Project
CMPS 470

Byte Me: Sanele Harmon Taeden Kitchens

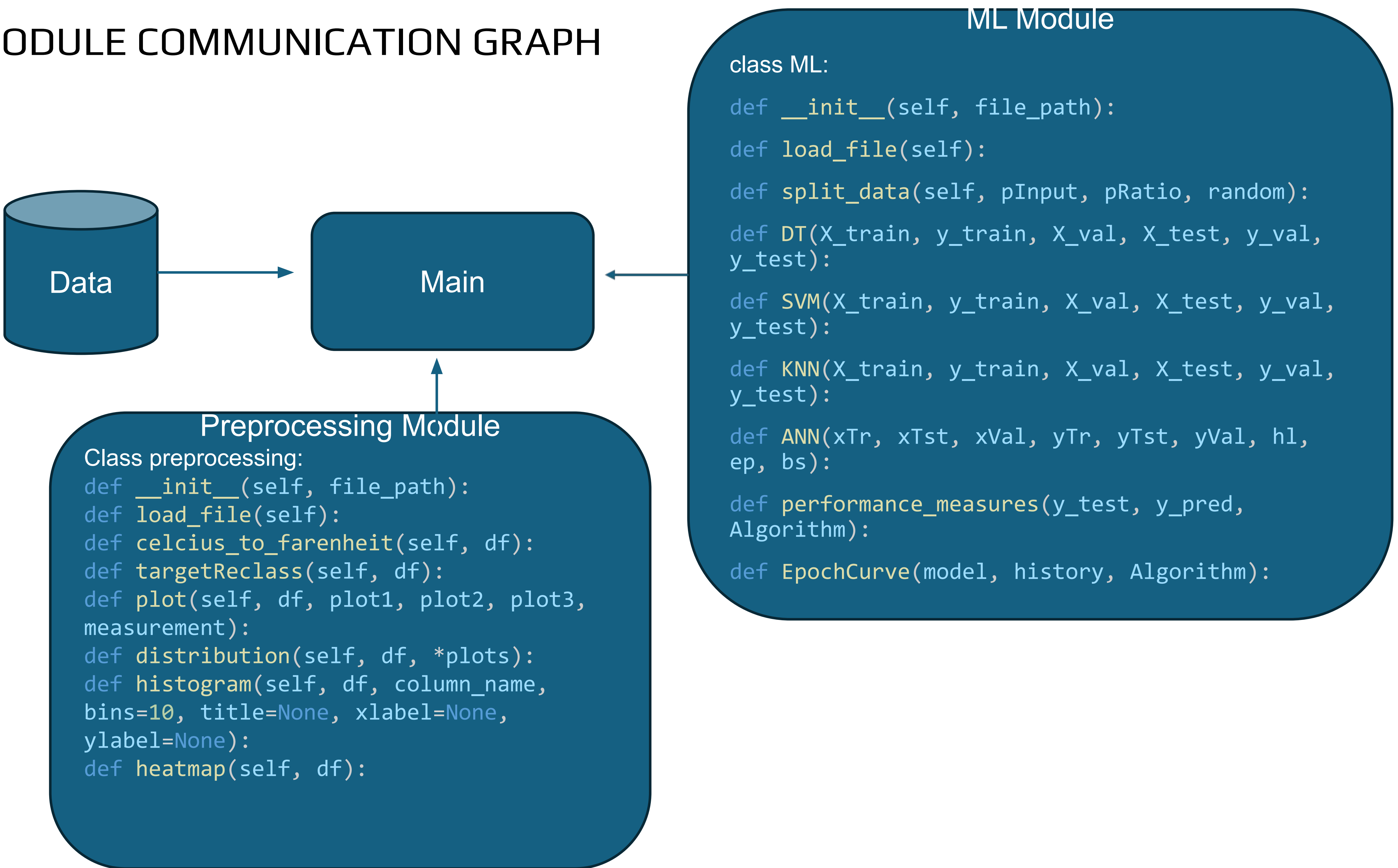
TEAM MEMBERS & ROLES

Sanele Harmon: Create the GitHub repository. Preprocess the temperature columns and plot. Implement SVM and DT. Assess the models

Taeden Kitchen: Preprocess the Rainfall column and plot. Visualize the distribution of each data column. Implement ANN and KNN. Assess the models

Both: Extract Features

MODULE COMMUNICATION GRAPH



ML Module

```
class ML:
def __init__(self, file_path):
def load_file(self):
def split_data(self, pInput, pRatio, random):
def DT(X_train, y_train, X_val, X_test, y_val, y_test):
def SVM(X_train, y_train, X_val, X_test, y_val, y_test):
def KNN(X_train, y_train, X_val, X_test, y_val, y_test):
def ANN(xTr, xTst, xVal, yTr, yTst, yVal, hl, ep, bs):
def performance_measures(y_test, y_pred, Algorithm):
def EpochCurve(model, history, Algorithm):
```

Preprocessing Module

```
Class preprocessing:
def __init__(self, file_path):
def load_file(self):
def celcius_to_farenheit(self, df):
def targetReclass(self, df):
def plot(self, df, plot1, plot2, plot3, measurement):
def distribution(self, df, *plots):
def histogram(self, df, column_name, bins=10, title=None, xlabel=None, ylabel=None):
def heatmap(self, df):
```

DESCRIPTION OF THE PROJECT

- Goal: Predicting whether there will be rainfall on a given day by means of ML algorithms ANN, SVM, DT, and K-NN.

Tasks:

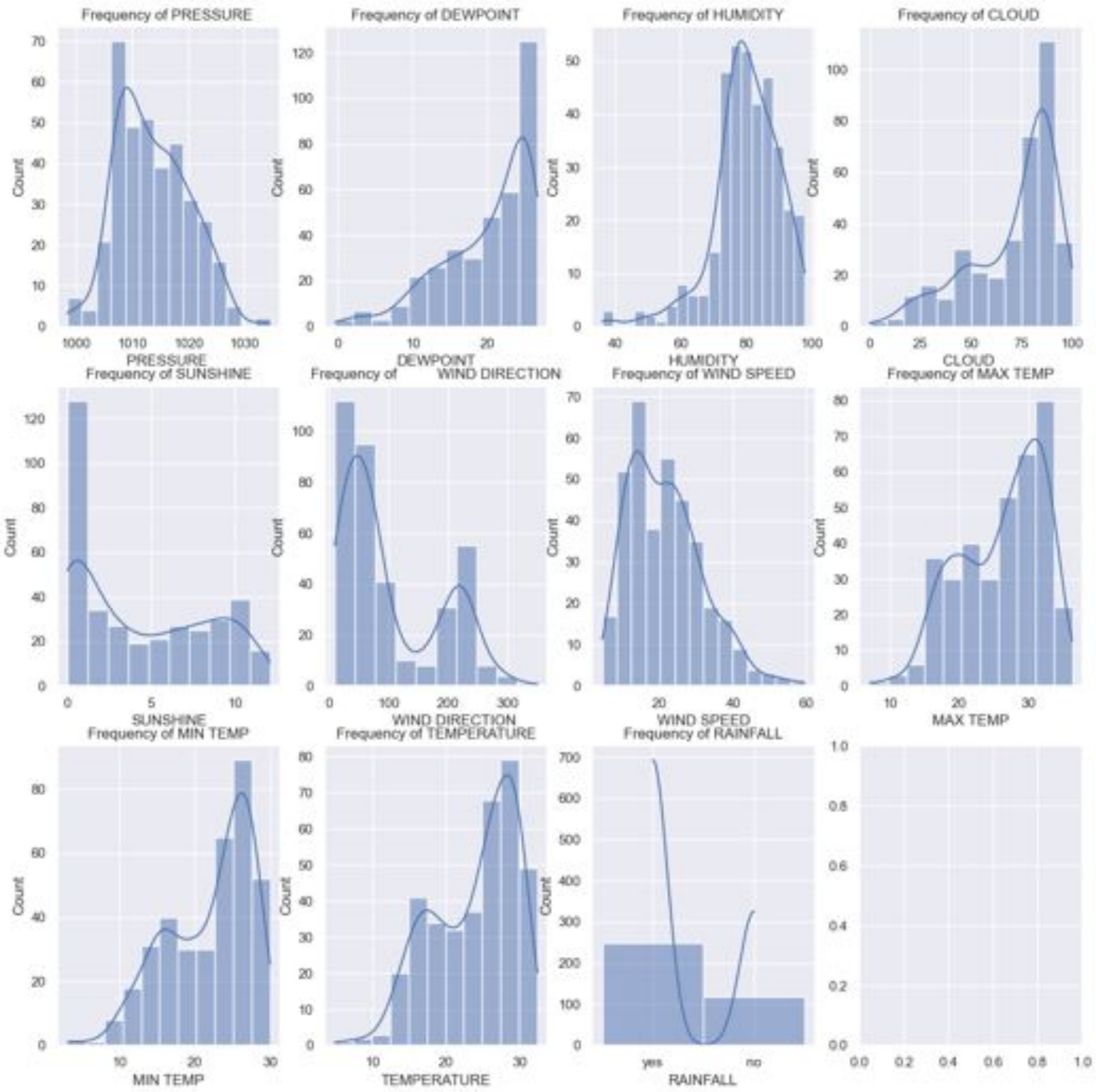
- Visualize data
- Preprocess data
    - Extract Features
    - Implement ML algorithms
    - Assess the models

DESCRIPTION OF THE RAW DATA

- Data Source: Seek Geek

ATTRIBUTE	TYPE
SAMPLE ID	INT
RAINFALL (TARGET)	STRING (‘yes’, ‘no’)
DAY	INT
PRESSURE	FLOAT
MAX TEMP	FLOAT
TEMPERATURE	FLOAT
MIN TEMP	FLOAT
DEW POINT	FLOAT
HUMIDITY	INT
CLOUD	INT
SUNSHINE	FLOAT
WIND DIRECTION	INT
WIND SPEED	FLOAT

