Foundational Machine Learning
Week 3 : Random Forests and Bias Variance Tradeoff

Rohit Babbar
rb2608@bath.ac.uk

UNIVERSITY OF BATH

This lecture

- Last week:
  - Decision trees

    Input $=$ real

    Output $=$ categorical (discrete)

This lecture

- Last week:
  - Decision trees

    Input = real

    Output = categorical (discrete)
- This week:
  - Decision trees with some further input and output types
  - Bias-variance tradeoff – some theoretical insights
  - Bagging - an ensembling technique
  - **Random forests**

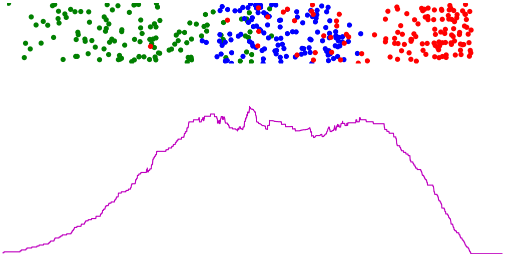
Inputs

Real input

(from last time)

- Choose best Gini impurity / info gain for all axes

  (1 axis shown; vertical offset for visualisation only)

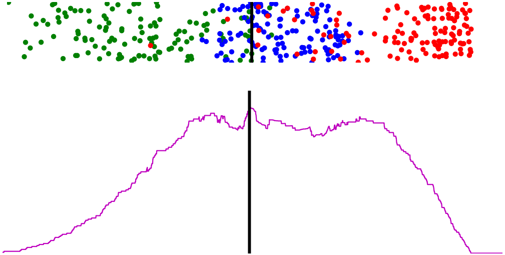


Real input

(from last time)

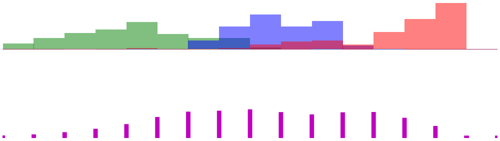
- Choose best Gini impurity / info gain for all axes

(1 axis shown; vertical offset for visualisation only)



Quantised real input

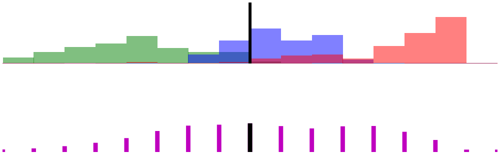
- Continuous data may be **quantised**
- e.g. “What is your age?”
  - From 12–17, 18–24 etc.
  - Or year only



Quantised real input

- Continuous data may be **quantised**
- e.g. “What is your age?”
  - From 12–17, 18–24 etc.
  - Or year only
- Split between bins

  (Information gain shown as spikes
  as only defined at bin transitions)



Categorical input

- Similar to quantised (histogram of catgories). . .
  . . . but unordered
- e.g. “What is your favourite cheese?”
- Spliting no longer makes sense!

Categorical input

- Similar to quantised (histogram of catgories)...

  ...but unordered
- e.g. “What is your favourite cheese?”
- Spliting no longer makes sense!

- Two choices :
  - Try every assignment of category to the left/right side and pick best

    (one category always goes left to account for symmetry)
  - One category goes down left branch, rest go right

Categorical input

- Similar to quantised (histogram of catgories)...
  ...but unordered
- e.g. “What is your favourite cheese?”
- Spliting no longer makes sense!

- Two choices :
  - Try every assignment of category to the left/right side and pick best
    (one category always goes left to account for symmetry)
  - One category goes down left branch, rest go right

- One left, rest right is preferred:
  - Fixed storage
  - Simpler code
  - Combinations to test grows linearly
    ($\mathcal{O}(n)$, not $\mathcal{O}(2^{n-1})$; where $n =$ numbers of catgories)

Outputs

Regression

- Classification:
  - Split to minimise Gini impurity or maximise information gain
  - Leaf gives answer as most common class to reach it

Regression

- Classification:
  - Split to minimise Gini impurity or maximise information gain
  - Leaf gives answer as most common class to reach it
- Regression:
  - Split to minimise variance or maximise information gain
  - Leaf gives answer as mean value to reach it

    (median may confer an advantage — any idea when?)
- Otherwise identical!

Variance reduction

- Variance of output measures how consistent a node is...

  ...so choose splits that minimise it

Variance reduction

- Variance of output measures how consistent a node is...

