Run the Cell to import the packages

In [1]:

```
import pandas as pd
import numpy as np
import dataframe as df
```

Data Loading Fill in the Command to load your CSV dataset "weather.csv" with pandas

In [2]:

```
weather = pd.read_csv('weather.csv', sep=',')
```

Data Analysis

- Get the shape of the dataset and print it.
- Get the column names in list and print it.
- Describe the dataset to understand the basic statistics of the dataset.
- Print the first three rows of the dataset

```
In [3]: data_size=weather.shape

        print(data_size)

        weather_col_names = list(weather.columns)

        print(weather_col_names)

        print(weather.describe())

        print(weather.head(3))
```

```
(25000, 25)
['Unnamed: 0', 'Date', 'Location', 'MinTemp', 'MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine', 'WindGustDir',
'WindGustSpeed', 'WindDir9am', 'WindDir3pm', 'WindSpeed9am', 'WindSpeed3pm', 'Humidity9am', 'Humidity3pm', 'Pres
sure9am', 'Pressure3pm', 'Cloud9am', 'Cloud3pm', 'Temp9am', 'Temp3pm', 'RainToday', 'RISK_MM', 'RainTomorrow']
         Unnamed: 0       MinTemp       MaxTemp      Rainfall  Evaporation  \
count  25000.000000  24669.000000  24824.000000  24721.000000  9432.000000
mean   12499.500000     13.294568     23.990558      2.674467     5.825138
std     7217.022701      5.848304      6.114348      9.720306     4.871567
min        0.000000     -3.300000      6.800000      0.000000     0.000000
25%     6249.750000      8.900000     19.500000      0.000000     3.000000
50%    12499.500000     14.000000     23.400000      0.000000     4.800000
75%    18749.250000     17.900000     27.700000      0.600000     7.200000
max    24999.000000     29.700000     47.300000    371.000000    86.200000

          Sunshine  WindGustSpeed  WindSpeed9am  WindSpeed3pm   Humidity9am  \
count  6664.000000   21545.000000  24428.000000  23770.000000  24609.000000
mean      7.811945      37.772755     12.686917     16.837106     69.822951
std       3.718698      13.212331      9.136115      9.095719     17.755908
min       0.000000       7.000000      0.000000      0.000000      3.000000
25%       5.500000      28.000000      6.000000      9.000000     58.000000
50%       8.900000      35.000000     11.000000     17.000000     71.000000
75%      10.600000      46.000000     19.000000     22.000000     83.000000
max      14.000000     135.000000    130.000000     83.000000    100.000000

        Humidity3pm   Pressure9am   Pressure3pm      Cloud9am      Cloud3pm  \
count  23936.000000  20172.000000  20173.000000  14136.000000  13815.000000
mean      52.762826   1018.173290   1015.627438      4.251556      4.409265
std       21.210121      6.481112      6.394829      2.968785      2.719235
min        1.000000    980.500000    979.000000      0.000000      0.000000
25%       37.000000   1013.800000   1011.300000      1.000000      2.000000
50%       54.000000   1018.200000   1015.700000      5.000000      5.000000
75%       68.000000   1022.600000   1020.000000      7.000000      7.000000
max      100.000000   1039.900000   1036.800000      8.000000      8.000000

            Temp9am       Temp3pm       RISK_MM
count  24755.000000  24082.000000  25000.000000
mean      17.953084     22.507171      2.677376
std        5.394685      5.954540      9.705604
min        0.300000      6.400000      0.000000
25%       14.200000     18.100000      0.000000
50%       18.400000     21.900000      0.000000
75%       21.900000     26.100000      0.800000
max       37.700000     46.700000    371.000000
   Unnamed: 0        Date Location  MinTemp  MaxTemp  Rainfall  Evaporation  \
0           0  2008-12-01   Albury     13.4     22.9       0.6          NaN
1           1  2008-12-02   Albury      7.4     25.1       0.0          NaN
2           2  2008-12-03   Albury     12.9     25.7       0.0          NaN

   Sunshine WindGustDir  WindGustSpeed  ... Humidity3pm Pressure9am  \
0       NaN           W           44.0  ...        22.0      1007.7
1       NaN         WNW           44.0  ...        25.0      1010.6
2       NaN         WSW           46.0  ...        30.0      1007.6

   Pressure3pm  Cloud9am  Cloud3pm  Temp9am  Temp3pm  RainToday  RISK_MM  \
0       1007.1       8.0       NaN     16.9     21.8         No      0.0
1       1007.8       NaN       NaN     17.2     24.3         No      0.0
2       1008.7       NaN       2.0     21.0     23.2         No      0.0

   RainTomorrow
0            No
1            No
2            No

[3 rows x 25 columns]
```

Target Identification

Execute the below cell to identify the target variables. If yes it will Rain Tommorow otherwise it will not Rain.

