Data Science Specialization

Academic Year 2019 - 2023

Advanced Data Analytics Project

Numerical Weather Prediction Model Forecasts of Extreme Air Temperatures

Exploratory Data Analysis and Comparative Analysis on Various Machine Learning Techniques

Group: 6
Members:
Preethi G - S20190020241
Subash J - S20190020253

1 Project Description

Forecasting Air Temperature at 2 meters above land surface helps in preparing for a potential weather disaster like heat waves (maximum daytime air temperature) and cold spells (minimum nighttime air temperature). The increasing intensity, duration and frequency of the extreme air temperatures during the summer season suggests that an accurate air temperature forecasting is essential. So, our project is to explore the given data and do a comparative model analysis on various regression models to see which model best predicts the Air Temperature for the given data. We labelled the data to be 'normal' & 'hot' temperatures using a threshold value to get a perspective on the given problem with a classification based approach as well.

2 Description of the Data

Data Set Information:
The data set is a multivariate data set that contains a total of 25 attributes including station, date, fourteen numerical weather prediction (NWP)'s meteorological forecast data, four in-situ observations and five geographical auxiliary variables over Seoul, South Korea in the summer with 7750 instances.

Attribute Information:

No.	Attribute	Description
1	station	used weather station number: 1 to 25
2	Date	Present day: yyyy-mm-dd (2013-06-30 to 2017-08-30)
3	Present_Tmax	Maximum air temperature between 0 and 21 h on the present-day
4	Present_Tmin	Minimum air temperature between 0 and 21 h on the present day
5	LDAPS_RHmin	Forecast of next-day minimum relative humidity
6	LDAPS_RHmax	Forecast of next-day maximum relative humidity
7	LDAPS_Tmax	Forecast of next-day maximum air temperature applied lapse rate
8	LDAPS_Tmin	Forecast of next-day minimum air temperature applied lapse rate
9	LDAPS_WS	Forecast of next-day avg wind speed
10	LDAPS_LH	Forecast of next-day avg latent heat flux
11	LDAPS_CC1	Forecast of next-day 1st 6-hour split avg cloud cover (0-5 h)
12	LDAPS_CC2	Forecast of next-day 2nd 6-hour split avg cloud cover (6-11 h)
13	LDAPS_CC3	Forecast of next-day 3rd 6-hour split avg cloud cover (12-17 h)
14	LDAPS_CC4	Forecast of next-day 4th 6-hour split avg cloud cover (18-23 h)
15	LDAPS_PPT1	Forecast of next-day 1st 6-hour split avg precipitation (0-5 h)
16	LDAPS_PPT2	Forecast of next-day 2nd 6-hour split avg precipitation (6-11 h)
17	LDAPS_PPT3	Forecast of next-day 3rd 6-hour split avg precipitation (12-17 h)
18	LDAPS_PPT4	Forecast of next-day 4th 6-hour split avg precipitation (18-23 h)
19	lat	Latitude
20	lon	Longitude
21	DEM	Elevation
22	Slope	Slope
23	Solar radiation	Daily incoming solar radiation
24	Next_Tmax	The next-day maximum air temperature
25	Next_Tmin	The next-day minimum air temperature

3 Methodology

3.1 Visualizing and Pre-processing the Data set

- The Data set is read and the data column is formatted. The data set is checked for any missing data points so as to delete those rows or replace the NaN values with mean/median values of respective columns.

```
station              2
Date                 2
Present_Tmax        70
Present_Tmin        70
LDAPS_RHmin         75
LDAPS_RHmax         75
LDAPS_Tmax_lapse    75
LDAPS_Tmin_lapse    75
LDAPS_WS            75
LDAPS_LH            75
LDAPS_CC1           75
LDAPS_CC2           75
LDAPS_CC3           75
LDAPS_CC4           75
LDAPS_PPT1          75
LDAPS_PPT2          75
LDAPS_PPT3          75
LDAPS_PPT4          75
lat                  0
lon                  0
DEM                  0
Slope                0
Solar radiation      0
Next_Tmax           27
Next_Tmin           27
dtype: int64
```

- The Null values in attributes station and date cannot be replaced so the rows are dropped. We get a detailed description about the count, mean, standard deviation and 5 number summary using the describe function.

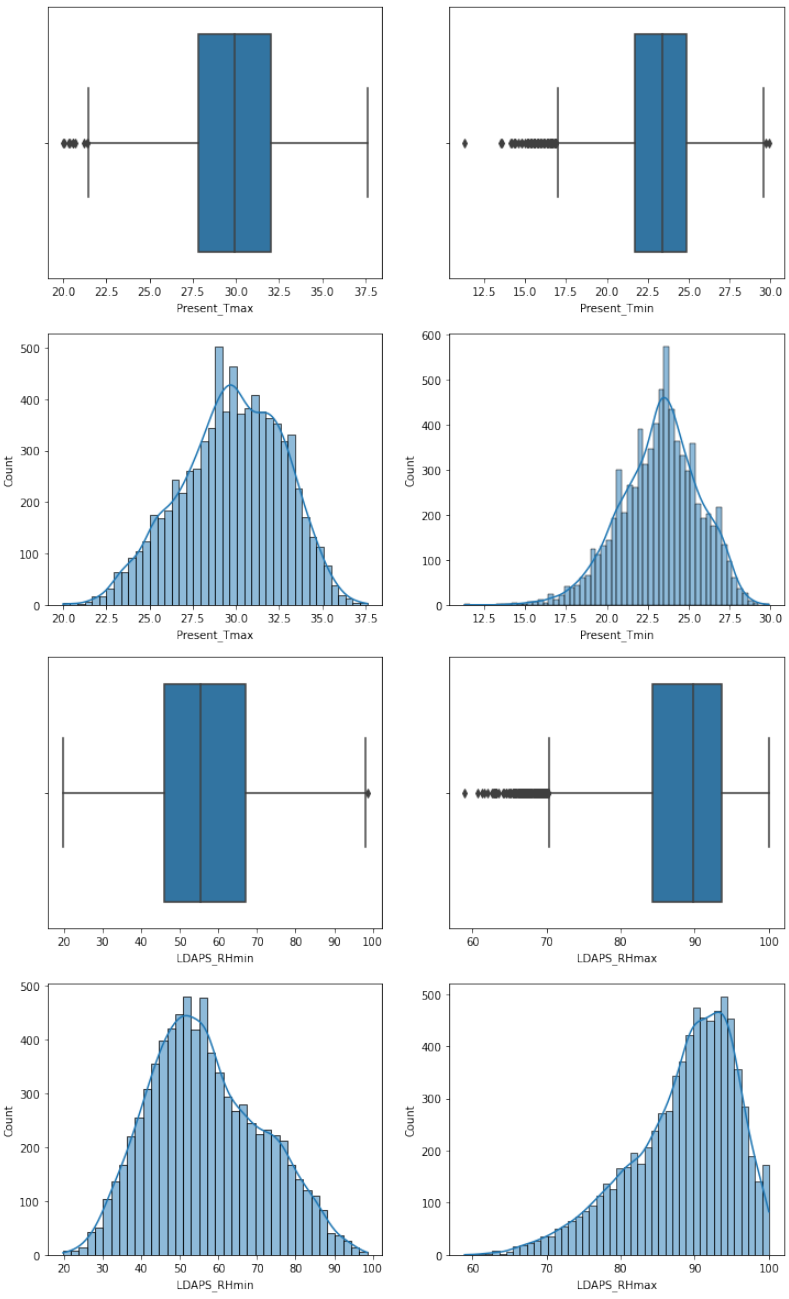
- To replace the NaN values present in the fourteen numerical weather prediction (NWP)'s meteorological forecast columns and the four in-situ observation columns, we first check for the Distribution of the data. If the data is skewed then we replace the NaN with median of the attribute and if the data is symmetric then replacement is done with mean of the attribute.
  1) Skewness between -0.5 to 0.5 indicates the data is fairly symmetrical
  2) Skewness between -1 to -0.5 or between 0.5 to 1 indicates the data is moderately skewed
  3) Skewness less than -1 and greater than 1 indicates the data is fairly symmetrical

