Classification using Decision Trees

Sources: Book – Machine Learning in R by Brett Lantz and few slides from Lecture Notes by Tan, Steinbach, Kumar

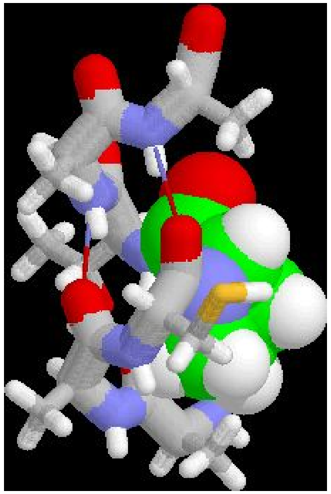
Classification: Definition

- Given a collection of records (*training set*)
  - Each record contains a set of *attributes*, one of the attributes is the *class*.
- Find a *model* for class attribute as a function of the values of other attributes.
- Goal: previously unseen records should be assigned a class as accurately as possible.
  - A *test set* is used to determine the accuracy of the model. Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it.

Examples of Classification Task

- Predicting tumor cells as benign or malignant

- Classifying credit card transactions as legitimate or fraudulent



- Classifying secondary structures of protein as alpha-helix, beta-sheet, or random coil
- Categorizing news stories as finance, weather, entertainment, sports, etc

Classification Techniques

- Decision Tree based Methods
- Rule-based Methods
- Memory based reasoning
- Neural Networks
- Naïve Bayes and Bayesian Belief Networks
- Support Vector Machines

Decision Tree

- It helps us explore the structure of a set of data, while developing easy to visualize decision rules for predicting a categorical (classification tree) or continuous (regression tree) outcome.
- Decision tree is an algorithm the can have a continuous or categorical dependent (DV) and independent variables (IV).
- DT uses, Recursive partitioning, which is a fundamental tool

Example 1

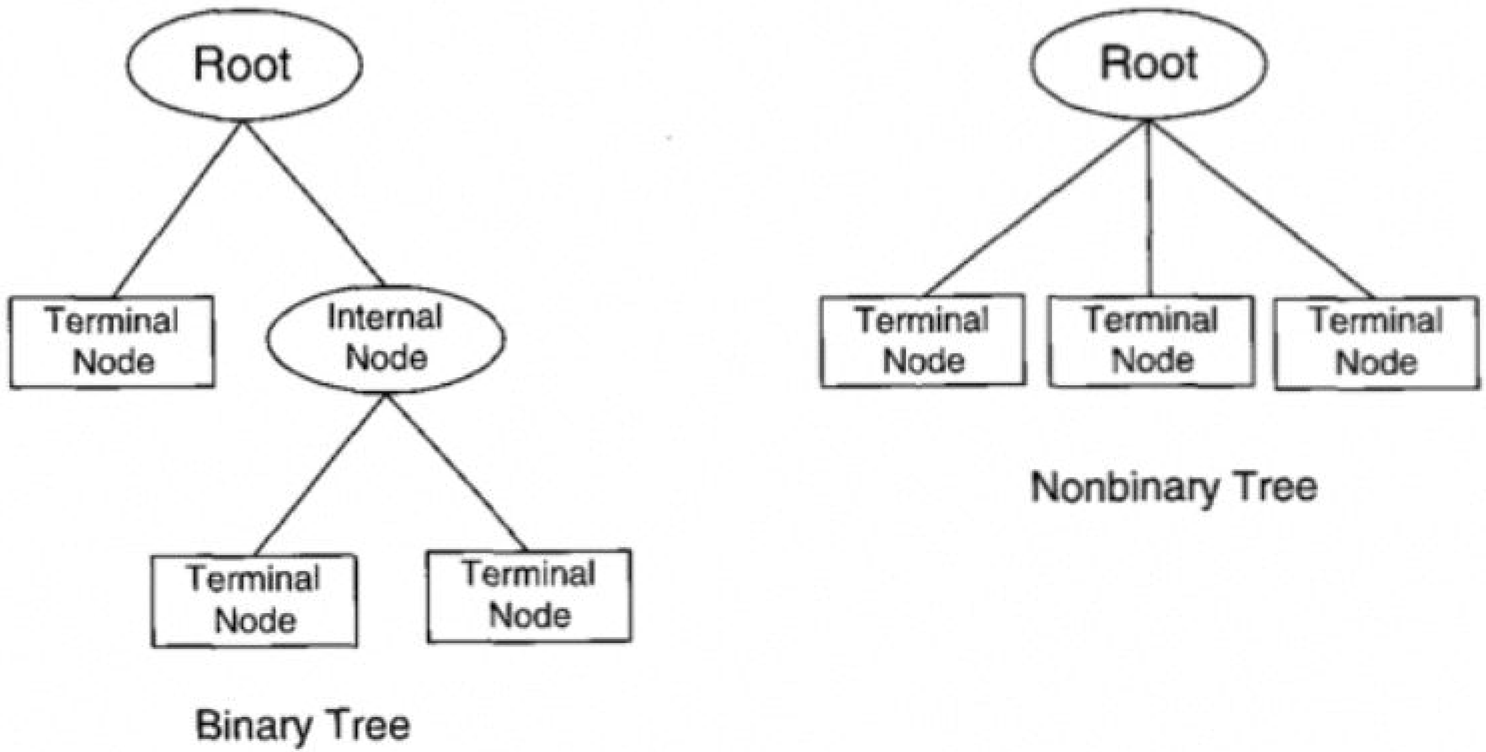


Fig. 5. Types of decision tree.

Advantages to using trees

- **Simple to understand and interpret**.
  - People are able to understand decision tree models after a brief explanation.
- **Requires little data preparation**.
  - Other techniques often require data normalization, dummy variables need to be created and blank values to be removed.
- **Able to handle both numerical and categorical data**.

Advantages to using trees

- **Uses a white box model.**
  - If a given situation is observable in a model the explanation for the condition is easily explained by Boolean logic
- **Possible to validate a model using statistical tests.**
  - That makes it possible to account for the reliability of the model.
- **Performs well with large data in a short time.**

General Algorithm

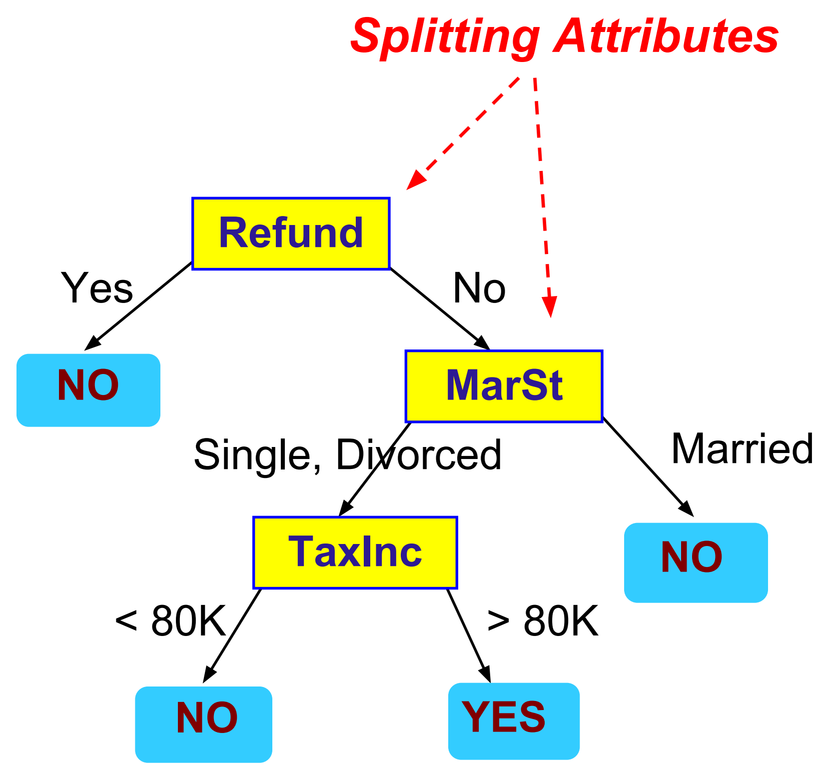
- To see how splitting a dataset can create a decision tree, imagine a bare root node that will grow into a mature tree.
- At first, the root node represents the entire dataset, since no splitting has transpired.
- Next, the decision tree algorithm must choose a feature to split upon; ideally, it chooses the feature most predictive of the target class.
- The examples are then partitioned into groups according to the distinct values of this feature, and the first set of tree branches are formed.
- Working down each branch, the algorithm continues to divide and conquer the data, choosing the best candidate feature each time to create another decision node, until a stopping criterion is reached.
- A way to split the data such that the resulting partitions contained examples primarily of a single class.

