```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

data = pd.read_csv("Dataset11-Weather-Data.csv")
data.head()
```

```
       Date/Time  Temp_C  Dew Point Temp_C  Rel Hum_%  Wind Speed_km/h
\
0  1/1/2012 0:00    -1.8              -3.9         86                4

1  1/1/2012 1:00    -1.8              -3.7         87                4

2  1/1/2012 2:00    -1.8              -3.4         89                7

3  1/1/2012 3:00    -1.5              -3.2         88                6

4  1/1/2012 4:00    -1.5              -3.3         88                7


   Visibility_km  Press_kPa               Weather
0            8.0     101.24                   Fog
1            8.0     101.24                   Fog
2            4.0     101.26  Freezing Drizzle,Fog
3            4.0     101.27  Freezing Drizzle,Fog
4            4.8     101.23                   Fog
```

```
print(data.shape)
print(" ")
print(data.columns)
print(" ")
print(data.dtypes)
print(" ")
print(data.info())
```

```
(8784, 8)

Index(['Date/Time', 'Temp_C', 'Dew Point Temp_C', 'Rel Hum_%',
       'Wind Speed_km/h', 'Visibility_km', 'Press_kPa', 'Weather'],
      dtype='object')

Date/Time            object
Temp_C              float64
Dew Point Temp_C    float64
Rel Hum_%             int64
Wind Speed_km/h       int64
Visibility_km       float64
Press_kPa           float64
```

```
Weather               object
dtype: object

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8784 entries, 0 to 8783
Data columns (total 8 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Date/Time         8784 non-null   object 
 1   Temp_C            8784 non-null   float64
 2   Dew Point Temp_C  8784 non-null   float64
 3   Rel Hum_%         8784 non-null   int64  
 4   Wind Speed_km/h   8784 non-null   int64  
 5   Visibility_km     8784 non-null   float64
 6   Press_kPa         8784 non-null   float64
 7   Weather           8784 non-null   object 
dtypes: float64(4), int64(2), object(2)
memory usage: 549.1+ KB
None

data.Weather.value_counts()

Weather
Mainly Clear                               2106
Mostly Cloudy                              2069
Cloudy                                     1728
Clear                                      1326
Snow                                        390
Rain                                        306
Rain Showers                                188
Fog                                         150
Rain,Fog                                    116
Drizzle,Fog                                  80
Snow Showers                                 60
Drizzle                                      41
Snow,Fog                                     37
Snow,Blowing Snow                            19
Rain,Snow                                    18
Thunderstorms,Rain Showers                   16
Haze                                         16
Drizzle,Snow,Fog                             15
Freezing Rain                                14
Freezing Drizzle,Snow                        11
Freezing Drizzle                              7
Snow,Ice Pellets                              6
Freezing Drizzle,Fog                          6
Snow,Haze                                     5
Freezing Fog                                  4
Snow Showers,Fog                              4
Moderate Snow                                 4
```

```
Rain,Snow,Ice Pellets                          4
Freezing Rain,Fog                              4
Freezing Drizzle,Haze                          3
Rain,Haze                                      3
Thunderstorms,Rain                             3
Thunderstorms,Rain Showers,Fog                 3
Freezing Rain,Haze                             2
Drizzle,Snow                                   2
Rain Showers,Snow Showers                      2
Thunderstorms                                  2
Moderate Snow,Blowing Snow                     2
Rain Showers,Fog                               1
Thunderstorms,Moderate Rain Showers,Fog        1
Snow Pellets                                   1
Rain,Snow,Fog                                  1
Moderate Rain,Fog                              1
Freezing Rain,Ice Pellets,Fog                  1
Drizzle,Ice Pellets,Fog                        1
Thunderstorms,Rain,Fog                         1
Rain,Ice Pellets                               1
Rain,Snow Grains                               1
Thunderstorms,Heavy Rain Showers               1
Freezing Rain,Snow Grains                      1
Name: count, dtype: int64
```

```
data.Weather.unique()
```

```
array(['Fog', 'Freezing Drizzle,Fog', 'Mostly Cloudy', 'Cloudy',
'Rain',
       'Rain Showers', 'Mainly Clear', 'Snow Showers', 'Snow',
'Clear',
       'Freezing Rain,Fog', 'Freezing Rain', 'Freezing Drizzle',
       'Rain,Snow', 'Moderate Snow', 'Freezing Drizzle,Snow',
       'Freezing Rain,Snow Grains', 'Snow,Blowing Snow', 'Freezing
Fog',
       'Haze', 'Rain,Fog', 'Drizzle,Fog', 'Drizzle',
       'Freezing Drizzle,Haze', 'Freezing Rain,Haze', 'Snow,Haze',
       'Snow,Fog', 'Snow,Ice Pellets', 'Rain,Haze',
'Thunderstorms,Rain',
       'Thunderstorms,Rain Showers', 'Thunderstorms,Heavy Rain
Showers',
       'Thunderstorms,Rain Showers,Fog', 'Thunderstorms',
       'Thunderstorms,Rain,Fog',
       'Thunderstorms,Moderate Rain Showers,Fog', 'Rain Showers,Fog',
       'Rain Showers,Snow Showers', 'Snow Pellets', 'Rain,Snow,Fog',
       'Moderate Rain,Fog', 'Freezing Rain,Ice Pellets,Fog',
       'Drizzle,Ice Pellets,Fog', 'Drizzle,Snow', 'Rain,Ice Pellets',
       'Drizzle,Snow,Fog', 'Rain,Snow Grains', 'Rain,Snow,Ice
Pellets',
```

```
       'Snow Showers,Fog', 'Moderate Snow,Blowing Snow'],
dtype=object)
```

```
data.Weather.nunique()
```

```
50
```

```
x='Thunderstorms,Moderate Rain Showers,Fog'
```

```
list_of_lists = [w.split() for w in x.split(',')]
list_of_lists
```

```
[['Thunderstorms'], ['Moderate', 'Rain', 'Showers'], ['Fog']]
```

```
from itertools import chain
flat_list = list(chain(*list_of_lists))
flat_list
```

```
['Thunderstorms', 'Moderate', 'Rain', 'Showers', 'Fog']
```

```
def create_list(x):
    list_of_lists = [w.split() for w in x.split(',')]
    flat_list = list(chain(*list_of_lists))
    return flat_list

def get_weather(list1):
    if 'Fog' in list1 and 'Rain' in list1:
        return "RAIN+FOG"
    elif 'Snow' in list1 and 'Rain' in list1:
        return "SNOW+RAIN"
    elif 'Snow' in list1:
        return "SNOW"
    elif 'Fog' in list1:
        return "FOG"
    elif 'Rain' in list1:
        return "RAIN"
    elif 'Clear' in list1:
        return "CLEAR"
    elif "Cloudy" in list1:
        return 'CLOUDY'
    elif "Thunderstorms" in list1:
        return 'THUNDERSTORMS'
    else:
        return "RAIN"

get_weather(create_list(x))
```

```
'RAIN+FOG'
```

```
data['Std_Weather']=data['Weather'].apply(lambda x:
get_weather(create_list(x)))
```

```
data.head()
```

```
       Date/Time  Temp_C  Dew Point Temp_C  Rel Hum_%  Wind Speed_km/h
\
0  1/1/2012 0:00    -1.8              -3.9         86                4

1  1/1/2012 1:00    -1.8              -3.7         87                4

2  1/1/2012 2:00    -1.8              -3.4         89                7

3  1/1/2012 3:00    -1.5              -3.2         88                6

4  1/1/2012 4:00    -1.5              -3.3         88                7


   Visibility_km  Press_kPa               Weather Std_Weather
0            8.0     101.24                   Fog         FOG
1            8.0     101.24                   Fog         FOG
2            4.0     101.26  Freezing Drizzle,Fog         FOG
3            4.0     101.27  Freezing Drizzle,Fog         FOG
4            4.8     101.23                   Fog         FOG
```

```
data.Std_Weather.value_counts()
```

```
Std_Weather
CLOUDY           3797
CLEAR            3432
RAIN              601
SNOW              556
FOG               241
RAIN+FOG          129
SNOW+RAIN          26
THUNDERSTORMS       2
Name: count, dtype: int64
```

```
cloudy_df = data[data['Std_Weather']=='CLOUDY'].sample(600)
cloudy_df.shape
```

```
(600, 9)
```

```
clear_df = data[data['Std_Weather']=='CLEAR'].sample(600)
clear_df.shape
```

```
(600, 9)
```

```
rain_df = data[data['Std_Weather']=='RAIN']
rain_df.shape
```

```
(601, 9)
```

```
snow_df = data[data['Std_Weather']=='SNOW']
snow_df.shape
```

```
(556, 9)
```

```
fog_df = data[data['Std_Weather']=='FOG']
fog_df.shape
```

```
(241, 9)
```

```
thunder_df = data[data['Std_Weather']=='THUNDERSTORMS']
thunder_df.shape
```

```
(2, 9)
```

```
weather_df = pd.concat([cloudy_df, clear_df, rain_df,
snow_df,fog_df,thunder_df],axis=0)
weather_df.head()
```

```
            Date/Time  Temp_C  Dew Point Temp_C  Rel Hum_%  Wind
Speed_km/h  \
5613  8/21/2012 21:00    19.2              12.9         67
20
1503   3/3/2012 15:00     5.3              -5.6         45
57
6072   9/10/2012 0:00    13.4              10.2         81
9
4069  6/18/2012 13:00    25.3              14.2         50
20
6746   10/8/2012 2:00     5.4               3.0         84
6

      Visibility_km  Press_kPa        Weather Std_Weather
5613           24.1     101.36  Mostly Cloudy      CLOUDY
1503           24.1      98.67  Mostly Cloudy      CLOUDY
6072           25.0     100.94  Mostly Cloudy      CLOUDY
4069           24.1     101.06  Mostly Cloudy      CLOUDY
6746           25.0     101.50  Mostly Cloudy      CLOUDY
```

```
weather_df.shape
```

```
(2600, 9)
```

```
weather_df.Std_Weather.value_counts()
```

```
Std_Weather
RAIN             601
CLOUDY           600
CLEAR            600
SNOW             556
FOG              241
THUNDERSTORMS      2
Name: count, dtype: int64
```

```
weather_df.drop(columns=['Date/Time','Weather'],axis=1,inplace=True)
```

```
weather_df
```

```
      Temp_C  Dew Point Temp_C  Rel Hum_%  Wind Speed_km/h
Visibility_km  \
5613    19.2              12.9         67               20
24.1
1503     5.3              -5.6         45               57
24.1
6072    13.4              10.2         81                9
25.0
4069    25.3              14.2         50               20
24.1
6746     5.4               3.0         84                6
25.0
...      ...               ...        ...              ...
...
8718   -13.8             -15.3         88                4
9.7
8719   -14.8             -16.4         88                7
8.0
8722   -12.0             -13.3         90                7
6.4
4456    26.7              20.1         67               15
24.1
4729    21.6              19.4         87                0
25.0

      Press_kPa    Std_Weather
5613     101.36         CLOUDY
1503      98.67         CLOUDY
6072     100.94         CLOUDY
4069     101.06         CLOUDY
6746     101.50         CLOUDY
...         ...            ...
8718     101.25            FOG
8719     101.22            FOG
8722     101.15            FOG
4456      99.84  THUNDERSTORMS
4729     100.62  THUNDERSTORMS

[2600 rows x 7 columns]
```

```
weather_df[weather_df.duplicated()]
```

```
Empty DataFrame
Columns: [Temp_C, Dew Point Temp_C, Rel Hum_%, Wind Speed_km/h,
Visibility_km, Press_kPa, Std_Weather]
Index: []
```

```
weather_df.isnull().sum()
```

```
Temp_C               0
Dew Point Temp_C     0
Rel Hum_%            0
Wind Speed_km/h      0
Visibility_km        0
Press_kPa            0
Std_Weather          0
dtype: int64
```

```
weather_df.describe()
```

```
            Temp_C  Dew Point Temp_C    Rel Hum_%  Wind Speed_km/h  \
count  2600.000000       2600.000000  2600.000000      2600.000000
mean      6.399923          1.708577    74.239615        15.881154
std      10.926134         10.288582    16.412344         9.219696
min     -21.400000        -26.800000    20.000000         0.000000
25%      -2.100000         -6.325000    63.000000         9.000000
50%       5.300000          0.900000    78.000000        15.000000
75%      15.700000         10.500000    87.000000        22.000000
max      32.800000         24.400000   100.000000        70.000000

       Visibility_km    Press_kPa
count    2600.000000  2600.000000
mean       21.497115   100.833638
std        13.092374     0.913165
min         0.200000    97.520000
25%         9.700000   100.320000
50%        24.100000   100.870000
75%        25.000000   101.410000
max        48.300000   103.650000
```

```
cols=["Temp_C", "Dew Point Temp_C",   "Rel Hum_%",'Wind
Speed_km/h','Visibility_km','Press_kPa']

cor_matrix = weather_df[cols].corr()
cor_matrix
```

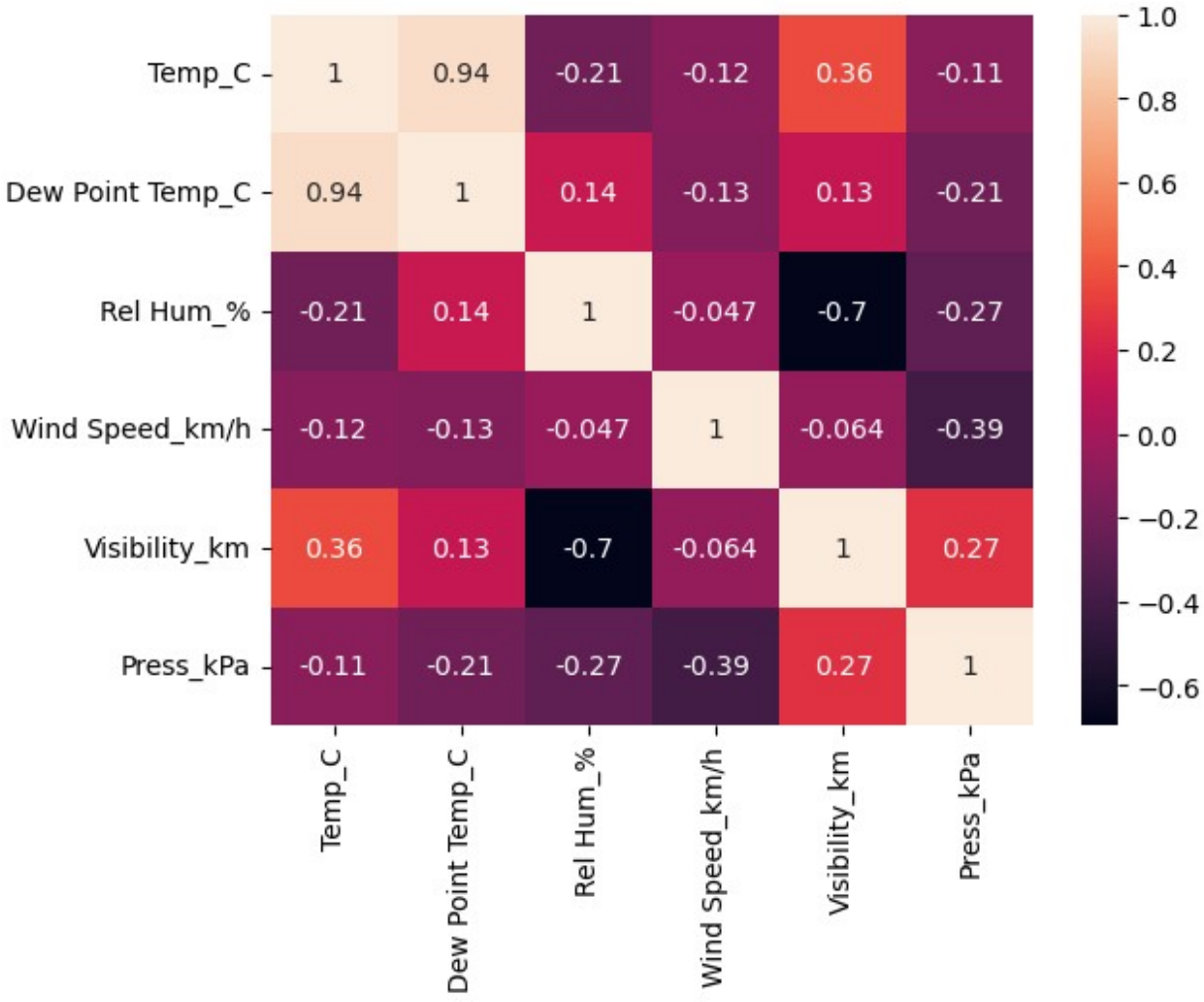
```
                     Temp_C  Dew Point Temp_C  Rel Hum_%  Wind
Speed_km/h  \
Temp_C             1.000000          0.936951  -0.205052        -
0.116830
Dew Point Temp_C   0.936951          1.000000   0.143490        -
0.131767
Rel Hum_%         -0.205052          0.143490   1.000000        -
0.047209
Wind Speed_km/h   -0.116830         -0.131767  -0.047209
1.000000
Visibility_km      0.362525          0.127725  -0.696840        -
0.063581
Press_kPa         -0.111014         -0.205848  -0.273712        -
```

```
0.393850
```

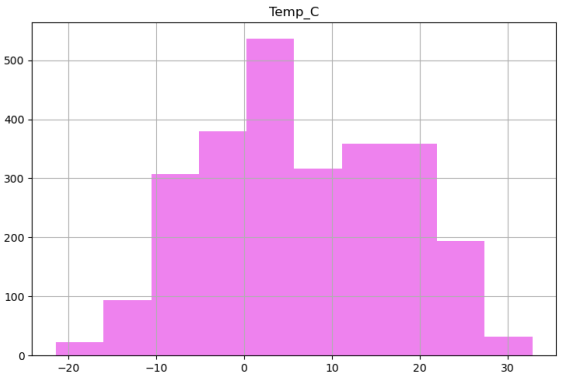
	Visibility_km	Press_kPa
Temp_C	0.362525	-0.111014
Dew Point Temp_C	0.127725	-0.205848
Rel Hum_%	-0.696840	-0.273712
Wind Speed_km/h	-0.063581	-0.393850
Visibility_km	1.000000	0.270757
Press_kPa	0.270757	1.000000

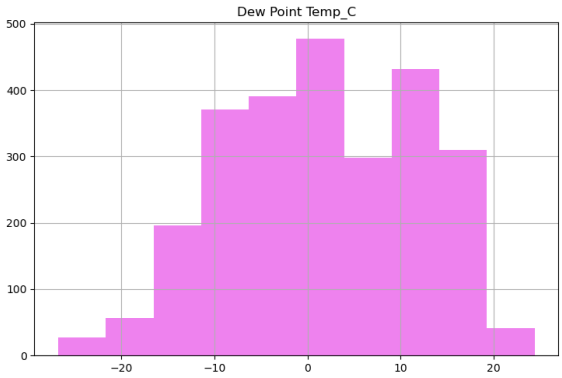
```
sns.heatmap(cor_matrix, annot=True)
```

