APSSDC

Andhra Pradesh State Skill Development Corporation

Day Objectives

Decision Tree

- Decision tree is the most powerful and popular tool for classification and prediction
- A decision tree is a very specific type of probability tree that enables you to make a decision about some kind of process.
- A Decision tree is a flowchart like tree structure
- Used in **data mining** for deriving a strategy to reach a particular goal, its also widely used in machine learning

Types of Algorithms

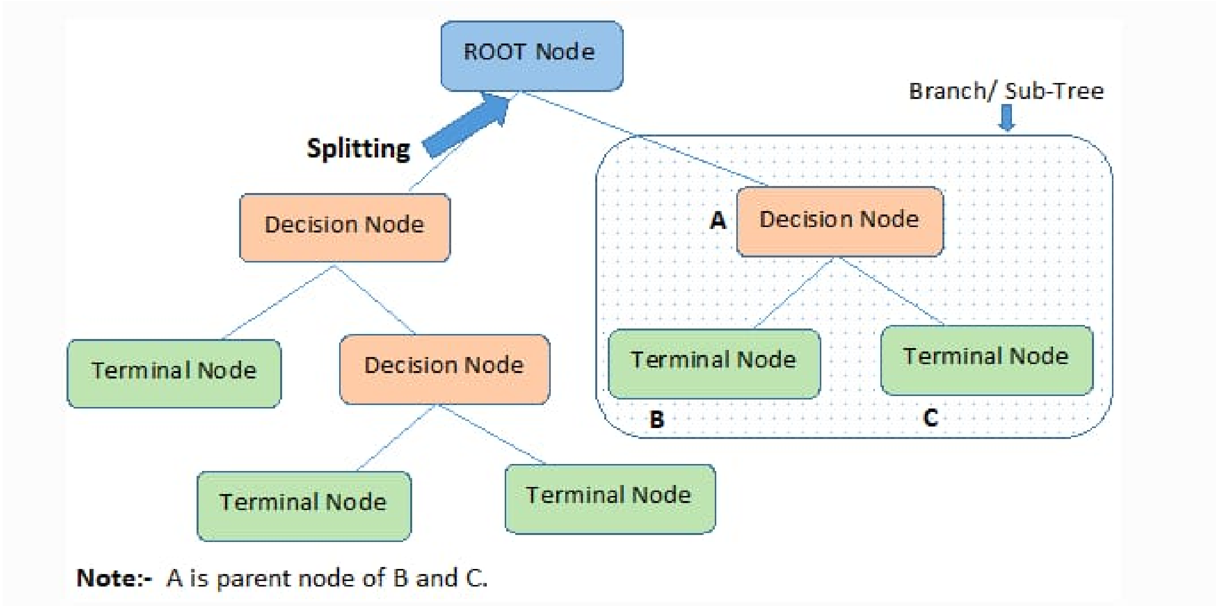
```
- CART  - classification and regression algorithm
    - gini index/ gini impurity
- ID3  - iterative dechomister 3
        - Information gain
        - log function / std deviation
```

Types of Decision Trees Types of decision trees are based on the type of target variable we have. It can be of two types:

- **Categorical Variable Decision Tree**: Decision Tree which has a categorical target variable then it called a Categorical variable decision tree.
- **Continuous Variable Decision Tree**: Decision Tree has a continuous target variable then it is called Continuous Variable Decision Tree.

Important Terminology related to Decision Trees

- **Root Node**: It represents the entire sample and this further gets divided into two or more homogeneous sets.
- **Splitting**: It is a process of dividing a node into two or more sub-nodes.
- **Decision Node**: When a sub-node splits into further sub-nodes, then it is called the decision node.
- **Leaf / Terminal Node**: Nodes do not split is called Leaf or Terminal node.
- **Pruning**: When we remove sub-nodes of a decision node, this process is called pruning. You can say the opposite process of splitting.
- **Branch / Sub-Tree**: A subsection of the entire tree is called branch or sub-tree.
- **Parent and Child Node**: A node, which is divided into sub-nodes is called a parent node of sub-nodes whereas sub-nodes are the child of a parent node.



