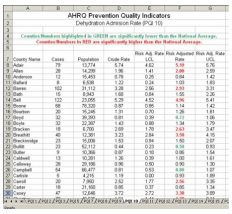
COMP24111: Machine Learning

Ensemble Models

Gavin Brown
www.cs.man.ac.uk/~gbrown



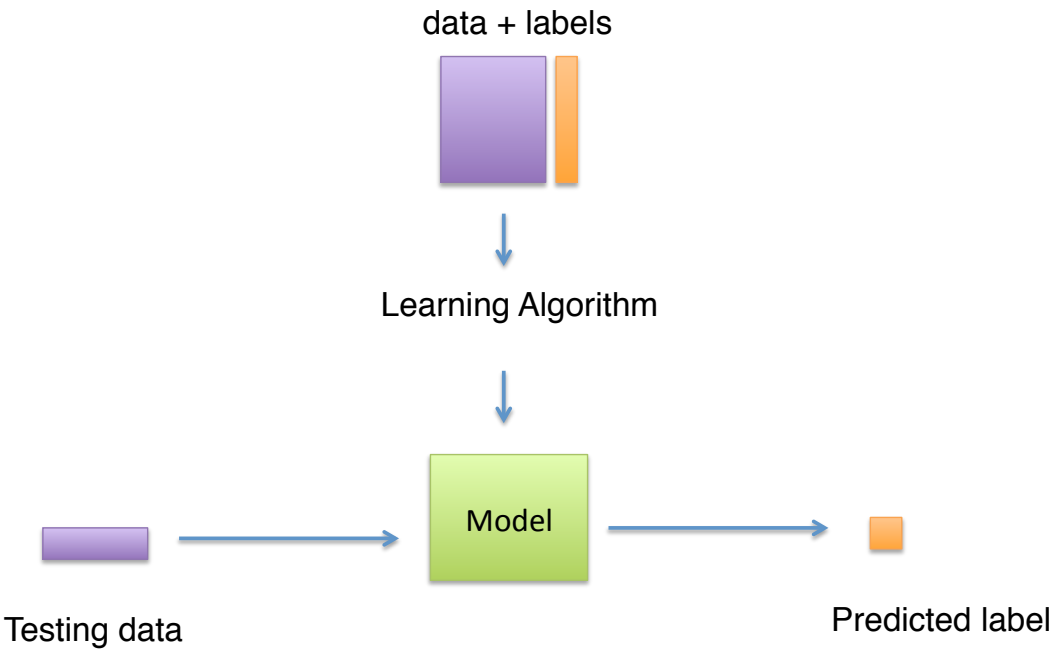
Projects...

Deadline is thursday for group G, Friday for group H.

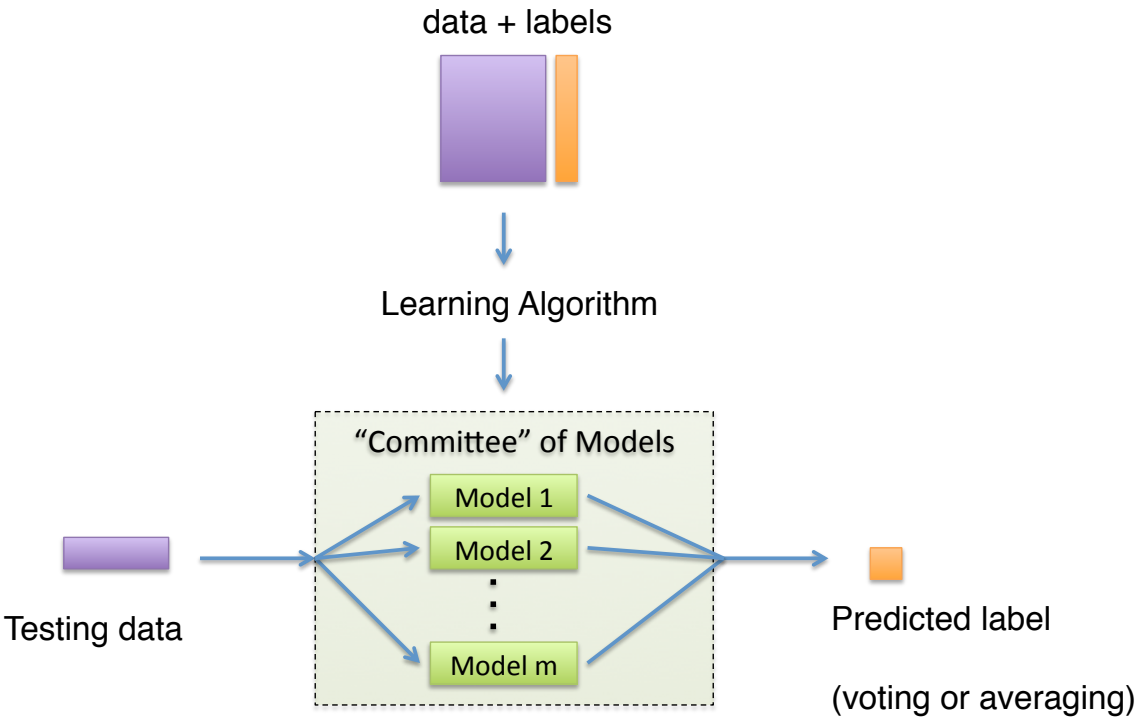
You should submit TWO files:

`COMP24111/ex2/ex2code.zip` ... (all your matlab files)
`COMP24111/ex2/ex2report.pdf` ... (your report in PDF)

The Usual Supervised Learning Approach



Ensemble Methods



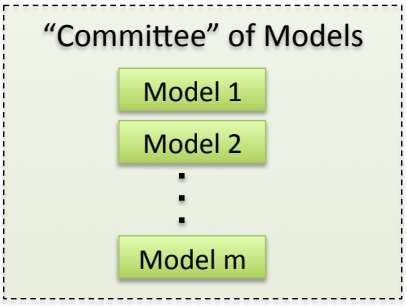
Ensemble Methods

Like a “committee”….

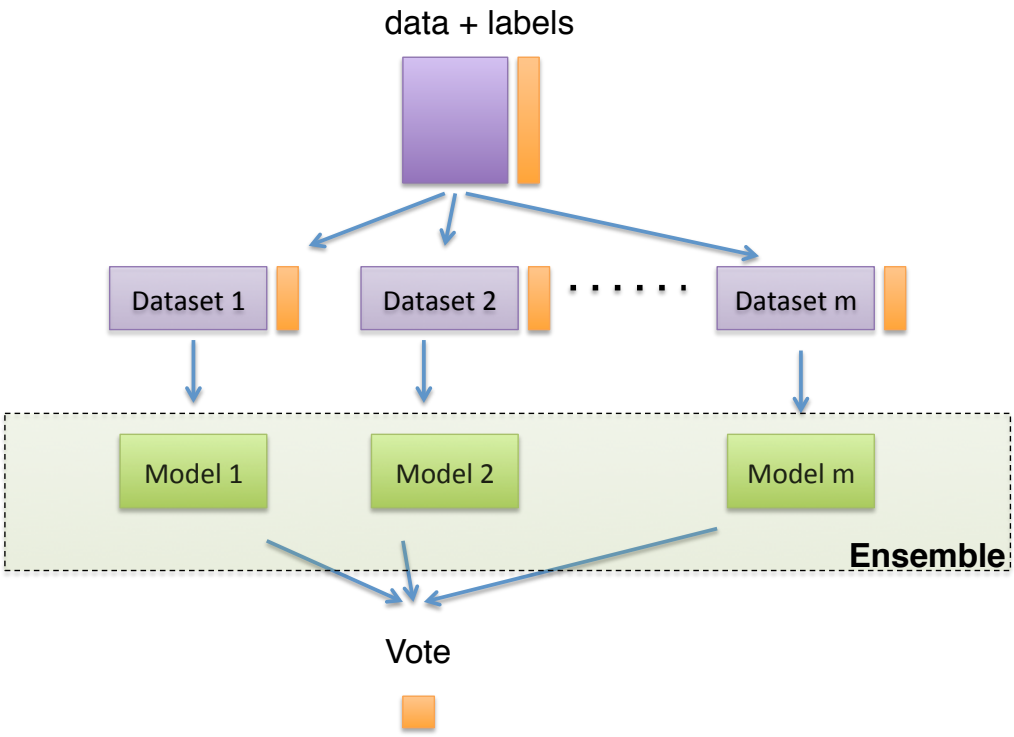
- don’t want all models to be identical.
- don’t want them to be different for the sake of it – sacrificing individual performance

This is the “Diversity” trade-off.

Equivalent to underfitting/overfitting – need the right level, just enough.



Parallel Ensemble Methods



Generating a new dataset by “Bootstrapping”

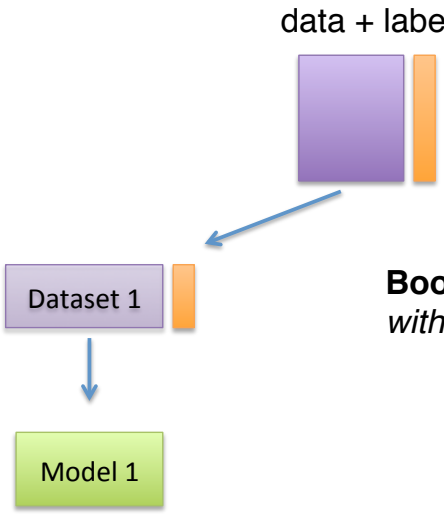
- sample N items with replacement from the original N

x_1	x_2	x_3	x_4	x_5	y
187	80	120	30	4.5	0
160	70	119	36	5.6	0
150	80	185	60	8.8	1
192	92	140	50	6.8	1
168	110	155	45	7.8	1

187	80	120	30	4.5	0
150	80	185	60	8.8	1
150	80	185	60	8.8	1
168	110	155	45	7.8	1
168	110	155	45	7.8	1

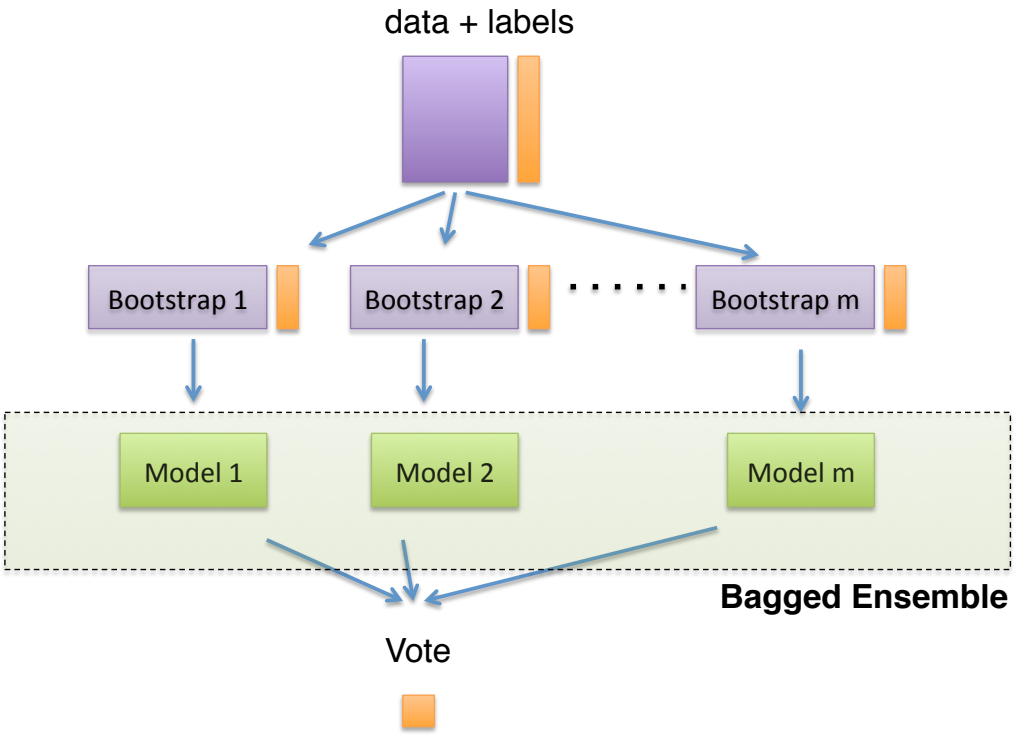
160	70	119	36	5.6	0
160	70	119	36	5.6	0
150	80	185	60	8.8	1
192	92	140	50	6.8	1
168	110	155	45	7.8	1

“Bagging” : **B**ootstrap **AGG**regat**ING**



Bootstrap: Sample N examples,
with replacement, from the original N.

"Bagging" : **B**ootstrap **AGG**regat**ING**



The "Bagging" algorithm

Bagging (input training data+labels T, number of models M)

---

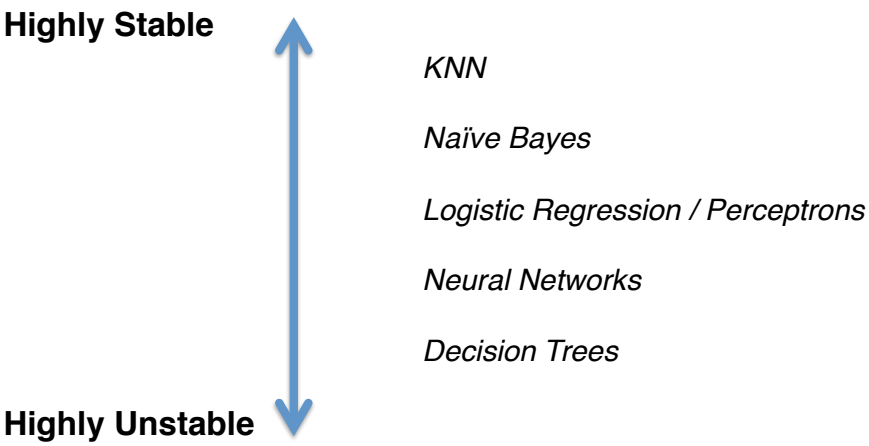
for $j = 1$ to M **do**
  Take a *bootstrap* sample T' from T
  Build a model using T'.
  Add the model to the set.
end for
return set of models

For a test point $\mathbf{x}$, get a response from each model, and take a majority vote.

Model Stability

Some models are almost completely unaffected by bootstrapping.

These are **stable** models. Not so suitable for ensemble methods.



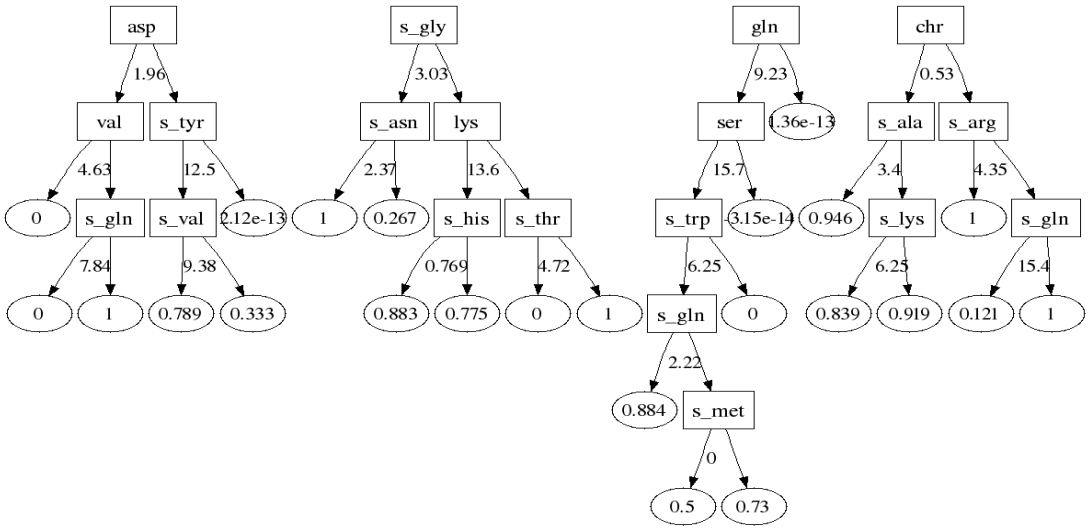
Encouraging diversity...

Give each model a slightly *different* dataset!

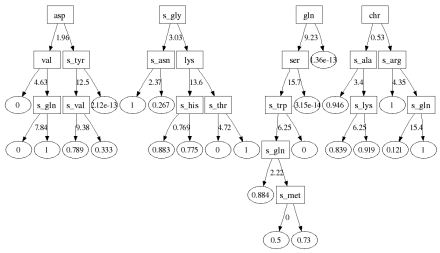
- Cross Validation
  - Split the data into N folds
  - Fit each model using a different fold as training data
- Feature Extraction
  - Generate n different feature sets
  - Fit each model using a different feature set
- Can you think of any more?

Use bagging on a set of decision trees.

How can we make them even more **diverse**?



Random Forests



Random Forests (input training data+labels T, number of trees M)

---

for $j = 1$ to M **do**
  Take a bootstrap sample T' from T
  Build a decision tree using T', but, at every split point:
    - Choose a random fraction K of the remaining features,
    - Pick the best feature (minimising cost) from that subset.
  Add the tree to the set, *without pruning*
end for
return set of trees

For a test point $\mathbf{x}$, get a response from each tree, and take a majority vote.

Real-time human pose recognition in parts from single depth images
Computer Vision and Pattern Recognition 2011
Shotton et al, Microsoft Research

- *Basis of Kinect controller*
- *Features are simple image properties*
- *Test phase: 200 frames per sec on GPU*
- *Train phase more complex but still parallel*

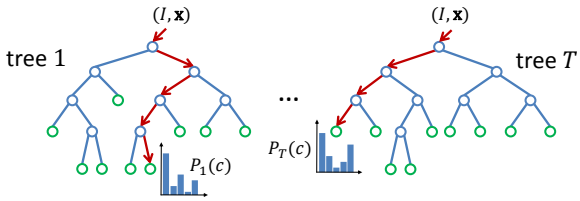


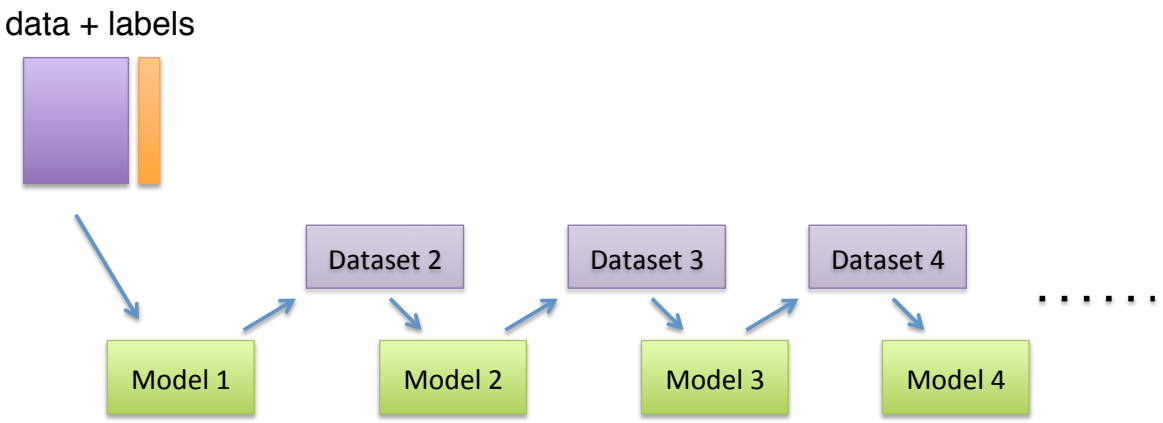
Figure 4. **Randomized Decision Forests.** A forest is an ensemble of trees. Each tree consists of split nodes (blue) and leaf nodes (green). The red arrows indicate the different paths that might be taken by different trees for a particular input.

3.3. Randomized decision forests

Randomized decision trees and forests [35, 30, 2, 8] have proven fast and effective multi-class classifiers for many tasks [20, 23, 36], and can be implemented efficiently on the GPU [34]. As illustrated in Fig. 4, a forest is an ensemble of T decision trees, each consisting of split and leaf nodes.

To keep the training times down we employ a distributed implementation. Training 3 trees to depth 20 from 1 million images takes about a day on a 1000 core cluster.

Sequential Ensemble Methods



Each model corrects the mistakes of its predecessor.

Boosting – an informal description

1. Get a dataset.
2. Take a bootstrap, and train a model on it.
3. See which examples the model got **wrong**.
4. Upweight those 'hard' examples, downweight the 'easy' ones.
5. Go back to step 2, but with a *weighted* bootstrap.

Each new member focuses on examples that the previous ones got wrong!

If you want a committee of M members, allow this to loop M times.
When a new testing datapoint arrives, vote!

Boosting

Input: Training set $T = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_N, y_N)\}$, where $y_i \in \{-1, +1\}$

Define a uniform distribution $D_1(i)$ over elements of T.

for $j = 1$ to M **do**
  Train a model h_j using a dataset sampled from T using distribution D_j.
  Calculate $\epsilon_j = \sum_{i \in T} \delta(h_j(\mathbf{x}_i) \neq y_i)$
  If $\epsilon_j \geq 0.5$ break
  Set $\alpha_j = \frac{1}{2} \ln \left(\frac{1-\epsilon_j}{\epsilon_j} \right)$
  Update $D_{j+1}(i) = \frac{D_j(i) \exp(-\alpha_j y_i h_j(\mathbf{x}_i))}{Z_j}$
  where Z_j is a normalization factor so that D_{j+1} is a valid distribution.
end for

For a new testing point $(\mathbf{x}', y')$, we take a weighted majority vote, which can be implemented as, $H(x') = sign\left(\sum_{j=1}^{M} \alpha_j h_j(\mathbf{x}') \right)$

Ensemble Methods: Summary

- Treats models as a committee
- Two main types – dependent and independent

§(boosting and bagging are two examples)

- Responsible for major advances in industrial ML

  Random Forests = Kinect

  Boosting = Face Recognition in Phone Cameras

- Tend to work extremely well “off-the-shelf” – no parameter tuning.

Finally... g’bye!

This is my last lecture.
I’ll be back nearer to Xmas for a revision lecture.

After reading week: Dr Ke Chen.

Enjoy reading week!