Example of a Decision Tree

categorical categorical continuous class

Tid	Refund	Marital Status	Taxable Income	Cheat
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

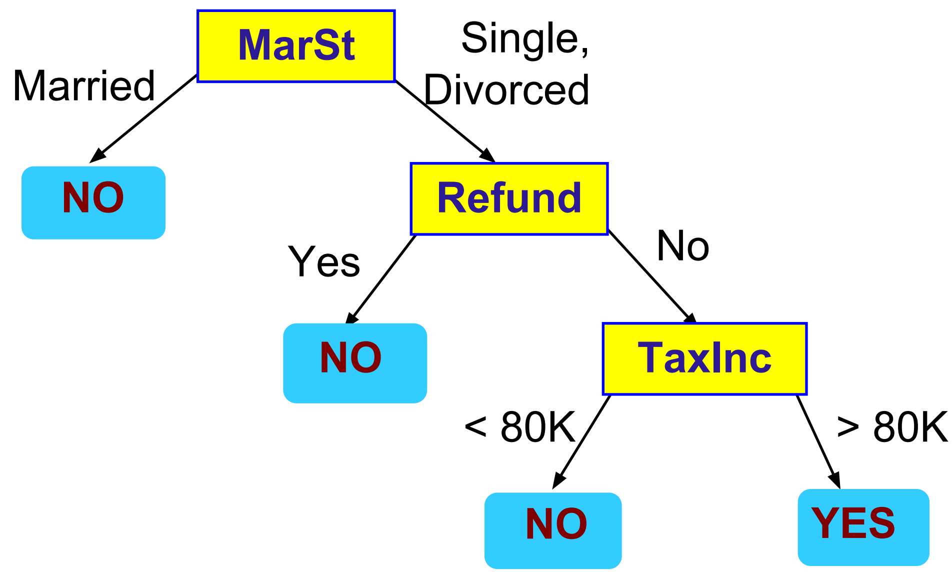
Training Data



Model: Decision Tree

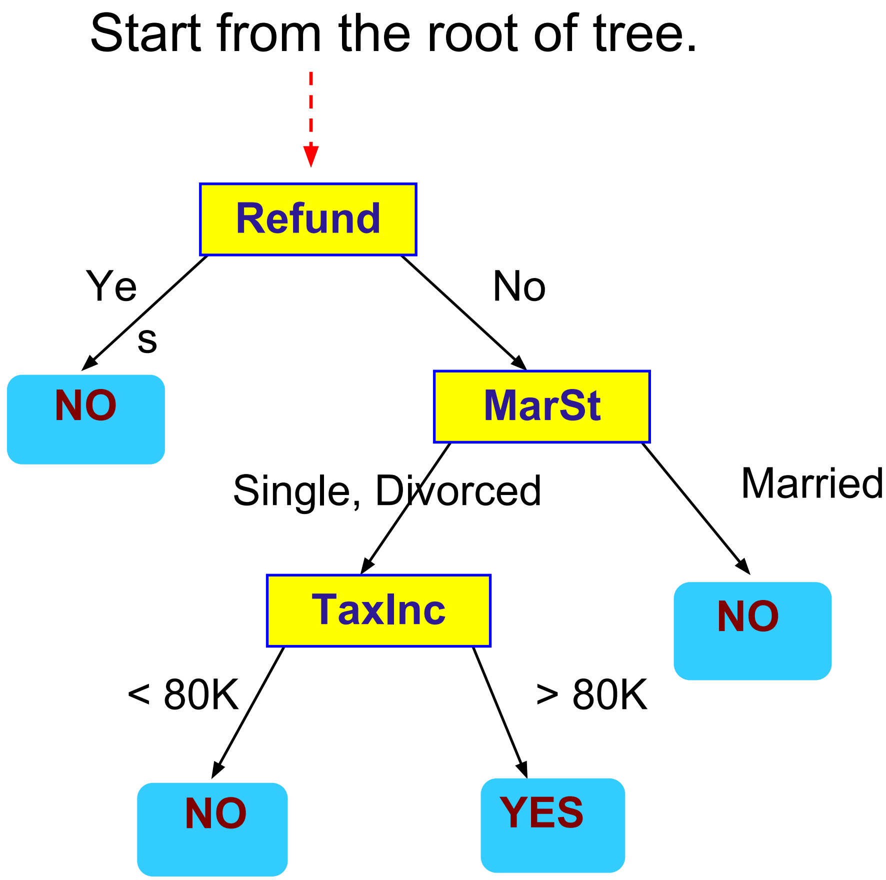
Another Example of Decision Tree

	categorical	categorical	continuous	class
Tid	**Refund**	**Marital Status**	**Taxable Income**	**Cheat**
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



There could be more than one tree that fits the same data!

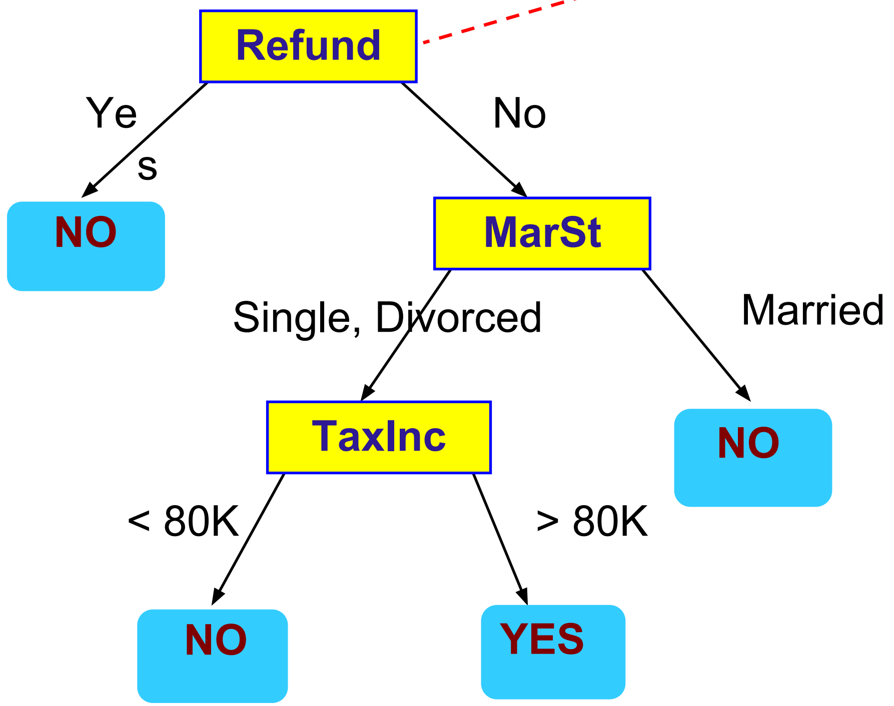
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data



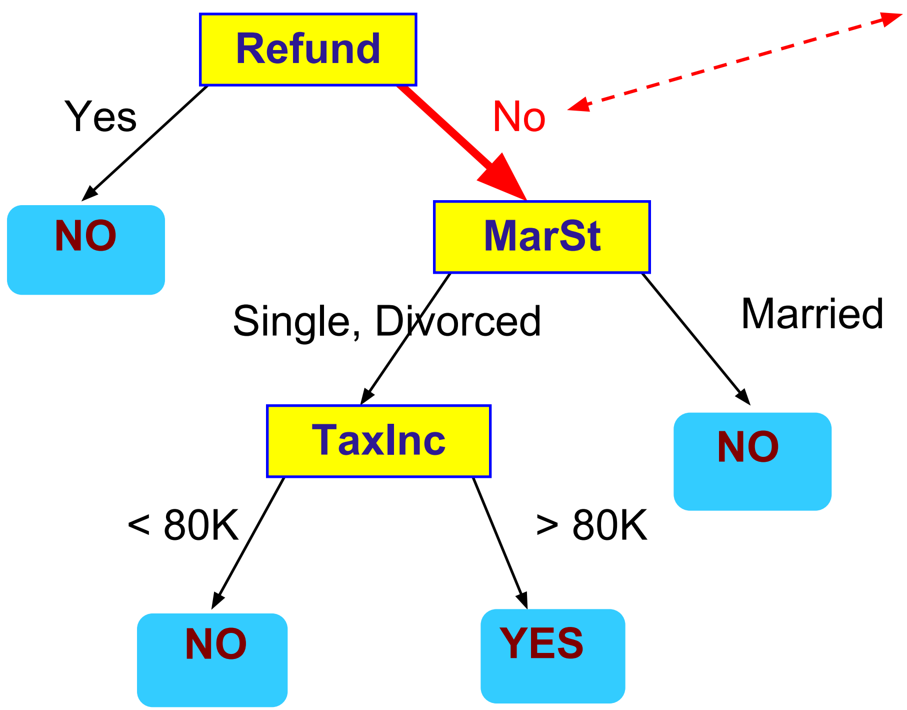
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

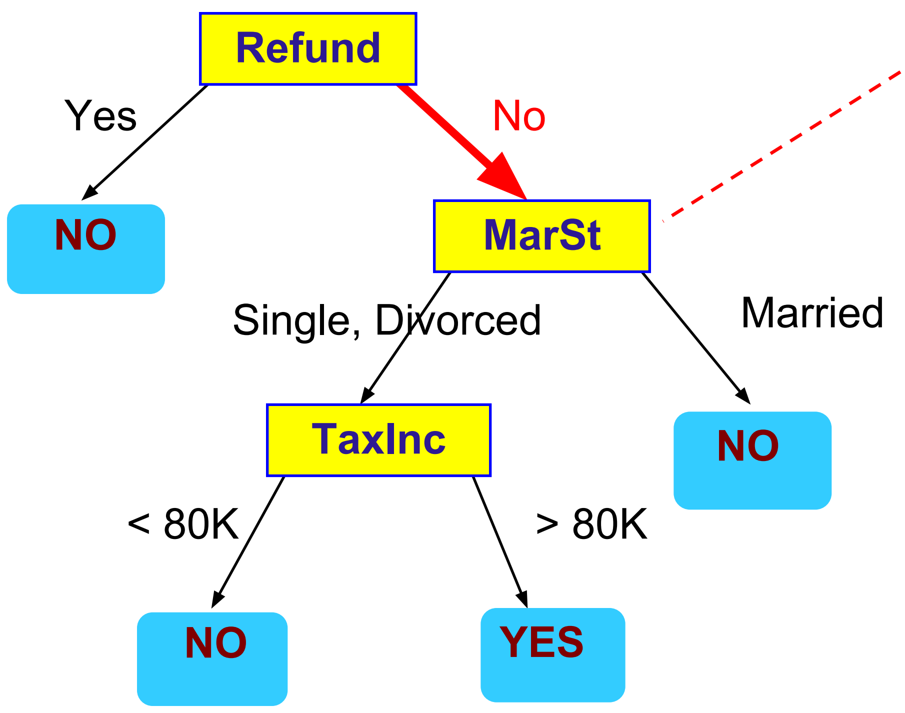
Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



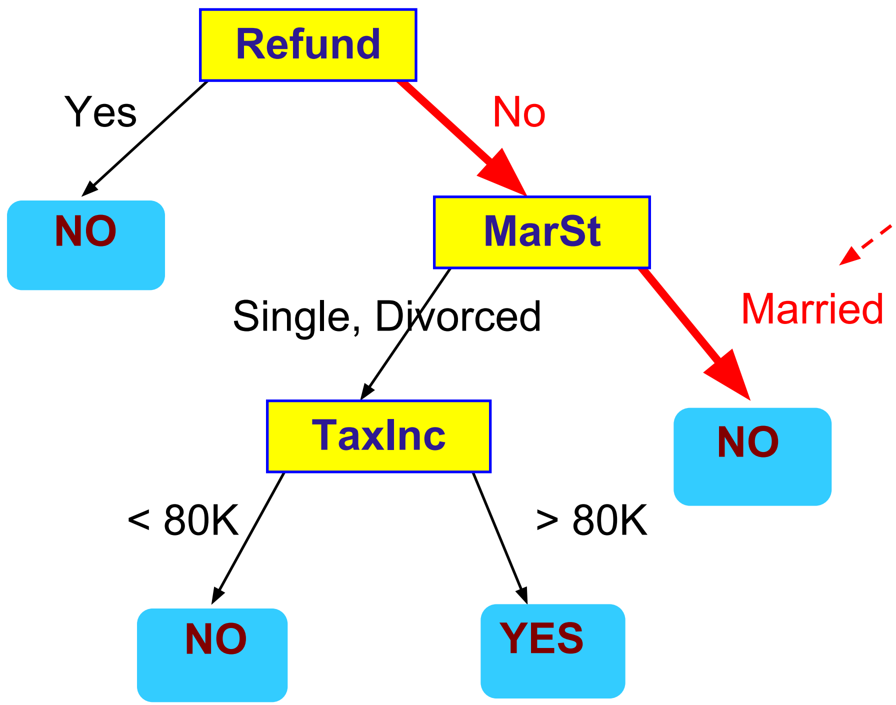
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data



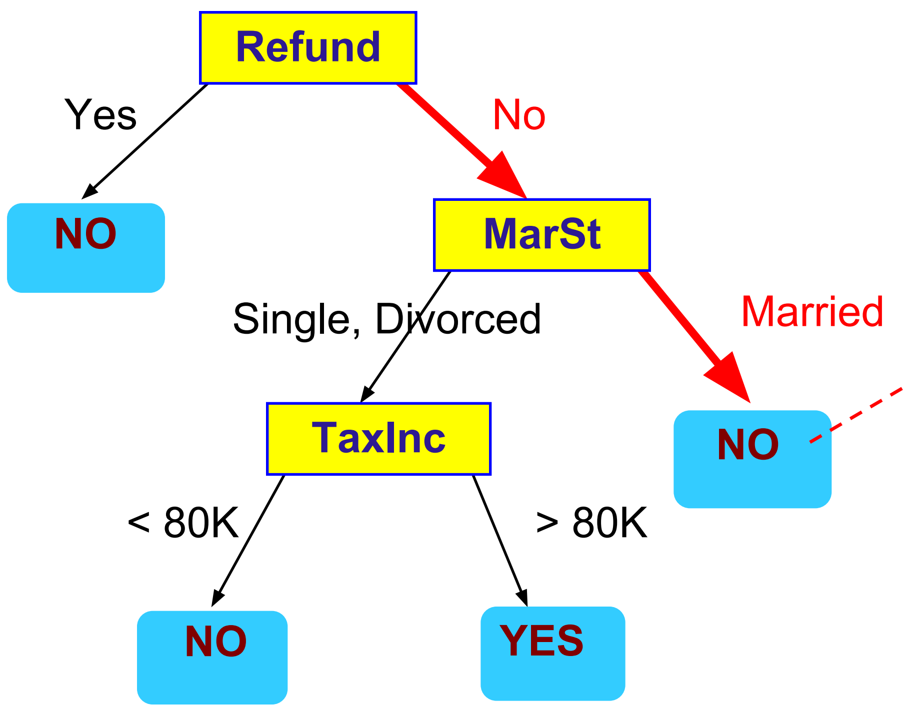
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

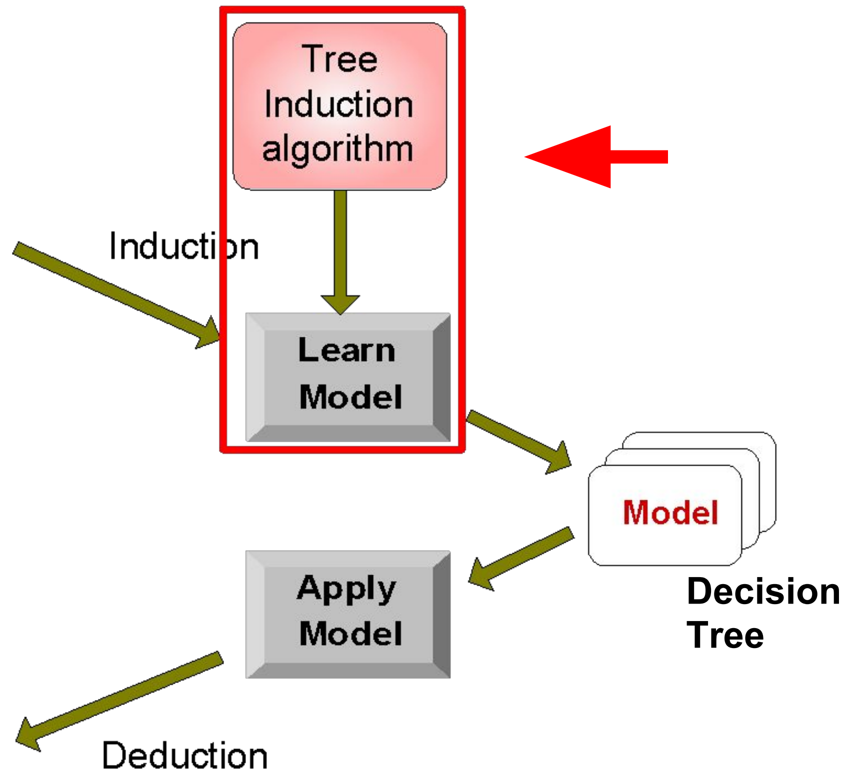


Assign Cheat to "No"

Decision Tree Classification Task

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set



Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set

Algorithms

- Decision trees algorithms are greedy so once test has been selected to partition the data other options will not be explored

- Popular Algorithms
  - Computer Science: ID3, C4.5, and C5.0
  - Statistics: Classification and Regression Trees (CART)

General Algorithm

- To construct tree T from training set S
  - If all examples in S belong to some class in C, or S is sufficiently "pure", then make a leaf labeled C.
  - Otherwise:
    - select the “most informative” attribute A
    - partition S according to A’s values
    - recursively construct sub-trees T1, T2, ..., for the subsets of S

- The details vary according to the specific algorithm – CART, ID3, C4.5 – but the general idea is the same

Tree Induction

- Greedy strategy.
  - Split the records based on an attribute test that optimizes certain criterion.

- Issues
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

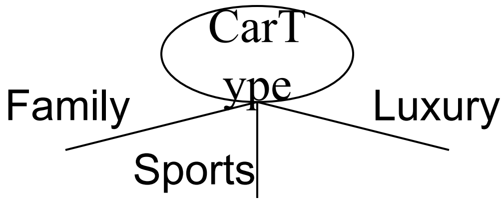
How to Specify Test Condition?

- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous

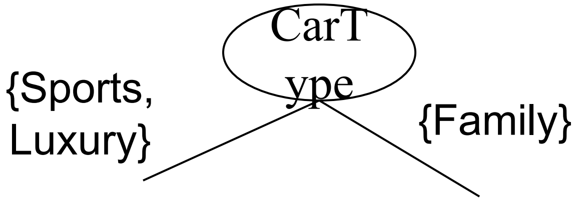
- Depends on number of ways to split
  - 2-way split
  - Multi-way split

Splitting Based on Nominal Attributes

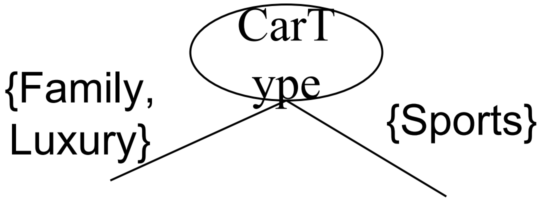
- Multi-way split: Use as many partitions as distinct values.



- Binary split: Divides values into two subsets.
  Need to find optimal partitioning.

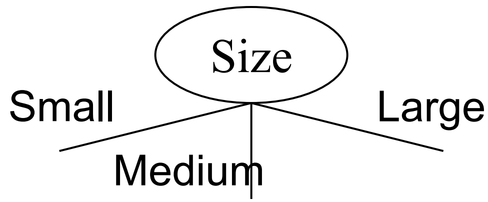


OR

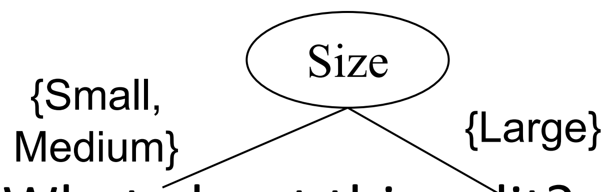


Splitting Based on Ordinal Attributes

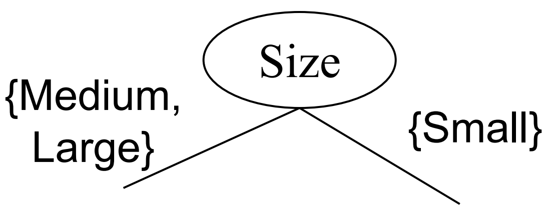
- Multi-way split: Use as many partitions as distinct values.



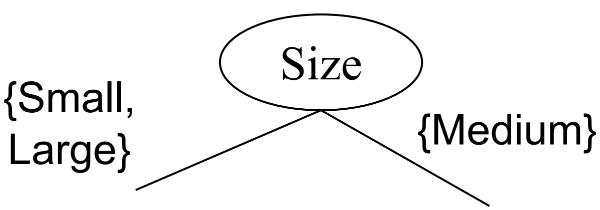
- Binary split: Divides values into two subsets.
  Need to find optimal partitioning.



OR



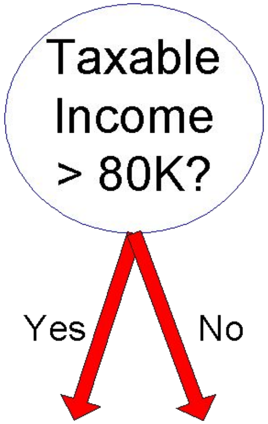
- What about this split?



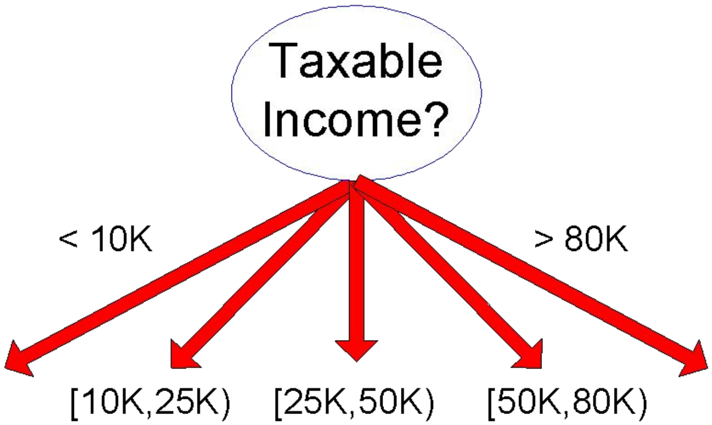
Splitting Based on Continuous Attributes

- Different ways of handling
  - Discretization to form an ordinal categorical attribute
    - Static – discretize once at the beginning
    - Dynamic – ranges can be found by equal interval bucketing, equal frequency bucketing (percentiles), or clustering.
  - Binary Decision: $(A < v)$ or $(A \geq v)$
    - consider all possible splits and finds the best cut
    - can be more compute intensive

Splitting Based on Continuous Attributes



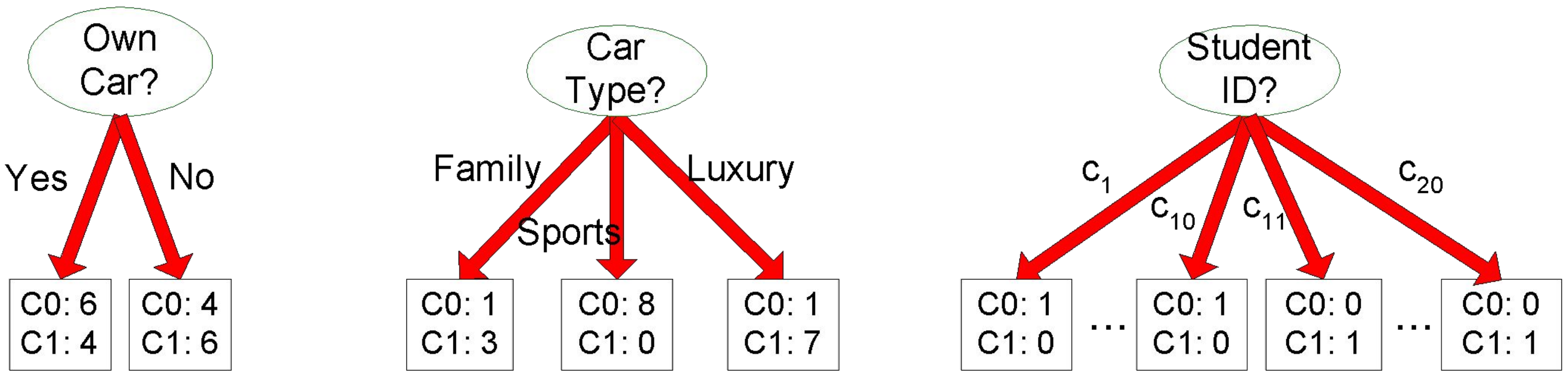
(i) Binary split



(ii) Multi-way split

How to determine the Best Split

Before Splitting: 10 records of class 0,
10 records of class 1



Which test condition is the best?

Measures of Node Impurity

- Gini Index
- Entropy & Information Gain
- Misclassification error

Stopping Criteria for Tree Induction

- Stop expanding a node when all the records belong to the same class

- Stop expanding a node when all the records have similar attribute values

- Early termination (to be discussed later)

Decision Tree Based Classification

- Advantages:
  - Inexpensive to construct
  - Extremely fast at classifying unknown records
  - Easy to interpret for small-sized trees
  - Accuracy is comparable to other classification techniques for many simple data sets

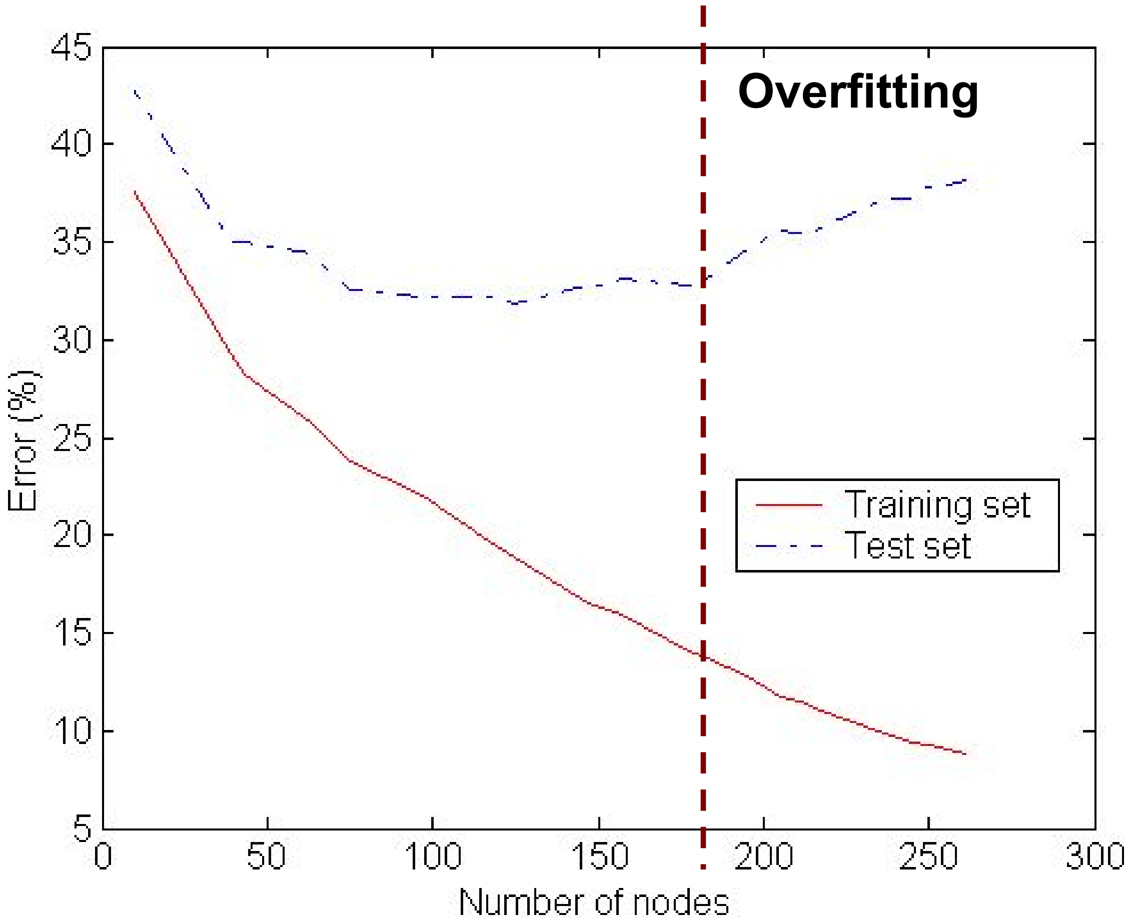
Practical Issues of Classification

- Underfitting and Overfitting

- Missing Values

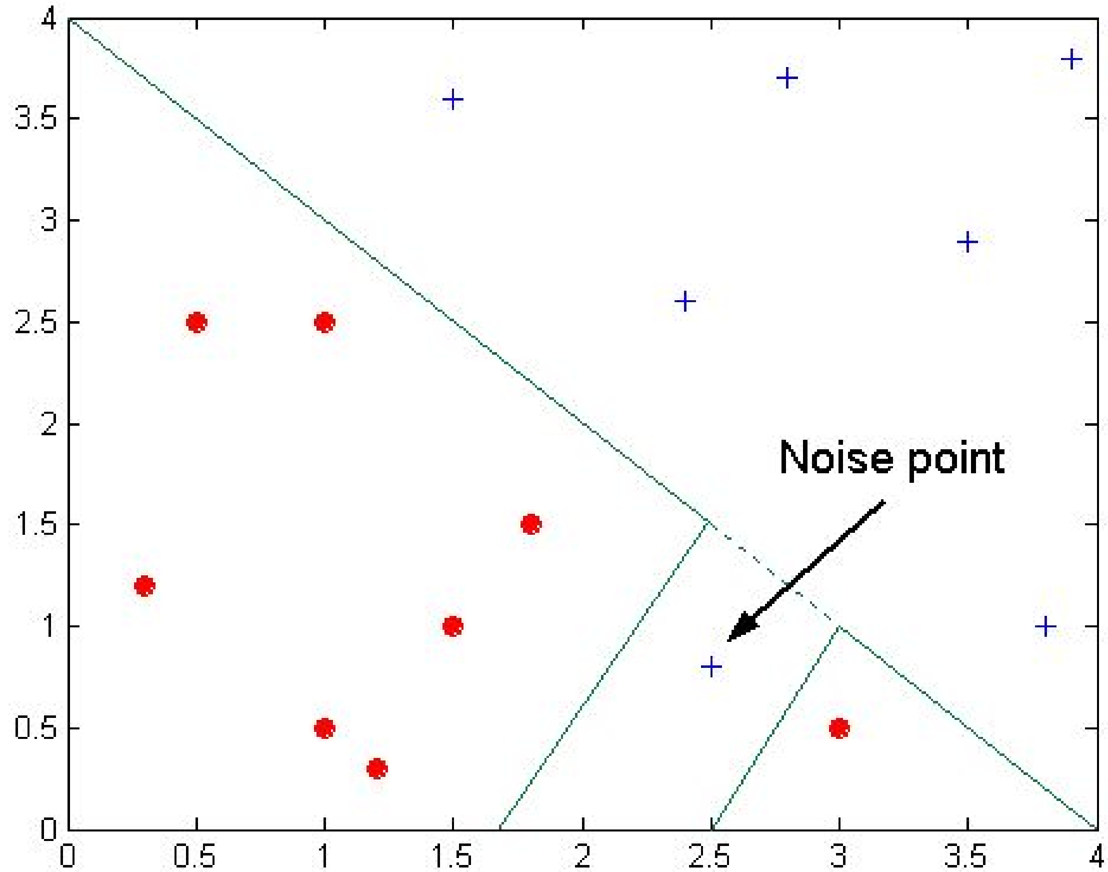
- Costs of Classification

Underfitting and Overfitting



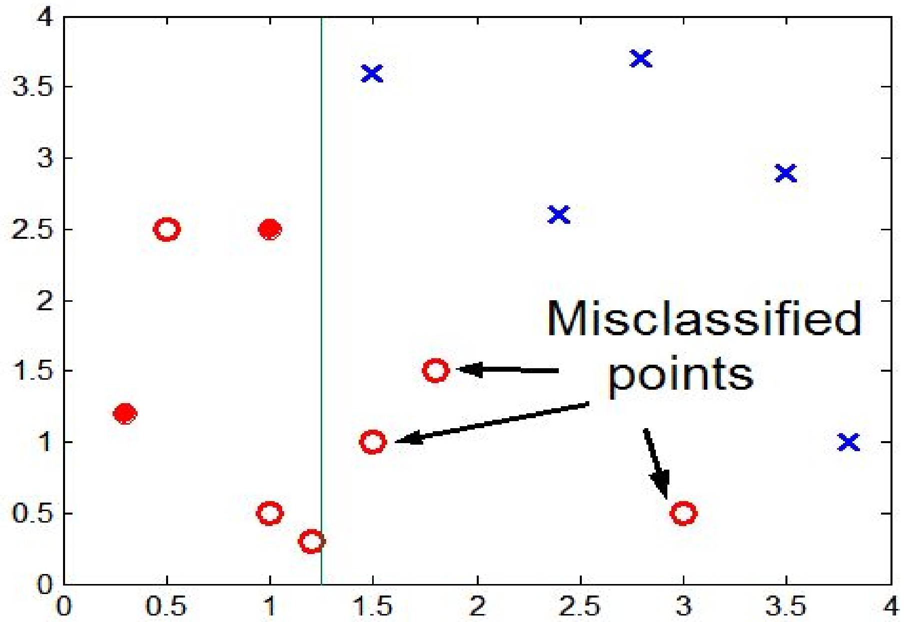
Underfitting: when model is too simple, both training and test errors are large

Overfitting due to Noise



Decision boundary is distorted by noise point

Overfitting due to Insufficient Examples



Lack of data points in the lower half of the diagram makes it difficult to predict correctly the class labels of that region

- Insufficient number of training records in the region causes the decision tree to predict the test examples using other training records that are irrelevant to the classification task

Notes on Overfitting

- Overfitting results in decision trees that are more complex than necessary
- Training error no longer provides a good estimate of how well the tree will perform on previously unseen records
- Need new ways for estimating errors

RPART, TREE, CTREE for IRIS Data

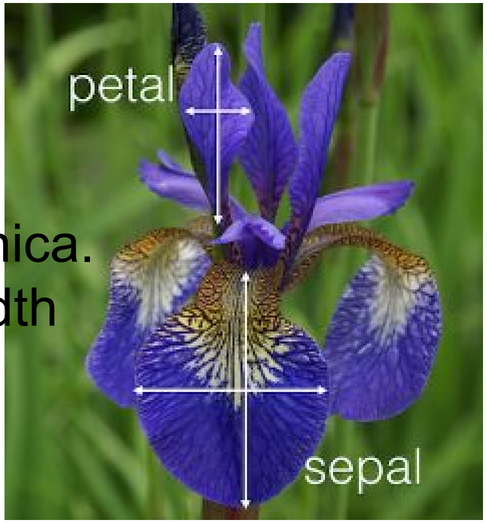
Dataset iris

- The iris dataset has been used for classification in many research publications. It consists of 50 samples from each of three classes of iris flowers [Frank and Asuncion, 2010]. One class is linearly separable from the other two, while the latter are not linearly separable from each other.

