Smart Building Sensor Data Forecasting Using Time Series Analysis

Anant Vignesh Mahadhevan

Introduction

- The objective of the experiment/project is to determine the approach that best fits for this particular dataset in order to plot projection for 1 month in the future from 12/31/2019.
- In general, Time Series Analysis can be performed by two different methods in order to forecast future data. These methods are,
  - Statistical Models such as Holt's Method(DES), Holt-Winters Method(TES), ARMA, ARIMA, VAR, VARMA, etc.
  - Recurrent Neural networks such as LSTM, GRU, Bidirectional LSTM
- In this particular experiment to determine the most optimum model, I'm going to perform tests with both Statistical Models and also with RNNs to find which approach is able to forecast more closely to the actual value.
- For this particular dataset that I've chosen, I'm going to test, train and evaluate models using the following methods,
  - Statistical Models
    - Holt's Method (Double Exponential Smoothing)
    - Holt-Winters Method (Triple Exponential Smoothing)
    - Auto Regressive Integrated Moving Average (ARIMA)
    - Vector Auto Regression (VAR) to perform Multivariate Time Series Analysis
  - Recurrent Neural Network Using LSTM to perform Multivariate Time Series Analysis

About The Dataset

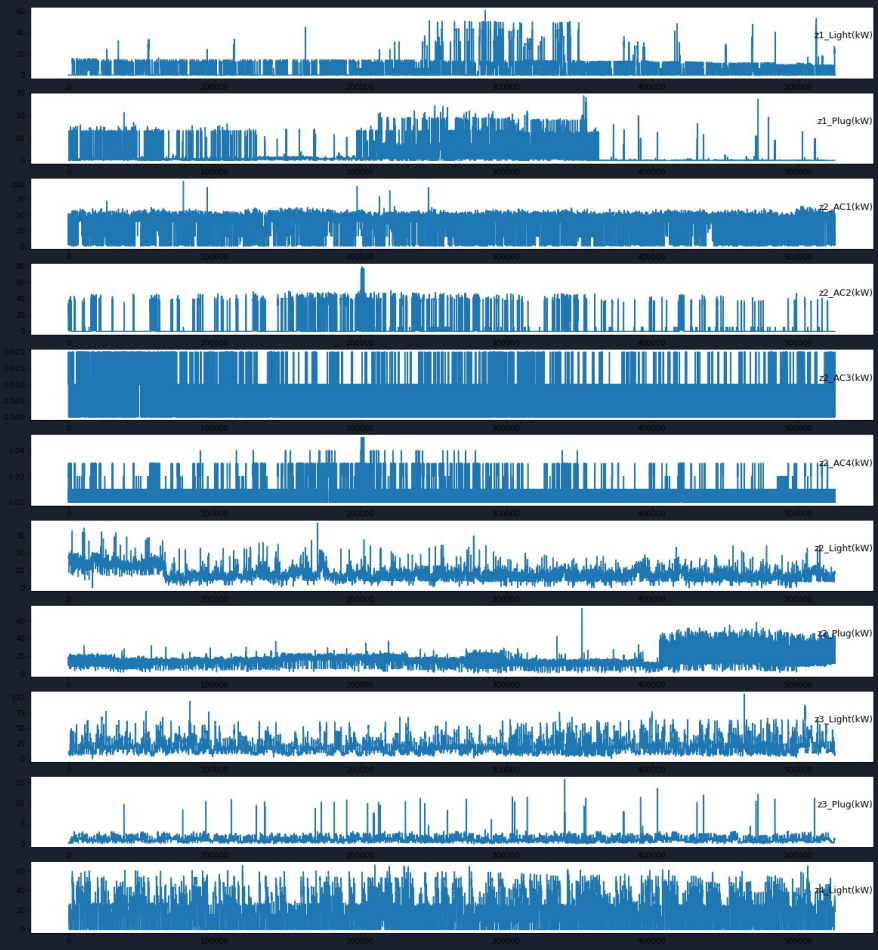
- CU-BEMS, smart building energy and IAQ data, is a data of Electricity and indoor measurements of an office building in Bangkok, Thailand.
- It is a detailed building operation data, including electricity consumption and indoor environmental measurements, of the seven-story 11,700-m2 office building located in Bangkok, Thailand.
- The data provided has the measurements available in each zone on the same floor of the building in a particular year.
- Each of the 2018 data files has 264,960 rows, which indicate one-minute interval data (1,440 data points/day) for 184 days during the second half year of 2018. Each of the 2019 data files has 525,600 rows, which indicate one-minute interval data (1,440 data points/day) for 365 days during the entire year of 2019.
- For the sake of simplicity, I am considering the data of just one floor which is the Floor 1 for the year 2018 and 2019.

- For example, the file 2019Floor1.csv, the data has 11 columns, and one timestamp column.
- These 11 data columns are:
  - Zone 1–Power consumption (kW) of lighting loads (one column)
  - Zone 2–Power consumption (kW) of four individual AC units
  - one lighting load and one plug load (six columns)
  - Zone 3–Power consumption (kW) of one lighting and one plug loads (two columns)
  - Zone 4–Power consumption (kW) of lighting and plug loads (two columns)

