Predictive Analytics

Session Objectives

Give a quick introduction to predictive analytics

Introduction to classification problem using decision tree learning

Hands-on Lab: Building a decision tree classifier

Some Applications

Family and Personal Life

Location: Microsoft and Nokia predict future location based on cellular phone and location data.

Friendship and connection: Facebook and LinkedIn

Love:

- Match.com: Predict potential matches
- OkCupid: Which message content is most likely to elicit a response

Pregnancy: Target predicts customer pregnancy

Divorce and infidelity: University and clinical researcher can predict this as well!

Direct Marketing

Cox Communication: Tripled direct mail responses by predicting propensity to buy

Harrah's Las Vegas: The casino predicts how much a customer will spend over the long term

Target: Increased revenue 15-30 percent with predictive models

PREMIER Bankcard: Reduced mailing cost by $12 million

datasciencedojo
unleash the data scientist in you

Telcos, Retail and More

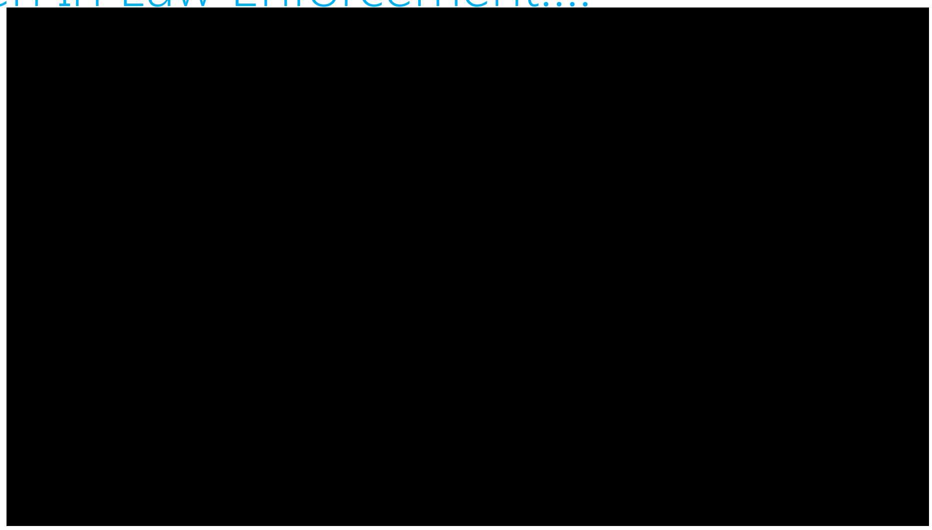
Fedex: predicts defection to a competitor with 65-90% accuracy

Telcos: Optus (Australia), Sprint, Telenor(Norway), 2degrees (New Zealand)

Amazon: 35% sales come from product recommendation

datasciencedojo
unleash the data scientist in you

Even In Law Enforcement….



datasciencedojo
unleash the data scientist in you

Decision trees

datasciencedojo
unleash the data scientist in you

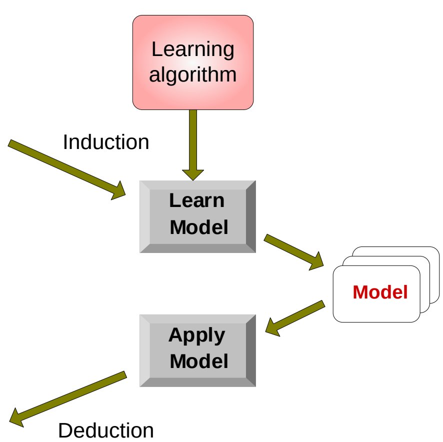
Classification: Definition

- 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 one of the class attributes 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.

Illustrating 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

datasciencedojo
unleash the data scientist in you

Examples of Classification Task

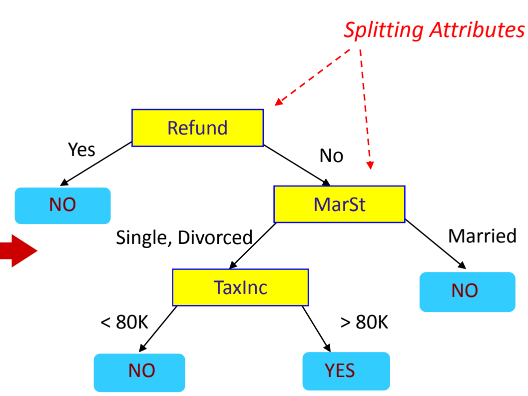
- Marketing: Customer churn
- Online: Bot detection in web traffic
- Medical: Predicting tumor cells as benign or malignant
- Finance: Credit card fraud detection
- Document Classification: Categorizing news stories as finance, weather, entertainment, sports, etc.
- Security/Surveillance: Face and fingerprint recognition

datasciencedojo
unleash the data scientist in you

Decision Tree classification

	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

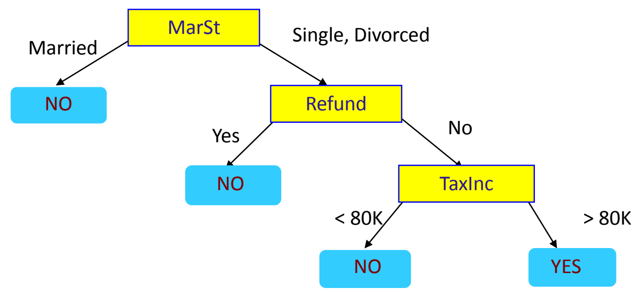


Model: Decision Tree

datasciencedojo
unleash the data scientist in you

A different Decision Tree

Tid	Refund (categorical)	Marital Status (categorical)	Taxable Income (continuous)	Cheat (class)
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!

datasciencedojo
unleash the data scientist in you

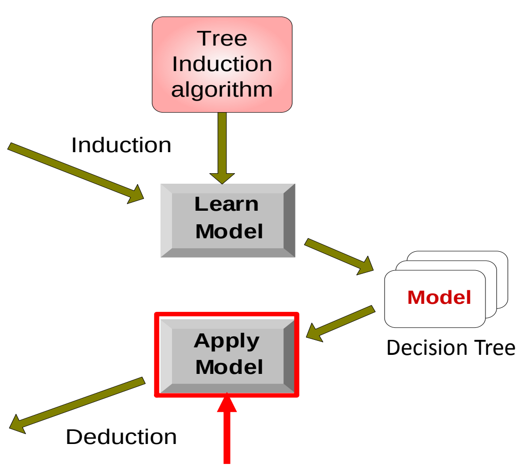
Decision Tree Classification Task

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	?

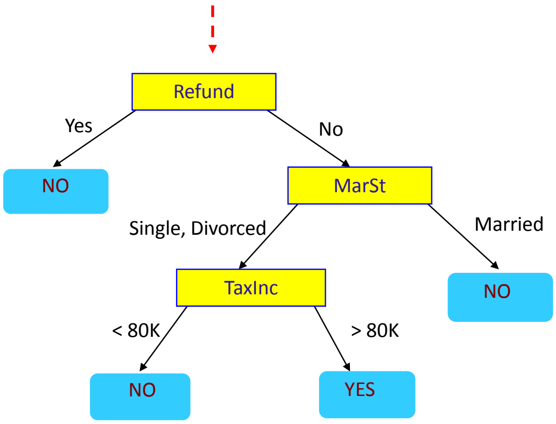
Test Set



datasciencedojo
unleash the data scientist in you

Apply Model to Test Data

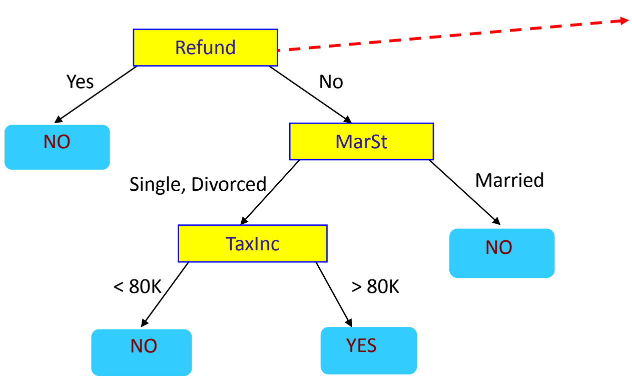
Start from the root of tree.



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

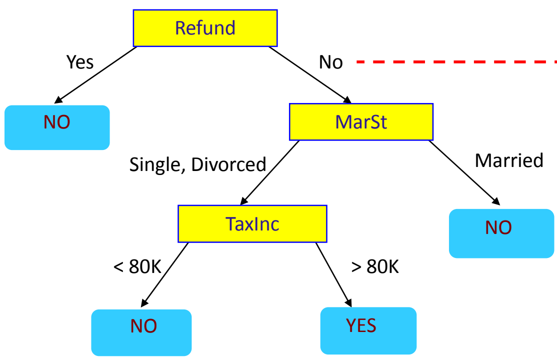
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

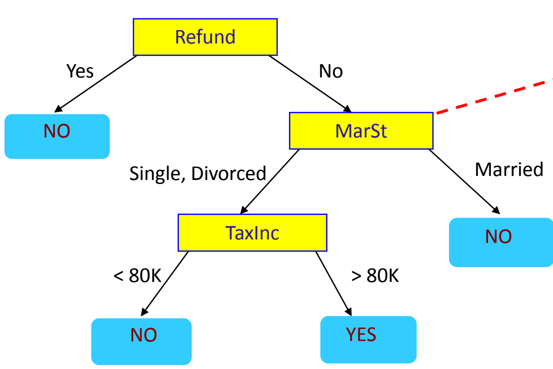
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

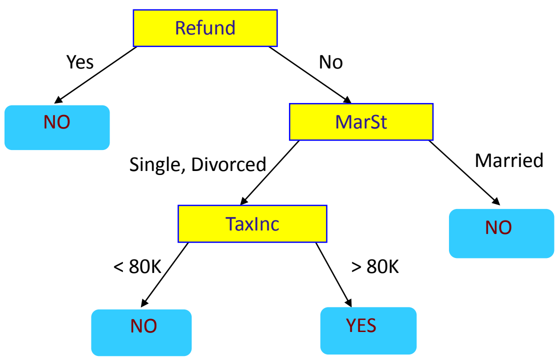
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

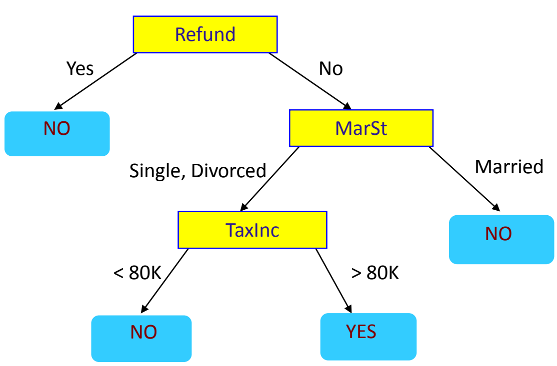
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data



Test Data

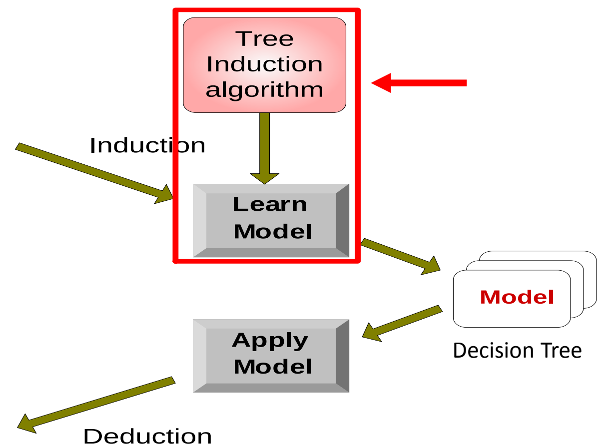
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Cheat = “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

