NAME: HAMMAD ABID

ID: 9134

ASSIGNMENT-2

Q1

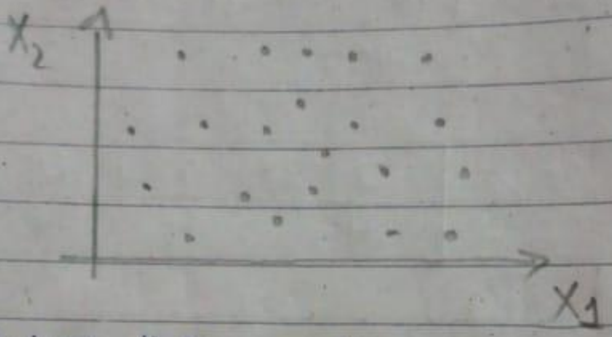
PART-A and B

NAME : HAMMAD ABID
ID : 9134

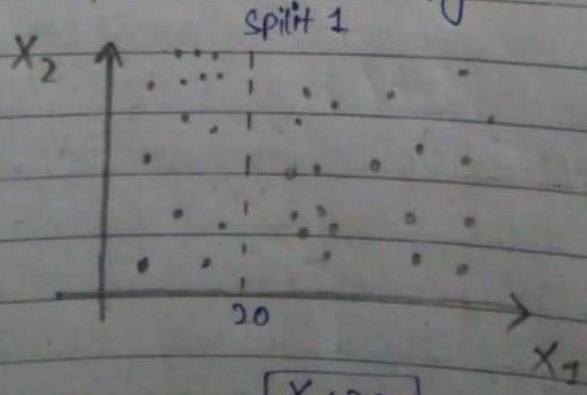
Assignment # 02

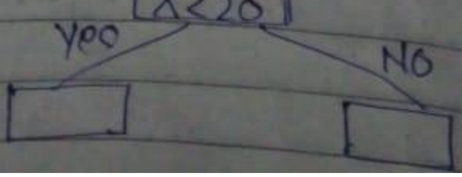
Q01

(a) Decision tree regression model is NON Linear and a Non Continous model



· To Solve this problem we use decision tree Algorithm. We have to spilit this data in to segments each spilit is Called leaf this spilit based on Principle OF Information entropy.

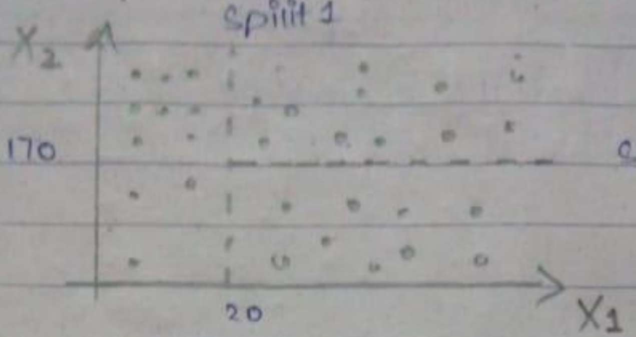


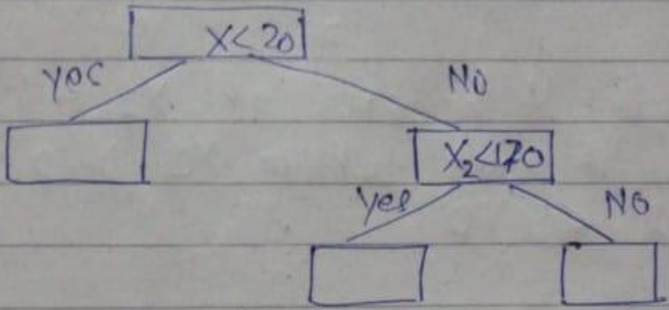


* Spilit
then g

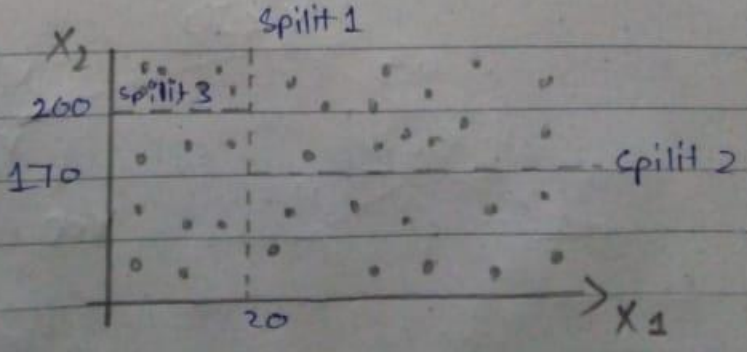
* Just
If x
else

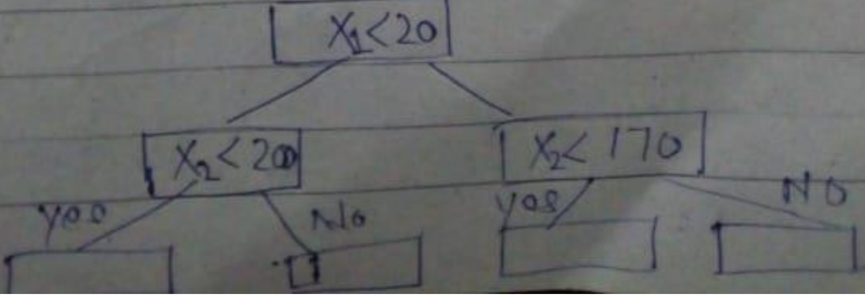
* Spilit $X_1 < 20$ into 2 segments if $X_1 < 20$ then go to yes else No.