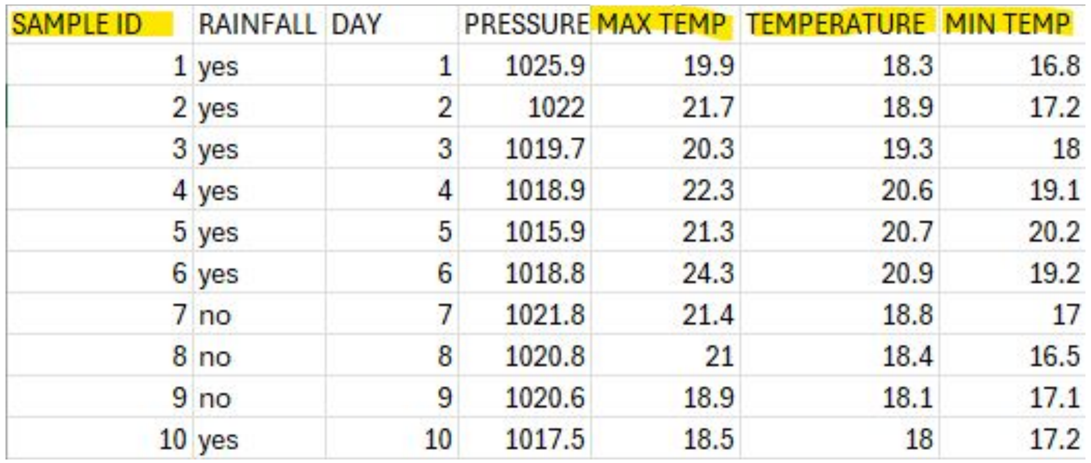
DISTRIBUTION OF DATA



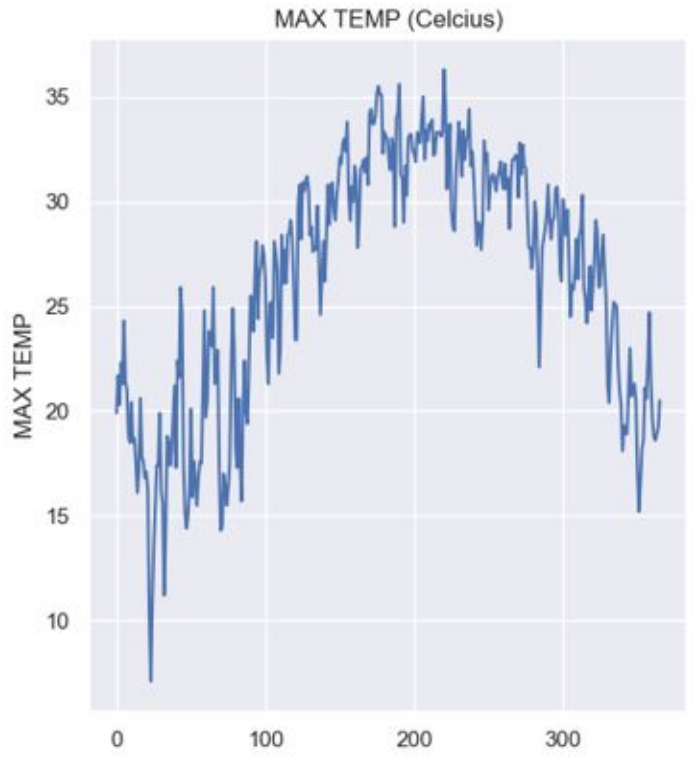
PREPROCESSING

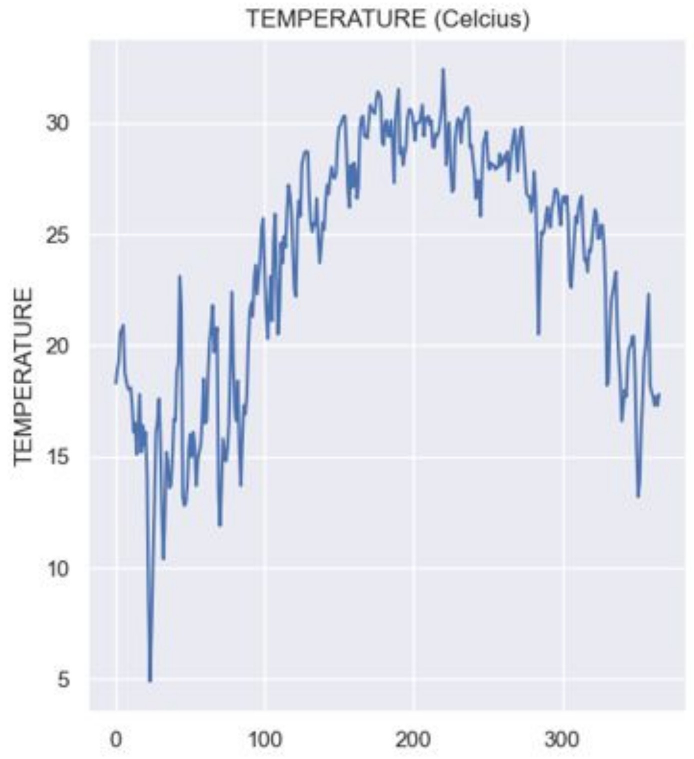
- For preprocessing, the values for the RAINFALL column were mapped from ‘yes’ and ‘no’ to 1 and 0, respectively
- The MAX TEMP, MIN TEMP, and TEMPERATURE columns were converted from Celsius to Fahrenheit

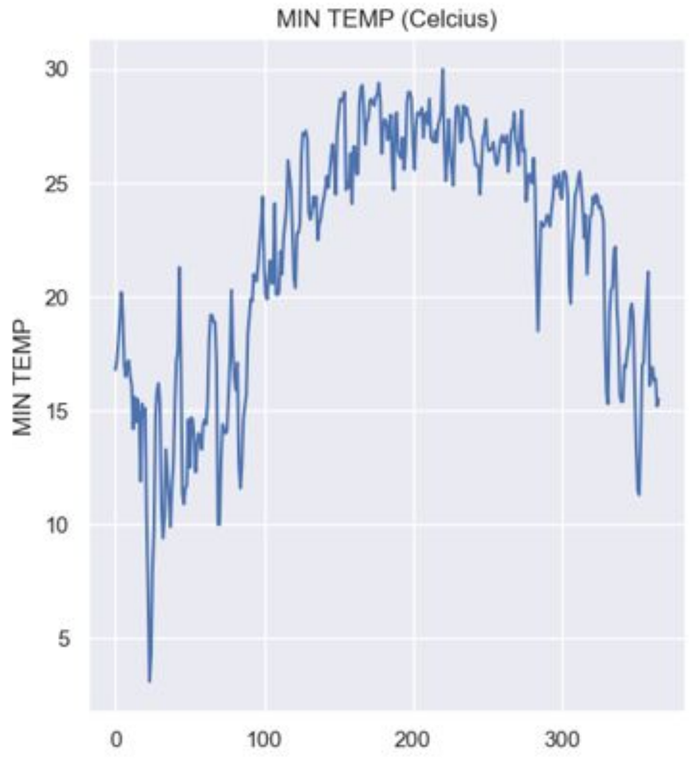
RAW DATA: MAX_TEMP, MIN_TEMP, TEMPERATURE



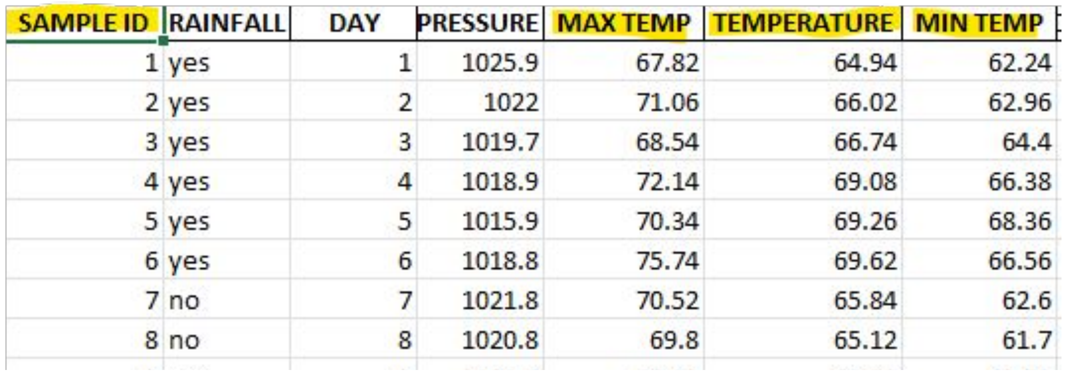
SAMPLE ID	RAINFALL	DAY	PRESSURE	MAX TEMP	TEMPERATURE	MIN TEMP
1	yes	1	1025.9	19.9	18.3	16.8
2	yes	2	1022	21.7	18.9	17.2
3	yes	3	1019.7	20.3	19.3	18
4	yes	4	1018.9	22.3	20.6	19.1
5	yes	5	1015.9	21.3	20.7	20.2
6	yes	6	1018.8	24.3	20.9	19.2
7	no	7	1021.8	21.4	18.8	17
8	no	8	1020.8	21	18.4	16.5
9	no	9	1020.6	18.9	18.1	17.1
10	yes	10	1017.5	18.5	18	17.2



