```
1
     import pandas as pd
    from sklearn.tree import DecisionTreeClassifier
    from sklearn.model selection import train test split
    from sklearn import metrics
    from sklearn import tree
    import matplotlib.pyplot as plt
    from sklearn.metrics import confusion_matrix
    from sklearn.metrics import accuracy_score
    from sklearn.tree import plot_tree
10
    from sklearn.metrics import classification_report
11
    import graphviz
    data=pd.read_csv('/content/Comp.csv')
    data.head()
```

	age	Income	Student	Credit_Rating	Buys_Computer
0	<=30	high	no	fair	no
1	<=30	high	no	excellent	no
2	3140	high	no	fair	yes
3	>40	medium	no	fair	yes
4	>40	low	yes	fair	yes

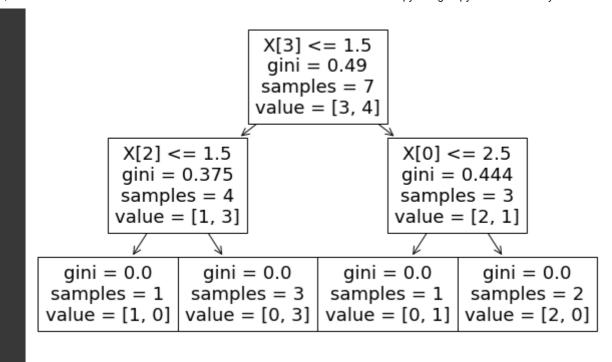
```
1 data=data.replace(['<=30','31...40','>40'],[1,2,3])
2 data.head()
3 data=data.replace(['high','medium','low'],[1,2,3])
4 data.head()
5 data=data.replace(['no','yes'],[1,2])
6 data.head()
7 data=data.replace(['fair','excellent'],[1,2])
8 data.head()
9 #X=data.drop(columns=['Outcome'])
10 #Y=data['Outcome']
11 #X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.9, random_state =0)
```

	age	Income	Student	Credit_Rating	Buys_Computer
0	1	1	1	1	1
1	1	1	1	2	1
2	2	1	1	1	2
3	3	2	1	1	2
4	3	3	2	1	2

```
1 X=data.drop(columns=['Buys_Computer'])
```

```
2 Y=data['Buys_Computer']
 3 X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.9, random_state =0)
 4 print("Entropy")
 5 model = DecisionTreeClassifier(criterion = "entropy")
 6 model.fit(X_train, y_train)
 7 y_pred = model.predict(X_test)
 8 print("Predicted values:")
9 print(y_pred)
10 print("Confusion Matrix: ",confusion_matrix(y_test, y_pred))
11 print("Accuracy : ",accuracy_score(y_test,y_pred)*100)
12 print("Report : ",classification_report(y_test, y_pred))
13
14 dtree = DecisionTreeClassifier()
15 dtree = dtree.fit(X_test, y_test)
16 features = ['age', 'Income', 'Student', 'Credit_Rating', 'Buys_Computer']
17 tree.plot_tree(dtree, feature_names=features)
```

```
Entropy
     Predicted values:
     [2 2 2 2 2 2 2 2 2 2 2 2 2 2]
     Confusion Matrix: [[0 5]
     [0 8]]
     Accuracy: 61.53846153846154
                              precision recall f1-score
     Report :
                                                                support
1 X=data.drop(columns=['Buys_Computer'])
2 Y=data['Buys_Computer']
3 X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.5, random_state =1)
4 print("Gini")
5 model=DecisionTreeClassifier(criterion = "gini")
6 model=model.fit(X_train, y_train)
7 y_pred = model.predict(X_test)
8 print("Predicted values:")
9 print(y_pred)
10 print("Confusion Matrix: ",confusion_matrix(y_test, y_pred))
11 print("Accuracy : ",accuracy_score(y_test,y_pred)*100)
12 print("Report : ",classification_report(y_test, y_pred))
     Gini
     Predicted values:
     [2 1 2 2 1 2 1]
     Confusion Matrix: [[2 0]
      [1 4]]
     Accuracy: 85.71428571428571
     Report :
                              precision
                                           recall f1-score
                                                                support
                1
                         0.67
                                   1.00
                                              0.80
                                                            2
                2
                         1.00
                                              0.89
                                   0.80
                                              0.86
         accuracy
                         0.83
                                   0.90
                                              0.84
        macro avg
                         0.90
                                   0.86
                                              0.86
     weighted avg
    2 | ' ),
1 plt.figure(figsize=(10,6))
2 plot_tree(model)
3 plt.show()
```



Colab paid products - Cancel contracts here