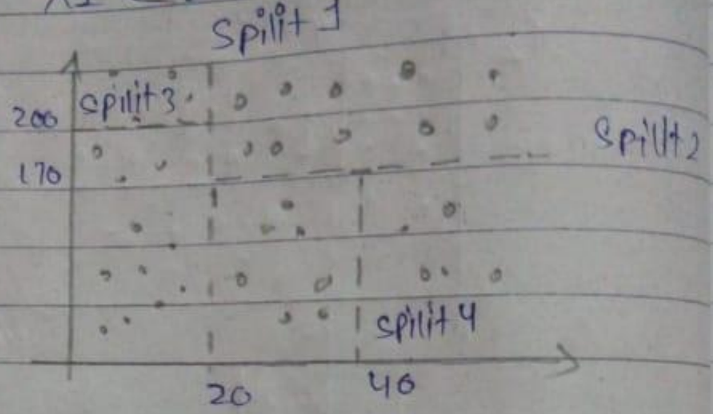


* Just like one Spilit 2 will Spilit $X_2 = 170$ if $X1 > 20$ and then $X_2 < 170$ go to yes else No.





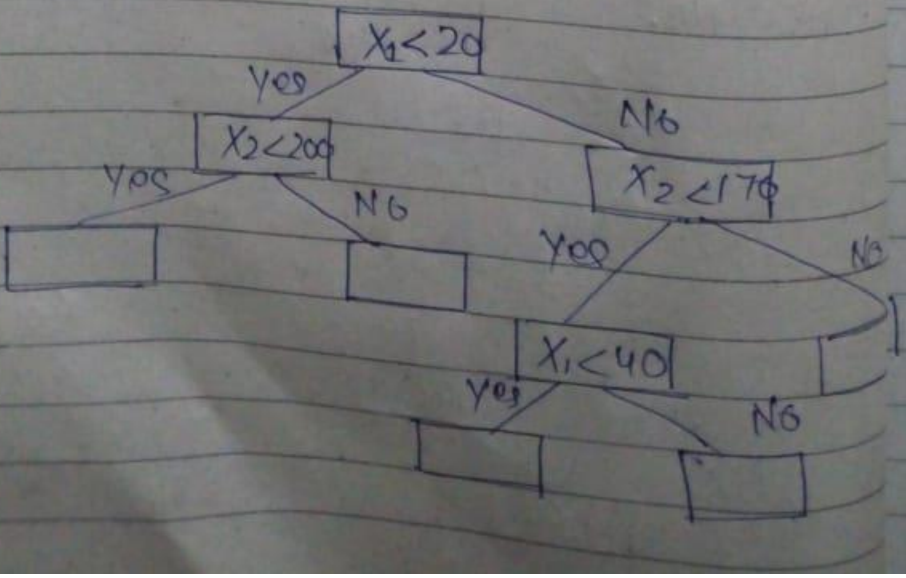
* Spilit 3 happens at $X_2 = 200$ but if $X1 < 20$.



* Spilit 4 happens at $X1 = 40$ but but if point whebe $X1 > 20$ and $X2 < 170$

(b)

