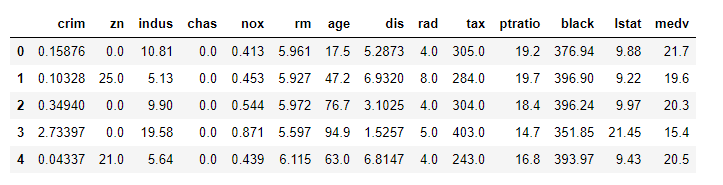
**Scope Of Project:**

After building Machine learning model we need to evaluate the model Results .In this Project we evaluate models by using different metrics that are there for both classification and regression algorithms.

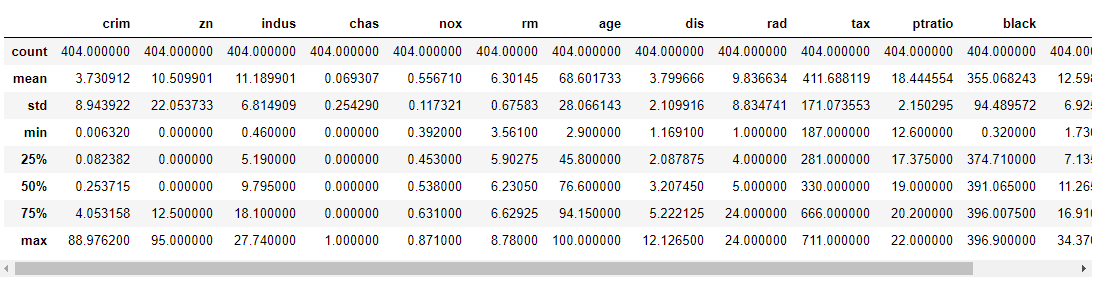
**Regression:**

1. import numpy as np
2. import pandas as pd
3. import matplotlib.pyplot as plt
4. from sklearn.linear\_model import LinearRegression
5. from sklearn.model\_selection import train\_test\_split
6. from sklearn.metrics import mean\_absolute\_error,r2\_score,mean\_squared\_error

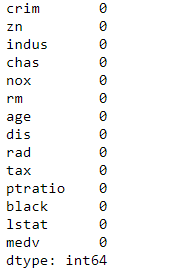
1. dataset=pd.read\_csv('diabetes.csv')
2. dataset.head()



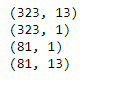
1. dataset.describe()



1. dataset.isnull().sum()



1. x\_train=dataset[['crim','zn','indus','chas','nox','rm','age','dis','rad','tax','ptratio','black','lstat']]
2. y\_train=dataset[['medv']]
3. x\_train=np.array(x\_train)
4. y\_train=np.array(y\_train)
5. X\_train,X\_test,Y\_train,Y\_test=train\_test\_split(x\_train,y\_train,test\_size=0.2)
6. print(X\_train.shape)
7. print(Y\_train.shape)
8. print(Y\_test.shape)
9. print(X\_test.shape)



1. model=LinearRegression()
2. model.fit(X\_train,Y\_train)



1. predictions=model.predict(X\_test)
2. error\_sq=mean\_absolute\_error(Y\_test,predictions)
3. error\_sq



1. error\_sqr=mean\_squared\_error(Y\_test,predictions)
2. error\_sqr



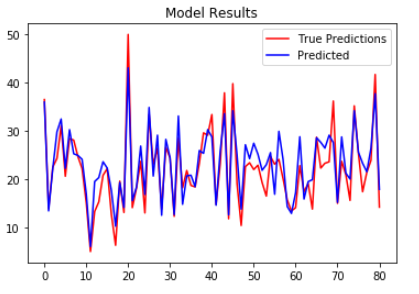
1. error\_nsqr=mean\_squared\_error(Y\_test,predictions,squared=False)
2. error\_nsqr



1. r2=r2\_score(Y\_test,predictions)
2. r2



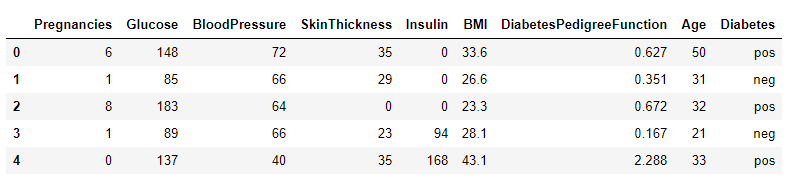
1. plt.plot(Y\_test,label='True Predictions',color='r')
2. plt.plot(predictions,label='Predicted',color='b')
3. plt.title('Model Results')
4. plt.legend()
5. plt.show()



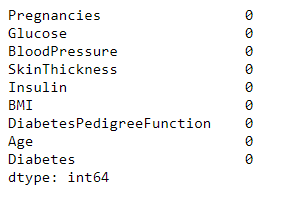
**Classification:**

1. import numpy as np
2. import pandas as pd
3. import matplotlib.pyplot as plt
4. from sklearn.preprocessing import LabelEncoder
5. from sklearn.preprocessing import StandardScaler
6. from sklearn.model\_selection import train\_test\_split
7. from sklearn.linear\_model import LogisticRegression
8. from sklearn.metrics import plot\_confusion\_matrix
9. from sklearn.metrics import accuracy\_score,confusion\_matrix,roc\_curve,auc

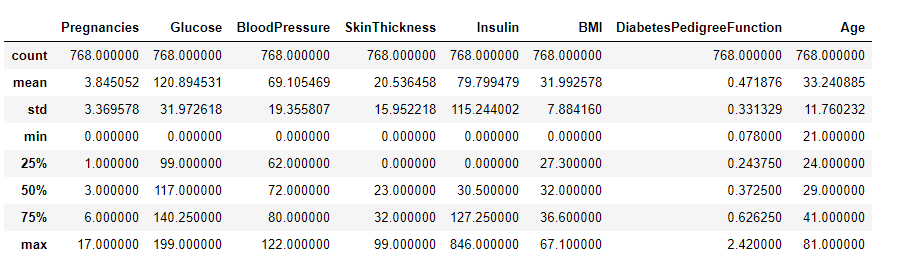
1. dataset=pd.read\_csv('diabetes.csv')
2. dataset.head()



1. dataset.isnull().sum()



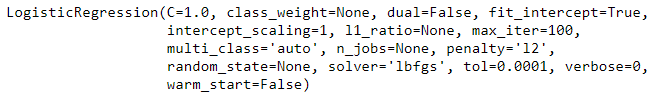
1. dataset.describe()



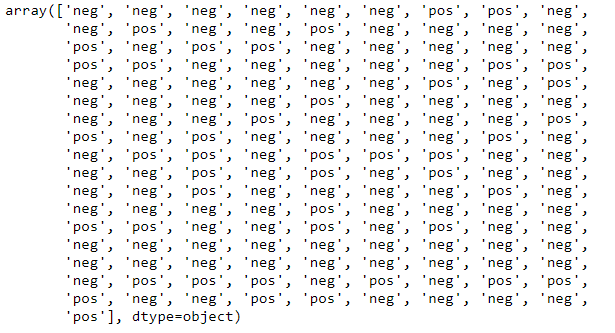
1. x=dataset.iloc[:,0:-1].values
2. y=dataset.iloc[:,-1].values
3. le=LabelEncoder()
4. y=le.fit\_transform(y)
5. sc=StandardScaler()
6. x=sc.fit\_transform(x)
7. x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2)
8. print(x\_train.shape)
9. print(y\_train.shape)
10. print(x\_test.shape)
11. print(y\_test.shape)



1. model1=LogisticRegression()
2. model1.fit(x\_train,y\_train)



1. pred1=model1.predict(x\_test)
2. pr1=le.inverse\_transform(pred1)
3. pr1



1. accuracy\_score(y\_test,pred1)



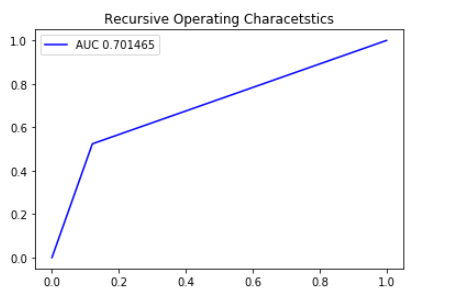
1. confusion\_matrix(y\_test,pred1)



1. fpr,tpr,threshold=roc\_curve(y\_test,pred1)
2. roc\_auc=auc(fpr,tpr)
3. roc\_auc



1. plt.title('Recursive Operating Characetstics')
2. plt.plot(fpr,tpr,'b',label='AUC %f'%roc\_auc)
3. plt.legend()
4. plt.show()



1. plot\_confusion\_matrix(model1,x\_test,y\_test)

