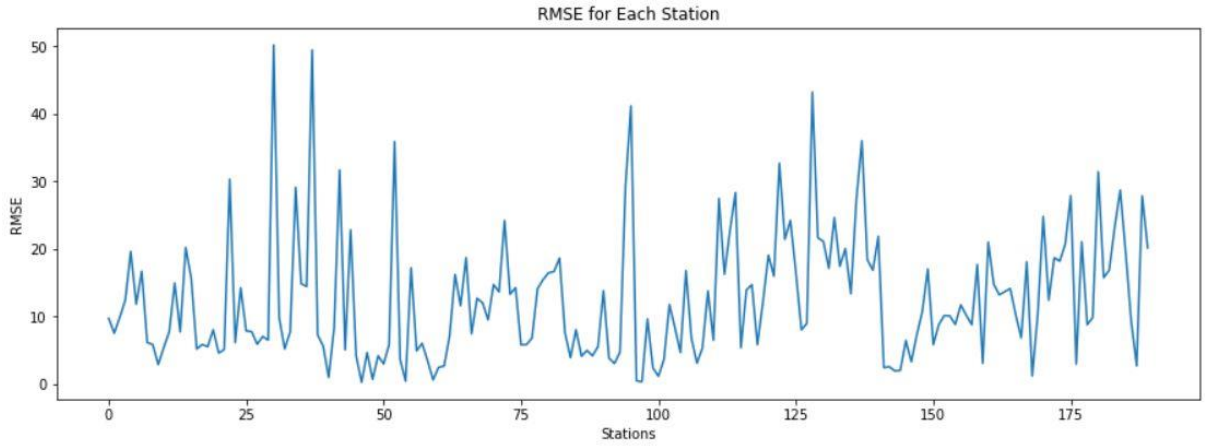


Finally this graph presents the RMSE values of each station for input size of 10 days and predicting one day, three day & five days. From the graph we can infer that the error is more while predicting more no of days with the same input of 10 days

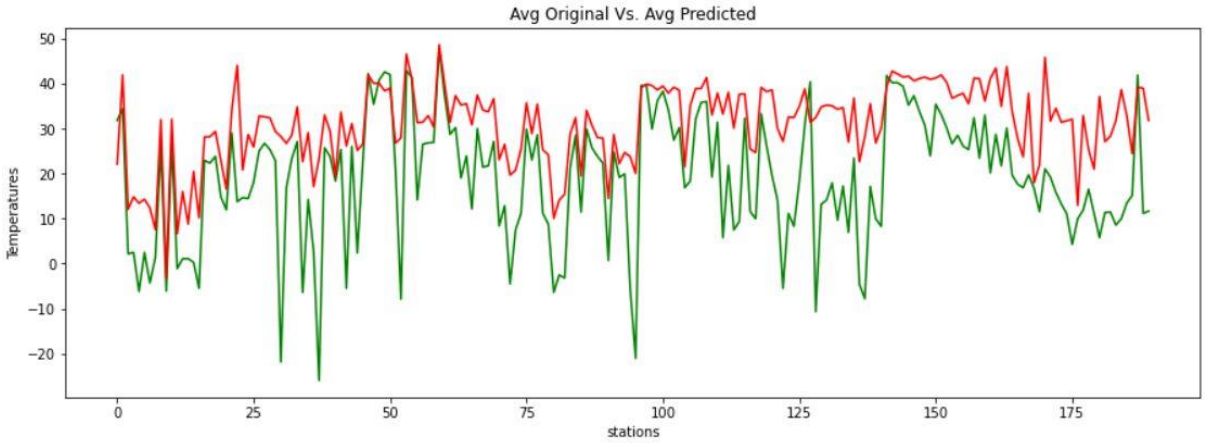
	RMSE
Out=1	5.647761
Out=3	7.432961
Out=5	12.435442