So our final diagbam of Decision tbee is :-



• for ne
ite v
Simple
our
Avg

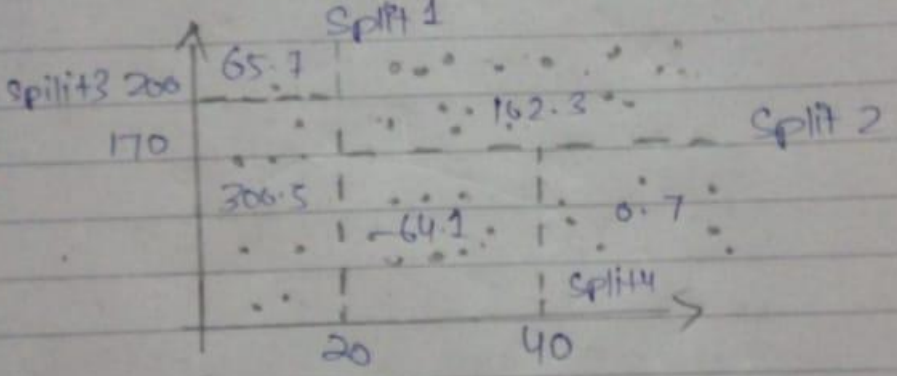
spilit3

• So IF
X1
in th
So
of

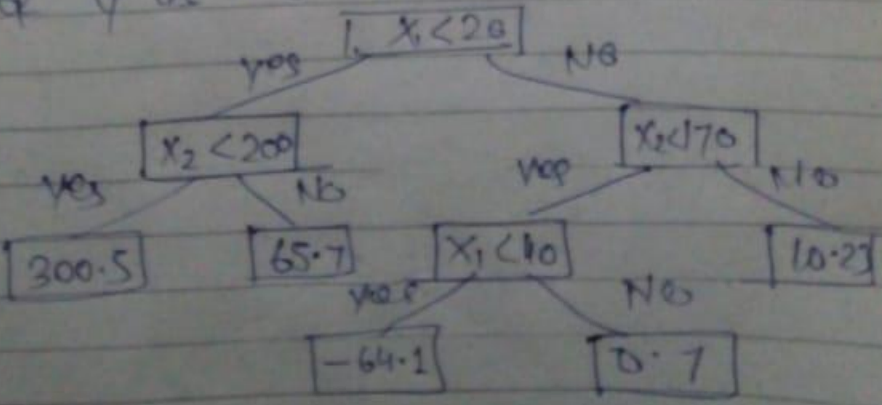
ye
300

- for new point ? how we determine it's value ?
Simple we can take the avg of our terminal leaves

$$Avg = \frac{\text{Total no of points values}}{\text{No of Total Point's}}$$



- So IF New data point Come whose X1 = 30 and X2 = 50 it will. Falls in the leaf whose Avg is -64.1 so Decision tree will Predict the value of y as -64.1



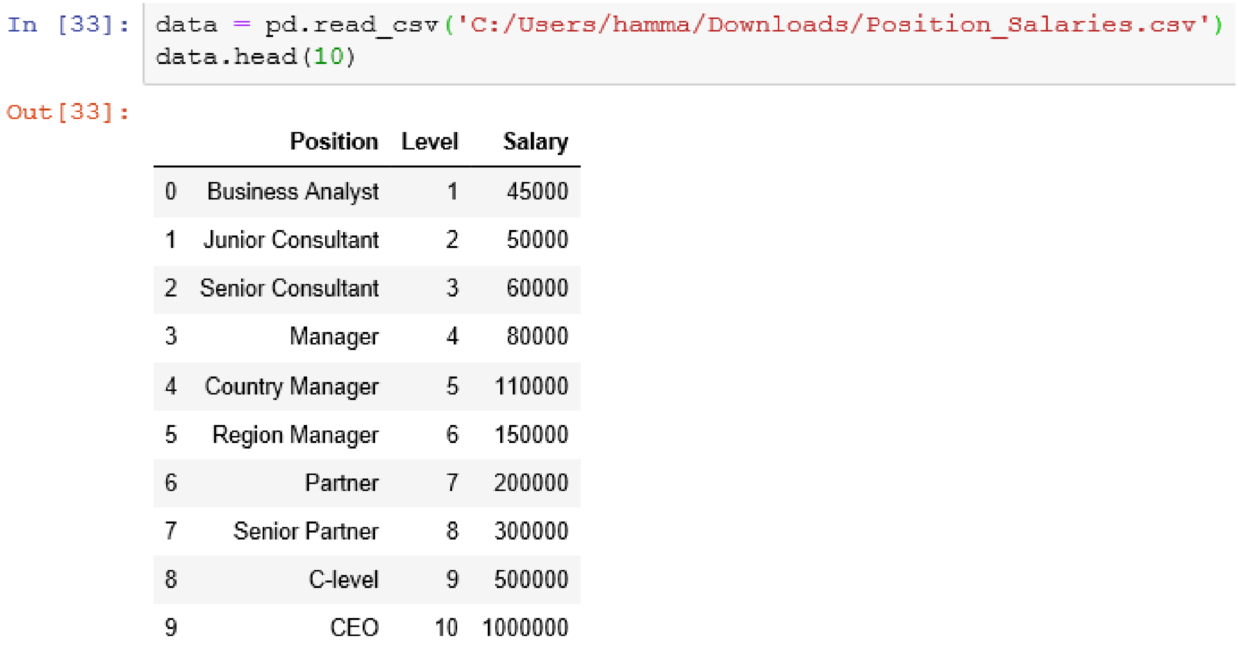
PART-C

Dataset

It has 3 columns — “Position”, “Level” and “Salary” and describes the approximate salary range for an employee based on what level he falls under.

Dataset file uploaded with this assignment on class room.

OUTPUT OF DATASET:



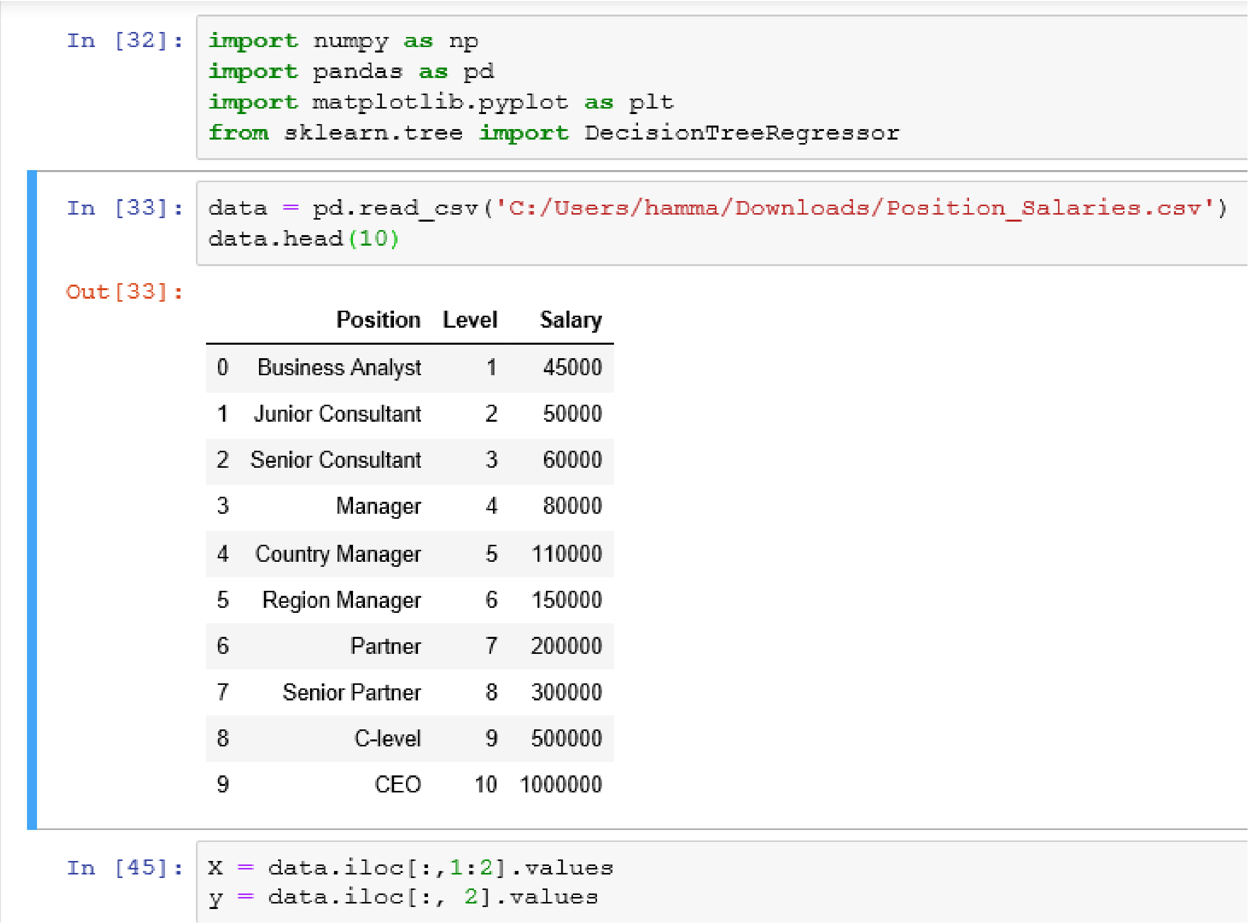
In [33]:

```
data = pd.read_csv('C:/Users/hamma/Downloads/Position_Salaries.csv')
data.head(10)
```

Out[33]:

	Position	Level	Salary
0	Business Analyst	1	45000
1	Junior Consultant	2	50000
2	Senior Consultant	3	60000
3	Manager	4	80000
4	Country Manager	5	110000
5	Region Manager	6	150000
6	Partner	7	200000
7	Senior Partner	8	300000
8	C-level	9	500000
9	CEO	10	1000000

IMPLEMENTATION OF DECISCION TREE REGGRESSION:



In [32]:

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeRegressor
```

In [33]:

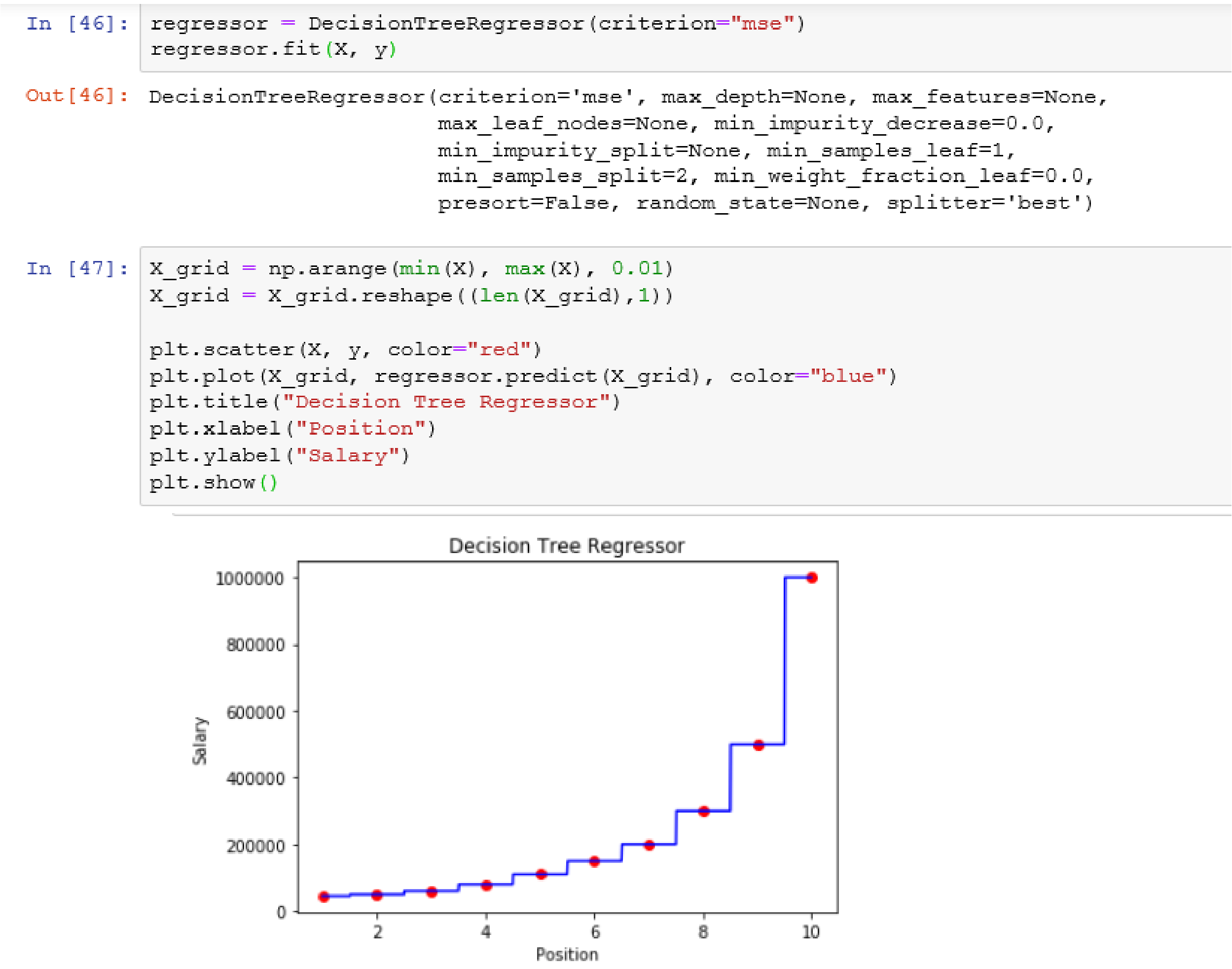
```python
data = pd.read_csv('C:/Users/hamma/Downloads/Position_Salaries.csv')
data.head(10)
```

Out[33]:

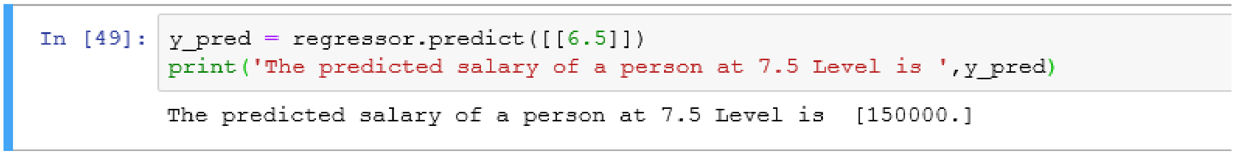
	Position	Level	Salary
0	Business Analyst	1	45000
1	Junior Consultant	2	50000
2	Senior Consultant	3	60000
3	Manager	4	80000
4	Country Manager	5	110000
5	Region Manager	6	150000
6	Partner	7	200000
7	Senior Partner	8	300000
8	C-level	9	500000
9	CEO	10	1000000

In [45]:

```python
X = data.iloc[:,1:2].values
y = data.iloc[:, 2].values
```



```
In [46]: regressor = DecisionTreeRegressor(criterion="mse")
         regressor.fit(X, y)

Out[46]: DecisionTreeRegressor(criterion='mse', max_depth=None, max_features=None,
                               max_leaf_nodes=None, min_impurity_decrease=0.0,
                               min_impurity_split=None, min_samples_leaf=1,
                               min_samples_split=2, min_weight_fraction_leaf=0.0,
                               presort=False, random_state=None, splitter='best')

In [47]: X_grid = np.arange(min(X), max(X), 0.01)
         X_grid = X_grid.reshape((len(X_grid),1))

         plt.scatter(X, y, color="red")
         plt.plot(X_grid, regressor.predict(X_grid), color="blue")
         plt.title("Decision Tree Regressor")
         plt.xlabel("Position")
         plt.ylabel("Salary")
         plt.show()
```



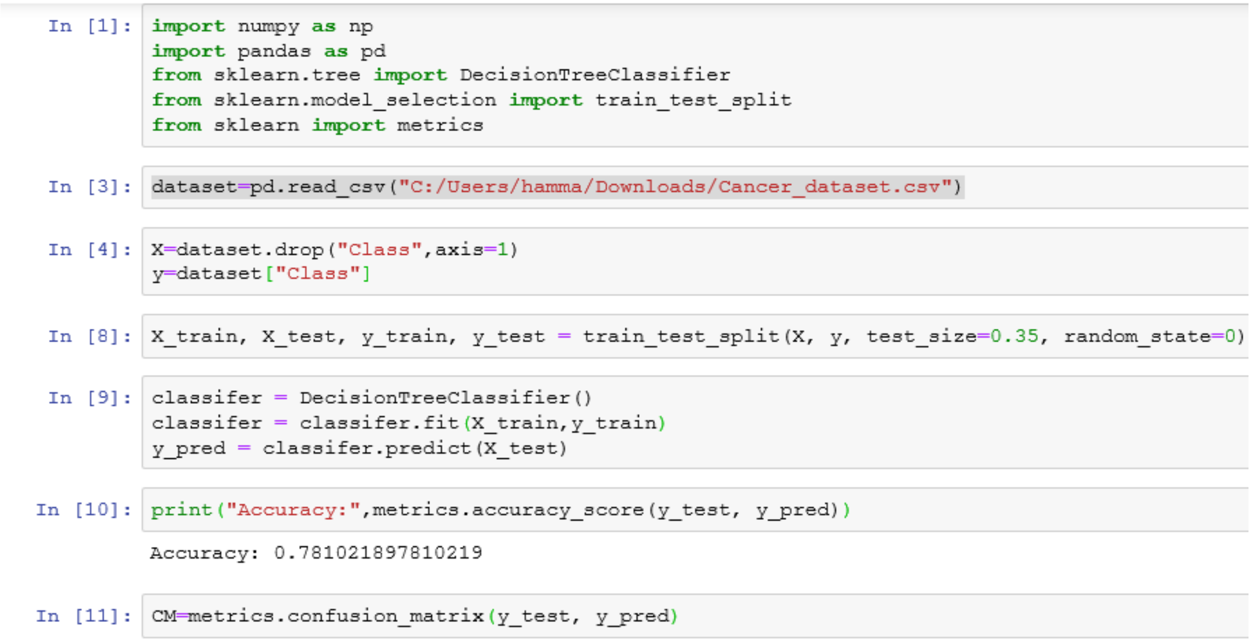
```
In [49]: y_pred = regressor.predict([[6.5]])
         print('The predicted salary of a person at 7.5 Level is ',y_pred)

         The predicted salary of a person at 7.5 Level is  [150000.]
```

PART -D:

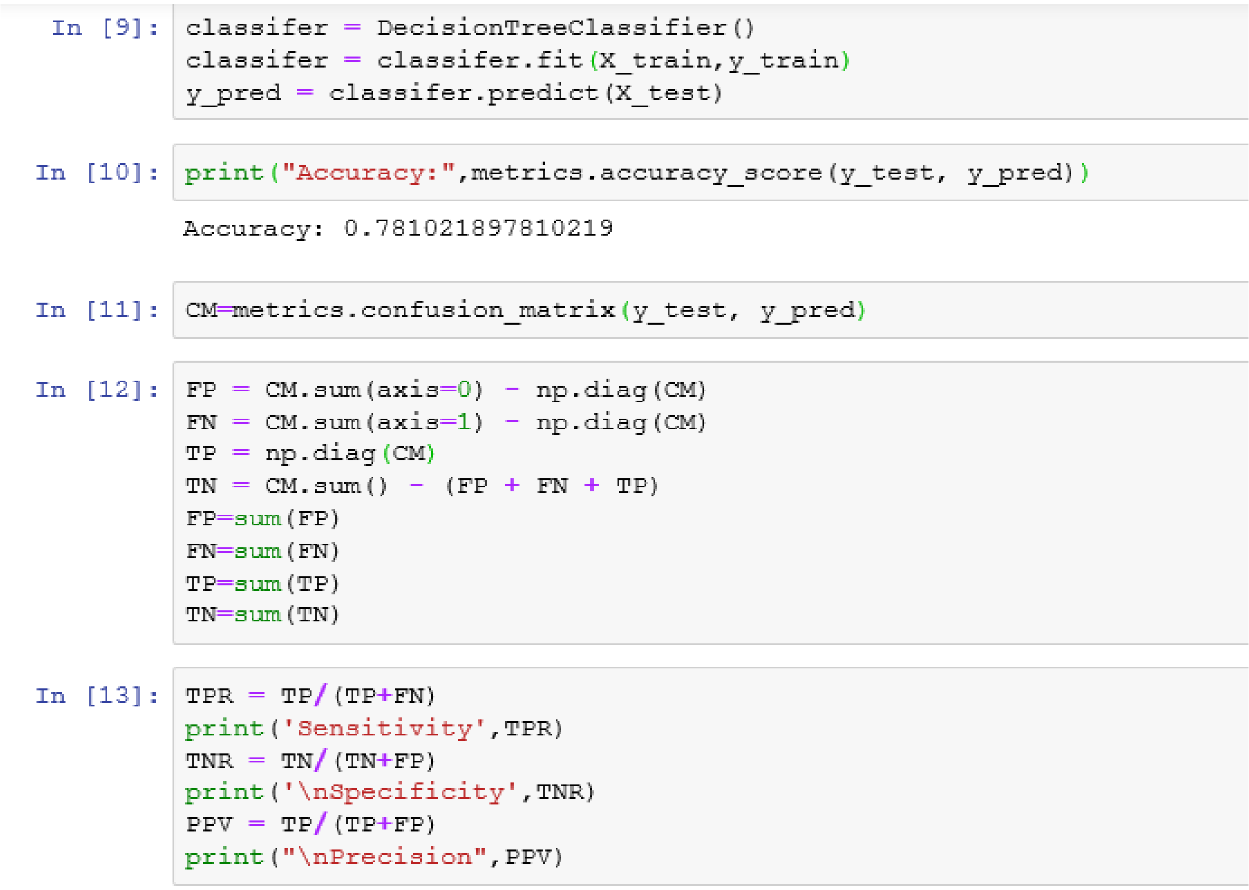
DECISION TREE	SENSITIVITY	SPECIFICITY	ACCURACY	PRECISION	AUC
IG	0.7810	0.9452	0.7810	0.7810	0.902
GINI INDEX	0.7153	0.9288	0.7153	0.7153	0.833

CODE:

INFORMATION GAIN METHOD:

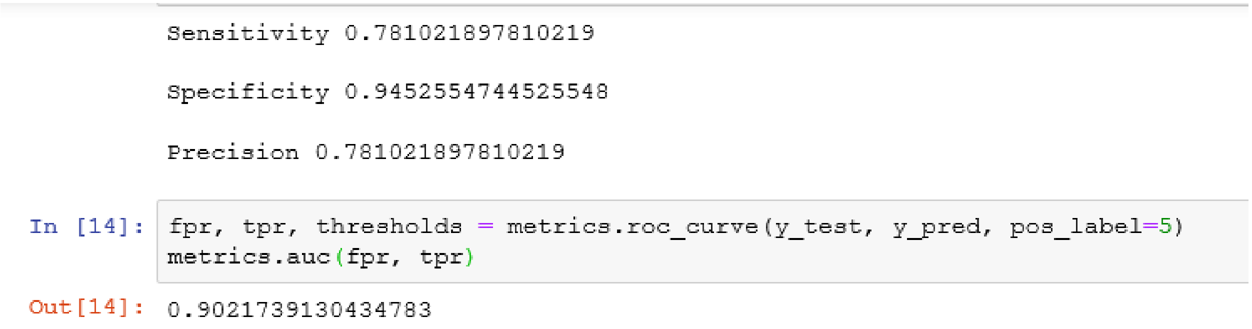