How do we get a tree

- Exponentially many decision trees are possible
- Finding the optimal tree is infeasible
- Greedy methods that find sub-optimal solutions do exist

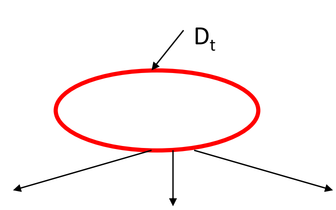
Decision Tree Induction

- Hunt's Algorithm (one of the earliest). Basis for many decision tree induction algorithms
  - CART
  - ID3
  - C4.5

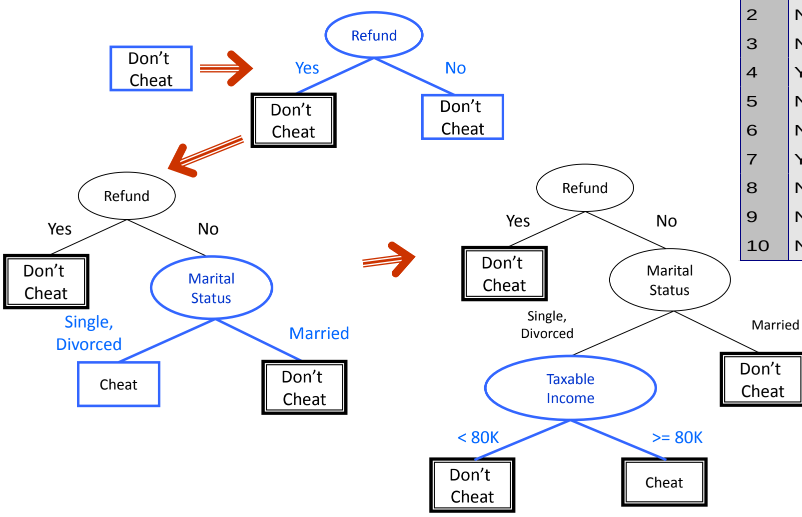
Hunt's Algorithm

- Let D_t be the set of training records that reach a node t
- General Procedure:
  - If D_t contains records that belong to the same class y_t, then t is a leaf node labeled as y_t
  - If D_t is an empty set, then t is a leaf node labeled by the default class, y_d
  - If D_t contains records that belong to more than one class, use an attribute test to split the data into smaller subsets. Recursively apply the procedure to each subset.

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



Hunt's Algorithm



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

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

datasciencedojo
unleash the data scientist in you

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

datasciencedojo
unleash the data scientist in you

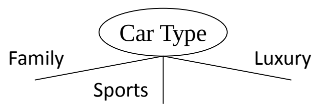
How to Specify Test Condition?

- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous
- Depends on number of ways to split
  - 2-way split
  - Multi-way split

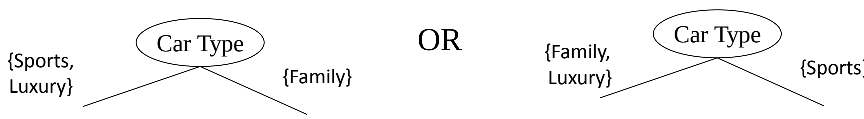
datasciencedojo
unleash the data scientist in you

Splitting on Nominal Attributes

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

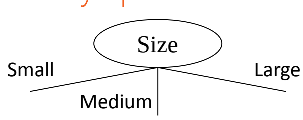


- Binary split: Divides values into two subsets. Need to find optimal partitioning.

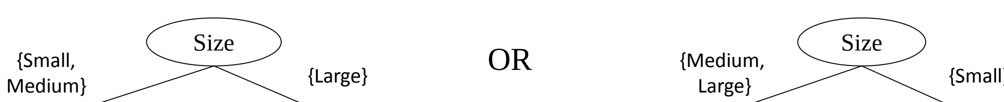


Splitting on Ordinal Attributes

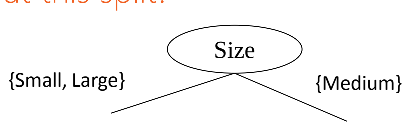
- Multi-way split: Use as many partitions as distinct values.



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



- What about this split?

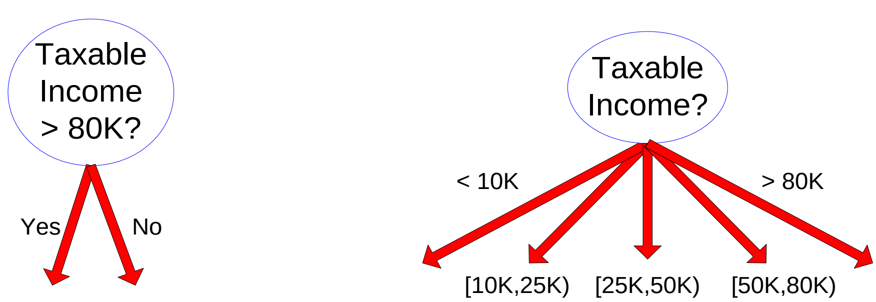


Splitting on Continuous Attributes

- 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 ≥ v)
    - Consider all possible splits and finds the best cut
    - Can be more compute intensive

datasciencedojo
unleash the data scientist in you

Splitting on Continuous Attributes



(i) Binary split (ii) Multi-way split

datasciencedojo
unleash the data scientist in you

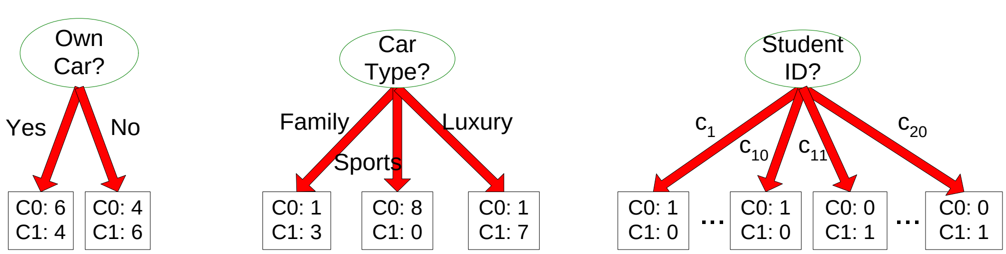
Tree Induction

- 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

datasciencedojo
unleash the data scientist in you

How to determine the Best Split

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



Which test condition is the best?

datasciencedojo
unleash the data scientist in you

How to determine the Best Split

- Greedy approach:
  - Nodes with homogeneous class distribution are preferred
- Need a measure of node impurity:

C0: 5
C1: 5

Non-homogeneous

High degree of impurity

C0: 9
C1: 1

Homogeneous

Low degree of impurity

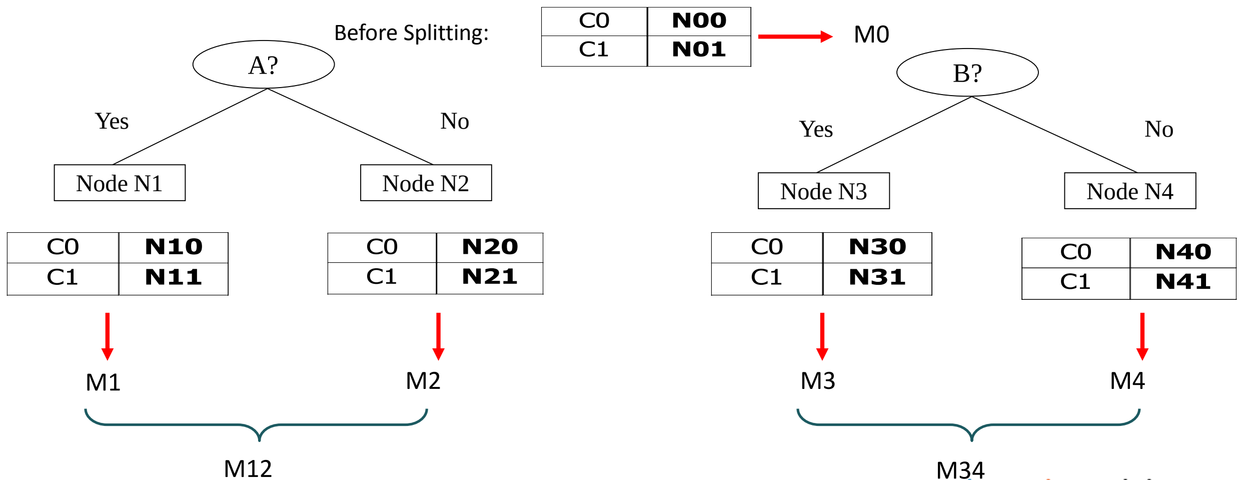
datasciencedojo
unleash the data scientist in you

Measures of Node Impurity

- Gini Index
- Entropy
- Misclassification error

datasciencedojo
unleash the data scientist in you

How to Find the Best Split



Gain = M0 – M12 vs M0 – M34

Measure of Impurity: GINI

- Gini Index for a given node t :

$$\boxed{GINI(t) = 1 - \sum_j [p(j \mid t)]^2}$$

(NOTE: $p(j \mid t)$ is the relative frequency of class j at node t).

- Maximum ($1 - 1/n_c$) when records are equally distributed among all classes, implying least interesting information
- Minimum (0.0) when all records belong to one class, implying most interesting information

C1	**0**
C2	**6**
Gini=0.000	

C1	**1**
C2	**5**
Gini=0.278	

C1	**2**
C2	**4**
Gini=0.444	

C1	**3**
C2	**3**
Gini=0.500	

Examples for computing GINI

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

C1	0
C2	6

P(C1) = 0/6 = 0   P(C2) = 6/6 = 1

Gini = 1 – $P(C1)^2$ – $P(C2)^2$ = 1 – 0 – 1 = 0

C1	1
C2	5

P(C1) = 1/6   P(C2) = 5/6

Gini = 1 – $(1/6)^2$ – $(5/6)^2$ = 0.278

C1	2
C2	4

P(C1) = 2/6   P(C2) = 4/6

Gini = 1 – $(2/6)^2$ – $(4/6)^2$ = 0.444

Splitting Based on GINI

- Used in CART, SLIQ, SPRINT.
- When a node p is split into k partitions (children), the quality of split is computed as:

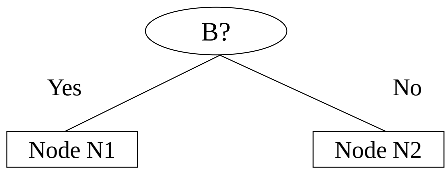
$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

where, n_i = number of records at child i,

n = number of records at node p.

Binary Attributes: Computing GINI Index

- Splits into two partitions
- Effect of Weighing partitions:
  - Larger and Purer Partitions are sought after



	Parent
C1	6
C2	6
Gini = 0.500	

Gini(N1)
$= 1 – (5/6)^2 – (2/6)^2$
$= 0.194$

Gini(N2)
$= 1 – (1/6)^2 – (4/6)^2$
$= 0.528$

	N1	N2
C1	5	1
C2	2	4
Gini=0.333		

Gini(Children)
= 7/12 * 0.194 +
5/12 * 0.528
= 0.333

Alternative Splitting Criteria - Entropy

- Entropy at a given node t:

$$Entropy(t) = -\sum_j p(j \mid t) \log p(j \mid t)$$

(NOTE: $p(j \mid t)$ is the relative frequency of class j at node t).

- Measures homogeneity of a node.
  - Maximum ($\log n_c$) when records are equally distributed among all classes implying least information
  - Minimum (0.0) when all records belong to one class implying most information
- Entropy based computations are similar to the GINI index computations

Examples for computing Entropy

$$Entropy(t) = -\sum_j p(j \mid t) \log_2 p(j \mid t)$$

C1	0
C2	6

P(C1) = 0/6 = 0 P(C2) = 6/6 = 1

Entropy = – 0 log 0 – 1 log 1 = – 0 – 0 = 0

C1	1
C2	5

P(C1) = 1/6 P(C2) = 5/6

Entropy = – (1/6) $\log_2$ (1/6) – (5/6) $\log_2$ (1/6) = 0.65

C1	2
C2	4

P(C1) = 2/6 P(C2) = 4/6

Entropy = – (2/6) $\log_2$ (2/6) – (4/6) $\log_2$ (4/6) = 0.92

datasciencedojo
unleash the data scientist in you

Splitting Based on INFO...

- Information Gain:

$$GAIN_{split} = Entropy(p) - \left(\sum_{i=1}^{k} \frac{n_i}{n} Entropy(i) \right)$$

Parent Node, p is split into k partitions;
n_i is number of records in partition i

- Measures Reduction in Entropy achieved because of the split. Choose the split that achieves most reduction (maximizes GAIN).
- Used in ID3 and C4.5
- Disadvantage: Tends to prefer splits that result in large number of partitions, each being small but pure.

datasciencedojo
unleash the data scientist in you

Splitting Based on INFO...

- Gain Ratio:

$$GainRATIO_{split} = \frac{GAIN_{Split}}{SplitINFO} \qquad SplitINFO = -\sum_{i=1}^{k} \frac{n_i}{n} \log \frac{n_i}{n}$$

Parent Node, p is split into k partitions
n_i is the number of records in partition i

- Adjusts Information Gain by the entropy of the partitioning (SplitINFO). Higher entropy partitioning (large number of small partitions) is penalized!
- Used in C4.5
- Designed to overcome the disadvantage of Information Gain

datasciencedojo
unleash the data scientist in you

Splitting Criteria based on Classification Error

- Classification error at a node t :

$$Error(t) = 1 - \max_i P(i \mid t)$$

- Measures misclassification error made by a node
  - Maximum (1 - $1/n_c$) when records are equally distributed among all classes, implying least interesting information
  - Minimum (0.0) when all records belong to one class, implying most interesting information

datasciencedojo
unleash the data scientist in you

Examples for Computing Error

$$Error(t) = 1 - \max_i P(i \mid t)$$

C1	0
C2	6

P(C1) = 0/6 = 0   P(C2) = 6/6 = 1

Error = 1 – max (0, 1) = 1 – 1 = 0

C1	1
C2	5

P(C1) = 1/6   P(C2) = 5/6

Error = 1 – max (1/6, 5/6) = 1 – 5/6 = 1/6

C1	2
C2	4

P(C1) = 2/6   P(C2) = 4/6

Error = 1 – max (2/6, 4/6) = 1 – 4/6 = 1/3

datasciencedojo
unleash the data scientist in you

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

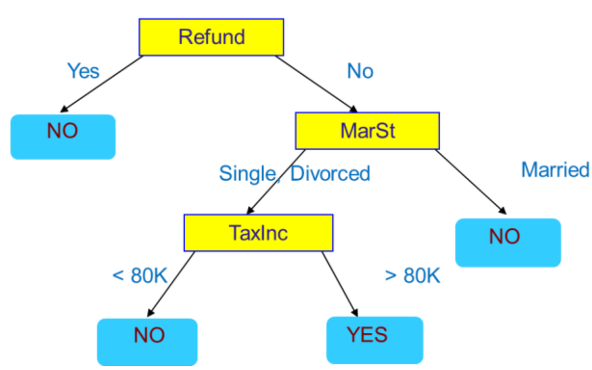
datasciencedojo
unleash the data scientist in you

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

Decision Trees - PROS

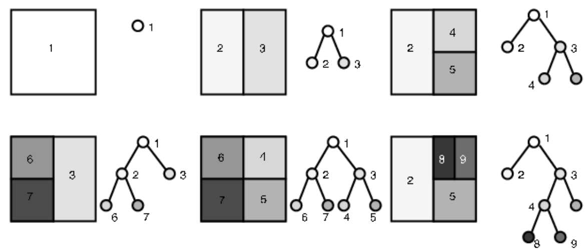
- Intuitive: Easy interpretation for small trees
- Non parametric: Easy to incorporate both numeric and categorical data
- Fast: Once the rules are developed, prediction (classification or regression) is fast
- Robust to outliers: The technique is largely robust to outliers



datasciencedojo
unleash the data scientist in you

Decision Trees - CONS

- Overfitting: Tend to over fit if not trained with care
- Rectangular Classification: One field at a time; recursive partitioning of data
- Tree replication: A tree may be replicated again



datasciencedojo
unleash the data scientist in you

RPART – Kyphosis Data

81 rows and 4 columns

Representing data on children who have had corrective spinal surgery

Kyphosis a factor with levels absent/present indicating if a kyphosis (a type of deformation) was present after the surgery

Age (in months)

Number the number of vertebrae involved

Start the number of the first (topmost) vertebra operated on

Kyphosis	Age	Number	Start
absent	71	3	5
absent	158	3	14
present	128	4	5
absent	2	5	1
absent	1	4	15
absent	1	2	16
absent	61	2	17
absent	37	3	16

datasciencedojo
unleash the data scientist in you

RPART ON IRIS DATA – Test TRAIN SPLIT

- sub <- c(sample(1:50, 25), sample(51:100, 25), sample(101:150, 25))

datasciencedojo
unleash the data scientist in you

RPART ON IRIS DATA – training the model

- fit <- rpart(Species ~ ., data = iris, subset = sub)

datasciencedojo
unleash the data scientist in you

RPART ON IRIS DATA – PREDICTING

- predict(fit, iris[-sub,])
- predict(fit, iris[-sub,], type = "class")

RPART ON IRIS DATA – Confusion Matrix

- table(predict(fit, iris[-sub,], type = "class"), iris[-sub, "Species"])

	setosa	versicolor	virginica
setosa	25	0	0
versicolor	0	24	4
virginica	0	1	21

SPLITTING INTO TRAIN AND TEST RANDOMLY

- splitdf <- function(dataframe, seed=NULL)
- {
- if (!is.null(seed)) set.seed(seed)
- index <- 1:nrow(dataframe)
- trainindex <- sample(index, trunc(length(index)/2))
- trainset <- dataframe[trainindex,]
- testset <- dataframe[-trainindex,]
- list(trainset=trainset,testset=testset)
- }

RPART

```
fit <- rpart(Kyphosis ~ Age + Number + Start, data = kyphosis)
fit2 <- rpart(Kyphosis ~ Age + Number + Start, data = kyphosis,
parms = list(prior = c(0.65, 0.35), split = "information"))
fit3 <- rpart(Kyphosis ~ Age + Number + Start, data=kyphosis,
control = rpart.control(cp = 0.05))
par(mfrow = c(1,2), xpd = TRUE)
plot(fit)
text(fit, use.n = TRUE)
plot(fit2)
text(fit2, use.n = TRUE)
```

Rpart package

http://www.statmethods.net/advstats/cart.html

- >rpartFormula = paste("V15~",paste(paste("V",1:14, sep=""),collapse="+"),sep="")
- >str(adult)
- > model = rpart(rpartFormula,data=adult,method="class")
- > str(model)
- > plot(model)
- > text(model)
- Prettier plots with Rpart
- http://tagteam.harvard.edu/hub_feeds/1981/feed_items/207424