**1.Training And Testing The Model**

```
from sklearn.ensemble import RandomForestClassifier

from sklearn.ensemble import GradientBoostingClassifier

RFC=RandomForestClassifier()

GBC=GradientBoostingClassifier()

np.any(np.isnan(x))

False

GBC.fit(x_train,y_train)

GradientBoostingClassifier()

RFC.fit(x_train,y_train)

RandomForestClassifier()

data.isnull().any()

Date            False

Location        False

MinTemp         False

MaxTemp         False

Rainfall        False

WindGustSpeed    False

WindSpeed9am     False

WindSpeed3pm     False

Humidity9am      False

Humidity3pm      False

Pressure9am      False

Pressure3pm      False

Temp9am         False
```

Temp3pm False

RainTomorrow False

RainToday False

WindGustDir False

WindDir9am False

WindDir3pm False

dtype: bool

x.isnull().any()

Location False

MinTemp False

MaxTemp False

Rainfall False

WindGustSpeed False

WindSpeed9am False

WindSpeed3pm False

Humidity9am False

Humidity3pm False

Pressure9am False

Pressure3pm False

Temp9am False

Temp3pm False

RainToday False

WindGustDir False

WindDir9am False

WindDir3pm False

dtype: bool

p1=RFC.predict(x_train)

p2=RFC.predict(x_test)

**2.Model Evaluation**

import sklearn.metrics as metrics

Accuracy_score

print(metrics.accuracy_score(y_train,p1))

0.9999472546020359

print(metrics.accuracy_score(y_test,p2))

0.8567460177924681

**3. Save The Model**

import pickle

pickle.dump(RFC,open('rainfall.pkl','wb'))

pickle.dump(LE,open('encoder.pkl','wb'))

pickle.dump(imp_mode,open('imputer.pkl','wb'))

pickle.dump(sc,open('scale.pkl