There are five attributes in the dataset:

Sepal.Length in cm,
Sepal.Width in cm,
Petal.Length in cm,
Petal.Width in cm, and
Species: Iris Setosa, Iris Versicolour, and Iris Virginica.

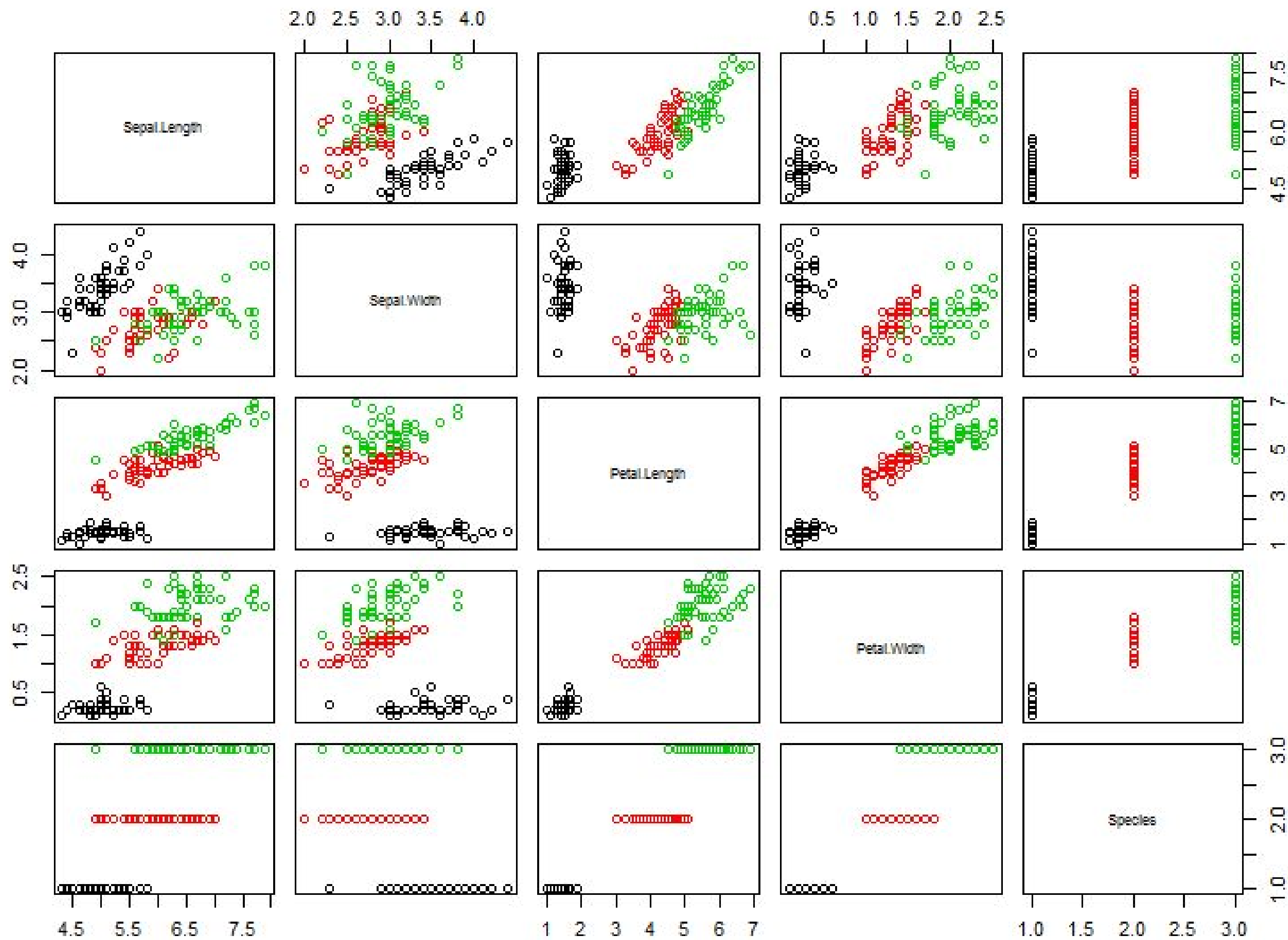
Sepal.Length, Sepal.Width, Petal.Length and Petal.Width are used to predict the Species of flowers.



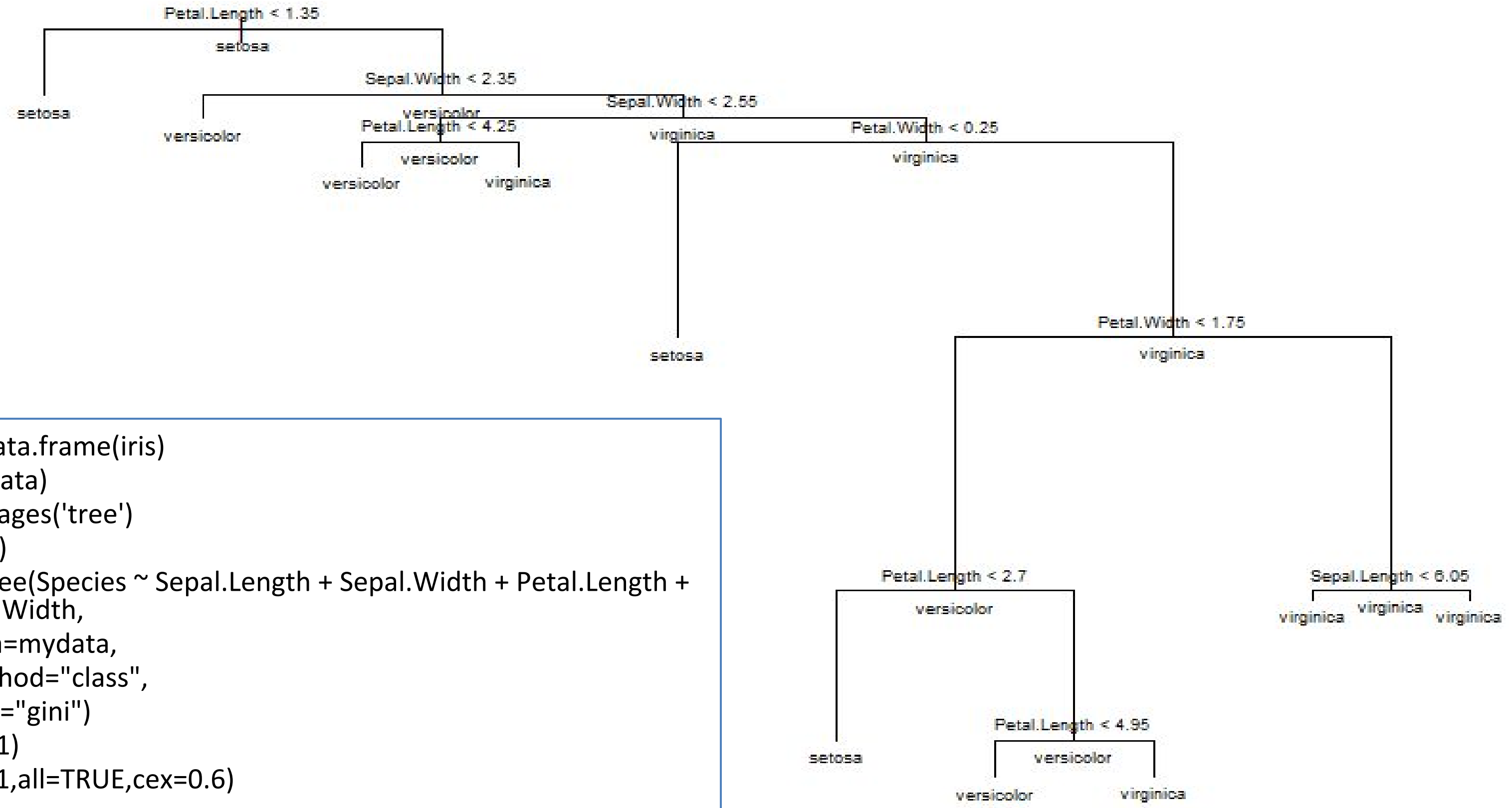
- head(iris)

