More Supervised Learning

Week 9

Decision trees
Random forest

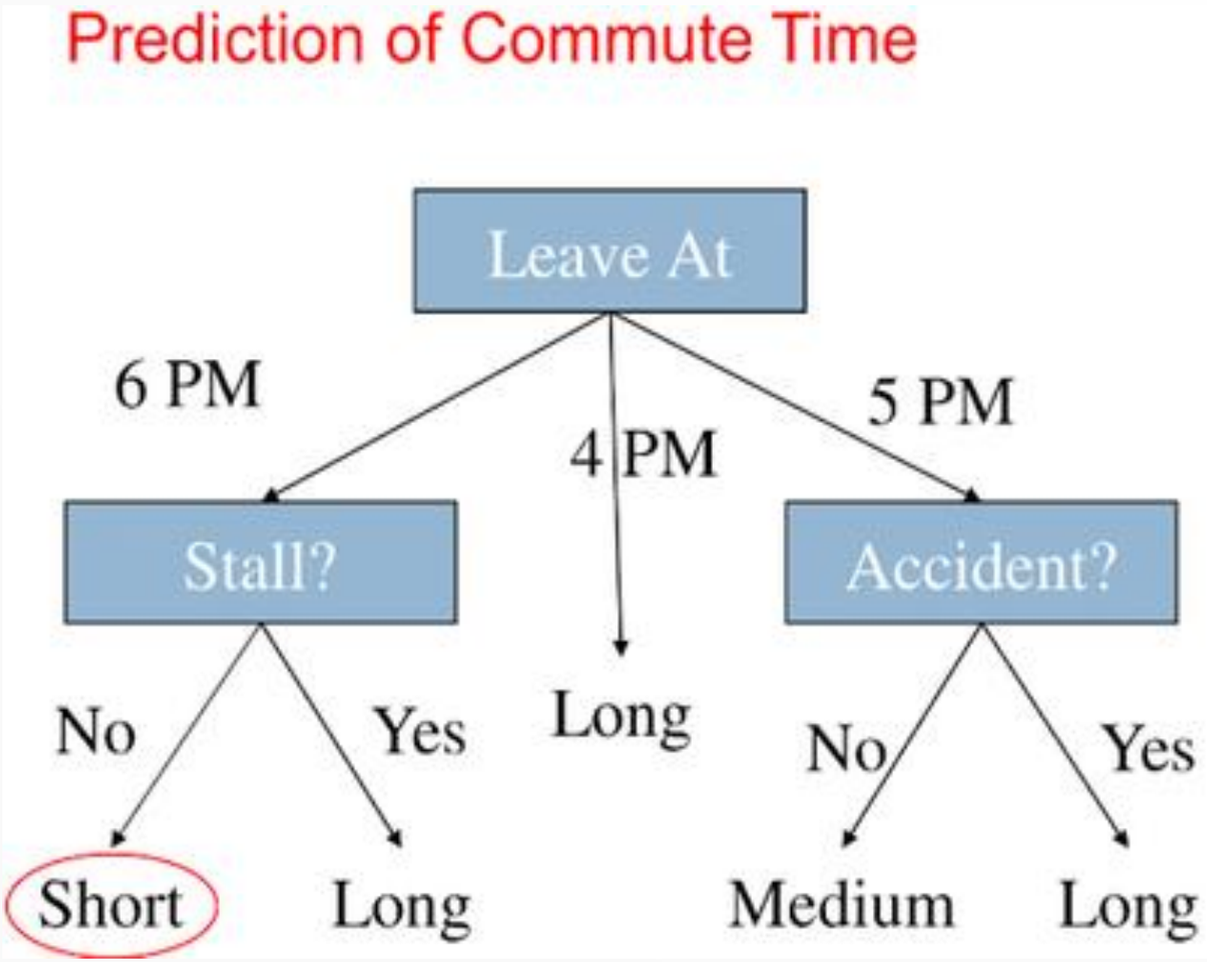
Revisiting the use of Training and Test Samples

- What is the reason behind dividing data into train and test set?
- What is generalisation?
- When to use training samples?
- When to use test samples?
- What is data leakage?

Decision Trees

Regression/classification trees
Decision tree algorithms
Model complexity & pruning

Decision Trees



If we leave at 6 PM and there are no cars stalled on the road, what will our commute time be?