In [4]:

```
weather_target=weather['RainTomorrow']

print(weather_target)
```

```
0         No
1         No
2         No
3         No
4         No
5         No
6         No
7         No
8        Yes
9         No
10       Yes
11       Yes
12       Yes
13        No
14        No
15       Yes
16       Yes
17        No
18        No
19        No
20        No
21        No
22        No
23        No
24        No
25        No
26        No
27       Yes
28        No
29        No
        ... 
24970     No
24971     No
24972     No
24973     No
24974     No
24975     No
24976     No
24977     No
24978     No
24979     No
24980     No
24981    Yes
24982    Yes
24983    Yes
24984    Yes
24985     No
24986    Yes
24987    Yes
24988    Yes
24989    Yes
24990     No
24991     No
24992     No
24993     No
24994     No
24995     No
24996     No
24997     No
24998     No
24999     No
Name: RainTomorrow, Length: 25000, dtype: object
```

Feature Identification

In our case by analyzing the dataset, we can understand that the columns like **Date** might be irrelevant as they are not dependent on call usage pattern.

Since **RainTomorrow** is our target variable, we will be removing it from the feature set.

- Perform appropriate operation to drop the columns **Date** and **RainTomorrow**

In [5]:
```
cols_to_drop = ['Date','RainTomorrow']

weather_feature = weather.drop(cols_to_drop,axis = 1)

print(weather_feature.head(5))
```

```
   Unnamed: 0 Location  MinTemp  MaxTemp  Rainfall  Evaporation  Sunshine  \
0           0   Albury     13.4     22.9       0.6          NaN       NaN   
1           1   Albury      7.4     25.1       0.0          NaN       NaN   
2           2   Albury     12.9     25.7       0.0          NaN       NaN   
3           3   Albury      9.2     28.0       0.0          NaN       NaN   
4           4   Albury     17.5     32.3       1.0          NaN       NaN   

  WindGustDir  WindGustSpeed WindDir9am  ... Humidity9am  Humidity3pm  \
0           W           44.0          W  ...        71.0         22.0   
1         WNW           44.0        NNW  ...        44.0         25.0   
2         WSW           46.0          W  ...        38.0         30.0   
3          NE           24.0         SE  ...        45.0         16.0   
4           W           41.0        ENE  ...        82.0         33.0   

   Pressure9am  Pressure3pm  Cloud9am  Cloud3pm  Temp9am  Temp3pm  RainToday  \
0       1007.7       1007.1       8.0       NaN     16.9     21.8         No   
1       1010.6       1007.8       NaN       NaN     17.2     24.3         No   
2       1007.6       1008.7       NaN       2.0     21.0     23.2         No   
3       1017.6       1012.8       NaN       NaN     18.1     26.5         No   
4       1010.8       1006.0       7.0       8.0     17.8     29.7         No   

   RISK_MM  
0      0.0  
1      0.0  
2      0.0  
3      1.0  
4      0.2  

[5 rows x 23 columns]
```

Categorical Data

In order to Identify the categorical variable in a data, use the following command in the below cell,

In [6]:
```
weather_categorical = weather.select_dtypes(include=[object])
print(weather_categorical.head(15))
```

```
          Date Location WindGustDir WindDir9am WindDir3pm RainToday  \
0   2008-12-01   Albury           W          W        WNW        No   
1   2008-12-02   Albury         WNW        NNW        WSW        No   
2   2008-12-03   Albury         WSW          W        WSW        No   
3   2008-12-04   Albury          NE         SE          E        No   
4   2008-12-05   Albury           W        ENE         NW        No   
5   2008-12-06   Albury         WNW          W          W        No   
6   2008-12-07   Albury           W         SW          W        No   
7   2008-12-08   Albury           W        SSE          W        No   
8   2008-12-09   Albury         NNW         SE         NW        No   
9   2008-12-10   Albury           W          S        SSE       Yes   
10  2008-12-11   Albury           N        SSE        ESE        No   
11  2008-12-12   Albury         NNE         NE        ENE       Yes   
12  2008-12-13   Albury           W        NNW        NNW       Yes   
13  2008-12-14   Albury          SW          W        SSW       Yes   
14  2008-12-16   Albury         WNW        NaN        WNW       NaN   

   RainTomorrow  
0            No  
1            No  
2            No  
3            No  
4            No  
5            No  
6            No  
7            No  
8           Yes  
9            No  
10          Yes  
11          Yes  
12          Yes  
13           No  
14           No  
```

Convert to boolean

Assign the column **RainToday** for the variable **yes_no_cols** and run the below cell to print first 5 rows of **weather_feature**