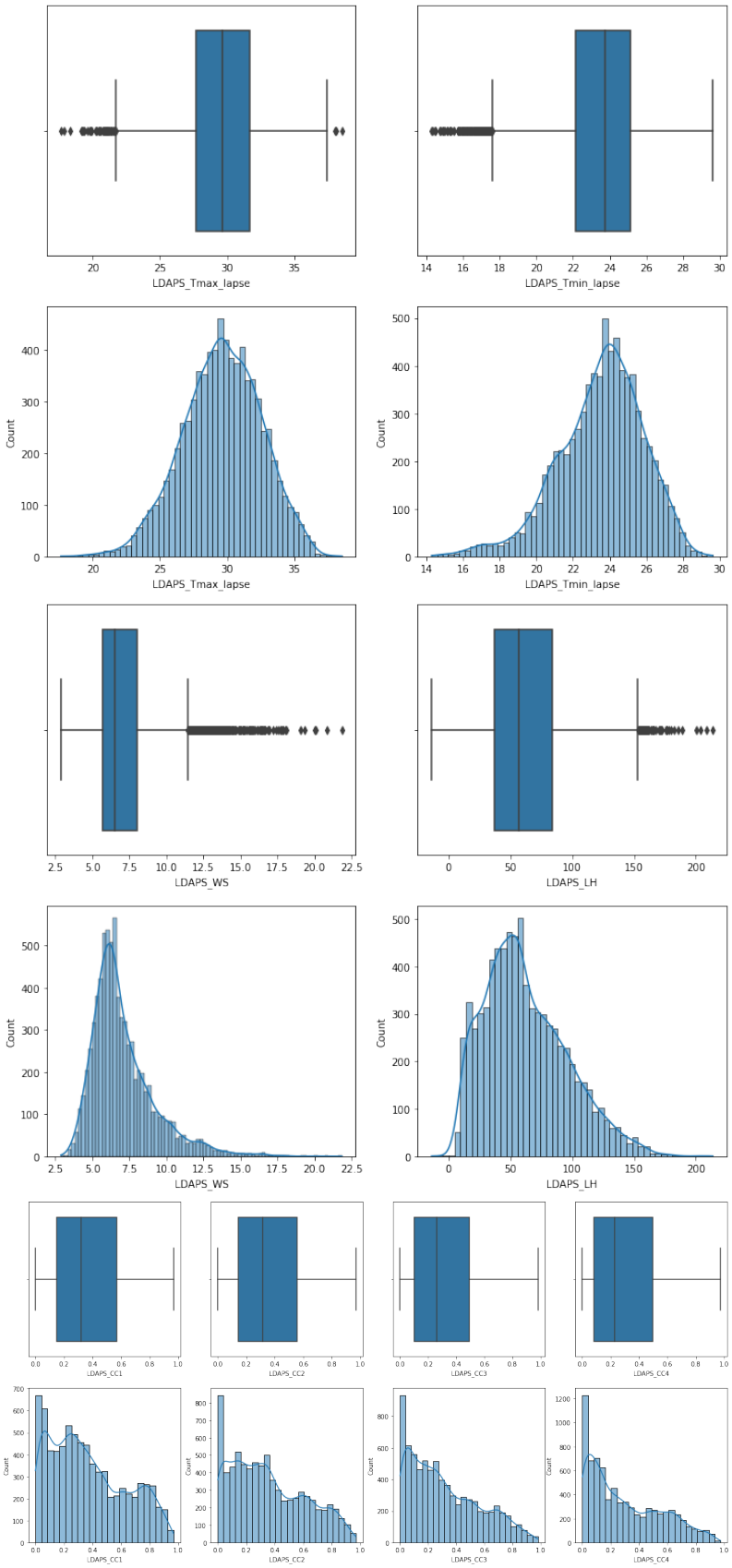
```
[-0.26177432 -0.35588523  0.29871612 -0.84637628 -0.22308025 -0.57611811
  1.54859124  0.6642241   0.45688054  0.4695883   0.63686565  0.6624763
  5.29759727  5.68447911  6.33662583  6.70927537 -0.33530714 -0.39610718]
```

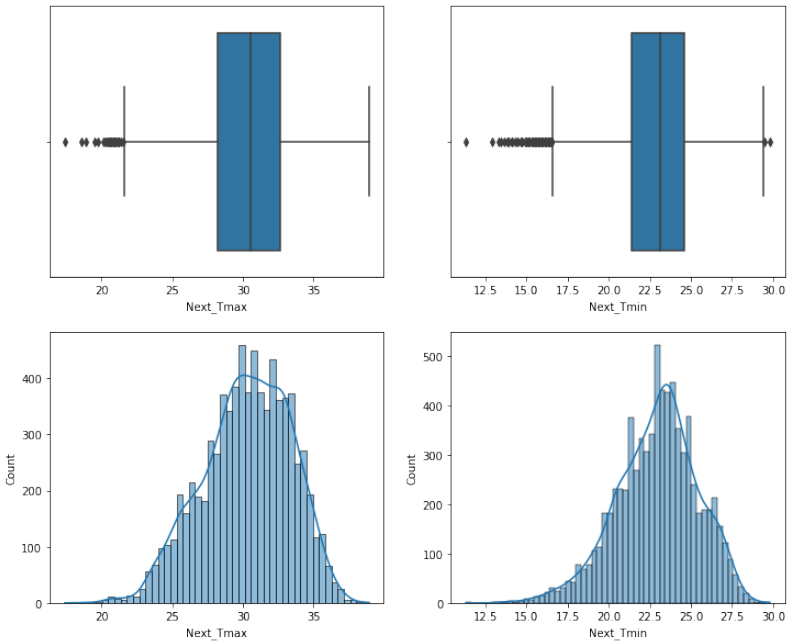
- After replacing we check for null values to make sure all the attributes are free of missing values. We again describe the dataframe and make sure that the replacement doesn't affect the statistical parameters much.

- The correlation for the Data Frame is calculated and the pair-plots are plotted.

