Weather Data Classification

In [2]:
```
import pandas as pd
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

In [4]:
```
data=pd.read_csv("C:\\Users\\Sushmitha T\\Downloads\\Rinex\\daily_weather .csv")
```

data Description

In [5]:
```
data.columns
```

Out[5]:
```
Index(['number', 'air_pressure_9am', 'air_temp_9am', 'avg_wind_direction_9am',
       'avg_wind_speed_9am', 'max_wind_direction_9am', 'max_wind_speed_9am',
       'rain_accumulation_9am', 'rain_duration_9am', 'relative_humidity_9am',
       'relative_humidity_3pm'],
      dtype='object')
```

In [6]:
```
data.head()
```

Out[6]:

	number	air_pressure_9am	air_temp_9am	avg_wind_direction_9am	avg_wind_speed_9am	max_wind_direction_9am	max_wind_s
0	0	918.060000	74.822000	271.100000	2.080354	295.400000	
1	1	917.347688	71.403843	101.935179	2.443009	140.471549	
2	2	923.040000	60.638000	51.000000	17.067852	63.700000	
3	3	920.502751	70.138895	198.832133	4.337363	211.203341	
4	4	921.160000	44.294000	277.800000	1.856660	136.500000	

Central Tendency and Dispersion

In [8]:
```
mean_values = data.mean()
print("Mean:\n", mean_values)
```

```
Mean:
 number                   547.000000
air_pressure_9am         918.882551
air_temp_9am              64.933001
avg_wind_direction_9am   142.235511
avg_wind_speed_9am         5.508284
max_wind_direction_9am   148.953518
max_wind_speed_9am         7.019514
rain_accumulation_9am      0.203079
rain_duration_9am        294.108052
relative_humidity_9am     34.241402
relative_humidity_3pm     35.344727
dtype: float64
```

In [9]:
```
median_values = data.median()
print("Median:\n", median_values)
```

```
Median:
 number                   547.000000
air_pressure_9am         918.921045
air_temp_9am              65.715479
avg_wind_direction_9am   166.000000
avg_wind_speed_9am         3.871333
max_wind_direction_9am   177.300000
max_wind_speed_9am         4.943637
rain_accumulation_9am      0.000000
rain_duration_9am          0.000000
relative_humidity_9am     23.179259
relative_humidity_3pm     24.380000
dtype: float64
```

In [10]:
```
mode_values = data.mode()
print("Mode:\n", mode_values)
```

```
Mode:
       number  air_pressure_9am  air_temp_9am  avg_wind_direction_9am  \
0          0             917.4        57.398                   171.9
1          1               NaN           NaN                     NaN
2          2               NaN           NaN                     NaN
3          3               NaN           NaN                     NaN
4          4               NaN           NaN                     NaN
...      ...               ...           ...                     ...
1090    1090               NaN           NaN                     NaN
1091    1091               NaN           NaN                     NaN
1092    1092               NaN           NaN                     NaN
1093    1093               NaN           NaN                     NaN
1094    1094               NaN           NaN                     NaN

      avg_wind_speed_9am  max_wind_direction_9am  max_wind_speed_9am  \
0               1.610597                   189.8             2.23694
1               1.722444                   190.8                 NaN
2               2.304048                   191.4                 NaN
3                    NaN                     NaN                 NaN
4                    NaN                     NaN                 NaN
...                  ...                     ...                 ...
1090                 NaN                     NaN                 NaN
1091                 NaN                     NaN                 NaN
1092                 NaN                     NaN                 NaN
1093                 NaN                     NaN                 NaN
1094                 NaN                     NaN                 NaN

      rain_accumulation_9am  rain_duration_9am  relative_humidity_9am  \
0                       0.0                0.0                  89.84
1                       NaN                NaN                    NaN
2                       NaN                NaN                    NaN
3                       NaN                NaN                    NaN
4                       NaN                NaN                    NaN
...                     ...                ...                    ...
1090                    NaN                NaN                    NaN
1091                    NaN                NaN                    NaN
1092                    NaN                NaN                    NaN
1093                    NaN                NaN                    NaN
1094                    NaN                NaN                    NaN

      relative_humidity_3pm
0                     56.93
1                       NaN
2                       NaN
3                       NaN
4                       NaN
...                     ...
1090                    NaN
1091                    NaN
1092                    NaN
1093                    NaN
1094                    NaN