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa

plot(iris, col=iris$Species)



TREE in R

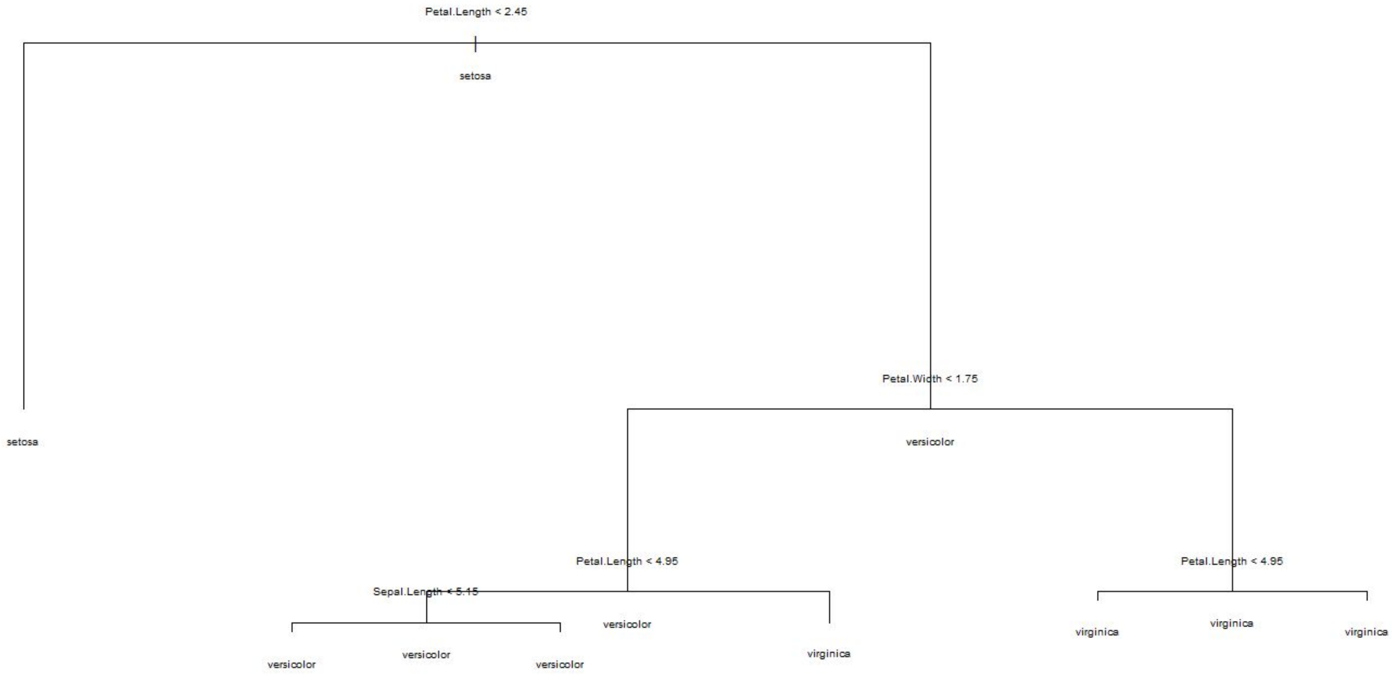



```
mydata<-data.frame(iris)
attach(mydata)
install.packages('tree')
library(tree)
model1<-tree(Species ~ Sepal.Length + Sepal.Width + Petal.Length +
        Petal.Width,
          data=mydata,
          method="class",
          split="gini")
plot(model1)
text(model1,all=TRUE,cex=0.6)
```

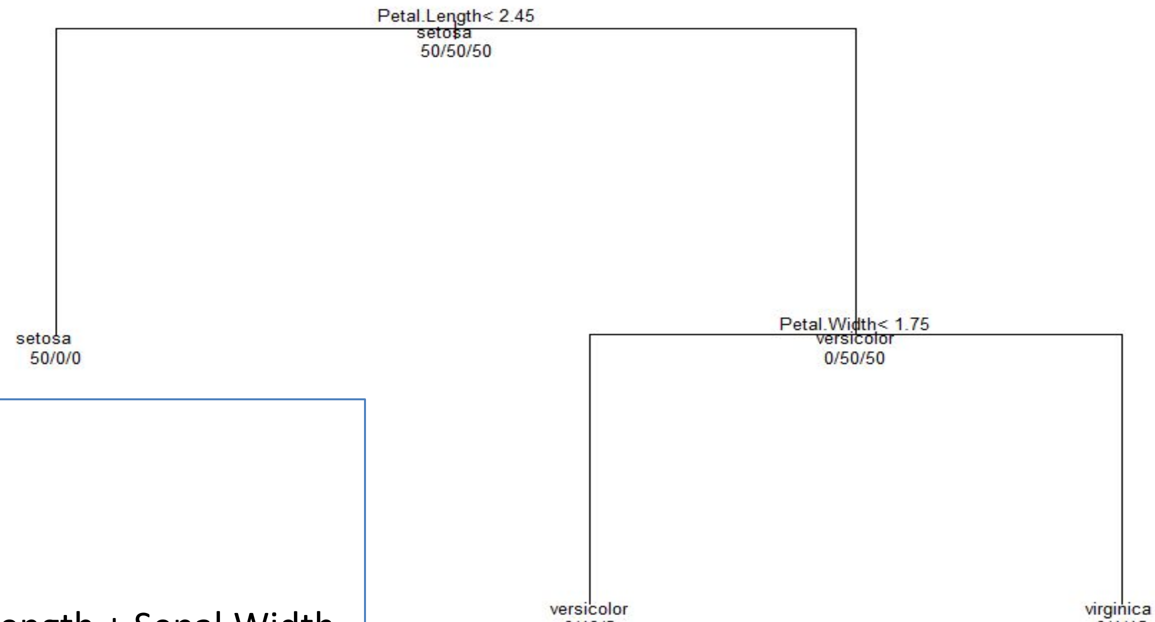
CTREE in R

```
mydata<-data.frame(iris)
attach(mydata)

install.packages('party')
library(party)
model2<-tree(Species ~ Sepal.Length + Sepal.Width + Petal.Length + Petal.Width, data=mydata, method="class", )
plot(model2)
text(model2,all=TRUE,cex=0.6)
```



RPART in R




```
mydata<-data.frame(iris)
attach(mydata)

library(rpart)
model<-rpart(Species ~ Sepal.Length + Sepal.Width
      + Petal.Length + Petal.Width,
        data=mydata,
        method="class")
plot(model)
text(model,use.n=TRUE,all=TRUE,cex=0.8)
```

Thank you