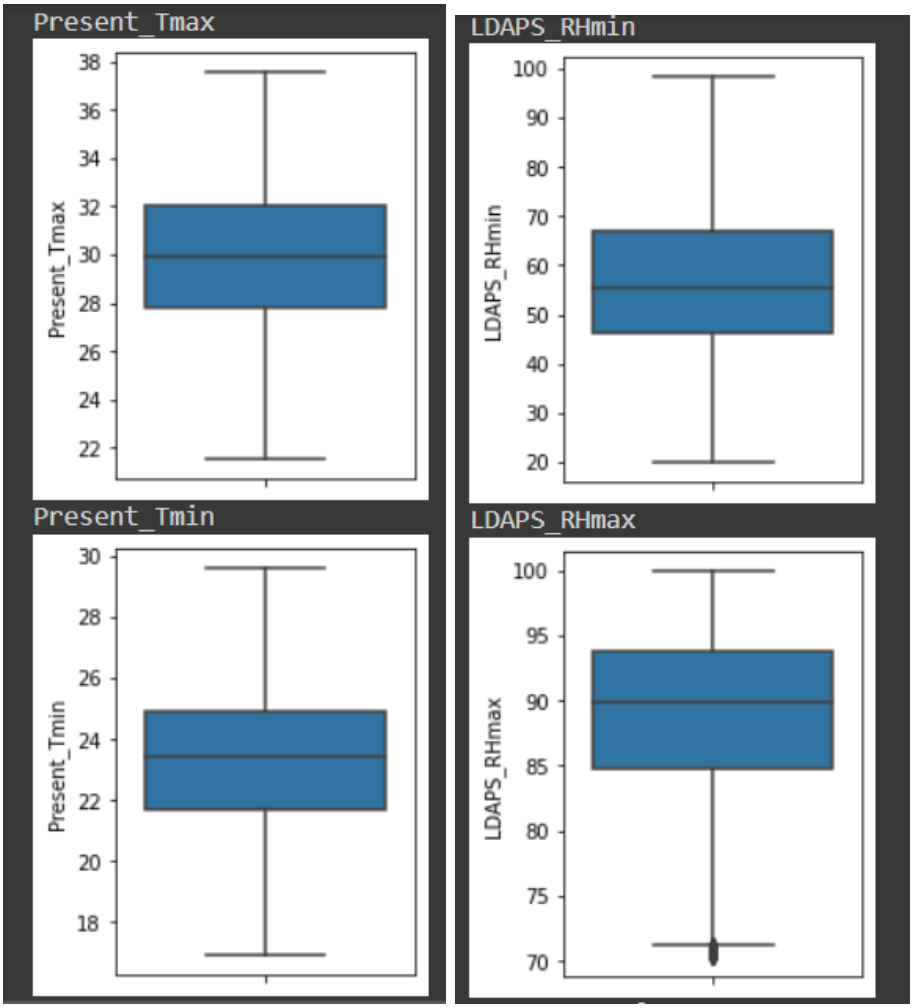
- Boxplot and Histplot of the in-situ and LDAPS (except precipitation) is plotted to check for outliers.

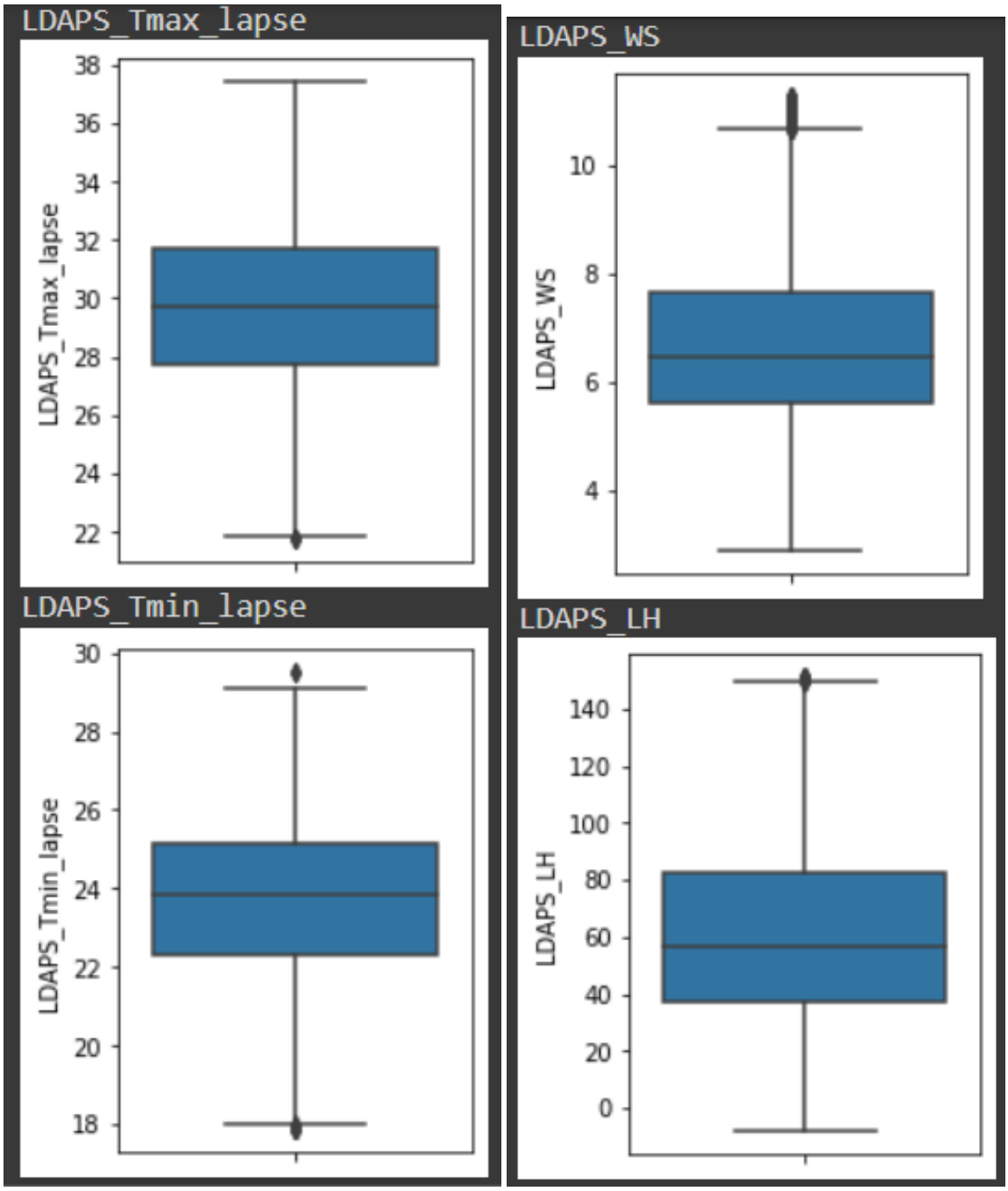


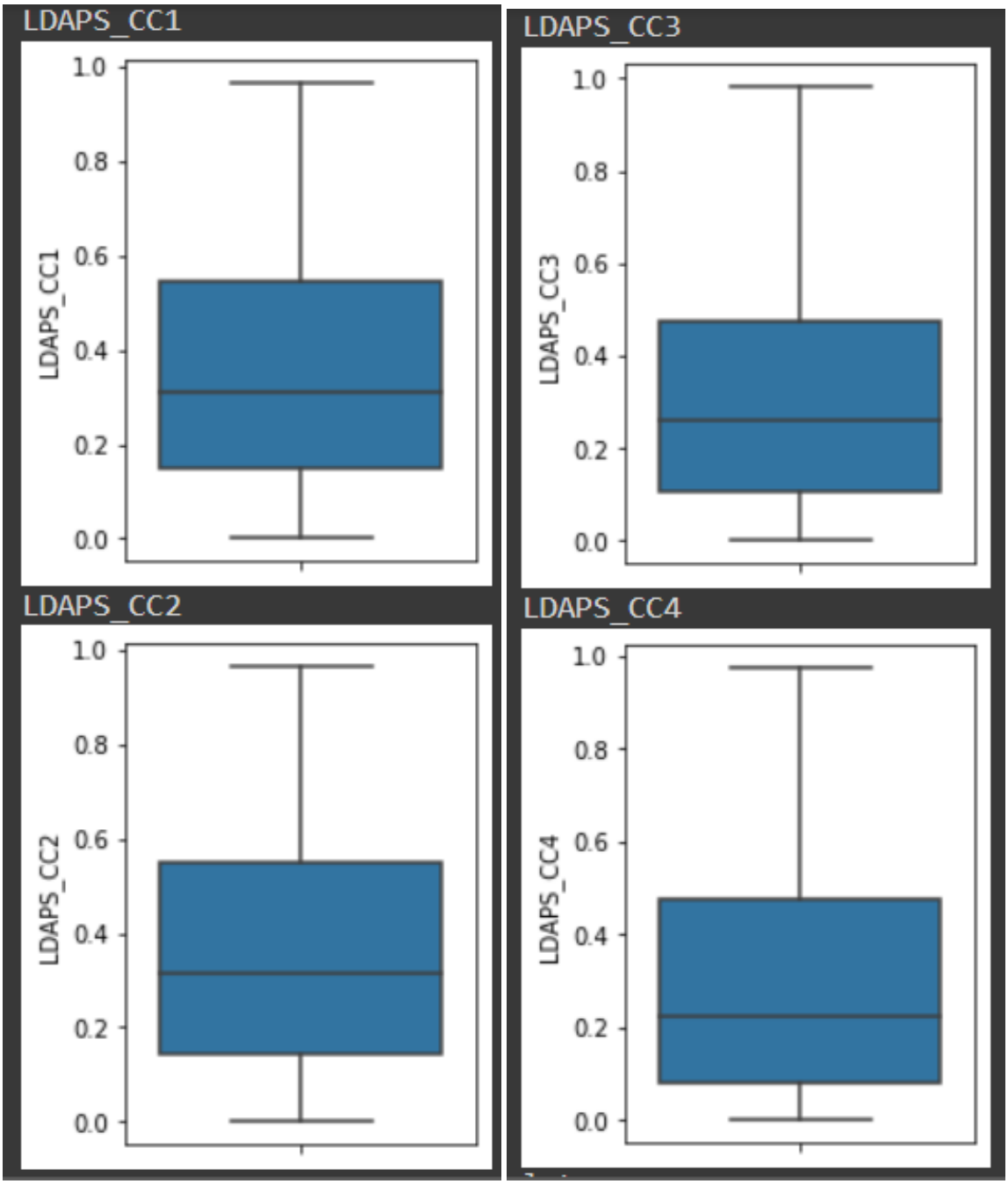

LDAPS_Tmax_lapse
LDAPS_Tmin_lapse
Count
LDAPS_Tmax_lapse
Count
LDAPS_Tmin_lapse
LDAPS_WS
LDAPS_LH
Count
LDAPS_WS
Count
LDAPS_LH
LDAPS_CC1
LDAPS_CC2
LDAPS_CC3
LDAPS_CC4
Count
LDAPS_CC1
Count
LDAPS_CC2
Count
LDAPS_CC3
Count
LDAPS_CC4