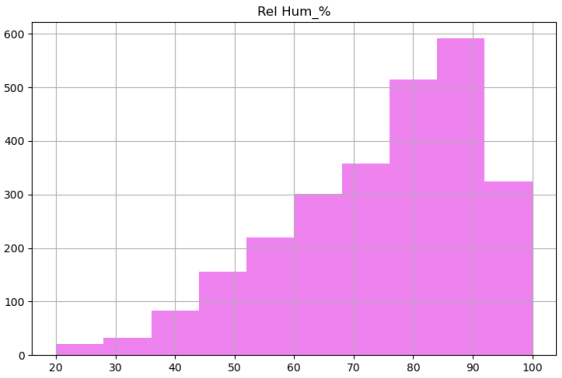
```
<Axes: >
```

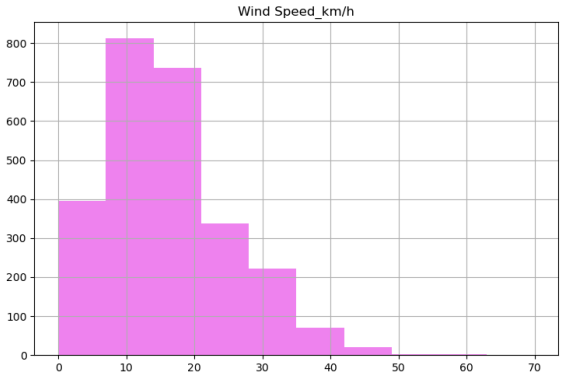


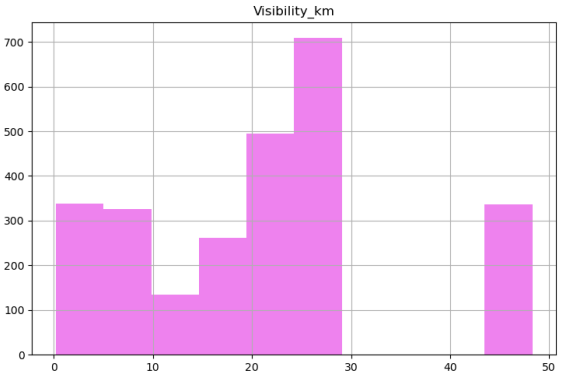

```
data_hist_plot = weather_df.hist(figsize=(20,20),color='VIOLET')
```

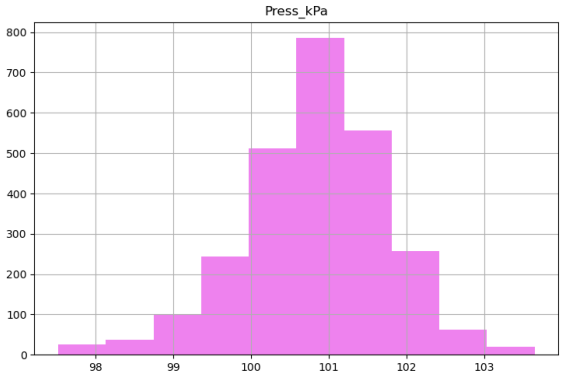










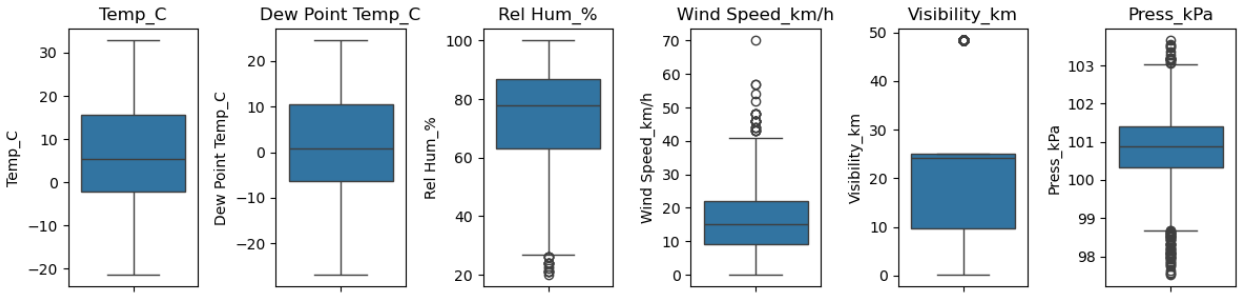



```
num_cols =
weather_df.select_dtypes(exclude=['object']).columns.tolist()
num_cols
```

```
['Temp_C',
 'Dew Point Temp_C',
 'Rel Hum_%',
 'Wind Speed_km/h',
 'Visibility_km',
 'Press_kPa']
```

```
fig, axes = plt.subplots(ncols = 6,figsize=(12,3))
for column,axis in zip(num_cols,axes):
```

```
    sns.boxplot(data= weather_df[column],ax=axis)
    axis.set_title(column)
plt.tight_layout()
plt.show()
```




```
from sklearn.preprocessing import LabelEncoder
label_Encoder = LabelEncoder()

weather_df['Std_Weather'] =
label_Encoder.fit_transform(weather_df['Std_Weather'])

label_Encoder.classes_
```

```
array(['CLEAR', 'CLOUDY', 'FOG', 'RAIN', 'SNOW', 'THUNDERSTORMS'],
      dtype=object)
```

```
cat_code =  dict(zip(label_Encoder.classes_,
label_Encoder.transform(label_Encoder.classes_)))
cat_code
```

```
{'CLEAR': 0, 'CLOUDY': 1, 'FOG': 2, 'RAIN': 3, 'SNOW': 4,
'THUNDERSTORMS': 5}
```

```
weather_df.Std_Weather.value_counts()
```

```
Std_Weather
3    601
1    600
0    600
4    556
2    241
5      2
Name: count, dtype: int64
```

```
X=weather_df.drop(['Std_Weather'],axis=1)
Y=weather_df['Std_Weather']

weather_df.head()
```

```
      Temp_C  Dew Point Temp_C  Rel Hum_%  Wind Speed_km/h
Visibility_km  \
```

```
5613    19.2              12.9         67                20
24.1
1503     5.3              -5.6         45                57
24.1
6072    13.4              10.2         81                 9
25.0
4069    25.3              14.2         50                20
24.1
6746     5.4               3.0         84                 6
25.0

      Press_kPa  Std_Weather
5613     101.36            1
1503      98.67            1
6072     100.94            1
4069     101.06            1
6746     101.50            1
```

```
from sklearn.preprocessing import StandardScaler
std_scaler = StandardScaler()

X_std = std_scaler.fit_transform(X)
X_std
```

```
array([[ 1.17173553,  1.08796095, -0.4411928 ,  0.44683017,
0.19884747,
         0.5765251 ],
       [-0.10068838, -0.71049466, -1.78190514,  4.46074953,
0.19884747,
        -2.36983859],
       [ 0.64079606,  0.82548364,  0.41198778, -0.74649721,  0.267603
,
         0.11649806],
       ...,
       [-1.68435266, -1.45904104,  0.96046101, -0.96346582, -
1.15334469,
         0.34651158],
       [ 1.85829519,  1.78790043, -0.4411928 , -0.09559136,
0.19884747,
        -1.08833468],
       [ 1.39143462,  1.71985076,  0.7776366 , -1.72285597,  0.267603
,
        -0.23399874]])
```