	A	B	C	D	E	F	G	H	I	J	K	L
1	Date	z1_Light(kW)	z1_Plug(kW)	z2_AC1(kW)	z2_AC2(kW)	z2_AC3(kW)	z2_AC4(kW)	z2_Light(kW)	z2_Plug(kW)	z3_Light(kW)	z3_Plug(kW)	z4_Light(kW)
2	01-01-2019 00:00	0.03	0.58	2.31	21.15	0.01	0.02	17.37	15.2	10.01	0.38	14.58
3	01-01-2019 00:01	0	0.58	2.31	35.07	0	0.02	17.34	19.16	9.98	0.37	14.57
4	01-01-2019 00:02	0.02	0.58	30.96	34.37	0.01	0.03	17.31	19.02	9.98	0.38	14.62
5	01-01-2019 00:03	0	0.57	51.32	18.91	0.01	0.01	17.39	18.85	10.01	0.37	14.6
6	01-01-2019 00:04	0.01	0.56	48.87	1.35	0.01	0.01	17.48	18.57	10.05	0.38	14.6
7	01-01-2019 00:05	0.02	0.58	40.85	1.35	0.01	0.01	17.54	18.16	10.03	0.37	14.59
8	01-01-2019 00:06	0.01	0.57	2.3	1.35	0.01	0	17.49	18.23	10.06	0.38	14.6
9	01-01-2019 00:07	0.02	0.57	2.31	1.36	0	0.01	17.54	14.03	10.05	0.38	14.6
10	01-01-2019 00:08	0.01	0.58	2.32	1.35	0.01	0	17.56	11.11	10.08	0.37	14.57
11	01-01-2019 00:09	0.02	0.56	2.31	1.35	0.01	0.01	17.41	11.08	10.03	0.38	14.59
12	01-01-2019 00:10	0.02	0.57	6.04	1.34	0	0.01	17.32	11.01	10.01	0.38	14.6
13	01-01-2019 00:11	0	0.57	51.83	1.34	0.01	0	17.25	9.81	9.95	0.37	14.61
14	01-01-2019 00:12	0.01	0.56	49.79	1.34	0.02	0.01	17.23	13.5	9.96	0.37	14.63
15	01-01-2019 00:13	0.01	0.58	49.21	1.36	0.01	0.01	17.68	15.67	10.12	0.37	14.6
16	01-01-2019 00:14	0.01	0.58	38.81	1.36	0.01	0	17.78	17.41	10.12	0.38	14.62
17	01-01-2019 00:15	0.02	0.58	2.33	15.16	0	0.02	17.75	17.4	10.1	0.38	14.59
18	01-01-2019 00:16	0.01	0.59	2.33	35.34	0.01	0.02	17.7	17.38	10.11	0.38	14.57
19	01-01-2019 00:17	0.02	0.58	2.31	34.69	0	0.03	17.68	17.27	10.11	0.38	14.62
20	01-01-2019 00:18	0.01	0.57	2.32	4.14	0.01	0	17.71	15.6	10.13	0.38	14.57

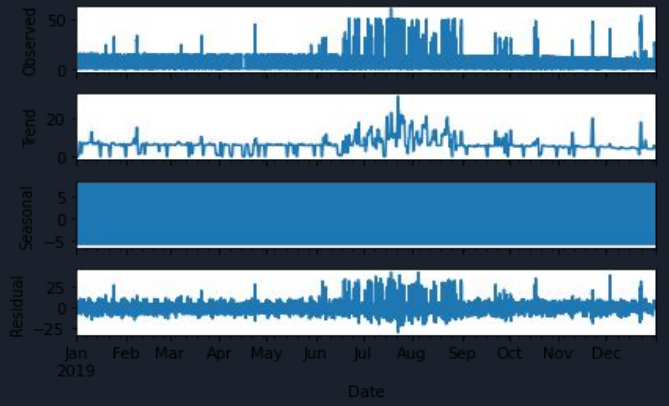
Exploratory Data Analysis

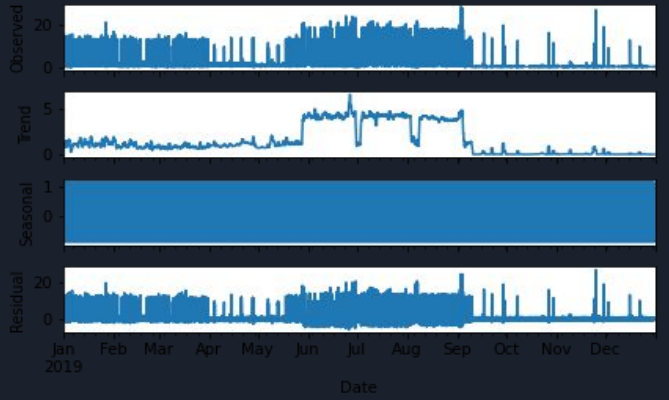
- Once the data has been imported from CSV to Dataframe, the first step is to make sure the data is free of ay null or NaN values.
- In order to handle Null values, I use an imputer object to calculate the mean of every column of the dataframe and fill in the null value elements with the column's mean value.
- Once the data is rid of all the Null values, a graph is plotted for every column of the dataframe against its Index Date column.
- From the visualization, it is very difficult to determine whether the entire data or any specific columns of the data has any Trend or Seasonality.
- Once the data is checked for Trend and Seasonality, we should also check for Stationarity in data using the Dickey-Fuller Test



Test For Trend and Seasonality In Data

- Its is really important to analyze the type of Trend and Seasonality that exist in the dataset that we have in order to perform Exponential Smoothing Methodology for Time Series Forecasting.
- The Trend and Seasonality of a data can be obtained by plotting the ETS Decomposition plot for all the columns of the data.
- From the ETS Decomposition plot, for one example column, I could clearly see the residual (E)rror, (T)rend and (S)easonality of that particular column.
- From the ETS decomposition of the data, we can see that the columns in this dataframe has a no visible Trend or Seasonality.
- Since there is an absence of Trend and Seasonality in the data, we can visually confirm that the data is stationary.





Test For Stationarity In Data

- As mentioned in the earlier slides, it is really difficult to determine whether the data is stationary or not by just manually looking at the data.
- To confirm that the data is stationary, we need to perform what is known as the Augmented Dickey Fuller Test.
- It is a statistical test, where the Null Hypothesis states there is a unit root for the given series.
- Like in any other statistical test, we're going to reject the Null Hypothesis if the p-value is less or equal to the significance level, which is typically 1%, 5% or 10%.
- For our time series to be stationary, the p-value has to be ≤ 0.05.
- Luckily for our case, all the columns of our data seems to be stationary from the results of the Dickey-Fuller test.

