In [59]:
```
import matplotlib.pyplot as plt
import pandas as pd
```

In [60]:
```
data=pd.read_csv("C:\\Users\\mm\\adult.csv")
```

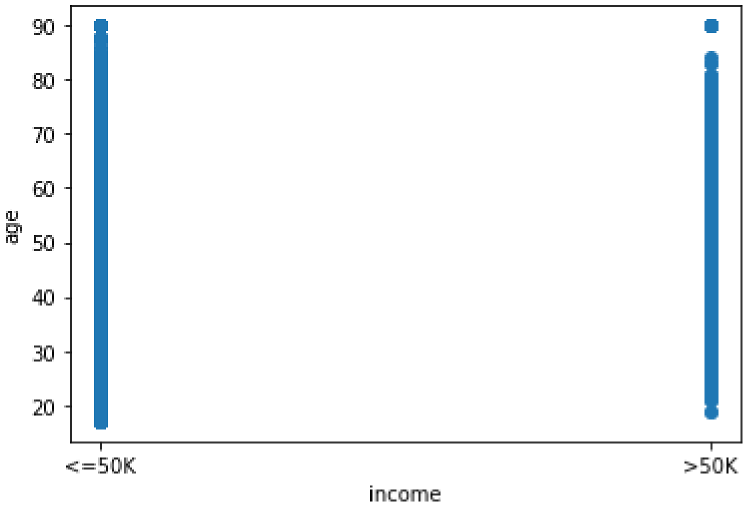
In [61]:
```
data.head()
```

Out[61]:

	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	rac
0	90	?	77053	HS-grad	9	Widowed	?	Not-in-family	Whi
1	82	Private	132870	HS-grad	9	Widowed	Exec-managerial	Not-in-family	Whi
2	66	?	186061	Some-college	10	Widowed	?	Unmarried	Blac
3	54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	Unmarried	Whi
4	41	Private	264663	Some-college	10	Separated	Prof-specialty	Own-child	Whi

In [62]:
```
plt.scatter(data['income'], data['age'])
plt.xlabel('income')
plt.ylabel('age')
```

Out[62]:
```
Text(0, 0.5, 'age')
```



In [80]:
```
data['age'].mean()
```

Out[80]:
```
38.58164675532078
```

In [84]:

```
list_mix=[]
for i in data['income']:

    if("<" in i ):
        list_mix.append(0)

    if(">" in i):
        list_mix.append(1)
```

In [116]:

```
# splitting the dataset into input and output datasets
X = data.iloc[:, [0,4,10,11,12]].values
y = list_mix
```

In [117]:

```
# splitting the dataaset into Training and Testing Data
from sklearn.model_selection import train_test_split

# random state is 0 and test size if 25%
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.25, random_st
```

In [118]:

```
# importing standard scalling method from sklearn
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()

# providing the inputs for the scalling purpose
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

In [119]:

```
# importing decision tree algorithm
from sklearn.tree import DecisionTreeClassifier

# entropy means information gain
classifer = DecisionTreeClassifier(criterion='entropy', random_state=0)

# providing the training dataset
classifer.fit(X_train,y_train)
```

Out[119]:

```
DecisionTreeClassifier(criterion='entropy', random_state=0)
```

In [120]:

```
y_pred = classifer.predict(X_test)
```

In [121]:

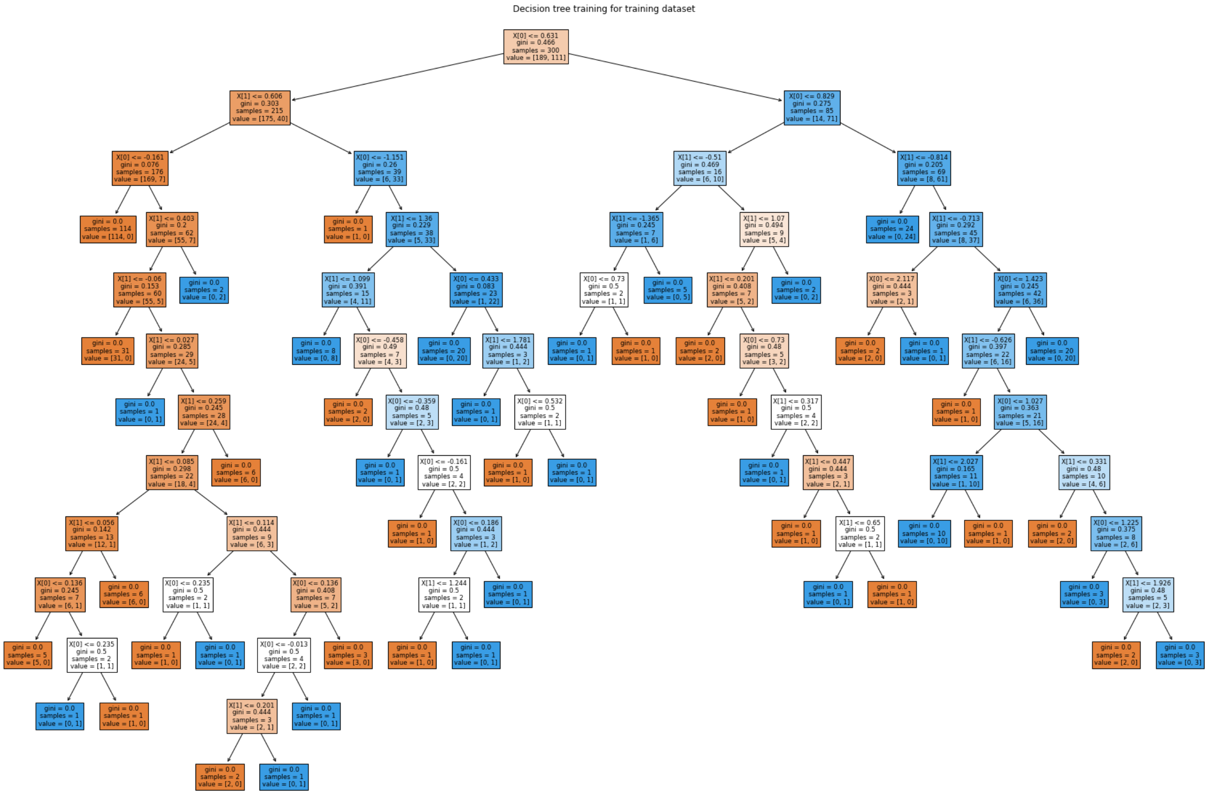
```
# importing the accuracy score
from sklearn.metrics import accuracy_score

# accuracy
accuracy_score(y_pred,y_test)
```

Out[121]:

```
0.8076403390246898
```

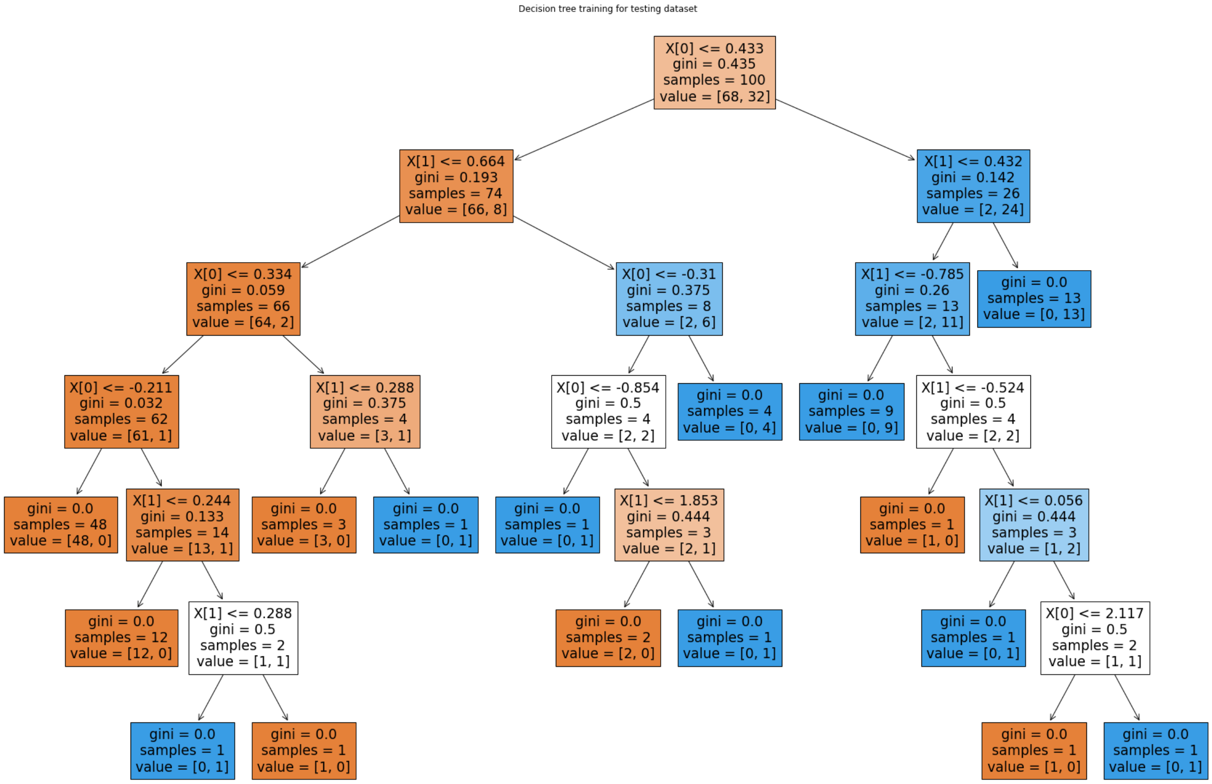
In [131]:

```
from sklearn.tree import DecisionTreeClassifier, plot_tree
clf = DecisionTreeClassifier()

# output size of decision tree
plt.figure(figsize=(30,20))

# providing the training dataset
clf = clf.fit(X_train, y_train)
plot_tree(clf, filled=True)
plt.title("Decision tree training for training dataset")
plt.show()
```


Decision tree training for training dataset
X[0] <= 0.631 gini = 0.466 samples = 300 value = [189, 111]
X[1] <= 0.606 gini = 0.303 samples = 215 value = [175, 40]
X[0] <= 0.829 gini = 0.275 samples = 85 value = [14, 71]
X[0] <= -0.161 gini = 0.076 samples = 176 value = [169, 7]
X[0] <= -1.151 gini = 0.26 samples = 39 value = [6, 33]
X[1] <= -0.51 gini = 0.469 samples = 16 value = [6, 10]
X[1] <= -0.814 gini = 0.205 samples = 69 value = [8, 61]
gini = 0.0 samples = 114 value = [114, 0]
X[1] <= 0.403 gini = 0.2 samples = 62 value = [55, 7]
gini = 0.0 samples = 1 value = [1, 0]
X[1] <= 1.36 gini = 0.229 samples = 38 value = [5, 33]
X[1] <= -1.365 gini = 0.245 samples = 7 value = [1, 6]
X[1] <= 1.07 gini = 0.494 samples = 9 value = [5, 4]
gini = 0.0 samples = 24 value = [0, 24]
X[1] <= -0.713 gini = 0.292 samples = 45 value = [8, 37]
X[1] <= -0.06 gini = 0.153 samples = 60 value = [55, 5]
gini = 0.0 samples = 2 value = [0, 2]
X[1] <= 1.099 gini = 0.391 samples = 15 value = [4, 11]
X[0] <= 0.433 gini = 0.083 samples = 23 value = [1, 22]
X[0] <= 0.73 gini = 0.5 samples = 2 value = [1, 1]
gini = 0.0 samples = 5 value = [0, 5]
X[1] <= 0.201 gini = 0.408 samples = 7 value = [5, 2]
gini = 0.0 samples = 2 value = [0, 2]
X[0] <= 2.117 gini = 0.444 samples = 3 value = [2, 1]
X[0] <= 1.423 gini = 0.245 samples = 42 value = [6, 36]
gini = 0.0 samples = 31 value = [31, 0]
X[1] <= 0.027 gini = 0.285 samples = 29 value = [24, 5]
gini = 0.0 samples = 8 value = [0, 8]
X[0] <= -0.458 gini = 0.49 samples = 7 value = [4, 3]
gini = 0.0 samples = 20 value = [0, 20]
X[1] <= 1.781 gini = 0.444 samples = 3 value = [1, 2]
gini = 0.0 samples = 1 value = [0, 1]
gini = 0.0 samples = 1 value = [1, 0]
gini = 0.0 samples = 2 value = [2, 0]
X[0] <= 0.73 gini = 0.48 samples = 5 value = [3, 2]
gini = 0.0 samples = 2 value = [2, 0]
gini = 0.0 samples = 1 value = [0, 1]
X[1] <= -0.626 gini = 0.397 samples = 22 value = [6, 16]
gini = 0.0 samples = 20 value = [0, 20]
gini = 0.0 samples = 1 value = [0, 1]
X[1] <= 0.259 gini = 0.245 samples = 28 value = [24, 4]
gini = 0.0 samples = 2 value = [2, 0]
X[0] <= -0.359 gini = 0.48 samples = 5 value = [2, 3]
gini = 0.0 samples = 1 value = [0, 1]
X[0] <= 0.532 gini = 0.5 samples = 2 value = [1, 1]
gini = 0.0 samples = 1 value = [1, 0]
X[1] <= 0.317 gini = 0.5 samples = 4 value = [2, 2]
gini = 0.0 samples = 1 value = [1, 0]
X[0] <= 1.027 gini = 0.363 samples = 21 value = [5, 16]
X[1] <= 0.085 gini = 0.298 samples = 22 value = [18, 4]
gini = 0.0 samples = 6 value = [6, 0]
gini = 0.0 samples = 1 value = [0, 1]
X[0] <= -0.161 gini = 0.5 samples = 4 value = [2, 2]
gini = 0.0 samples = 1 value = [1, 0]
gini = 0.0 samples = 1 value = [0, 1]
gini = 0.0 samples = 1 value = [0, 1]
X[1] <= 0.447 gini = 0.444 samples = 3 value = [2, 1]
X[1] <= 2.027 gini = 0.165 samples = 11 value = [1, 10]
X[1] <= 0.331 gini = 0.48 samples = 10 value = [4, 6]
X[1] <= 0.056 gini = 0.142 samples = 13 value = [12, 1]
X[1] <= 0.114 gini = 0.444 samples = 9 value = [6, 3]
gini = 0.0 samples = 1 value = [1, 0]
X[0] <= 0.186 gini = 0.444 samples = 3 value = [1, 2]
gini = 0.0 samples = 1 value = [1, 0]
X[1] <= 0.65 gini = 0.5 samples = 2 value = [1, 1]
gini = 0.0 samples = 10 value = [0, 10]
gini = 0.0 samples = 1 value = [1, 0]
gini = 0.0 samples = 2 value = [2, 0]
X[0] <= 1.225 gini = 0.375 samples = 8 value = [2, 6]
X[0] <= 0.136 gini = 0.245 samples = 7 value = [6, 1]
gini = 0.0 samples = 6 value = [6, 0]
X[0] <= 0.235 gini = 0.5 samples = 2 value = [1, 1]
X[0] <= 0.136 gini = 0.408 samples = 7 value = [5, 2]
X[1] <= 1.244 gini = 0.5 samples = 2 value = [1, 1]
gini = 0.0 samples = 1 value = [0, 1]
gini = 0.0 samples = 1 value = [0, 1]
gini = 0.0 samples = 1 value = [1, 0]
gini = 0.0 samples = 3 value = [0, 3]
X[1] <= 1.926 gini = 0.48 samples = 5 value = [2, 3]
gini = 0.0 samples = 5 value = [5, 0]
X[0] <= 0.235 gini = 0.5 samples = 2 value = [1, 1]
gini = 0.0 samples = 1 value = [1, 0]
gini = 0.0 samples = 1 value = [0, 1]
X[0] <= -0.013 gini = 0.5 samples = 4 value = [2, 2]
gini = 0.0 samples = 3 value = [3, 0]
gini = 0.0 samples = 1 value = [1, 0]
gini = 0.0 samples = 1 value = [0, 1]
gini = 0.0 samples = 2 value = [2, 0]
gini = 0.0 samples = 3 value = [0, 3]
gini = 0.0 samples = 1 value = [0, 1]
gini = 0.0 samples = 1 value = [1, 0]
X[1] <= 0.201 gini = 0.444 samples = 3 value = [2, 1]
gini = 0.0 samples = 1 value = [0, 1]
gini = 0.0 samples = 2 value = [2, 0]
gini = 0.0 samples = 1 value = [0, 1]

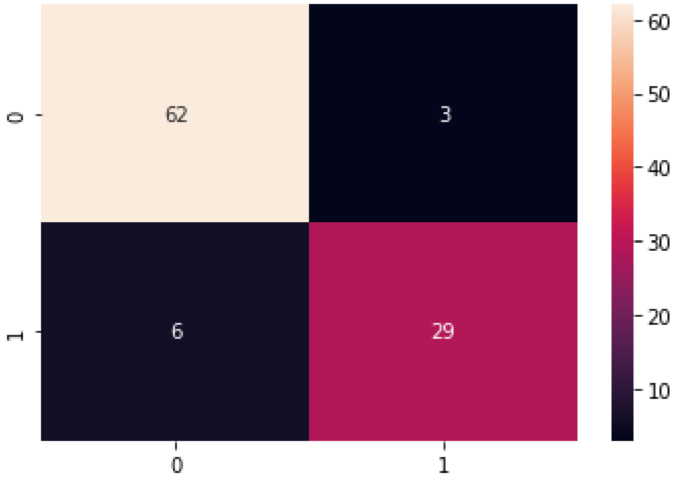
In [132]:

```
# importing the plot tree method
from sklearn.tree import DecisionTreeClassifier, plot_tree
clf = DecisionTreeClassifier()

# output size of decision tree
plt.figure(figsize=(30,20))

# providing the training dataset
clf = clf.fit(X_test, y_test)
plot_tree(clf, filled=True)
plt.title("Decision tree training for testing dataset")
plt.show()
```


Decision tree training for testing dataset
X[0] <= 0.433 gini = 0.435 samples = 100 value = [68, 32]
X[1] <= 0.664 gini = 0.193 samples = 74 value = [66, 8]
X[1] <= 0.432 gini = 0.142 samples = 26 value = [2, 24]
X[0] <= 0.334 gini = 0.059 samples = 66 value = [64, 2]
X[0] <= -0.31 gini = 0.375 samples = 8 value = [2, 6]
X[1] <= -0.785 gini = 0.26 samples = 13 value = [2, 11]
gini = 0.0 samples = 13 value = [0, 13]
X[0] <= -0.211 gini = 0.032 samples = 62 value = [61, 1]
X[1] <= 0.288 gini = 0.375 samples = 4 value = [3, 1]
X[0] <= -0.854 gini = 0.5 samples = 4 value = [2, 2]
gini = 0.0 samples = 4 value = [0, 4]
gini = 0.0 samples = 9 value = [0, 9]
X[1] <= -0.524 gini = 0.5 samples = 4 value = [2, 2]
gini = 0.0 samples = 48 value = [48, 0]
X[1] <= 0.244 gini = 0.133 samples = 14 value = [13, 1]
gini = 0.0 samples = 3 value = [3, 0]
gini = 0.0 samples = 1 value = [0, 1]
gini = 0.0 samples = 1 value = [0, 1]
X[1] <= 1.853 gini = 0.444 samples = 3 value = [2, 1]
gini = 0.0 samples = 1 value = [1, 0]
X[1] <= 0.056 gini = 0.444 samples = 3 value = [1, 2]
gini = 0.0 samples = 12 value = [12, 0]
X[1] <= 0.288 gini = 0.5 samples = 2 value = [1, 1]
gini = 0.0 samples = 2 value = [2, 0]
gini = 0.0 samples = 1 value = [0, 1]
gini = 0.0 samples = 1 value = [0, 1]
X[0] <= 2.117 gini = 0.5 samples = 2 value = [1, 1]
gini = 0.0 samples = 1 value = [0, 1]
gini = 0.0 samples = 1 value = [1, 0]
gini = 0.0 samples = 1 value = [1, 0]
gini = 0.0 samples = 1 value = [0, 1]

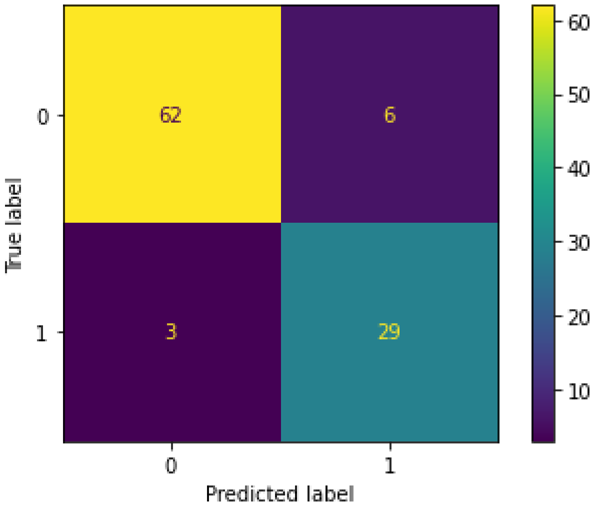
In [124]:

```
# importing the required modules
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns

# Importing the dataset using pandas module
dataset = pd.read_csv('decisionTree_Data.csv')

# splitting the dataset into input and output datasets
X = dataset.iloc[:, [0,1]].values
y = dataset.iloc[:, 2].values

# splitting the dataaset into Training and Testing Data
from sklearn.model_selection import train_test_split

# random state is 0 and test size if 25%
X_train, X_test, y_train, y_test =train_test_split(X,y,test_size=0.25, random_sta

# importing standard scalling method from sklearn
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()

# providing the inputs for the scalling purpose
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# importing decision tree algorithm
from sklearn.tree import DecisionTreeClassifier

# entropy means information gain
classifer=DecisionTreeClassifier(criterion='entropy', random_state=0)

# providing the training dataset
classifer.fit(X_train,y_train)
y_pred= classifer.predict(X_test)

# creating confusion matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,y_pred)

# Making the Confusion Matrix
cm = confusion_matrix(y_pred, y_test)
sns.heatmap(cm,annot=True)
plt.savefig('confusion.png')
```



In [125]:

```
# importing the required modules
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay

# Plot the confusion matrix in graph
cm = confusion_matrix(y_test,y_pred, labels=classifer.classes_)

# ploting with labels
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=classifer.class
disp.plot()

# showing the matrix
plt.show()
```



In [126]:

```
# importing the required module and methods
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_sco
print(f'Accuracy-score: {accuracy_score(y_test, y_pred):.3f}')
print(f'Precision-score: {precision_score(y_test, y_pred):.3f}')
print(f'Recall-score: {recall_score(y_test, y_pred):.3f}')
print(f'F1-score: {f1_score(y_test, y_pred):.3f}')
```

```
Accuracy-score: 0.910
Precision-score: 0.829
Recall-score: 0.906
F1-score: 0.866
```

In [127]:

```
# importing the tree
from sklearn import tree

# text based tree
text_representation = tree.export_text(clf)
print(text_representation)
```

```
|   |   |   |   |   |   |--- feature_4 >  -0.32
|   |   |   |   |   |   |   |--- feature_0 <= 1.61
|   |   |   |   |   |   |   |   |--- class: 1
|   |   |   |   |   |   |   |--- feature_0 >  1.61
|   |   |   |   |   |   |   |   |--- feature_2 <= 2.67
|   |   |   |   |   |   |   |   |   |--- class: 1
|   |   |   |   |   |   |   |   |--- feature_2 >  2.67
|   |   |   |   |   |   |   |   |   |--- feature_0 <= 1.72
|   |   |   |   |   |   |   |   |   |   |--- class: 0
|   |   |   |   |   |   |   |   |   |--- feature_0 >  1.72
|   |   |   |   |   |   |   |   |   |   |--- class: 1
|   |   |   |   |   |--- feature_1 >  -0.23
|   |   |   |   |   |   |--- class: 1
|   |   |   |   |--- feature_0 >  3.51
|   |   |   |   |   |--- feature_2 <= 1.19
|   |   |   |   |   |   |--- class: 1
|   |   |   |   |   |--- feature_2 >  1.19
|   |   |   |   |   |   |--- class: 0
```

In [ ]:

In [ ]:

In [ ]: