Prediction of Mean Stratospheric Temperature Over Equator Using LSTM Based Recurrent Neural Network

By

Rajnish Maurya (CSE 18067/378)

Subham Pal (CSE 18093/404)

Subrata Kumar Biswas (CSE 18094/405)

Bachelor Thesis submitted to

Indian Institute of Information Technology Kalyani

for the partial fulfillment of the degree of

Bachelor of Technology

in

Computer Science and Engineering

May, 2022

Certificate

This is to certify that the thesis entitled “Prediction of Mean Stratospheric Temperature Over Equator Using LSTM Based Recurrent Neural Network” being submitted by **Rajnish Maurya (CSE 18067/378)**, **Subham Pal (CSE 18093/404) and Subrata Kumar Biswas (CSE 18094/405)**, undergraduate students, in the Department of Computer Science and Engineering, Indian Institute of Information Technology Kalyani, West Bengal 741235, India, for the award of Bachelors of Technology in Computer Science Engineering is an original research work carried by them under my supervision and guidance. The thesis has fulfilled all the requirements as per the regulation of Indian Institute of Information Technology Kalyani and in my opinion, has reached the standards needed for submission. The works, techniques and the results presented have not been submitted to any other university or Institute for the award of any other degree or diploma.

Name of the Guide: Dr. Uma Das

Position: Assistant Professor (Physics)

Departmental address: IIIT Kalyani, Webel IT Park, Kalyani – 741235 Nadia, West Bengal

Declaration

I hereby declare that the work which being presented in the thesis entitled "Prediction of Mean Stratospheric Temperature Over Equator Using LSTM Based Recurrent Neural Network" is submitted to Indian Institute of Information Technology Kalyani in partial fulfillment for the award of the degree of **Bachelor of Technology** in Computer Science and Engineering does not contain any classified information during the period from **Jan, 2022 to May, 2022** under the supervision of Dr. Uma Das, Indian Institute of Information Technology Kalyani, West Bengal 741235, India.

Name of the Candidates: Rajnish Maurya, Subham Pal & Subrata Kumar Biswas

Reg No/Roll No: CSE 18067/378, CSE 18093/404 & CSE 18094/405 respectively.

Name of the Department: Computer Science Engineering

Institute Name: IIIT Kalyani

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

....

Dr. Uma Das

Assistant Professor (Physics)

IIIT Kalyani, Kalyani – 741235 Nadia, West Bengal

Place:

Date:

Acknowledgments

First of all, I would like to take this opportunity to thank my supervisor Dr. Uma Das for her unwavering support right from selecting the topic to writing the final thesis for this project. She gave us the opportunity to work in a project that we were looking forward to doing and then giving us the creative freedom to design and work on it how we wished to but simultaneously giving us her constructive criticism and advice on what problems we might face along the way specially her advice on how to plan out a timeline or what we could work on really helped us complete this project in a timely manner. This project would not be possible without her constant support throughout. Last but not the least, she provided us with the dataset without which this project would not have been possible.

Name of the Candidates: Rajnish Maurya, Subham Pal & Subrata Kumar Biswas

Reg No/Roll No: CSE 18067/378, CSE 18093/404 & CSE 18094/405 respectively.

Name of the Department: Computer Science Engineering

Institute Name: IIIT Kalyani

Place:

Date:

Table of Contents

Abstract

Prediction of Temperature is one of the major concerns in the domain of meteorology and atmospheric sciences, especially when it comes to global warming. Several techniques have been formerly proposed to predict temperature based on statistical analysis, models, machine learning and deep learning techniques. Prediction of time series data in meteorology can assist in decision-making processes carried out by organizations responsible for the prevention of disasters. This work presents LSTM based Recurrent Neural Network (RNN) to predict the temperatures in the stratosphere at various pressure levels / altitude. The neural network is trained and tested using ERA5 reanalysis dataset containing monthly mean zonal temperatures for a period of 51 years (1979-2019). The trained network (from 1979-2017) will predict stratospheric temperatures for the next 2 years (2018-2019).

Chapter 1: Introduction

The Atmosphere

The earth is enveloped by a layer of mixture of different gases called atmosphere. The atmosphere is composed of different layers and in each layer there is a different composition. Thus resulting in different phenomena going on in different layers. The layers are as follows:

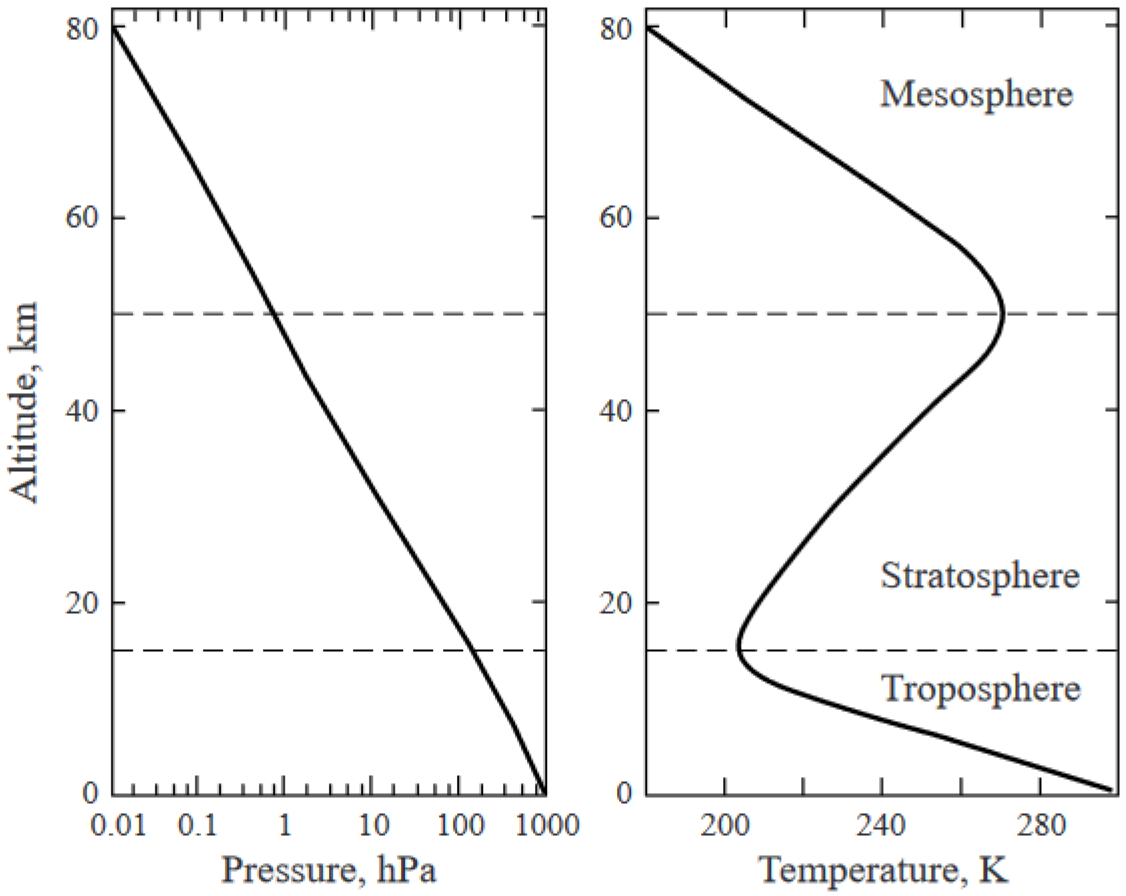


Figure 1.1.
Mean pressure and temperature vs. altitude
[Credits: https://projects.iq.harvard.edu/files/acmg/files/intro_atmo_chem_bookchap2.pdf]

Troposphere

It is the layer closest to the surface of earth, It stretches till 10 KM above the sea level. This is where we humans live and here most of our weather occurs such as rain, clouds, and snow. This is because the troposphere contains 99 percent of water vapours of the atmosphere. As we move higher in the troposphere, the temperature decreases by around 6.5°C every Kilometer. The height of the troposphere is different at the pole and the equator. At the equator, the troposphere is higher and in the poles, the troposphere layer is thinner. The lower part of the troposphere (boundary layer) is motioned by the properties of earth's surface.

Stratosphere

This layer stretches from tropopause to about 50 KM above the sea level. The layer has a significant ozone layer within it which absorbs the harmful ultraviolet rays from the sun. This results in the increase in temperature in the stratosphere. This layer holds around 19 percent of the total gases of the atmosphere but very little water vapours. As we go higher, the temperature increases due to the ozone layer.

The increase in temperature with the height means the warmer air is present above the cooler air. This prevents the 'convection' as there is no upward movement of the gases. This results in less turbulence than the troposphere and for that reason, airplanes fly in this layer right above the troposphere. Also, the jet stream flows in the border of the stratosphere and the troposphere.

Mesosphere

It extends till 85 KM above the sea level. Most of the meteors burn up in this layer because the air is very denser here. Again temperature decreases as we move upwards in this layer.

Thermosphere and Ionosphere

The thermosphere lies above the mesopause and again the temperature starts rising from here. This is because of the absorption of powerful UV rays and X-rays from the sun directly. There is an extreme rise in the temperature as we go upwards.

About 80Km above the surface of the earth in the thermosphere, there lies the Ionosphere. This layer is formed by the high energetic solar radiations knocking-off electrons from molecules and atoms. Thus ions are present in this layer with positive charge. The temperature varies in this region with day and night.

Importance of Temperatures in Stratosphere

The stratosphere exhibits a diverse range of variability on a spectrum of timescales with, in many cases, a well-established influence on the tropospheric circulation below. As a result, knowledge of the state of the stratosphere has the potential to enhance the predictability of the troposphere on sub-seasonal to seasonal (S2S) timescales and beyond. (*Butler, Amy, 2019*)

In the tropics, the dominant feature of stratospheric variability is a remarkably regular succession of downward-migrating easterly and westerly zonal jets known as the Quasi-biennial Oscillation (QBO). Over a period of roughly 28 months, the equatorial winds transition between westerly QBO (WQBO) and easterly QBO (EQBO) as a result of the selective absorption of tropical waves propagating upward from the troposphere below (Lindzen and Holton, 1968; Holton and Lindzen, 1972; Baldwin et al., 2001).

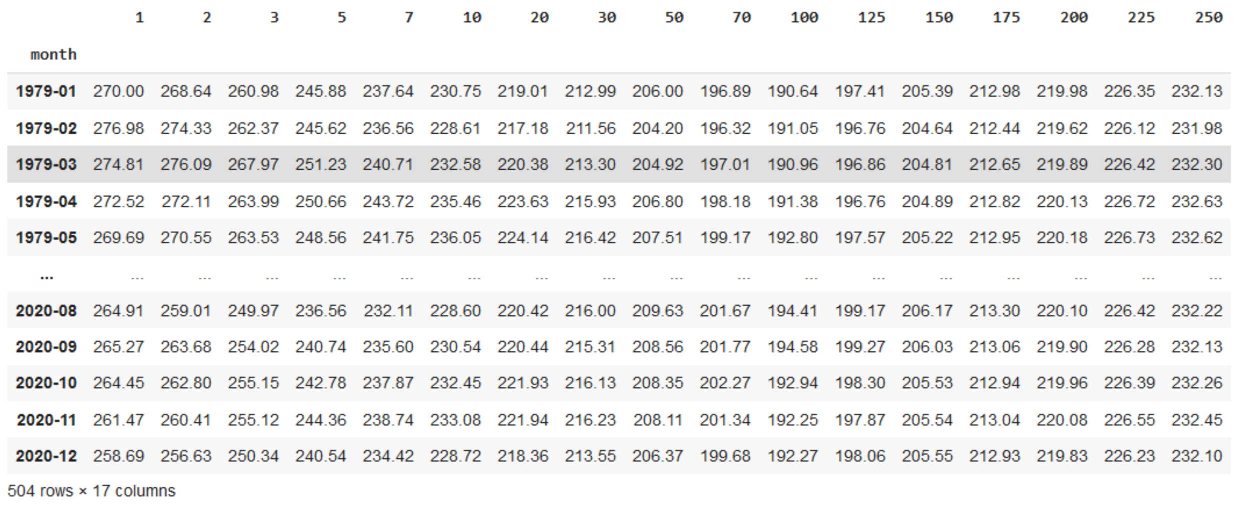
The QBO can affect the characteristics of tropical deep convection (e.g., Collimore et al., 2003; Liess and Geller, 2012). Satellite observations and numerical model simulations indicate that tropical deep convection across the western Pacific is stronger during EQBO winters than during WQBO winters

(*Collimore et al., 2003*). Additionally, sub-seasonal Madden Julian Oscillation (MJO)-like convective activity is significantly modulated by the QBO, with stronger and more organized MJO convection during 50-hPa EQBO winters (Liu et al., 2014b; Yoo and Son, 2016; Son et al., 2017; Nishimoto and Yoden, 2017)

Problem Statement

Make a model to forecast Temperature given in Kelvin (K) for TWO more years after 2018, i.e 2019 & 2020 for various pressure levels (in Hectopascals). Data is present from 1979-2020. After prediction we can tally that with the existing data. Prediction is based on Recurrent Neural Network based LSTM.

We have collected the data containing the mean Temperature on the Equator in Kelvin for various Pressure regions from *Hersbach* et al. Individual records for 504 monthly average i.e 42 years starting from 01-1979 to 12-2020 in kilokelvin (kK).



	1	2	3	5	7	10	20	30	50	70	100	125	150	175	200	225	250
month																	
1979-01	270.00	268.64	260.98	245.88	237.64	230.75	219.01	212.99	206.00	196.89	190.64	197.41	205.39	212.98	219.98	226.35	232.13
1979-02	276.98	274.33	262.37	245.62	236.56	228.61	217.18	211.56	204.20	196.32	191.05	196.76	204.64	212.44	219.62	226.12	231.98
1979-03	274.81	276.09	267.97	251.23	240.71	232.58	220.38	213.30	204.92	197.01	190.96	196.86	204.81	212.65	219.89	226.42	232.30
1979-04	272.52	272.11	263.99	250.66	243.72	235.46	223.63	215.93	206.80	198.18	191.38	196.76	204.89	212.82	220.13	226.72	232.63
1979-05	269.69	270.55	263.53	248.56	241.75	236.05	224.14	216.42	207.51	199.17	192.80	197.57	205.22	212.95	220.18	226.73	232.62
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
2020-08	264.91	259.01	249.97	236.56	232.11	228.60	220.42	216.00	209.63	201.67	194.41	199.17	206.17	213.30	220.10	226.42	232.22
2020-09	265.27	263.68	254.02	240.74	235.60	230.54	220.44	215.31	208.56	201.77	194.58	199.27	206.03	213.06	219.90	226.28	232.13
2020-10	264.45	262.80	255.15	242.78	237.87	232.45	221.93	216.13	208.35	202.27	192.94	198.30	205.53	212.94	219.96	226.39	232.26
2020-11	261.47	260.41	255.12	244.36	238.74	233.08	221.94	216.23	208.11	201.34	192.25	197.87	205.54	213.04	220.08	226.55	232.45
2020-12	258.69	256.63	250.34	240.54	234.42	228.72	218.36	213.55	206.37	199.68	192.27	198.06	205.55	212.93	219.83	226.23	232.10

504 rows × 17 columns

Figure 1.2.