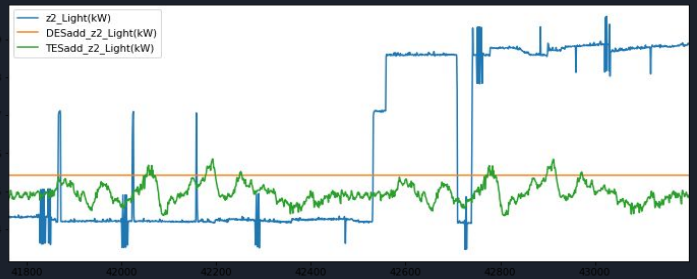
```
Augmented Dickey-Fuller Test: z1_Light(kW)
ADF test statistic     -1.037917e+01
p-value                 2.161542e-18
# lags used             5.100000e+01
# observations          8.634800e+04
critical value (1%)    -3.430426e+00
critical value (5%)    -2.861573e+00
critical value (10%)   -2.566788e+00
Strong evidence against the null hypothesis
Reject the null hypothesis
Data has no unit root and is stationary


Augmented Dickey-Fuller Test: z1_Plug(kW)
ADF test statistic        -26.567879
p-value                     0.000000
# lags used                66.000000
# observations          86333.000000
critical value (1%)        -3.430426
critical value (5%)        -2.861573
critical value (10%)       -2.566788
Strong evidence against the null hypothesis
Reject the null hypothesis
Data has no unit root and is stationary


Augmented Dickey-Fuller Test: z2_AC1(kW)
ADF test statistic     -1.217215e+01
p-value                 1.412758e-22
# lags used             6.400000e+01
# observations          8.633500e+04
critical value (1%)    -3.430426e+00
critical value (5%)    -2.861573e+00
critical value (10%)   -2.566788e+00
Strong evidence against the null hypothesis
Reject the null hypothesis
Data has no unit root and is stationary
```

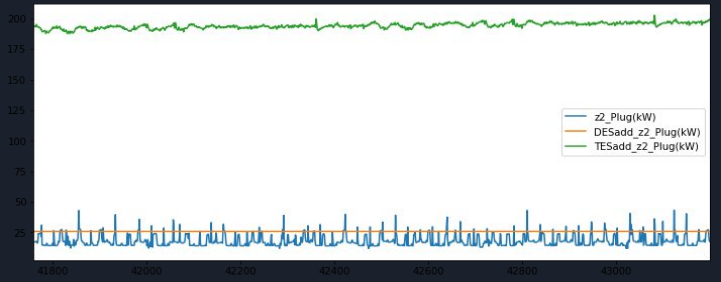
General Forecasting Methodology

- The method that we are going to follow to perform this experiment is pretty straight forward.
- The first step of this process is to read the data as a dataframe and perform data preprocessing in order to ensure that that are no NaN values in the data.
- Once the data has been cleaned and ready to be utilized, we perform a general projection of all the features of the data to see how the data trends over the time period.
- After performing necessary steps to determine which models to use, we will go ahead and train our data of various models and predict the values of our test data.
- The RMSE values of our models for the predicted data against the original test data will tell wus which model is the best performing model.
- After that, we can use the best performing model to train on the entire dataset and go ahead to forecast the values of the next 30 days in future.

Training And Prediction Using Statistical Models

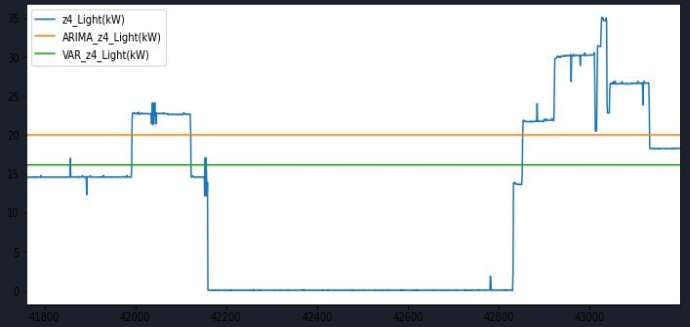
- After obtaining enough information about the dataset, I proceeded further to perform Model Training, Validating and Evaluation.
- Before starting to perform training, the data was split into Training set and Testing Set.
- As mentioned earlier, the first model that I chose to train on the dataset was Holt's Method or Double Exponential Smoothing.
- Once a model was trained using the DES for the first 11 months, the model was then used to predict the data for the next one month.
- Similarly, the next model I chose to train the data on was the Triple Exponential smoothing or the Holt-Winters Method.
- Once a model was trained using the TES for the first 11 months, the model was then used to predict the data for the next one month.





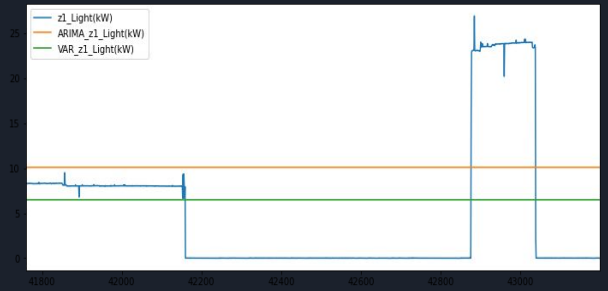
Training And Prediction Using Statistical Models

- Once the results of the simpler statistical model have been curated, the next step was to perform the same steps of training and predicting test data using more complex statistical model such as ARIMA and VAR.
- **ARIMA: Auto-regressive Integrated Moving Average**
- The ARIMA model can be broken down into three different components, each one with a parameter representing the characteristics of the time series. AR, I and MA.
- In order to pick the required parameters for ARIMA(p,d,q). We need to either perform ACF and PACF plotting which is a more difficult method to determine the p and q values. Or we can also perform Grid search or Auto Arima to determine them.
- We already know how many times we've had to difference the dataset, so the value of parameter d is 0.
- For our case, we will be using the Auto Arima to determine the optimum p, d and q values as we have multiple columns to predict.
- Once the optimum model for every column is determined by the Auto Arima, we use the model from Auto Arima that was trained on the Training set to predict the values for the next 30 days.



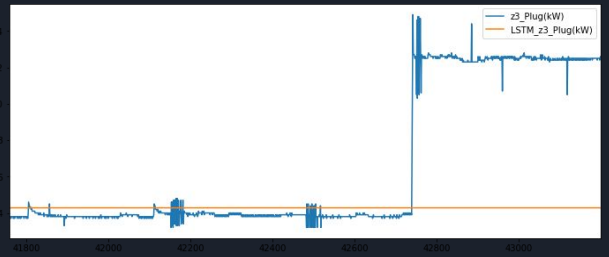
Training And Prediction Using Statistical Models

- Once the results of the simpler statistical model have been curated, the next step was to perform the same steps of training and predicting test data using more complex statistical model such as ARIMA and VAR.
- **VAR: Vector Auto-regression**
- The VAR model unlike the ARIMA model, has only the AR component to it.
- One huge advantage of using VAR model is because, VAR is used for Multivariate Time Series Analysis.
- In order to pick the required parameters for VAR(p). We need to perform Grid search by running different VAR models of various q values or orders..
- The optimum VAR model is chosen based on the AIC value for each model with respect to other models.
- Lower the AIC value, better fitted the model is to the data.
- Once the optimum model for the dataset is determined by the Grid Search, we use the model with that particular order and train it on the training set and predict the values for the next 30 days.



Training And Prediction Using RNN

- Neural networks like Long Short-Term Memory (LSTM) recurrent neural networks are able to almost seamlessly model problems with multiple input variables.
- This is a great benefit in time series forecasting, where classical linear methods can be difficult to adapt to multivariate or multiple input forecasting problems.
- The steps involved in performing Multivariate Time Series Analysis using LSTM are,
  - Scaling the Training and testing Data.
  - Deciding an apt input length ad batch size for the model the predict.
  - Building a simple model with LSTM along with proper regularization set up to avoid overfitting of data.
  - Setting up Early Stoppings and Call Backs to stop the model from training at the right moment to avoid overfitting.
  - Save the model.
- Once the optimum model for is determined by the LSTM Network, we use the model that was trained on the Training set to predict the values for the next 30 days.



Test Data Prediction Results

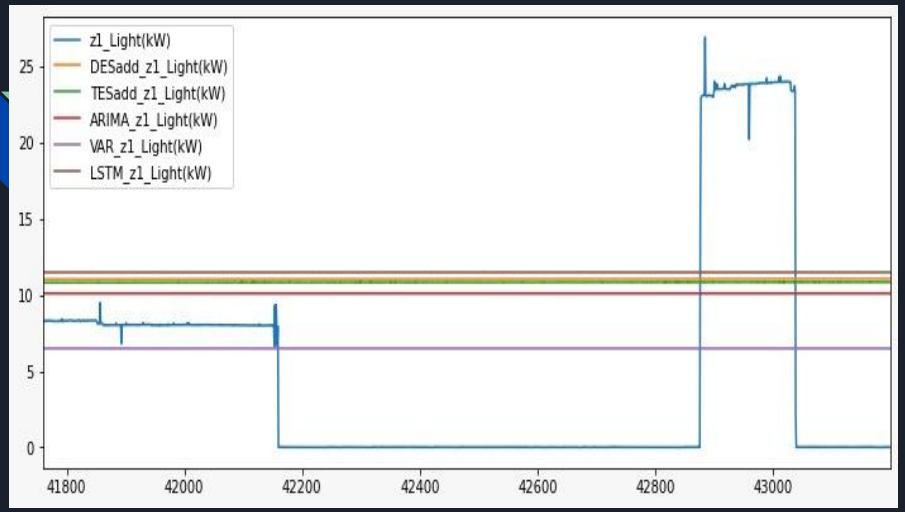
- In order to determine which model performed the best, the RMSE (Root Mean Squared Error) evaluation metric is used for the predicted values against the original test data.
- RMSE is a standard and a reliable method that will stay closed to the real MSE when the error is small and will also punish more when the error is large.

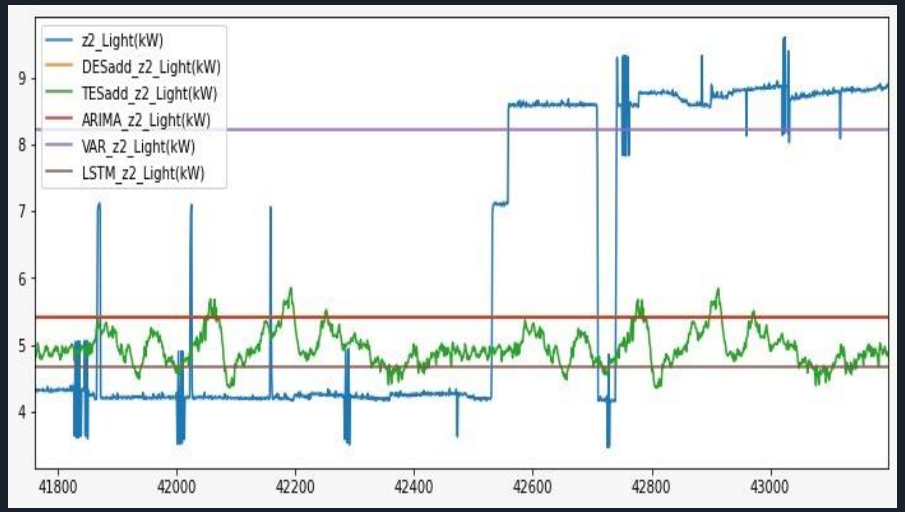
$$RMSE = \sqrt{\frac{1}{L}\sum_{l=1}^{L}(y_{T+l} - \hat{y}_{T+l})^2}$$

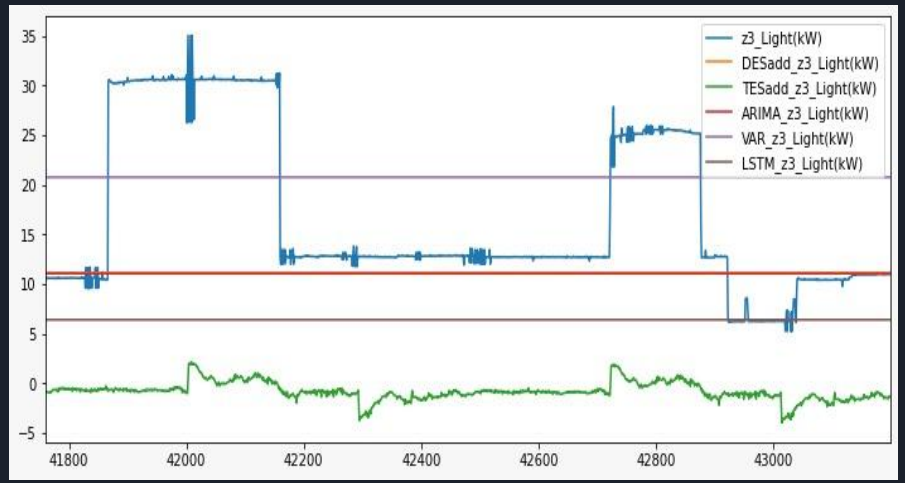
where T is the last observation period and l is the lag.