```
In [1]: import numpy as np
        import pandas as pd
        from sklearn.tree import DecisionTreeClassifier
        from sklearn.model_selection import train_test_split
        from sklearn import metrics

In [3]: dataset=pd.read_csv("C:/Users/hamma/Downloads/Cancer_dataset.csv")

In [4]: X=dataset.drop("Class",axis=1)
        y=dataset["Class"]

In [8]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.35, random_state=0)

In [9]: classifer = DecisionTreeClassifier()
        classifer = classifer.fit(X_train,y_train)
        y_pred = classifer.predict(X_test)

In [10]: print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

         Accuracy: 0.781021897810219

In [11]: CM=metrics.confusion_matrix(y_test, y_pred)
```



```
In [9]: classifer = DecisionTreeClassifier()
        classifer = classifer.fit(X_train,y_train)
        y_pred = classifer.predict(X_test)

In [10]: print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

         Accuracy: 0.781021897810219

In [11]: CM=metrics.confusion_matrix(y_test, y_pred)
```

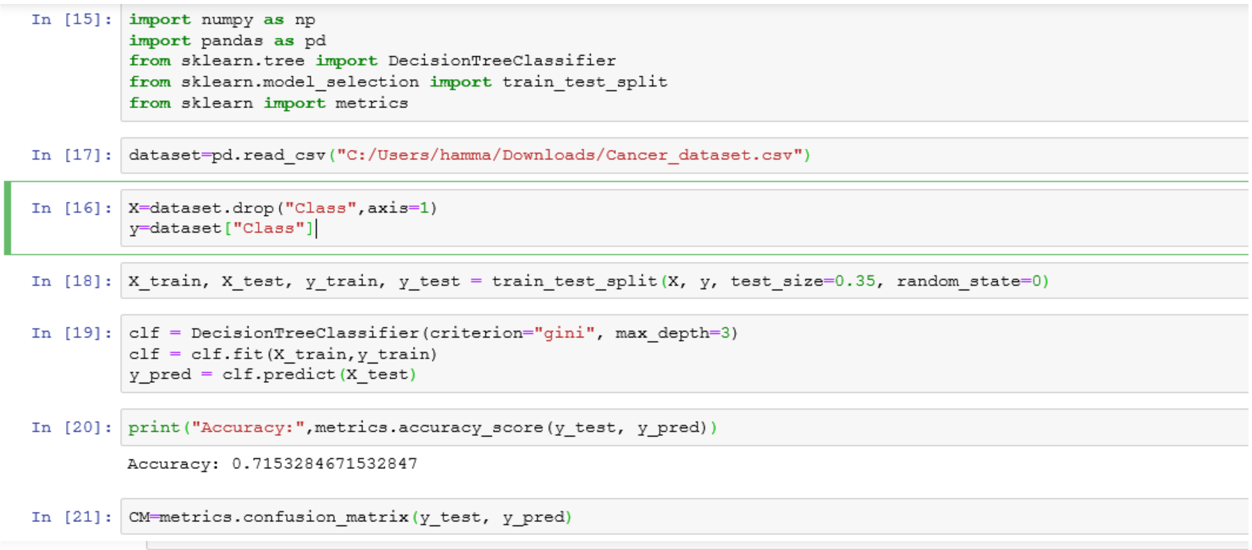
```
In [12]: FP = CM.sum(axis=0) - np.diag(CM)
         FN = CM.sum(axis=1) - np.diag(CM)
         TP = np.diag(CM)
         TN = CM.sum() - (FP + FN + TP)
         FP=sum(FP)
         FN=sum(FN)
         TP=sum(TP)
         TN=sum(TN)