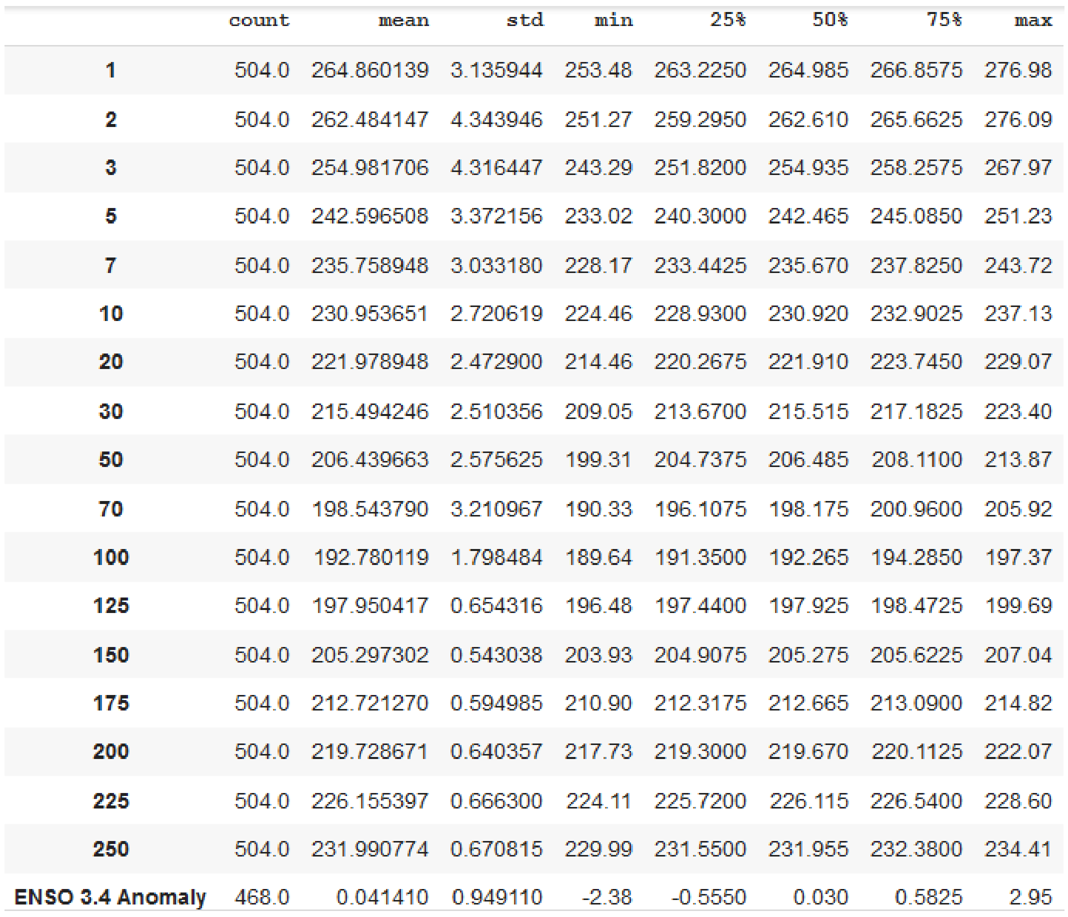
Data Representation from 1979 to 2020.

Chapter 2: Data and Methods

Description of Data

The data contains the monthly mean of the temperature over various pressure levels from *Hersbach* et al [1]. It is the zonal (averaged over all longitudes) mean of monthly mean temperature over the equator from ERA5, which provides hourly estimates for a large number of atmospheric, ocean-wave and land-surface quantities. An uncertainty estimate is sampled by an underlying 10-member ensemble at three-hourly intervals. Ensemble mean and spread have been pre-computed for convenience. Such uncertainty estimates are closely related to the information content of the available observing system which has evolved considerably over time. They also indicate flow-dependent sensitive areas. To facilitate many climate applications, monthly-mean averages have been pre-calculated too, though monthly means are not available for the ensemble mean and spread. This information about ERA5 is cited from *Hersbach* et al [1].

Data received contains 504 Monthly Means of Temperature (in Kelvin) over the Equator data for various Pressure levels (in Hectopascal). The Excel Sheet contains two dimensional arrays of data - 504 x 18. For each Pressure level there is one column, the last column denotes ENSO 3.4 Anomaly.



	count	mean	std	min	25%	50%	75%	max
1	504.0	264.860139	3.135944	253.48	263.2250	264.985	266.8575	276.98
2	504.0	262.484147	4.343946	251.27	259.2950	262.610	265.6625	276.09
3	504.0	254.981706	4.316447	243.29	251.8200	254.935	258.2575	267.97
5	504.0	242.596508	3.372156	233.02	240.3000	242.465	245.0850	251.23
7	504.0	235.758948	3.033180	228.17	233.4425	235.670	237.8250	243.72
10	504.0	230.953651	2.720619	224.46	228.9300	230.920	232.9025	237.13
20	504.0	221.978948	2.472900	214.46	220.2675	221.910	223.7450	229.07
30	504.0	215.494246	2.510356	209.05	213.6700	215.515	217.1825	223.40
50	504.0	206.439663	2.575625	199.31	204.7375	206.485	208.1100	213.87
70	504.0	198.543790	3.210967	190.33	196.1075	198.175	200.9600	205.92
100	504.0	192.780119	1.798484	189.64	191.3500	192.265	194.2850	197.37
125	504.0	197.950417	0.654316	196.48	197.4400	197.925	198.4725	199.69
150	504.0	205.297302	0.543038	203.93	204.9075	205.275	205.6225	207.04
175	504.0	212.721270	0.594985	210.90	212.3175	212.665	213.0900	214.82
200	504.0	219.728671	0.640357	217.73	219.3000	219.670	220.1125	222.07
225	504.0	226.155397	0.666300	224.11	225.7200	226.115	226.5400	228.60
250	504.0	231.990774	0.670815	229.99	231.5500	231.955	232.3800	234.41
ENSO 3.4 Anomaly	468.0	0.041410	0.949110	-2.38	-0.5550	0.030	0.5825	2.95

Table 2.1.

Column Wise Description of Data, all are representing temperature in Kelvin except ENSO 3.4 Anomaly. (Where **Count** means No. of occurrences, **Mean** is simple AM and **std** denotes the Standard Deviation)

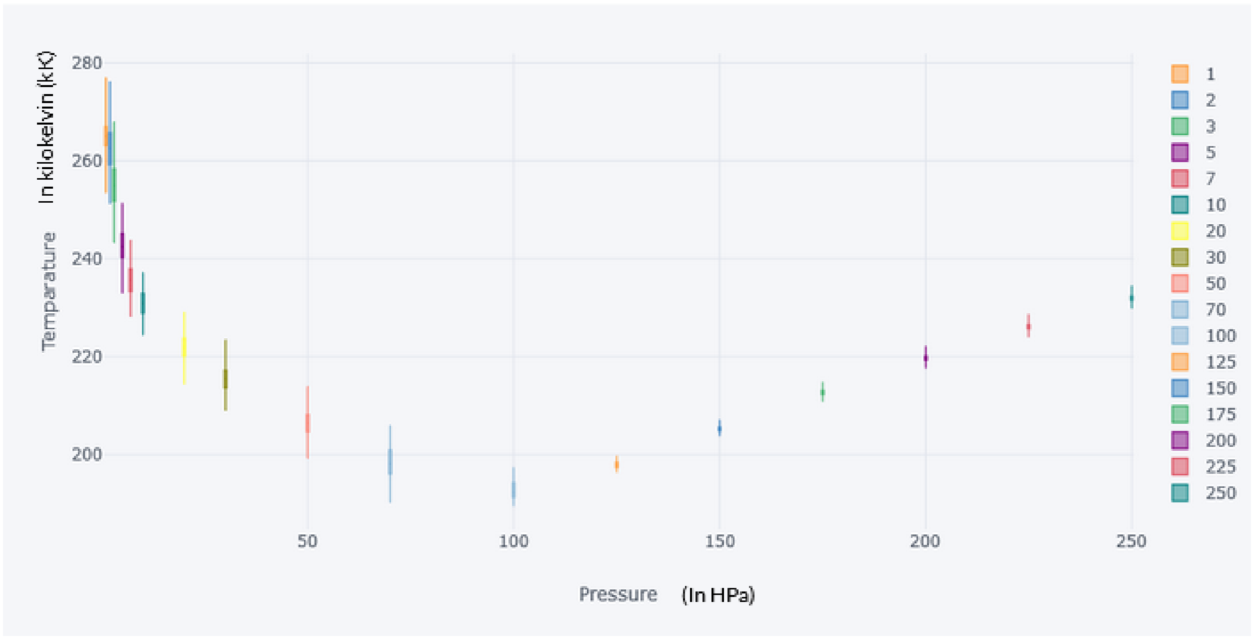


Figure 2.2.

The box plots denote the distribution of the data visually as represented in Table 2.1, from where we can observe the mean, standard deviation, etc.

Train-Test Splitting of Data

Ideally in any Data Science model involving some Machine Learning or Deep Learning methodology for statistical prediction, we divide the data in **Train**, **Test** and **Validation** sets. The Train set is used to prepare/ **train** the model, then we test that model by providing data from the **Validation** set to know whether our model is Overfitting or Underfitting. After we finalize our model then we make predictions and test it over our **Test** dataset.

Purpose	**Split Range**	**Percentage Allocated (Approx.)**
Train	1979-01 to 2016-05	90% (36 years)
Validation	2016-06 to 2018-12	5% (2 years)
Test	2019-01 to 2020-12	5% (2 years)

Table 2.3

Train, Validation and Test Split.

From Table 2.3, we can see the distribution of the entire data which has been used for various purposes.

Preprocessing of Data

The data we extract from various sources have a lot of missing values and unnecessary information that can affect the performance of our model. Therefore, preprocessing is an integral part of any ML/DL project. It includes handling or deleting null values, normalizing/ scaling data for faster processing, manipulating categorical variables, etc.

Data Scaling

Scaling refers to transforming the data into a specific range, say [0-1] or [0-100]. Scaling of data improves the performance of many Machine Learning and Deep Learning algorithms including the RNN architecture. It is different from normalization as normalization changes the shape of the distribution of data points. Normalization converts the data into normal distribution. Scaling is preferred in our problem because we do not want to alter the distribution of the data.

We used Min-Max Scaler to scale the data. It scales all the data features in the range [0, 1] or else we can provide custom range. This scaling compresses all the inliers in the narrow range [0, 0.005]. For each value in a feature, MinMaxScaler subtracts the minimum value in the feature and then divides by the range. The range is the difference between the original maximum and original minimum. MinMaxScaler preserves the shape of the original distribution. The Min-Max Scaler outperforms the Standard Scaler when there is a large dataset with outliers value.

The algorithm behind MinMaxScaler is,
X_std = (X - X.min(axis=0)) / (X.max(axis=0) - X.min(axis=0))
X_scaled = X_std * (max - min) + min
where min, max = feature_range, eg. [0,1] or [0,100]

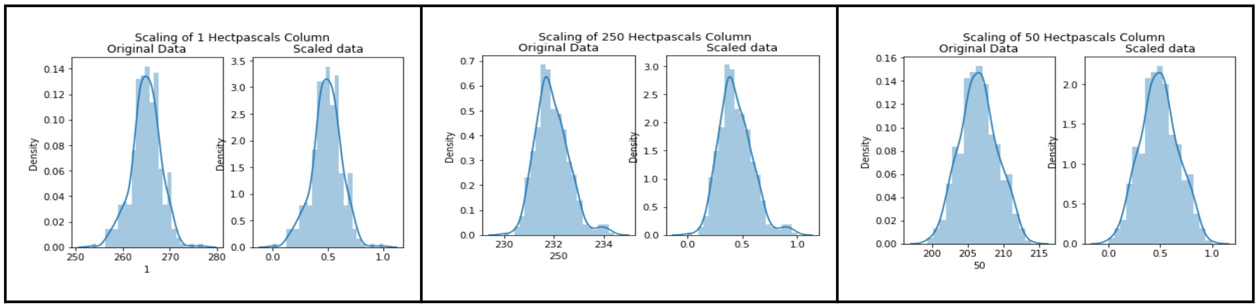


Figure 2.4
Illustration graphs before (Left) & after (Right) Scaling to show how scaling works.

Now we have applied the Min-Maxscaler on Training set and the reason behind that is to avoid dependency on the values. If we apply Min-Maxscaler on the training set, and fit that model on our test and validation set then it becomes independent of the mean of its own value. Now, it can be seen that the range of the values has now been reduced within 0 and 1. But, we can see that the distribution of the values remains the same, only the range has changed.

Use of Deep Neural Networks: Sequence Modeling Problems

Sequence Modeling problems refer to the problems where either the input and/or the output is a sequence of data. Sequence problems can be broadly classified into the following categories:

- One-to-one
- One-to-many
- Many-to-one
- Many-to-many

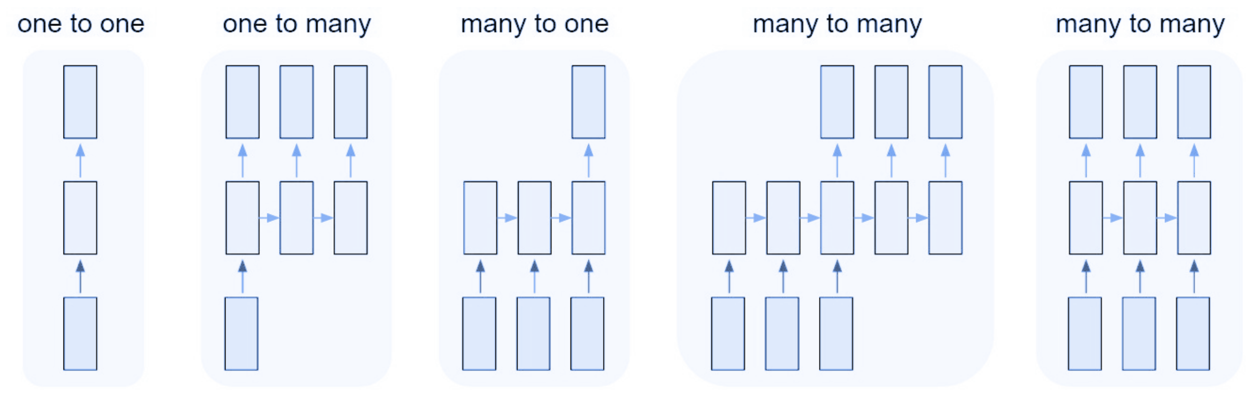


Figure 2.5
Sequence Modeling problems.

In Figure 2.5, each rectangle is a vector and arrows represent the functions.Input vectors are in red , output vectors are in blue and green vectors hold the RNN's state.
From left to right:
(1) Vanilla mode of processing without RNN, from fixed-sized input to fixed-sized output.
(2) Sequence output(e.g. Image captioning takes an image and outputs a sentence of words).
(3) Sequence input (e.g. sentiment analysis where a given sentence is classified as expressing positive or negative sentiment).
(4) Sequence input and sequence output.
(5) Synced sequence input and output (e.g. video classification where we wish to label each frame of the video).

Encoder-Decoder LSTM

Sequence-to-Sequence (Seq2Seq) problems is a special class of Sequence Modeling Problems in which both, the input and the output is a sequence. Encoder-Decoder models were built to solve such Seq2Seq problems.
Encoder-Decoder models are useful for the following applications:

- Machine translation
- Speech recognition
- Video captioning
- Text summarization

We have picked up the **Encoder-Decoder LSTM model** for multi-step forecasting with multivariate input data because the architecture of this model supports multivariate input and output at the same time. An encoder-decoder LSTM is a model comprised of two sub-models: one called the encoder that reads the input sequences and compresses it to a fixed-length internal representation, and an output model called the decoder that interprets the internal representation and uses it to predict the output sequence.