Advantages:

```
- Simple to understand, interpret, visualize.
- Decision trees implicitly perform variable screening or feature select
ion.
- Can handle both numerical and categorical data. Can also handle multi-
output problems.
- Nonlinear relationships between parameters do not affect tree performa
nce.
```

DisAdvantage:

```
- OverFitting Problem
```

Decision Tree Classification

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [13]:
```
data = pd.read_csv("https://raw.githubusercontent.com/LavanyaPolamarasetty/D
data.head()
```

Out[13]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	A
0	6	148	72	35	0	33.6	0.627	
1	1	85	66	29	0	26.6	0.351	
2	8	183	64	0	0	23.3	0.672	
3	1	89	66	23	94	28.1	0.167	
4	0	137	40	35	168	43.1	2.288	

In [3]:
```
data.shape
```

Out[3]:
```
(768, 9)
```

In [4]:
```
data.columns
```

Out[4]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

In [6]:
```
data.info()
```

...

In [7]:
```
data.isnull().sum()
```

...

In [8]:
```
data.describe()
```

...

In [9]:
```
data["Outcome"].value_counts()
```

Out[9]:
```
0    500
1    268
Name: Outcome, dtype: int64
```

In [11]:
```
data["Age"].min()
```

Out[11]:
```
21
```

In [12]:
```
data["Age"].max()
```

Out[12]: 81

In [22]:
```
data.columns
```

Out[22]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

In [23]:
```
X = data[["Glucose","BMI","Age"]]
X.head()
```

Out[23]:

	Glucose	BMI	Age
0	148	33.6	50
1	85	26.6	31
2	183	23.3	32
3	89	28.1	21
4	137	43.1	33

In [19]:
```
import seaborn as sns
sns.pairplot(data)
```

...

In [21]:
```
data.corr()
```

...