These are the RMSE value for the respective output for input 10 days

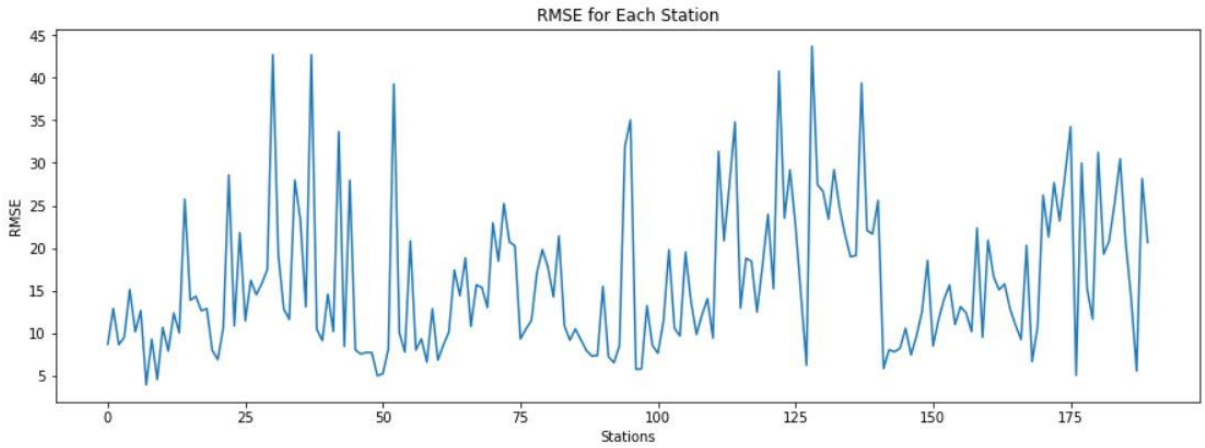
For Input window size of 20



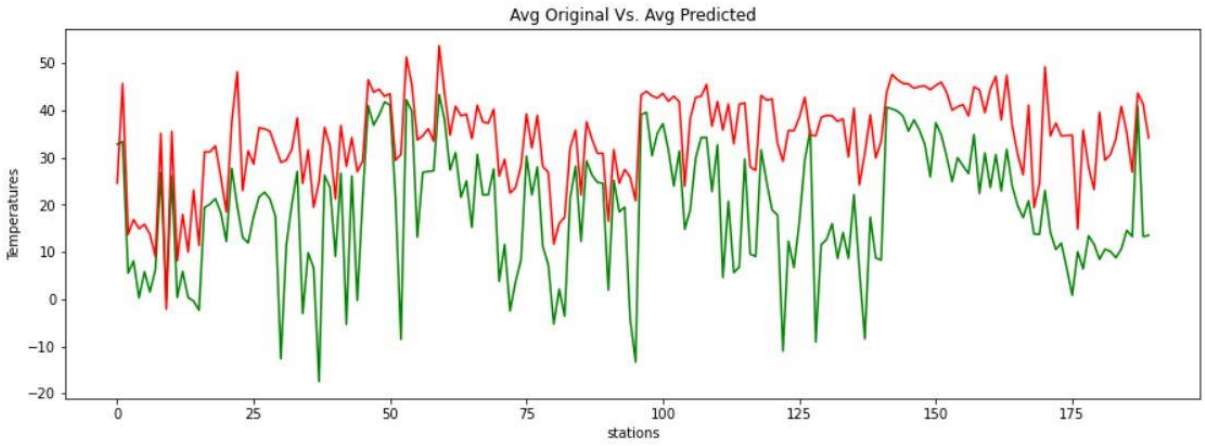
The graph is RMSE value VS station graph for input of 20 days and predicting one day's temperature



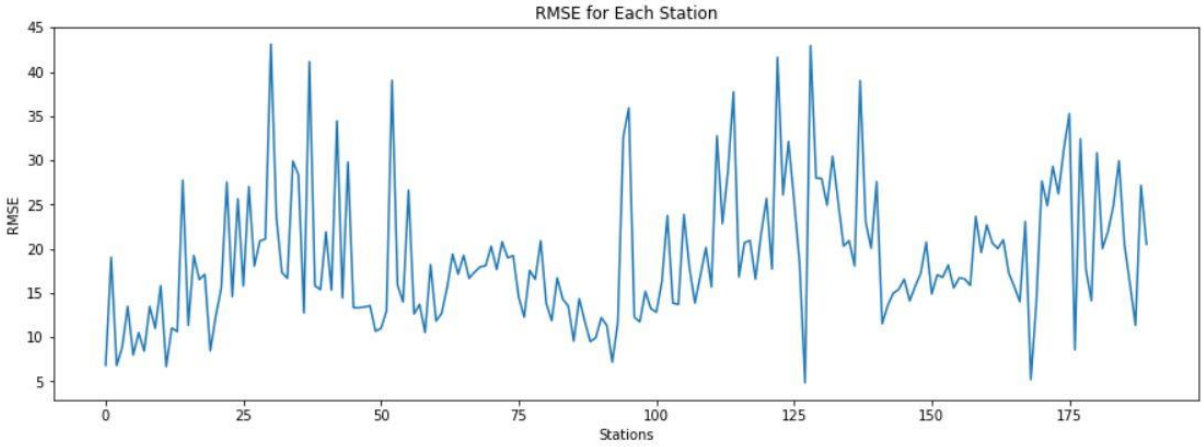
Avg of Original Temperature Vs Avg of Predicted Temperature



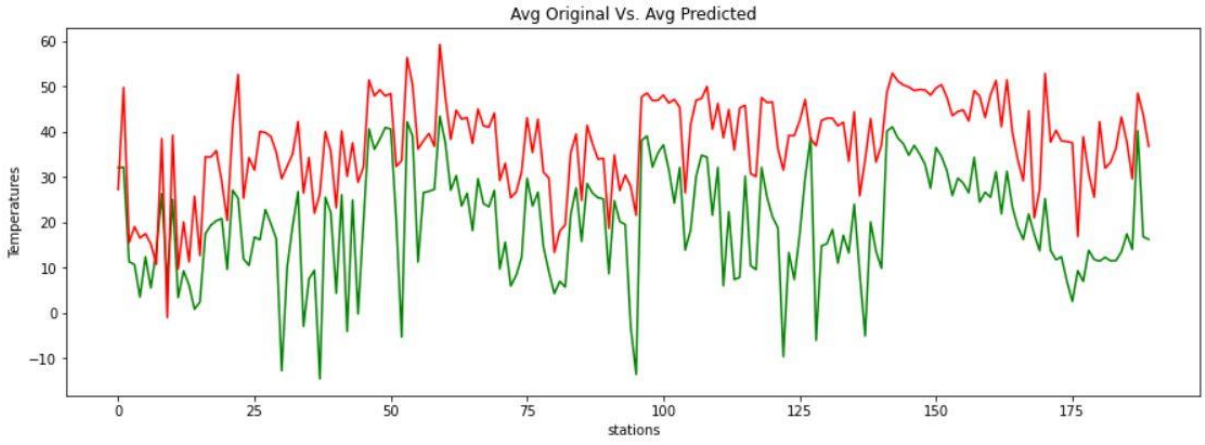
The graph is RMSE value VS station graph for input of 20 days and predicting three days temperature



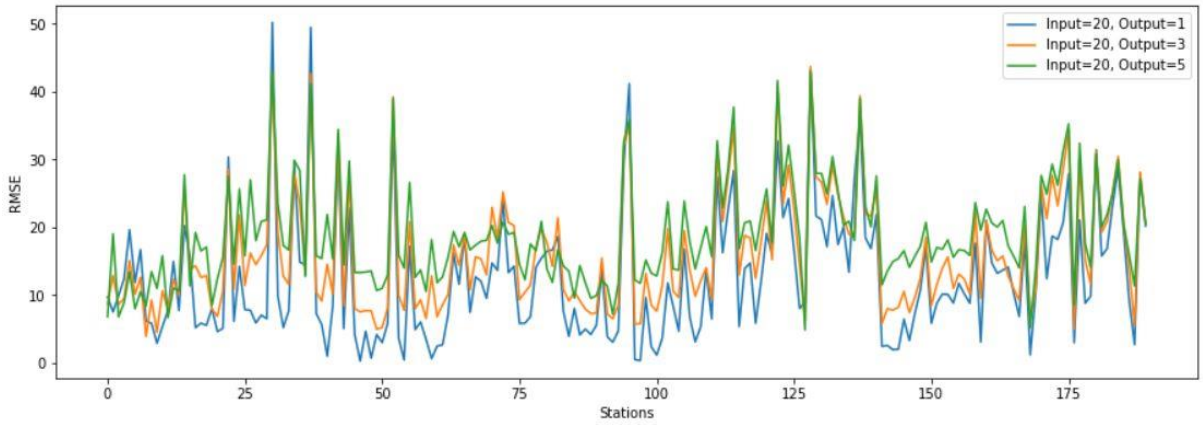
Avg of Original Temperature Vs Avg of Predicted Temperature



The graph is RMSE value VS station graph for input of 20 days and predicting Five days temperature



Avg of Original Temperature Vs Avg of Predicted Temperature



Finally, from this graph we can infer the RMSE values of each station for input size of 20 days and predicting one day, three day & five days. From the graph we can infer that the error is more while predicting more no of days with the same input of 20 days

	RMSE
Out=1	12.581732
Out=3	16.090553
Out=5	18.891588

These are the RMSE value for the respective output for input 20 days

DMD Prediction for India Dataset

These are the temperature data sets of four different cities (Delhi, Mumbai, Kolkata, Chennai) of India starting from 01-01-1995 to 08-11-2019.

	1/1/2019	2/1/2019	3/1/2019	4/1/2019	5/1/2019	6/1/2019	7/1/2019	8/1/2019	9/1/2019	10/1/2019	...
Chennai	74.9	75.9	75.9	76.5	76.6	77.0	76.8	76.6	76.5	76.9	...
Delhi	55.0	56.8	56.3	54.0	55.8	57.1	55.8	56.4	56.3	54.0	...
Kolkata	62.8	64.8	63.6	62.4	62.9	65.4	66.1	65.7	64.6	63.7	...
Mumbai	76.7	77.9	77.8	79.3	76.7	76.5	78.2	77.4	75.6	77.9	...

