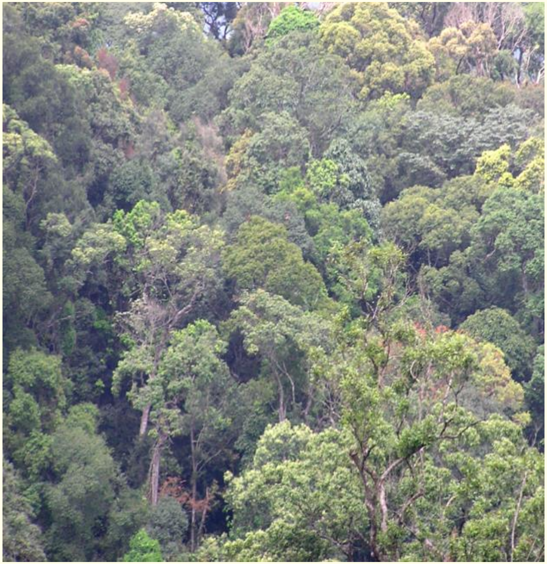
Overview of Random Forests

November 21, 2017



What is a Random Forest?

- An Ensemble method based on trees
- Can also be used in unsupervised manner
- Developed by Leo Brieman
- Good website

http://www.stat.berkeley.edu/b̃reiman/RandomForests/

What is a Random Forest?

- An ensemble of trees
- Two kind of randomness built in
- Cases are selected at random with replacement - training set
- At each split a random sample of m from M variables are selected
- m can be any number typically $\sqrt{M}$
- No pruning takes place theoretically

And More?

- Typically 1000 trees are grown
- For each tree about 36% of the data is not used
- These data are called out of bag samples (oob)
- Each tree votes for each case in the oob samples
- Aggregated over all trees
- Every tree carries equal weight
- Each case is assigned to the class with the most votes

Generic Algorithm for Ensembles

$$F(x) = c_0 + \sum_{m=1}^{M} c_m T_m(x) \qquad (1)$$

- Two stage process
- Determine base learners
- Determine weights $c_i's$

Generic algorithm for Ensembles

```
Step 1- Choose $\{\mathbf{p_m}\}$
$F_0(\mathbf{x}) = \mathbf{0}$
For m=1 to M {
    $\mathbf{p_m} = \mathbf{argmin_p} \sum_{\mathbf{i} \in \mathbf{S_{n(\eta)}}} \mathbf{L}(\mathbf{y_i}, \mathbf{F_{m-1}}(\mathbf{x_i}) + \mathbf{T}(\mathbf{x_i}; \mathbf{p}))$
    $T_m(\mathbf{x}) = \mathbf{T}(\mathbf{x_i}; \mathbf{p})$
    $F_m(\mathbf{x}) = \mathbf{F_{m-1}}(\mathbf{x}) + \nu \mathbf{T_m}(\mathbf{x})$
}
write $\{T_m(x)\}_1^M$
```

Ensembles - Determining $c_i's$

- We can use simple formula
- Determine them at each step along the way
- Calculate them afterwards using penalized regression function
- Post processing

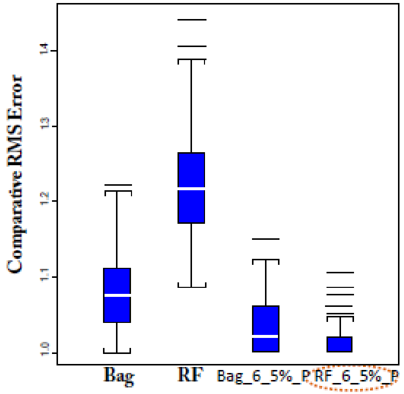
Ensembles - Post processing

- At this point all base learners have been selected
- Have to the determine the $c_i's$
- Do this by regularised regression

$$\{c_m\} = \operatorname*{argmin}_{c_m} \sum_{i=1}^{N} L\left((y_i, c_0 + \sum_{m=1}^{M} c_m T_m(x_i)) \right) + \lambda.P(c) \qquad (2)$$

- P(c) is the complexity penalty and λ controlling the amount of regularisation.