In [7]:

```
yes_no_cols = ["RainToday"]

weather_feature[yes_no_cols] = weather_feature[yes_no_cols] == 'Yes'

print(weather_feature.head(5))
```

```
   Unnamed: 0 Location  MinTemp  MaxTemp  Rainfall  Evaporation  Sunshine  \
0           0   Albury     13.4     22.9       0.6          NaN       NaN
1           1   Albury      7.4     25.1       0.0          NaN       NaN
2           2   Albury     12.9     25.7       0.0          NaN       NaN
3           3   Albury      9.2     28.0       0.0          NaN       NaN
4           4   Albury     17.5     32.3       1.0          NaN       NaN

  WindGustDir  WindGustSpeed WindDir9am  ... Humidity9am  Humidity3pm  \
0           W           44.0          W  ...        71.0         22.0
1         WNW           44.0        NNW  ...        44.0         25.0
2         WSW           46.0          W  ...        38.0         30.0
3          NE           24.0         SE  ...        45.0         16.0
4           W           41.0        ENE  ...        82.0         33.0

   Pressure9am  Pressure3pm  Cloud9am  Cloud3pm  Temp9am  Temp3pm  RainToday  \
0       1007.7       1007.1       8.0       NaN     16.9     21.8      False
1       1010.6       1007.8       NaN       NaN     17.2     24.3      False
2       1007.6       1008.7       NaN       2.0     21.0     23.2      False
3       1017.6       1012.8       NaN       NaN     18.1     26.5      False
4       1010.8       1006.0       7.0       8.0     17.8     29.7      False

   RISK_MM
0      0.0
1      0.0
2      0.0
3      1.0
4      0.2

[5 rows x 23 columns]
```

One Hot Encoding

Execute the below cells to perform **One Hot Encoding**

In [8]:

```
weather_dumm=pd.get_dummies(weather_feature, columns=["Location","WindGustDir","WindDir9am","WindDir3pm"], prefix

weather_matrix = weather_dumm.values.astype(np.float)
```

Imputing-Missing Values

Do the Imputing-Missing Values by using the following parameters

- missing_values=np.nan
- strategy=mean
- fill_value=None
- verbose=0
- copy=True

In [9]:

```
from sklearn.impute import SimpleImputer

imp=SimpleImputer(missing_values=np.nan,strategy='mean', fill_value=None,verbose=0,copy=True)

weather_matrix=imp.fit_transform(weather_matrix)
```

Standardization

Run the below cell to perform standardization

In [10]:

```
from sklearn.preprocessing import StandardScaler

#Standardize the data by removing the mean and scaling to unit variance

scaler = StandardScaler()

#Fit to data, then transform it.

weather_matrix = scaler.fit_transform(weather_matrix)
```

Train and Test Data

Splitting the data for training and testing(90% train,10% test)

- Perform train-test split on **weather_matrix** and **weather_target** with 90% as train data and 10% as test data and set random_state as seed.

In [11]:

```
from sklearn.model_selection import train_test_split

seed=5000

train_data,test_data, train_label, test_label = train_test_split(weather_matrix,weather_target,test_size=0.1,rando
```

Decision Tree Classification

- Initialize **SVM** classifier with following parameters
  - kernel = linear
  - C= 0.025
  - random_state=seed
- Train the model with train_data and train_label
- Now predict the output with test_data
- Evaluate the classifier with score from test_data and test_label
- Print the predicted score

In [12]:

```
from sklearn.svm import SVC

classifier = SVC(kernel="linear",C=0.025,random_state=seed )

classifier = classifier.fit(train_data,train_label)

churn_predicted_target=classifier.predict(test_data)

score = classifier.score(test_data,test_label)

print('SVM Classifier : ',score)

with open('output.txt', 'w') as file:

    file.write(str(np.mean(score)))
```

```
SVM Classifier :  0.9648
```

Random Forest Classifier

- Do the **Random Forest** Classifier of the Dataset using the following parameters.
  - max_depth=5
  - n_estimators=10
  - max_features=10
  - random_state=seed
- Train the model with train_data and train_label.
- Now predict the output with test_data.
- Evaluate the classifier with score from test_data and test_label.

In [14]:

```
from sklearn.ensemble import RandomForestClassifier

classifier = RandomForestClassifier(max_depth=5,n_estimators=10,max_features=10,random_state=seed)

classifier = classifier.fit(train_data,train_label)

churn_predicted_target=classifier.predict(test_data)

score = classifier.score(test_data,test_label)

print('Random Forest Classifier : ',score)

with open('output1.txt', 'w') as file:

    file.write(str(np.mean(score)))
```

```
Random Forest Classifier :  0.9484
```

In [ ]: