

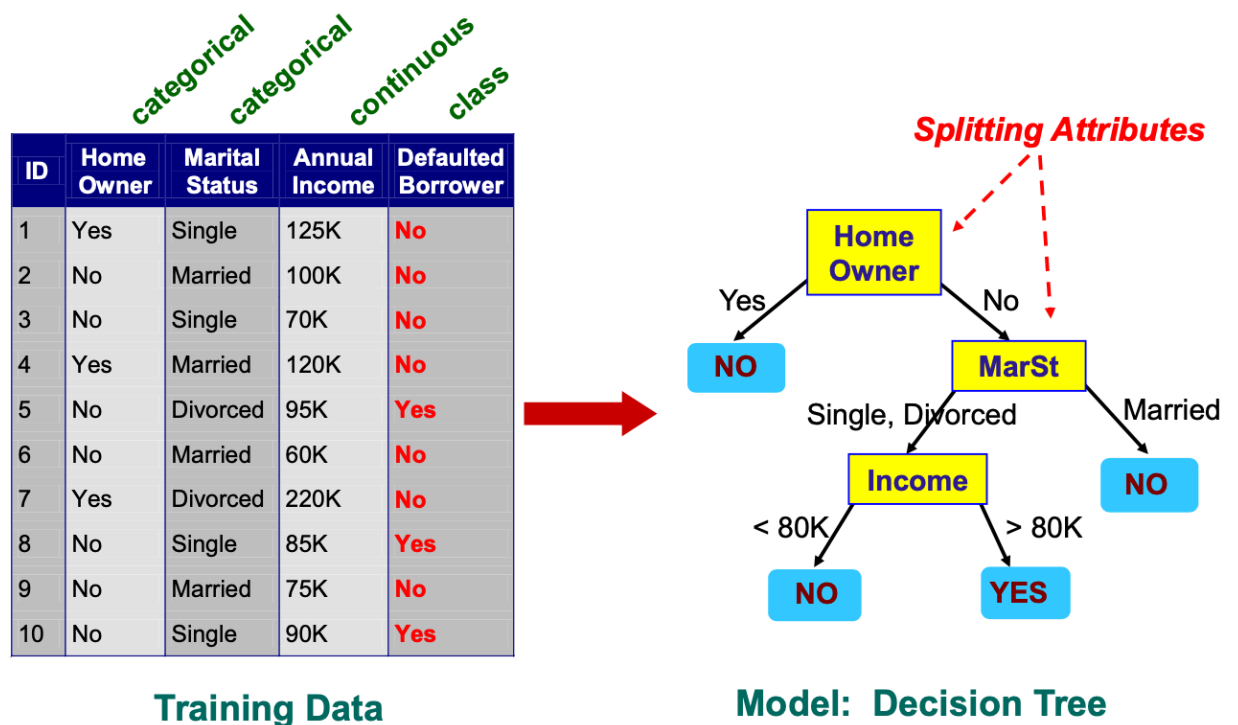
DS 440 Data Mining

Lecture 20: Decision Trees

```
In [2]: import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
```

Intuition of a decision tree model

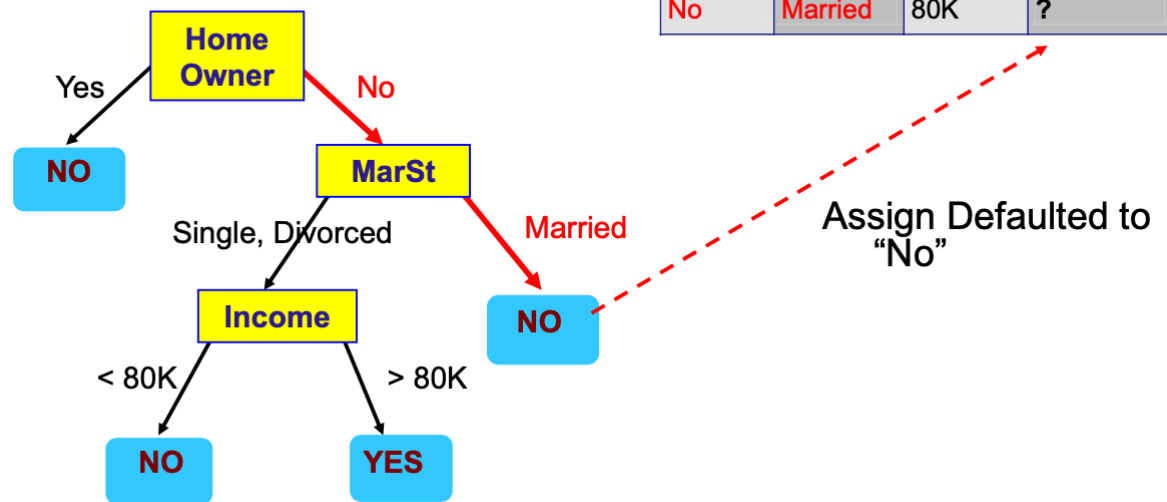
1. A trained decision tree model



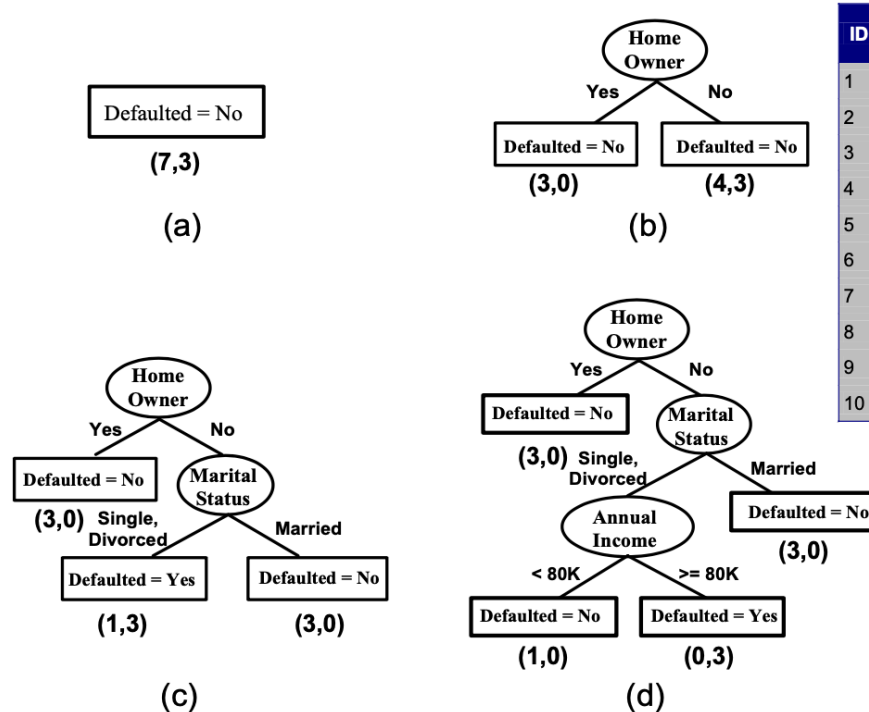
2. Predicting on the test data using this model

Test Data

Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?



3. Hunt's Algorithm for constructing a decision tree



ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

2. Questions to answer

1. How to select the feature to split
2. How to choose the splitting criterion for the selected feature
3. What should be the termination criterion for the algorithm

Answer to 1 and 2

Compute Measure of node impurity: Gini Index

$$\text{Gini Index (GI)} = 1 - \sum_{i=0}^{c-1} p_i^2 \quad (1)$$

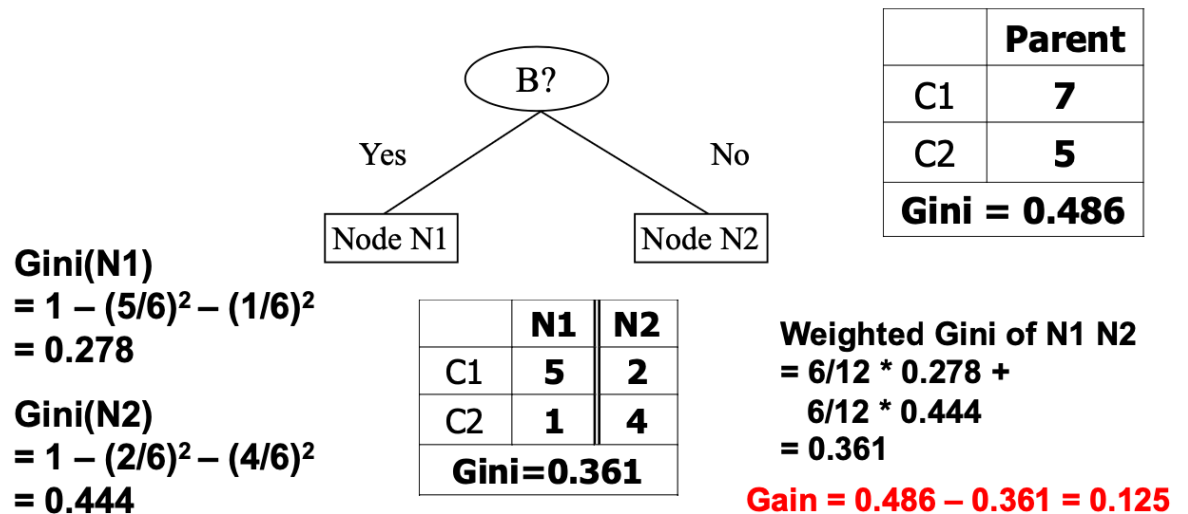
Here p_i is the frequency of class i and c is the total number of classes

Hence we follow the procedure below to find the best split:

1. Compute impurity measure (P) before splitting
2. Compute impurity measure (M) after splitting
 - Compute impurity measure of each child node
 - M is the weighted impurity of child nodes
3. Choose feature splitting criterion that produces the highest gain

$$\text{Gain} = P - M \quad (2)$$

An example Gain computation



Answer to 3

1. Stop splitting if all node become pure node.
2. Stop splitting early. This also helps to avoid overfitting.

3.Using sklearn to fit a decision tree model

```
In [4]: iris = load_iris()
```

```
In [5]: X = iris.data
y = iris.target
print(X.shape,y.shape)

(150, 4) (150,)
```

```
In [8]: iris.feature_names
```

```
Out[8]: ['sepal length (cm)',
'sepal width (cm)',
'petal length (cm)',
'petal width (cm)']
```

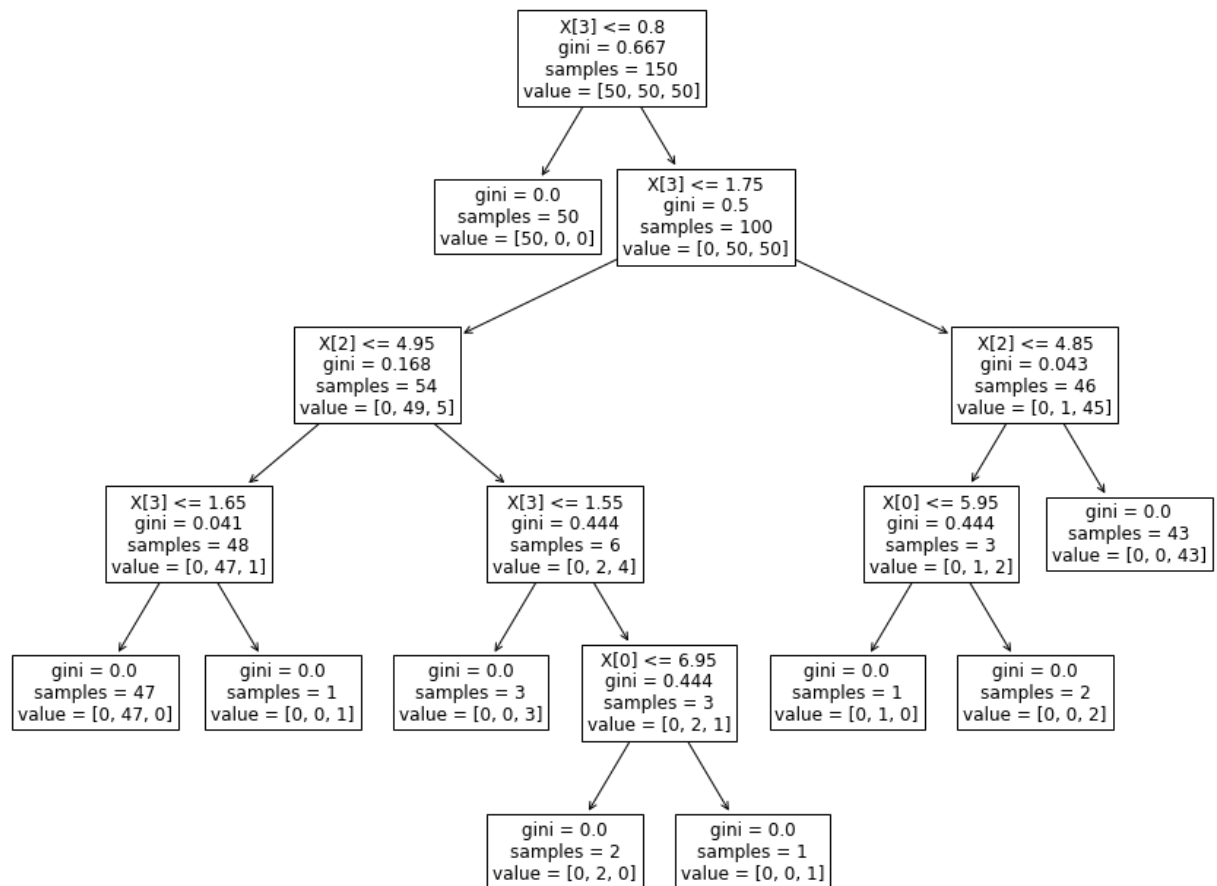
```
In [6]: %%time
decision_tree = DecisionTreeClassifier()
decision_tree = decision_tree.fit(X,y)
```

CPU times: user 1.62 ms, sys: 1.33 ms, total: 2.94 ms
Wall time: 2.02 ms

```
In [7]: from sklearn.tree import export_text
r = export_text(decision_tree, feature_names=iris.feature_names)
print(r)
```

```
|--- petal width (cm) <= 0.80
|   |--- class: 0
|--- petal width (cm) > 0.80
|   |--- petal width (cm) <= 1.75
|   |   |--- petal length (cm) <= 4.95
|   |   |   |--- petal width (cm) <= 1.65
|   |   |   |   |--- class: 1
|   |   |   |   |--- petal width (cm) > 1.65
|   |   |   |   |   |--- class: 2
|   |   |--- petal length (cm) > 4.95
|   |   |   |--- petal width (cm) <= 1.55
|   |   |   |   |--- class: 2
|   |   |   |   |--- petal width (cm) > 1.55
|   |   |   |       |--- sepal length (cm) <= 6.95
|   |   |   |       |   |--- class: 1
|   |   |   |       |   |--- sepal length (cm) > 6.95
|   |   |   |       |   |   |--- class: 2
|   |--- petal width (cm) > 1.75
|   |   |--- petal length (cm) <= 4.85
|   |   |   |--- sepal length (cm) <= 5.95
|   |   |   |   |--- class: 1
|   |   |   |   |--- sepal length (cm) > 5.95
|   |   |   |   |   |--- class: 2
|   |   |--- petal length (cm) > 4.85
|   |   |   |--- class: 2
```

```
In [9]: plt.figure(figsize=(15,12))
tree.plot_tree(decision_tree, fontsize=12);
```



Train/test split

```
In [11]: 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,
```

```
In [12]: clf = tree.DecisionTreeClassifier()
clf.fit(X_train,y_train)
```

```
Out[12]: DecisionTreeClassifier()
```

```
In [13]: ypred = clf.predict(X_test)
ypred
```

```
Out[13]: array([0, 1, 2, 1, 2, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 2, 2, 2, 1, 1, 2,
1,
          1, 2, 1, 0, 1, 1, 0, 2, 2, 2, 1, 0, 1, 2, 0, 0, 0, 1, 0, 0, 0,
1,
          2])
```

```
In [14]: from sklearn.metrics import accuracy_score
accuracy_score(y_test,ypred)
```

```
Out[14]: 0.9333333333333333
```

```
In [15]: accuracy_score(y_train,clf.predict(X_train))
```

```
Out[15]: 1.0
```

```
In [ ]:
```