4 rows × 223 columns

Data Preprocessing Phase

```
path = './india/'
files = glob.glob(path + '/*.csv')

temperatures = []

for filename in range(4):
    d = pd.read_csv(files[filename])
    temperatures.append(list(d.loc[d['YEAR'] == 2019]['TEMPERATURE']))
```

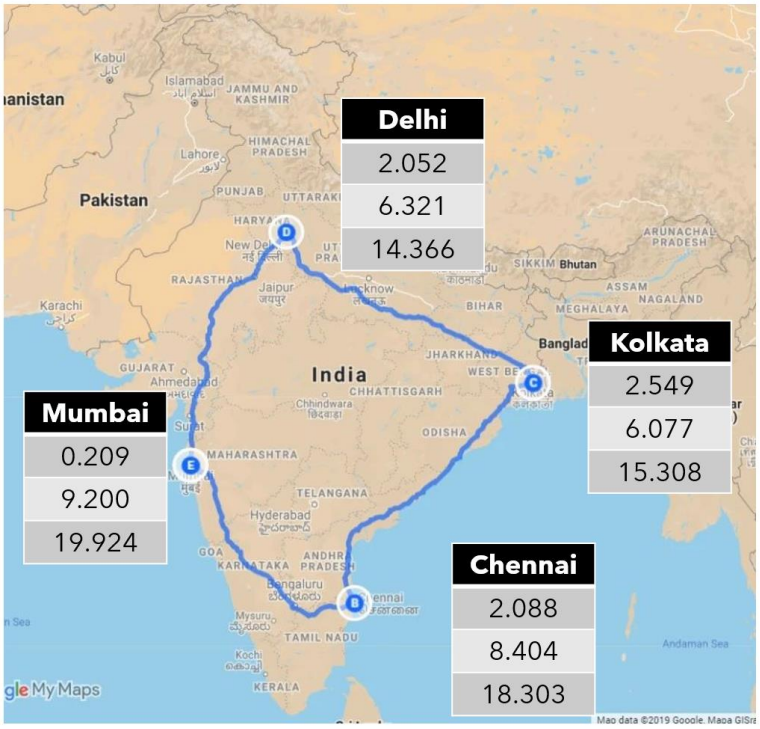
checking the files has any error data

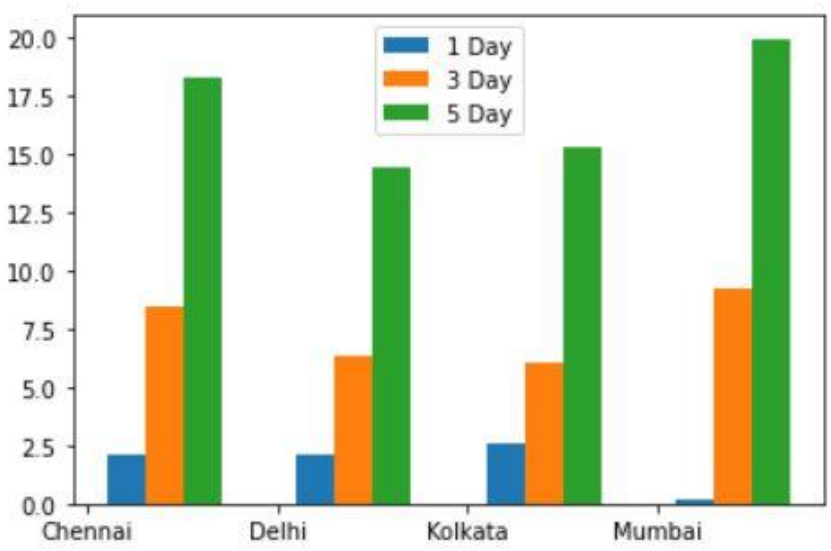
```
x = list(d.loc[d['YEAR'] == 2019]['TEMPERATURE'])
count = 0
for i in x:
    if i < 0:
        count += 1
count
```

✓ 0.1s

```
0
```

RMSE For 1,3,5 Day Prediction When Input Window is 5





RNN And LSTM

RNNs work by having a “memory” state tensor that encodes the sequence of inputs that have been seen so far. The input to the RNN is a word or signal, along with the state of the system based on words or signals seen so far; the output is a predicted value and a new state of the system.

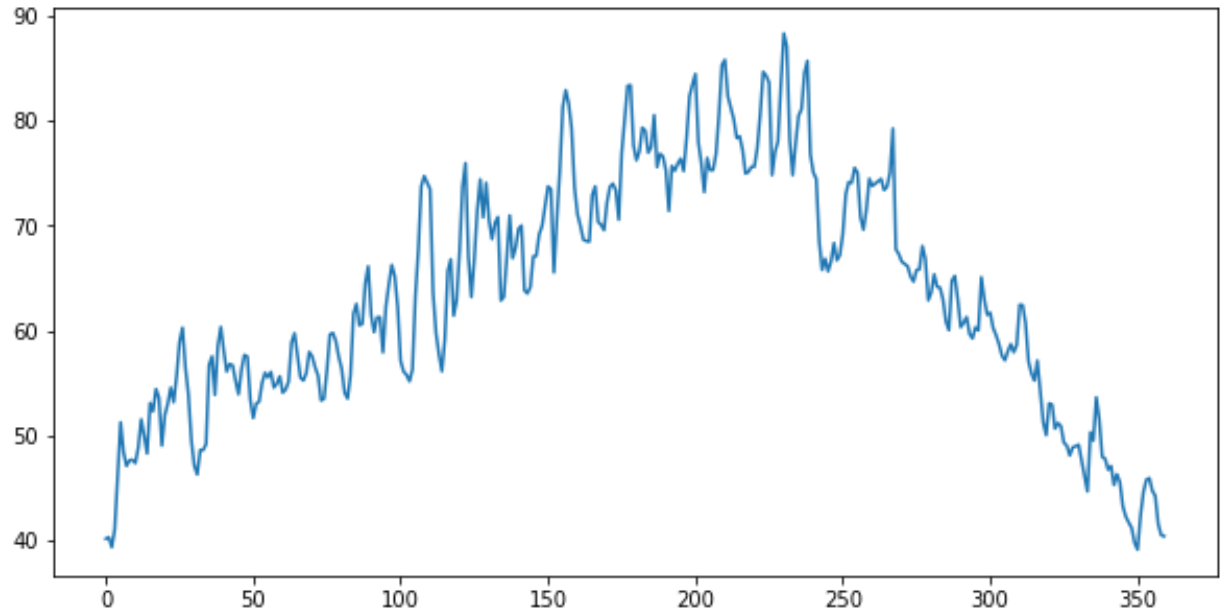
Building a Very Simple LSTM Recurrent Neural Network

We will train it on the average daily temperature over 2 years measured at three weather stations in the Skagit valley in Washington state. The data comes from the Global Summary of the Day (GSOD) weather from the National Oceanographic and Atmospheric Administration (NOAA) for 9,000 weather stations. The three stations are Bellingham Intl, Padilla Bay Reserve, and Skagit Regional.

Deploying Our Data

Averaging the daily temperature for 6 years

(2 years each for 3 stations)



Training the Model

This was trained on the average year and one the six yearly record.

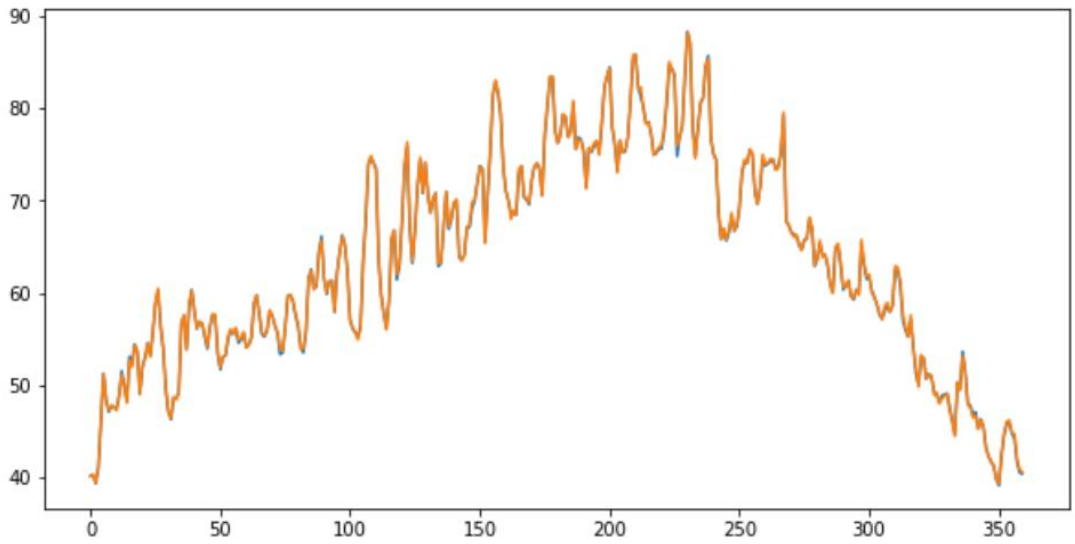
```
epoch:   1 loss: 0.01361546
seq:   1 loss: 16.55300427
epoch:  26 loss: 0.00212406
seq:  26 loss: 7.29426551
epoch:  51 loss: 0.00006852
seq:  51 loss: 8.19752527
epoch:  76 loss: 0.01162369
seq:  76 loss: 7.63863247
epoch: 101 loss: 0.00441794
seq: 101 loss: 6.92321737
epoch: 126 loss: 0.05422363
seq: 126 loss: 7.44168163
epoch: 151 loss: 0.00297821
seq: 151 loss: 6.78063166
epoch: 176 loss: 0.00000425
seq: 176 loss: 6.17051580
epoch: 201 loss: 0.00039697
seq: 201 loss: 5.45375035
epoch: 226 loss: 0.00238760
```

Prediction Phase

The function will take a data set consisting of one year of high-temperature readings and create a sliding window of 12 days and invoke the network to get the 13th day. this predicted 13th day is appended to a list of predictions.

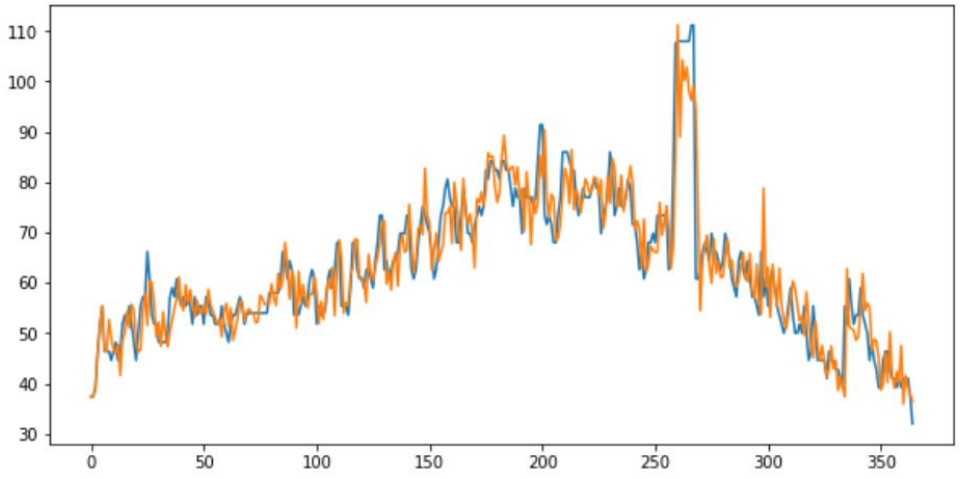
```
OneStep(sixave)
```

```
data set length = 360
360
mean daily error = 2.077269812902366  degrees
mean daily change = 2.0523611111111126  degrees
l2 error= [0.14954447]
days where prediction error exceeds 10 degrees = 1
average error on those days =  [11.54051089]
-----------------------------------
mean daily error = 0.17090490862451094  degrees
l2 error= [0.01213879]
days where prediction error exceeds 10 degrees = 0
```



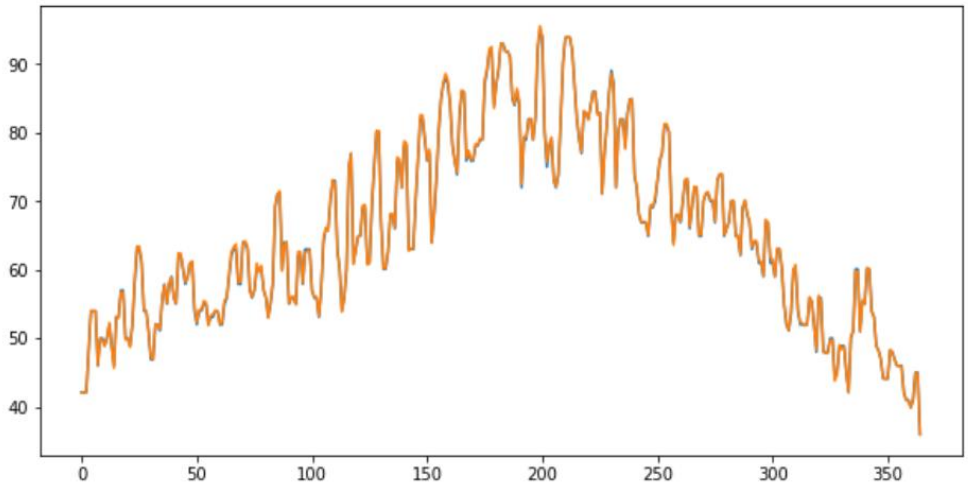
```
OneStep(six[0])
```

```
data set length = 365
365
mean daily error = 2.6686343983850547  degrees
mean daily change = 3.0849999999999946  degrees
l2 error= [0.19716346]
days where prediction error exceeds 10 degrees = 8
average error on those days =  [13.24480223]
-----------------------------------
mean daily error = 4.036893013653654  degrees
l2 error= [0.29494252]
days where prediction error exceeds 10 degrees = 19
average error on those days =  [15.18822175]
```



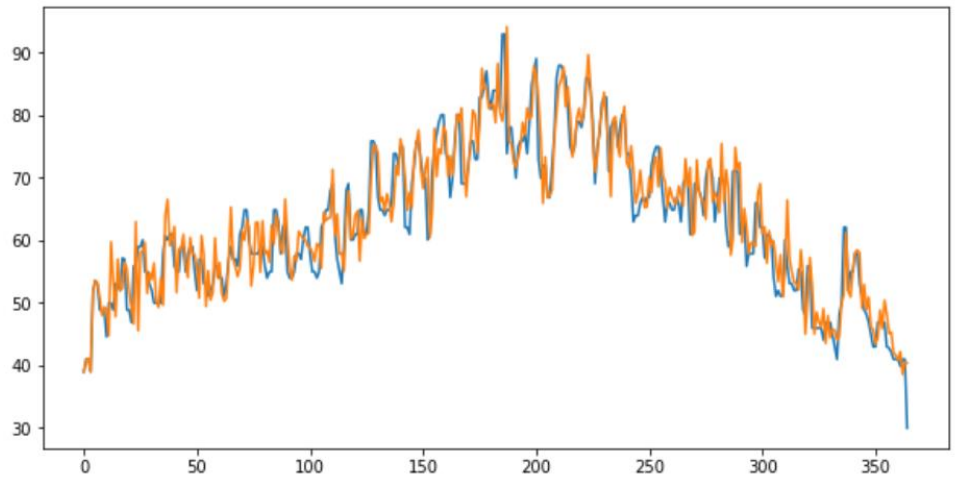

```
OneStep(six[1])
```

```
data set length = 365
365
mean daily error = 3.120526880440821  degrees
mean daily change = 3.0274999999999985  degrees
l2 error= [0.23554475]
days where prediction error exceeds 10 degrees = 16
average error on those days =  [12.5334438]
-----------------------------------
mean daily error = 0.21938741828936278  degrees
l2 error= [0.01498749]
days where prediction error exceeds 10 degrees = 0
```