  ...so choose splits that minimise it

- Variance of left node:

  $$\sigma_l^2 = \mathbb{E}\left[(Y_l - \mathbb{E}[Y_l])^2\right]$$

  similarly for right, σ_r^2 ($Y_l =$ data that goes left)

Variance reduction

- Variance of output measures how consistent a node is...
  ...so choose splits that minimise it

- Variance of left node:

$$\sigma_l^2 = \mathbb{E}\left[(Y_l - \mathbb{E}[Y_l])^2\right]$$

  similarly for right, σ_r^2 ($Y_l =$ data that goes left)

- Minimise weighted combination:

$$L(\texttt{split}) = \frac{n_l}{n}\sigma_l^2 + \frac{n_r}{n}\sigma_r^2$$

  $n =$ total exemplar count
  $n_l =$ exemplars traveling down left branch
  $n_r =$ exemplars traveling down right branch

Information gain

- Information gain is not often used for regression tasks with decision trees, as it is not immediately applicable

[1] https://www.biopsychology.org/norwich/isp/chap8.pdf

Information gain

- Information gain is not often used for regression tasks with decision trees, as it is not immediately applicable

In order to apply,

- First fit Gaussian distribution to output variable
- Then, compute entropy [1]

$$\frac{1}{2}\log\left(2\pi e\sigma^2\right)$$

- Information gain, thus, is

$$I(\texttt{split}) = \frac{1}{2}\log\left(2\pi e\sigma_p^2\right) - \frac{n_l}{2n}\log\left(2\pi e\sigma_l^2\right) - \frac{n_r}{2n}\log\left(2\pi e\sigma_r^2\right)$$

(same variables as previous slide, with p subscript for parent)

[1] `https://www.biopsychology.org/norwich/isp/chap8.pdf`

From Decision Trees to Random Forests via Bias-variance tradeoff

Bias-Variance Tradeoff

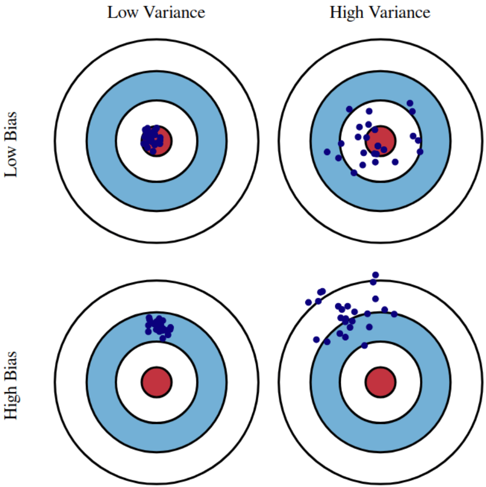


Figure: Pictorial depiction of the components of bias-variance tradeoff

Unbiased Estimator Concept

Definition

An estimator $\hat{\theta}$ of a population parameter θ is **unbiased** if:

$$\mathbb{E}[\hat{\theta}] = \theta$$

That is, on average, it neither overestimates nor underestimates θ.

Sample Mean Setup

Let $X_1, X_2, \ldots, X_n$ be i.i.d. random variables with:

$$\mathbb{E}[X_i] = \mu \quad \text{and} \quad Var(X_i) = \sigma^2$$

The sample mean is defined as:

$$\bar{X} = \frac{1}{n} \sum_{i=1}^{n} X_i$$

Expectation of the Sample Mean

Using the linearity of expectation:

$$\mathbb{E}[\bar{X}] = \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} X_i\right] = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[X_i] = \frac{1}{n}\cdot n\mu = \mu$$

Conclusion

$$\boxed{\mathbb{E}[\bar{X}] = \mu}$$

The sample mean is therefore an **unbiased estimator** of the population mean.

A common way to compute Sample variance

Biased estimator (uses n in denominator)

$$\tilde{s}^2 = \frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X})^2$$

Key identity:

$$\sum_{i=1}^{n}(X_i - \bar{X})^2 = \sum_{i=1}^{n}(X_i - \mu)^2 - n(\bar{X} - \mu)^2$$

Take expectations (i.i.d., $E[X_i] = \mu$, $\mathrm{Var}(X_i) = \sigma^2$):

$$\mathbb{E}\left[\sum_{i=1}^{n}(X_i - \mu)^2\right] = n\sigma^2, \qquad \mathbb{E}\left[n(\bar{X} - \mu)^2\right] = n \cdot \mathrm{Var}(\bar{X}) = n \cdot \frac{\sigma^2}{n} = \sigma^2$$

Therefore:

$$\mathbb{E}\left[\tilde{s}^2\right] = \frac{1}{n}\left(n\sigma^2 - \sigma^2\right) = \left(1 - \frac{1}{n}\right)\sigma^2 = \frac{n-1}{n}\sigma^2$$

Turns out that using $(n-1)$ in the denominator makes it unbiased

Bias-variance tradeoff[2] [3]

Consider regression problem with squared error

- Given a training set $D = \{(x_i, y_i)\}_{i=1}^{n}$ such that $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$.
- Assume that $y_i = f(x_i) + \epsilon$, where ϵ is a random variable representing noise with mean 0 and variance σ^2
- Since $f(.)$ is unknown, we try to approximate it using the training data, and let $\hat{f}(.)$ denote our approximation. Imagine $\hat{f}(.)$ to be decision tree (for regression) that we constructed,
- In regression, this is done by minimising the squared error between y and $f(x)$, i.e. $(y - \hat{f}(x))^2$, where the pair (x, y) could be a training data point or a novel/unseen test data point.

[2] Based on Wikipedia article `https://en.wikipedia.org/wiki/Bias-variance_tradeoff`

[3] Derivation as a whole is non-examinable, but you should still have an understanding of the individual parts and the overall idea of the proof

Bias-variance tradeoff - II

Irrespective of how the classifier $\hat{f}(.)$ is learnt on the data, it's **expected** error on an unseen sample (test) sample x can be decomposed as follows :

$$\mathbb{E}_{D,\varepsilon}\left[\left(y - \hat{f}(x;D)\right)^2\right] = \left(\mathrm{Bias}_D\left[\hat{f}(x;D)\right]\right)^2 + \mathrm{Var}_D\left[\hat{f}(x;D)\right] + \sigma^2$$

where

$$\mathrm{Bias}_D\left[\hat{f}(x;D)\right] \triangleq \mathbb{E}_D\left[\hat{f}(x;D)\right] - f(x)$$

$$\mathrm{Var}_D\left[\hat{f}(x;D)\right] \triangleq \mathbb{E}_D\left[\left(\mathbb{E}_D[\hat{f}(x;D)] - \hat{f}(x;D)\right)^2\right]$$

$$\sigma^2 = \mathrm{E}_y\left[\left(y - f(x)\right)^2\right]$$

Note $\triangleq$ means this is a definition

Bias-variance tradeoff - III

- Rewriting the LHS of the equation on the previous slide as MSE (Mean squared error)

$$\begin{aligned}
\mathrm{MSE} &\triangleq \mathbb{E}\left[\left(y - \hat{f}(x)\right)^2\right] && \textit{writing the expectation } \mathbb{E} \textit{ without subsripts} \\
&= \mathbb{E}\left[\left(f(x) + \varepsilon - \hat{f}(x)\right)^2\right] && \text{since } y \triangleq f(x) + \varepsilon \\
&= \mathbb{E}\left[\left(f(x) - \hat{f}(x)\right)^2\right] + 2\,\mathbb{E}\left[\left(f(x) - \hat{f}(x)\right)\varepsilon\right] + \mathbb{E}[\varepsilon^2]
\end{aligned}$$

- Using the fact that for independent r.v. X and Y, $\mathbb{E}(XY) = \mathbb{E}(X)\mathbb{E}(Y)$

$$\begin{aligned}
\mathbb{E}\left[\left(f(x) - \hat{f}(x)\right)\varepsilon\right] &= \mathbb{E}\left[f(x) - \hat{f}(x)\right]\,\mathbb{E}\left[\varepsilon\right] && \text{since } \varepsilon \text{ is independent from } x \\
&= 0 && \text{since } \mathbb{E}\left[\varepsilon\right] = 0
\end{aligned}$$

- Expanding the first term below :

$$\begin{aligned}
\mathbb{E}\left[\left(f(x) - \hat{f}(x)\right)^2\right] &= \mathbb{E}\left[\left(f(x) - \mathbb{E}\left[\hat{f}(x)\right] + \mathbb{E}\left[\hat{f}(x)\right] - \hat{f}(x)\right)^2\right] \\
&= \mathbb{E}\left[\left(f(x) - \mathbb{E}\left[\hat{f}(x)\right]\right)^2\right] + 2\,\mathbb{E}\left[\left(f(x) - \mathbb{E}\left[\hat{f}(x)\right]\right)\left(\mathbb{E}\left[\hat{f}(x)\right] - \hat{f}(x)\right)\right] \\
&+ \mathbb{E}\left[\left(\mathbb{E}\left[\hat{f}(x)\right] - \hat{f}(x)\right)^2\right]
\end{aligned}$$

Bias-variance tradeoff - IV

$$
\begin{aligned}
\mathbb{E}\Big[\big(f(x)-\mathbb{E}[\hat{f}(x)]\big)^2\Big] &= \mathbb{E}[f(x)^2] - 2\,\mathbb{E}\Big[f(x)\,\mathbb{E}[\hat{f}(x)]\Big] + \mathbb{E}\Big[\mathbb{E}[\hat{f}(x)]^2\Big] \\
&= f(x)^2 - 2\,f(x)\,\mathbb{E}[\hat{f}(x)] + \mathbb{E}[\hat{f}(x)]^2 \\
&= \Big(f(x)-\mathbb{E}[\hat{f}(x)]\Big)^2
\end{aligned}
$$

The term in red :

$$
\begin{aligned}
\mathbb{E}\Big[\big(f(x)-\mathbb{E}[\hat{f}(x)]\big)\big(\mathbb{E}[\hat{f}(x)]-\hat{f}(x)\big)\Big] &= \mathbb{E}\Big[f(x)\,\mathbb{E}[\hat{f}(x)] - f(x)\hat{f}(x) - \mathbb{E}[\hat{f}(x)]^2 + \mathbb{E}[\hat{f}(x)]\,\hat{f}(x)\Big] \\
&= f(x)\,\mathbb{E}[\hat{f}(x)] - f(x)\,\mathbb{E}[\hat{f}(x)] - \mathbb{E}[\hat{f}(x)]^2 + \mathbb{E}[\hat{f}(x)]^2 \\
&= 0
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{MSE} &= \Big(f(x)-\mathbb{E}[\hat{f}(x)]\Big)^2 + \mathbb{E}\Big[\big(\mathbb{E}[\hat{f}(x)]-\hat{f}(x)\big)^2\Big] + \sigma^2 \\
&= \mathrm{Bias}\,\big(\hat{f}(x)\big)^2 + \mathrm{Var}\,[\hat{f}(x)] + \sigma^2
\end{aligned}
$$

Bias-variance tradeoff - V

- Variance: Captures how much the classifier changes if trained on a (slightly) different training set. How "over-specialized" is it to a particular training set (overfitting)?
- Bias: What is the inherent error that classifiers incurs even with infinite training data? This is due to the classifier being "biased" to a particular kind of solution (e.g. linear classifier).
- Noise: How much is the data-intrinsic noise? It's a measure of the ambiguity due to the data distribution and feature representation (ϵ in the above example).
- The goal in Random Forest (next) is to lower the variance $\mathbb{E}\left[\left(\mathbb{E}\left[\hat{f}(x)\right] - \hat{f}(x)\right)^2\right]$ by having more and more estimates of $\hat{f}(x)$, i.e. growing lots of decision trees into (random) forests !

Random Forests

Ensemble learning I

- Random forest = decision trees + ensemble learning

Ensemble learning I

- Random forest = decision trees + ensemble learning
- Ensemble learning = combining multiple estimators
  - Different models (e.g. linear regression, decision tree, SVM, neural network)

    or
  - Same model, randomised training so each is different

(using estimator to distinguish from model)

Ensemble learning I

- Random forest = decision trees + ensemble learning

- Ensemble learning = combining multiple estimators
  - Different models (e.g. linear regression, decision tree, SVM, neural network)

    or
  - Same model, randomised training so each is different

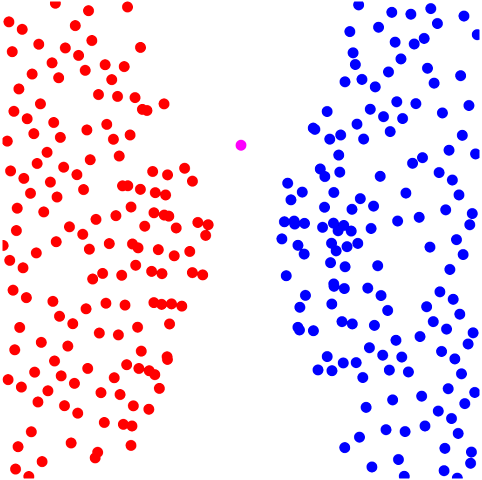
  (using estimator to distinguish from model)

- Random forest:
  - Many decision trees (hence name)
  - Randomised training using **bagging**,

    a specific ensemble learning technique

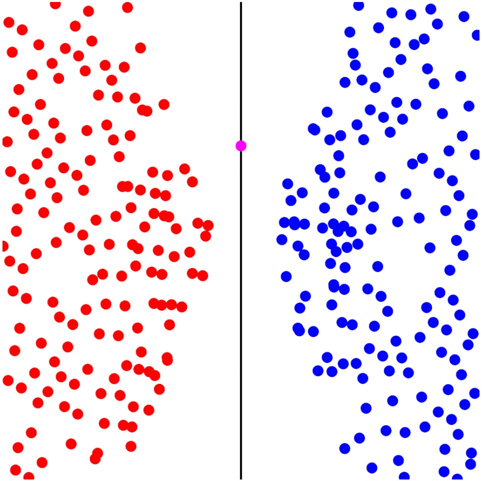
Ensemble learning II

- Which class (red or blue) should the magneta dot be?



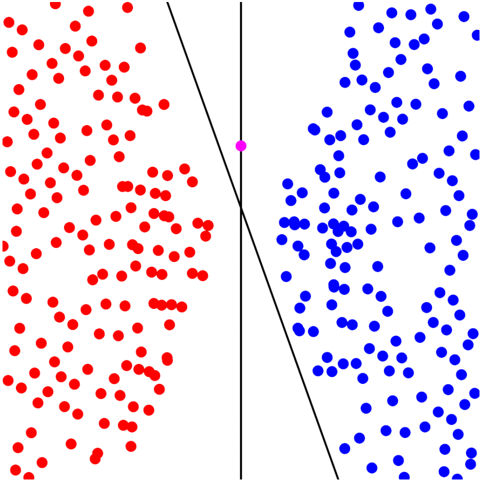
Ensemble learning II

- Which class (red or blue) should the magneta dot be?
- An obvious classification boundary . . .



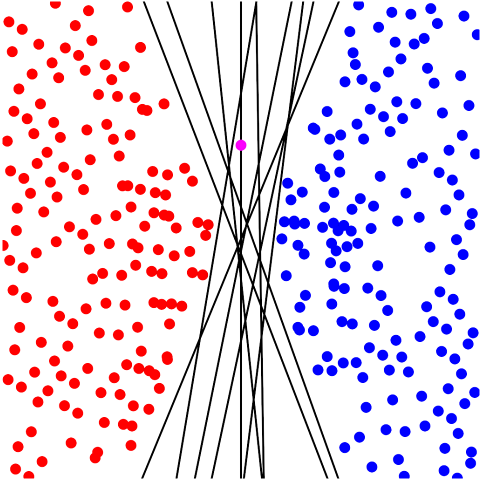
Ensemble learning II

- Which class (red or blue) should the magneta dot be?
- An obvious classification boundary . . .
- But this is just as good (suggesting blue)



Ensemble learning II

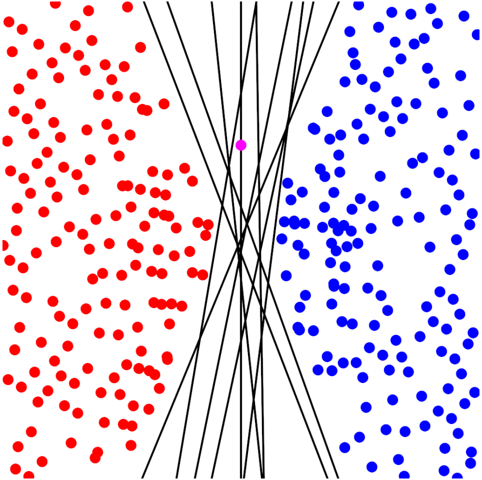
- Which class (red or blue) should the magneta dot be?
- An obvious classification boundary . . .
- But this is just as good (suggesting blue)
- As are all of these!



Ensemble learning II

- Which class (red or blue) should the magneta dot be?
- An obvious classification boundary . . .
- But this is just as good (suggesting blue)
- As are all of these!

- Models can be fit in many ways due to:
  - Insufficient data
  - Noisy data
  - **Model not complex enough** (curved boundaries can also separate this data!)

