**PROJECT : Forest Fires Prediction**

By Samridhi

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sb

import pandas as pd

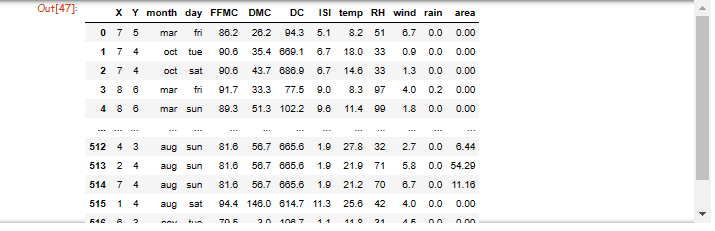
%matplotlib inline

dataset = pd.read\_csv('forestfires.csv')

X = dataset.iloc[:, 0:12].values

y = dataset.iloc[:, 12].values

dataset

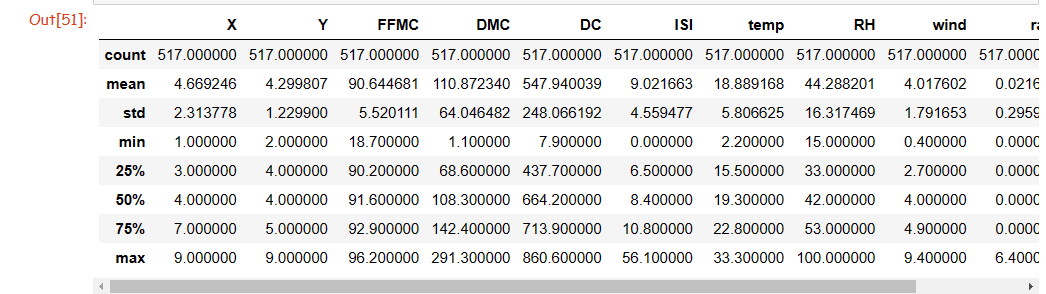


print(dataset.columns)

dataset.shape

print(list(dataset.isnull().any()))

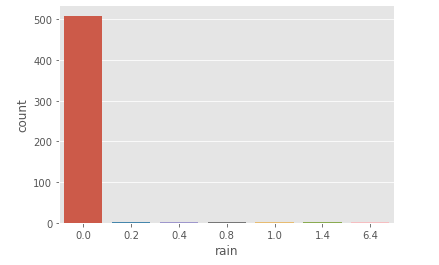
dataset.describe()



dataset.rain.value\_counts()

sb.countplot(x='rain', data=dataset)

plt.show()

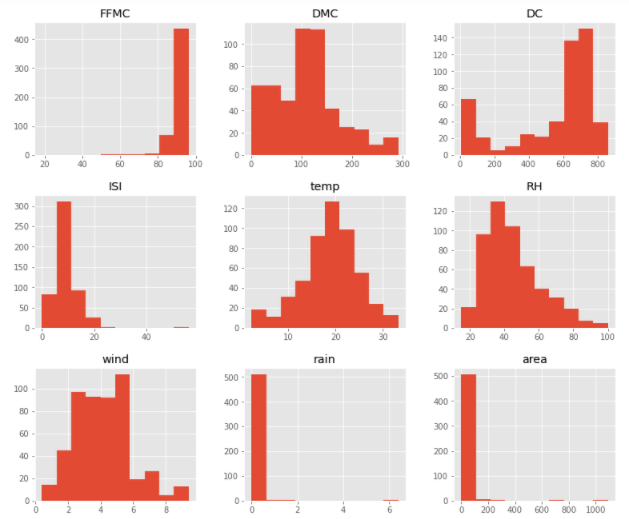


col= ['FFMC', 'DMC', 'DC', 'ISI', 'temp', 'RH','wind', 'rain', 'area']

train = dataset[col]

train.hist(figsize=(13, 11))

plt.show()



plt.style.use('ggplot')

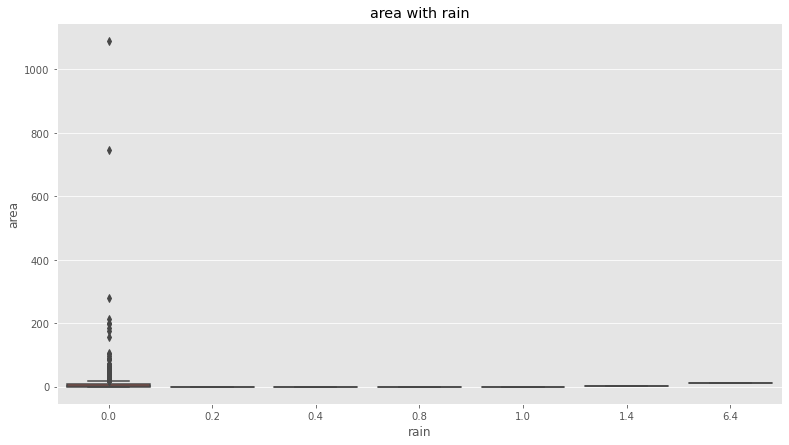
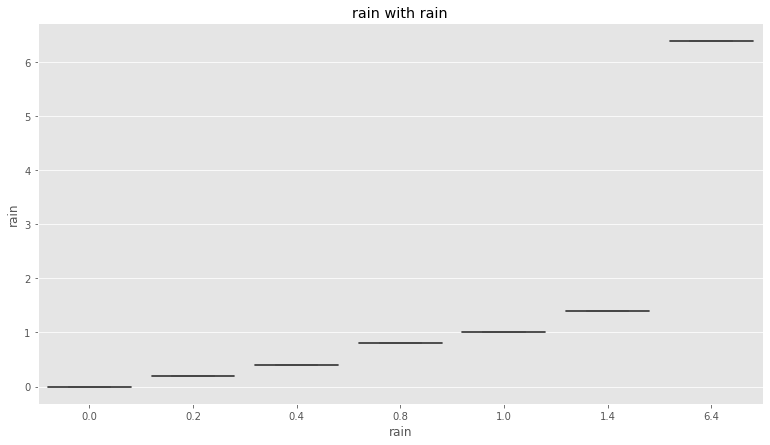
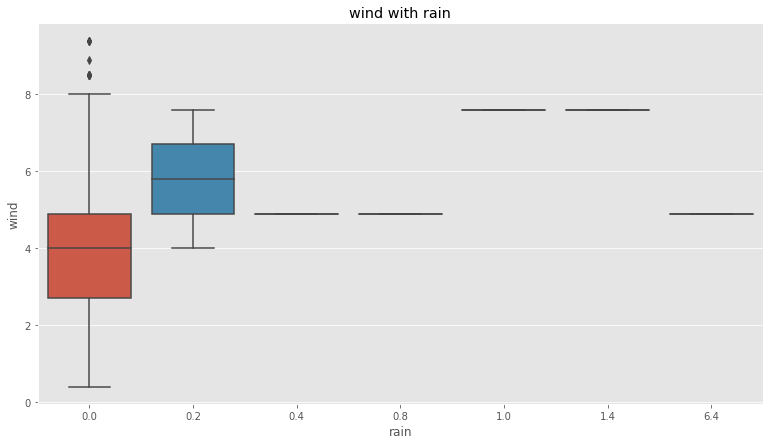
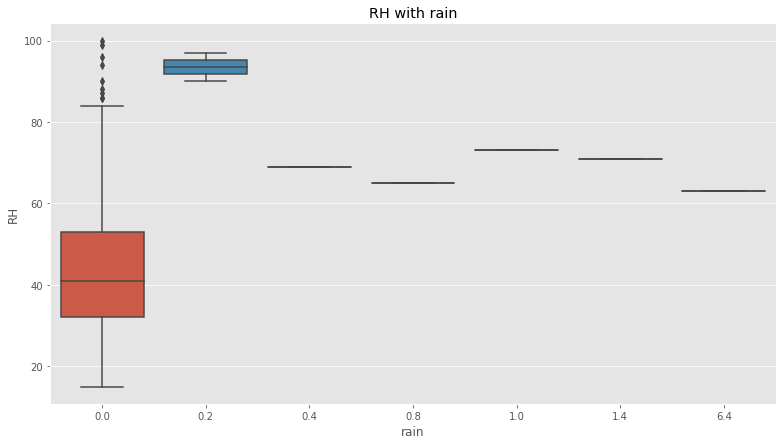
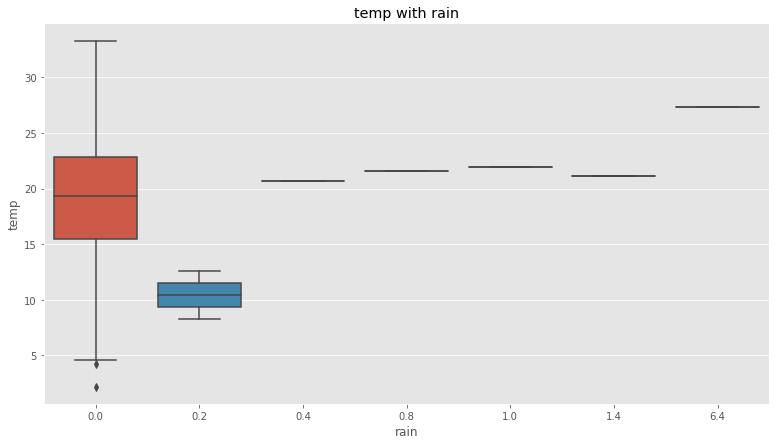
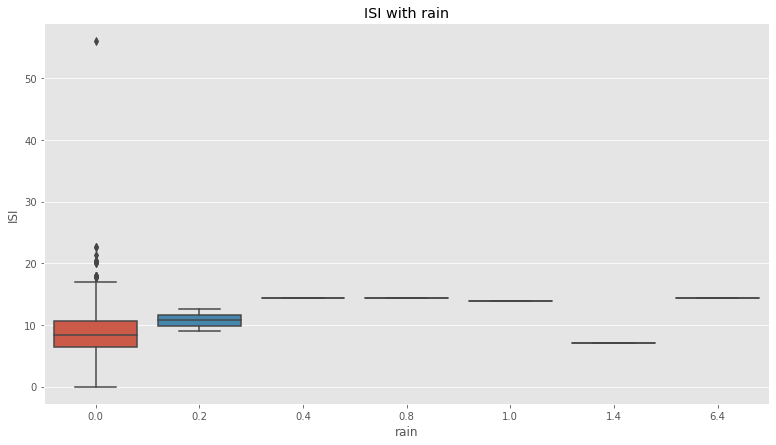
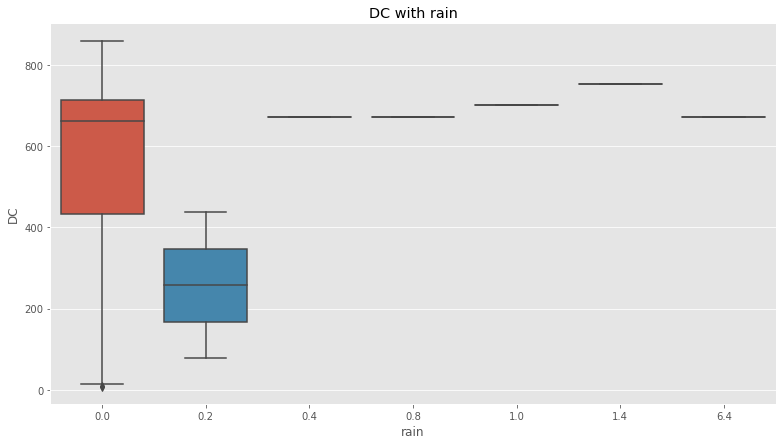
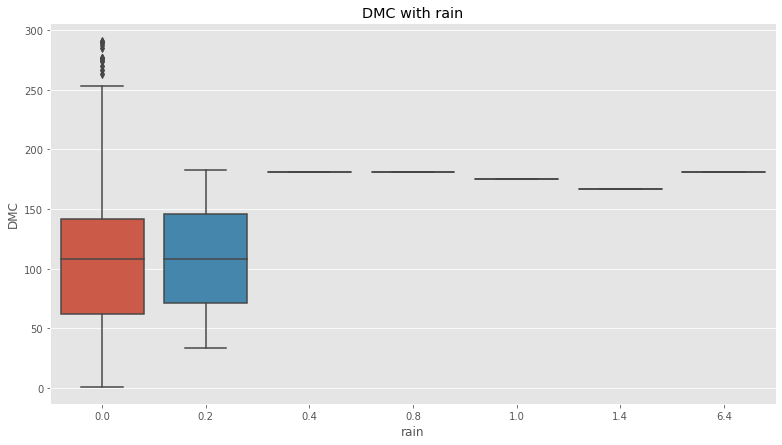
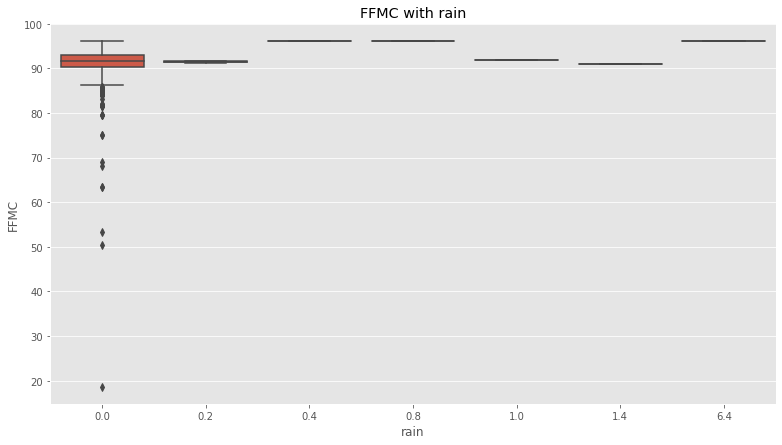
for i in col:

plt.figure(figsize=(13, 7))

plt.title(str(i) + " with " + str('rain'))

sb.boxplot(x=dataset.rain, y=train[i])

plt.show()

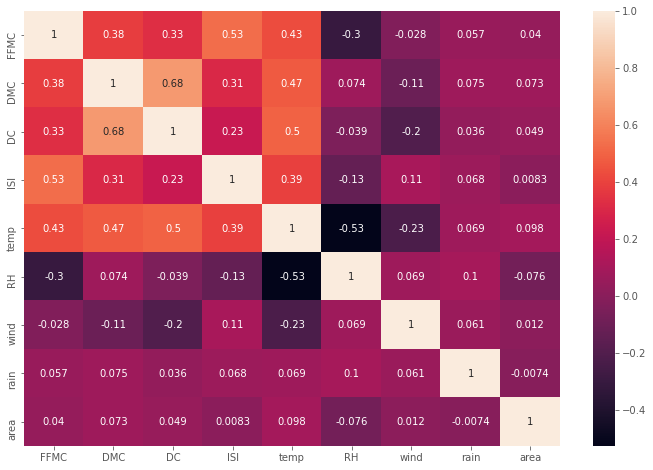


plt.figure(figsize=(12, 8))

corr = train.corr()

sb.heatmap(corr, annot=True)

plt.show()



# encoding categorical data

from sklearn.preprocessing import LabelEncoder, OneHotEncoder

labelencoder\_X\_1 = LabelEncoder()

X[:, 2] = labelencoder\_X\_1.fit\_transform(X[:, 2]) #For month

labelencoder\_X\_2 = LabelEncoder()

X[:, 3] = labelencoder\_X\_2.fit\_transform(X[:, 3]) #For weekday

#splitting data

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.2)

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

X\_train = sc.fit\_transform(X\_train)

X\_test = sc.transform(X\_test)

MAE

The Mean absolute error represents the average of the absolute difference between the actual and predicted values in the dataset. It measures the average of the residuals in the dataset.

MSE

Mean Squared Error represents the average of the squared difference between the original and predicted values in the data set.

It measures the variance of the residuals.

R square

The coefficient of determination or R-squared represents the proportion of the variance in the dependent variable which is explained by the linear regression model. It is a scale-free score i.e. irrespective of the values being small or large,

the value of R square will be less than one.

# Linear Regression

Linear Regression is a direct way to deal with displaying the connection between a scalar reaction (or dependent variable) and at least one illustrative factors (or independent factors).

from sklearn.linear\_model import LinearRegression

model = LinearRegression()

model.fit(X\_train, y\_train)

y\_pred=model.predict(X\_test)

from sklearn.metrics import mean\_squared\_error as mse

from sklearn.metrics import mean\_absolute\_error as mae

from sklearn.metrics import r2\_score

print('MSE =', mse(y\_pred, y\_test))

print('MAE =', mae(y\_pred, y\_test))

print('R2 Score =', r2\_score(y\_pred, y\_test))

# Decision Tree Regression

The decision trees are utilized to fit a sine curve with expansion uproarious perception. Therefore, it learns nearby straight relapses approximating the sine bend.We can see that if the greatest profundity of the tree (constrained by the max\_depth parameter) is set excessively high, the choice trees adapt too fine subtleties of the preparation information and gain from the clamor, for example, they overfit.

from sklearn.tree import DecisionTreeRegressor as dtr

reg = dtr(random\_state = 42)

reg.fit(X\_train, y\_train)

y\_pred = reg.predict(X\_test)

print('MSE =', mse(y\_pred, y\_test))

print('MAE =', mae(y\_pred, y\_test))

print('R2 Score =', r2\_score(y\_pred, y\_test))

# Random forest

Random forest is a Supervised Learning algorithm that uses ensemble learning methods for regression.

from sklearn.ensemble import RandomForestRegressor

regr = RandomForestRegressor(max\_depth=2, random\_state=0, n\_estimators=100)

regr.fit(X\_train, y\_train)

y\_pred = regr.predict(X\_test)

print('MSE =', mse(y\_pred, y\_test))

print('MAE =', mae(y\_pred, y\_test))

print('R2 Score =', r2\_score(y\_pred, y\_test))

RESULT : From the above testing, we can see that the Decision Tree regression model the best model as it has the maximum R2 score in negative so we can use decision tree method to predict

Thank you !