Data Analytics and Business Intelligence (8696/8697)

Decision Trees

Graham.Williams@togaware.com

Data Scientist
Australian Taxation Office

Adjunct Professor, University of Canberra

`Graham.Williams@togaware.com`
`http://datamining.togaware.com`

Overview

Overview

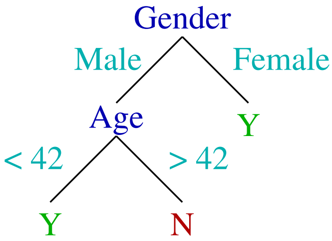
Overview

Predictive Modelling: Classification

- Goal of classification is to build models (sentences) in a knowledge representation (language) from examples of past decisions.
- The model is to be used on unseen cases to make decisions.
- Often referred to as supervised learning.
- Common approaches: decision trees; neural networks; logistic regression; support vector machines.

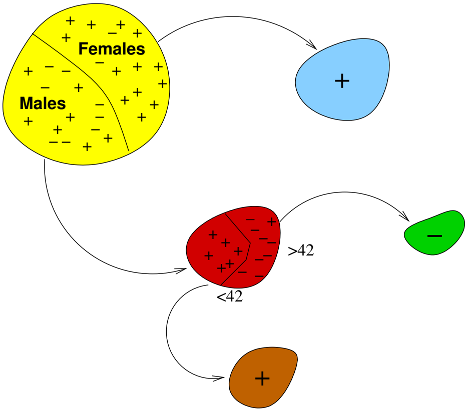
Language: Decision Trees

- Knowledge representation: A flow-chart-like tree structure
- Internal nodes denotes a test on a variable
- Branch represents an outcome of the test
- Leaf nodes represent class labels or class distribution



Tree Construction: Divide and Conquer

- Decision tree induction is an example of a recursive partitioning algorithm: divide and conquer.
- At start, all the training examples are at the root
- Partition examples recursively based on selected variables



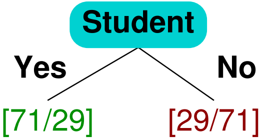
Training Dataset: Buys Computer?

What rule would you "learn" to identify who buys a computer?

Age	Income	Student	Credit	Buys
< 30	High	No	Poor	No
< 30	High	No	Good	Yes
$30 - 40$	High	No	Poor	Yes
> 40	Medium	No	Poor	Yes
> 40	Low	Yes	Poor	Yes
> 40	Low	Yes	Good	No
$30 - 40$	Low	Yes	Good	Yes
< 30	Medium	No	Poor	No
< 30	Low	Yes	Poor	No
> 40	Medium	Yes	Poor	Yes
< 30	Medium	Yes	Good	Yes
$30 - 40$	Medium	No	Good	Yes
$30 - 40$	High	Yes	Poor	Yes
> 40	Medium	No	Good	No

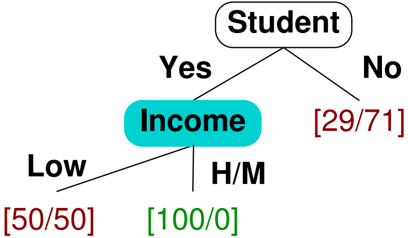
Output: Decision Tree for Buys Computer

One possible tree:



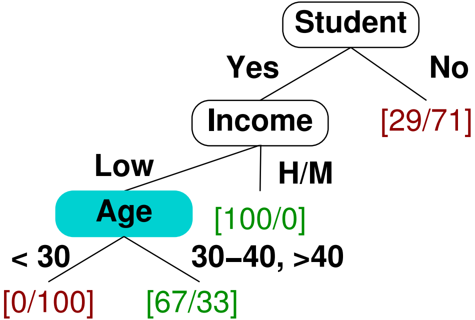
Output: Decision Tree for Buys Computer

One possible tree:



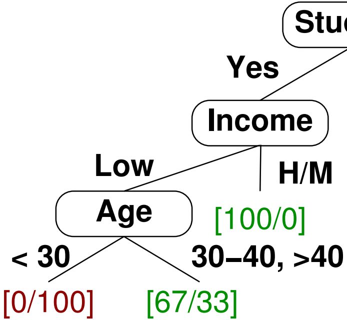
Output: Decision Tree for Buys Computer

One possible tree:



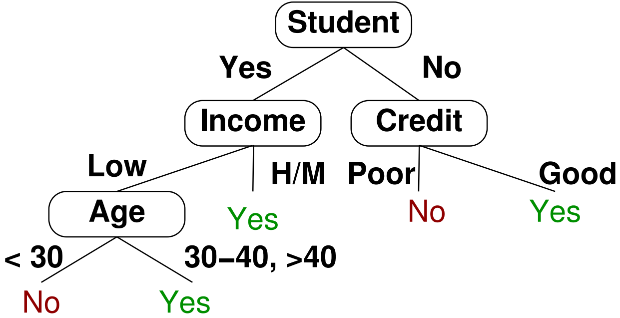
Output: Decision Tree for Buys Computer

One possible tree:



Output: Decision Tree for Buys Computer

One possible tree:



Algorithm for Decision Tree Induction

- A greedy algorithm: takes the best immediate (local) decision while building the overall model
- Tree constructed top-down, recursive, divide-and-conquer
- Begin with all training examples at the root
- Data is partitioned recursively based on selected variables
- Select variables on basis of a measure
- Stop partitioning when?
  - All samples for a given node belong to the same class
  - There are no remaining variables for further partitioning – majority voting is employed for classifying the leaf
  - There are no samples left

Basic Motivation: Entropy

We are trying to predict output Y (e.g., *Yes*/*No*) from input X.

- A random data set may have high entropy:
  - Y is from a uniform distribution
  - a frequency distribution would be flat!
  - a sample will include uniformly random values of Y
- A data set with low entropy:
  - Y's distribution will be very skewed
  - a frequency distribution will have a single peak
  - a sample will predominately contain just *Yes* or just *No*
- Work towards reducing the amount of entropy in the data!

Basic Motivation: Entropy

We are trying to predict output Y (e.g., *Yes*/*No*) from input X.

- A random data set may have high entropy:
  - Y is from a uniform distribution
  - a frequency distribution would be flat!
  - a sample will include uniformly random values of Y
- A data set with low entropy:
  - Y's distribution will be very skewed
  - a frequency distribution will have a single peak
  - a sample will predominately contain just *Yes* or just *No*
- Work towards reducing the amount of entropy in the data!

Basic Motivation: Entropy

We are trying to predict output Y (e.g., *Yes*/*No*) from input X.

- A random data set may have high entropy:
  - Y is from a uniform distribution
  - a frequency distribution would be flat!
  - a sample will include uniformly random values of Y
- A data set with low entropy:
  - Y's distribution will be very skewed
  - a frequency distribution will have a single peak
  - a sample will predominately contain just *Yes* or just *No*
- Work towards reducing the amount of entropy in the data!

Basic Motivation: Entropy

We are trying to predict output Y (e.g., *Yes*/*No*) from input X.

- A random data set may have high entropy:
  - Y is from a uniform distribution
  - a frequency distribution would be flat!
  - a sample will include uniformly random values of Y
- A data set with low entropy:
  - Y's distribution will be very skewed
  - a frequency distribution will have a single peak
  - a sample will predominately contain just *Yes* or just *No*
- Work towards reducing the amount of entropy in the data!

Entropy

We are trying to predict output Y from input X.

X = Course
Y = Purchase Neo1973

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Assuming this represents true probabilities:

$P(Yes) = 0.5$
$P(Math) = 0.5$
$P(Math \ \& \ Yes) = 0.25$
$P(History \ \& \ Yes) = 0$

http://www.cs.cmu.edu/~awm/tutorials

Entropy

We are trying to predict output Y from input X.

X = Course
Y = Purchase Neo1973

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Assuming this represents true probabilities:

$$P(Yes) = 0.5$$
$$P(Math) = 0.5$$
$$P(Math \ \& \ Yes) = 0.25$$
$$P(History \ \& \ Yes) = 0$$

http://www.cs.cmu.edu/~awm/tutorials

Entropy

We are trying to predict output Y from input X.

X = Course
Y = Purchase Neo1973

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Focus on Y

$P(Yes) = 0.5$
$P(No) = 0.5$

Uniform distribution of Y
Entropy of Y is 1

$$E(p, n) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

$log_2(0.5) = -1$

http://www.cs.cmu.edu/~awm/tutorials

Entropy

We are trying to predict output Y from input X.

X = Course
Y = Purchase Neo1973

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Focus on Y

$P(Yes) = 0.5$
$P(No) = 0.5$

Uniform distribution of Y
Entropy of Y is 1

$$E(p,n) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

$log_2(0.5) = -1$

http://www.cs.cmu.edu/~awm/tutorials

Entropy

We are trying to predict output Y from input X.

X = Course
Y = Purchase Neo1973

X	Y
Math	Yes
History	No
CS	Yes
Math	No
Math	No
CS	Yes
History	No
Math	Yes

Focus on Y

$P(Yes) = 0.5$
$P(No) = 0.5$

Uniform distribution of Y
Entropy of Y is 1

$$E(p,n) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

$log_2(0.5) = -1$

http://www.cs.cmu.edu/~awm/tutorials

Entropy

We are trying to predict output Y from input X.

X = Course
Y = Purchase Neo1973

X	Y
Math	Yes
History	**No**
CS	Yes
Math	No
Math	No
CS	Yes
History	**No**
Math	Yes

Focus on just students of History

$P(Yes) = 0$
$P(No) = 1$

Skewed distribution of Y
Entropy of Y is 0

$$E(p,n) = -\frac{0}{0+2}\log_2\frac{0}{0+2} - \frac{2}{0+2}\log_2\frac{2}{0+2}$$

$log_2(0) = -Inf \quad log_2(1) = 0$

http://www.cs.cmu.edu/~awm/tutorials

ENTROPY

We are trying to predict output Y from input X.

X = Course
Y = Purchase Neo1973

X	Y
Math	Yes
History	**No**
CS	Yes
Math	No
Math	No
CS	Yes
History	**No**
Math	Yes

Focus on just students of History

$P(Yes) = 0$
$P(No) = 1$

Skewed distribution of Y
Entropy of Y is 0

$$E(p, n) = -\frac{0}{0+2}\log_2\frac{0}{0+2} - \frac{2}{0+2}\log_2\frac{2}{0+2}$$

$log_2(0) = -Inf$ $log_2(1) = 0$

http://www.cs.cmu.edu/~awm/tutorials

Entropy

We are trying to predict output Y from input X.

X = Course
Y = Purchase Neo1973

X	Y
Math	Yes
History	**No**
CS	Yes
Math	No
Math	No
CS	Yes
History	**No**
Math	Yes

Focus on just students of History

$P(Yes) = 0$
$P(No) = 1$

Skewed distribution of Y
Entropy of Y is 0

$$E(p, n) = -\frac{0}{0+2}\log_2\frac{0}{0+2} - \frac{2}{0+2}\log_2\frac{2}{0+2}$$

$log_2(0) = -Inf \quad log_2(1) = 0$

http://www.cs.cmu.edu/~awm/tutorials

Variable Selection Measure: Entropy

- Information gain (ID3/C4.5)
- Select the variable with the highest information gain
- Assume there are two classes: P and N
- Let the data S contain p elements of class P and n elements of class N
- The amount of information, needed to decide if an arbitrary example in S belongs to P or N is defined as

$$I_E(p, n) = -\frac{p}{p+n}\log_2\frac{p}{p+n} - \frac{n}{p+n}\log_2\frac{n}{p+n}$$

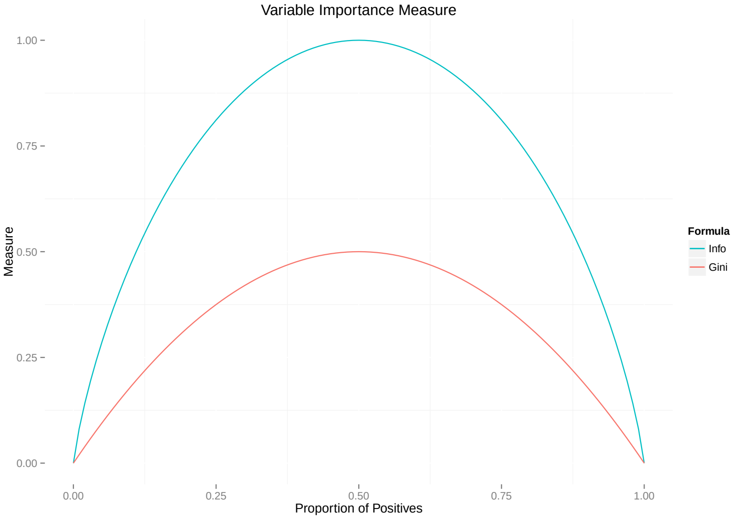
Variable Selection Measure: Gini

- Gini index of impurity – traditional statistical measure – CART
- Measure how often a randomly chosen observation is incorrectly classified if it were randomly classified in proportion to the actual classes.
- Calculated as the sum of the probability of each observation being chosen times the probability of incorrect classification, equivalently:

$$I_G(p, n) = 1 - (p^2 + (1 - p)^2)$$

- As with Entropy, the Gini measure is maximal when the classes are equally distributed and minimal when all observations are in one class or the other.

Variable Selection Measure



Information Gain

- Now use variable A to partition S into v cells: $\{S_1, S_2, \ldots, S_v\}$
- If S_i contains p_i examples of P and n_i examples of N, the information now needed to classify objects in all subtrees S_i is:

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

- So, the information gained by branching on A is:

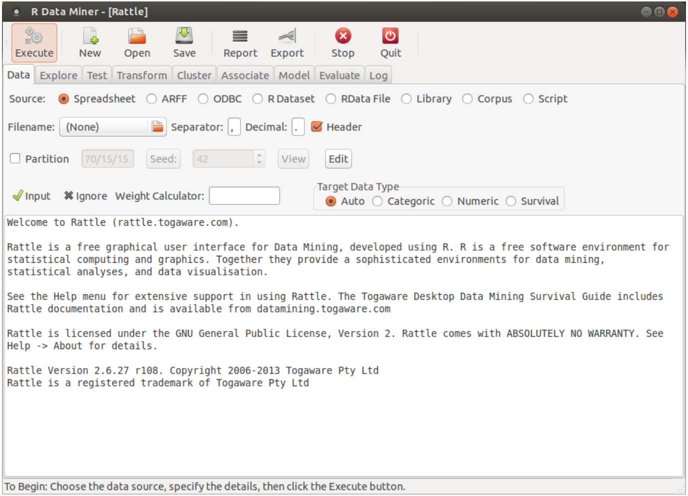
$$Gain(A) = I(p, n) - E(A)$$

So choose the variable A which results in the greatest gain in information.

Overview

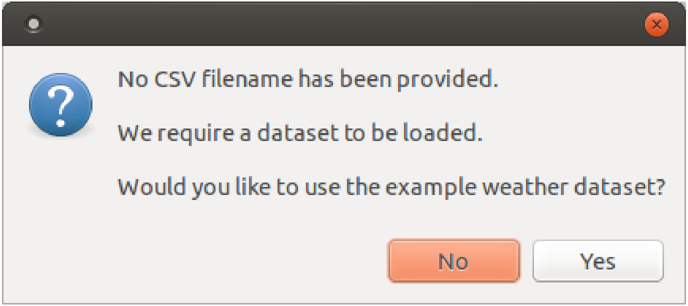
Startup Rattle

```
library(rattle)
rattle()
```



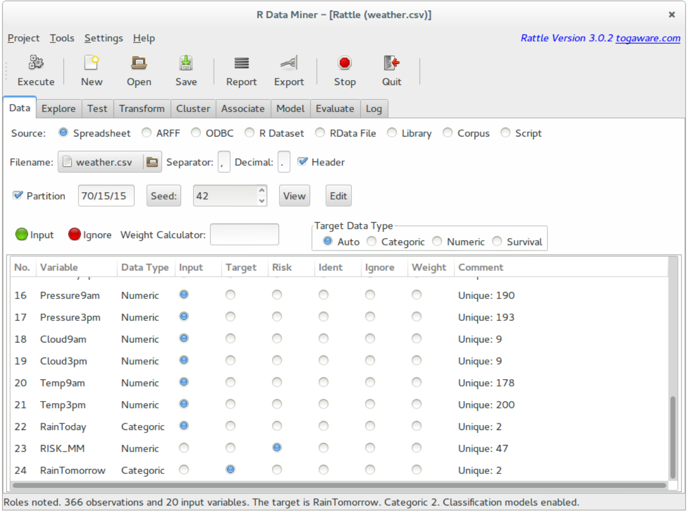
Load Example Weather Dataset

- Click on the Execute button and an example dataset is offered.
- Click on Yes to load the weather dataset.



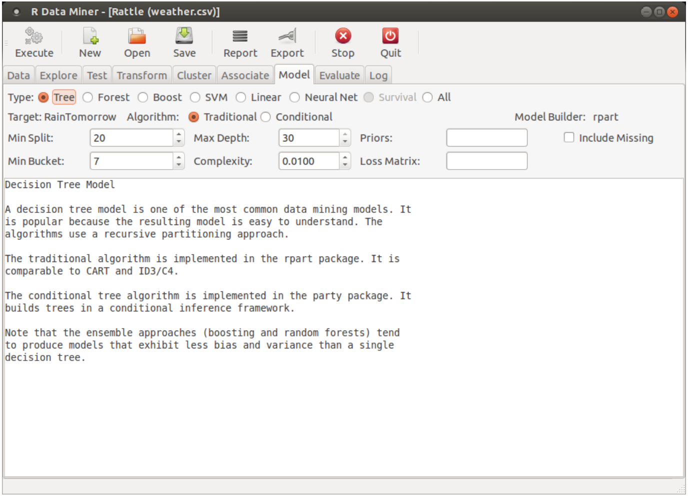
Summary of the Weather Dataset

- A summary of the weather dataset is displayed.



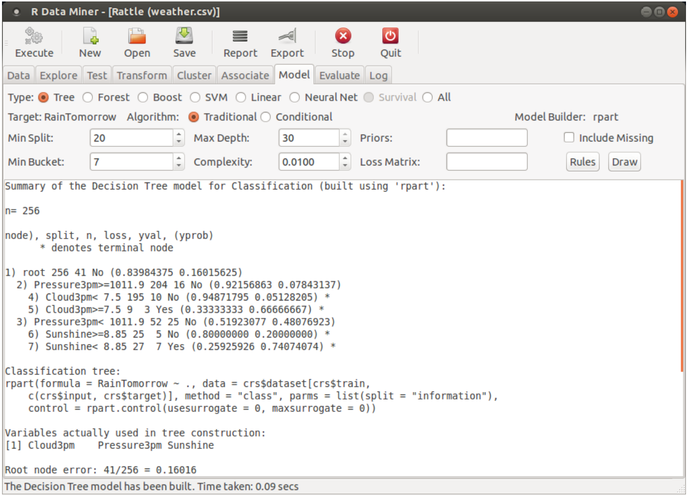
Model Tab — Decision Tree

- Click on the Model tab to display the modelling options.



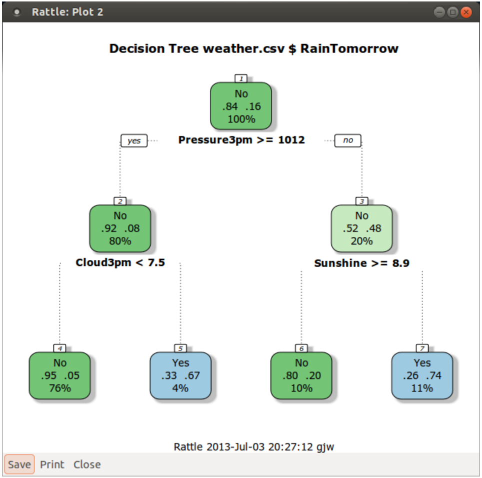
Build Tree to Predict RainTomorrow

- Decision Tree is the default model type—simply click Execute.



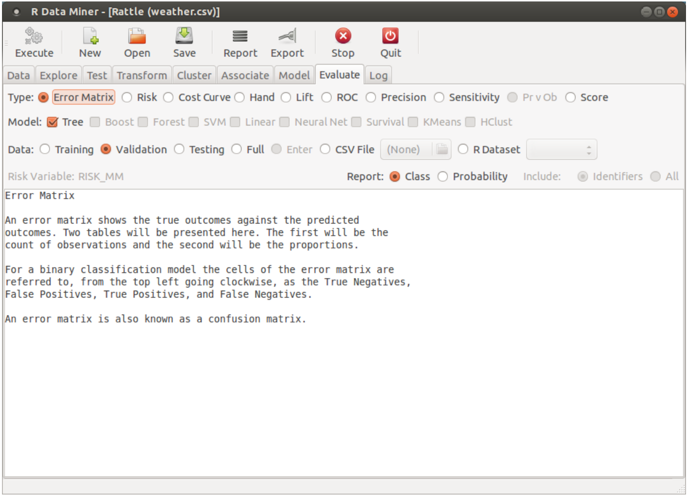
Decision Tree Predicting RainTomorrow

- Click the Draw button to display a tree (Settings → Advanced Graphics).



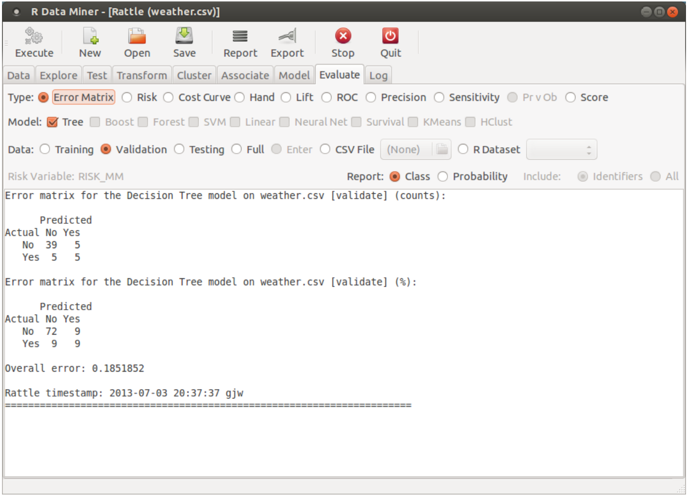
Evaluate Decision Tree

- Click Evaluate tab—options to evaluate model performance.



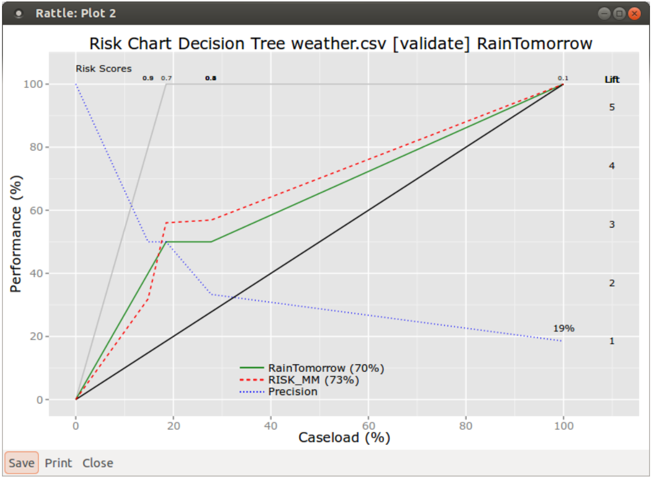
Evaluate Decision Tree—Error Matrix

- Click Execute to display simple error matrix.
- Identify the True/False Positives/Negatives.



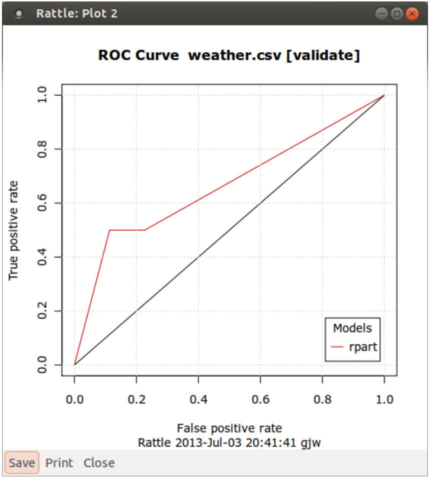
Decision Tree Risk Chart

- Click the Risk type and then Execute.



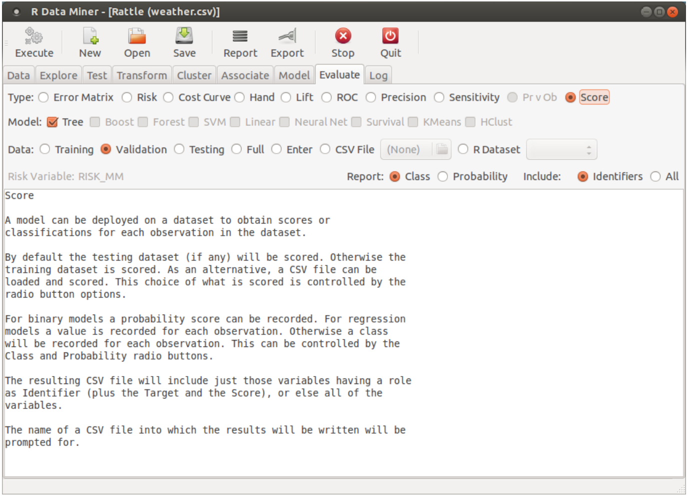
Decision Tree ROC Curve

- Click the ROC type and then Execute.



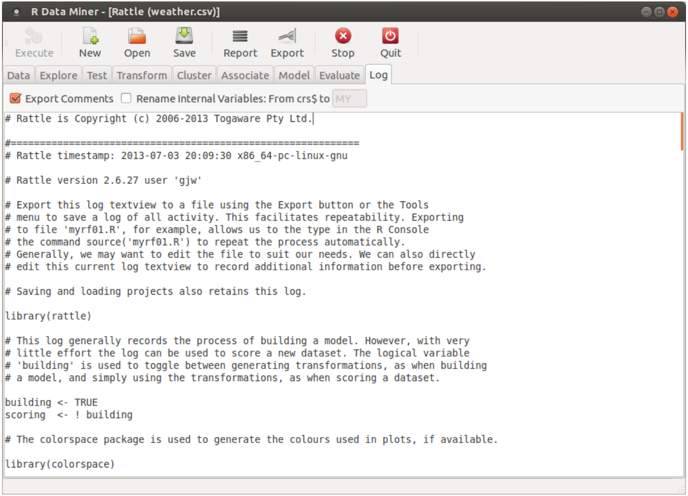
Score a Dataset

- Click the Score type to score a new dataset using model.



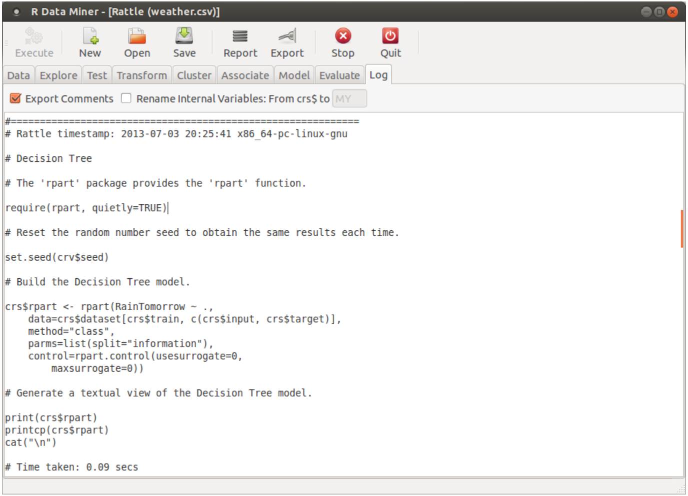
Log of R Commands

- Click the Log tab for a history of all your interactions.
- Save the log contents as a script to repeat what we did.



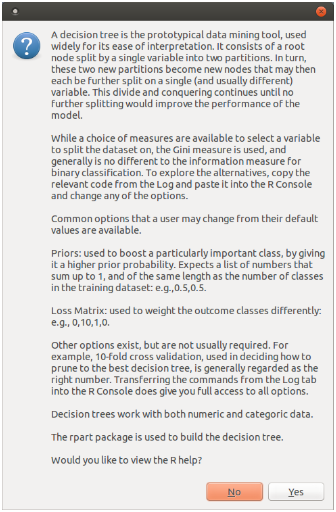
Log of R Commands—rpart()

- Here we see the call to rpart() to build the model.
- Click on the Export button to save the script to file.



Help → Model → Tree

- Rattle provides some basic help—click Yes for R help.



Overview

Weather Dataset - Inputs

```
ds <- weather
head(ds, 4)

##         Date Location MinTemp MaxTemp Rainfall Evaporation Sunshine
## 1 2007-11-01 Canberra     8.0    24.3      0.0         3.4      6.3
## 2 2007-11-02 Canberra    14.0    26.9      3.6         4.4      9.7
## 3 2007-11-03 Canberra    13.7    23.4      3.6         5.8      3.3
## 4 2007-11-04 Canberra    13.3    15.5     39.8         7.2      9.1
....

summary(ds[c(3:5,23)])

##     MinTemp          MaxTemp         Rainfall          RISK_MM
##  Min.   :-5.30   Min.   : 7.6   Min.   : 0.00   Min.   : 0.00
##  1st Qu.: 2.30   1st Qu.:15.0   1st Qu.: 0.00   1st Qu.: 0.00
##  Median : 7.45   Median :19.6   Median : 0.00   Median : 0.00
##  Mean   : 7.27   Mean   :20.6   Mean   : 1.43   Mean   : 1.43
....
```

Weather Dataset - Target

```
target <- "RainTomorrow"
summary(ds[target])

##  RainTomorrow
##  No :300
##  Yes: 66

(form <- formula(paste(target, "~ .")))

## RainTomorrow ~ .

(vars <- names(ds)[-c(1, 2, 23)])

##  [1] "MinTemp"       "MaxTemp"       "Rainfall"      "Evaporation"
##  [5] "Sunshine"      "WindGustDir"   "WindGustSpeed" "WindDir9am"
##  [9] "WindDir3pm"    "WindSpeed9am"  "WindSpeed3pm"  "Humidity9am"
## [13] "Humidity3pm"   "Pressure9am"   "Pressure3pm"   "Cloud9am"
## [17] "Cloud3pm"      "Temp9am"       "Temp3pm"       "RainToday"
## [21] "RainTomorrow"
```

Simple Train/Test Paradigm

```
set.seed(1421)
train <- c(sample(1:nrow(ds), 0.70*nrow(ds)))  # Training dataset
head(train)

## [1] 288 298 363 107  70 232

length(train)

## [1] 256

test  <- setdiff(1:nrow(ds), train)            # Testing dataset
length(test)

## [1] 110
```

Display the Model

```
model <- rpart(form, ds[train, vars])
model

## n= 256
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 256 44 No (0.82812 0.17188)
##    2) Humidity3pm< 59.5 214 21 No (0.90187 0.09813)
##      4) WindGustSpeed< 64 204 14 No (0.93137 0.06863)
##        8) Cloud3pm< 6.5 163  5 No (0.96933 0.03067) *
##        9) Cloud3pm>=6.5 41  9 No (0.78049 0.21951)
##         18) Temp3pm< 26.1 34  4 No (0.88235 0.11765) *
##         19) Temp3pm>=26.1 7  2 Yes (0.28571 0.71429) *
....
```

- Notice the legend to help interpret the tree.

Performance on Test Dataset

- The predict() function is used to score new data.

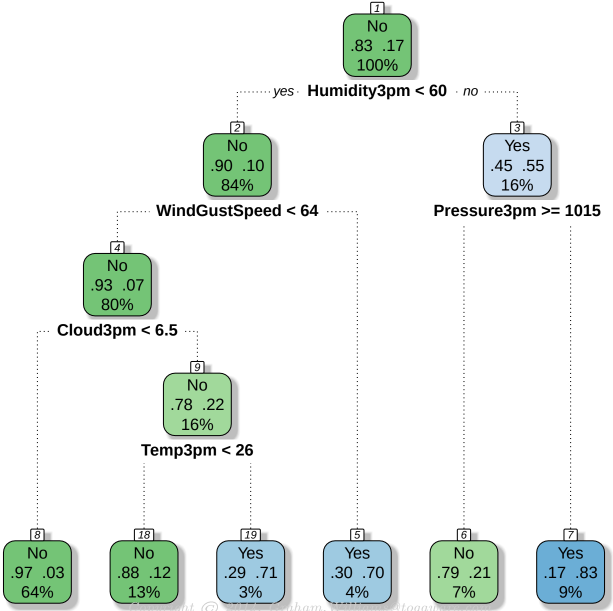
```
head(predict(model, ds[test,], type="class"))

##  2  4  6  8 11 12
## No No No No No No
## Levels: No Yes

table(predict(model, ds[test,], type="class"), ds[test, target])

##
##       No Yes
##   No  77  14
##   Yes 11   8
```


Example DTree Plot using Rattle



An R Scripting Hint

- Notice the use of variables `ds`, `target`, `vars`.
- Change these variables, and the remaining script is unchanged.
- Simplifies script writing and reuse of scripts.

```
ds <- iris
target <- "Species"
vars <- names(ds)
```

- Then repeat the rest of the script, without change.

An R Scripting Hint — Unchanged Code

- This code remains the same to build the decision tree.

```
form  <- formula(paste(target, "~ ."))
train <- c(sample(1:nrow(ds), 0.70*nrow(ds)))
test  <- setdiff(1:nrow(ds), train)
model <- rpart(form, ds[train, vars])
model

## n= 105
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 105 69 setosa (0.34286 0.32381 0.33333)
##   2) Petal.Length< 2.6 36  0 setosa (1.00000 0.00000 0.00000) *
##   3) Petal.Length>=2.6 69 34 virginica (0.00000 0.49275 0.50725)
##     6) Petal.Length< 4.95 35  2 versicolor (0.00000 0.94286 0.05714) *
##     7) Petal.Length>=4.95 34  1 virginica (0.00000 0.02941 0.97059) *
```

An R Scripting Hint — Unchanged Code

- Similarly for the predictions.

```
head(predict(model, ds[test,], type="class"))

##      3      8      9     10     11     12
## setosa setosa setosa setosa setosa setosa
## Levels: setosa versicolor virginica

table(predict(model, ds[test,], type="class"), ds[test, target])

##
##              setosa versicolor virginica
##   setosa         14          0         0
##   versicolor      0         15         4
##   virginica       0          1        11
```


Modelling Framework

Language Tree with single variable tests

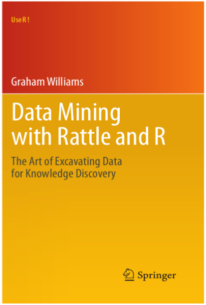
Measure Entropy, Gini, . . .

Search Recursive partitioning

Summary

- Decision Tree Induction.
- Most widely deployed machine learning algorithm.
- Simple idea, powerful learner.
- Available in R through the rpart package.
- Related packages include party, Cubist, C50, RWeka (J48).

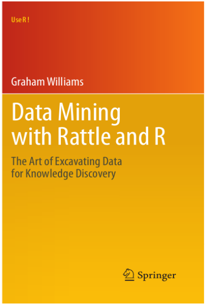
Reference Book



Data Mining with Rattle and R
Graham Williams
2011, Springer, Use R!
ISBN: 978-1-4419-9889-7.
Chapter 11.

~

Reference Book



Data Mining with Rattle and R
Graham Williams
2011, Springer, Use R!
ISBN: 978-1-4419-9889-7.
Chapter 11.