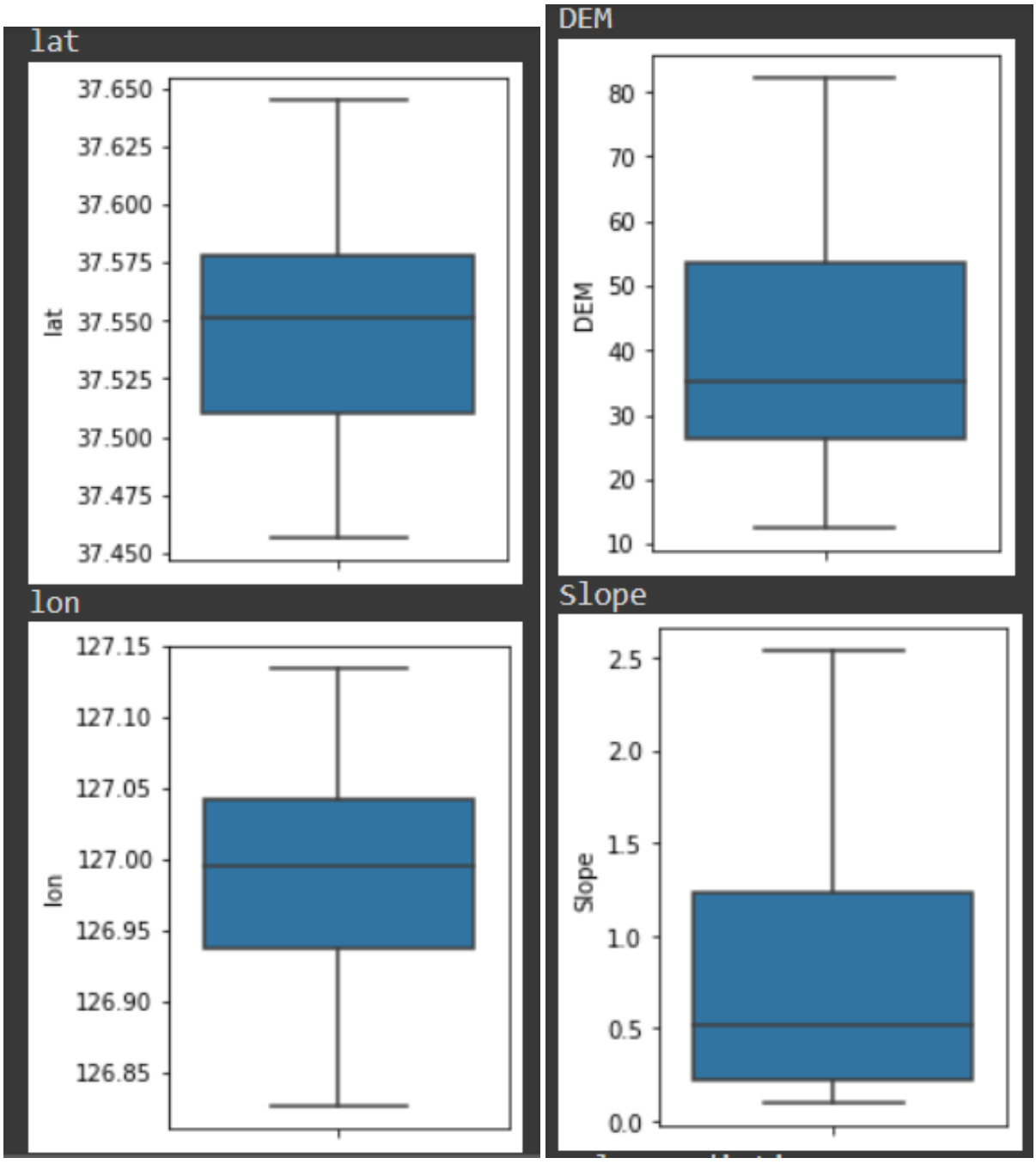


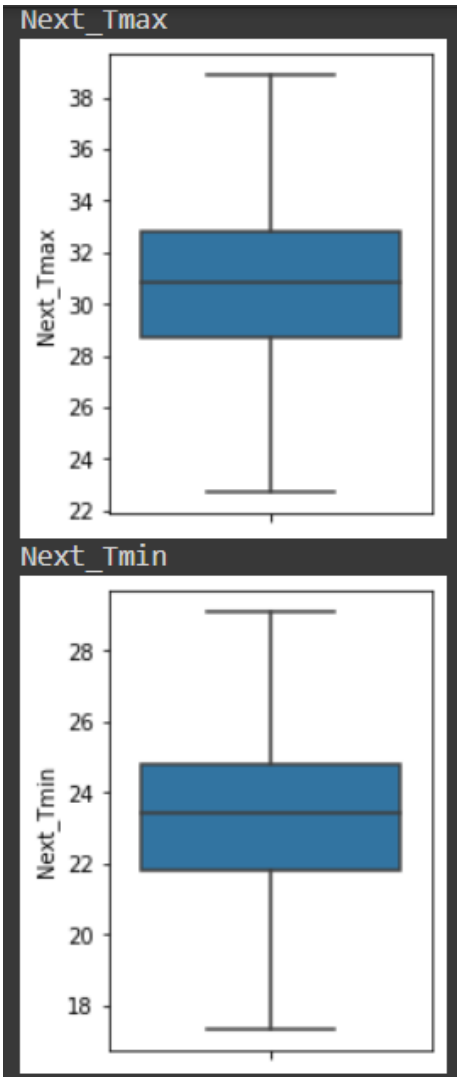
- The outliers are then removed and this can be again observed by the box plots.



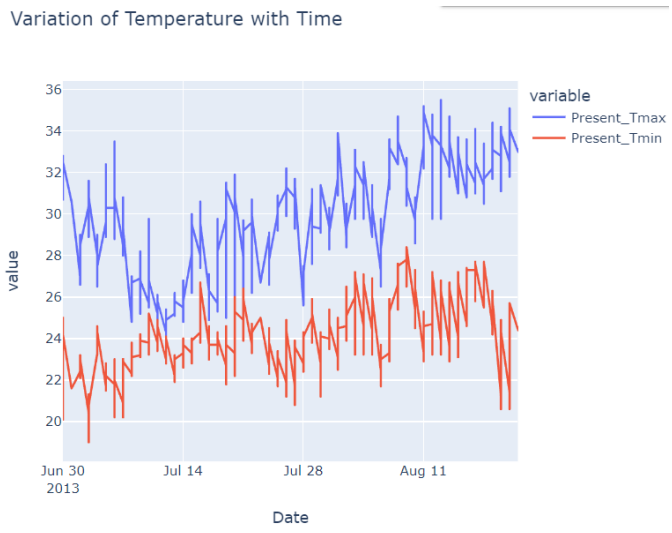

LDAPS_Tmax_lapse
LDAPS_Tmax_lapse
38
36
34
32
30
28
26
24
22
LDAPS_WS
LDAPS_WS
10
8
6
4
LDAPS_Tmin_lapse
LDAPS_Tmin_lapse
30
28
26
24
22
20
18
LDAPS_LH
LDAPS_LH
140
120
100
80
60
40
20
0


LDAPS_CC1
LDAPS_CC1
1.0
0.8
0.6
0.4
0.2
0.0
LDAPS_CC3
LDAPS_CC3
1.0
0.8
0.6
0.4
0.2
0.0
LDAPS_CC2
LDAPS_CC2
1.0
0.8
0.6
0.4
0.2
0.0
LDAPS_CC4
LDAPS_CC4
1.0
0.8
0.6
0.4
0.2
0.0

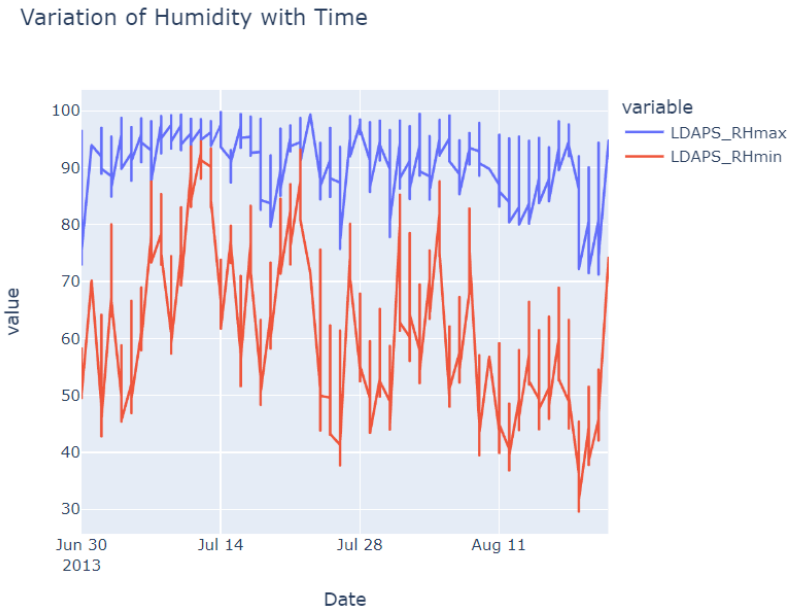

lat
37.650
37.625
37.600
37.575
37.550
37.525
37.500
37.475
37.450
lat
DEM
80
70
60
50
40
30
20
10
DEM
lon
127.15
127.10
127.05
127.00
126.95
126.90
126.85
lon
Slope
2.5
2.0
1.5
1.0
0.5
0.0
Slope



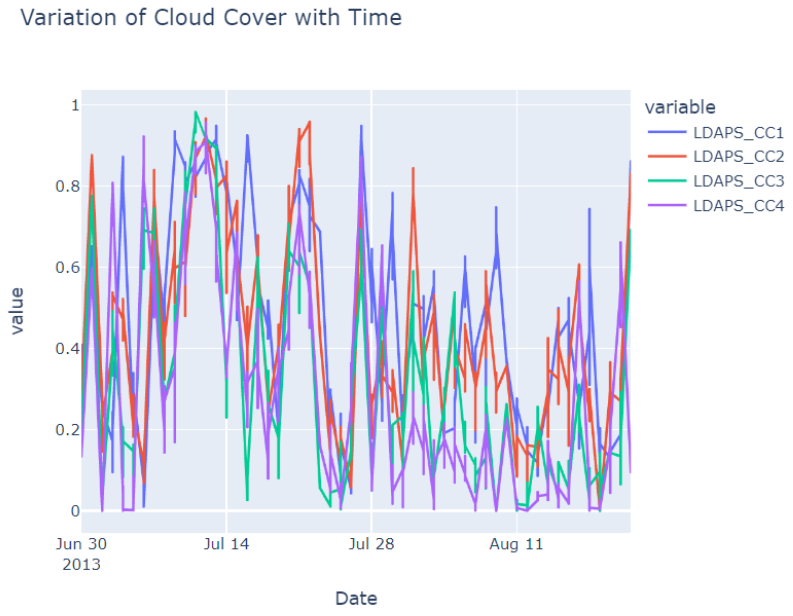
- Followed by the removal of outliers, We plotted graphs of Variation of:
  1) Temperature vs. Time



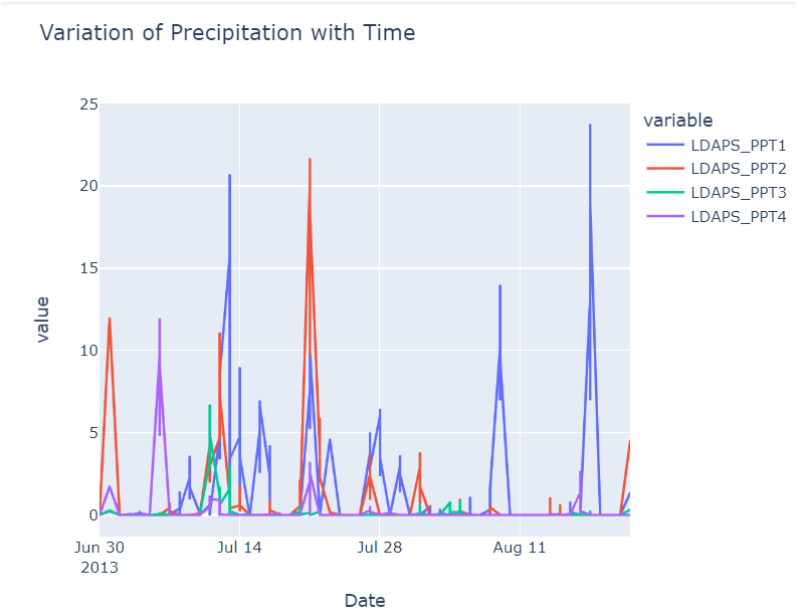
2) Humidity vs. Time



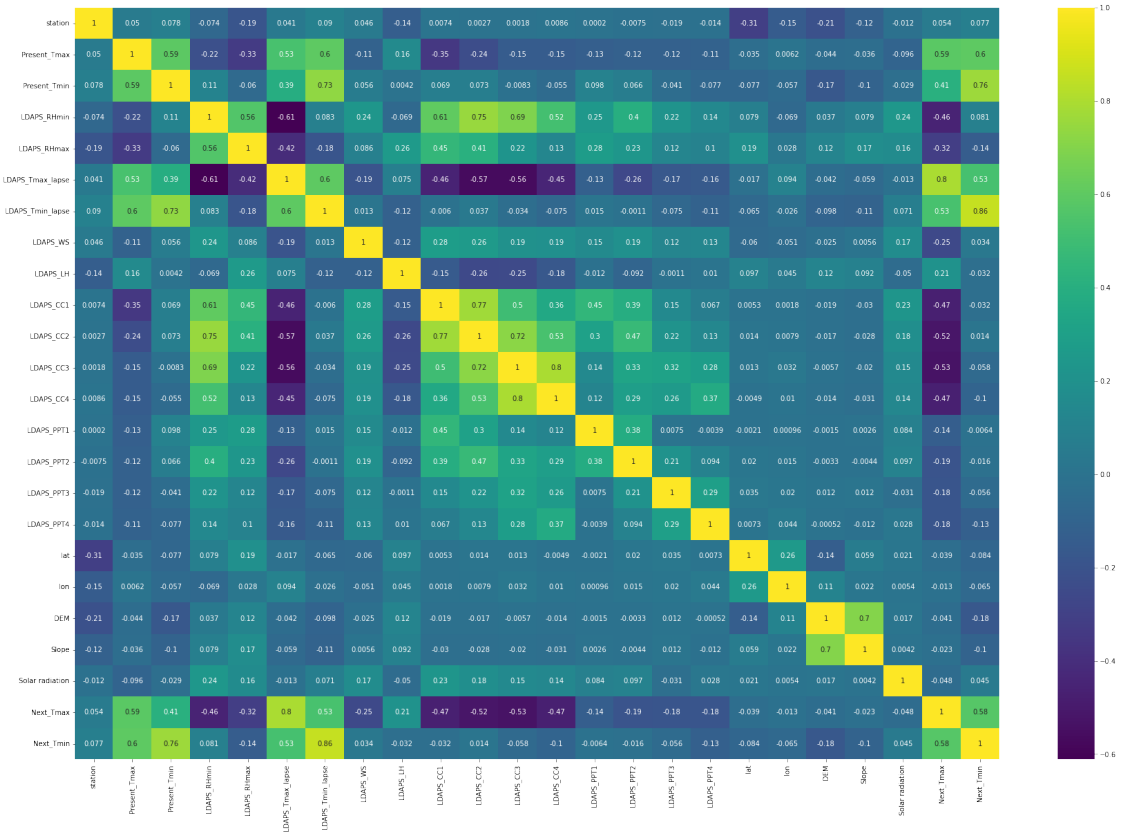
3) Cloud Cover vs. Time



4) Precipitation vs. Time



- The correlation of the updated data after the removal of outliers is plotted as a heatmap for visualization.



3.2 Principal Component Analysis

The total number of features that we could use for the prediction is 17. We tried to reduce the dimensions using PCA. We assumed the number of PCA components to be 3 and calculated the explained variance ratio for each component.

```
Explained variation per principal component: [0.2856587  0.17261231 0.09496715]
```

But reducing the number of components to 3 with just 17 features decreases the accuracy of the models by a significant amount. This due to the trade-off between information loss and dimensionality reduction. Thus we decided to proceed without any reduction of dimensions.

3.3 Regression Models

The Date column is dropped and Next_Tmax & Next_Tmin are saved as ymax and ymin (variables to predict). A copy of the data frame is created (x) and the variables to predict are dropped. The data is then scaled using default setting in MinMaxScalar to put all the features in same scale (i.e. scaling every attributes between 0 to 1).
Also to choose the best random state for train-test split, Logistic regression model has been used.

3.3.1 Regression models used:

- **Linear Regression:**
  Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output).
  The Linear Regression fits a linear model with coefficients to minimize the residual sum of squares between the observed targets in the data set, and the targets predicted by the linear approximation.