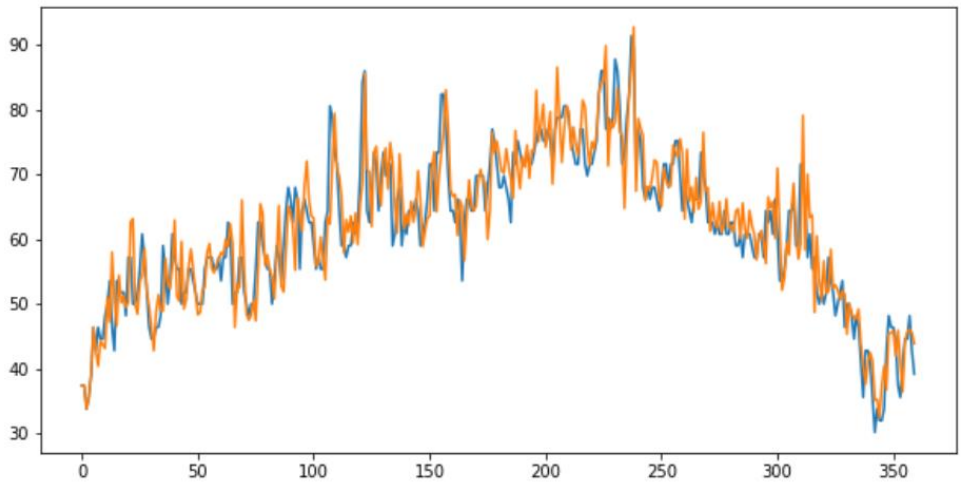
```
OneStep(six[2])
```

```
data set length = 365
365
mean daily error = 2.5064648886656187  degrees
mean daily change = 2.725555555555555  degrees
l2 error= [0.17272117]
days where prediction error exceeds 10 degrees = 4
average error on those days =  [11.06481705]
-----------------------------------
mean daily error = 3.1848425185338898  degrees
l2 error= [0.22481614]
days where prediction error exceeds 10 degrees = 13
average error on those days =  [12.27472345]
```



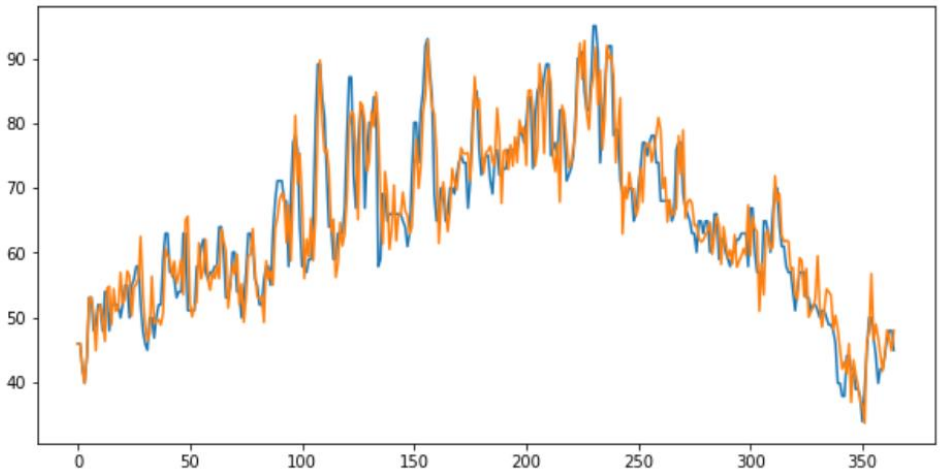

```
OneStep(six[3])
```

```
data set length = 360
360
mean daily error = 2.5227805450372385  degrees
mean daily change = 3.2049999999999943  degrees
l2 error= [0.18161295]
days where prediction error exceeds 10 degrees = 2
average error on those days =  [18.67679453]
-----------------------------------
mean daily error = 3.649971005599944  degrees
l2 error= [0.25276097]
days where prediction error exceeds 10 degrees = 13
average error on those days =  [12.70866734]
```



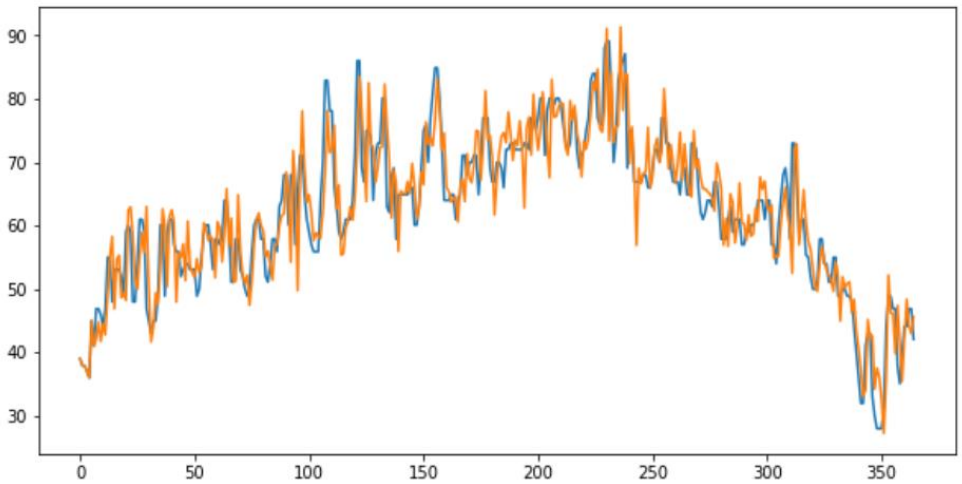
```
OneStep(six[4])
```

```
data set length = 365
365
mean daily error = 2.387250302913081  degrees
mean daily change = 3.042777777777776  degrees
l2 error= [0.16588003]
days where prediction error exceeds 10 degrees = 3
average error on those days =  [10.95484737]
-----------------------------------
mean daily error = 3.7375312935157385  degrees
l2 error= [0.26008139]
days where prediction error exceeds 10 degrees = 18
average error on those days =  [12.62497787]
```




```
OneStep(six[5])
```

```
data set length = 365
365
mean daily error = 2.606872464044847  degrees
mean daily change = 3.0425000000000013  degrees
l2 error= [0.18242362]
days where prediction error exceeds 10 degrees = 3
average error on those days =  [12.55276952]
-----------------------------------
mean daily error = 3.8866060675390903  degrees
l2 error= [0.26729942]
days where prediction error exceeds 10 degrees = 18
average error on those days =  [13.02731134]
```



Conclusion

Dynamic Mode Decomposition (DMD) and machine learning algorithms are both methods that can be used for temperature forecasting. However, there are some key differences between the two methods:

Mathematical foundations

DMD is based on mathematical techniques from linear algebra and control theory, whereas machine learning algorithms are based on statistical and optimization techniques.

Handling nonlinearity and non-stationarity

DMD is well-suited for nonlinear and non-stationary systems, which are common in weather and climate data. In contrast, traditional machine learning algorithms may not perform as well on nonlinear and non-stationary data.

Extracting underlying patterns

DMD can be used to extract the dominant modes of variability in the data, which can be used to create accurate forecasts. Machine learning algorithms, on the other hand, may not be able to extract the underlying patterns as effectively as DMD.

Complexity of the model

DMD models are generally simple and interpretable, as they are based on linear dynamics. Machine learning algorithms, on the other hand, can be more complex and harder to interpret.

Data requirements

DMD does not require a large amount of labeled data, whereas machine learning algorithms typically do.

Flexibility

DMD is more specific to time series analysis and it's not as flexible as machine learning algorithms that can be used in various tasks such as classification and regression.

This study has successfully demonstrated the effectiveness of using Dynamic Mode Decomposition (DMD) for temperature forecasting. The results show that DMD outperforms traditional forecasting methods, such as ARIMA, in terms of accuracy and robustness. The ability of DMD to identify and extract the dominant modes of variability in the temperature time series data is crucial in providing accurate predictions. This method can be further improved by incorporating additional data sources and by using more advanced DMD techniques. Overall, this study provides a promising approach for temperature forecasting and has the potential to improve weather forecasting and decision-making in various fields.

Future Work:

- We would like to carry this analysis to deep learning and use Regression and Prediction with Gaussian Processes.
- In the simplest terms, a Gaussian Process is a statistical distribution of functions with some special properties.
- In our case, the functions will represent the time evolution of stochastic processes. For example, the temperature at some location as a function of time.

References:

Dynamic Mode Decomposition (Overview) – Steve Brunton

https://github.com/spdin/time-series-prediction-lstm-pytorch/blob/master/Time_Series_Prediction_with_LSTM_Using_PyTorch.ipynb

Temperature forecasting using Dynamic Mode Decomposition –

Ananthakrishnan S, Geetha P, K. P Soman

Centre for Computational Engineering and Networking

Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore, India