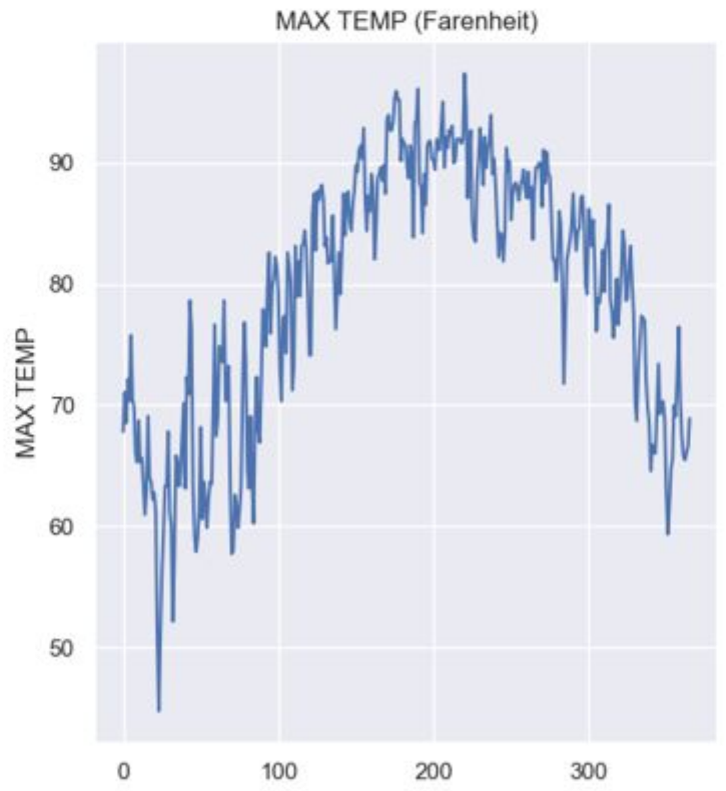


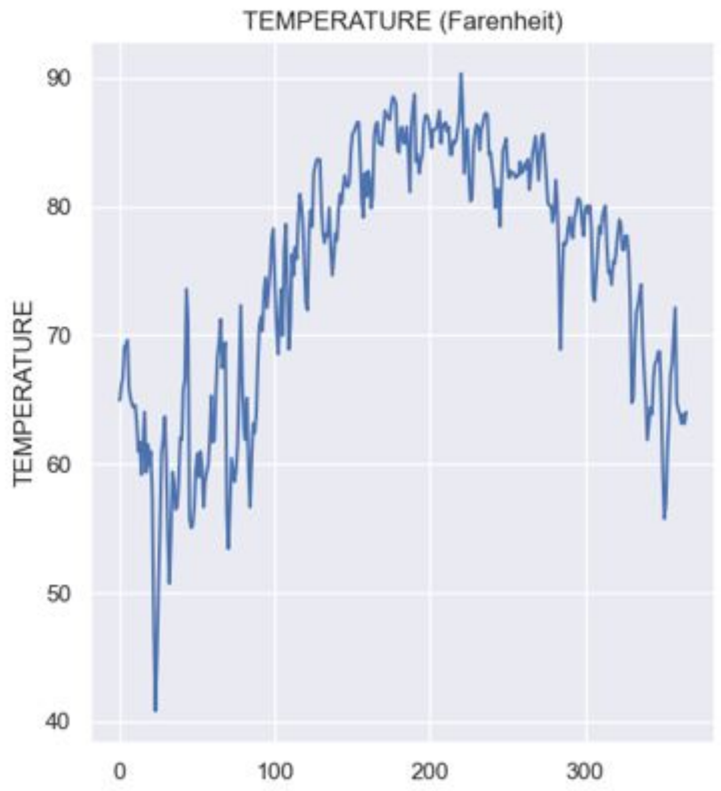


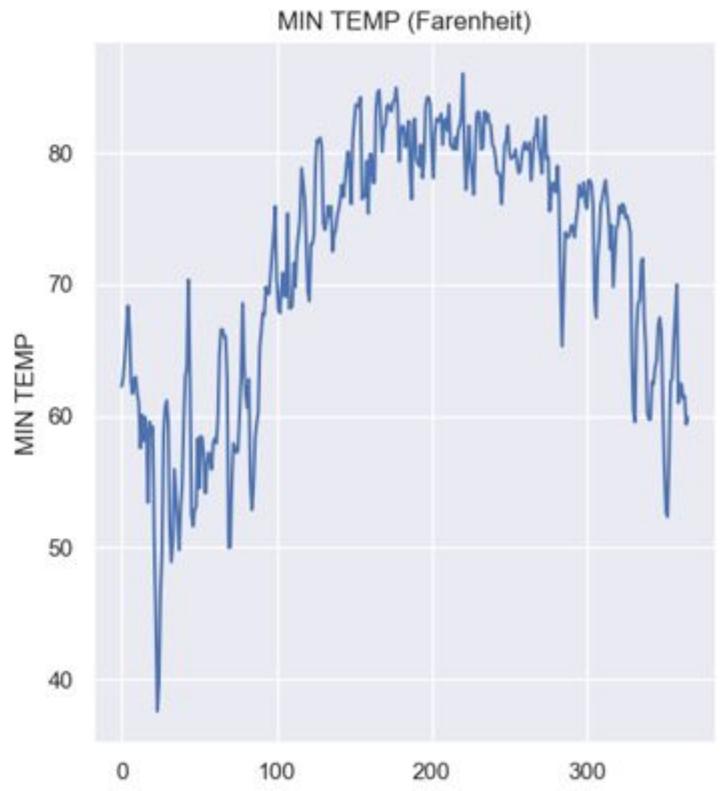
PREPROCESSED DATA: MAX_TEMP, MIN_TEMP, TEMPERATURE



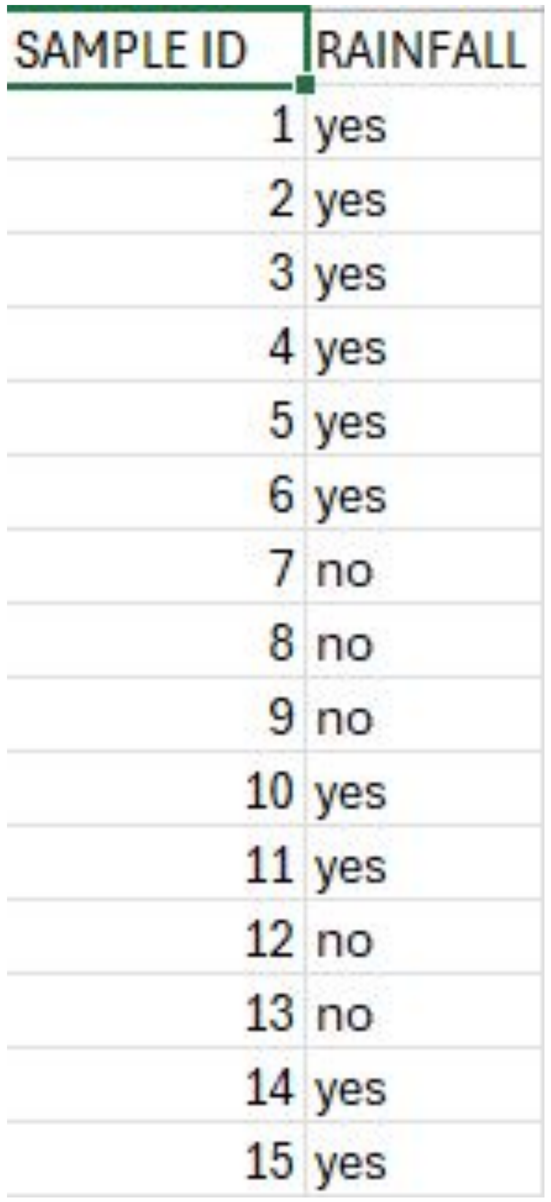
SAMPLE ID	RAINFALL	DAY	PRESSURE	MAX TEMP	TEMPERATURE	MIN TEMP
1	yes	1	1025.9	67.82	64.94	62.24
2	yes	2	1022	71.06	66.02	62.96
3	yes	3	1019.7	68.54	66.74	64.4
4	yes	4	1018.9	72.14	69.08	66.38
5	yes	5	1015.9	70.34	69.26	68.36
6	yes	6	1018.8	75.74	69.62	66.56
7	no	7	1021.8	70.52	65.84	62.6
8	no	8	1020.8	69.8	65.12	61.7



