K2 Analytics
Building Skills, Building Individuals

Classification Tree

- Rajesh Jakhotia

21-Mar-2017

Earning is in Learning
- Rajesh Jakhotia

About K2 Analytics

At K2 Analytics, we believe that skill development is very important for the growth of an individual, which in turn leads to the growth of Society & Industry and ultimately the Nation as a whole. For this it is important that access to knowledge and skill development trainings should be made available easily and economically to every individual.

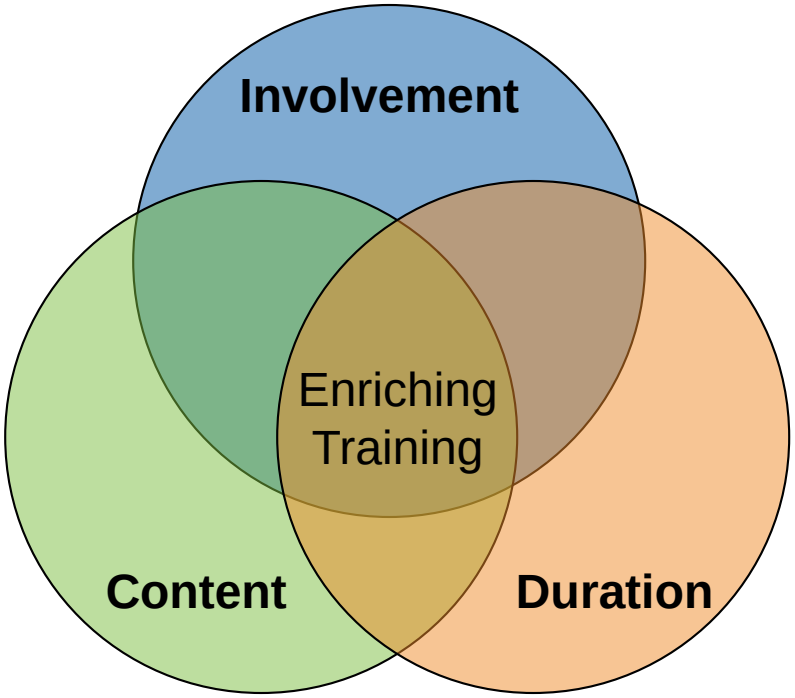
Our Vision: *"To be the preferred partner for training and skill development"*

Our Mission: *"To provide training and skill development training to individuals, make them skilled & industry ready and create a pool of skilled resources readily available for the industry"*

*We have chosen Business Intelligence and Analytics as our focus area. With this endeavour we make this presentation on "**Classification Tree - CART**" accessible to all those who wish to learn Analytics. We hope it is of help to you. For any feedback / suggestion or if you are looking for job in analytics then feel free to write back to us at ar.jakhotia@k2analytics.co.in*

Welcome to CART!!!

Enriching training and learning session…



- **Training Checklist**
  - Sitting arrangement F2F
  - Quality over Quantity
  - Everyone to have their own machines for hands-on practice
  - Illuminated and happy glowing training room (no candle light dinner ambience)
  - Anyone wanting to step-out, feel free
  - Feel free to ask for breaks
  - Feel free to ask same question again till you understand
  - Let me know if you want me to skip Practice Exercises in between the session
  - Brief side-talks are okay
  - **I don't speak to walls, respect each other**

K2 Analytics

Building Skills, Building Individuals

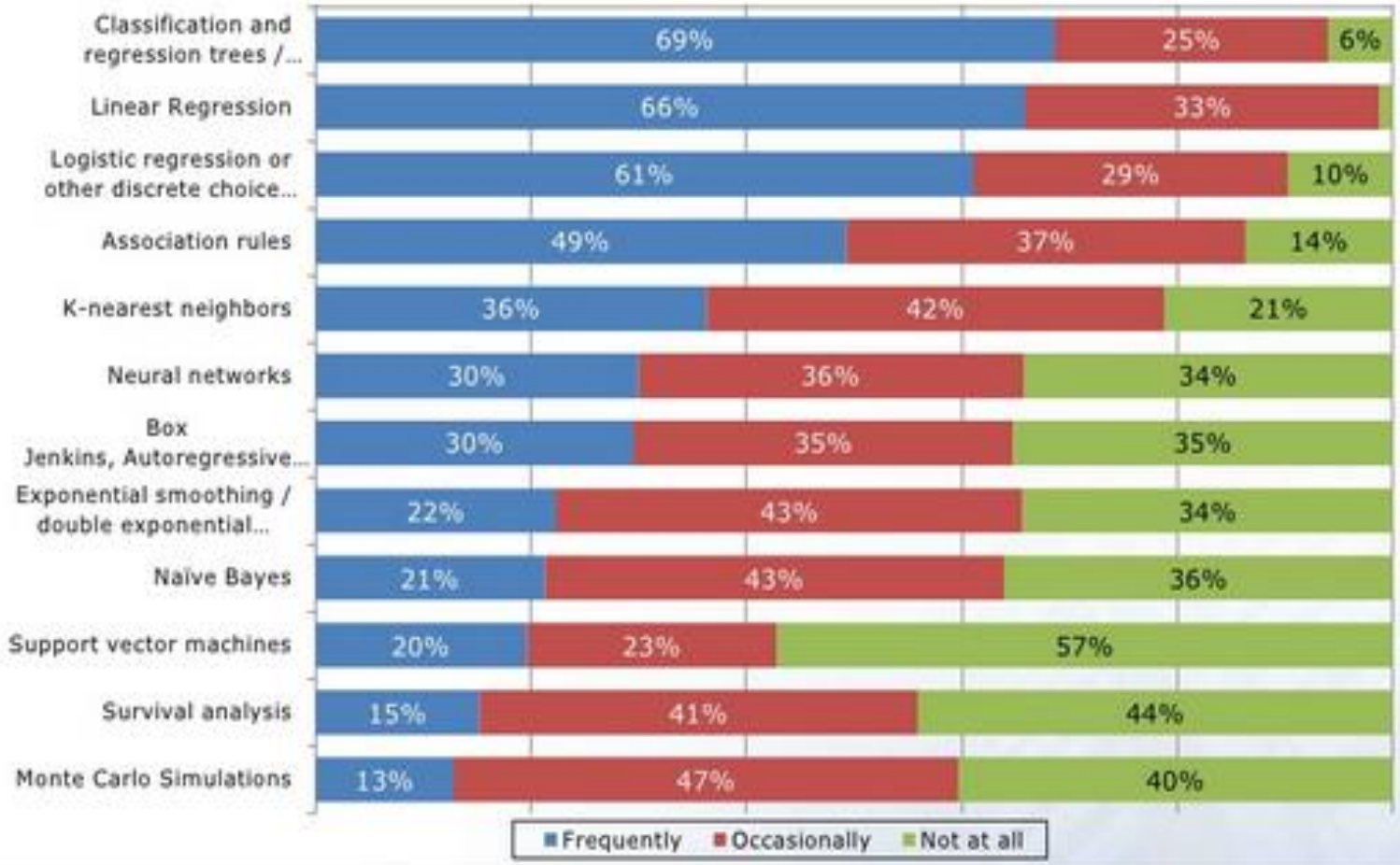
Classification Tree

CART

Learning Objectives

- What is Classification Technique?
- CHAID, CART, C4.5 Intro
- Gini Gain Computation
- Why are Classification Tree algorithms Recursive?
- What is pre-pruning and post-pruning in Classification Tree?
- What is Loss?
- What is Validation? What is Cross-Validation?
- Why you should avoid over-fitting?
- Rank Ordering Model Performance Measure

Analytics that are Actually Used

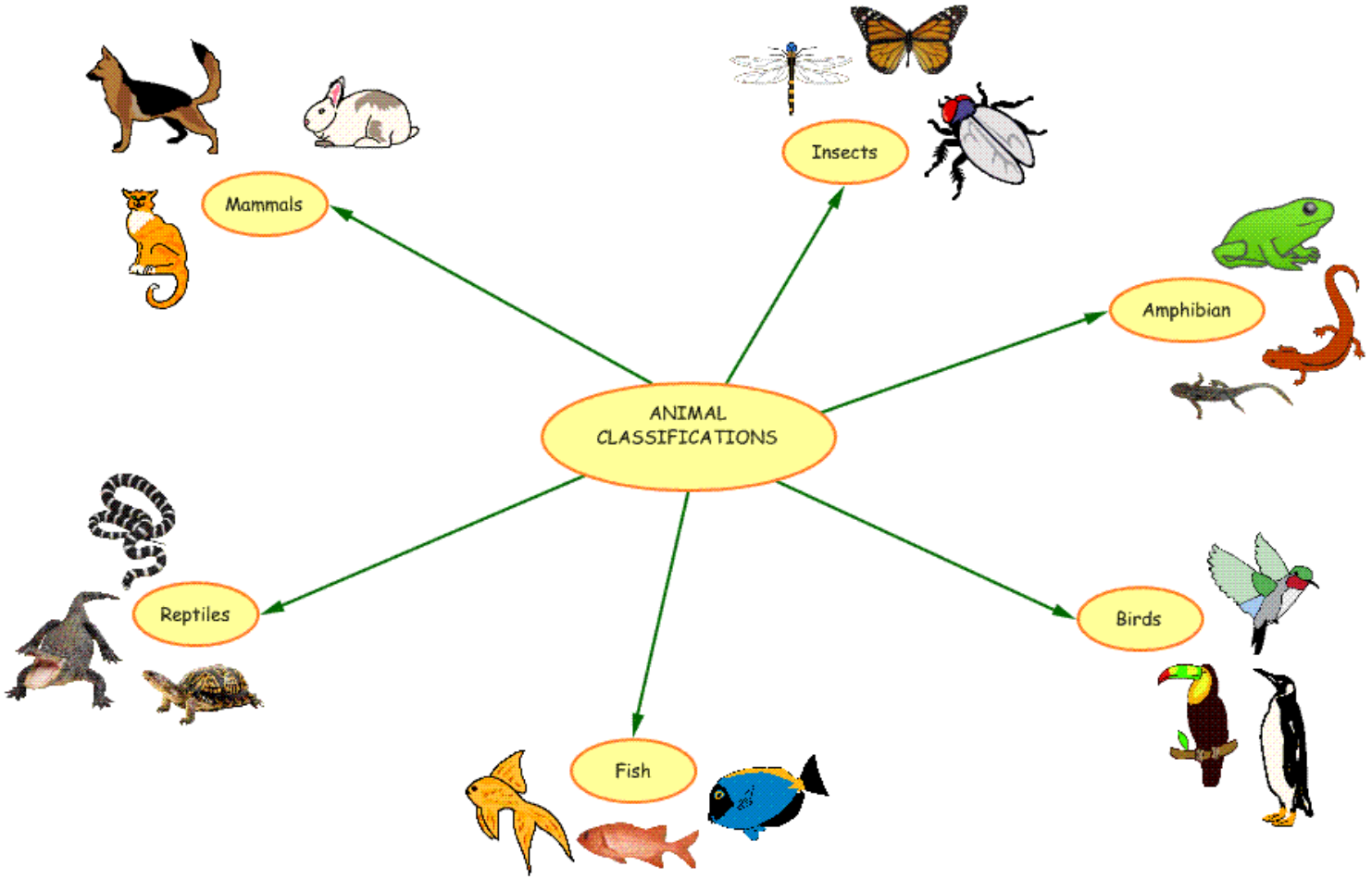


Classification and regression trees / decision trees and Linear Regression are the most popular predictive analytics techniques used.

Source: Ventana Research Predictive Analytics Benchmark Research

What is Classification?

The action or process of classifying something according to shared qualities or characteristics.



Defining Characteristics of each animal classification

- Mammals – Mammals are vertebrates (backboned animals). Mammals are warm-blooded and have hair. Mammals are able to move around using limbs
- Birds – Birds are warm-blooded vertebrates, having a body covered with feathers, forelimbs modified into wings, scaly legs, a beak, and no teeth, and bearing young ones in a hard-shelled egg
- Insects – any of small invertebrate animals which typically have a well defined head, thorax, and abdomen, only three pairs of legs, and typically one or two pair of wings
- Amphibian - any cold-blooded vertebrate that live on land but breed in water
- Reptiles - class of cold-blooded air-breathing vertebrates with completely ossified skeleton and a body usually covered with scales or horny plates
- Fish - A limbless cold-blooded vertebrate animal with gills and fins and living wholly in water

Why Classify?

To Explain (Profile)

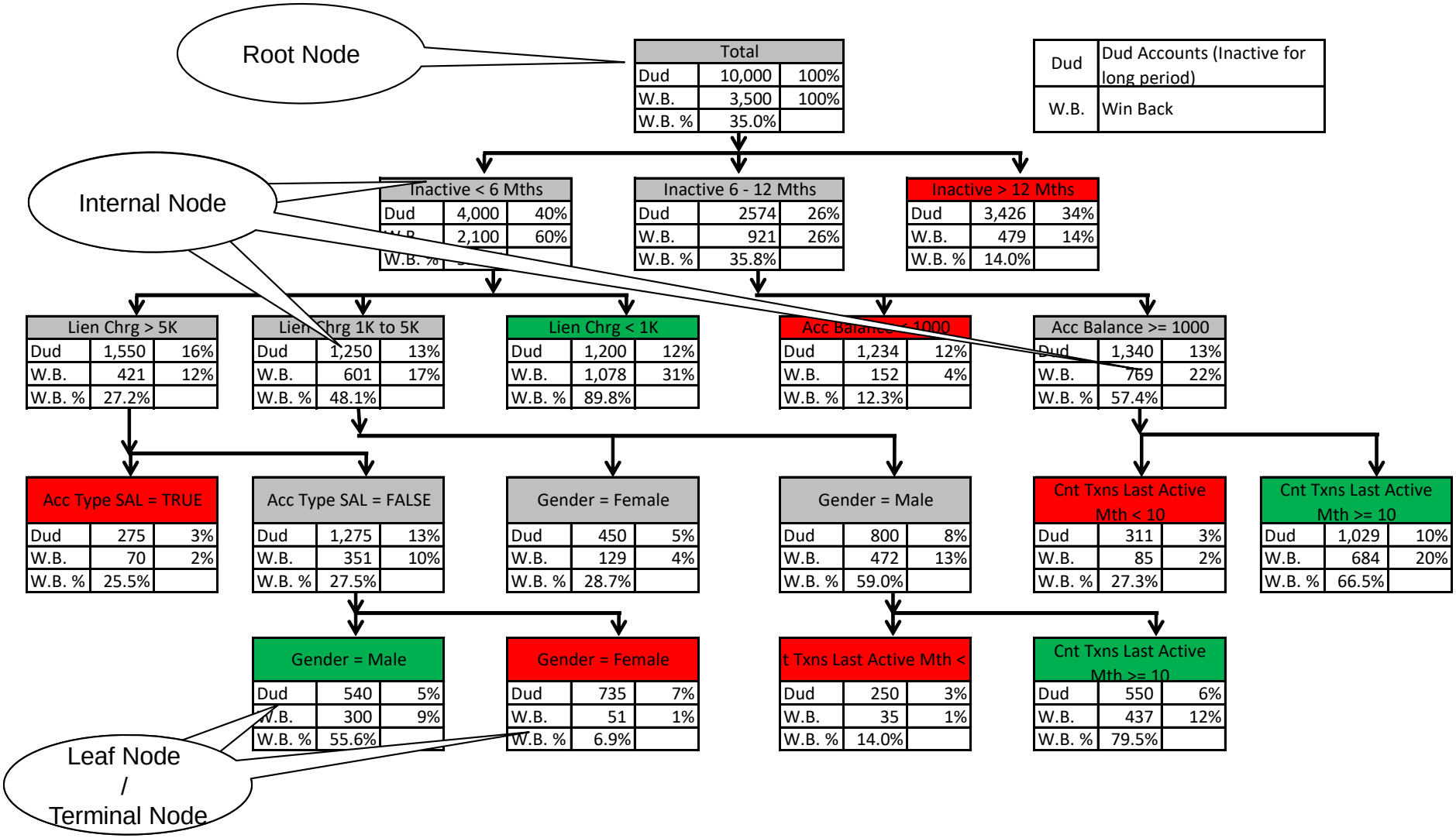
Explaining in the classification world is called Profiling

or

To Predict (Classify)

Predicting the class of new records is called Classifying

Win Back Campaign Classification Analysis



Main issues of classification tree learning

- Choosing the splitting criterion
  - Impurity based criteria
  - Information gain
  - Statistical measures of association
- Binary or multiway splits
  - Multiway split
  - Binary split
- Finding the right sized tree
  - Pre-pruning
  - Post-pruning

Popular Classification Techniques

- **CHAID - CHi-squared Automatic Interaction Detector**. The "*Chi-squared" part of the name arises because the technique* essentially involves automatically constructing many cross-tabs, and working out statistical significance of the proportions. The most significant relationships are used to control the structure of a tree diagram
  - CHAID is a non-binary decision tree; **Recursive Partitioning Algorithm**
  - Continuous variables must be grouped into a finite number of bins to create categories.
- **CLASSIFICATION AND REGRESSION TREES (CART)** are binary decision trees, which split a single variable at each node.
  - The CART algorithm recursively goes though an exhaustive search of all variables and split values to find the optimal splitting rule for each node.
- **C4.5** builds decision trees from a set of training data using the concept of information entropy

K2 Analytics

Building Skills, Building Individuals

CART

CART | Splitting Criteria

- CART uses the Gini Index as measure of impurity
- Gini of a Node

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

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

- Gini of Split Node is computed as Weighted Avg Gini of each Node at Split Node level

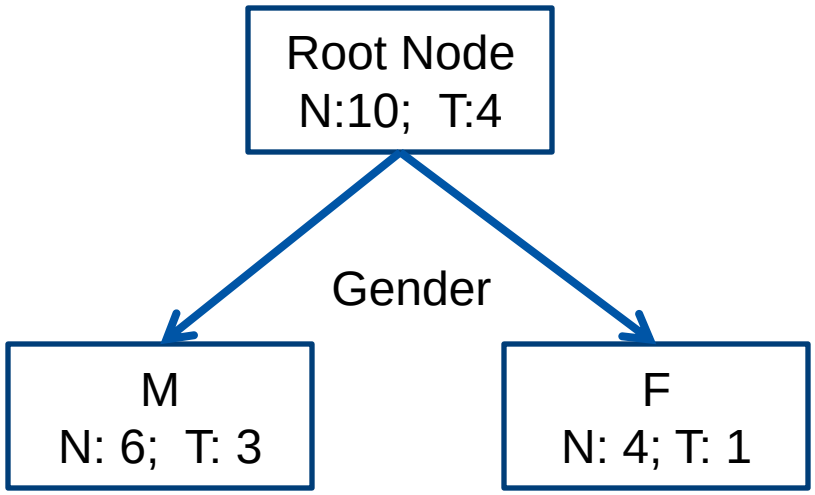
$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

n_i = number of records at child i,
n = Total number of records in parent node

- Gini Gain = Gini(t) – Gini(split)

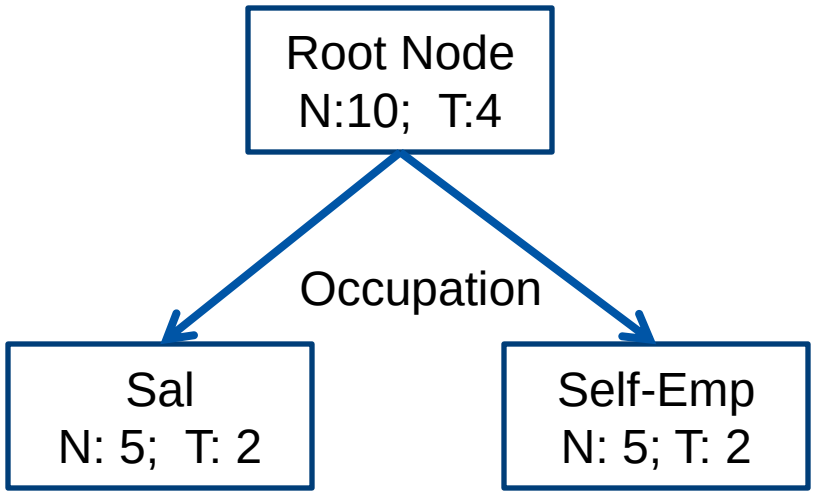
Gini calculations

Cust_ID	Gender	Occupation	Age	Target
1	M	Sal	22	1
2	M	Sal	22	0
3	M	Self-Emp	23	1
4	M	Self-Emp	23	0
5	M	Self-Emp	24	1
6	M	Self-Emp	24	0
7	F	Sal	25	1
8	F	Sal	25	0
9	F	Sal	26	0
10	F	Self-Emp	26	0



Node	Gini Computation Formula	Gini Index
Overall	= 1 - ((4/10)^2 + (6/10)^2)	0.48
Gender = M	= 1 - ((3/6)^2 + (3/6)^2)	0.50
Gender = F	= 1 - ((1/4)^2 + (3/4)^2)	0.375
Gender	= (6/10) * 0.5 + (4/10) * 0.375	0.45
Gini Gain	= Gini (Overall) – Gini (Gender)	0.03

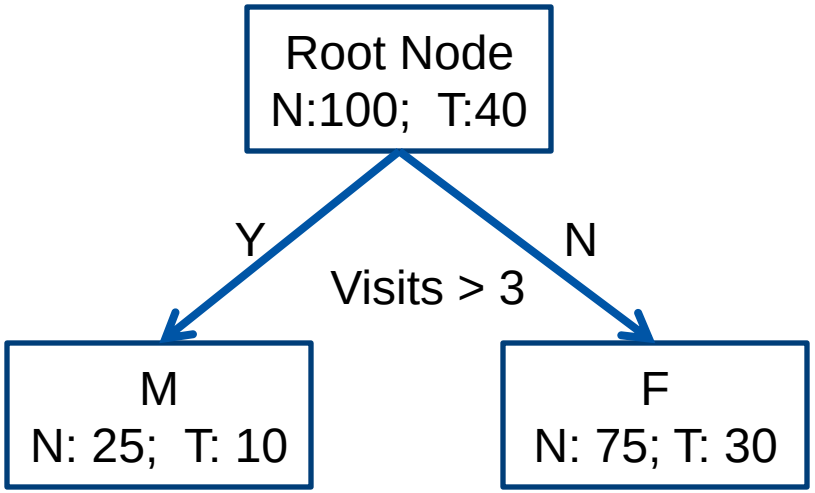
Gini calculations



Node	Gini Computation Formula	Gini Index
Overall	= 1 - ((4/10)^2 + (6/10)^2)	0.48
Occ = Sal	= 1 - ((2/5)^2 + (3/5)^2)	0.48
Occ = Self-Emp	= 1 - ((2/5)^2 + (3/5)^2)	0.48
Occupation	= (5/10) * 0.48 + (5/10) * 0.48	0.48
Gini Gain	= Gini (Overall) – Gini (Occupation)	0.0

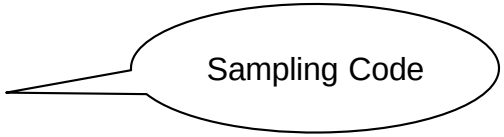
Age	<=22	<=23	<=24	<=25
Gini (Left)	0.5	0.5	0.5	0.5
Gini (Right)	0.47	0.44	0.38	0
Gini Split	0.48	0.47	0.45	0.40
Gini Gain	0.0	0.01	0.03	0.08

Exercise… Compute Gini Gain



Sampling…

```
## Creating Development and Validation Sample
## CTDF$random <- runif (nrow(CTDF), 0, 1);
## CTDF.dev <- CTDF [which(CTDF$random <= 0.7),]
## CTDF.holdout <- CTDF [which(CTDF$random > 0.7),]
## c (nrow(CTDF.dev), nrow(CTDF.holdout))
```



CTDF.dev <- read.table ("datafile/DEV_SAMPLE.csv", sep = ",", header = T)

CTDF.holdout <- read.table ("datafile/HOLDOUT_SAMPLE.csv", sep = ",", header = T)

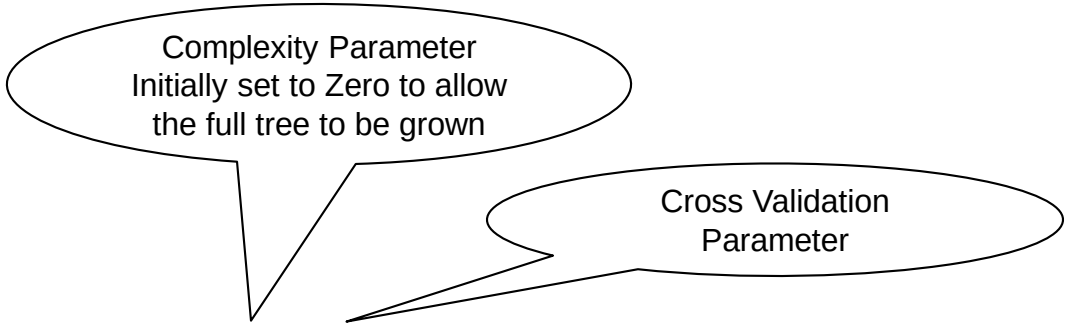
rpart code to build CART Tree

```
## installing rpart package for CART
## install.packages("rpart")
## install.packages("rpart.plot")

## loading the library
library(rpart)
library(rpart.plot)

## setting the control parameter inputs for rpart
r.ctrl = rpart.control(minsplit=100, minbucket = 10, cp = 0, xval = 10)

## calling the rpart function to build the tree
m1 <- rpart(formula = Target ~ ., data = CTDF.dev[,-1], method = "class", control = r.ctrl)
m1
```



rpart.control arguments

- **minsplit:** the minimum number of observations that must exist in a node in order for a split to be attempted.
- **minbucket**: the minimum number of observations in any terminal leaf node. If only one of minbucket or minsplit is specified, the code either sets minsplit to minbucket*3 or minbucket to minsplit/3, as appropriate.
- **cp complexity parameter**: Any split that does not decrease the overall lack of fit by a factor of cp is not attempted. The main role of this parameter is to save computing time by pruning off splits that are obviously not worthwhile. Essentially,the user informs the program that any split which does not improve the fit by cp will likely be pruned off by cross-validation, and that hence the program need not pursue it.
- **xval**: number of cross-validations

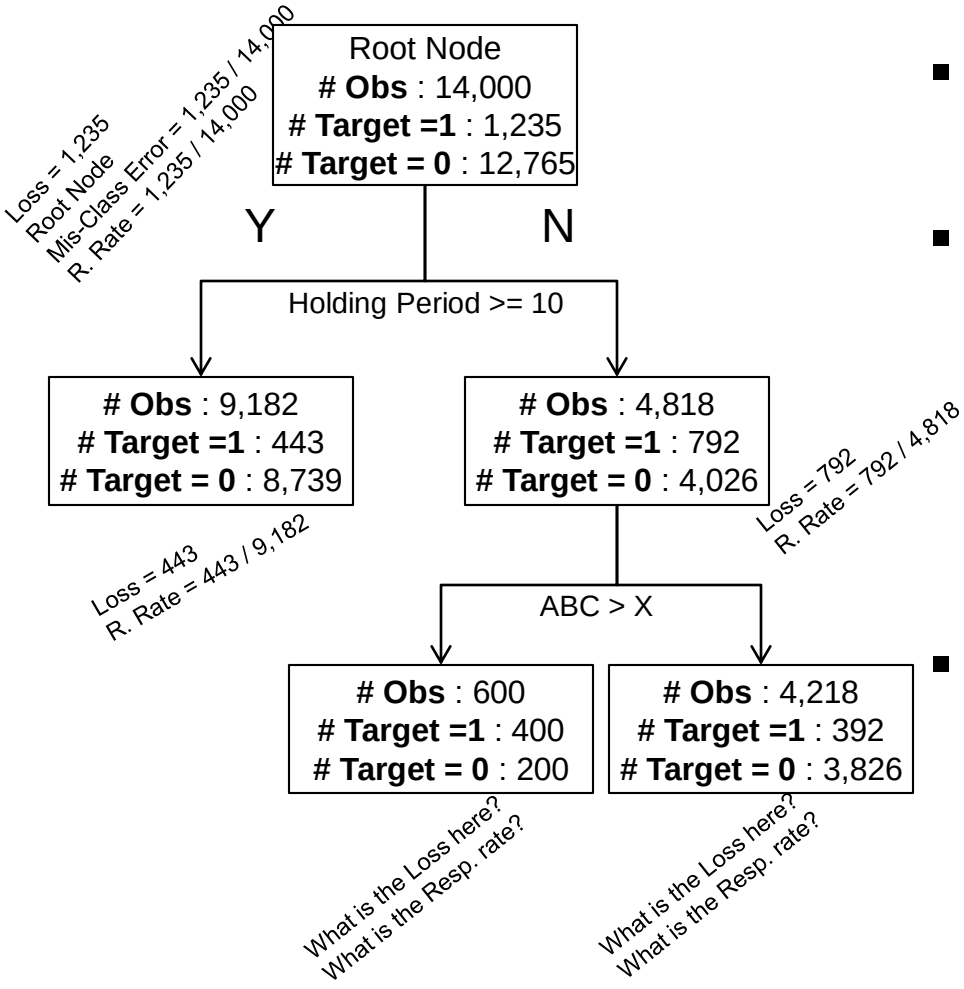
rpart output

```
n= 14000

node), split, n, loss, yval, (yprob)
      * denotes terminal node

    1) root 14000 1235 0 (0.91178571 0.08821429)
      2) Holding_Period>=10.5 9182  443 0 (0.95175343 0.04824657)
        4) No_OF_CR_TXNS< 20.5 6858  212 0 (0.96908720 0.03091280) *
        5) No_OF_CR_TXNS>=20.5 2324  231 0 (0.90060241 0.09939759)
         10) Occupation=PROF,SAL 1814  124 0 (0.93164278 0.06835722) *
         11) Occupation=SELF-EMP,SENP 510  107 0 (0.79019608 0.20980392)
           22) SCR< 334.5 120    9 0 (0.92500000 0.07500000) *
           23) SCR>=334.5 390   98 0 (0.74871795 0.25128205)
             46) Gender=M,O 370   85 0 (0.77027027 0.22972973) *
             47) Gender=F 20    7 1 (0.35000000 0.65000000) *
      3) Holding_Period< 10.5 4818  792 0 (0.83561644 0.16438356)
        6) Occupation=PROF,SAL,SENP 3971  546 0 (0.86250315 0.13749685)
         12) No_OF_CR_TXNS< 20.5 2832  317 0 (0.88806497 0.11193503)
           24) Balance>=10853.5 2551  259 0 (0.89847119 0.10152881)
             48) SCR< 697.5 1618  129 0 (0.92027194 0.07972806) *
             49) SCR>=697.5 933  130 0 (0.86066452 0.13933548)
               98) Holding_Period>=1.5 791   98 0 (0.87610619 0.12389381)
                196) SCR>=732.5 712   81 0 (0.88623596 0.11376404)
                  392) AGE_BKT=>50,26-30,31-35 358   30 0 (0.91620112 0.08379888) *
                  393) AGE_BKT=<25,36-40,41-45,46-50 354   51 0 (0.85593220 0.14406780)
                    786) Balance>=34458.97 278   33 0 (0.88129496 0.11870504)
                     1572) Balance< 48554.42 28    0 0 (1.00000000 0.00000000) *
                     1573) Balance>=48554.42 250   33 0 (0.86800000 0.13200000)
                       3146) Balance>=212700.4 103    8 0 (0.92233010 0.07766990) *
                       3147) Balance< 212700.4 147   25 0 (0.82993197 0.17006803)
                         6294) Holding_Period< 2.5 23    0 0 (1.00000000 0.00000000) *
                         6295) Holding_Period>=2.5 124   25 0 (0.79838710 0.20161290)
                          12590) Holding_Period>=3.5 108   16 0 (0.85185185 0.14814815) *
                          12591) Holding_Period< 3.5 16    7 1 (0.43750000 0.56250000) *
                    787) Balance< 34458.97 76   18 0 (0.76315789 0.23684211) *
                197) SCR< 732.5 79   17 0 (0.78481013 0.21518987) *
               99) Holding_Period< 1.5 142   32 0 (0.77464789 0.22535211) *
           25) Balance< 10853.5 281   58 0 (0.79359431 0.20640569) *
         13) No_OF_CR_TXNS>=20.5 1139  229 0 (0.79894644 0.20105356)
           26) Occupation=PROF,SAL 1048  174 0 (0.83396947 0.16603053) *
           27) Occupation=SENP 91   36 1 (0.39560440 0.60439560) *
        7) Occupation=SELF-EMP 847  246 0 (0.70956316 0.29043684)
         14) SCR< 725 538  107 0 (0.80111524 0.19888476)
           28) No_OF_CR_TXNS< 29.5 434   68 0 (0.84331797 0.15668203) *
           29) No_OF_CR_TXNS>=29.5 104   39 0 (0.62500000 0.37500000)
             58) Balance>=4171.13 93   31 0 (0.66666667 0.33333333) *
             59) Balance< 4171.13 11    3 1 (0.27272727 0.72727273) *
         15) SCR>=725 309  139 0 (0.55016181 0.44983819)
           30) Balance>=13166.99 205   72 0 (0.64878049 0.35121951)
             60) No_OF_CR_TXNS< 20.5 149   39 0 (0.73825503 0.26174497) *
             61) No_OF_CR_TXNS>=20.5 56   23 1 (0.41071429 0.58928571) *
           31) Balance< 13166.99 104   37 1 (0.35576923 0.64423077)
             62) Age>=48 17    6 0 (0.64705882 0.35294118) *
             63) Age< 48 87   26 1 (0.29885057 0.70114943) *
```

Loss, Mis-Classification Error and Response Rate

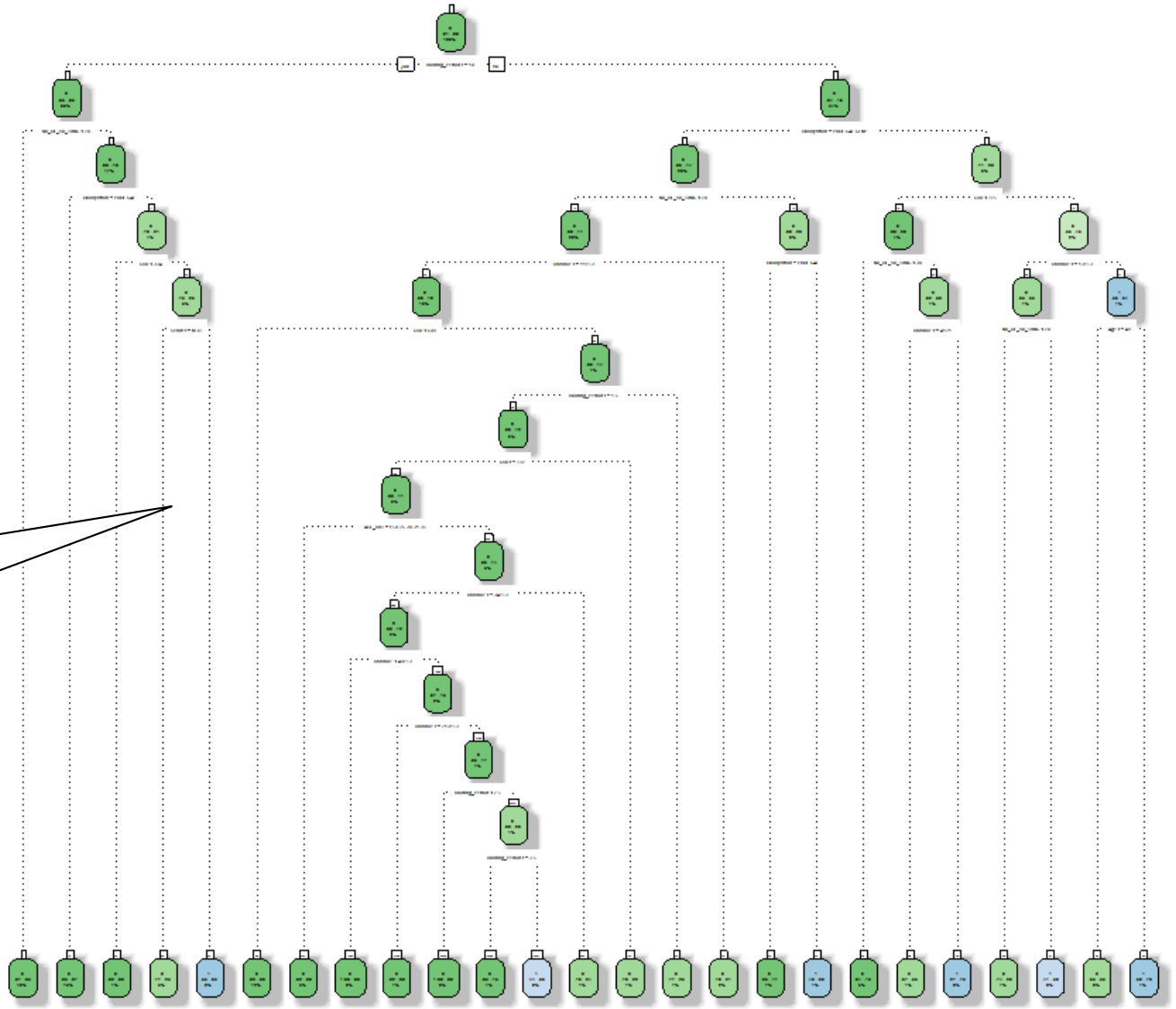


- Loss is the number of cases mis-classified in a given node
- Mis-Classification Error is the ratio of total number of cases mis-classified to total number of cases
  - We are interested in mis-classification error for the full tree
- Response Rate is the ratio of number of responders (Target = 1) to the total number of cases
  - We are interested in finding nodes where the response rate is very high

What is the mis-classification error for the above tree?

Plotting the Classification Tree

```
library(rattle)
library(RColorBrewer
fancyRpartPlot(m1)
```



Let us export the output to PDF format to have a clear view of the tree

Concepts | Greedy Algorithm

Make 31 Paise using any combination of above coins

Optimal solution with few coins : 25 + 5 + 1

What if the 5 paise coin is not there?

Optimal solution with few coins : 10 * 3 + 1

Greedy Algorithm solution: 25 + 1 * 6

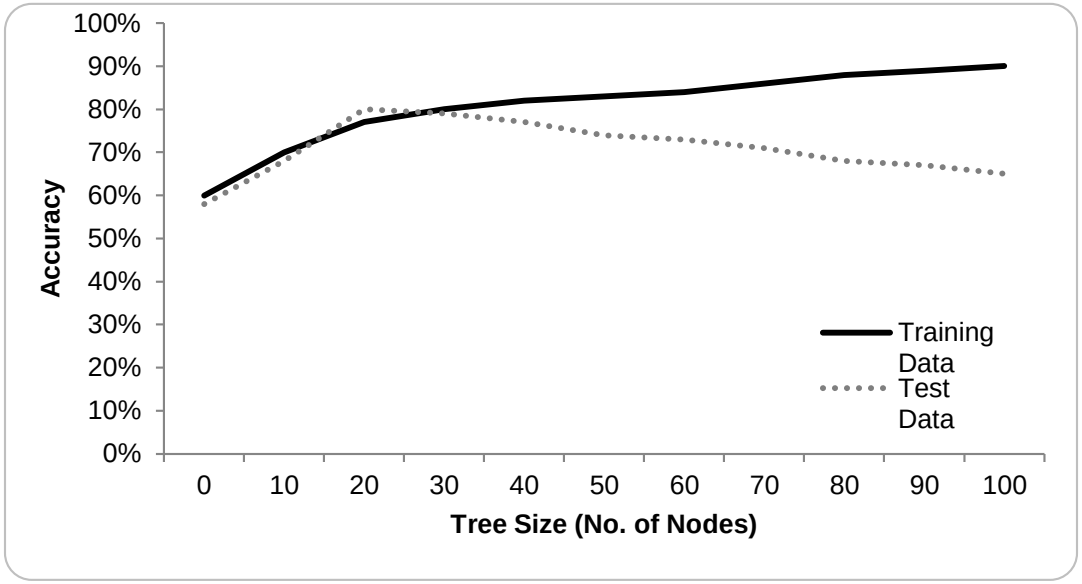
Concepts | Cross Validation

K Fold CV	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
Fold 1	Train	Train	Train	Train	Train	Train	Train	Train	Train	Test
Fold 2	Train	Train	Train	Train	Train	Train	Train	Train	Test	Train
Fold 3	Train	Train	Train	Train	Train	Train	Train	Test	Train	Train
Fold 4	Train	Train	Train	Train	Train	Train	Test	Train	Train	Train
Fold 5	Train	Train	Train	Train	Train	Test	Train	Train	Train	Train
Fold 6	Train	Train	Train	Train	Test	Train	Train	Train	Train	Train
Fold 7	Train	Train	Train	Test	Train	Train	Train	Train	Train	Train
Fold 8	Train	Train	Test	Train	Train	Train	Train	Train	Train	Train
Fold 9	Train	Test	Train	Train	Train	Train	Train	Train	Train	Train
Fold 10	Test	Train	Train	Train	Train	Train	Train	Train	Train	Train

- Cross Validation is part of the CART algorithm
- Method to see how well the model performs to unseen data
- Typically xval parameter for cross-validation is set to 10

Concepts | Over-fitting

- If you grow the tree too long you will run the risk of over-fitting
- Classification model may not work well on unseen data



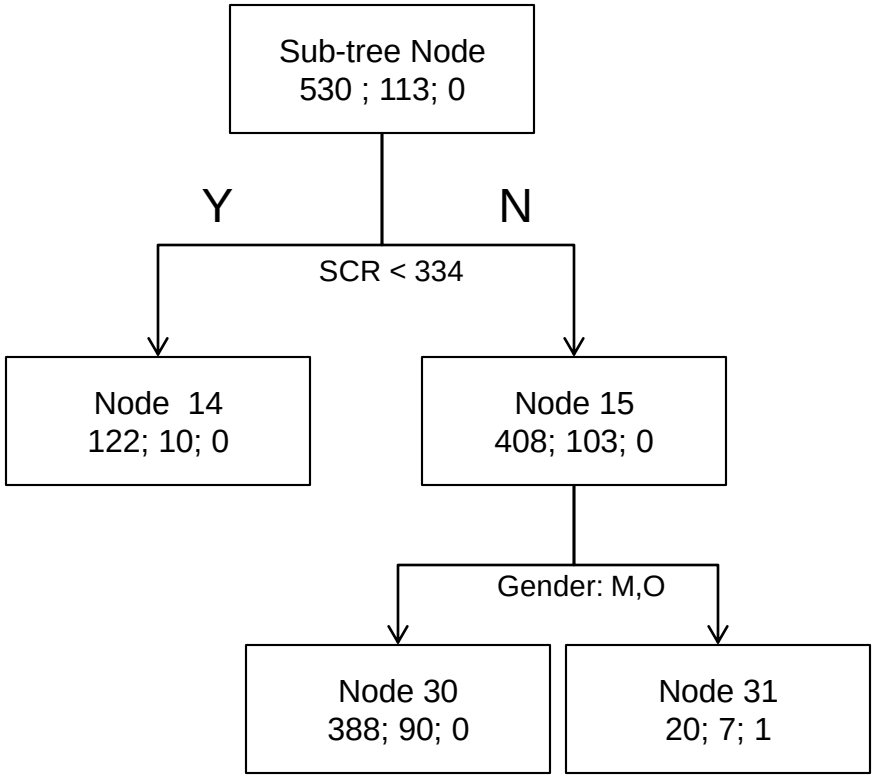
How do we avoid Over-fitting?

Stopping Rule: don't expand a node if the impurity reduction of the best split is below some threshold

Pruning: grow a very large tree and merge back nodes

Concepts | Parsimony Principle & Re-substitution Error

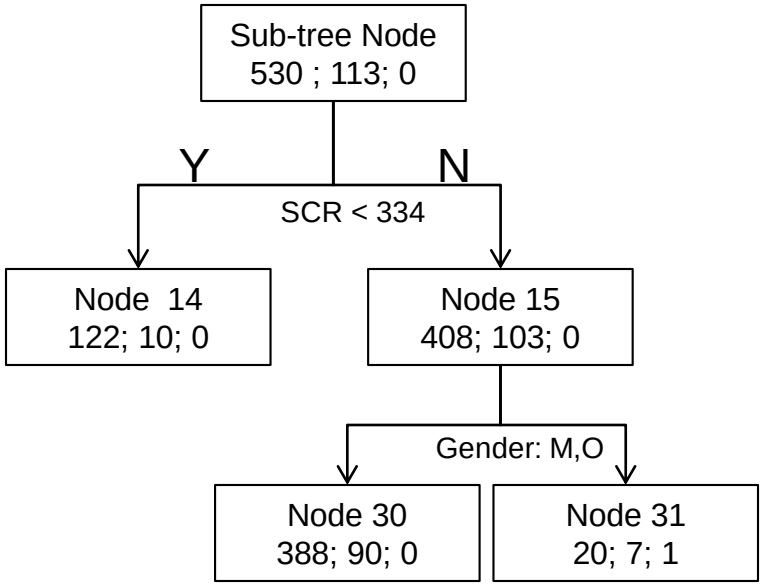
- **Parsimony principle** is basic to all science and tells us to choose the simplest scientific explanation that fits the evidence.
- **Resubstitution Error**: It measures what fraction of the cases in a node is classified incorrectly if we assign every case to the majority class in that node; It always favours large tree
- To counter balance the resubstitution error we need a penalty component that favours smaller tree



Re (prunded) = 113 / 530
Re (leaves) = 107 / 530

Cost Component Pruning

- "cost-complexity" – a measure of avg. error reduced per leaf
- Calculate number of errors for each node if collapsed to leaf
- Compare to errors in leaves, taking into account more nodes used



Re (prunded) + 1 α
= Re (leaves) + 3 α

$113 / 530 + 1\ \alpha = 107 / 530 + 3\ \alpha$

$\alpha = 0.0056$

Print Optimal Pruning table based on Complexity Parameter

to find how the tree performs

printcp(m1)

plotcp(m1)

```
> printcp(m1)

Classification tree:
rpart(formula = Target ~ ., data = CT_DF.dev[, -1], method = "class",
    control = r.ctrl)

Variables actually used in tree construction:
[1] Age            AGE_BKT        Balance        Gender         Holding_Period No_OF_CR_TXNS
[7] Occupation     SCR

Root node error: 1235/14000 = 0.088214

n= 14000

          CP nsplit rel error  xerror     xstd
1 0.00607287      0   1.00000 1.00000 0.027171
2 0.00404858      7   0.95223 0.98138 0.026941
3 0.00202429      8   0.94818 0.98057 0.026931
4 0.00121457     10   0.94413 0.97895 0.026911
5 0.00016194     14   0.93927 0.99514 0.027112
6 0.00000000     24   0.93765 0.99676 0.027132
```

Pruning criteria based on cp table

	CP	nsplit	rel error	xerror	xstd
1	0.00607287	0	1.00000	1.00000	0.027171
2	0.00404858	7	0.95223	0.98138	0.026941
3	0.00202429	8	0.94818	0.98057	0.026931
4	0.00121457	10	0.94413	0.97895	0.026911
5	0.00016194	14	0.93927	0.99514	0.027112
6	0.00000000	24	0.93765	0.99676	0.027132

- xerror – Prune the tree at cp where xerror is minimum
- 1 SE Rule : Look form minimum xerror and then
- … or probably use business rule to decide the number of nodes

Pruning Code

```
ptree<- prune(m1, cp= 0.0015,"CP")

printcp(ptree)

fancyRpartPlot(ptree, uniform=TRUE,  main="Pruned Classification Tree")
```

1 SE rule example

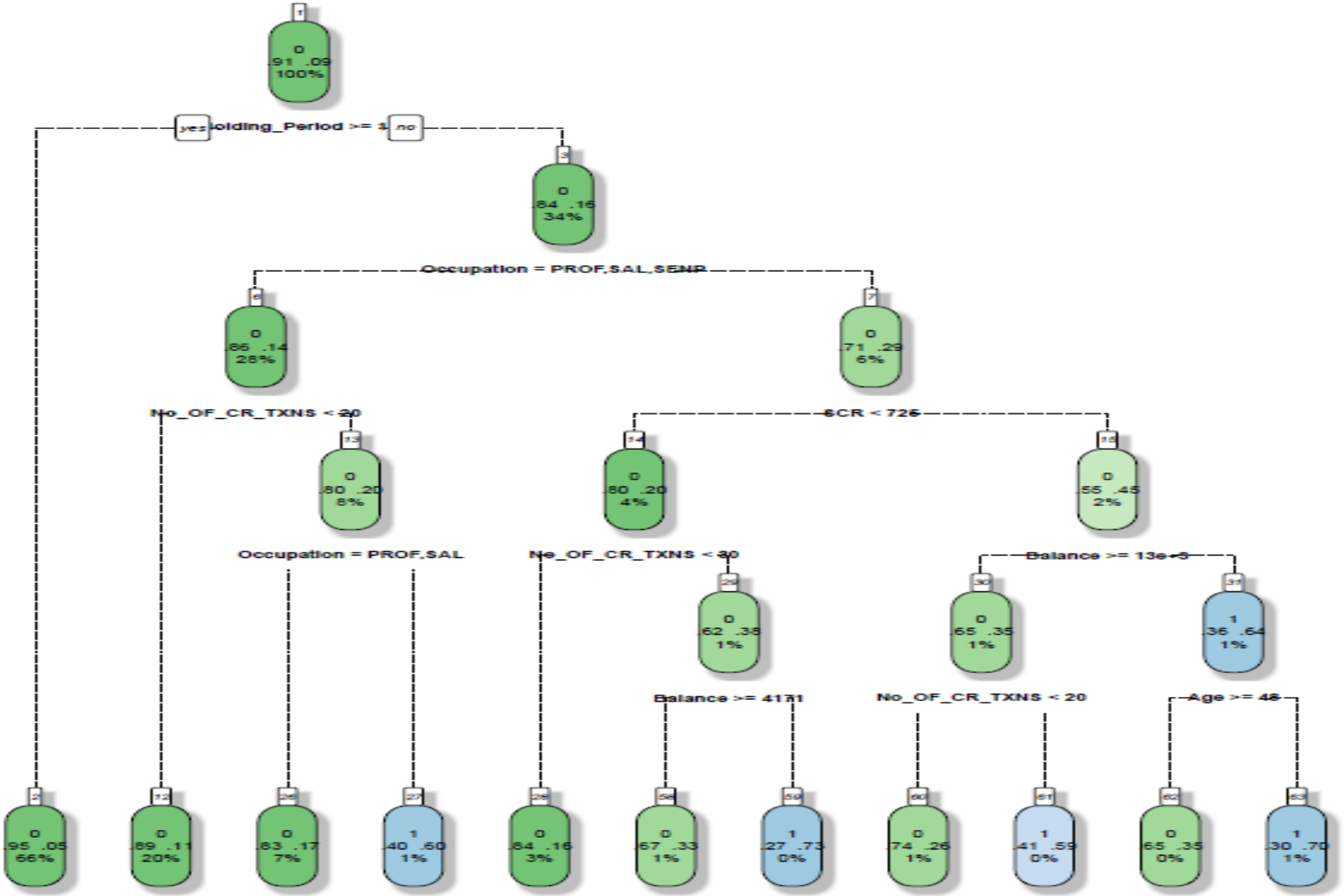
	CP	nsplit	rel error	xerror	xstd
1	0.161992664	0	1.0000000	1.0002790	0.01853630
2	0.043985638	1	0.8380073	0.8385070	0.01749290
3	0.030278222	2	0.7940217	0.7963870	0.01709283
4	0.013881619	3	0.7637435	0.7695997	0.01653832
5	0.010181164	4	0.7498619	0.7560406	0.01606136
6	0.008004043	5	0.7396807	0.7466449	0.01600352
7	0.007026176	6	0.7316767	0.7356289	0.01549501
8	0.006614587	8	0.7176243	0.7388091	0.01559568
9	0.005312278	10	0.7043951	0.7254237	0.01522645
10	0.004883811	11	0.6990828	0.7248227	0.01526605

e.g. taken for explanation purpose only

- Based on xerror criteria we will stop at row 10
- Using column xstd, that would suggest using 0.7248227 + 1*0.01526605 = 0.7400887 and thus pruning should occur at row 7

http://stats.stackexchange.com/questions/92547/r-rpart-cross-validation-and-1-se-rule-why-is-the-column-in-cptable-called-xst
https://stats.stackexchange.com/questions/13471/how-to-choose-the-number-of-splits-in-rpart

Pruned Classification Tree

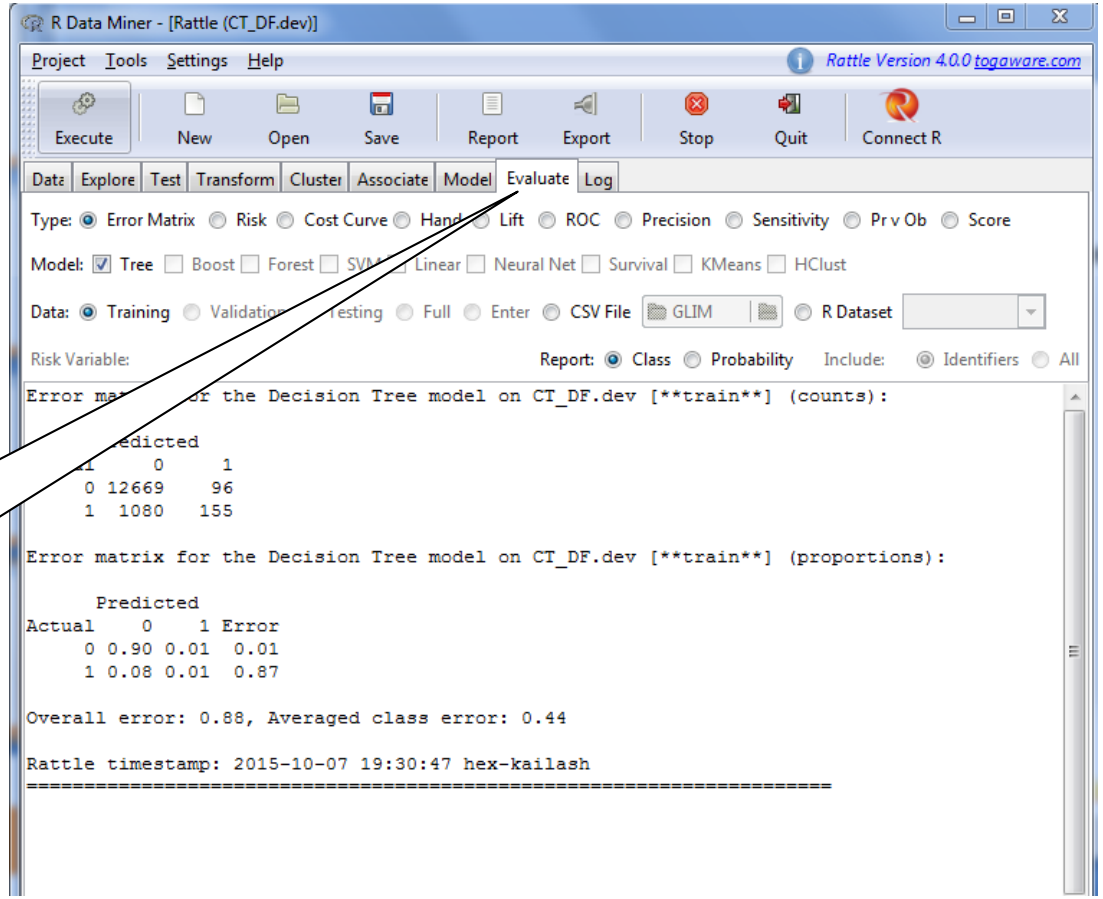


Model Evaluation

Various measures to see the model performance

- Error Matrix
- Gini Coefficient
- AUC
- KS
- Lift Chart

Demo of Rattle interface to build model and generate various model evaluation measures



https://www.youtube.com/watch?v=OAl6eAyP-yo

Detour to Rank Ordering ppt

Rank Ordering

Just in case you are interested in coding… ☺☺☺


```
## scoring step
CTDF.dev$predict.score <- predict(m1, CTDF.dev)

## deciling code
decile <- function(x){
  deciles <- vector(length=10)
  for (i in seq(0.1,1,.1)){
    deciles[i*10] <- quantile(x, i, na.rm=T)
  }
  return (
  ifelse(x<deciles[1], 1,
  ifelse(x<deciles[2], 2,
  ifelse(x<deciles[3], 3,
  ifelse(x<deciles[4], 4,
  ifelse(x<deciles[5], 5,
  ifelse(x<deciles[6], 6,
  ifelse(x<deciles[7], 7,
  ifelse(x<deciles[8], 8,
  ifelse(x<deciles[9], 9, 10
  ))))))))))
}

## deciling
CTDF.dev$deciles <- decile(CTDF.dev$predict.score[,2])
```

Some coding continued… ☺☺☺

```
## Ranking code
library(data.table)
tmp_DT = data.table(CTDF.dev)
rank <- tmp_DT[, list(
  cnt = length(Target),
  cnt_resp = sum(Target),
  cnt_non_resp = sum(Target == 0)) ,
  by=deciles][order(deciles)]
rank$rrate <- rank$cnt_resp * 100 / rank$cnt;
rank$cum_resp <- cumsum(rank$cnt_resp)
rank$cum_non_resp <- cumsum(rank$cnt_non_resp)
rank$cum_rel_resp <- rank$cum_resp / sum(rank$cnt_resp);
rank$cum_rel_non_resp <- rank$cum_non_resp /
sum(rank$cnt_non_resp);
rank$ks <- abs(rank$cum_rel_resp -
rank$cum_rel_non_resp);
rank
```

```
## Plotting ROC Curve and AUC
library(ROCR)
pred <- prediction(CTDF.dev$predict.score[,2],
CTDF.dev$Target)
perf <- performance(pred, "tpr", "fpr")
plot(perf)
KS <- max(attr(perf, 'y.values')[[1]]-attr(perf, 'x.values')[[1]])
auc <- performance(pred,"auc");
auc <- as.numeric(auc@y.values)
```

```
## Computing Gini Index
library(ineq)
gini = ineq(CTDF.dev$predict.score[,2], type="Gini")
```

```
## Output all the values
with(CTDF.dev, table(Target, predict.class))
auc
KS
gini
```

```
> with(CTDF.dev, table(Target, predict.class))
      predict.class
Target     0     1
     0 12663   102
     1  1056   179
> auc
[1] 0.7578043
> KS
[1] 0.4009363
> gini
[1] 0.4701245
```

Mis-Class = 8.3%

Area Under Curve

Classification Matrix		Predicted	
		Y	N
Actual	Y	a	b
	N	c	d

Sensitivity = True Positive Rate

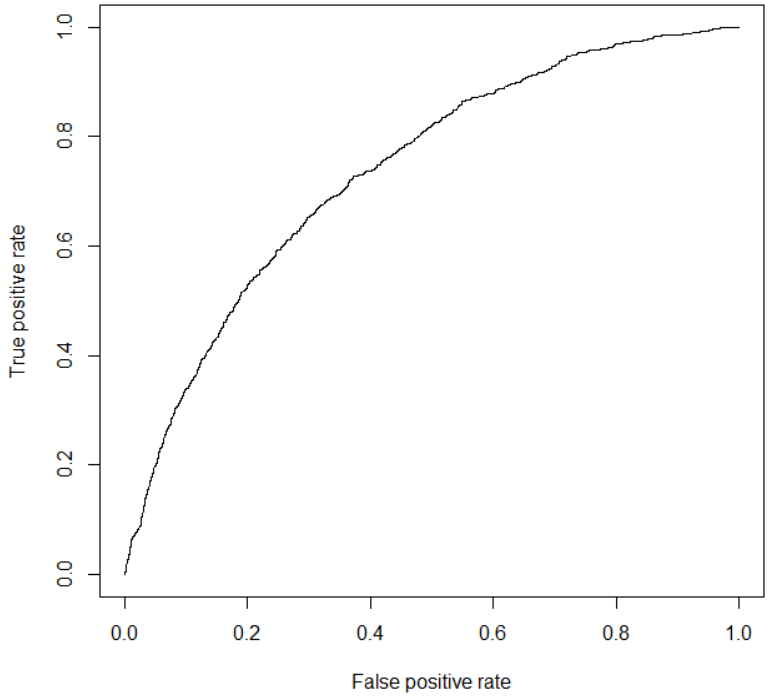
= True Positive / Total Positive

= a / (a + b)

Specificity = True Negative / Total Negative

= d / (c + d)

False Positive Rate = 1 - Specificity



Scoring

Syntax to get the node path

tree.path <- path.rpart (ptree, node = c(2, 12))

Scoring syntax

CTDF.dev$predict.class <- predict(m1, CTDF.dev, type="class")

CTDF.dev$predict.score <- predict(m1, CTDF.dev)

What to ensure when scoring a new dataset?

The **predictor variables** which are included in the final classification model should be present in the dataset to be scored

We can use the above syntax for scoring the Hold Out Sample also

CTDF.holdout$predict.class <- predict(m1, CTDF.holdout, type="class")

CTDF.holdout$predict.score <- predict(m1, CTDF.holdout)

Checking performance of mode on Hold Out Sample

with(CTDF.holdout, table(Target, predict.class))

```
> with(CTDF.holdout, table(Target, predict.class))
      predict.class
Target    0    1
     0 5454   48
     1  443   55
```

Mis-Class = 8.2%

K2 Analytics
Building Skills, Building Individuals

Questions?? … Thankyou

Contact Us
ar.jakhotia@k2analytics.co.in