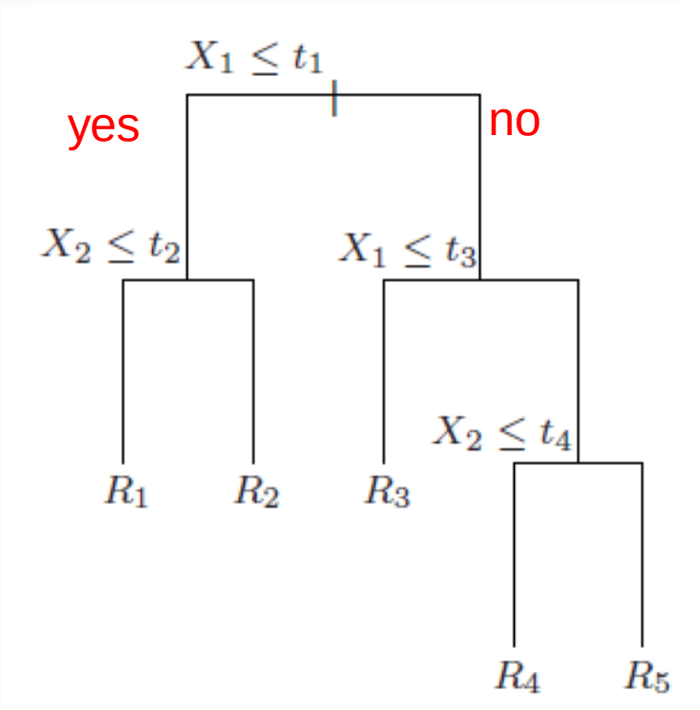
Decision Trees

- A decision tree is a map of the possible outcomes of a series of related options.
- It weighs possible actions against one another
  - Costs, probabilities, and benefits
- Typically starts with a single node
  - Branches into possible outcomes.

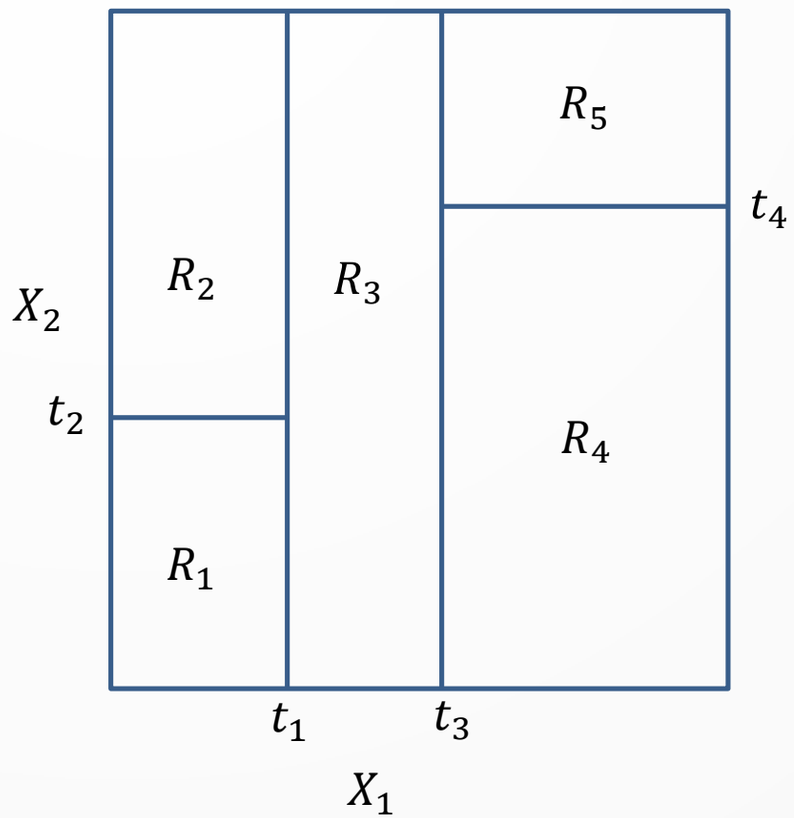
Decision Trees

- Simple and easy to interpret!
- May not be competitive with the best supervised learning algorithms in terms of prediction accuracy.
- We will also discuss random forests. A random forest uses multiple decision trees (ensemble) and combines them to yield a single prediction.
- Combining a large number of trees can often result in dramatic improvements in prediction accuracy, although at the expense of some loss in interpretation.

Partition of Feature Space



[Figure taken from Hastie's ESL book]



Decision Trees

- After partitioning the feature space, we can fit a simple model in each sub-region (R_1, R_2 …).
- We can fit a regression model. Such decision trees are called regression trees.
- We can also fit a classification model. Such decision trees are called classification trees.
- Usually, extremely simple models such as majority (classification) or mean (regression) are used.

