import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

plt.rcParams["figure.figsize"] = (10.0,10.0)

%matplotlib inline

from sklearn.linear\_model import LinearRegression, LogisticRegression

from sklearn.metrics import confusion\_matrix, classification\_report, accuracy\_score, mean\_squared\_error,r2\_score

from sklearn.tree import DecisionTreeClassifier

from sklearn.cross\_validation import train\_test\_split

from sklearn.datasets import load\_iris

data1=pd.read\_csv(r"C:\Users\kumars262\Desktop\DS\DataSets\headbrain.csv")

X1 = data1.iloc[:,2].values.reshape(-1,1)

Y1 = data1.iloc[:,3].values.reshape(-1,1)

data2=pd.read\_csv(r"C:\Users\kumars262\Downloads\train.csv")

data2[data2["Sex"].str.match("female")].count()

#sns.countplot(x="Survived", data=data2,hue="Sex")

#sns.boxplot(x="Pclass", y="Age", hue="Sex", data=data2)

#data2["Age"].plot.hist()

data2.isnull().any()

data2.isnull().sum()

#sns.heatmap(data=data2.isnull())

mean=data2["Age"].mean()

data2["Age"] = data2["Age"].fillna(mean)

data2.isnull().any()

data2.isnull().sum()

data2.drop("Cabin", inplace=True, axis=1)

data2.head()

data2.dropna(inplace=True)

data2.head()

data2.isnull().any()

Sex=pd.get\_dummies(data2["Sex"], drop\_first=True)

Pclass=pd.get\_dummies(data2["Pclass"], drop\_first=True)

Embarked = pd.get\_dummies(data2["Embarked"], drop\_first=True)

data2 = pd.concat([data2,Sex,Pclass,Embarked], axis=1)

data2.head()

data2.drop(["PassengerId", "Pclass","Name","Sex","Ticket","Fare","Embarked"], inplace=True, axis=1 )

data2.head()

X2=data2.drop("Survived", axis=1)

X2

Y2=data2["Survived"]

Y2

data3=load\_iris()

data3.data

data3.feature\_names

data3.target

data3.target\_names

df=pd.DataFrame(data3.data, columns=data3.feature\_names)

df.head

df["Type of Flower"] = data3.target

df.head()

df["Species"]= pd.Categorical.from\_codes(data3.target, data3.target\_names)

df.head()

df.drop("Type of Flower",inplace=True, axis=1)

df.head()

X3=df.drop("Species", axis=1)

Y3=df["Species"]

X1\_train,X1\_test,Y1\_train,Y1\_test = train\_test\_split(X1,Y1,test\_size=0.30, random\_state=1)

X2\_train,X2\_test,Y2\_train,Y2\_test = train\_test\_split(X2,Y2,test\_size=0.30, random\_state=1)

X3\_train,X3\_test,Y3\_train,Y3\_test = train\_test\_split(X3,Y3,test\_size=0.30, random\_state=1)

model1 = LinearRegression()

model1.fit(X1\_train, Y1\_train)

model2 = LogisticRegression()

model2.fit(X2\_train, Y2\_train)

model3 = DecisionTreeClassifier(criterion = 'entropy', min\_samples\_split = 50)

model3.fit(X3\_train, Y3\_train)

Y1\_pred = model1.predict(X1\_test)

Y2\_pred = model2.predict(X2\_test)

Y3\_pred = model3.predict(X3\_test)

print("Model1: Linear Regression Model:\n a. Intercept fo the Linear Line is: ", model1.intercept\_)

print(" b. Slope of the Linear Model is: " , model1.coef\_)

print(" c. Accuracy using Root Mean Square Method: ", np.sqrt(mean\_squared\_error(Y1\_test, Y1\_pred)))

print(" d. Accuracy using R2 method: ", r2\_score(Y1\_test, Y1\_pred)\*100)

plt.scatter(X1\_test, Y1\_test, label='True Values', color=’gray’)

plt.plot(X1\_test, Y1\_pred, 'bo-', label='Predicted Values', color='red', linewidth=2)

plt.title('Linear Regression Model')

plt.xlabel('Head Size(cm^3)')

plt.ylabel('Brain Weight(grams)')

plt.legend()

plt.show()

df1=pd.DataFrame({'Actual Y1 values': Y1\_test.flatten(), 'Predicted Y1 values' :Y1\_pred.flatten() })

print(df1.head())

print("\n\nModel2: Logistic Regression Model:\n a. Accuracy using Confusion Matrix: \n ", confusion\_matrix(Y2\_test, Y2\_pred))

print(" b. Classification Report: \n ", classification\_report(Y2\_test, Y2\_pred))

df2=pd.DataFrame({'Actual Y2 values': Y2\_test, 'Predicted Y2 values' :Y2\_pred })

print(df2.head())

print("\n\nModel3: DecisionTreeClassifier:\n a. Accuracy Score: ", accuracy\_score(Y3\_test, Y3\_pred)\*100)

df3=pd.DataFrame({'Actual Y3 values': Y3\_test, 'Predicted Y3 values' :Y3\_pred })

print("\n",df3.head())

# Output:

Model1: Linear Regression Model:

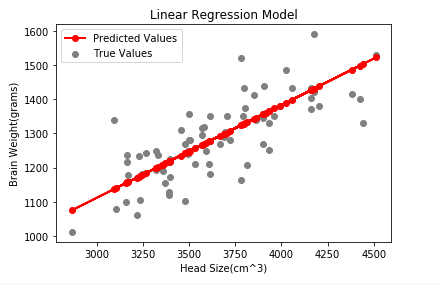
a. Intercept fo the Linear Line is: [297.7691189]

b. Slope of the Linear Model is: [[0.27136304]]

c. Accuracy using Root Mean Square Method: 74.42824457153844

d. Accuracy using R2 method: 58.53027944222984

|  |
| --- |
| Actual Y1 values Predicted Y1 values |
| 0 1235 1173.729010 |
| 1 1270 1292.586021 |
| 2 1280 1249.167935 |
| 3 1165 1323.250044 |
| 4 1320 1269.248799 |



Model2: Logistic Regression Model:

a. Accuracy using Confusion Matrix:

[[145 21]

[ 25 76]]

b. Classification Report:

precision recall f1-score support

0 0.85 0.87 0.86 166

1 0.78 0.75 0.77 101

avg / total 0.83 0.83 0.83 267

|  |
| --- |
| Actual Y2 values Predicted Y2 values |
| 387 1 1 |
| 259 1 1 |
| 771 0 0 |
| 208 1 1 |
| 683 0 0 |

Model3: DecisionTreeClassifier:

a. Accuracy Score: 95.55555555555556

|  |
| --- |
| Actual Y3 values Predicted Y3 values |
| 14 setosa setosa |
| 98 versicolor versicolor |
| 75 versicolor versicolor |
| 16 setosa setosa |
| 131 virginica virginica |

# Visualization :

|  |  |
| --- | --- |
| plt.figure(figsize=(5,5))  plt.tight\_layout()  sns.distplot(data2["Age"]) |  |

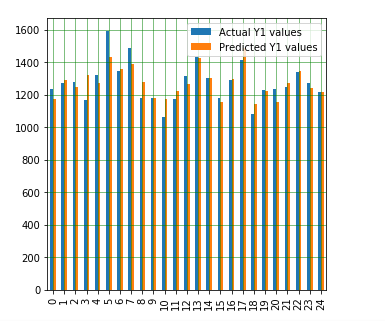
df = df1.head(25)

df.plot(kind='bar' , figsize=(5,5))

plt.grid(which='major', linewidth =0.5, linestyle='-' , color='green')

plt.grid(which='minor', linewidth =0.5, linestyle=':' , color='green')

plt.show()



From sklearn.metrics import mean\_absolute\_error

print("Mean absolute Error for linear model: ", mean\_absolute\_error(Y1\_test, Y1\_pred))

outpu: Mean absolute Error for linear model: 56.935401199136244

(MAE = mean of the absolute value of the errors) = MAE = 1/n\*Sum)