- **K-Neighbors Regressor:** KNN algorithm uses 'feature similarity' to predict the values of any new data points. This means that the new point is assigned a value based on how closely it resembles the points in the training set.
  The target is predicted by local interpolation of the targets associated of the nearest neighbors in the training set i.e. KNN takes the mean of the nearest k neighbors. (k is taken as default value i.e. 5)

- **Support Vector Regression:**
  A support vector machine constructs a hyper-plane or set of hyper-planes in a high or infinite dimensional space, which can be used for classification, regression or other tasks. Intuitively, a good separation is achieved by the hyper-plane that has the largest distance to the nearest training data points of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier.
  The model produced by Support Vector Regression depends only on a subset of the training data, because the cost function ignores samples whose prediction

is close to their target (i.e. doesn't care about the training points that lie beyond the margin).

- **Ridge Regression:**
  Ridge Regression is an extension of linear regression that adds a regularization penalty (L2 penalty) to the loss function during training. This has the effect of shrinking the coefficients for those input variables that do not contribute much to the prediction task.
  Minimizes the objective function: $||y - Xw||_2^2 + alpha * ||w||_2^2$

- **Decision Tree Regressor:**
  Decision tree regression observes features of an object and trains a model in the structure of a tree to predict data in the future.
  The decision trees are used to fit a sine curve with addition noisy observation. As a result, it learns local linear regressions approximating the sine curve.

- **Random Forest Regressor:**
  Random Forest is also a "Tree"-based algorithm that uses the qualities features of multiple Decision Trees for making decisions i.e. merges the output of multiple Decision Trees to generate the final output.
  It basically fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

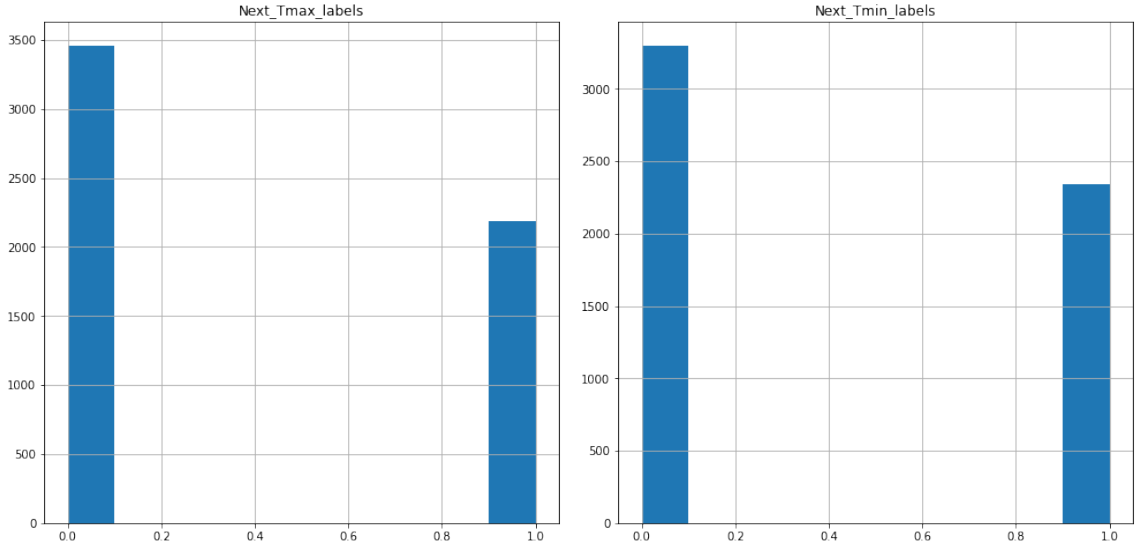
- **Ada Boost Regressor:**
  The AdaBoost algorithm involves using very short (one-level) decision trees as weak learners that are added sequentially to the ensemble. Each subsequent model attempts to correct the predictions made by the model before it in the sequence by weighing the training data set to put more focus on training examples on which prior models made prediction errors.
  Initially, the weights are all set to $w_i = 1/N$, so that the first step simply trains a weak learner on the original data. For each successive iteration, the sample weights are individually modified and the learning algorithm is re-applied to the re-weighted data. At a given step, those training examples that were incorrectly predicted by the boosted model induced at the previous step have their weights increased, whereas the weights are decreased for those that were predicted correctly. As iterations proceed, examples that are difficult to predict receive ever-increasing influence.

- **Gradient Boosting Regressor:**
  Gradient Boosting also uses Decision Trees as weak learners. Trees are constructed in a greedy manner, choosing the best split points based on purity scores like Gini or to minimize the loss. These regression trees output real values for splits and can be added together, allowing subsequent models outputs to be added and "correct" the residuals in the predictions.

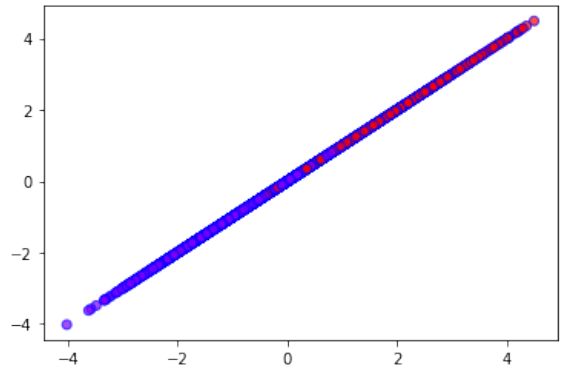
- **XGB Regressor:**
  XGBoost is an efficient implementation of gradient boosting that can be used for regression predictive modeling. It is designed to be both computationally efficient (e.g. fast to execute) and highly effective.

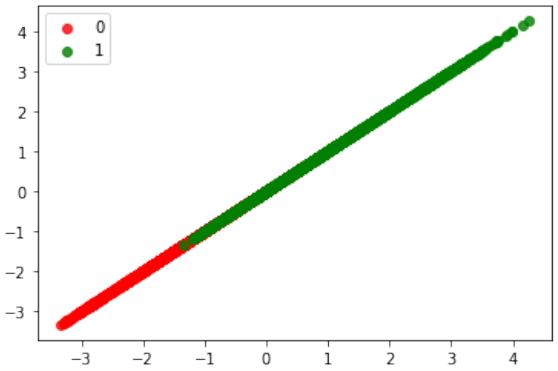
3.4 Classification Models

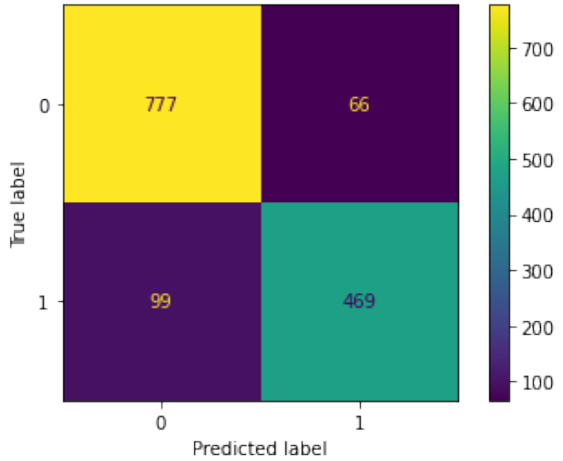
The Next_Tmax and Next_Tmin columns are labeled by taking a threshold of Temperature greater than 30 degree Celsius (approximately the mean value) as hot for Next_Tmax and greater than 23 degree Celsius (approximately the mean value) for Next_Tmin. This threshold value is just an assumption to attempt a classification problem using the same attributes.
The label encoded data can be observed in the histograms below



We tried to apply Linear Discriminant Analysis with 1 component and the precision score was 0.88 approx.








```
Precision score 0.8766355140186916
```

3.4.1 Classification models used:

- **Gaussian Naive Bayes Classifier:**
  The Gaussian Naive Bayes Classifier is an extension of Naive Bayes Classifier. The algorithm assumes the likelihood of the features to be Gaussian.
  $P(x_i|y) = 1/\sqrt{2\pi\sigma_y^2} * exp(-(x_i - \mu_y)^2/2\sigma_y^2)$

- **K-Neighbor Classifier:**
  K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
  K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.