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test =
train_test_split(X_std,Y,test_size=0.2,random_state=42, stratify=Y)
x_train.shape, x_test.shape
```

```
((2080, 6), (520, 6))
```

```
from sklearn.linear_model import LogisticRegression as LR
from sklearn.tree import DecisionTreeClassifier as DT
from sklearn.ensemble import RandomForestClassifier as RF
from sklearn.ensemble import ExtraTreesClassifier as ETC
from sklearn.svm import SVC as SVM
from sklearn.neighbors import KNeighborsClassifier as KNN
from sklearn.ensemble import GradientBoostingClassifier as GBC
from sklearn.naive_bayes import GaussianNB as NB
from sklearn.ensemble import AdaBoostClassifier as ABC
from sklearn.model_selection import cross_val_score

import warnings
warnings.filterwarnings('ignore')

models = [LR,DT,RF,ETC,SVM,KNN,GBC,NB,ABC]
features = X_std
labels= Y
CV = 5
accu_list = []
ModelName = []

for model in models:
    model_instance = model()  # Instantiate the model
    model_name = model_instance.__class__.__name__
    accuracies = cross_val_score(model_instance, features, labels,
scoring='accuracy', cv=CV)
    accu_list.append(accuracies.mean() * 100)
    ModelName.append(model_name)


model_acc_df = pd.DataFrame({"Model":ModelName,
"Cross_Val_Accuracy":accu_list})
model_acc_df
```

	Model	Cross_Val_Accuracy
0	LogisticRegression	58.000000
1	DecisionTreeClassifier	51.961538
2	RandomForestClassifier	58.923077
3	ExtraTreesClassifier	57.307692
4	SVC	58.615385
5	KNeighborsClassifier	52.230769
6	GradientBoostingClassifier	58.807692
7	GaussianNB	57.538462
8	AdaBoostClassifier	43.730769

```
model_acc_df.sort_values(by =['Cross_Val_Accuracy'],ascending=False)
```

	Model	Cross_Val_Accuracy
2	RandomForestClassifier	58.923077
6	GradientBoostingClassifier	58.807692
4	SVC	58.615385

```
0          LogisticRegression            58.000000
7                  GaussianNB            57.538462
3        ExtraTreesClassifier            57.307692
5        KNeighborsClassifier            52.230769
1      DecisionTreeClassifier            51.961538
8          AdaBoostClassifier            43.730769
```

```
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix

from sklearn.ensemble import RandomForestClassifier
RF = RandomForestClassifier()
RF.fit(x_train, y_train)
y_pred_rf=RF.predict(x_test)

print(classification_report(y_test, y_pred_rf))
```

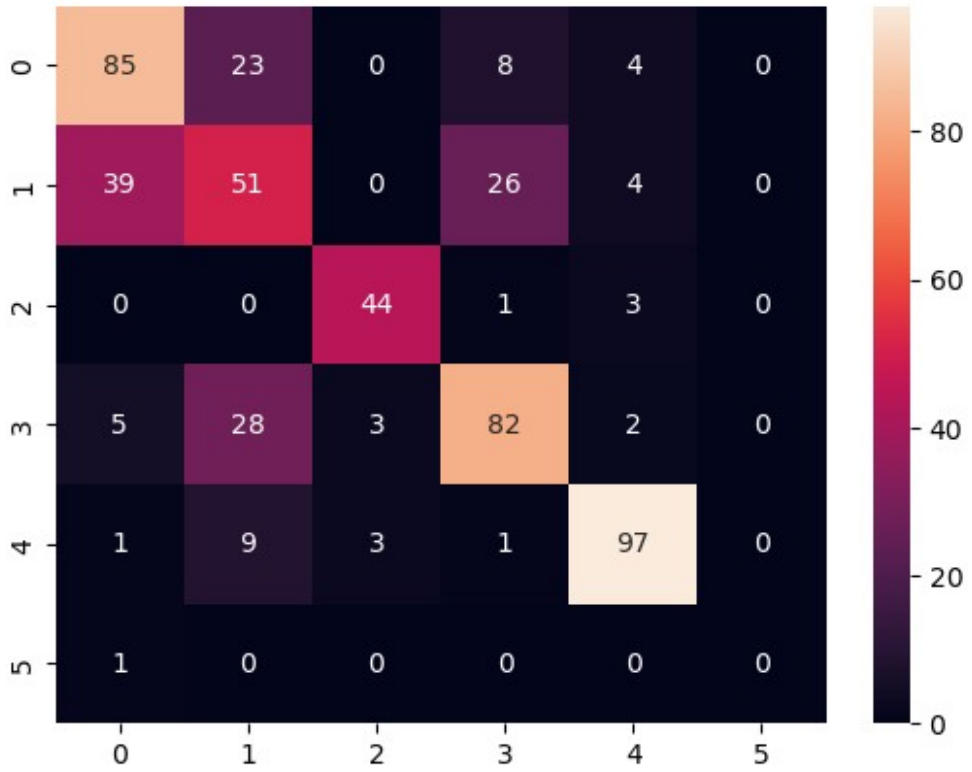
```
              precision    recall  f1-score   support