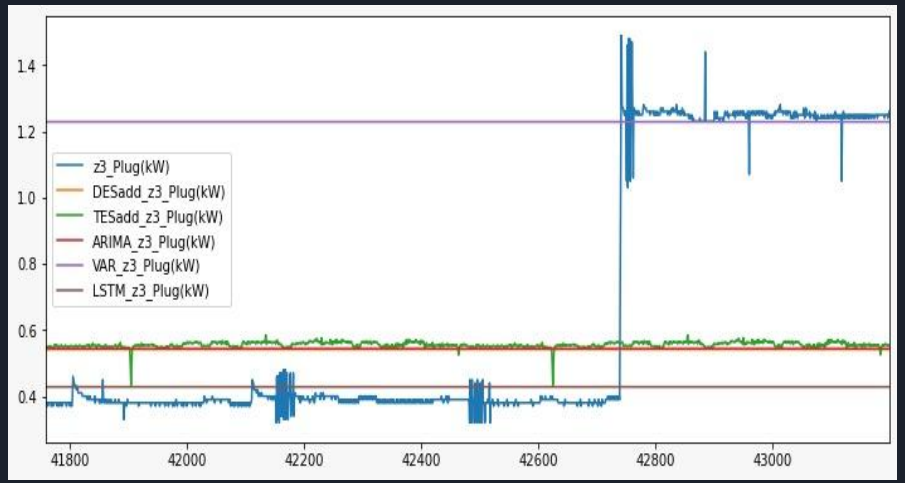
- This property of RMSE ensures that we do not miss when the model under performs and this is the reason why RMSE is popular.
- Since their performance was similar, I decided to finally choose VAR as the winner for our particular case, as VAR can take input as a Vector and perform multivariate TSA and predict the future values.

```
For Column z1_Light(kW):
DES RMSE: 8.448
TES RMSE: 8.378
ARIMA RMSE: 8.140
VAR RMSE: 6.497
LSTM RMSE: 9.074


For Column z1_Plug(kW):
DES RMSE: 0.394
TES RMSE: 6.663
ARIMA RMSE: 0.394
VAR RMSE: 1.685
LSTM RMSE: 3.073


For Column z2_AC1(kW):
DES RMSE: 35.652
TES RMSE: 8560.173
ARIMA RMSE: 23.110
VAR RMSE: 22.354
LSTM RMSE: 21.872


For Column z2_AC2(kW):
DES RMSE: 6.561
TES RMSE: 84.990
ARIMA RMSE: 6.560
VAR RMSE: 7.795
LSTM RMSE: 6.652
```


z1_Light(kW)
DESadd_z1_Light(kW)
TESadd_z1_Light(kW)
ARIMA_z1_Light(kW)
VAR_z1_Light(kW)
LSTM_z1_Light(kW)
25
20
15
10
5
0
41800
42000
42200
42400
42600
42800
43000

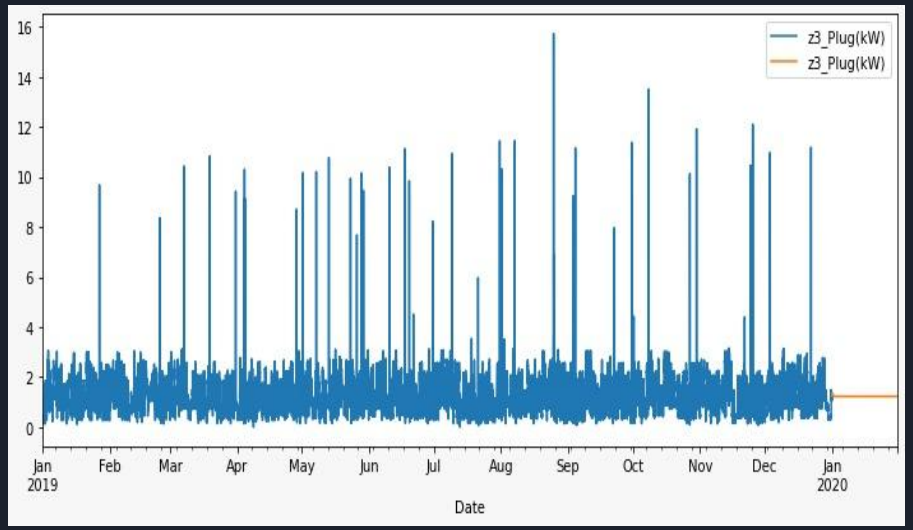

z2_Light(kW)
DESadd_z2_Light(kW)
TESadd_z2_Light(kW)
ARIMA_z2_Light(kW)
VAR_z2_Light(kW)
LSTM_z2_Light(kW)
9
8
7
6
5
4
41800
42000
42200
42400
42600
42800
43000

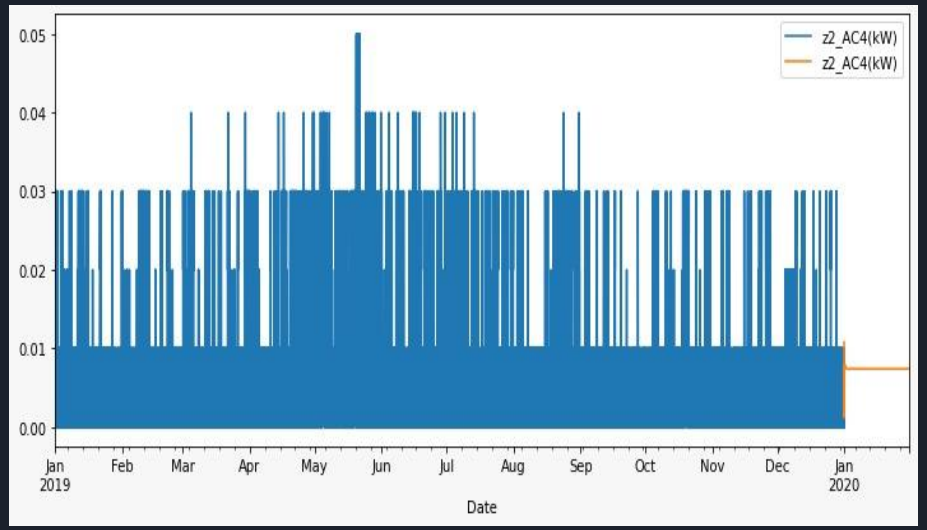

z3_Light(kW)
DESadd_z3_Light(kW)
TESadd_z3_Light(kW)
ARIMA_z3_Light(kW)
VAR_z3_Light(kW)
LSTM_z3_Light(kW)
35
30
25
20
15
10
5
0
−5
41800
42000
42200
42400
42600
42800
43000

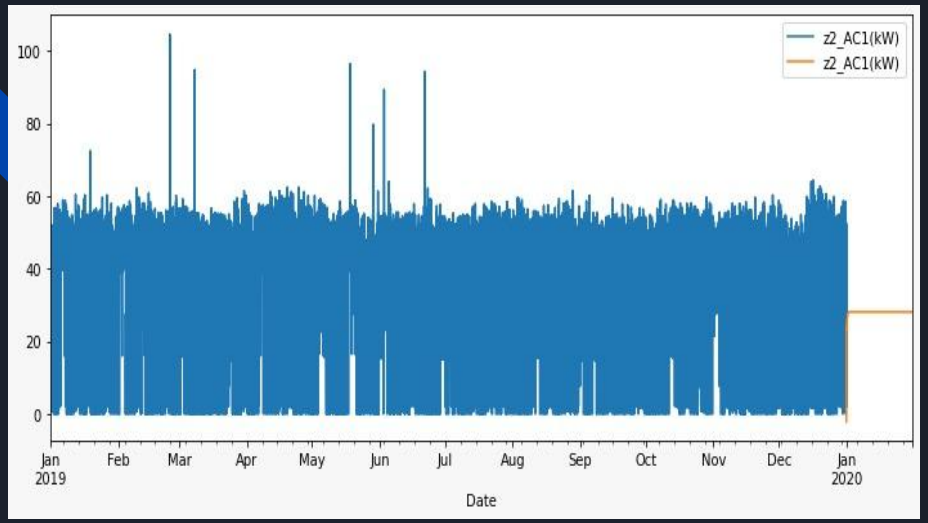

z3_Plug(kW)
DESadd_z3_Plug(kW)
TESadd_z3_Plug(kW)
ARIMA_z3_Plug(kW)
VAR_z3_Plug(kW)
LSTM_z3_Plug(kW)
1.4
1.2
1.0
0.8
0.6
0.4
41800
42000
42200
42400
42600
42800
43000

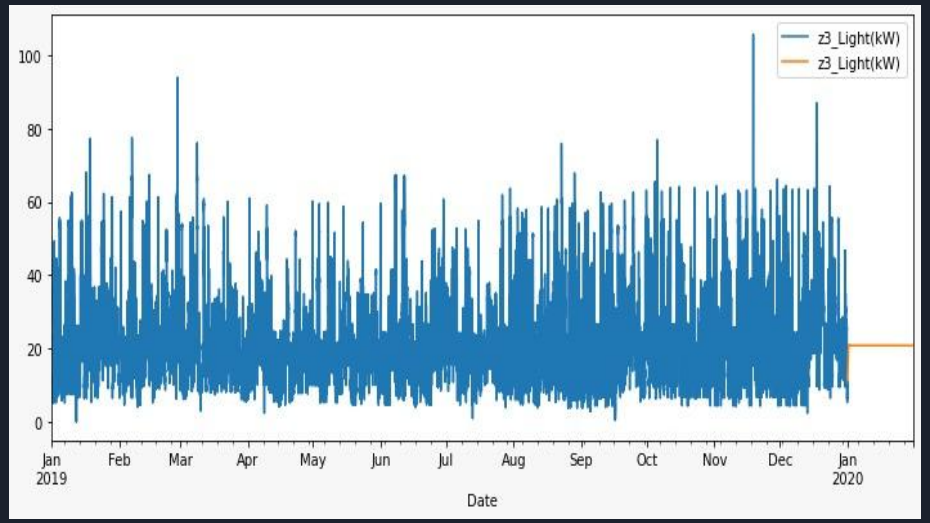
Final Data Projection

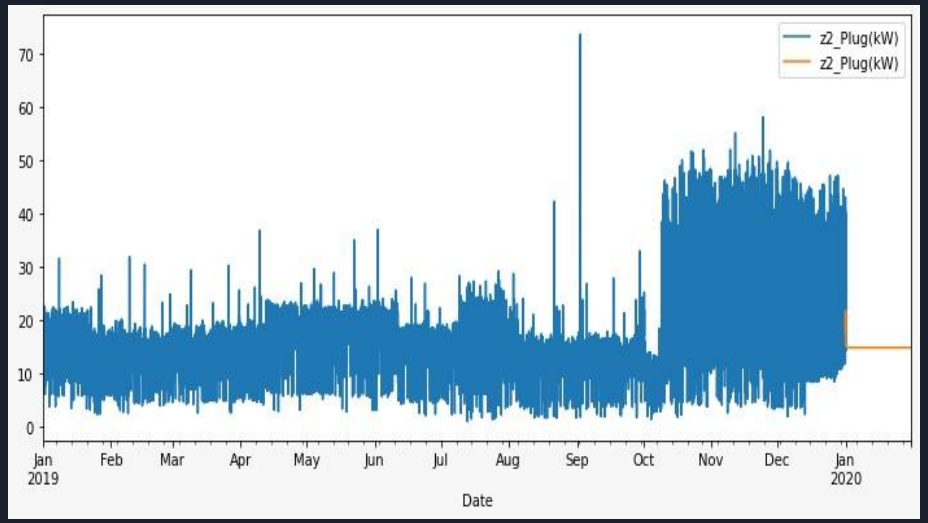
- From our experimentation on Time Series analysis using Simple Statistical models such as DES, TES, Complex Statistical Models such as ARIMA, VAR and finally LSTM based Multivariate Time Series Analysis, I have concluded that for the dataset of First Floor sensor data of the Smart Building for the year 2019, VAR(15) model predicts the data with the least RMSE values.
- As VAR(15) is the best model from this experiment, I went ahead and trained the entire dataset for the year 2019 starting from 01-01-2019 to 12-31-2019.
- Once the model was trained, now I performed forecasting of the future data for the next one one month, from 01-01-2020 to 01-31-2020.

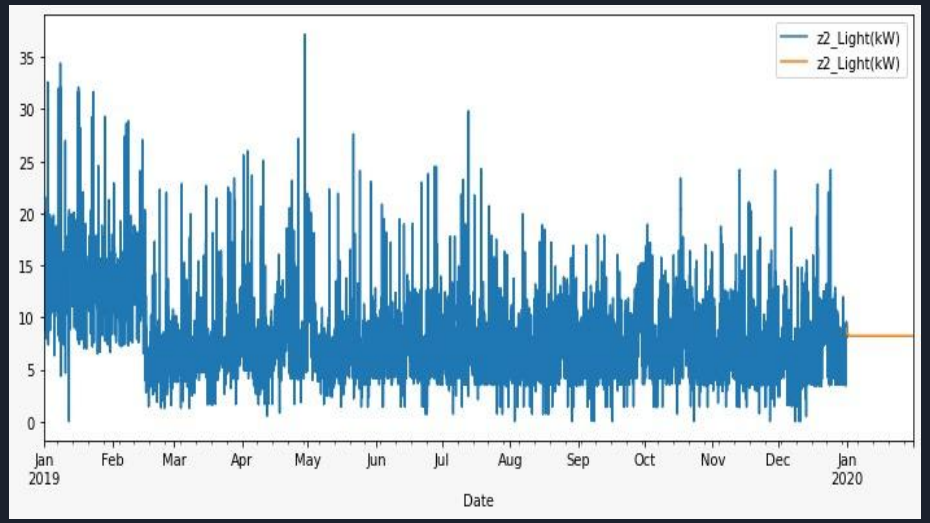





z2_AC1(kW)
z2_AC1(kW)
100
80
60
40
20
0
Jan
2019
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2020
Date


z3_Light(kW)
z3_Light(kW)
100
80
60
40
20
0
Jan
2019
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2020
Date


z2_Plug(kW)
z2_Plug(kW)
70
60
50
40
30
20
10
0
Jan
2019
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2020
Date


z2_Light(kW)
z2_Light(kW)
35
30
25
20
15
10
5
0
Jan
2019
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
2020
Date

Inference From The Forecasting

- We looked at the effectiveness of two methods for making time series forecasts.
- Since our data was smaller, it might have performed much better in statistical methods such as ARIMA and VAR
- LSTM might have performed much better than this, if the data that I had was spanning over a longer period of time.
- LSTM might have performed much better than this, if I had a much deeper architecture with more LSTM layers.
- But the main disadvantages of having a larger or Deeper network is the training time of this model might be really huge and very resource consuming.
- If a traditional statistical model like ARIMA, VARMA, or VAR can perform much better or same as the LSTM with lesser training time and with lesser resource requirement, then statistical models are better than LSTM for such cases.

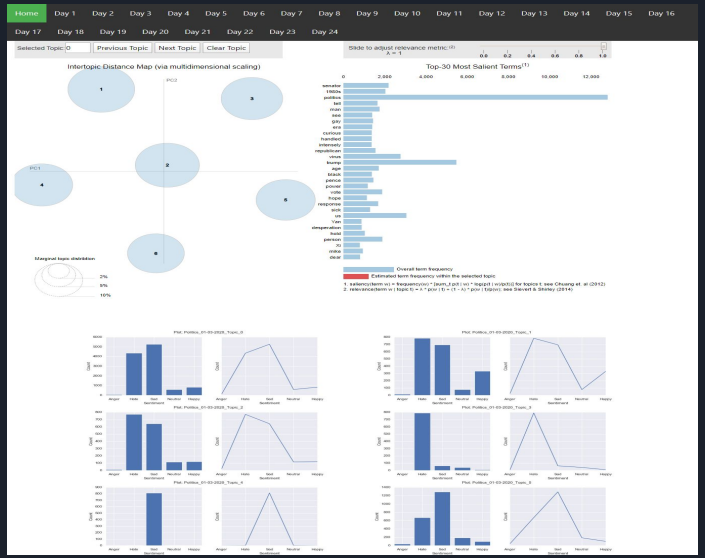
Future Works To Improve Forecasting

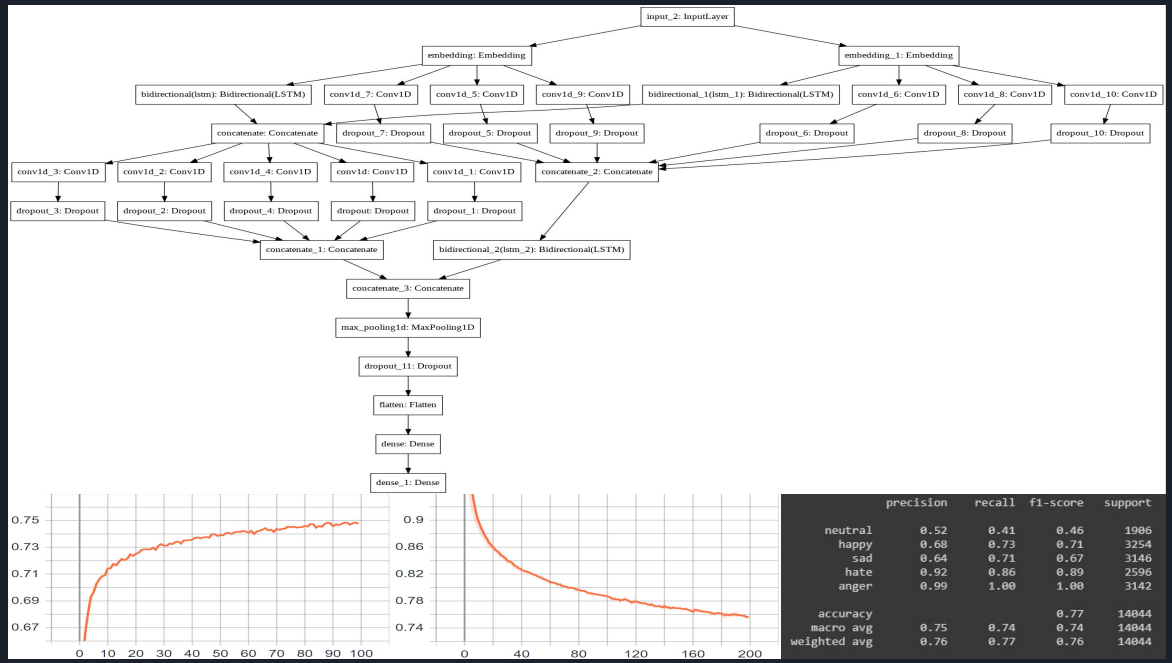
- In my current experiment, I could clearly see that powerful statistical models perform much better than the models based on neural networks.
- As mentioned in our earlier slide, this is mainly because LSTM are a black box and statistical model are more of a straight forward formulae and can work on smaller data muh better.
- But the LSTMs still have a potential to learn much better and the statistical model such as VAR, can do much better than just correctly detecting the trend with a little help.
- This can be done by clubbing both VAR and LSTM together or ARIMA and LSTM together.
- Our workflow can be summarized as follow:
    - Estimate a VAR/ARIMA properly on our training data
    - Extract what VAR/ARIMA has learned and use it to improve the training process of an LSTM model performing a two-step training.

Other Deep Learning Projects

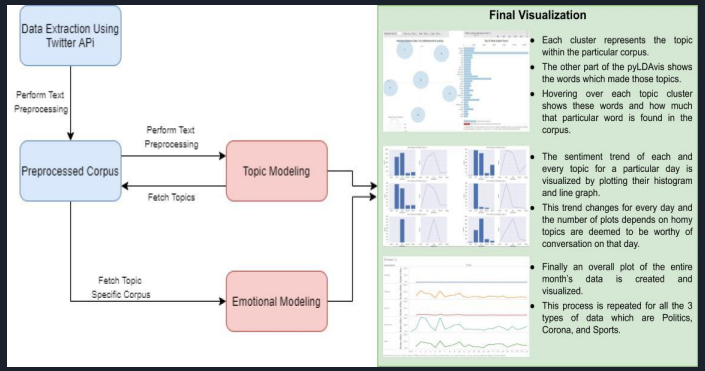
Graduate Project: Analyzing Conversational Traits In Social Media With Topic Modeling And Emotional Modeling

- Given a large collection of tweets from Twitter, their topics and their emotions can be analyzed using separate topic & emotion models. Genres of interests are : 'Politics', 'Sports', and 'Corona'.
- Topic modeling is performed using Latent Dirichlet Allocation (LDA) and Emotion Modeling is performed with the assistance of Deep Learning using LSTM-CNN.



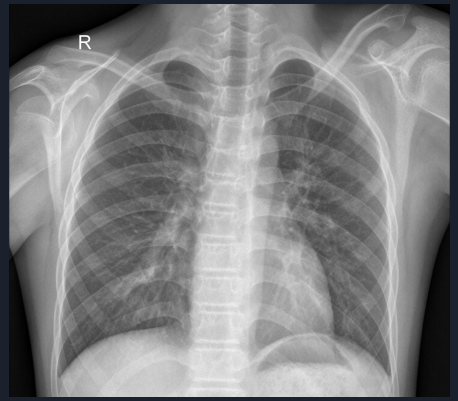


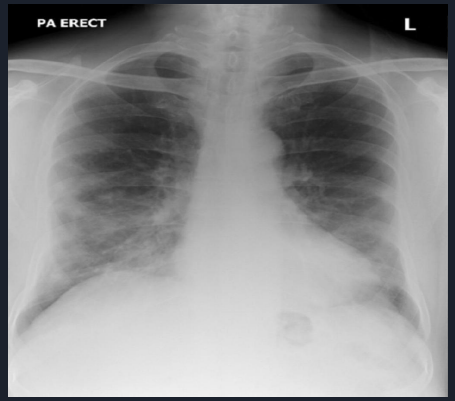
	precision	recall	f1-score	support
neutral	0.52	0.41	0.46	1906
happy	0.68	0.73	0.71	3254
sad	0.64	0.71	0.67	3146
hate	0.92	0.86	0.89	2596
anger	0.99	1.00	1.00	3142
accuracy			0.77	14044
macro avg	0.75	0.74	0.74	14044
weighted avg	0.76	0.77	0.76	14044

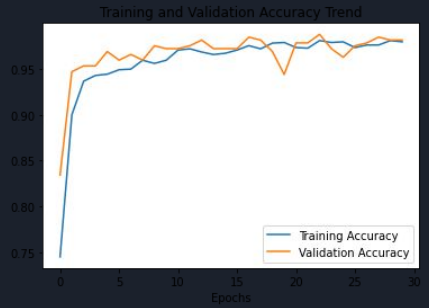


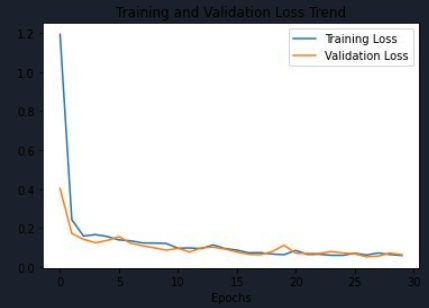
Covid19 Pneumonia Classification Using Deep Learning

- This is a personal project that involves creating a Deep Learning Model using CNN to analyze a chest X-Ray and classify it as Covid Positive or Covid Negative.
- A model was trained using CNN architecture.



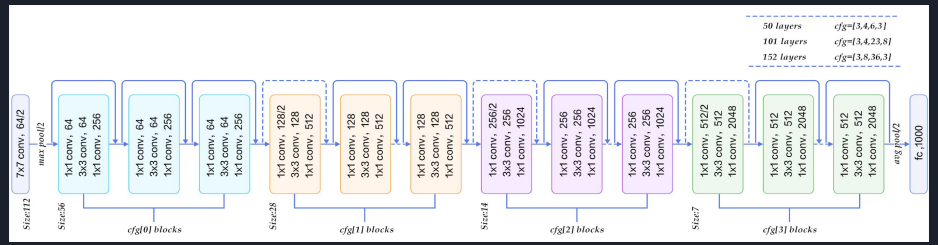


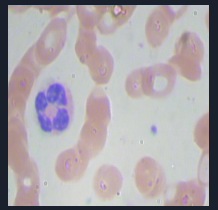


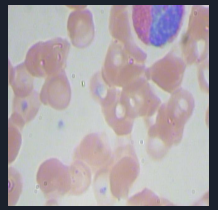


CV Model To Classify Blood Cells Using RESNET

- The task is to use computer vision techniques to classify different blood cells. The dataset for this task consists of 4 different types of blood cell images.
- A model using the RESNET architecture was trained using the pre trained weights of the RESNET architecture using IMAGENET data.
- This model was trained by removing the top part of the RESNET architecture and by using Transfer Learning, we use the pretrained RESNET weights and a custom dense layer in the top to perform prediction.







Thank You