[1095 rows x 11 columns]
```

In [11]:
```
#Dispersion
range_values = data.max() - data.min()
print("Range:\n", range_values)
```

```
Range:
 number                     1094.000000
air_pressure_9am             21.330000
air_temp_9am                 62.154000
avg_wind_direction_9am      327.900000
avg_wind_speed_9am           22.861527
max_wind_direction_9am      283.300000
max_wind_speed_9am           28.655201
rain_accumulation_9am        24.020000
rain_duration_9am         17704.000000
relative_humidity_9am        86.530000
relative_humidity_3pm        86.950000
dtype: float64
```

In [12]:
```
variance_values = data.var()
print("Variance:\n", variance_values)
```

```
Variance:
 number                   1.000100e+05
air_pressure_9am          1.013888e+01
air_temp_9am              1.248921e+02
avg_wind_direction_9am    4.780044e+03
avg_wind_speed_9am        2.072811e+01
max_wind_direction_9am    4.520950e+03
max_wind_speed_9am        3.133995e+01
rain_accumulation_9am     2.540683e+00
rain_duration_9am         2.553856e+06
relative_humidity_9am     6.488262e+02
relative_humidity_3pm     5.073342e+02
dtype: float64
```

In [13]:
```
std_dev_values = data.std()
print("Standard Deviation:\n", std_dev_values)
```

```
Standard Deviation:
 number                     316.243577
air_pressure_9am             3.184161
air_temp_9am                11.175514
avg_wind_direction_9am      69.137859
avg_wind_speed_9am           4.552813
max_wind_direction_9am      67.238013
max_wind_speed_9am           5.598209
rain_accumulation_9am        1.593952
rain_duration_9am         1598.078779
relative_humidity_9am       25.472067
relative_humidity_3pm       22.524079
dtype: float64
```

perform data preprocessing such as nullvalue

In [14]:
```
data[data.isnull().any(axis=1)].head()
```

Out[14]:

	number	air_pressure_9am	air_temp_9am	avg_wind_direction_9am	avg_wind_speed_9am	max_wind_direction_9am	max_wind
16	16	917.890000	NaN	169.200000	2.192201	196.800000	
111	111	915.290000	58.820000	182.600000	15.613841	189.000000	
177	177	915.900000	NaN	183.300000	4.719943	189.900000	
262	262	923.596607	58.380598	47.737753	10.636273	67.145843	
277	277	920.480000	62.600000	194.400000	2.751436	NaN	

In [16]:
```
Q1 = data.quantile(0.25)
Q3 = data.quantile(0.75)
IQR = Q3 - Q1
outliers = ((data < (Q1 - 1.5 * IQR)) | (data > (Q3 + 1.5 * IQR)))
```

In [18]:
```
data_no_outliers = data[~((data< (Q1 - 1.5 * IQR)) | (data > (Q3 + 1.5 * IQR))).any(axis=1)]
```

In [25]:
```
Q1 = data.quantile(0.25)
Q3 = data.quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
data_capped = data.copy()
for column in df.columns:
    data_capped[column] = np.where(data[column] < lower_bound[column], lower_bound[column], data[column])
    data_capped[column] = np.where(data[column] > upper_bound[column], upper_bound[column], data[column])
print(data_capped.head())
```

```
   number  air_pressure_9am  air_temp_9am  avg_wind_direction_9am  \
0     0.0        918.060000     74.822000              271.100000
1     1.0        917.347688     71.403843              101.935179
2     2.0        923.040000     60.638000               51.000000
3     3.0        920.502751     70.138895              198.832133
4     4.0        921.160000     44.294000              277.800000

   avg_wind_speed_9am  max_wind_direction_9am  max_wind_speed_9am  \
0            2.080354              295.400000            2.863283
1            2.443009              140.471549            3.533324
2           14.969755               63.700000           17.768185
3            4.337363              211.203341            5.190045
4            1.856660              136.500000            2.863283

   rain_accumulation_9am  rain_duration_9am  relative_humidity_9am  \
0                    0.0                0.0              42.420000
1                    0.0                0.0              24.328697
2                    0.0                0.0               8.900000
3                    0.0                0.0              12.189102
4                    0.0                0.0              90.861635

   relative_humidity_3pm
0              36.160000
1              19.426597
2              14.460000
3              12.742547
4              76.740000
```

In [29]:
```
del data['number']
before_rows=data.shape[0]
print(before_rows)
after_rows=data.shape[0]
print(after_rows)
```

```
1095
1095
```

In [30]:
```
before_rows-after_rows
```

Out[30]: 0

Classify the values in the final column(y)(I.e. relative humidity at 3pm to be ‘0’ if the value is below 25 and ‘ 1’ if it is above 25)

In [33]:
```
def classify_humidity(value):
    if value<25:
        return 0
    else:
        return 1
data['Humidity_Class'] = data['relative_humidity_3pm'].apply(classify_humidity)
print(data[['relative_humidity_3pm', 'Humidity_Class']].head())
```

```
   relative_humidity_3pm  Humidity_Class
0              36.160000               1
1              19.426597               0
2              14.460000               0
3              12.742547               0
4              76.740000               1
```

split the data into X and Y to make it ready for training purposes.

In [35]:
```
X = data.drop('relative_humidity_3pm', axis=1)
Y = data['relative_humidity_3pm']
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=42)
print(f"X_train shape: {X_train.shape}")
print(f"X_test shape: {X_test.shape}")
print(f"Y_train shape: {Y_train.shape}")
print(f"Y_test shape: {Y_test.shape}")
```

```
X_train shape: (876, 10)
X_test shape: (219, 10)
Y_train shape: (876,)
Y_test shape: (219,)
```

train the data with a Classification Model(say Decision Tree Model) with appropriate train test split.

In [40]:
```
Y = data['relative_humidity_3pm']
Y = Y.apply(lambda x: 0 if x < 25 else 1)
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=42)
model = DecisionTreeClassifier(random_state=42)
model.fit(X_train, Y_train)
Y_pred = model.predict(X_test)
accuracy = accuracy_score(Y_test, Y_pred)
print(f"Accuracy: {accuracy}")
```

```
Accuracy: 1.0
```

Test the data by giving X-test as a parameter. Now you can get the value for Y-predicted, which is your futuristic value.

In [45]:
```
humidity_classifier=DecisionTreeClassifier(max_leaf_nodes=10,random_state=0)
humidity_classifier.fit(X_train,Y_train)
type(humidity_classifier)
Y_predicted=humidity_classifier.predict(X_test)
Y_predicted[:10]
```

Out[45]:
```
array([1, 0, 0, 0, 0, 0, 0, 0, 1, 1])
```

In [ ]:
```
```

In [52]:
```
data['high_humidity_label'] = data['relative_humidity_3pm'].apply(lambda x: 0 if x < 25 else 1)
X = data.drop(['relative_humidity_3pm', 'high_humidity_label'], axis=1)
Y = data['high_humidity_label']
X_train,X_test,Y_train,Y_test= train_test_split(X,Y,test_size=0.2,random_state=42)
model = DecisionTreeClassifier(random_state=42)
model.fit(X_train, Y_train)
Y_pred = model.predict(X_test)
accuracy = accuracy_score(Y_test, Y_pred)
print(f"Accuracy: {accuracy}")
```

```
Accuracy: 1.0
```

In [53]:
```
comparison = pd.DataFrame({'Actual': Y_test, 'Predicted': Y_pred})
print(comparison.head())
```

```
     Actual  Predicted
533       1          1
139       0          0
88        0          0
841       0          0
985       0          0
```

In [54]:
```
accuracy = accuracy_score(Y_test, Y_pred)
print(f"Accuracy: {accuracy}")
```

```
Accuracy: 1.0
```

In [ ]:
```
```