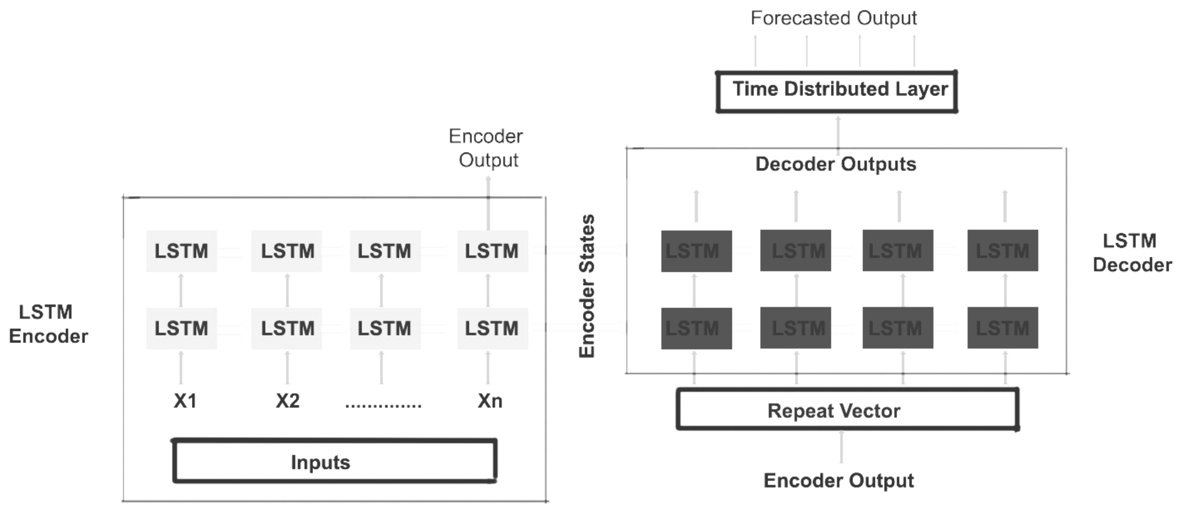


Figure 2.6

Encoder Decoder LSTM Architecture.

According to Brownlee 2020, the encoder-decoder approach to sequence prediction has proven much more effective than outputting a vector directly and is the preferred approach. Using multivariate inputs is helpful for those problems where the output sequence is some function of the observations at prior time steps from multiple different features, not just (or including) the feature being forecasted. Which in our case, it is true, our data is dependent on the previous values.

The Architecture of Encoder-Decoder Model

An encoder-decoder can be thought of as two blocks, the encoder and the decoder connected by a vector which we will refer to as the ‘context vector’.

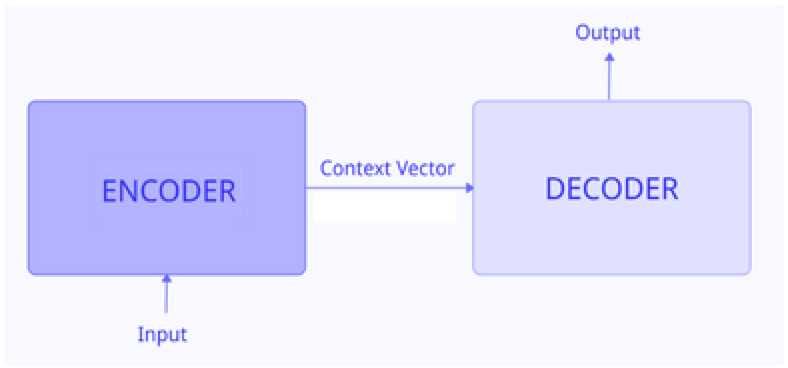


Figure 2.7

Encoder Decoder LSTM Architecture (High Level Overview).

Encoder

The encoder processes each token in the input-sequence. It tries to cram all the information about the input-sequence into a vector of fixed length i.e. the ‘context vector’. After going through all the tokens, the encoder passes this vector onto the decoder.

Context Vector
The vector is built in such a way that it's expected to encapsulate the whole meaning of the input-sequence and help the decoder make accurate predictions.

Decoder
The decoder reads the context vector and tries to predict the target-sequence token by token.

The Encoder Block
The encoder part is an LSTM cell. It is fed in the input-sequence over time and it tries to encapsulate all its information and store it in its final internal states h (hidden state) and c (cell state). The internal states are then passed onto the decoder part, which it will use to try to produce the target-sequence.

The Decoder Block
So after reading the whole input-sequence, the encoder passes the internal states to the decoder and this is where the prediction of output-sequence begins.
The decoder block is also an LSTM cell. The main thing here is that the initial states (h_0, c_0) of the decoder are set to the final states (h , c) of the encoder. These act as the 'context' vector and help the decoder produce the desired target-sequence.

Now, we have created 3 architectures intuitively and tried to observe the results. Out of 3 we picked the best model and did some parameter tuning based on the validation set results. The detailed architecture of the 3 models are given below.

Model Architecture - E1D1

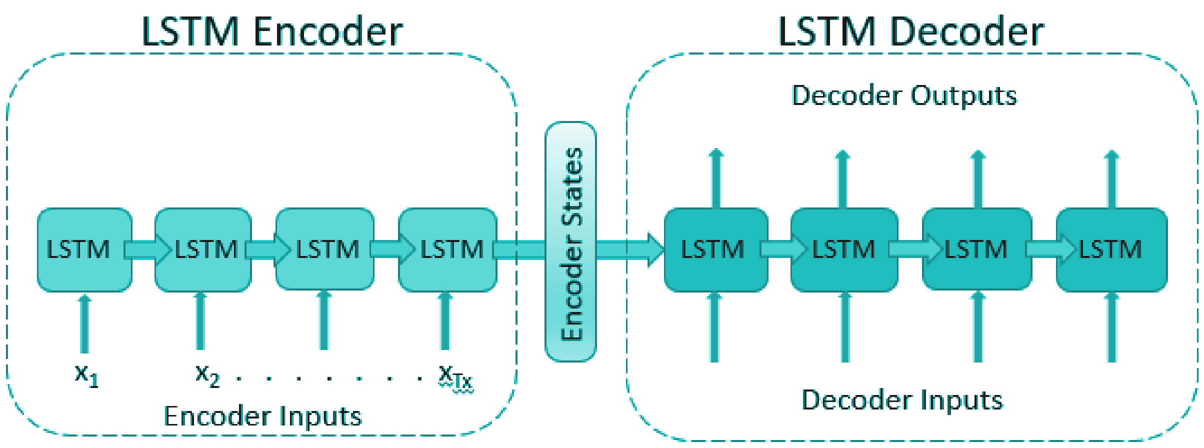


Figure 2.8
Encoder Decoder LSTM Architecture with one encoder and one decoder.

```
Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
input_7 (InputLayer)           [(None, 10, 18)]     0           []

lstm_28 (LSTM)                 [(None, 100),        47600       ['input_7[0][0]']
                                (None, 100),
                                (None, 100)]

repeat_vector_6 (RepeatVector)  (None, 5, 100)      0           ['lstm_28[0][0]']

lstm_29 (LSTM)                 (None, 5, 100)       80400       ['repeat_vector_6[0][0]',
                                                                 'lstm_28[0][1]',
                                                                 'lstm_28[0][2]']

dropout_4 (Dropout)            (None, 5, 100)       0           ['lstm_29[0][0]']

time_distributed_6 (TimeDistri  (None, 5, 18)       1818        ['dropout_4[0][0]']
buted)

==================================================================================================
Total params: 129,818
Trainable params: 129,818
Non-trainable params: 0
```

Table 2.9
Model summary of E1D1 LSTM.

E1D1 comprises one layer of LSTM for encoder and one layer for decoder.LSTM takes only one element at a time, so if the input sequence is of length n, then LSTM takes n time steps to read the entire sequence. Here , Xt the input at time step t and each Xt represented as a vector of fixed length. In sequence modeling problems , we generate the outputs when we read the entire input sequence so output of each LSTM in a encoder is of no use so we discard it. The first state of the decoder is set to the final states of the decoder.LSTM in the decoder process single word at every time step.

Model Architecture - E2D2

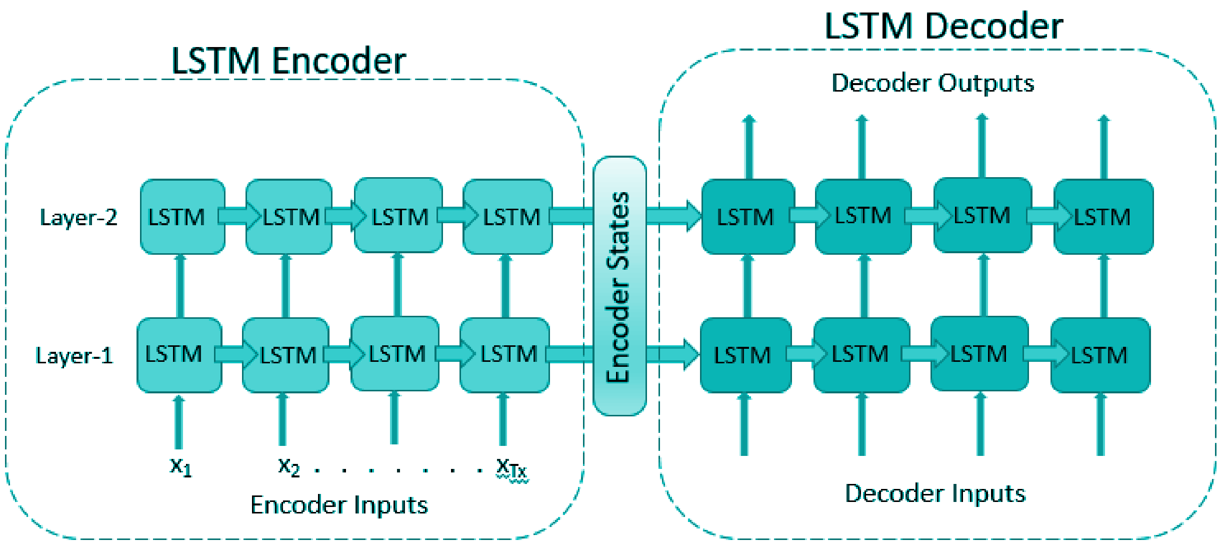


Figure 2.10
Encoder Decoder LSTM Architecture with two encoders and two decoders.

```
Layer (type)                   Output Shape          Param #     Connected to
==================================================================================================
input_8 (InputLayer)           [(None, 10, 18)]      0           []

lstm_30 (LSTM)                 [(None, 10, 100),     47600       ['input_8[0][0]']
                                (None, 100),
                                (None, 100)]

lstm_31 (LSTM)                 [(None, 100),         80400       ['lstm_30[0][0]']
                                (None, 100),
                                (None, 100)]

repeat_vector_7 (RepeatVector)  (None, 5, 100)       0           ['lstm_31[0][0]']

lstm_32 (LSTM)                 (None, 5, 100)        80400       ['repeat_vector_7[0][0]',
                                                                  'lstm_30[0][1]',
                                                                  'lstm_30[0][2]']

lstm_33 (LSTM)                 (None, 5, 100)        80400       ['lstm_32[0][0]',
                                                                  'lstm_31[0][1]',
                                                                  'lstm_31[0][2]']

dropout_5 (Dropout)            (None, 5, 100)        0           ['lstm_33[0][0]']

time_distributed_7 (TimeDistri  (None, 5, 18)        1818        ['dropout_5[0][0]']
buted)

==================================================================================================
Total params: 290,618
Trainable params: 290,618
Non-trainable params: 0
```

Table 2.11

Model summary of E2D2 LSTM.

E2D2 comprises two layers of LSTM for encoder and two layers for decoder.Here , Xt the input at time step t and each Xt represented as a vector of fixed length and output of the first LSTM encoder layer fed to as an input to the second layer of LSTM encoder. The first state of the decoder is set to the final states of the decoder.LSTM in the decoder process single word at every time step and output of the first LSTM layer of decoder fed to as an input to the second layer of LSTM decoder and output of second layer of LSTM decoder considered as a final output.

Model Architecture - E4D4

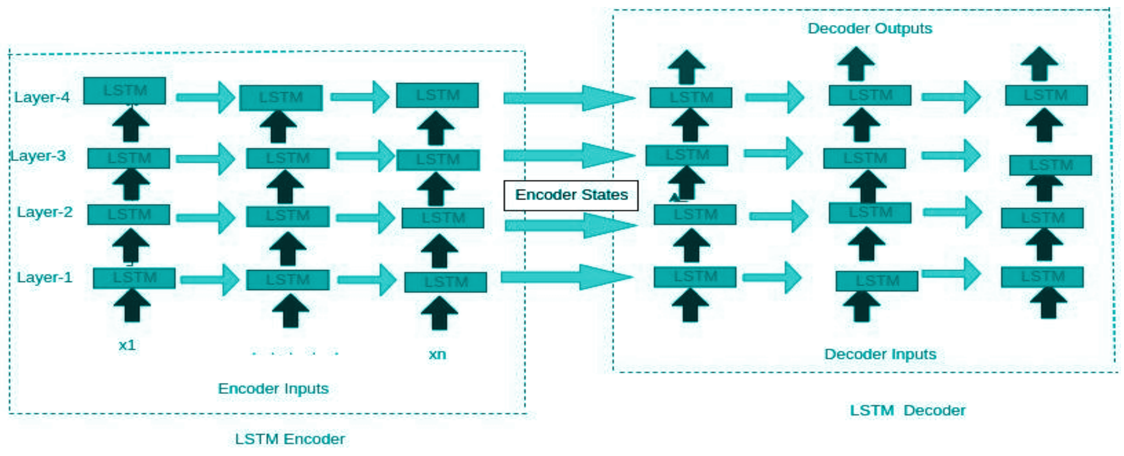
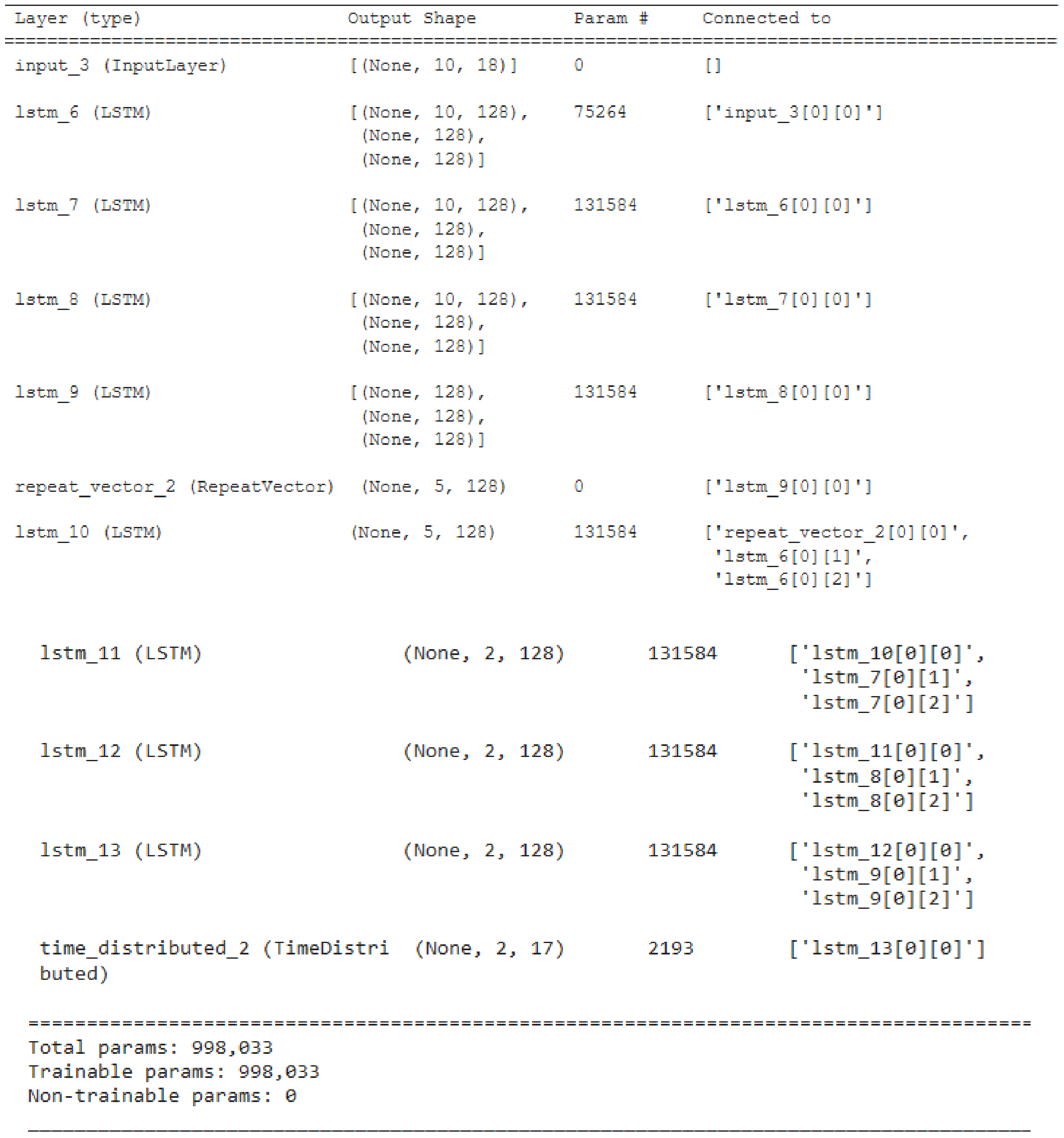


Figure 2.12

Encoder Decoder LSTM Architecture with four encoders and four decoders.