Process of building a DT

- Let's start with the procedure:
  - We divide the feature space, i.e., the set of possible values for $x_1, \ldots, x_d$ into J distinct and non-overlapping regions, $R_1, \ldots, RJ$
  - For every instance that falls into region R_j, we make the same prediction, which is simply the mean (or mode) of response values for the training observations in R_j.

Formulation of Regression Trees

- The overall goal of regression trees is to find regions $R_1, R_2, \ldots, RJ$ that minimize the training error:

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$

where $\hat{y}_{R_j}$ is the mean of the target values of the training instances in the j^{th} region.

But how do we exactly perform these actions?

Solution

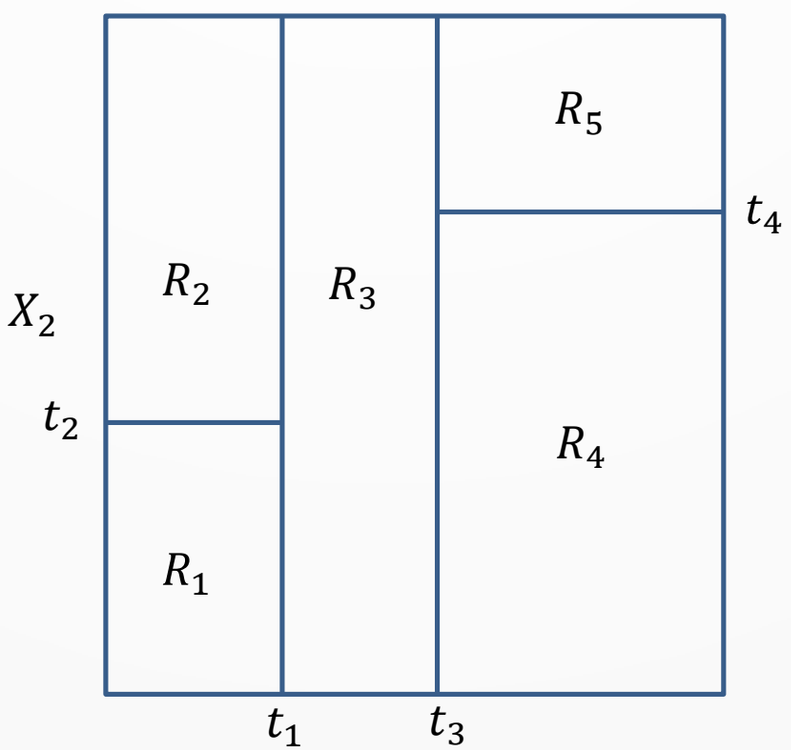
- It is computationally infeasible to consider every possible partition of the feature space into J regions.
- For this reason, a top-down, greedy approach is used
  - Known as recursive binary splitting.
- Rather using a brute-force solution, we would like to work in a heuristic way.

Solution

- How the heuristic method works?
  - Select a feature x_j and a threshold s such that
    - Split the feature space
      - Regions $\{x|x_j \leq s\}$ and $\{x|x_j \geq s\}$
        - leads to the best possible reduction in training error
  - Not going into the joint space of all features
    - Use independent feature form such as x_j with a threshold s .
  - Repeat the process
    - Looking for the best feature and the best threshold
      - Minimize the error in each of the resulting regions.
    - instead of splitting the entire feature space, we only split one of the two previously identified regions.
    - The splitting process continues until a stopping criterion is reached.

Prediction using DT

- We predict the response for a given test instance
  - using the mean (or mode) of the training instances in the region where the test observation falls.



Classification trees…

- In the classification setting
  - we replace the sum of square error by the classification error rate as a criterion for making the binary splits.
  - The classification error rate (E) is defined as the fraction of the training instances in that region that do not belong to the most common class.

$$E = 1 - \max_k \hat{p}_{jk}$$

$where\ \hat{p}_{jk}$ represents the proportion (fraction) of training instances in the j^{th} region that are from k^{th} class

$$CoD = \max_k \hat{p}_{jk}$$

- CoD (certainty of distribution) and close to 1
  - almost all of the training points inside a region are voting for a certain class label.

Classification trees…

- Classification error is being less sensitive for tree-growing
- Alternative solution:
  - Gini index (G) is defined as

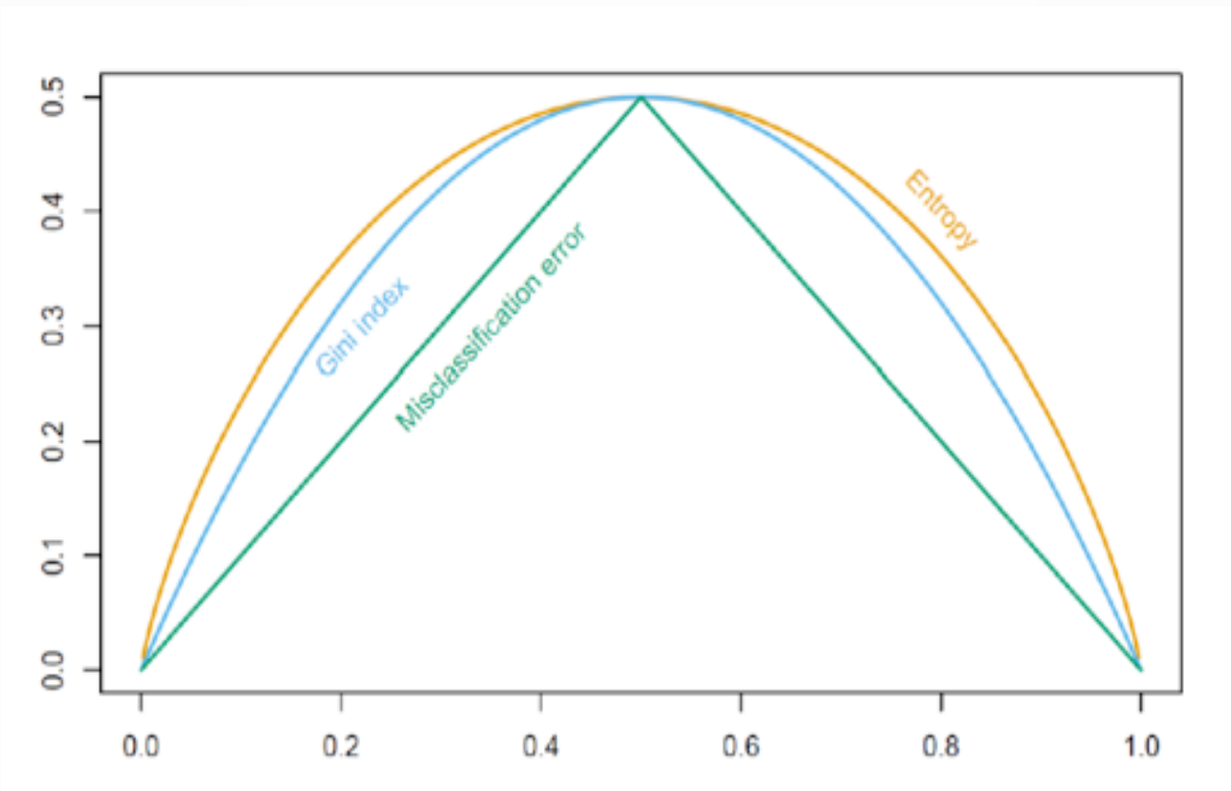
$$G = \sum_{k=1}^{K} \hat{p}_{jk}(1 - \hat{p}_{jk})$$

    - It is a measure of node purity. G becomes small as $\hat{p}_{jk}$ closes to either 0 or 1.
  - Entropy defined as:

$$D = -\sum_{k=1}^{K} \hat{p}_{jk} \log \hat{p}_{jk}$$

Classification trees…

- Pattern of Error, Gini Index and Entropy for different probabilities of class distributions:



Decision tree algorithms

- Three of the more popular ones are listed:
  - ID3 (Iterative Dichotomiser 3)
    - uses Entropy
  - *C4.5 (Successor of ID3)*
    - *slightly more advanced version of ID3 and also uses Entropy*
  - *CART (Classification and Regression Tree)*
    - *uses Gini impurity*

Decision tree algorithms: ID3 Algorithm...

- Calculate the entropy of every feature using the data set S.
- Split the set S into subsets using the feature for which entropy is minimum.
  - So lesser values of entropy means it should be a good choice for selection of the attribute
- Make a decision tree node containing that feature.
- Recurse on subsets using remaining features.

Decision tree algorithms: ID3 Algorithm…

- If the tree is very deep
  - It partitions the feature space into small regions.
    - Small number of training points in sub-regions.
      - Increases variance and estimation becomes poor
- If the tree is shallow
  - Large regions
    - Small variance but large bias
- Need to find the sweet point
  - Depth of the decision tree
  - Cross-validation as discussed earlier (previous lecture)

Model complexity and pruning

- Pruning is a technique that reduces the size of decision trees
  - Removes sections of the tree
    - Little power to classify instances.
- The tree-building process
  - Overfit (creating deep trees)
  - Underfit (creating small number of regions)
- Generally there are several ways of pruning trees:
  - Pre-pruning (forward pruning)
  - Post-pruning (backward pruning)

Model complexity and pruning: Pre-pruning

- In pre-pruning
  - Decision is made during the building process
    - Stop adding nodes (e.g. by looking at entropy).
- In case of Entropy
  - Check the amount of entropy reduction by selecting different features.
  - Stop splitting when the entropy reduction is not significant.
- Pre-pruning can be problematic
  - Sometimes attributes individually do not contribute much to a decision, but combined, they may have a significant impact.

Model complexity and pruning: Post-pruning

- Post-pruning waits until the full decision tree has been built
  - Then prunes the attributes by subtree replacement.
- Replace an entire subtree with a single region or node
  - It reproduces the smallest error.
- Select a subtree
  - Check – replacing it with a single node or feature incurs a small amount of change in Entropy.
  - If yes, trim the tree. If not, keep that subtree

Decision trees : Advantages/Disadvantages

- Advantages:
  - Decision trees are very easy to understand
  - Decision trees are capable of modelling nonlinear functions.
  - Decision tree can handle categorical variable
- Disadvantages:
  - Sensitive to small changes in the data.
    - Adding few data points or change some small values will change the DT
  - May overfit easily
    - Deep decision trees increases risk of overfitting and high variance model.
  - Only axis-aligned splits.
    - Considers each feature independently
    - No joint probabilities of features
  - Performance is not competitive
    - SVM, KNN or Neural network

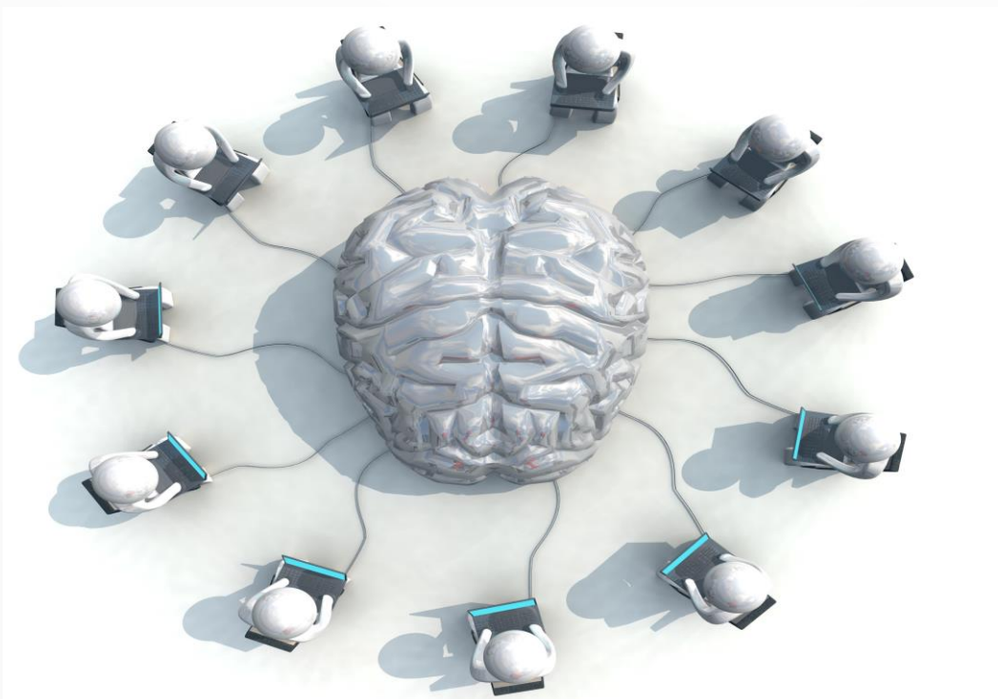
Random Forest

Ensemble Learning
Bootstrap Estimation
Bagging
RF algorithm

Wisdom of Crowds

The collective knowledge of a **diverse and independent** body of people typically exceeds the knowledge of any single individual, and can be harnessed by voting.