           0       0.65      0.71      0.68       120
           1       0.46      0.42      0.44       120
           2       0.88      0.92      0.90        48
           3       0.69      0.68      0.69       120
           4       0.88      0.87      0.88       111
           5       0.00      0.00      0.00         1

    accuracy                           0.69       520
   macro avg       0.59      0.60      0.60       520
weighted avg       0.69      0.69      0.69       520
```

```
cm= confusion_matrix(y_test, y_pred_rf)
sns.heatmap(cm,annot=True, fmt='d')
```

```
<Axes: >
```




```
from sklearn.model_selection import GridSearchCV, RandomizedSearchCV

parameters = {
    'n_estimators':[50,100],
    'max_features':['sqrt','log2'],
    'criterion':['gini','entropy']
}

grid_search = GridSearchCV(estimator = RF,
                           param_grid = parameters)

grid_search.fit(x_train, y_train)

GridSearchCV(estimator=RandomForestClassifier(),
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_features': ['sqrt', 'log2'],
                         'n_estimators': [50, 100]})

grid_search.best_params_

{'criterion': 'gini', 'max_features': 'log2', 'n_estimators': 100}

Random_forest_model_new=RandomForestClassifier(criterion = 'gini',
max_features='log2',n_estimators=50)
```

```
Random_forest_model_new.fit(x_train, y_train)
y_pred_rf = Random_forest_model_new.predict(x_test)
accuracy_score(y_test, y_pred_rf)
```

```
0.7
```

```
weather_df
```

```
      Temp_C  Dew Point Temp_C  Rel Hum_%  Wind Speed_km/h
Visibility_km  \
5613    19.2              12.9         67               20
24.1
1503     5.3              -5.6         45               57
24.1
6072    13.4              10.2         81                9
25.0
4069    25.3              14.2         50               20
24.1
6746     5.4               3.0         84                6
25.0
...      ...               ...        ...              ...
...
8718   -13.8             -15.3         88                4
9.7
8719   -14.8             -16.4         88                7
8.0
8722   -12.0             -13.3         90                7
6.4
4456    26.7              20.1         67               15
24.1
4729    21.6              19.4         87                0
25.0

      Press_kPa  Std_Weather
5613     101.36            1
1503      98.67            1
6072     100.94            1
4069     101.06            1
6746     101.50            1
...         ...          ...
8718     101.25            2
8719     101.22            2
8722     101.15            2
4456      99.84            5
4729     100.62            5

[2600 rows x 7 columns]
```

```
Temp=float(input('Enter the Temp_C= '))
dpt=float(input('Enter the Dew Point Temp_C= '))
```

```
rh=float(input('Enter the Relatity humidity %= '))
ws=float(input('Enter the Wind_speed(km/hr) = '))
vs=float(input('Enter the Visibility_km = '))
pr=float(input('Enter the Pressure_Kpa = '))

input_data=[Temp,dpt,rh,ws,vs,pr]
scaled_data = std_scaler.transform([input_data])
prediction = Random_forest_model_new.predict(scaled_data)
prediction
print("====================================\n")
if prediction[0]==0:
    print("Weather is 'CLEAR'")
elif prediction[0]==1:
    print("Weather is 'CLOUDY'")
elif prediction[0]==2:
    print("Weather is 'FOG'")
elif prediction[0]==3:
    print("Weather is 'RAIN'")
elif prediction[0]==4:
    print("Weather is 'SNOW'")
else:
     print("Weather is 'THUNDERSTORMS'")
```

```
Enter the Temp_C=  27
Enter the Dew Point Temp_C=  76
Enter the Relatity humidity %=  987798
Enter the Wind_speed(km/hr) =  755
Enter the Visibility_km =  8776
Enter the Pressure_Kpa =  989

====================================

Weather is 'CLEAR'
```