Add few points describing E4D4 (3-4 lines)



```
 Layer (type)                Output Shape         Param #     Connected to
==================================================================================================
 input_3 (InputLayer)        [(None, 10, 18)]     0           []

 lstm_6 (LSTM)               [(None, 10, 128),    75264       ['input_3[0][0]']
                              (None, 128),
                              (None, 128)]

 lstm_7 (LSTM)               [(None, 10, 128),    131584      ['lstm_6[0][0]']
                              (None, 128),
                              (None, 128)]

 lstm_8 (LSTM)               [(None, 10, 128),    131584      ['lstm_7[0][0]']
                              (None, 128),
                              (None, 128)]

 lstm_9 (LSTM)               [(None, 128),        131584      ['lstm_8[0][0]']
                              (None, 128),
                              (None, 128)]

 repeat_vector_2 (RepeatVector)  (None, 5, 128)   0           ['lstm_9[0][0]']

 lstm_10 (LSTM)              (None, 5, 128)       131584      ['repeat_vector_2[0][0]',
                                                               'lstm_6[0][1]',
                                                               'lstm_6[0][2]']

 lstm_11 (LSTM)              (None, 2, 128)       131584      ['lstm_10[0][0]',
                                                               'lstm_7[0][1]',
                                                               'lstm_7[0][2]']

 lstm_12 (LSTM)              (None, 2, 128)       131584      ['lstm_11[0][0]',
                                                               'lstm_8[0][1]',
                                                               'lstm_8[0][2]']

 lstm_13 (LSTM)              (None, 2, 128)       131584      ['lstm_12[0][0]',
                                                               'lstm_9[0][1]',
                                                               'lstm_9[0][2]']

 time_distributed_2 (TimeDistri  (None, 2, 17)    2193        ['lstm_13[0][0]']
 buted)

==================================================================================================
Total params: 998,033
Trainable params: 998,033
Non-trainable params: 0
__________________________________________________________________________________________________
```

Table 2.13

Model summary of E4D4 LSTM.

Architecture of E4D4 is the same as E1D1 or E2D2 but the number of lstm layers for encoder and decoder is increased to four. Output of the previous lstm layer will feed as input to the next lstm layer and output of the last lstm layer would store in a context vector which will further feed to as input to the first layer of decoder and output of the lstm last layer in decoder would considered as a final output.

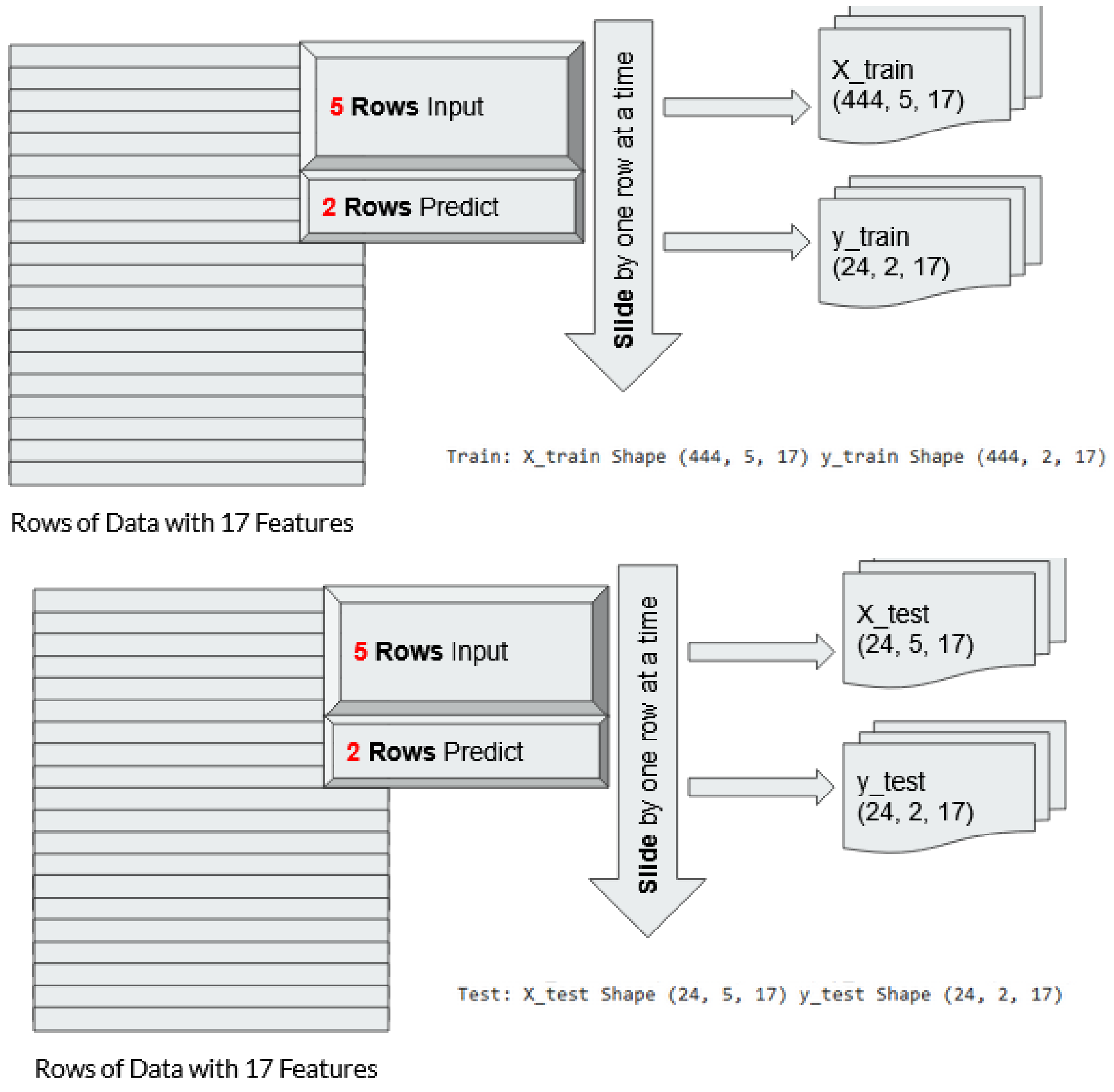
Adding more lstm layers in encoder-decoder will overfit the model. In lstm we are using the sliding window approach with a fixed window size and in a feed forward network keeping the sliding window size fixed the network is exposed to the same information regions for a very long time that would result in an overfit of our model.

Training of LSTM Model

LSTM is an architecture which is based on Recurrent Neural Network (RNN), there are some modifications in terms of dimension we need to do on the data before we fit that into our model. The first thing we will do is to Split Series into small batches of Time Series.

Time Series Generation

Our architecture is designed in a way to predict output of 2 months or rows after passing 5 months or rows to it. So the Input dimension of the data becomes (,5,17) and the Output dimension becomes (,2,17)



```
Train: X_train Shape (444, 5, 17) y_train Shape (444, 2, 17)
```

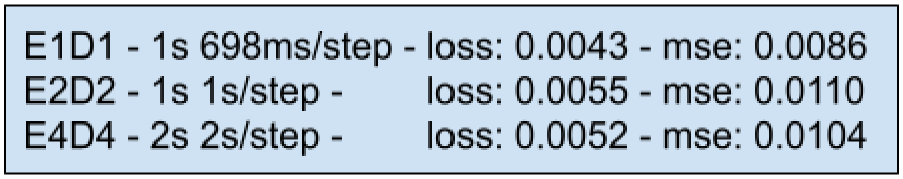
```
Test: X_test Shape (24, 5, 17) y_test Shape (24, 2, 17)
```

Figure 2.14
Showing the generation of time series and dimension of data passed into the model and its output.

From Figure 2.14, we can see that the shape is 3D, there is one extra dimension which basically represents the sliding window mechanism. This works like we iterate from the first row to the 5th row, and we predict from 6th row and 7th row. Now in the next iteration we will insert the 3rd row to the 7th row and get a prediction of the 8th and 9th row. A similar approach is followed while passing the validation sets in terms of dimension.

Architecture Selection

So far we have created 3 Encoder-Decoder LSTM Architectures, now we will compare their performance and pick up a model. Using the 3 architectures, we have trained the model and validated at the same time.



```
E1D1 - 1s 698ms/step - loss: 0.0043 - mse: 0.0086
E2D2 - 1s 1s/step -    loss: 0.0055 - mse: 0.0110
E4D4 - 2s 2s/step -    loss: 0.0052 - mse: 0.0104
```

Table 2.15

Comparative Study of 3 Architectures, where ‘loss’ and ‘mse’ means Validation Loss and Mean Squared Error.

Based on the results as shown in Figure 2.15, we can conclude that Model E1D1 is performing better with. Before finalizing model E3D3, we will do another comparative study.

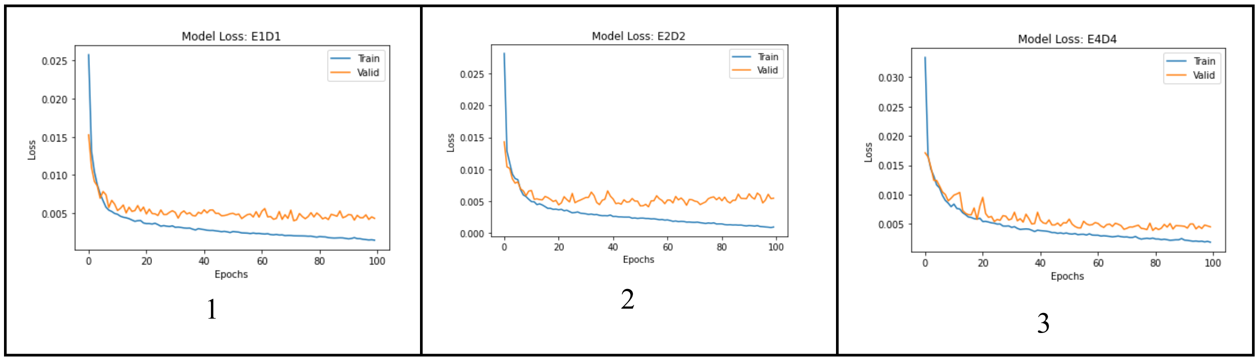


Figure 2.16

Comparative Study of 3 Architectures comparing training and validation loss. Model E1D1 is training slightly better than the others.

After comparing all the models we are moving ahead with the E1D1 architecture since it is showing the minimum loss and mean squared error.

Time Series Forecasting of Data

We have our model trained, now it is the time for prediction. This model will forecast data for next 24 months, but we have made the architecture of this model to take 5 months/ rows as input and give next 5 months as output. It is an obvious problem to predict more than 2 months, so there are multiple approaches available as suggested by *Brownlee 2020* are as follows.

1. Direct Multi-step Forecast Strategy:
   Here we make individual models for prediction of each step.
   prediction(t+1) = model1(obs(t-1), obs(t-2), ..., obs(t-n))
   prediction(t+2) = model2(obs(t-2), obs(t-3), ..., obs(t-n))
   This model is having one model for each time step, which is an added computational and maintenance burden, especially as the number of time steps to be forecasted increases beyond the trivial.
2. Recursive Multi-step Forecast:
   The recursive strategy involves using a one-step model multiple times where the prediction for the prior time step is used as an input for making a prediction on the following time step.
   prediction(t+1) = model(obs(t-1), obs(t-2), ..., obs(t-n))
   prediction(t+2) = model(prediction(t+1), obs(t-1), ..., obs(t-n))
   Predictions that are used in place of observations, the recursive strategy allows prediction errors to accumulate such that performance can quickly degrade as the prediction time horizon increases.

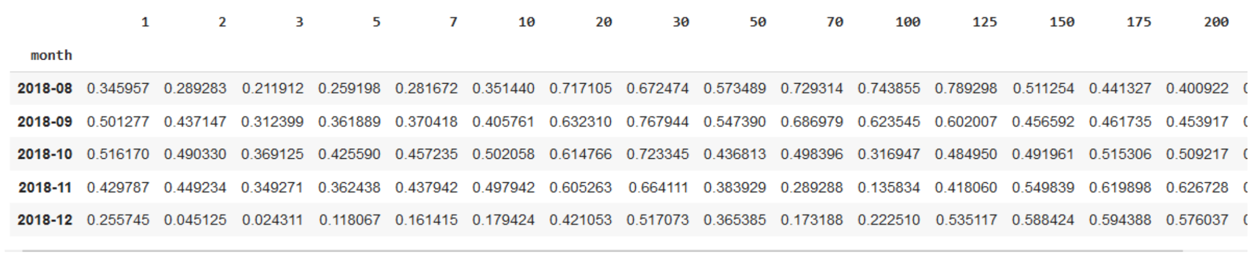
3. Direct-Recursive Hybrid Strategies:
   The direct and recursive strategies can be combined to offer the benefits of both methods.
   prediction(t+1) = model1(obs(t-1), obs(t-2), ..., obs(t-n))
   prediction(t+2) = model2(prediction(t+1), obs(t-1), ..., obs(t-n))

4. Multiple Output Strategy:
   The multiple output strategy involves developing one model that is capable of predicting the entire forecast sequence in a one-shot manner.
   prediction(t+1), prediction(t+2) = model(obs(t-1), obs(t-2), ..., obs(t-n))
   Multiple output models are more complex as they can learn the dependence structure between inputs and outputs as well as between outputs. Being more complex may mean that they are slower to train and require more data to avoid overfitting the problem.

Out of 4 multistep prediction strategies, we have selected "**Recursive multi-step forecast**". In our case we did not train the new models for every prediction since we had trained a single model with a huge amount of data.