Comparisons



Other output from Random Forests

- Misclassification matrix
- Margin of classifier
- Variable importance
- Proximity matrix
- Missing value imputation
- Partial Dependency plot

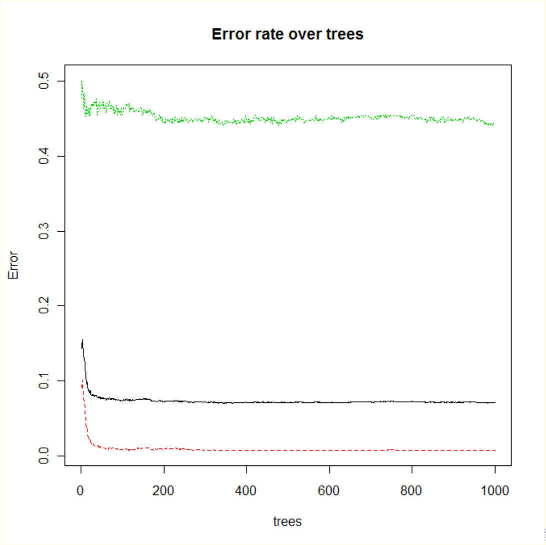
Some data from the churn dataset

- No. of variables tried at each split: 3
- OOB estimate of error rate: 7.23%
- Confusion matrix:

	No	Yes	class.error
No	2827	23	0.008
Yes	218	265	0.45

- For 1 tree on training set - 7.65%

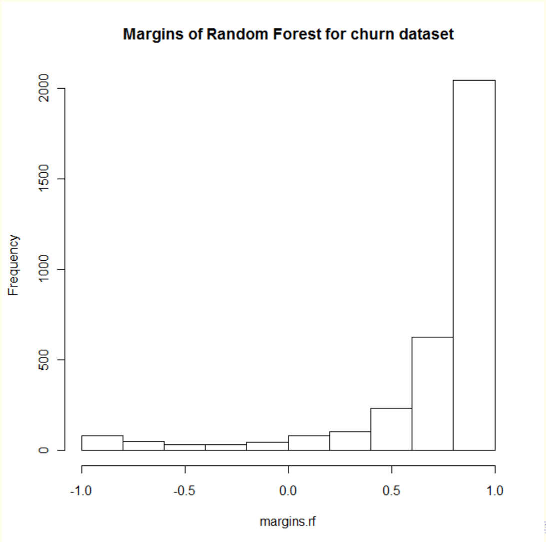
Trace of error



Margins of Classifier

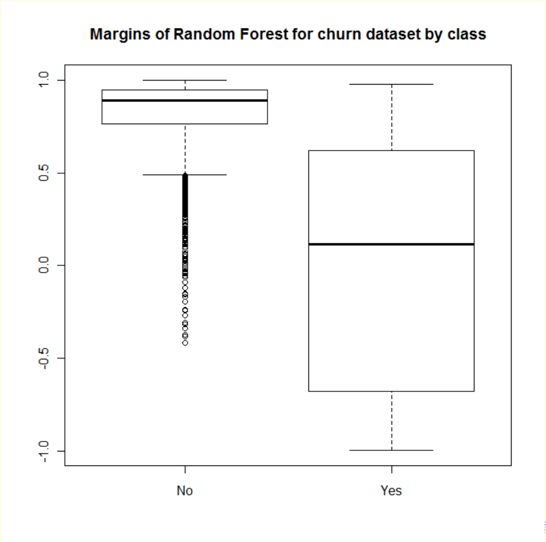
- Proportion of votes for each class
- Assign case to class with highest proportion of votes
- Margin of a case = proportion of votes for correct class - max proportion assigned other classes
- Should these be big or small?

Margins



Margins of Random Forest for churn dataset
Frequency
0
500
1000
1500
2000
-1.0
-0.5
0.0
0.5
1.0
margins.rf

And yet again ...