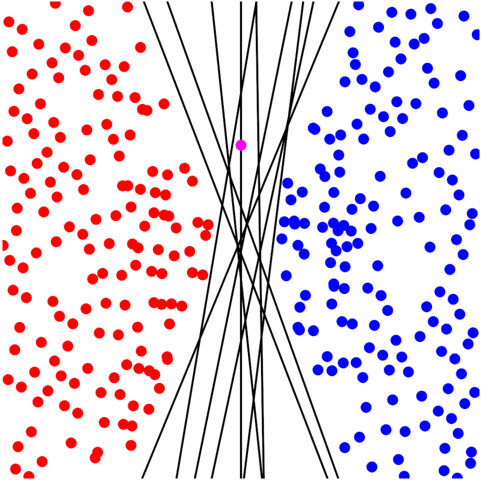


Ensemble learning II

- Which class (red or blue) should the magneta dot be?
- An obvious classification boundary . . .
- But this is just as good (suggesting blue)
- As are all of these!

- Models can be fit in many ways due to:
  - Insufficient data
  - Noisy data
  - **Model not complex enough**
    (curved boundaries can also separate this data!)

- Ensembles have many estimators. . .
  . . . to capture this ambiguity



Ensemble learning III

- Ensembles need **diversity**
- Estimators must make **different mistakes**
  i.e. if all make same mistake $\implies$ ensemble will repeat it
- Increasing estimator diversity at expense of individual performance → better ensemble!
  (up to a limit)

Bootstrapping

- However, constructing an ensemble would require more data
- Instead of collecting more data. . .

  . . . fake it from available data

Bootstrapping

- However, constructing an ensemble would require more data
- Instead of collecting more data. . .
  . . . fake it from available data
- Given data set of size n:
  Create new data set by drawing, with replacement, n times
  (there will be repetitions of data-points due to replacement)

Bagging

- An ensemble technique!

- Short for “Bootstrap AGGregatING”
- Bagging = Bootstrapping applied to estimator output
  (via optimised parameters)

Bagging

- An ensemble technique!

- Short for “Bootstrap AGGregatING”
- Bagging = Bootstrapping applied to estimator output

  (via optimised parameters)

- Algorithm:
  1. Select S, size of ensemble
  2. Create S bootstrap draws of original data set
  3. Train estimator on each
  4. Combine outputs of all estimators for each query

Random subspace method

- Another ensemble technique!
- Bootstrap applied to features
  i.e. fit each estimator with a random subset of features
- For decision tree: new bootstrap for each split

Random forests

- Random forest = decision trees + bagging + random subspace method
- Algorithm:
  1. Select S, number of trees (more is better, up to a limit)
  2. Create S bootstrap draws of original data set
  3. Train decision tree on each, with random subspace method
  4. Combine outputs of all trees for each query
     - Classification : The decision trees vote – ensemble outputs winner
     - Regression : Take the mean/median of all of the estimates

Explainability

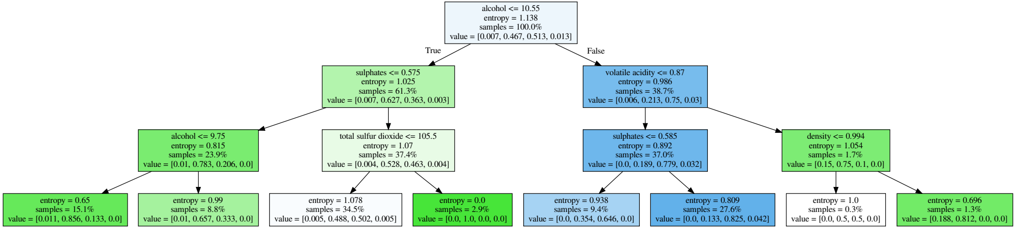
Red wine I

- 1599 exemplars, split: 1199 train, 400 test.
- Input: 11 measurable features:
  - fixed acidity
  - volatile acidity
  - citric acid
  - residual sugar
  - chlorides
  - free sulfur dioxide
  - total sulfur dioxide
  - density
  - pH
  - sulphates
  - alcohol
- Output: 1–10 human rating

  (reduced to 1–4 here, to fit on screen)
- Can be used for classification or regression!

Red wine II

Classification tree, max depth 3:



alcohol <= 10.55
entropy = 1.138
samples = 100.0%
value = [0.007, 0.467, 0.513, 0.013]

True

False

sulphates <= 0.575
entropy = 1.025
samples = 61.3%
value = [0.007, 0.627, 0.363, 0.003]

volatile acidity <= 0.87
entropy = 0.986
samples = 38.7%
value = [0.006, 0.213, 0.75, 0.03]

alcohol <= 9.75
entropy = 0.815
samples = 23.9%
value = [0.01, 0.783, 0.206, 0.0]

total sulfur dioxide <= 105.5
entropy = 1.07
samples = 37.4%
value = [0.004, 0.528, 0.463, 0.004]

sulphates <= 0.585
entropy = 0.892
samples = 37.0%
value = [0.0, 0.189, 0.779, 0.032]

density <= 0.994
entropy = 1.054
samples = 1.7%
value = [0.15, 0.75, 0.1, 0.0]

entropy = 0.65
samples = 15.1%
value = [0.011, 0.856, 0.133, 0.0]

entropy = 0.99
samples = 8.8%
value = [0.01, 0.657, 0.333, 0.0]

entropy = 1.078
samples = 34.5%
value = [0.005, 0.488, 0.502, 0.005]

entropy = 0.0
samples = 2.9%
value = [0.0, 1.0, 0.0, 0.0]

entropy = 0.938
samples = 9.4%
value = [0.0, 0.354, 0.646, 0.0]

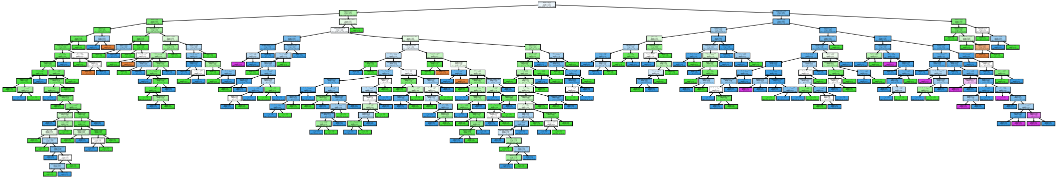
entropy = 0.809
samples = 27.6%
value = [0.0, 0.133, 0.825, 0.042]

entropy = 1.0
samples = 0.3%
value = [0.0, 0.5, 0.5, 0.0]

entropy = 0.696
samples = 1.3%
value = [0.188, 0.812, 0.0, 0.0]

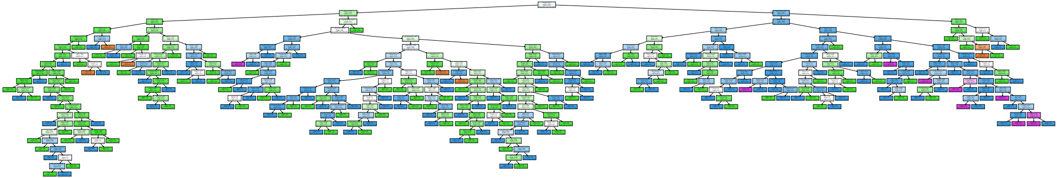
Accuracy = 69% (and explainable)

Red wine III



- A deeper classification tree:
    - Accuracy = 71% (somewhat less explainable)

Red wine III



- A deeper classification tree:
  - Accuracy = 71% (somewhat less explainable)

- Random forests with 32 trees
  - Classification: Accuracy = 79% (better than 71%)
  - Impossible to visualise/understand!

Summary

- “Upgraded” decision trees to
  - Handle more types of inputs & outputs
  - Random forests! (still one of the best)
  - Output probabilities
- Notes:
  - One of the fastest algortihms
  - Many variants, e.g. gradient boosting
- Next week :

  Making sure a ML system is working!

Further reading

- For more kinds of random forest:
  “Decision Forests for Classification, Regression, Density Estimation, Manifold Learning and Semi-Supervised Learning“ by **Criminisi, Shotton and Konukoglu**
  `http://research.microsoft.com/apps/pubs/default.aspx?id=155552`

- For more on ensemble methods:
  “Diversity creation methods: a survey and categorisation”
  by **Brown, Wyatt, Harris and Yao**
  `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.421.349&rep=rep1&type=pdf`

Legal

Cheese board, CC Worm That Turned, Attribution-Share Alike 4.0 International, https://commons.wikimedia.org/wiki/File:Welsh_cheese_board.JPG

Wine data set from https://archive.ics.uci.edu/ml/datasets/wine+quality