In [24]:
```
y = data["Outcome"]
```

In [25]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.3, random
```

In [26]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [46]:
```
dcls = DecisionTreeClassifier( max_depth = 3)
dcls.fit(X_train,y_train)
```

Out[46]: DecisionTreeClassifier(max_depth=3)

In [47]:
```
y_pred = dcls.predict(X_test)
X_test.shape
```

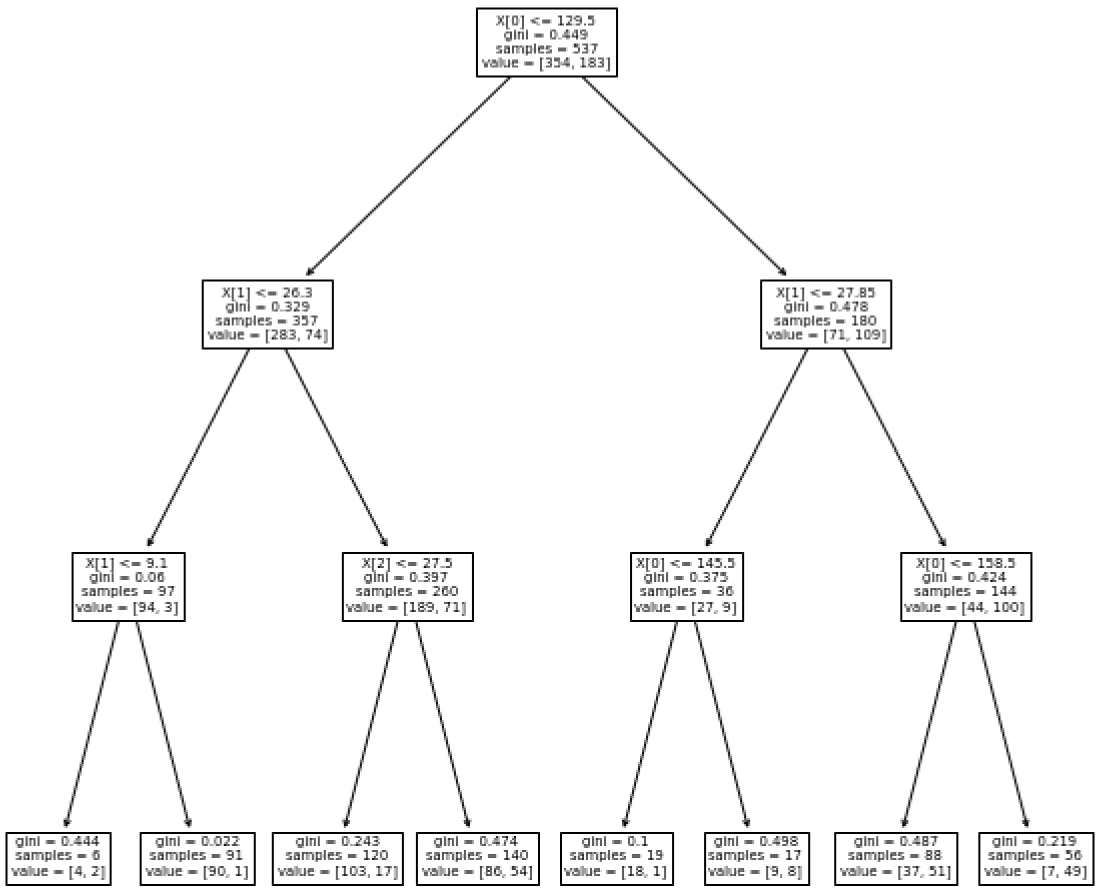
Out[47]: (231, 3)

In [48]:
```
from sklearn.metrics import confusion_matrix, accuracy_score
```

In [49]:
```
accuracy_score(y_test,y_pred) * 100
```

Out[49]: 75.75757575757575

In [53]:
```
from sklearn import tree
plt.figure(figsize = (10,10))
tree.plot_tree(dcls)
plt.show()
```



Calculation of Gini Index

- this is worked on Probability

$$GI = 1 - \sum_{i=1}^{n} (p)^2$$

$$GI = 1 - \left[(P_{(+)})^2 + (P_{(-)})^2 \right]$$

Past Trend	Open Interest	Trading Volume	Return
Positive	Low	High	Up
Negative	High	Low	Down
Positive	Low	High	Up
Positive	High	High	Up
Negative	Low	High	Down
Positive	Low	Low	Down
Negative	High	High	Down
Negative	Low	High	Down
Positive	Low	Low	Down
Positive	High	High	Up

Return

up probability - 4/10

down prob - 6/10

Past Trend

positive Prob - 6/10

Negative prob - 4/10

Past Trend prob with Return

prob(Positive , up) - 4/ 6

prob (Negative , up) - 2/ 6

prob (Positive , down) - 0

prob (Negative, down) - 4/4 = 1

In [57]:
```
gini = 1 - (((4/6)**2) + ((2/6)**2))
gini   = 0.45  # past Trend with rep return
```

Out[57]: 0.4444444444444444

In [58]:
```
1 - (0 + 1)
```

Out[58]: 0

Decision Tree Regressor

In [59]:
```
df = pd.read_csv("https://raw.githubusercontent.com/LavanyaPolamarasetty/Dat
df.head()
```

...

In [60]:
```
df.columns
```

Out[60]:
```
Index(['make', 'fuel-type', 'num-of-doors', 'body-style', 'engine-location',
       'length', 'width', 'height', 'num-of-cylinders', 'horsepower',
       'peak-rpm', 'city-mpg', 'highway-mpg', 'price'],
      dtype='object')
```

In [61]:
```
df.shape
```

Out[61]: (201, 14)

In [62]:
```
df.info()
```

...

In [63]:
```
df.isnull().sum()
```

...

In [64]:
```
df.corr()
```

Out[64]:

	length	width	height	city-mpg	highway-mpg	price
length	1.000000	0.857170	0.492063	-0.665192	-0.698142	0.690628
width	0.857170	1.000000	0.306002	-0.633531	-0.680635	0.751265
height	0.492063	0.306002	1.000000	-0.049800	-0.104812	0.135486
city-mpg	-0.665192	-0.633531	-0.049800	1.000000	0.972044	-0.686571
highway-mpg	-0.698142	-0.680635	-0.104812	0.972044	1.000000	-0.704692
price	0.690628	0.751265	0.135486	-0.686571	-0.704692	1.000000

In [65]:
```
X = df[["length","width","height"]]
```

In [66]:
```
y = df["price"]
```

In [67]:
```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.3)
```

In [68]:
```
from sklearn.tree import DecisionTreeRegressor
```

In [109]:
```
dreg = DecisionTreeRegressor(max_depth=3)
```

In [110]:
```
dreg.fit(X_train,y_train)
```

Out[110]:
```
DecisionTreeRegressor(max_depth=3)
```

In [111]:
```
y_pred = dreg.predict(X_test)
```

In [112]:
```
X_train.iloc[0]
```

Out[112]:
```
length    178.5
width      67.9
height     49.7
Name: 103, dtype: float64
```

In [113]:
```