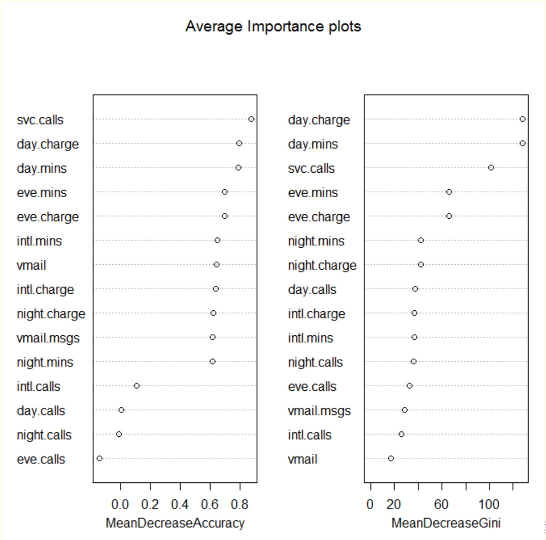
Variable Importance

- Two Approaches
  - Contribution to Fit
    - Decrease in fitting measure e.g. Gini
  - Contribution to prediction method
- Can calculate these for each class
- Can calculate for overall result

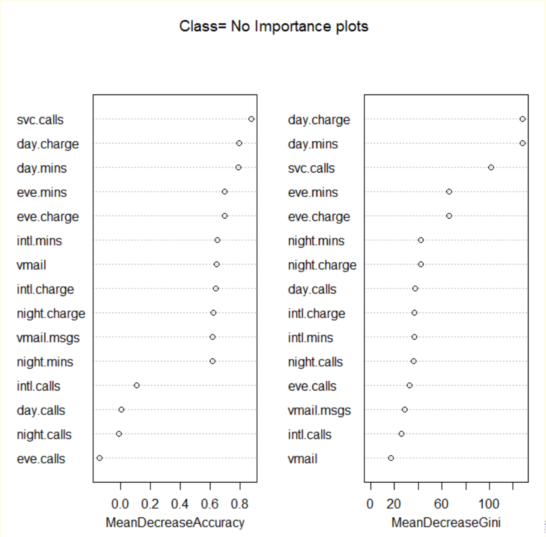
Prediction method

- For each tree calculated % misclassified (v_1) for each class and overall using oob cases
- For each predictor randomly sort the cases and put cases down the tree again
- Calculate % misclassified again (v_2) for each class and overall
- Calculate difference $v_1 - v_2$
- Average results over all trees

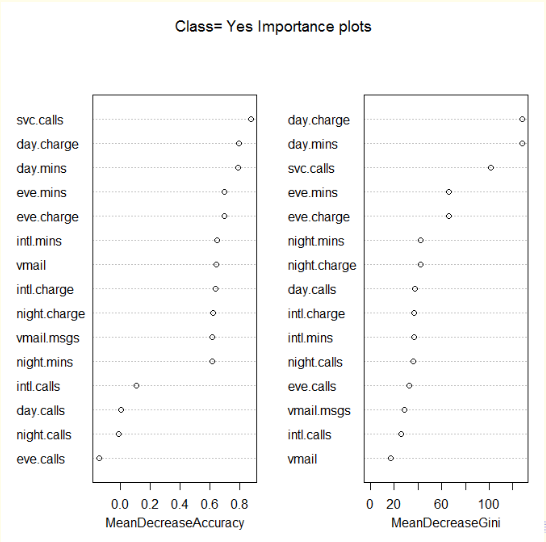
Variable Importance



Variable Importance



Variable Importance



And more ...

- Only 1 predictor at a time
- Independent of other predictors
- This is important

Proximity of cases

- Calculates a NxN proximity matrix $P(i,j)$
- Every element initially set to 0
- If case i and j end up in the same node $P(i,j) = P(i,j) + 1$
- Accumulate over all trees and normalise
- Can use this proximity matrix as an input to MDS

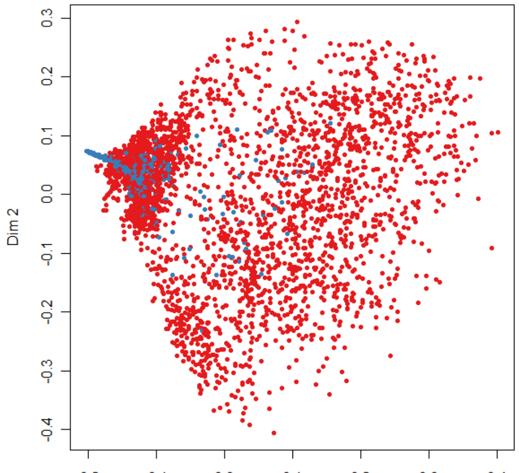
Proximity Matrix

- Degree to which individual observations are classified alike
- Grow a tree as usual
- Drop all the training (in bag and out of bag) down the tree
- For all possible pairs of cases, if a pair lands in the same terminal node increase their proximity by 1
- Repeat until all trees have been grown
- Normalise by dividing by the number of trees

Use of the Proximity Matrix

- Can be used as input to multidimensional scaling
- Can be used for imputing data
- Quantitative data
    - For each variable calculate median value and use for missing cases.
    - Grow tree and calculate proximities
    - Weighted average over non-missing cases using proximities as weights -
    - assign this as new imputed value
    - Reiterate

MDS PLot



More on Proximity

- Does not work on very large datasets
- Only the top 100 closest cases are recorded ???
- Does work for all types of data

Advantages of Random Forests I

- Simple to implement
- Good classifier
- Lots of other information
- Runs efficiently on large databases
- Gives estimates of what variables are important
- Generates an internal unbiased estimate of the generalisation error
- Works with large number of variables
- Can be extended to unlabelled data
- Different types of variables

Advantages of Random Forests II

- Can use the proximity matrix as an input to MDS
- Can grow them in parallel
- Not too many parameters to estimate
- has been found to work well with highly nonlinear classifiers

Disadvantages of Random Forests

- Model size
- May not be fast enough to calculate real time predictions
- They sometimes overfit data with noisy classification or regression tasks
- Where they are categorical varaibles with different number of levels, RFs are biased in favour of variables with more levels

Other Points

- B i.i.d variables each with variance σ^2
- Assuming independence average has variance $\frac{1}{B}\sigma^2$
- For non-independence variance of average is
- $$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2$$
- ρ is the correlation between two trees.
- RFs aims to reduce ρ
- RF's work well with highly non-linear classifiers

Other Points

- ρ correlation between the trees depends on m
- Increasing ρ increases the forest error rate
- Increasing the strength of the individual trees decreases the forest error rate.
- The larger m is the "better" the tree
- Reducing m reduces both the correlation and the strength.
- Increasing m increases both the correlation and the strength
- Find optimum m - results suggests $\sqrt{p}$ where p is the number of variables

Tuning parameters

- Node Size for growing trees
- Number of trees
- Number of predictors sampled

Costs and Random Forests

- Cannot include costs like in a single tree
- But we can alter priors
- We can change the cutoff used to assign classes
- In other words do not use majority voting

Costs and priors for 2 classes

- For 2 classes we can alter the priors to reflect differences in costs of misclassification

$$\pi^{new}(j) = \frac{C(j|i) * \pi(j)}{\sum_i C(i|j) * \pi(i)}$$

- $C(j|i) = C(j) =$ cost of misclassifying j

$$\pi^{new}(j) = \frac{C(j) * \pi(j)}{\sum_i C(i) * \pi(i)}$$

Costs and priors for 2 classes

- equal priors; C(1) =2 and C(2)=1

$$\pi^n(1) = \frac{C(1) * \pi(1)}{\sum_{i=1,2} C(i) * \pi(i)} = \frac{2 * .5}{2 * .5 + 1 * .5} = .67$$

$$\pi^n(2) = \frac{C(2) * \pi(2)}{\sum_{i=1,2} C(i) * \pi(i)} = \frac{1 * .5}{2 * .5 + 1 * .5} = .33$$

- The new priors reflect the different cost structure
- You can use this technique for more than 2 classes provided that the matrix has a certain structure

Unsupervised learning and random Forests

- We can just use a RF to create a proximity matrix as input to Clustering or Multi dimensional scaling
- There is another very unusual approach here
- Create a new outcome variable with two classes
- The first class is all the original data
- Create a synthetic second class of the same size - class 2
- For class 2 the independent variables are created by sampling at random from the univariate distributions of the original data
- No dependency among the variables in class 2

Results

- The object is to see if there is structure in the original data
- If the misclassification rate is high (40%) this suggests that there is no structure
- The original data is like random independent data
- Low misclassification rates suggest that there is structure in the original data.

Quotation from Brieman and Cutler

Take the output of random forests not as absolute truth, but as smart computer generated guesses that may be helpful in leading to a deeper understanding of the problem.