- **Support Vector Classifier:**
  Like explained in the Support Vector Regressor, the support vector machine constructs a hyper-plane or set of hyper-planes in a high or infinite dimensional space, which can be used for classification, regression or other tasks. Intuitively, a good separation is achieved by the hyper-plane that has the largest distance to the nearest training data points of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier.

- **Logistic Regression:**
  Logistic regression uses logistic function i.e. sigmoid function which is an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits. regression uses an equation as the representation, very much like linear regression. Input values (x) are combined linearly using weights or coefficient values (β) to predict an output value (y). A key difference from linear regression is that the output value being modeled is a binary values (0 or 1) rather than a numeric value.

- **Decision Tree Classifier:**
  As explained for Decision Tree Regressor, Decision Tree Classifier is made of a decision tree and is capable of performing multi-class classification on a data set with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.

- **Random Forest Classifier:**
  Again, as explained for Random Forest Regressor, A random forest fits a number of decision tree classifiers on various sub-samples of the data set and uses averaging to improve the predictive accuracy and control over-fitting.

- **Ada Boost Classifier:**
  The Ada Boost Classifier begins by fitting a classifier on the original data set and then fits additional copies of the classifier on the same data set but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

- **Gradient Boosting Classifier:**
  GB builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage n classes regression trees are fit on the negative gradient of the binomial or multinomial deviance loss function. Binary classification is a special case where only a single regression tree is induced.

- **XGB Classifier:**
  XGBoost Classifier is an efficient implementation of gradient boosting classifier. It is designed to be both computationally efficient (e.g. fast to execute) and highly effective.

4 Results

4.1 Regression:

4.1.1 Performance Evaluation Table for Prediction of Next_Tmax

Model	Mean Absolute Error	Mean Squared Error	Root Mean Squared Error	R2 Score	Mean of Cross Validation Score
KNeighborsRegressor	1.0774	2.0791	1.4419	67.00	65.0
SVR	1.0884	2.0656	1.4372	63.00	51.0
LinearRegression	0.9004	1.6296	1.2765	74.00	63.0
Ridge	1.0744	2.0746	1.4403	67.00	65.0
DecisionTreeRegressor	1.0492	2.1833	1.4776	73.00	36.0
RandomForestRegressor	0.7370	1.0018	1.0009	85.00	62.0
AdaBoostRegressor	1.1325	2.0592	1.4350	56.00	59.0
GradientBoostingRegressor	0.8834	1.3646	1.1682	78.00	64.0
XGBRegressor	0.8847	1.3726	1.1716	78.00	65.0

4.1.2 Performance Evaluation Table for Prediction of Next_Tmin

Model	Mean Absolute Error	Mean Squared Error	Root Mean Squared Error	R2 Score	Mean of Cross Validation Score
KNeighborsRegressor	0.7356	0.8959	0.9465	78.00	73.0
SVR	0.7843	1.0455	1.0225	67.00	56.0
LinearRegression	0.6423	0.7087	0.8419	83.00	72.0
Ridge	0.7354	0.8951	0.9461	77.00	73.0
DecisionTreeRegressor	0.8011	1.1879	1.0899	77.00	49.0
RandomForestRegressor	0.5375	0.4858	0.6970	88.00	73.0
AdaBoostRegressor	0.8180	1.0081	1.0041	70.00	68.0
GradientBoostingRegressor	0.6107	0.6019	0.7758	85.00	74.0
XGBRegressor	0.6141	0.6096	0.7807	85.00	73.0

4.2 Classification:

4.2.1 Performance Evaluation Table for Classification of Next_Tmax

Model	Accuracy	Mean Cross Validation	AUC ROC Score	Precision	Recall	F1-Score
GausianNaiveBayes	0.8292	0.8081	0.9045	0.8292	0.8292	0.8292
KNeighbourClassifier	0.8547	0.7893	0.9204	0.8547	0.8547	0.8547
SVC	0.8831	0.8377	0.9532	0.8831	0.8831	0.8831
LogisticRegression	0.8717	0.8475	0.9362	0.8717	0.8717	0.8717
DecisionTreeClassifier	0.8880	0.7576	0.8821	0.8880	0.8880	0.8880
RandomForestClassifier	0.9383	0.8218	0.9832	0.9383	0.9383	0.9383
AdaBoostClassifier	0.8653	0.8273	0.9429	0.8653	0.8653	0.8653
GradientBoostingClassifier	0.8930	0.8351	0.9670	0.8930	0.8930	0.8930
XGBClassifier	0.9001	0.8296	0.9660	0.9001	0.9001	0.9001

4.2.2 Performance Evaluation Table for Classification of Next_Tmin

Model	Accuracy	Mean Cross Validation	AUC ROC Score	Precision	Recall	F1-Score
GausianNaiveBayes	0.8292	0.8081	0.9045	0.8292	0.8292	0.8292
KNeighbourClassifier	0.8547	0.7893	0.9204	0.8547	0.8547	0.8547
SVC	0.8831	0.8377	0.9532	0.8831	0.8831	0.8831
LogisticRegression	0.8717	0.8475	0.9362	0.8717	0.8717	0.8717
DecisionTreeClassifier	0.8816	0.7585	0.8719	0.8816	0.8816	0.8816
RandomForestClassifier	0.9369	0.8262	0.9833	0.9369	0.9369	0.9369
AdaBoostClassifier	0.8653	0.8273	0.9429	0.8653	0.8653	0.8653
GradientBoostingClassifier	0.8930	0.8347	0.9669	0.8930	0.8930	0.8930
XGBClassifier	0.9001	0.8296	0.9660	0.9001	0.9001	0.9001

5 Conclusion

The Random Forest Regressor best predicts the air temperatures, followed by Gradient Boosting Regressor and XGB Regressor for both Next_Tmax and Next_Tmin.

Similar trend is followed for the Classification as well. The Random Forest Classifier Best Classifies the labels, followed by XGB Classifier and Gradient Boosting Classifier for bothNext_Tmax and Next_Tmin.

6 References

1. https://scikit-learn.org/stable/supervised_learning.html
2. https://agupubs.onlinelibrary.wiley.com/doi/full/10.1029/2019EA000740
3. k-nearest-neighbor-introduction-regression-python
4. a-comprehensive-guide-for-linear-ridge-and-lasso-regression