In [13]: TPR = TP/(TP+FN)
         print('Sensitivity',TPR)
         TNR = TN/(TN+FP)
         print('\nSpecificity',TNR)
         PPV = TP/(TP+FP)
         print("\nPrecision",PPV)
```



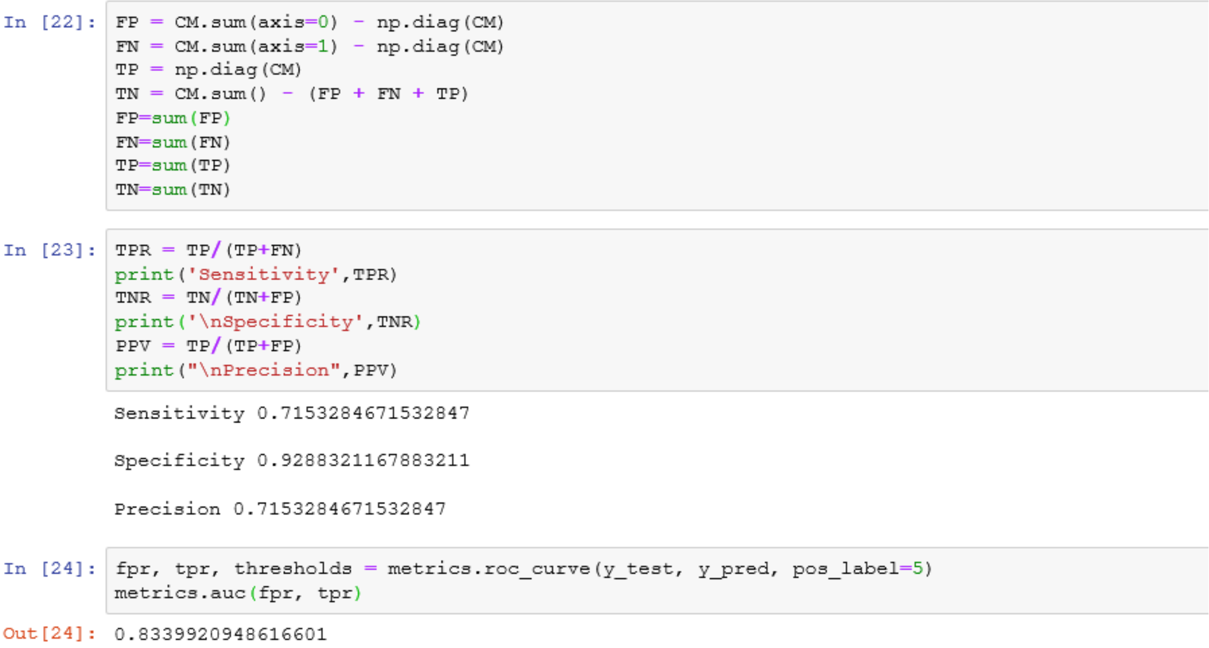
```
Sensitivity 0.781021897810219

Specificity 0.9452554744525548

Precision 0.781021897810219
```

```
In [14]: fpr, tpr, thresholds = metrics.roc_curve(y_test, y_pred, pos_label=5)
         metrics.auc(fpr, tpr)

Out[14]: 0.9021739130434783
```

GINI INDEX METHOD:



```
In [15]: import numpy as np
         import pandas as pd
         from sklearn.tree import DecisionTreeClassifier
         from sklearn.model_selection import train_test_split
         from sklearn import metrics

In [17]: dataset=pd.read_csv("C:/Users/hamma/Downloads/Cancer_dataset.csv")

In [16]: X=dataset.drop("Class",axis=1)
         y=dataset["Class"]

In [18]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.35, random_state=0)

In [19]: clf = DecisionTreeClassifier(criterion="gini", max_depth=3)
         clf = clf.fit(X_train,y_train)
         y_pred = clf.predict(X_test)

In [20]: print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

         Accuracy: 0.7153284671532847

In [21]: CM=metrics.confusion_matrix(y_test, y_pred)
```



```
In [22]: FP = CM.sum(axis=0) - np.diag(CM)
         FN = CM.sum(axis=1) - np.diag(CM)
         TP = np.diag(CM)
         TN = CM.sum() - (FP + FN + TP)
         FP=sum(FP)
         FN=sum(FN)
         TP=sum(TP)
         TN=sum(TN)

In [23]: TPR = TP/(TP+FN)
         print('Sensitivity',TPR)
         TNR = TN/(TN+FP)
         print('\nSpecificity',TNR)
         PPV = TP/(TP+FP)
         print("\nPrecision",PPV)

         Sensitivity 0.7153284671532847

         Specificity 0.9288321167883211

         Precision 0.7153284671532847

In [24]: fpr, tpr, thresholds = metrics.roc_curve(y_test, y_pred, pos_label=5)
         metrics.auc(fpr, tpr)

Out[24]: 0.8339920948616601
```