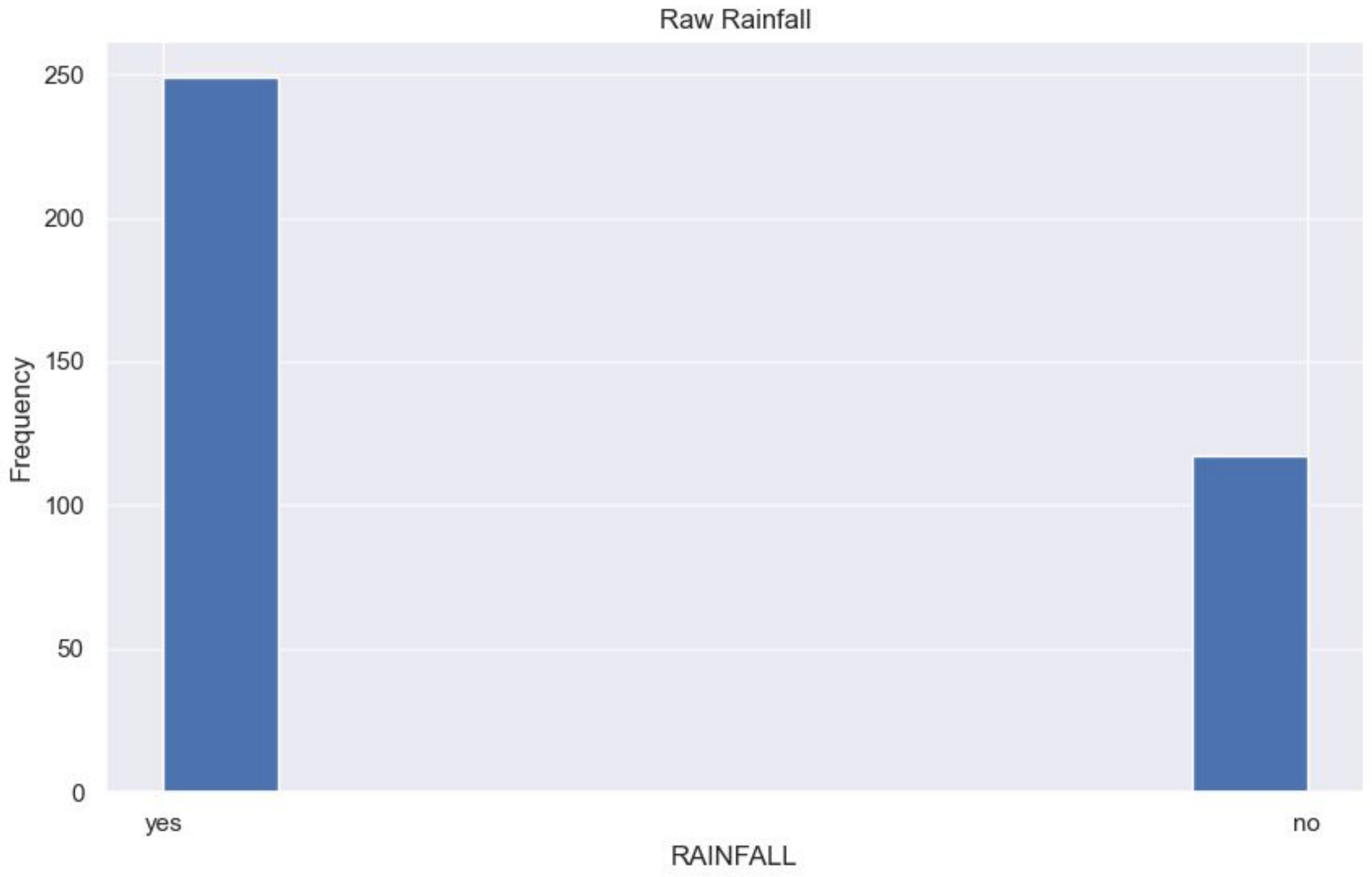




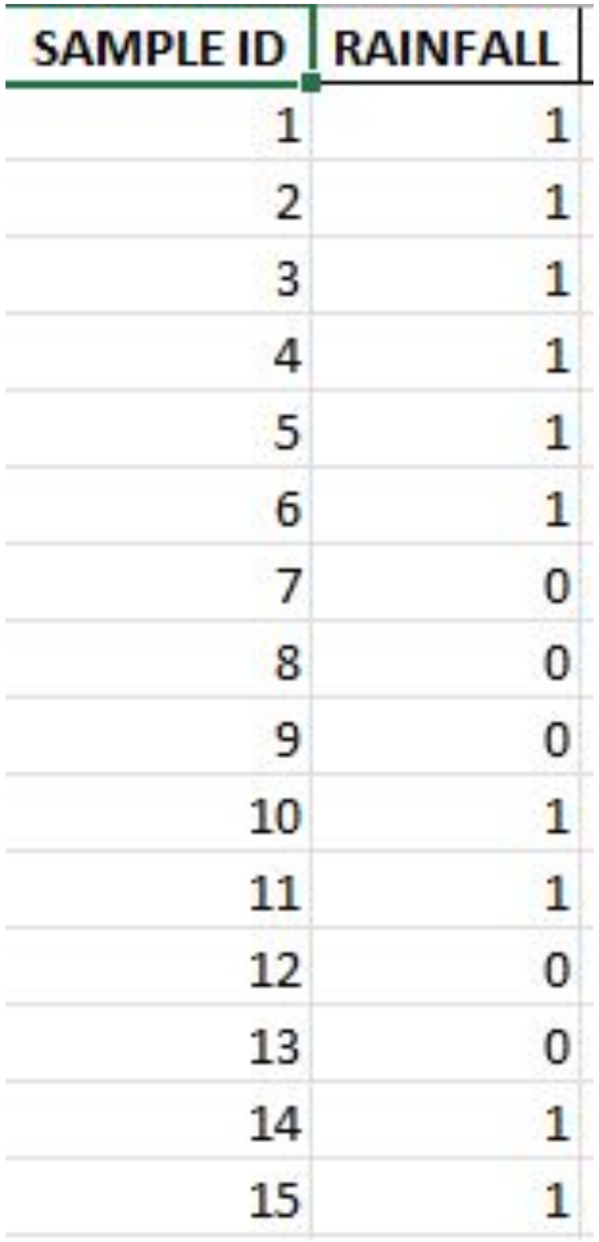
RAW DATA: Rainfall



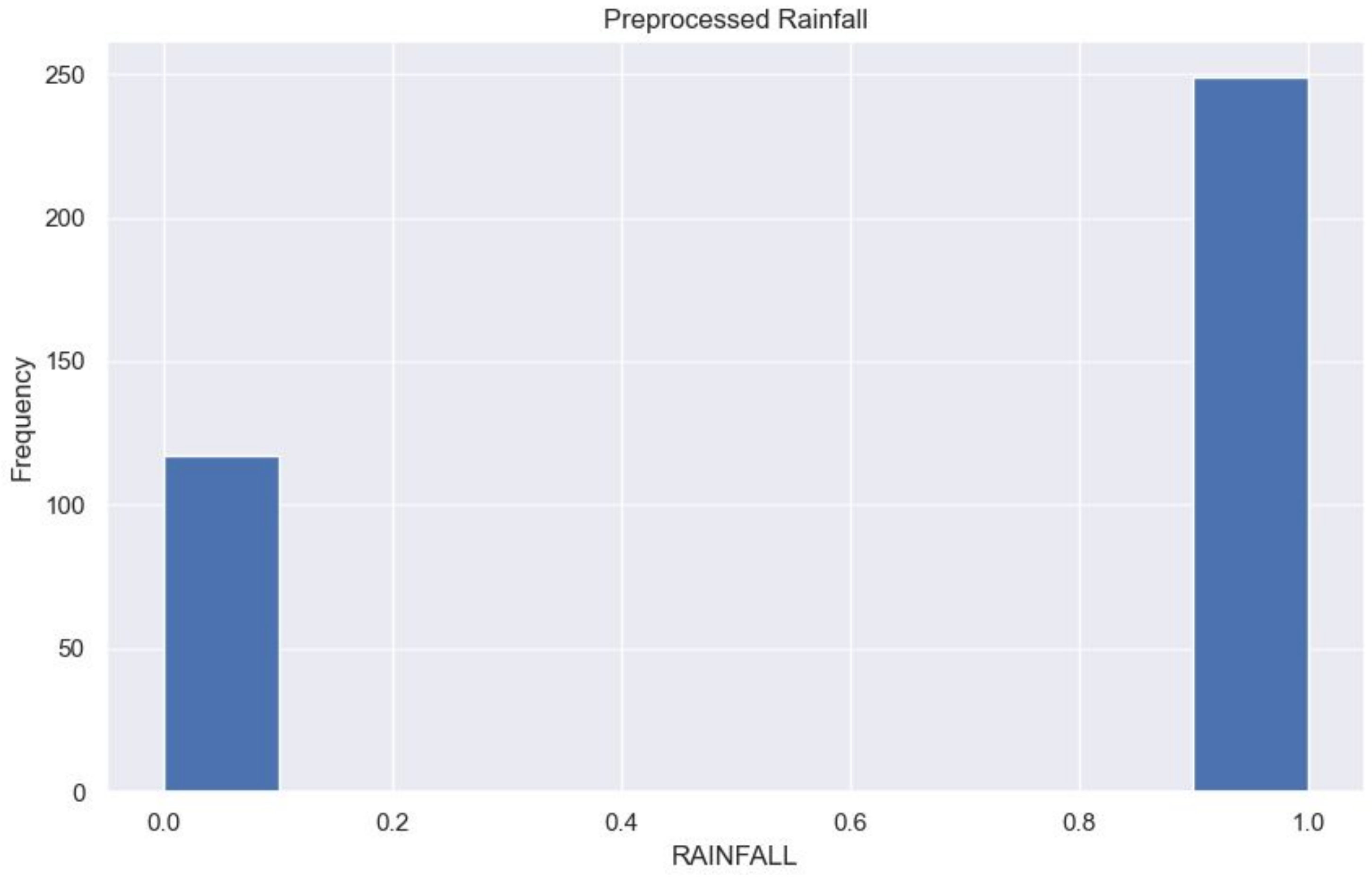
SAMPLE ID	RAINFALL
1	yes
2	yes
3	yes
4	yes
5	yes
6	yes
7	no
8	no
9	no
10	yes
11	yes
12	no
13	no
14	yes
15	yes



PREPROCESSED DATA: Rainfall



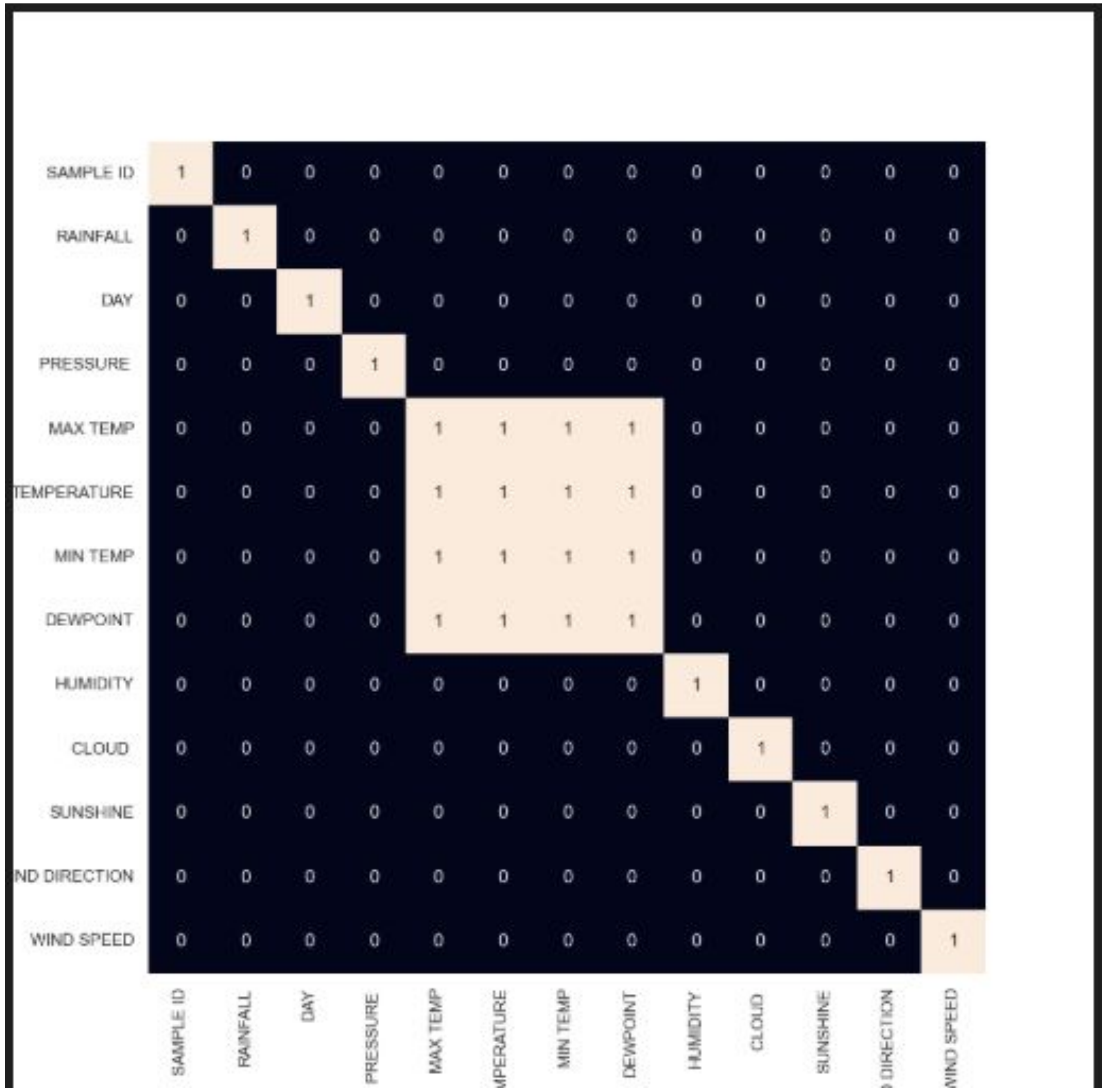
SAMPLE ID	RAINFALL
1	1
2	1
3	1
4	1
5	1
6	1
7	0
8	0
9	0
10	1
11	1
12	0
13	0
14	1
15	1



FEATURE EXTRACTION

- For feature extraction, we made a heat map for all the existing features, DAY, PRESSURE, MAX TEMP, TEMPERATURE, MIN_TEMP, DEWPOINT, HUMIDITY, CLOUD, SUNSHINE, WIND DIRECTION, WIND SPEED, to see which were highly correlated.
- Since TEMPERATURE, MIN TEMP, AND MAX_TEMP are highly correlated and provide redundant information, we dropped MIN_TEMP and MAX_TEMP.
- The DAY column provided no information for the ML algorithms, so that feature was also dropped.

Feature Heat Map:

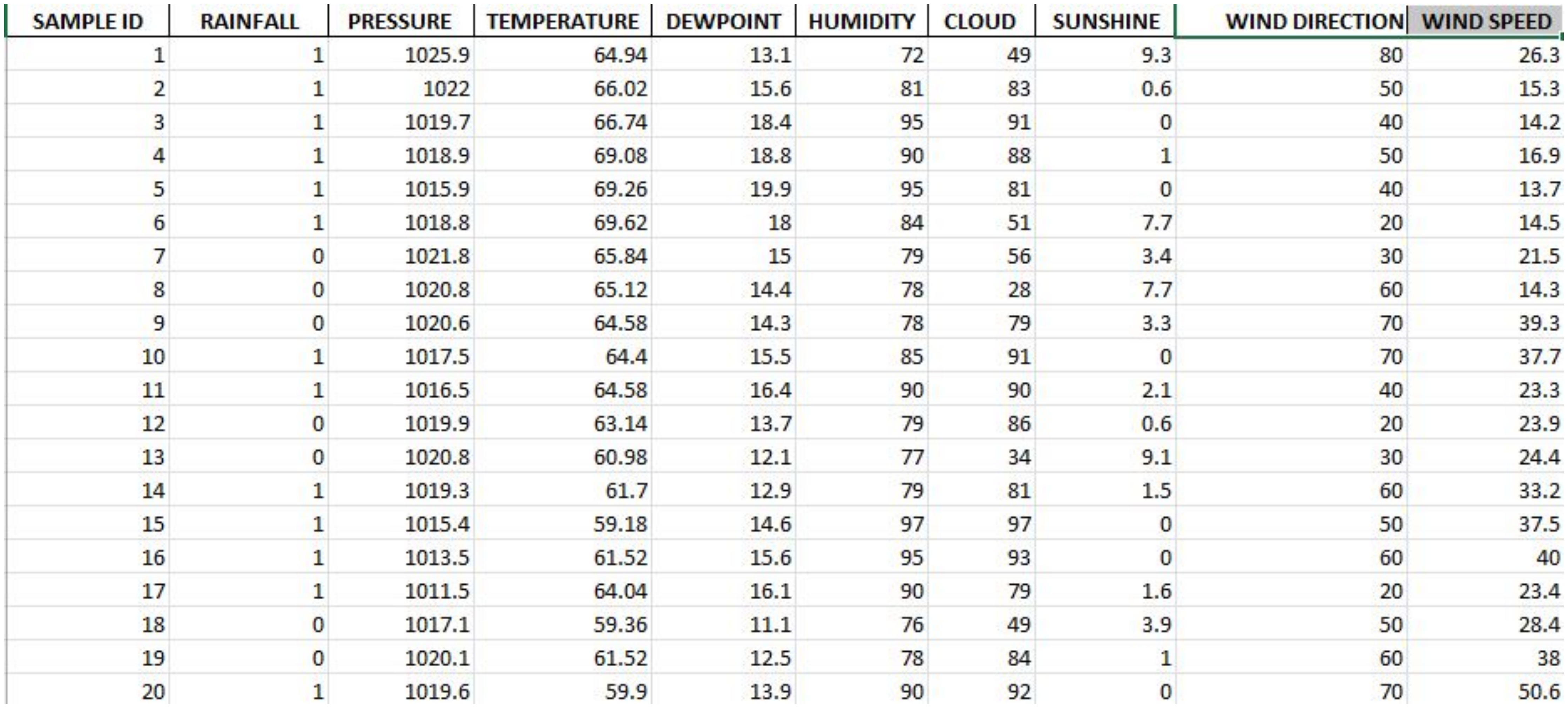


	SAMPLE ID	RAINFALL	DAY	PRESSURE	MAX TEMP	MPERATURE	MIN TEMP	DEWPOINT	HUMIDITY	CLOUD	SUNSHINE	DIRECTION	WIND SPEED
SAMPLE ID	1	0	0	0	0	0	0	0	0	0	0	0	0
RAINFALL	0	1	0	0	0	0	0	0	0	0	0	0	0
DAY	0	0	1	0	0	0	0	0	0	0	0	0	0
PRESSURE	0	0	0	1	0	0	0	0	0	0	0	0	0
MAX TEMP	0	0	0	0	1	1	1	1	0	0	0	0	0
TEMPERATURE	0	0	0	0	1	1	1	1	0	0	0	0	0
MIN TEMP	0	0	0	0	1	1	1	1	0	0	0	0	0
DEWPOINT	0	0	0	0	1	1	1	1	0	0	0	0	0
HUMIDITY	0	0	0	0	0	0	0	0	1	0	0	0	0
CLOUD	0	0	0	0	0	0	0	0	0	1	0	0	0
SUNSHINE	0	0	0	0	0	0	0	0	0	0	1	0	0
ND DIRECTION	0	0	0	0	0	0	0	0	0	0	0	1	0
WIND SPEED	0	0	0	0	0	0	0	0	0	0	0	0	1

Extracted Features

ATTRIBUTE	TYPE
SAMPLE ID	INT
RAINFALL (TARGET)	STRING (‘yes’, ‘no’)
PRESSURE	FLOAT
TEMPERATURE	FLOAT
DEW POINT	FLOAT
HUMIDITY	INT
CLOUD	INT
SUNSHINE	FLOAT
WIND DIRECTION	INT
WIND SPEED	FLOAT

Feature Samples



SAMPLE ID	RAINFALL	PRESSURE	TEMPERATURE	DEWPOINT	HUMIDITY	CLOUD	SUNSHINE	WIND DIRECTION	WIND SPEED
1	1	1025.9	64.94	13.1	72	49	9.3	80	26.3
2	1	1022	66.02	15.6	81	83	0.6	50	15.3
3	1	1019.7	66.74	18.4	95	91	0	40	14.2
4	1	1018.9	69.08	18.8	90	88	1	50	16.9
5	1	1015.9	69.26	19.9	95	81	0	40	13.7
6	1	1018.8	69.62	18	84	51	7.7	20	14.5
7	0	1021.8	65.84	15	79	56	3.4	30	21.5
8	0	1020.8	65.12	14.4	78	28	7.7	60	14.3
9	0	1020.6	64.58	14.3	78	79	3.3	70	39.3
10	1	1017.5	64.4	15.5	85	91	0	70	37.7
11	1	1016.5	64.58	16.4	90	90	2.1	40	23.3
12	0	1019.9	63.14	13.7	79	86	0.6	20	23.9
13	0	1020.8	60.98	12.1	77	34	9.1	30	24.4
14	1	1019.3	61.7	12.9	79	81	1.5	60	33.2
15	1	1015.4	59.18	14.6	97	97	0	50	37.5
16	1	1013.5	61.52	15.6	95	93	0	60	40
17	1	1011.5	64.04	16.1	90	79	1.6	20	23.4
18	0	1017.1	59.36	11.1	76	49	3.9	50	28.4
19	0	1020.1	61.52	12.5	78	84	1	60	38
20	1	1019.6	59.9	13.9	90	92	0	70	50.6

Description of the Model

Parameters for SVM: {'C': 10, 'gamma': 0.01, 'kernel': 'rbf'}

C balances correctly classifying training points and a smooth decision boundary. 10 is a high C value which chooses to classify all the training points correctly, but could lead to overfitting. A low gamma value like 0.1 results in a smoother decision boundary. The kernel 'rbf' is the radial basis function. RBF handles non-linear relationships between features. These parameter were chosen using cross validation

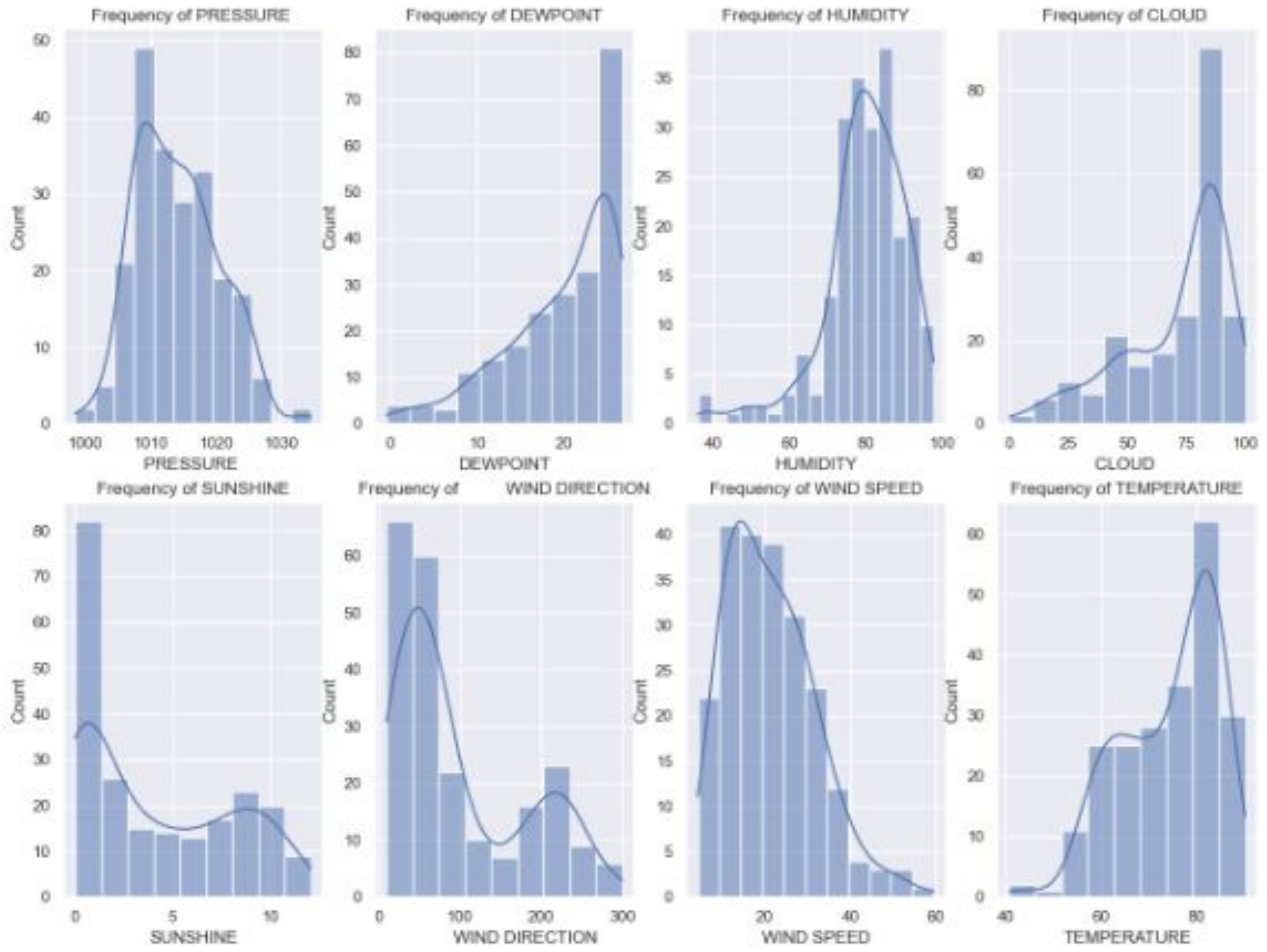
Parameters for DT: {ccp_alpha: 0.030645135275776503}

Ccp_alpha is cost-complexity pruning alpha. The pruning process balances the tree's complexity with how well it fits the training data. As ccp_alpha gets increased, more nodes get pruned, which results in a simpler model. 0.03 was found to be the best ccp_alpha value. The rest of the parameters are default.

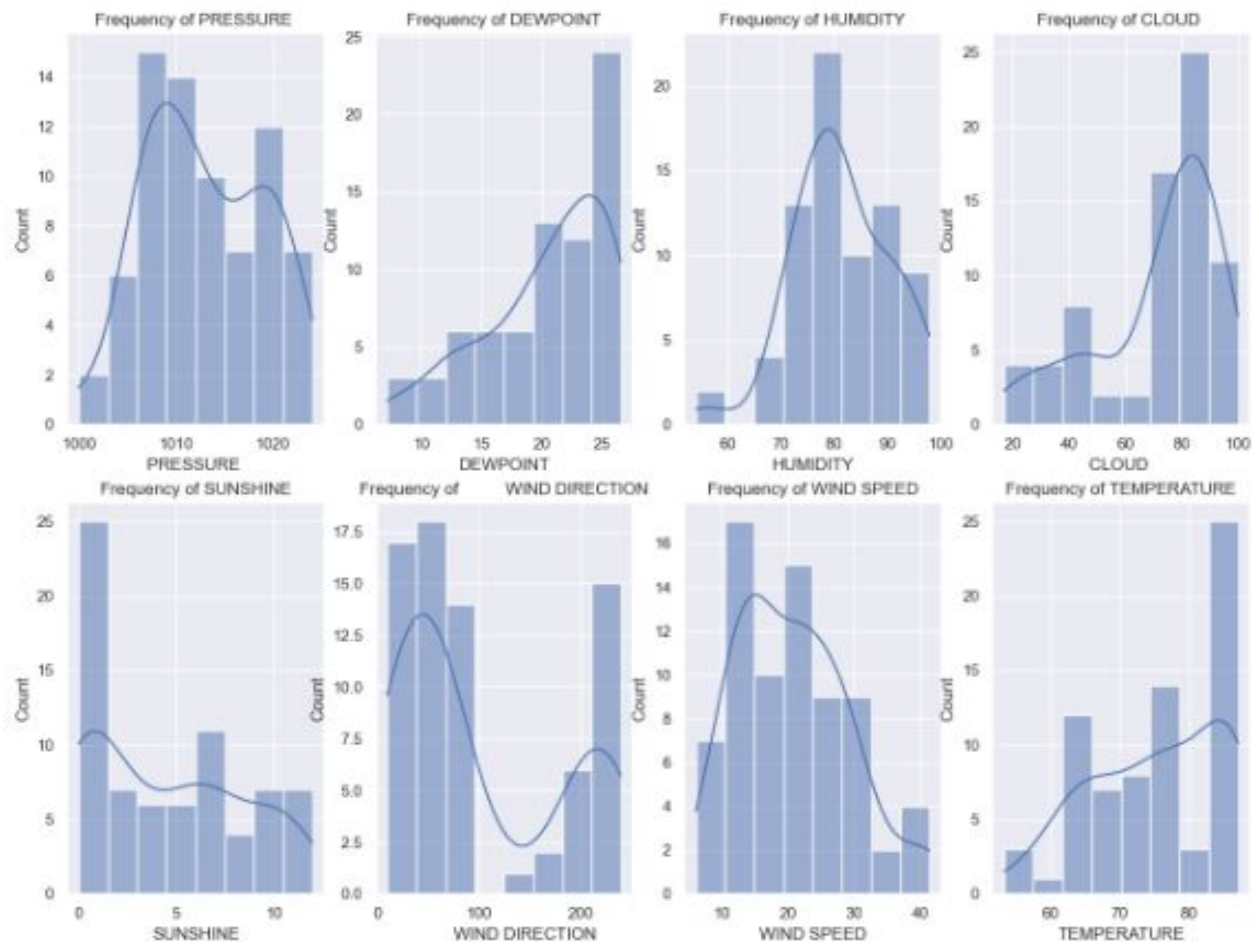
Parameters for ANN:

Parameters for KNN: {k=17}

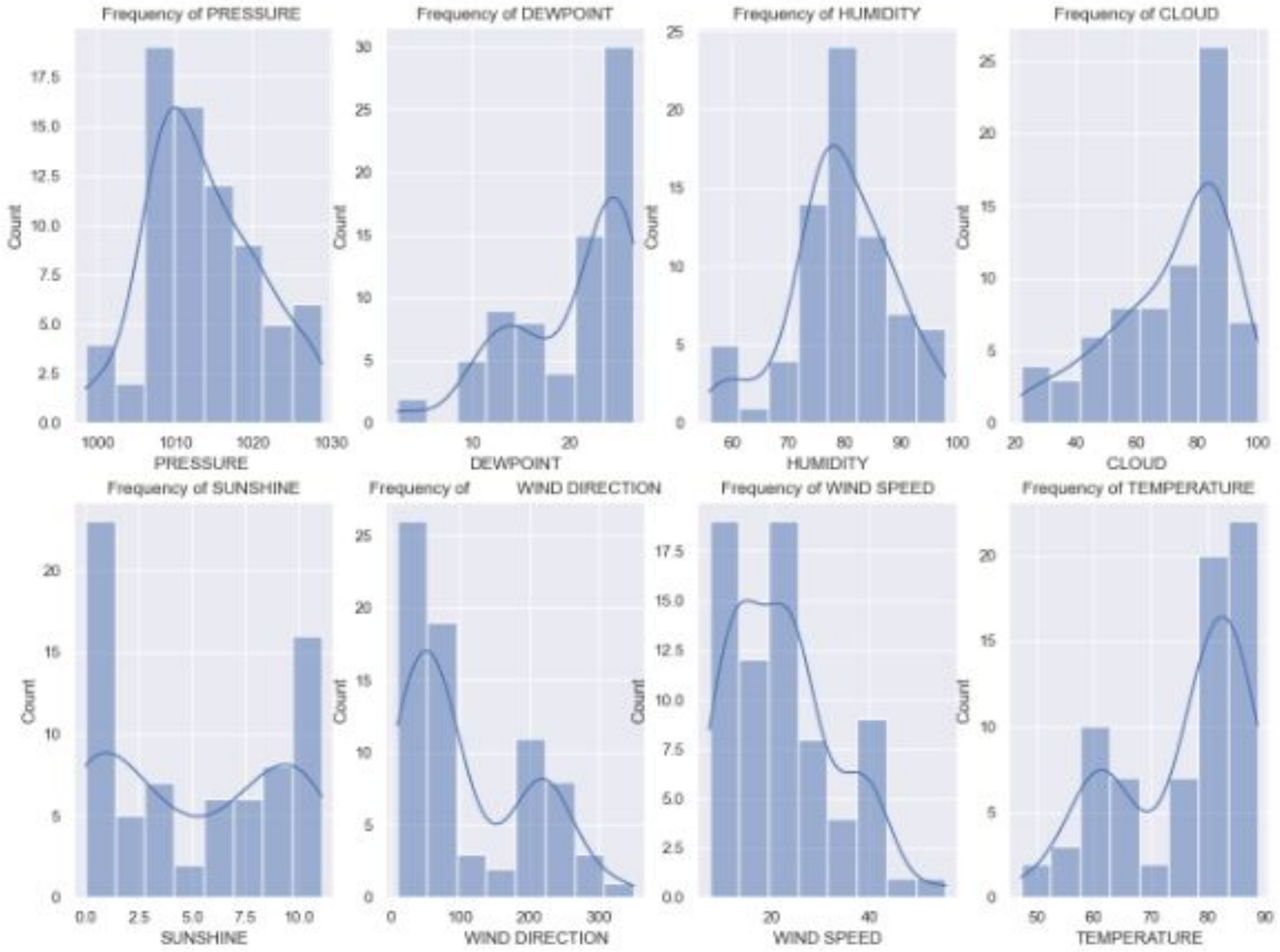
Distribution of Training data



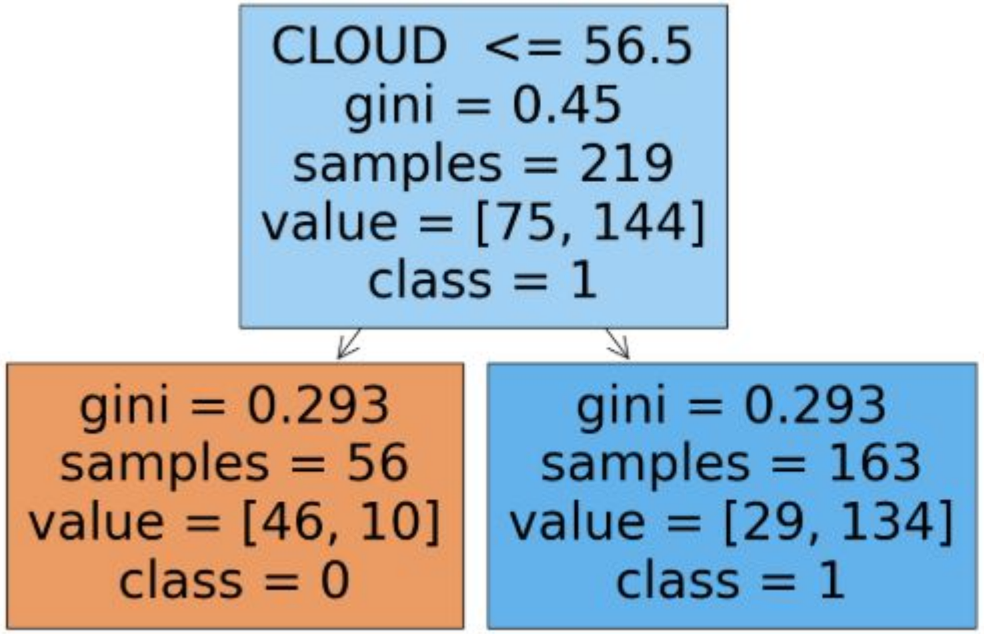
Distribution of Validation data

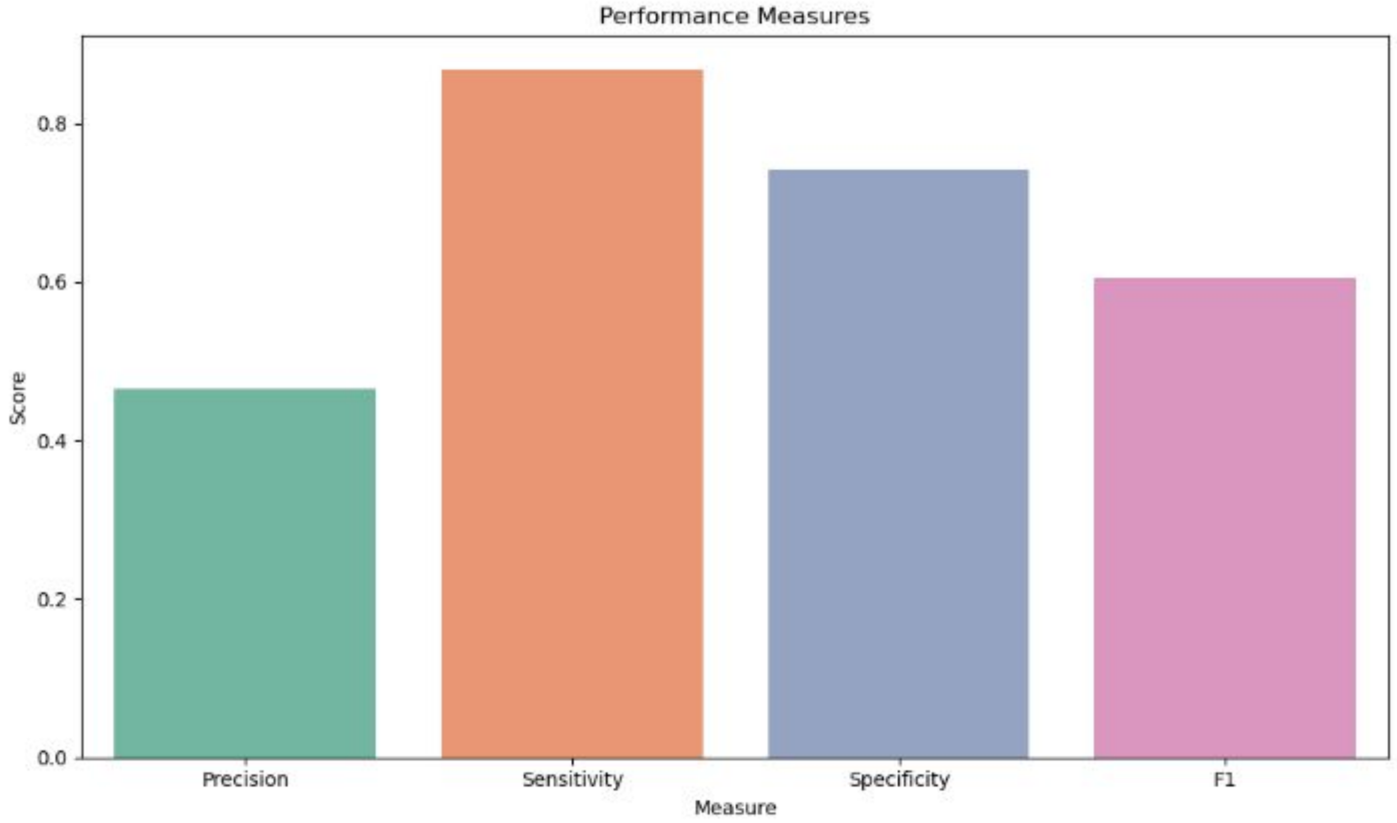


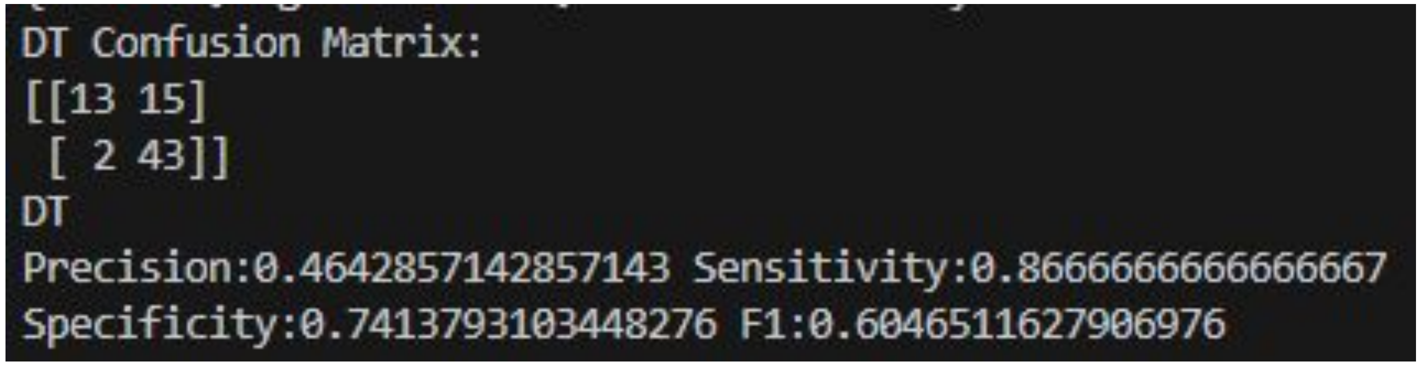
Distribution of Test Data



Performance of Decision Tree







```
DT Confusion Matrix:
[[13 15]
 [ 2 43]]
DT
Precision:0.4642857142857143 Sensitivity:0.8666666666666667
Specificity:0.7413793103448276 F1:0.6046511627906976
```

Discussion of DT Results

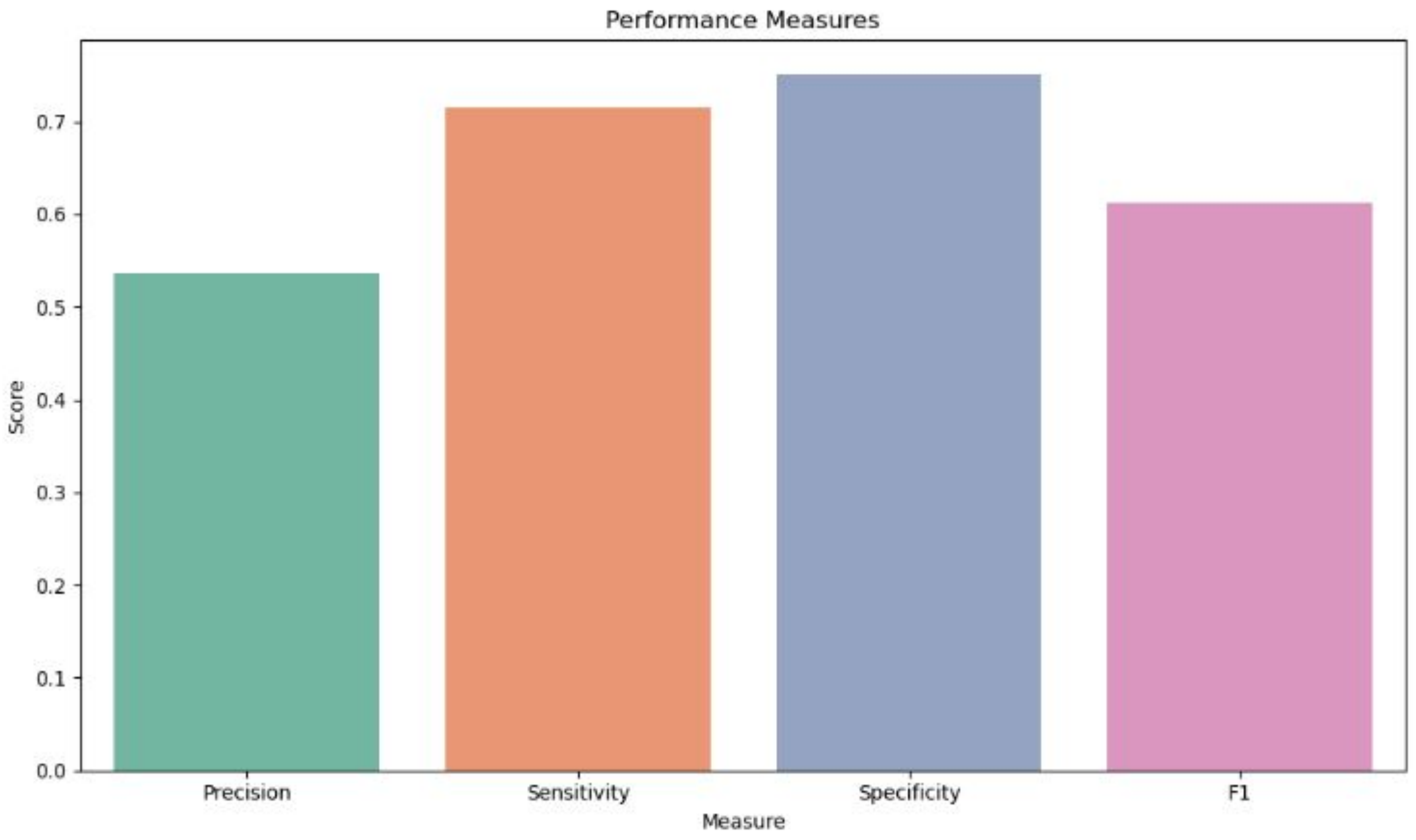
Precision: 0.464
Only 46.4% of the DTs positive predictions were correct. This is low so there were many false positives. This is ok in this case since we are predicting rain. The public can have their umbrellas ready even if there will not be rain.

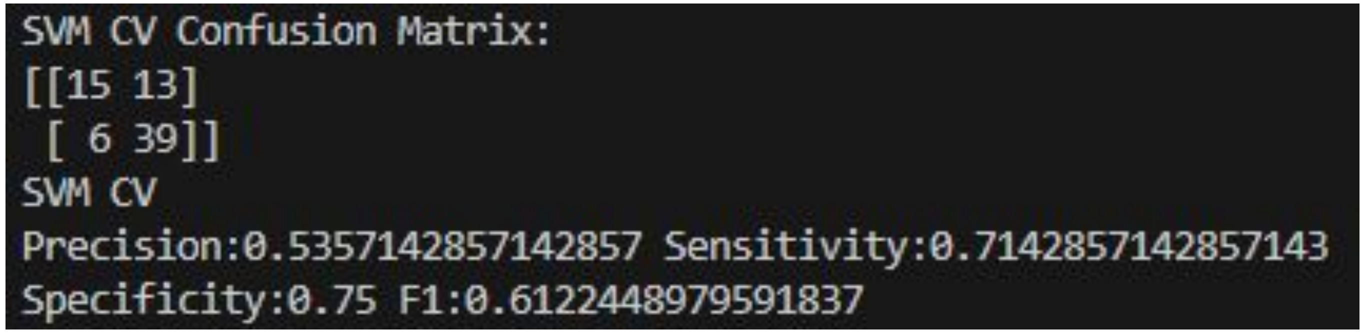
Sensitivity (recall): 0.867
The DT catches 86.7% of all true positive cases for rain. This model is sensitive to correctly detecting positive cases which is good for predicting rainfall.

Specificity: 0.741
DT correctly identifies 74.1% o the True negative cases. It is fairly good at avoiding false positives but it could be improved.

F1: 0.65
This model has a high bias towards recall rather than precision.

Performance of SVM





```
SVM CV Confusion Matrix:
[[15 13]
 [ 6 39]]
SVM CV
Precision:0.5357142857142857 Sensitivity:0.7142857142857143
Specificity:0.75 F1:0.6122448979591837
```

Discuss:

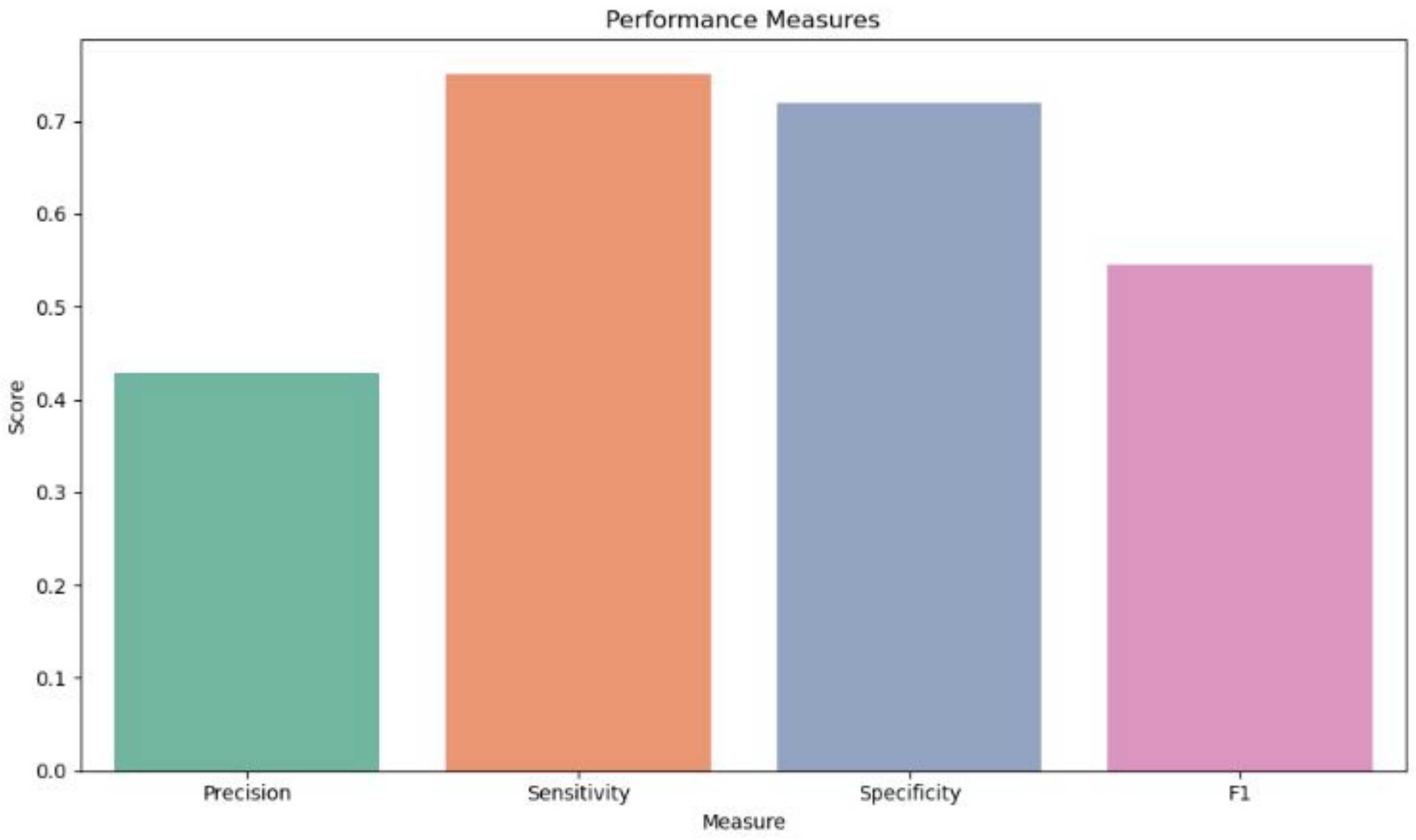
Precision: 0.536
About 53.6% of predicted positives are correct, so about half of the predicted positives are false positives. This is not ideal, but ok in this case since we are predicting rain. The public can have their umbrellas ready even if there will not be rain. SVM has a higher precision than DT.

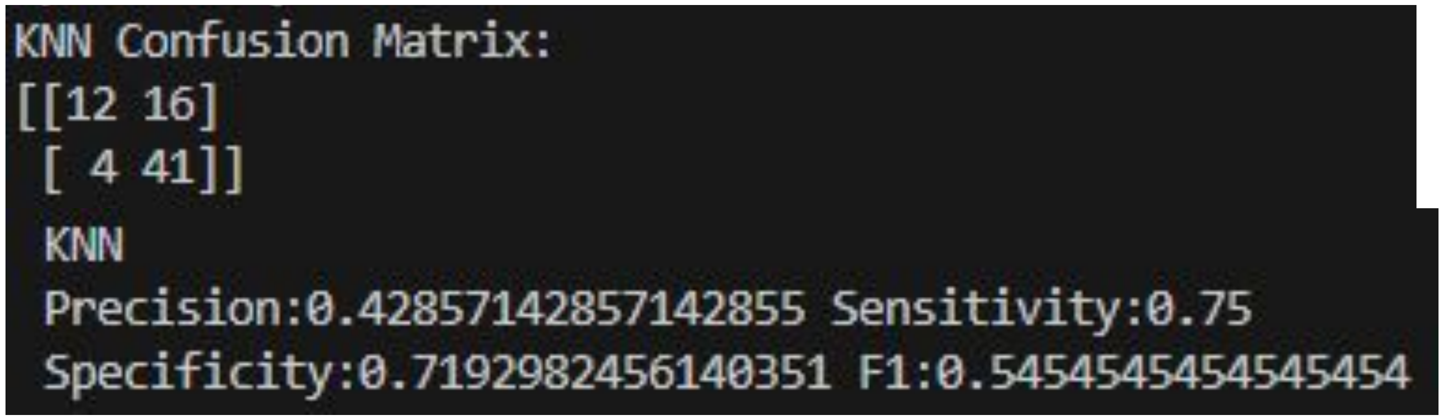
Sensitivity (recall): 0.714
71.4% of true positives are correctly predicted so there is room for improvement. SVM misses more positives than DT

Specificity: 0.75
75% of true negatives are correctly identified. This is a good percentage since we are predicting rainfall

F1: 0.612
There is good balance between precision and recall

Performance of KNN





```
KNN Confusion Matrix:
[[12 16]
 [ 4 41]]
 KNN
 Precision:0.42857142857142855 Sensitivity:0.75
 Specificity:0.7192982456140351 F1:0.5454545454545454
```

Discuss:

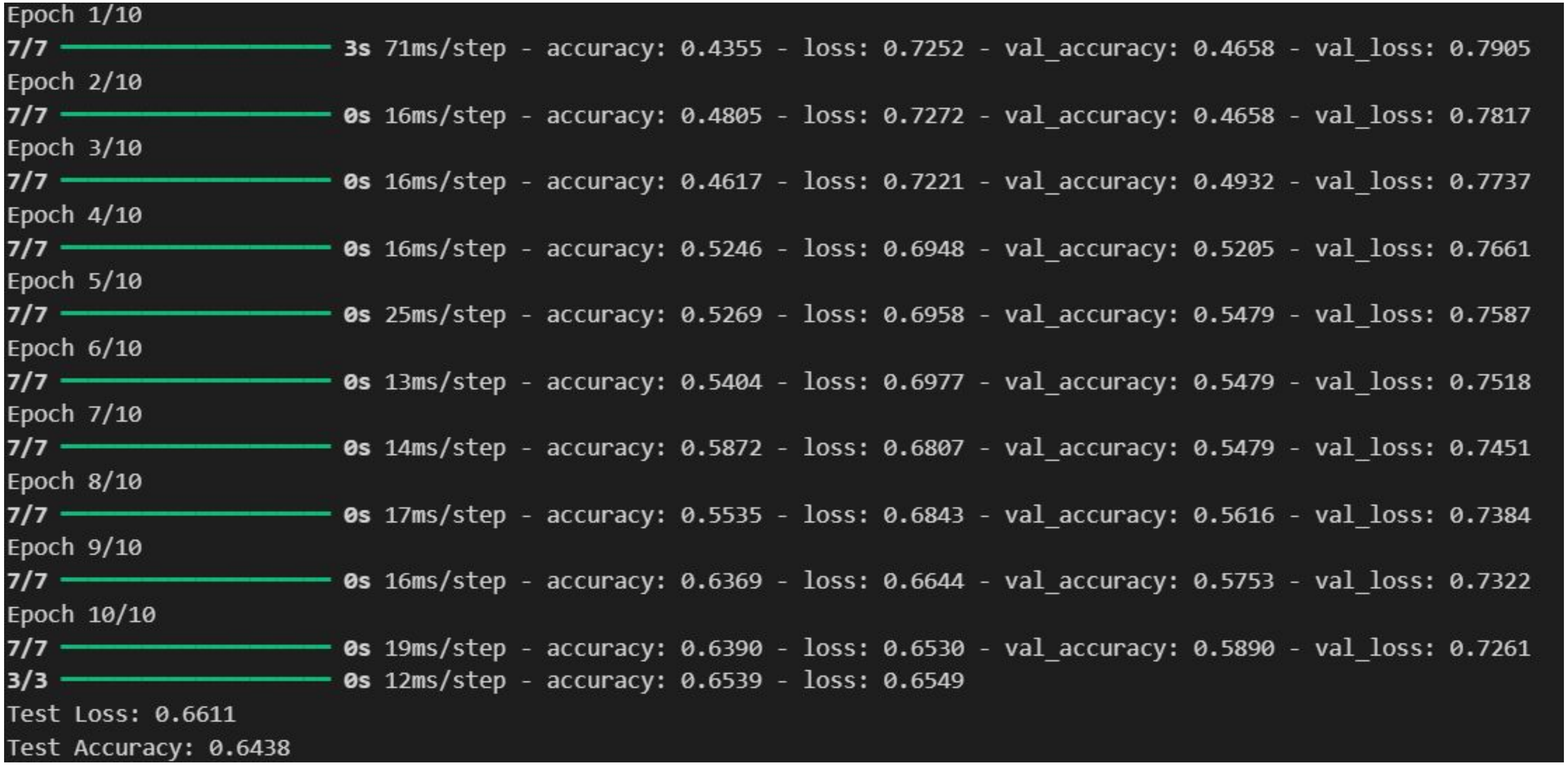
Precision: 0.429
Using a KNN algorithm, it predicted positive values correctly about 43% of the time. That means the model had a false positive rate of about 57% given the dataframe. This is also not the ideal case since more than half the time, it predicts that rainfall will occur on a day where the weather patterns lead to know rain more that half of the time.

Sensitivity (Recall): 0.75
A Sensitivity of 75% means that the model correctly predicted a majority of the true positive values correctly.

Specificity: 0.79
A Specificity of almost 80% shows that the model correctly predicts true negative values better that the true positive values.

F1: 0.545
An F1 core of 55% shows the balance between the precision and recall

Performance of ANN



```
Epoch 1/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 3s 71ms/step - accuracy: 0.4355 - loss: 0.7252 - val_accuracy: 0.4658 - val_loss: 0.7905
Epoch 2/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - accuracy: 0.4805 - loss: 0.7272 - val_accuracy: 0.4658 - val_loss: 0.7817
Epoch 3/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - accuracy: 0.4617 - loss: 0.7221 - val_accuracy: 0.4932 - val_loss: 0.7737
Epoch 4/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - accuracy: 0.5246 - loss: 0.6948 - val_accuracy: 0.5205 - val_loss: 0.7661
Epoch 5/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step - accuracy: 0.5269 - loss: 0.6958 - val_accuracy: 0.5479 - val_loss: 0.7587
Epoch 6/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - accuracy: 0.5404 - loss: 0.6977 - val_accuracy: 0.5479 - val_loss: 0.7518
Epoch 7/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step - accuracy: 0.5872 - loss: 0.6807 - val_accuracy: 0.5479 - val_loss: 0.7451
Epoch 8/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step - accuracy: 0.5535 - loss: 0.6843 - val_accuracy: 0.5616 - val_loss: 0.7384
Epoch 9/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - accuracy: 0.6369 - loss: 0.6644 - val_accuracy: 0.5753 - val_loss: 0.7322
Epoch 10/10
7/7 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step - accuracy: 0.6390 - loss: 0.6530 - val_accuracy: 0.5890 - val_loss: 0.7261
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 0.6539 - loss: 0.6549
Test Loss: 0.6611
Test Accuracy: 0.6438
```

Performance of ANN (cont.)

Discussion:

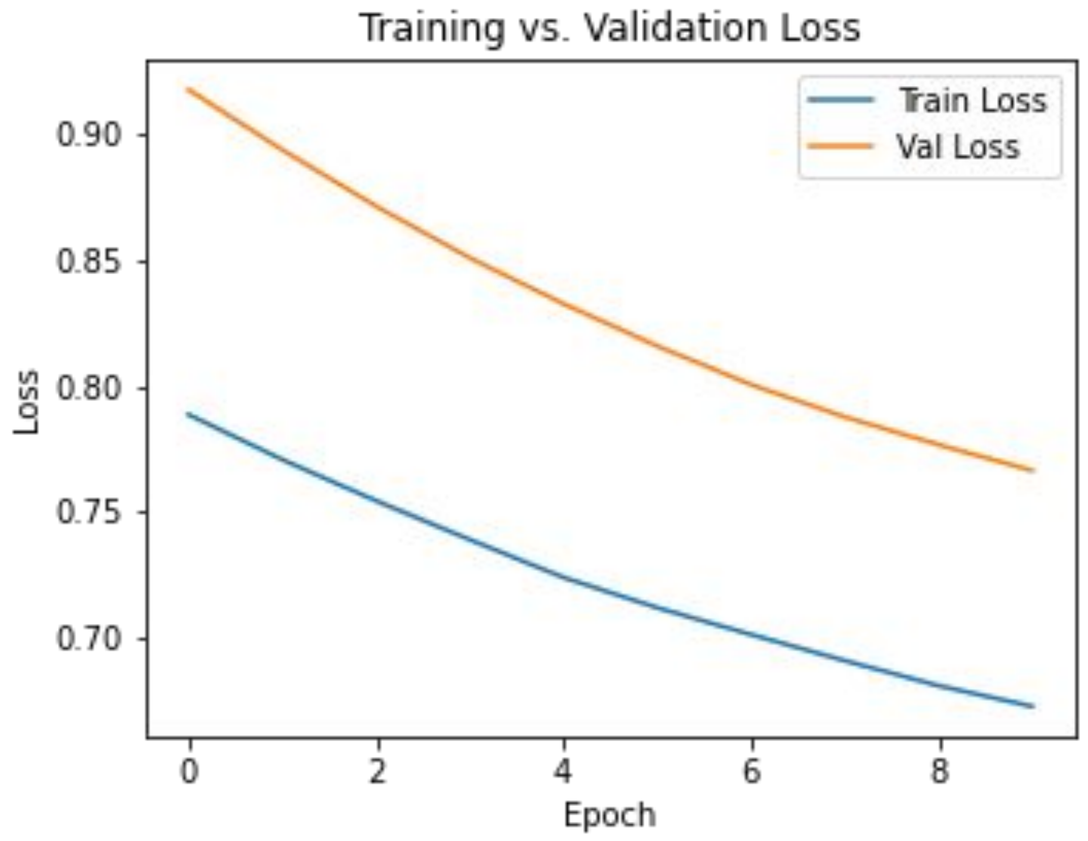
Test Loss: 0.6611

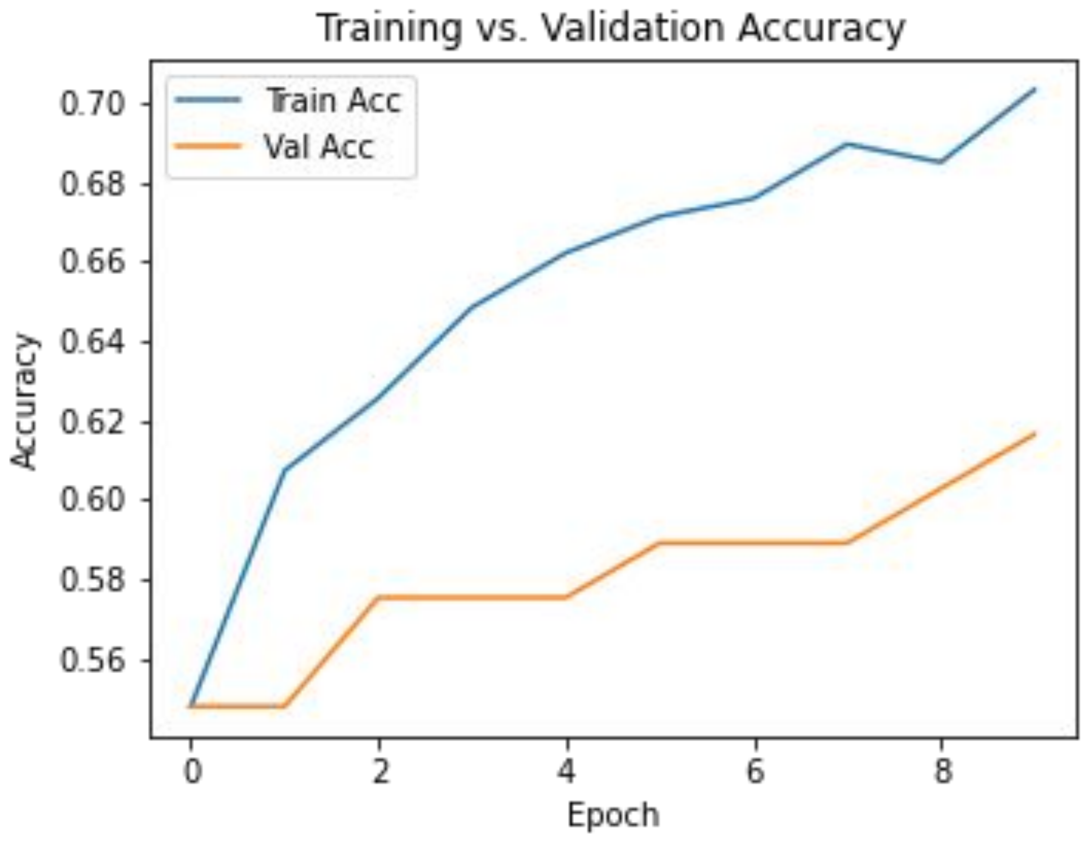
Loss correlates to how well the model can generalize new data given the data that it was trained on. A loss of 66% means that the model does poorly in correctly generalizing new data, and could be caused by overfitting data when training the model.

Test Accuracy: 0.6438

Accuracy is the correct predictions the model makes based on its training. An accuracy of 64% means that there is still room for improvement, and further tweaking could yield better results. This measure is likely due to the high amount of loss that is experienced in the model as well.

Epoch Error Curve





TASK COMPLETION REPORT

DATE	TASK NAME	STATUS	PERSON
2/26/25	CREATE GITHUB REPOSITORY	COMPLETE	SANELE HARMON
2/27/25	PLOT DISTRIBUTIONS OF THE DATA	COMPLETE	TAEDEN KITCHENS
2/29/25	PREPROCESS TEMPERATURE COLUMNS AND PLOT	COMPLETE	SANELE HARMON
2/29/25	PREPROCESS RAINFALL COLUMN AND PLOT	COMPLETE	TAEDEN KITCHENS
4/05/25	Discuss which features to drop	Complete	Sanele Harmon, Taeden Kitchens
4/07/25	Create Heat map	Complete	Taeden Kitchens
4/10/25	Add to PA2 report	Complete	Sanele Harmon
4/2825	Implement SVM and DT	Complete	Sanele Harmon
4/28/25	Implement ANN and KNN	Complete	Taeden Kitchens
5/1/25	Assess the models	Complete	Sanele Harmon, Taeden Kitchens