Machine Learning on Weather Data (classification model)

Decision Tree

In [1]:

```
import pandas as pd
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

In [2]:

```
# Creating a Pandas DataFrame from a CSV file
data = pd.read_csv('./weather/daily_weather.csv')
```

In [3]:

```
data.columns
```

Out[3]:

```
Index(['number', 'air_pressure_9am', 'air_temp_9am', 'avg_wind_direction
_9am',
       'avg_wind_speed_9am', 'max_wind_direction_9am', 'max_wind_speed_9
am',
       'rain_accumulation_9am', 'rain_duration_9am', 'relative_humidity_
9am',
       'relative_humidity_3pm'],
      dtype='object')
```

In [4]:

```
data.head()
```

Out[4]:

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

In [5]:

```
del data['number']
```

In [6]:

```
data.isnull().any()
```

Out[6]:

```
air_pressure_9am           True
air_temp_9am               True
avg_wind_direction_9am     True
avg_wind_speed_9am         True
max_wind_direction_9am     True
max_wind_speed_9am         True
rain_accumulation_9am      True
rain_duration_9am          True
relative_humidity_9am     False
relative_humidity_3pm     False
dtype: bool
```

In [7]:

```
# Data Cleaning Steps
before_rows = data.shape[0]
data = data.dropna()
after_rows = data.shape[0]
print("It was deleted %d rows" %(before_rows - after_rows))
```

```
It was deleted 31 rows
```

In [8]:

```
# Binarize the relative_humidity_3pm to 0 or 1.
clean_data = data.copy()
clean_data['high_humidity_label'] = (clean_data['relative_humidity_3pm'] > 24.99)*1
#print(clean_data['high_humidity_label'])
```

In [9]:

```
# Target is stored in 'y'
y=clean_data[['high_humidity_label']].copy()
#y
```

In [10]:

```
# Use 9am Sensor Signals as Features to Predict Humidity at 3pm

morning_features = ['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']

X = clean_data[morning_features].copy()
```

Train / Test (split dataset)

1. Split the dataset into two pieces: **a training set** and **a test set**
2. Train the model on **the training set**
3. Test the model on **the testing set,** and evaluate how well we did.

In [11]:

```
# Step 1: split X and y into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_sta
te=324)
```

Scikit-Learn 4-step Modelling Pattern

Step 1: Import the class you plan to use

In [12]:

```
from sklearn.tree import DecisionTreeClassifier
```

Step 2: Make an Instance of the "Estimator"

In [13]:

```
humidity_classifier = DecisionTreeClassifier(max_leaf_nodes=10, random_state=0)
# You can understand all parameters using the print command
print(humidity_classifier)
```

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=No
ne,
            max_features=None, max_leaf_nodes=10,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=0,
            splitter='best')
```

Step 3: Fit the model with data (aka "model training")

- Model is learning the relationship between X and Y
- Occurs in-place

In [14]:

```
humidity_classifier.fit(X_train, y_train)
```

Out[14]:

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=No
ne,
            max_features=None, max_leaf_nodes=10,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=0,
            splitter='best')
```

Step 4: Predict the response for a new observation

- New observations are called "out-of-sample" data
- Uses the information it learned during the model training process

In [15]:

```
y_pred = humidity_classifier.predict(X_test)
```

Measure Accuracy of the Classifier

In [16]:

```
print(accuracy_score(y_test, y_pred))
```

```
0.8153409090909091
```