Implementation of Recursive Multi-Step Forecast

Our model can predict 2 more rows/months after we provide 5 rows/months as input. That is the way the sliding window mechanism works. As the problem statement says that, we need to predict all the values in the years 2019 and 2020, comprising 24 months. So, we have taken the scaled data frame containing actual values from 2018-03 to 2018-12, which matches the size of input of 5 rows.



month	1	2	3	5	7	10	20	30	50	70	100	125	150	175	200
2018-08	0.345957	0.289283	0.211912	0.259198	0.281672	0.351440	0.717105	0.672474	0.573489	0.729314	0.743855	0.789298	0.511254	0.441327	0.400922
2018-09	0.501277	0.437147	0.312399	0.361889	0.370418	0.405761	0.632310	0.767944	0.547390	0.686979	0.623545	0.602007	0.456592	0.461735	0.453917
2018-10	0.516170	0.490330	0.369125	0.425590	0.457235	0.502058	0.614766	0.723345	0.436813	0.498396	0.316947	0.484950	0.491961	0.515306	0.509217
2018-11	0.429787	0.449234	0.349271	0.362438	0.437942	0.497942	0.605263	0.664111	0.383929	0.289288	0.135834	0.418060	0.549839	0.619898	0.626728
2018-12	0.255745	0.045125	0.024311	0.118067	0.161415	0.179424	0.421053	0.517073	0.365385	0.173188	0.222510	0.535117	0.588424	0.594388	0.576037

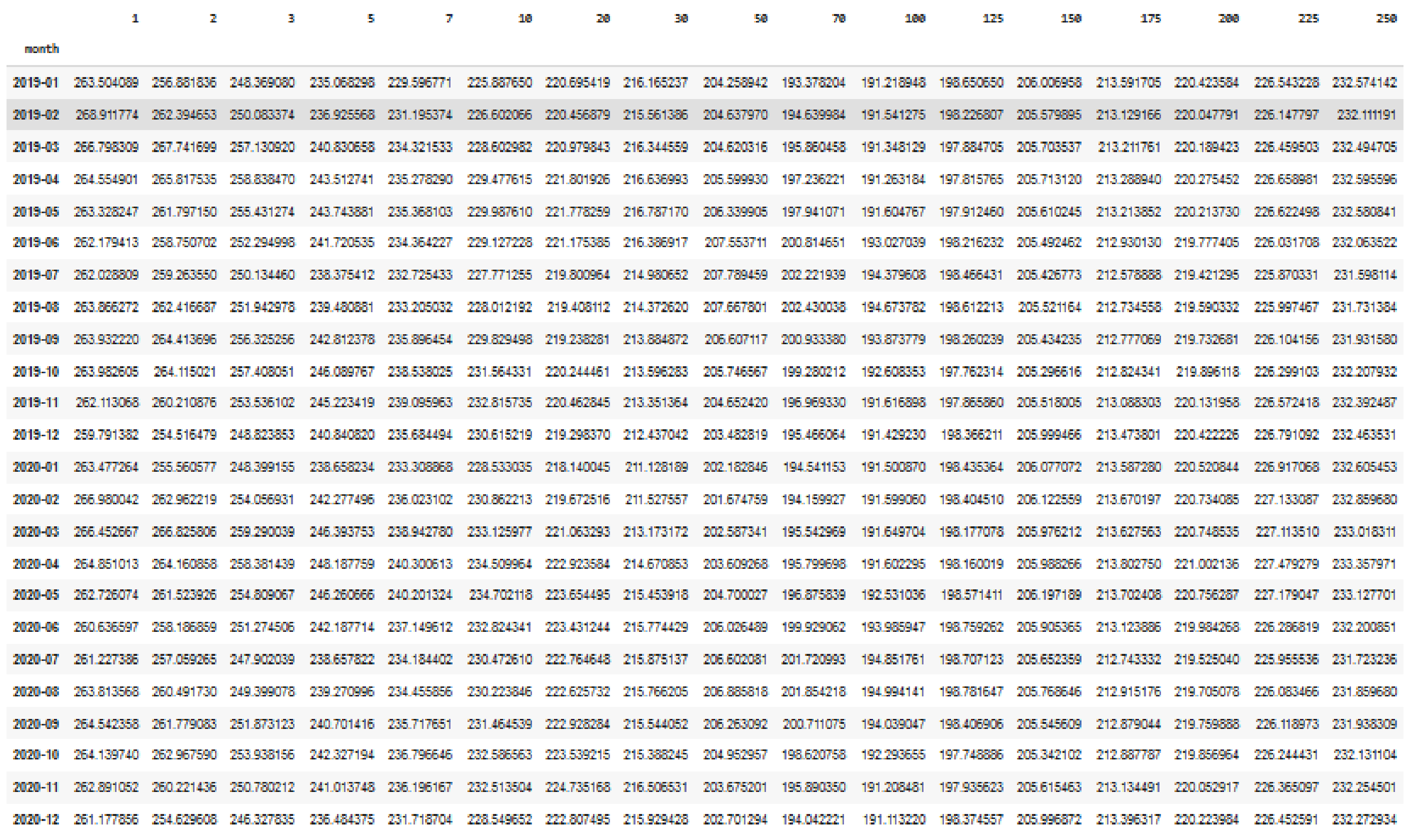
Table 2.17

5 rows or samples to be passed as input in the model.

One we have passed this data present in Figure 2.17, we are getting 5 new rows predicted. Again in the next iteration will drop the first row and append the first row from the predicted data. With every iteration we keep adding the first row that is generated in the prediction output to a separate data frame. Once we complete 24 iterations, we get a dataset containing all predicted values.

Results

After the model makes the prediction, we get a dataframe as a result that contains predicted values of 2019 and 2020.



month	1	2	3	5	7	10	20	30	50	70	100	125	150	175	200	225	250
2019-01	263.504089	256.881836	248.369080	235.068298	229.596771	225.887650	220.695419	216.165237	204.258942	193.378204	191.218948	198.650650	206.006958	213.591705	220.423584	226.543228	232.574142
2019-02	268.911774	262.394653	250.083374	236.925568	231.195374	226.602066	220.456879	215.561386	204.637970	194.639984	191.541275	198.226807	205.579895	213.129166	220.047791	226.147797	232.111191
2019-03	266.798309	267.741699	257.130920	240.830658	234.321533	228.602982	220.979843	216.344559	204.620316	195.860458	191.348129	197.884705	205.703537	213.211761	220.189423	226.459503	232.494705
2019-04	264.554901	265.817535	258.838470	243.512741	235.278290	229.477615	221.801926	216.636993	205.599930	197.236221	191.263184	197.815765	205.713120	213.288940	220.275452	226.658981	232.595596
2019-05	263.328247	261.797150	255.431274	243.743881	235.368103	229.987610	221.778259	216.787170	206.339905	197.941071	191.604767	197.912460	205.610245	213.213852	220.213730	226.622498	232.580841
2019-06	262.179413	258.750702	252.294998	241.720535	234.364227	229.127228	221.175385	216.386917	207.553711	200.814651	193.027039	198.216232	205.492462	212.930130	219.777405	226.031708	232.063522
2019-07	262.028809	259.263550	250.134460	238.375412	232.725433	227.771255	219.800964	214.980652	207.789459	202.221939	194.379608	198.466431	205.426773	212.578888	219.421295	225.870331	231.598114
2019-08	263.866272	262.416687	251.942978	239.480881	233.205032	228.012192	219.408112	214.372620	207.667801	202.430038	194.673782	198.612213	205.521164	212.734558	219.590332	225.997467	231.731384
2019-09	263.932220	264.413696	256.325256	242.812378	235.896454	229.829498	219.238281	213.884872	206.607117	200.933380	193.873779	198.260239	205.434235	212.777069	219.732681	226.104156	231.931580
2019-10	263.982605	264.115021	257.408051	246.089767	238.538025	231.564331	220.244461	213.596283	205.746567	199.280212	192.608353	197.762314	205.296616	212.824341	219.896118	226.299103	232.207932
2019-11	262.113068	260.210876	253.536102	245.223419	239.095963	232.815735	220.462845	213.351364	204.652420	196.969330	191.616898	197.865860	205.518005	213.088303	220.131958	226.572418	232.392487
2019-12	259.791382	254.516479	248.823853	240.840820	235.684494	230.615219	219.298370	212.437042	203.482819	195.466064	191.429230	198.366211	205.999466	213.473801	220.422226	226.791092	232.463531
2020-01	263.477264	255.560577	248.399155	238.658234	233.308868	228.533035	218.140045	211.128189	202.182846	194.541153	191.500870	198.435364	206.077072	213.587280	220.520844	226.917068	232.605453
2020-02	266.980042	262.962219	254.056931	242.277496	236.023102	230.862213	219.672516	211.527557	201.674759	194.159927	191.599060	198.404510	206.122559	213.670197	220.734085	227.133087	232.859680
2020-03	266.452667	266.825806	259.290039	246.393753	238.942780	233.125977	221.063293	213.173172	202.587341	195.542969	191.649704	198.177078	205.976212	213.627563	220.748535	227.113510	233.018311
2020-04	264.851013	264.160858	258.381439	248.187759	240.300613	234.509964	222.923584	214.670853	203.609268	195.799698	191.602295	198.160019	205.988266	213.802750	221.002136	227.479279	233.357971
2020-05	262.726074	261.523926	254.809067	246.260666	240.201324	234.702118	223.654495	215.453918	204.700027	196.875839	192.531036	198.571411	206.197189	213.702408	220.756287	227.179047	233.127701
2020-06	260.636597	258.186859	251.274506	242.187714	237.149612	232.824341	223.431244	215.774429	206.026489	199.929062	193.985947	198.759262	205.905365	213.123886	219.984268	226.286819	232.200851
2020-07	261.227386	257.059265	247.902039	238.657822	234.184402	230.472610	222.764648	215.875137	206.602081	201.720993	194.851761	198.707123	205.652359	212.743332	219.525040	225.955536	231.723236
2020-08	263.813568	260.491730	249.399078	239.270996	234.455856	230.223846	222.625732	215.766205	206.885818	201.854218	194.994141	198.781647	205.768646	212.915176	219.705078	226.083466	231.859680
2020-09	264.542358	261.779083	251.873123	240.701416	235.717651	231.464539	222.928284	215.544052	206.263092	200.711075	194.039047	198.406906	205.545609	212.879044	219.759888	226.118973	231.938309
2020-10	264.139740	262.967590	253.938156	242.327194	236.796646	232.586563	223.539215	215.388245	204.952957	198.620758	192.293655	197.748886	205.342102	212.887787	219.856964	226.244431	232.131104
2020-11	262.891052	260.221436	250.780212	241.013748	236.196167	232.513504	224.735168	216.506531	203.675201	195.890350	191.208481	197.935623	205.615463	213.134491	220.052917	226.365097	232.254501
2020-12	261.177856	254.629608	246.327835	236.484375	231.718704	228.549652	222.807495	215.929428	202.701294	194.042221	191.113220	198.374557	205.996872	213.396317	220.223984	226.452591	232.272934

Table 2.18

Dataframe of predicted values

Chapter 3: Results and Discussion

The evaluation of the model's accuracy is a crucial stage in every deep learning model. In regression analysis, the Mean Squared Error, Mean Absolute Error, Root Mean Squared Error, and R-Squared or Coefficient of Determination metrics are used to assess the model's performance.

Finding correlation between Actual and Predicted

Pearson Correlation Coefficient

Correlation coefficients are used to measure how strong a relationship is between two variables. There are several types of correlation coefficient, but here we will be using Pearson Correlation method that indicates the presence or absence of correlation between any two variables and determines the exact extent or degree to which they are correlated. It is a statistical measure of the degree of linear correlation between two variables.

X = Actual data for a particular pressure level.
Y = Predicted data for a particular pressure level.

covariance(X, Y) = (sum (x - mean(X)) * (y - mean(Y))) * 1/(n-1)

Pearson's correlation coefficient = covariance(X, Y) / (stdv(X) * stdv(Y))

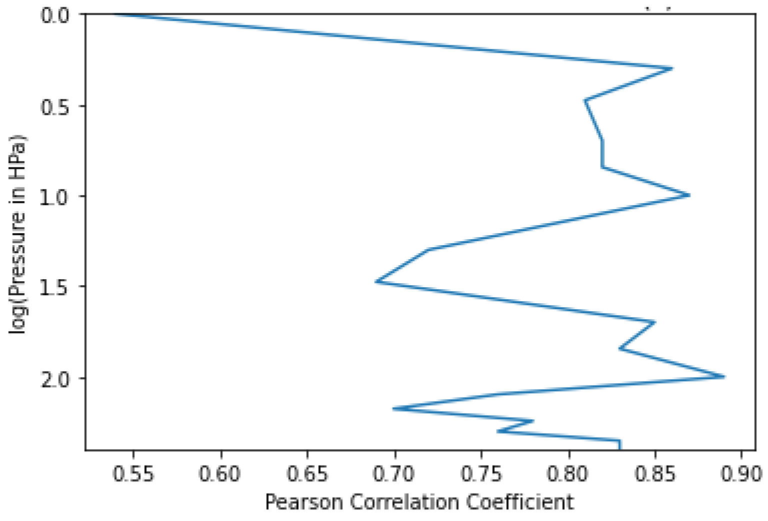


Figure 3.1

Plot showing the relationship between log(Pressure in HPa) in reversed order which means the altitude (lower the value, higher the altitude) and the pearson correlation coefficient

Root Mean Squared Error (RMSE)

The average of the squared difference between the actual and projected values in the data set is the Mean Squared Error (MSE). It calculates the residuals' variance. The square root of Mean Squared Error is Root Mean Squared Error.

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2}$$

y = Actual data for a particular pressure level

$\hat{y}$ = Predicted data for a particular pressure level

When compared to Mean Absolute Error (MAE), Mean Squared Error (MSE) and Root Mean Square Error (RMSE) penalizes big prediction mistakes. RMSE, on the other hand, is more commonly employed than MSE to compare the performance of a regression model to that of other random models.

Linear Regression Line

Given a scatter plot, we can draw the line that best fits the data.
Let, the equation of the line is "y = a + bx".
Where a is the y-intercept and b is the slope. x is the independent or predictor variable and y is the dependent or response variable. To find a and b we follow the steps:

$$b = \frac{\Sigma xy - (\Sigma x \Sigma y)/n}{\Sigma x^2 - (\Sigma x)^2/n}$$

$$a = \bar{y} - b\bar{x}$$

Using the results discussed above we matched the predicted values with the original values by visualizing them into graphs, scatter plots and calculating some important features like pearson's correlation and RMSE to check the accuracy of our model. Actual vs. predicted scatter plots are one of the most versatile types of data visualization, where all points should be near a regressed diagonal line.

Mean Absolute Error

Mean absolute error is the average of the difference between true value and the predicted value. It is one of the most widely accepted loss functions. The advantage of using MAE as a loss function is that it is not sensitive to outliers and gives a magnitude of how far the predicted values are from actual values. Thus, it is a good option for assessing the performance of regression models. It is given by:

$$\mathbf{MAE} = \frac{1}{n}\sum_{i=1}^{n} | x_i - x |$$

Mean Absolute Percentage Error (MAPE)

According to Swamidass et al, the mean absolute percentage error (MAPE) is the mean or average of the absolute percentage errors of forecasts. Error is defined as actual or observed value minus the forecasted value. Percentage errors are summed without regard to sign to compute MAPE. The smaller the MAPE the better the forecast.

$$\mathrm{M} = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right|$$

Where,

- n is the number of fitted points,
- A_t is the actual value,
- F_t is the forecast values

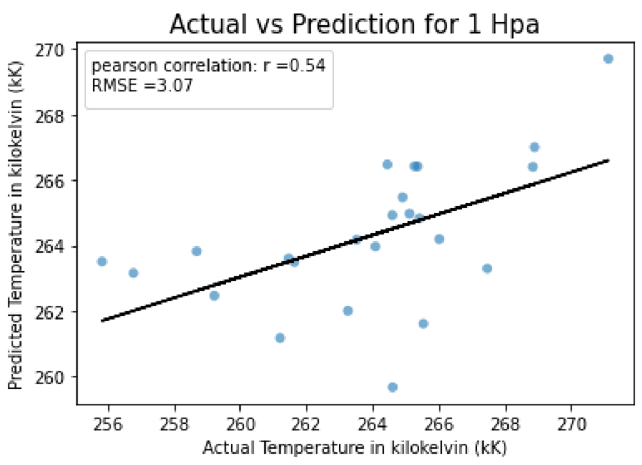
R-Squared Score

Lower MAE, MSE, and RMSE values indicate that a regression model is more accurate. A greater R square value, on the other hand, is regarded favourable. R squared is a statistical measure in a regression model that describes the proportion of variance in the dependent variable with respect to the independent variables. In other words, it shows how well the data fits the regression model. We needed this metric to assess the fitting of data into our selected model. The value of R2 score ranges between 0 and 1. If the value is 1, it shows that the model completely fits the data. It is given by:

$$R^2 = 1 - \frac{\overset{\text{Sum Squared Regression Error}}{SS_{Regression}}}{\underset{\text{Sum Squared Total Error}}{SS_{Total}}}$$

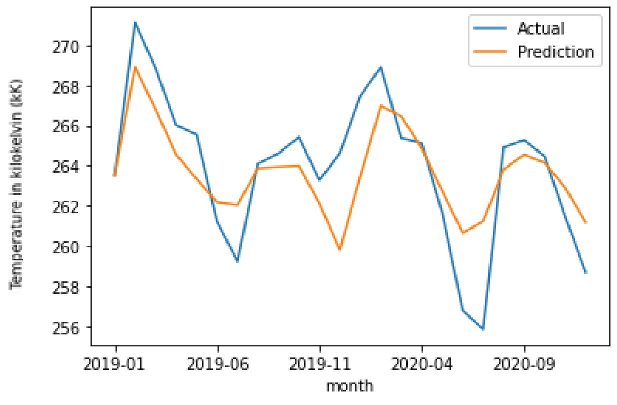
On calculating, we found that the R2 score of our model is 0.4005 which means that around 40% of our data exactly fits in the model and predicted accurately.

Individual column predictions



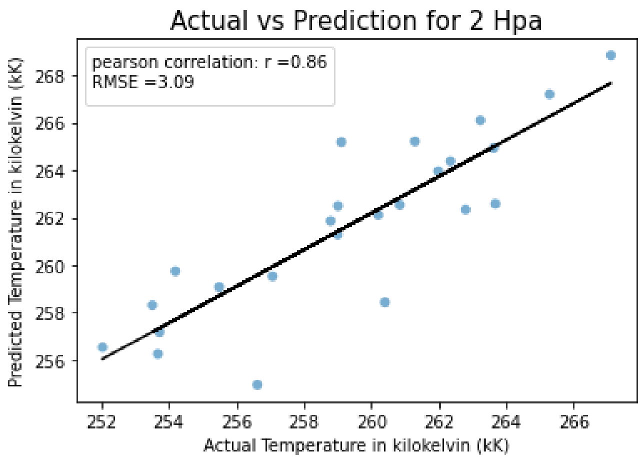
1(a)

R-squared: 0.28



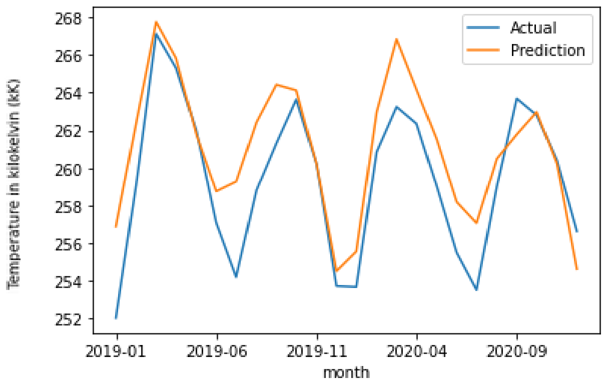
1(b)

MAE: 2.29
MAPE: 0.87



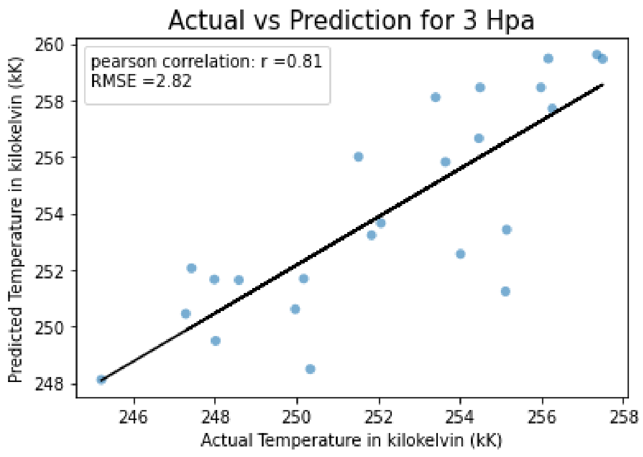
2(a)

R-squared: 0.41



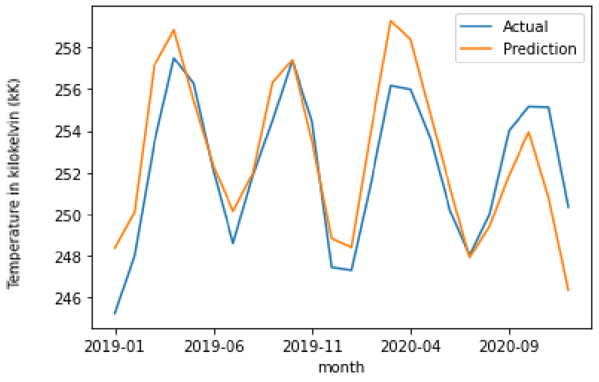
2(b)

MAE: 2.76
MAPE: 1.06



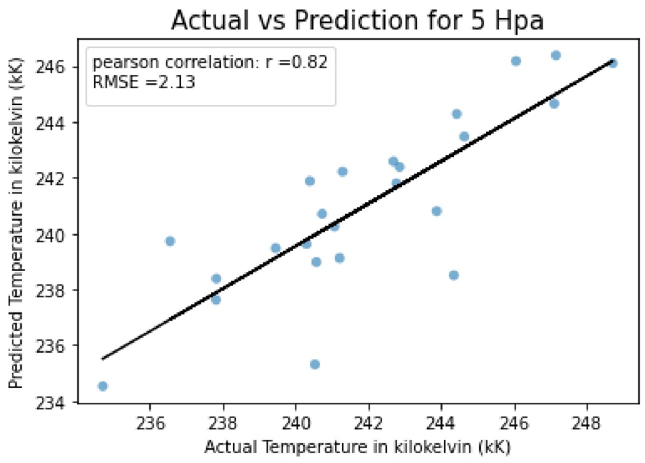
3(a)

R-squared: 0.35



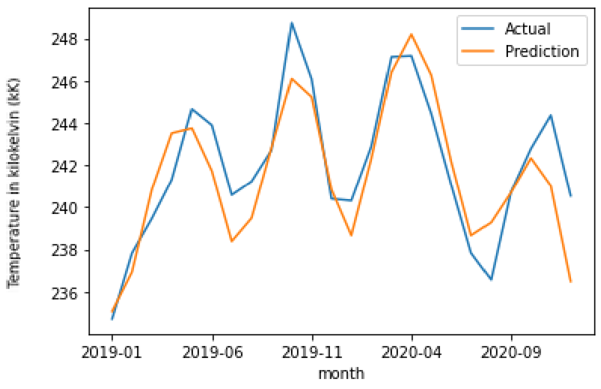
3(b)

MAE: 2.57
MAPE: 1.01



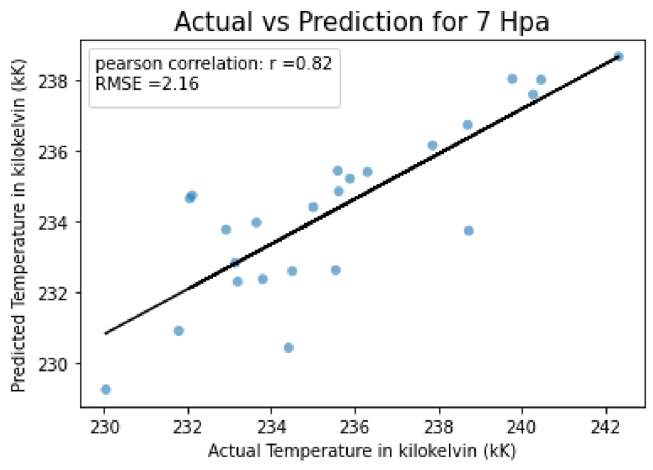
4(a)

R-squared: 0.6



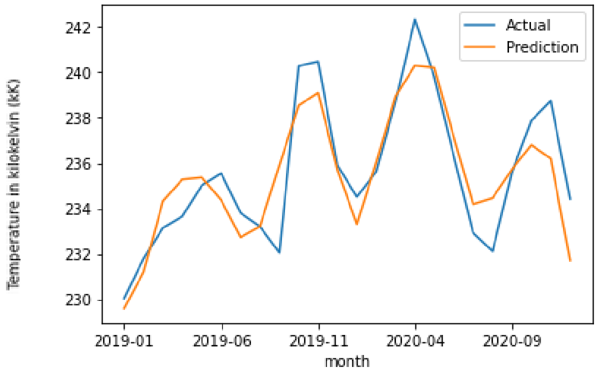
4(b)

MAE: 1.45
MAPE: 0.6



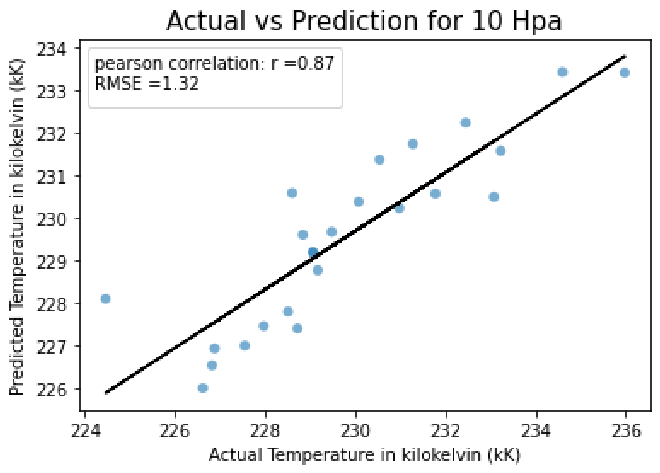
5(a)

R-squared: 0.52



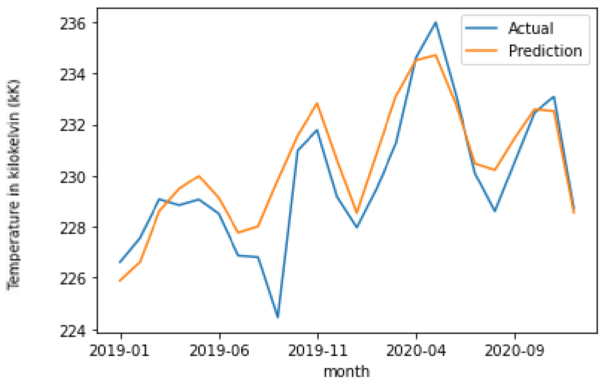
5(b)

MAE: 1.76
MAPE: 0.75



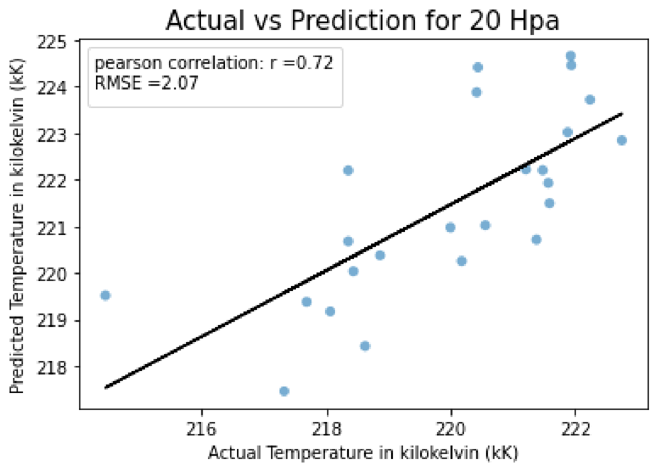
6(a)

R-squared: 0.75



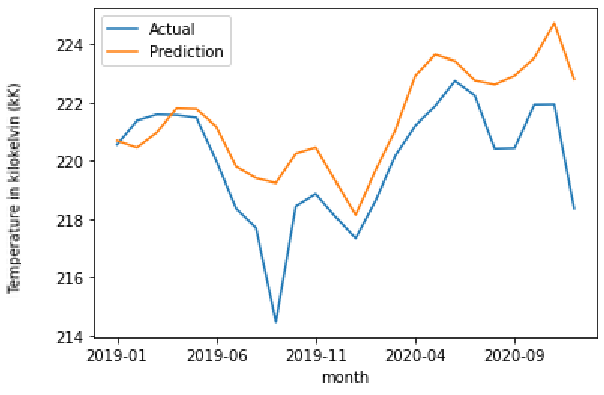
6(b)

MAE: 0.96
MAPE: 0.42



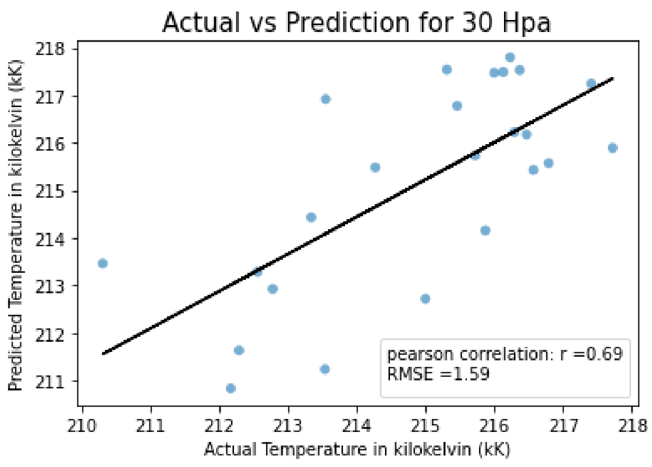
7(a)

R-squared: -0.11



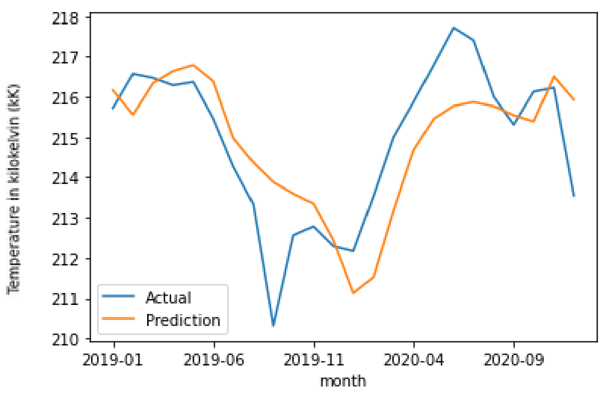
7(b)

MAE: 1.55
MAPE: 0.7



8(a)

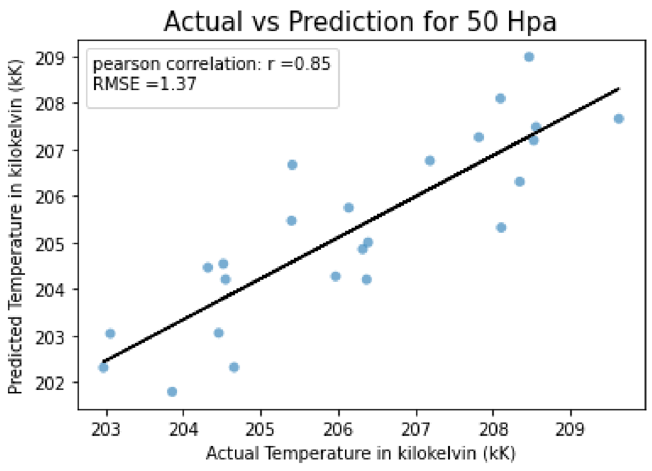
R-squared: 0.29



8(b)

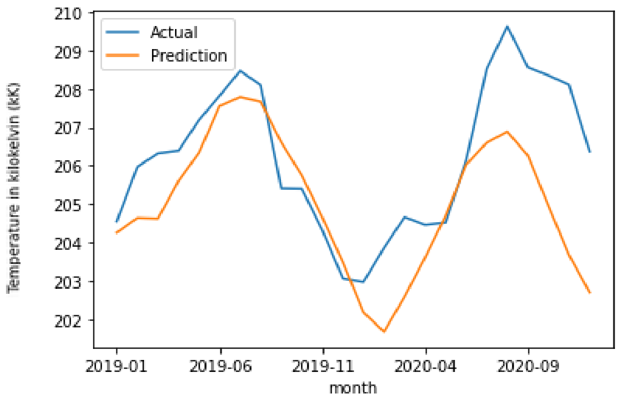
MAE: 1.33

MAPE: 0.62



9(a)

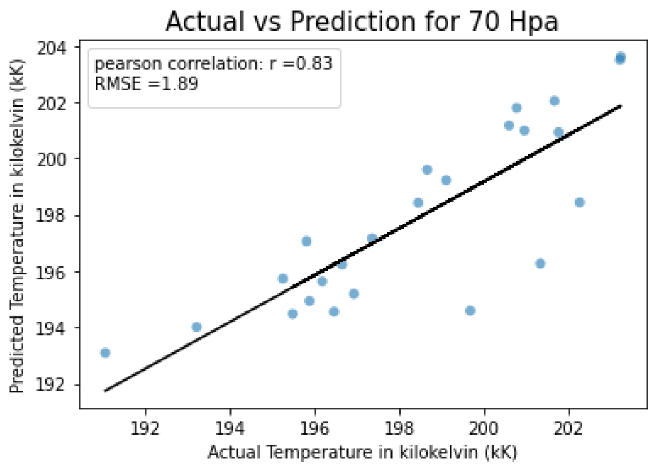
R-squared: 0.47



9(b)

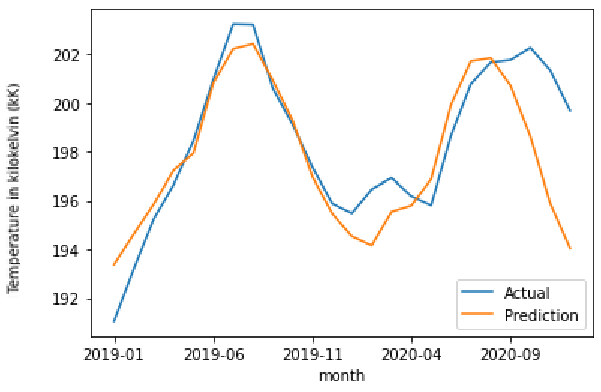
MAE: 1.09

MAPE: 0.53



10(a)

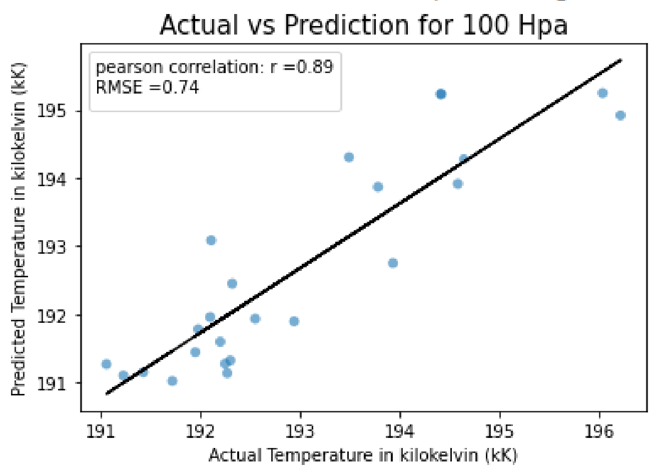
R-squared: 0.64



10(b)

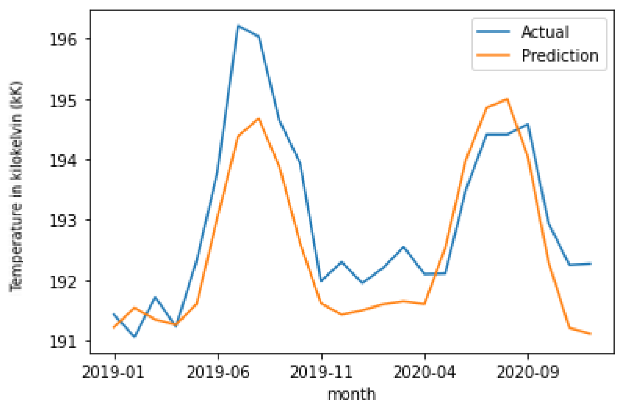
MAE: 1.25

MAPE: 0.63



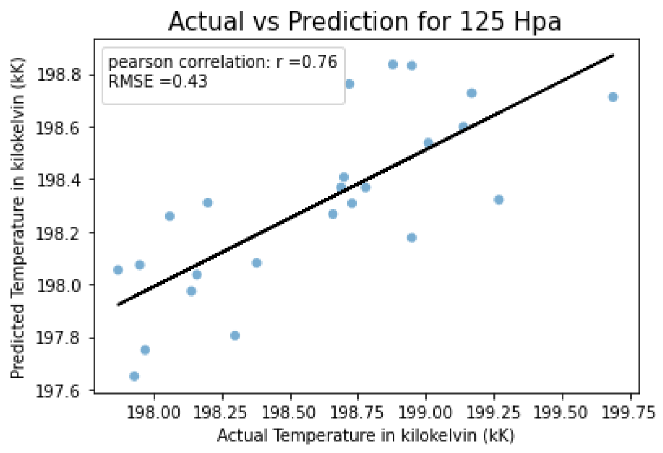
11(a)

R-squared: 0.72



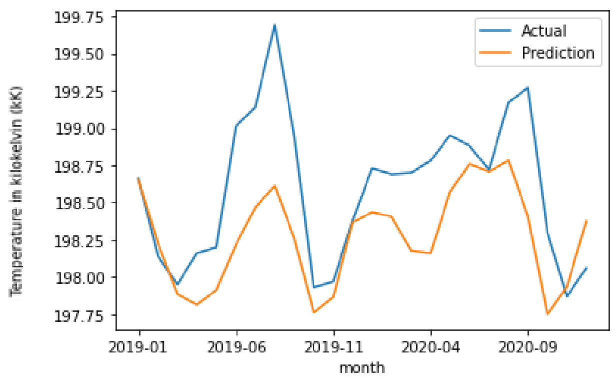
11(b)

MAE: 0.64
MAPE: 0.33



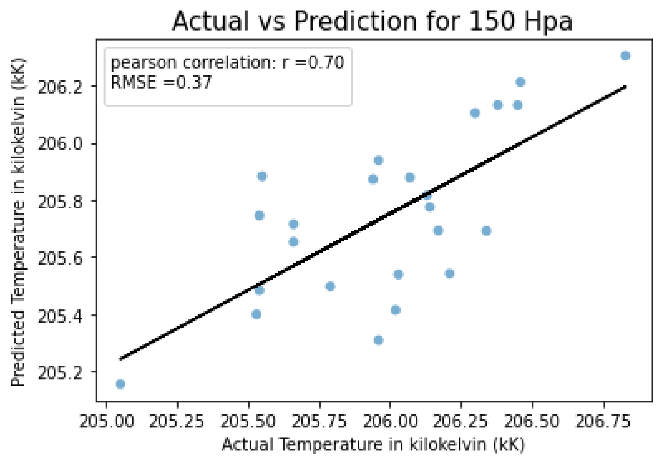
12(a)

R-squared: 0.2



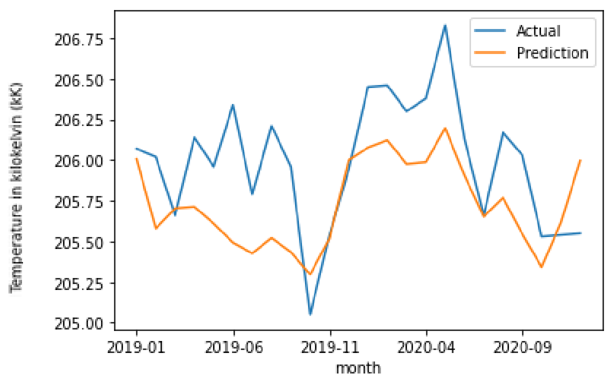
12(b)

MAE: 0.35
MAPE: 0.18



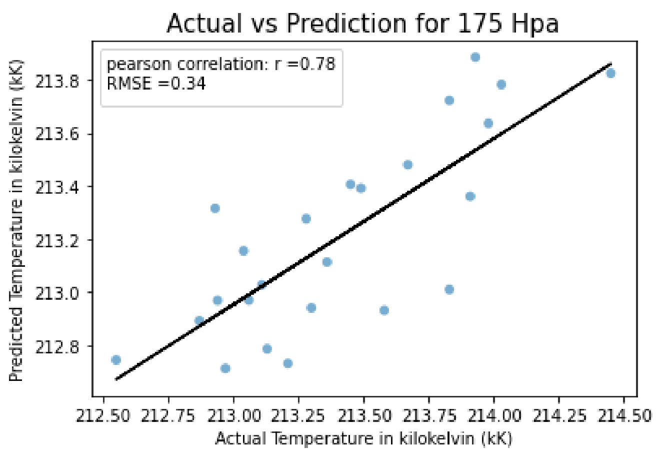
13(a)

R-squared: 0.11



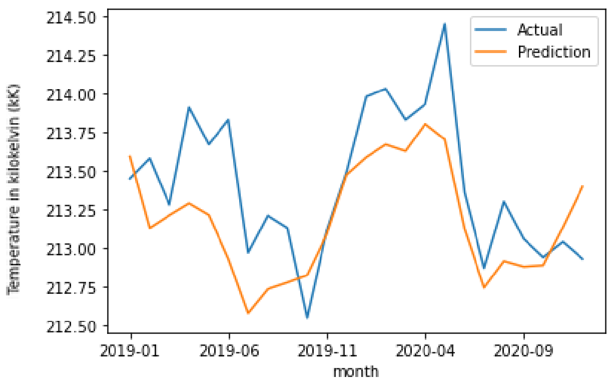
13(b)

MAE: 0.3
MAPE: 0.15



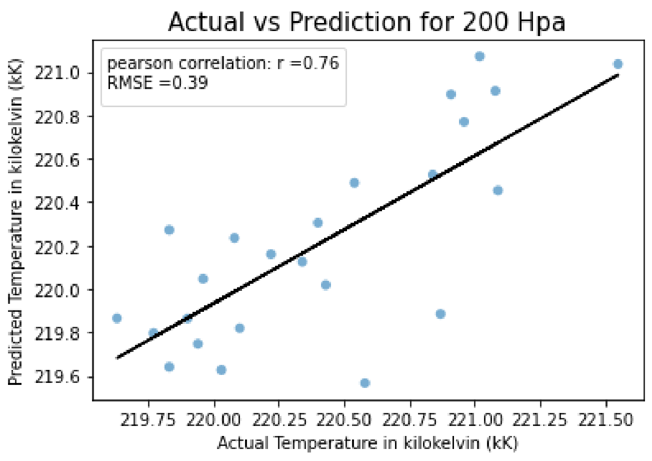
14(a)

R-squared:0.42



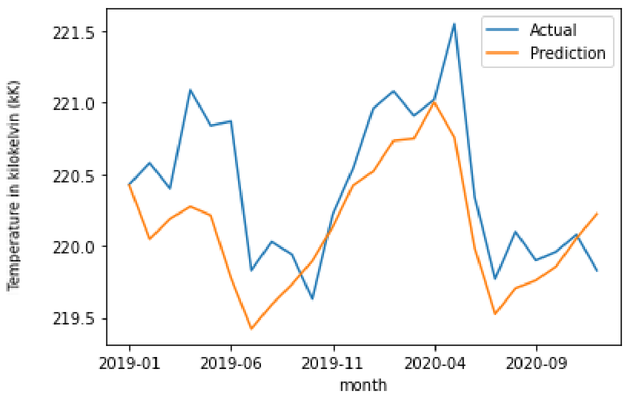
14(b)

MAE: 0.26
MAPE: 0.12



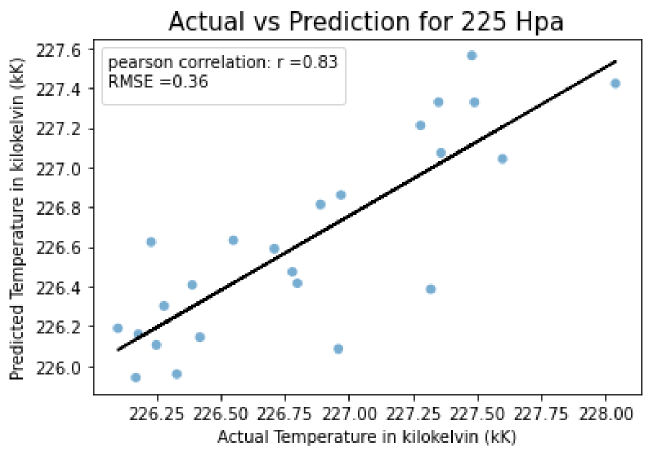
15(a)

R-squared: 0.42



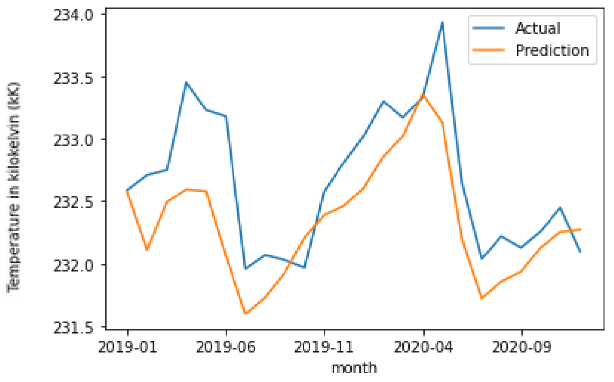
15(b)

MAE: 0.28
MAPE: 0.42



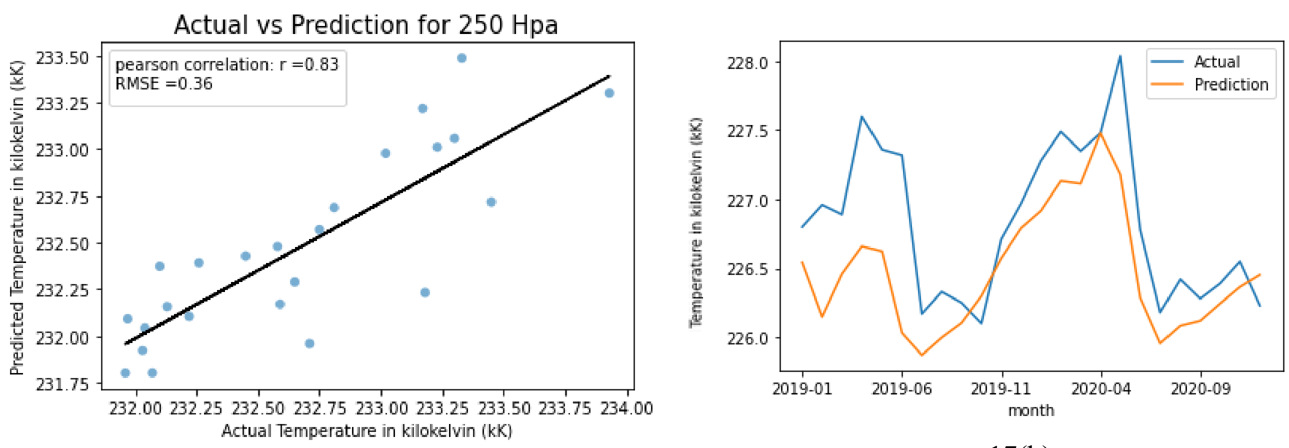
16(a)

R-squared: 0.55



16(b)

MAE: 0.26
MAPE: 0.11



17(a)

R-squared: 0.57

17(b)

MAE: 0.26

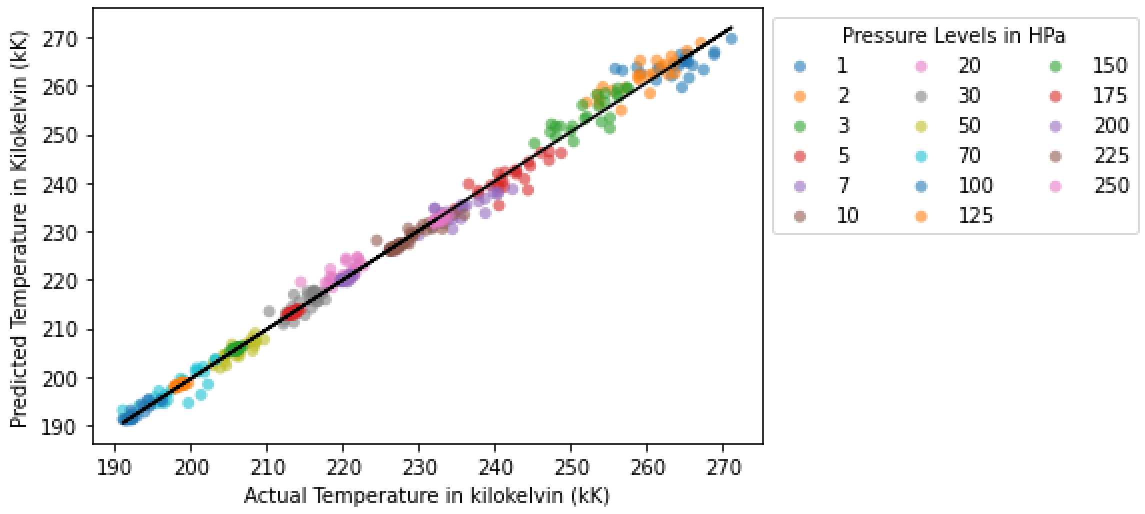
MAPE: 0.11

Figure 3.2

Actual vs Predicted Values Plot from 1 HPa to 250 HPa. In the Left column **(a)**: The scattered plot along with a regression line, Pearson correlation and RMSE is displayed. In the Right column **(b)**: We can visually see the results wrt time.