y_train.iloc[0]

```

Out[113]:
```
18399
```

In [114]:
```
#18399-11177 = 7222

18399-9378
```

Out[114]: 9021

In [115]:
```
dreg.predict([[178,67.9,49.7]])
```

Out[115]: array([13846.39726027])

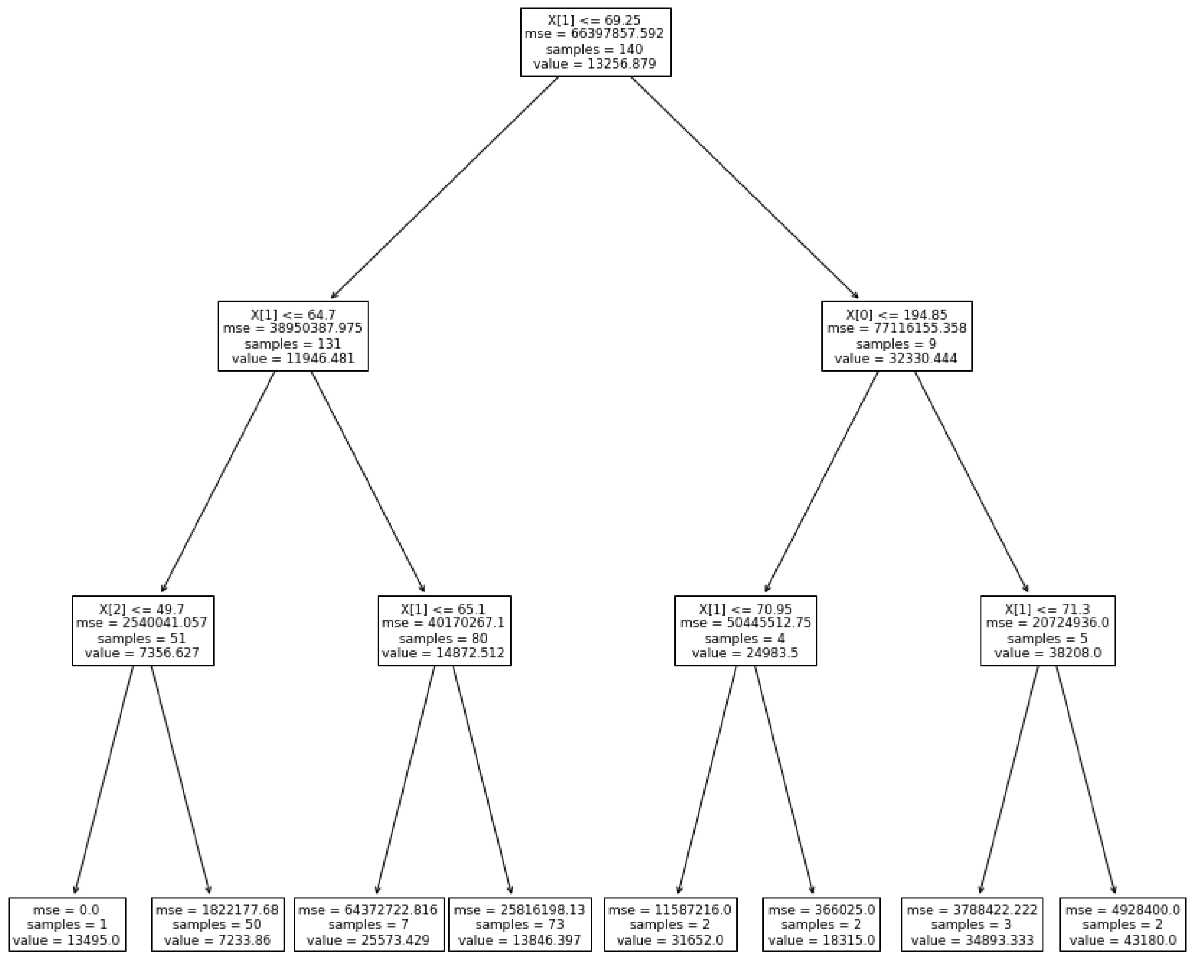
In [116]:
```
from sklearn.metrics import r2_score, mean_squared_error
r2_score(y_test,y_pred) * 100
```

Out[116]: 60.78815249740719

In [117]:
```
mean_squared_error(y_test,y_pred)**0.5
```

Out[117]: 4629.649178693528

In [120]:
```
plt.figure(figsize = (15,15))
tree.plot_tree(dreg)
plt.show()
```



In [ ]:

```
1
```