Overall prediction

```
pearson correlation: r = 0.99
RMSE = 1.74
```



Mean Actual Error (MAE): 1.14

R Squared: 0.99

Figure 3.3

Pearson Correlation and RMSE indicating the linear regression between the predictions and the actual data at various pressure levels.

The overall scores are better than the individual predictions because here the range is significantly higher. Here the range lies from 185kK to 270kK, whereas in the individual pressure levels the range (difference between the highest and lowest values) varies between 1kK to 10kK. The Pearson correlation coefficient and the R Square Score are almost perfect.

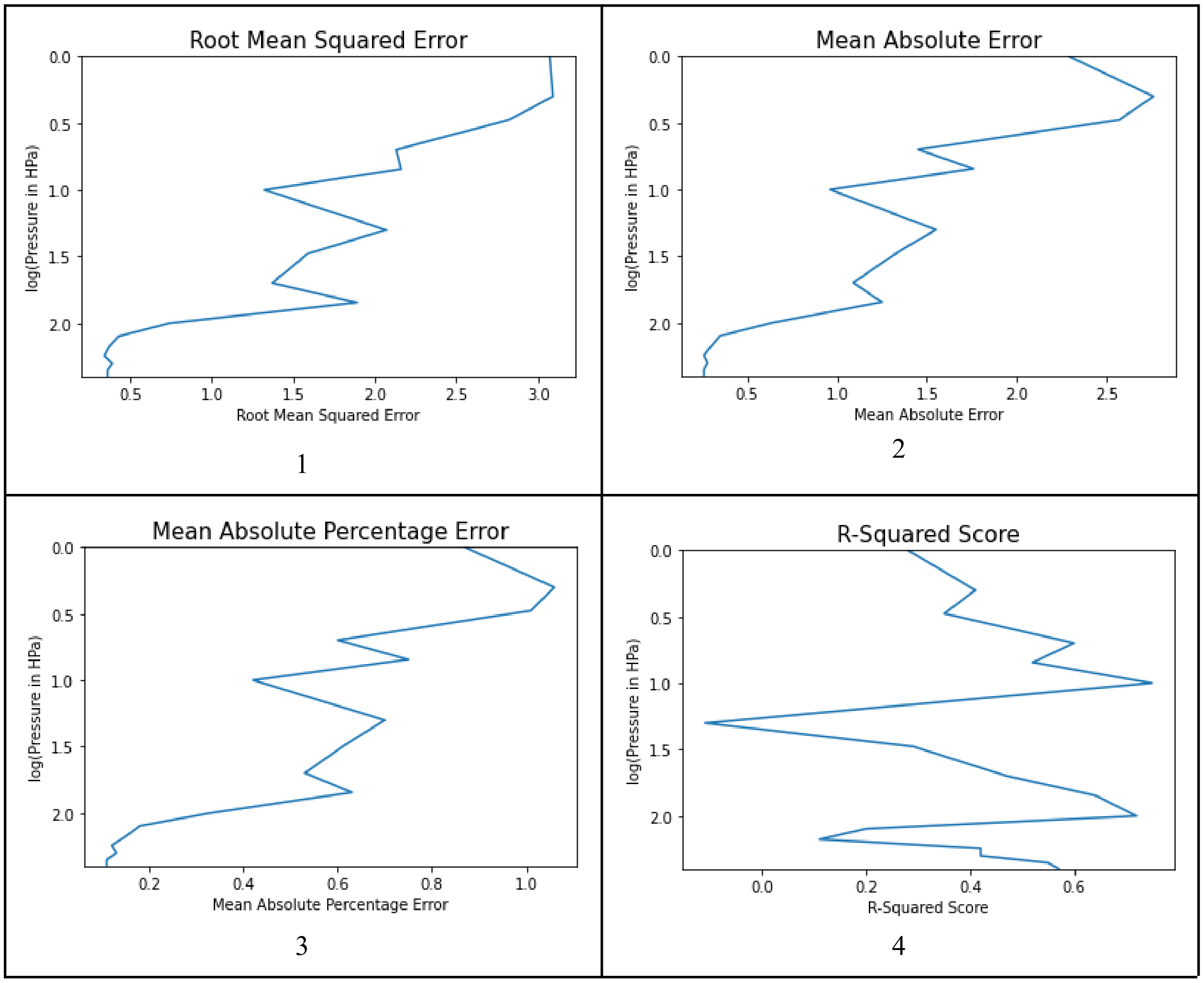
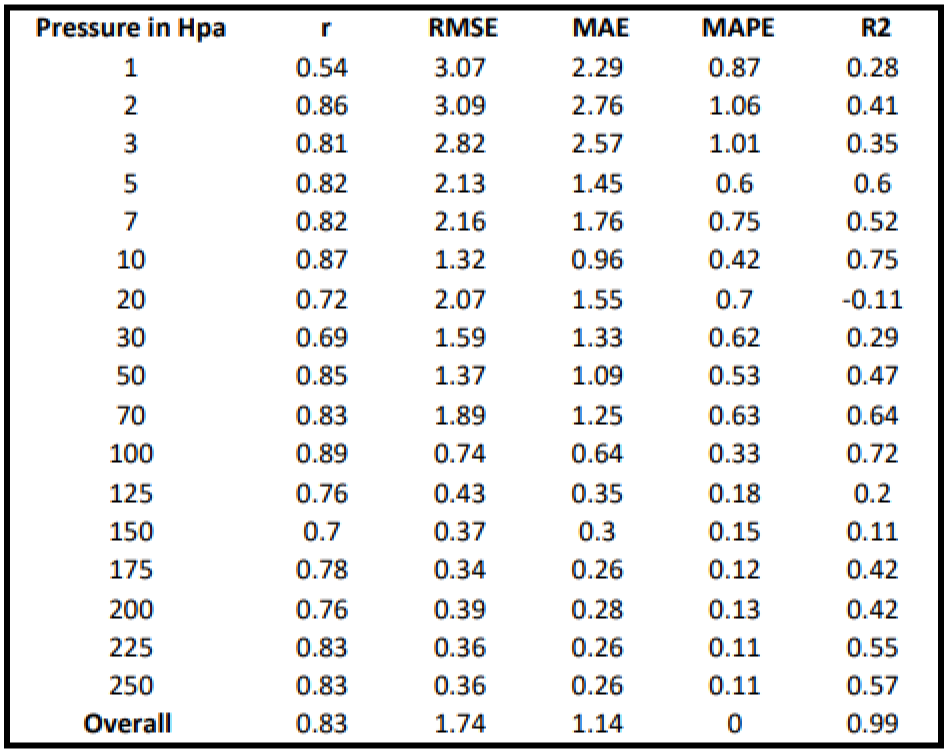


Figure 3.4

Plots of various regression accuracy metrics showing the accuracy of the LSTM model at various altitudes which is represented by the log(Pressure in HPa) in reverse order.
(1) Root Mean Squared Error - RMSE (Lower is better)
(2) Mean Absolute Error - MAE (Lower is better)
(3) Mean Absolute Percentage Error - MAPE (Lower is better)
and (4) R-Squared Score (Higher is better)

By observing the plots in Figure 3.3, we can see that there is a huge similarity among the results of Root Mean Squared Error - RMSE, Mean Absolute Error - MAE and Mean Absolute Percentage Error - MAPE. According to the metrics, the errors are higher in the higher altitude.



Pressure in Hpa	r	RMSE	MAE	MAPE	R2
1	0.54	3.07	2.29	0.87	0.28
2	0.86	3.09	2.76	1.06	0.41
3	0.81	2.82	2.57	1.01	0.35
5	0.82	2.13	1.45	0.6	0.6
7	0.82	2.16	1.76	0.75	0.52
10	0.87	1.32	0.96	0.42	0.75
20	0.72	2.07	1.55	0.7	-0.11
30	0.69	1.59	1.33	0.62	0.29
50	0.85	1.37	1.09	0.53	0.47
70	0.83	1.89	1.25	0.63	0.64
100	0.89	0.74	0.64	0.33	0.72
125	0.76	0.43	0.35	0.18	0.2
150	0.7	0.37	0.3	0.15	0.11
175	0.78	0.34	0.26	0.12	0.42
200	0.76	0.39	0.28	0.13	0.42
225	0.83	0.36	0.26	0.11	0.55
250	0.83	0.36	0.26	0.11	0.57
Overall	0.83	1.74	1.14	0	0.99

Table 3.5

Table showing various metrics indicating the performance of the model.

Discussion

Model selection is an important factor in predicting satisfactory results. Out of many other models, LSTM yields better data-fitting in long term modelling according to D. T. Hoang et al. 2020. In different random training of the model, it was found reduction in error rates. While predicting, LSTM modelling has the advantage of better error handling when it comes to presence of huge deviation in seasonal dataset. The reason is the possibility of learning noisy and nonlinear relationships with every step and explicit handling of ordered observations by adapting itself.

As observed in the results, the model performed very well in most of the pressure levels (columns). The accuracy can be compared with the one of the standard predictors used in other research works.

Chapter 4: Future Work

While predicting the future, there are many other approaches apart from multistep prediction involving recursion, we will work on those as this one is dependent on the error of the predicted outputs. We can track and predict the global warming alerts in future years.

Our good results with the temperature opens up the possibility of analyzing and predicting some other natural events such as rainfall, cyclones, etc.

References

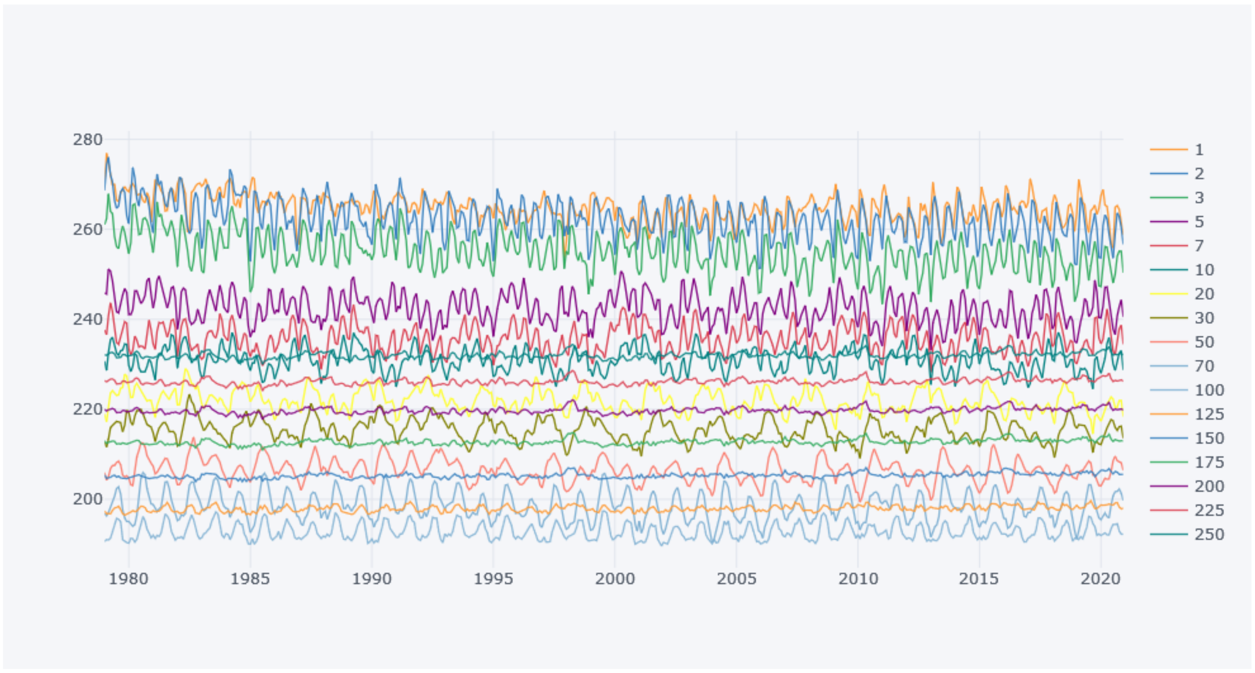
1. Hersbach, H., Bell, B., Berrisford, P., Biavati, G., Horányi, A., Muñoz Sabater, J., Nicolas, J., Peubey, C., Radu, R., Rozum, I., Schepers, D., Simmons, A., Soci, C., Dee, D., Thépaut, J-N. (2019): ERA5 monthly averaged data on pressure levels from 1979 to present. Copernicus Climate Change Service (C3S) Climate Data Store (CDS), 10.24381/cds.6860a573. DOI: 10.24381/cds.6860a573
2. Yan, Jining, Lin Mu, Liizhe Wang, Rajiv Ranjan, and Albert Y. Zomaya. n.d. "Temporal Convolutional Networks for the Advance Prediction of ENSO." 15 May, 2020. https://doi.org/10.1038/s41598-020-65070-5.
3. S. Poornima, and M. Pushpalatha. "Prediction of Rainfall Using Intensified LSTM Based Recurrent Neural Network with Weighted Linear Units." Atmosphere 2019 10. http://dx.doi.org/10.3390/atmos10110668
4. D. T. Hoang, Pr. L. Yang, L. D. P Cuong, and Dr. P. D. Trung. 2020. "Weather prediction based on the LSTM model implemented on AWS Machine Learning Platform." International Journal for Research in Applied Science and Engineering Technology IJRASET 8, no. 5 (May): 11. http://doi.org/10.22214/ijraset.2020.5046.
5. Brownlee, Jason. 2020. *Machine Learning Mastery With Python*. N.p.: Jason Brownlee.
6. Das, Uma, and C.J. Pan. (2016) "Equatorial atmospheric Kelvin waves during El Niño episodes and their effect on stratospheric QBO." *Science of the Total Environment*, 11. https://doi.org/10.1016/j.scitotenv.2015.12.009.
7. Butler, Amy (2019). Sub-Seasonal to Seasonal Prediction || Sub-seasonal Predictability and the Stratosphere. , (), 223–241. doi:10.1016/B978-0-12-811714-9.00011-5
8. Callimore, Christopher C. (2003). Journal of Climate Vol. 16, Issue 15 || On The Relationship between the QBO and Tropical Deep Convection.
   https://doi.org/10.1175/1520-0442(2003)016%3C2552:OTRBTQ%3E2.0.CO;2
9. M. P. Baldwin,L. J. Gray,T. J. Dunkerton,K. Hamilton,P. H. Haynes,W. J. Randel,J. R. Holton,M. J. Alexander,I. Hirota,T. Horinouchi,D. B. A. Jones,J. S. Kinnersley,C. Marquardt,K. Sato,M. Takahashi (2001). The quasi-biennial oscillation. May, 2001.
   https://doi.org/10.1029/1999RG000073
10. Daniel J. Jacob and Loretta J. Mickley, Introduction to Atmospheric Chemistry, Atmospheric Chemistry Modeling Group - Harvard University, Princeton University Press, 1999
11. Swamidass P.M. (eds) Encyclopedia of Production and Manufacturing Management. Springer, Boston, MA . MEAN ABSOLUTE PERCENTAGE ERROR (MAPE), 2000
   https://doi.org/10.1007/1-4020-0612-8_580

Appendix

Link To Notebooks/ Repository

Notebook 1 Model:
https://colab.research.google.com/drive/1WL1Gbgri76CP66XyxWdcKmVtg5VAm7y6?usp=sharing
Notebook 2 Plots:
https://colab.research.google.com/drive/1GT6bLitRbVZYEAPn1-LK8YsCXa-j91i6?usp=sharing
Notebook 3 Plots:
https://nbviewer.org/github/subhmm/MeanEquatorTemperature/blob/main/7thSemProject.ipynb

Plots Showing Temperature Variation at Each Pressure Levels on HPa.



Y-axis shows temperature and x-Axis shows time.