- James Surowiecki (2004)



Ensemble learning

- Probable outcomes of developing machine learning models
  - Sometimes weak and inaccurate
  - Some performs better on specific occasions.
- Ensemble learning
  - Generating and combining multiple models (classifiers or experts) to solve a particular computational intelligence problem.

Ensemble learning...

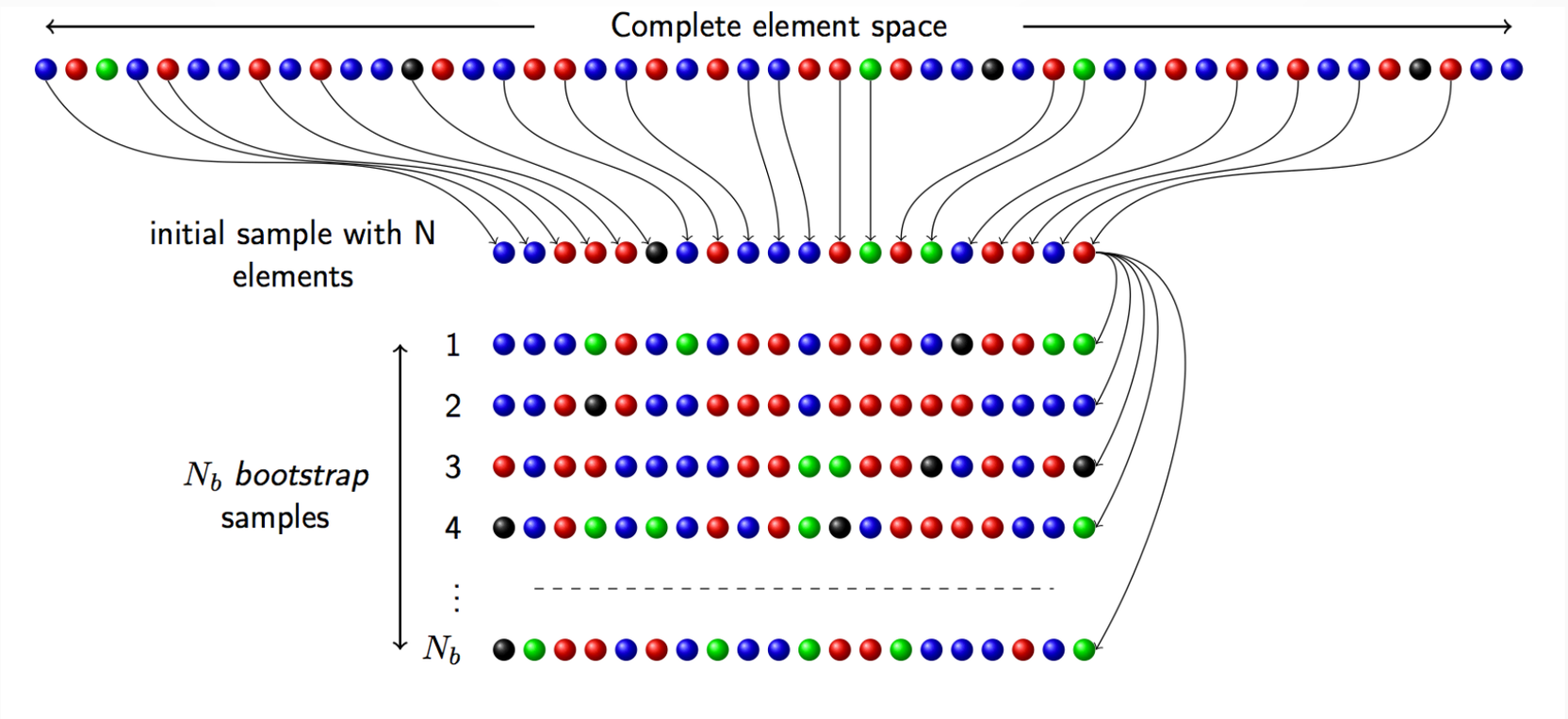
- Consider this scenario:
  - We know that a single decision tree might not perform well
    - But, it is super fast
  - What if we create multiple trees?
    - We just have to make sure that they do not all learn the same thing.

Ensemble learning…

- Problem with single decision tree
  - Risk of overfitting or increased variance
- To reduce the variance of unstable learning methods (such as DT) use ensemble method
  - Train multiple decision trees, each with slightly different subsets of data
  - take their combined decision
    - Classification
      - Voting
    - Regression
      - Averaging
- Random Forest
  - Ensemble of DT

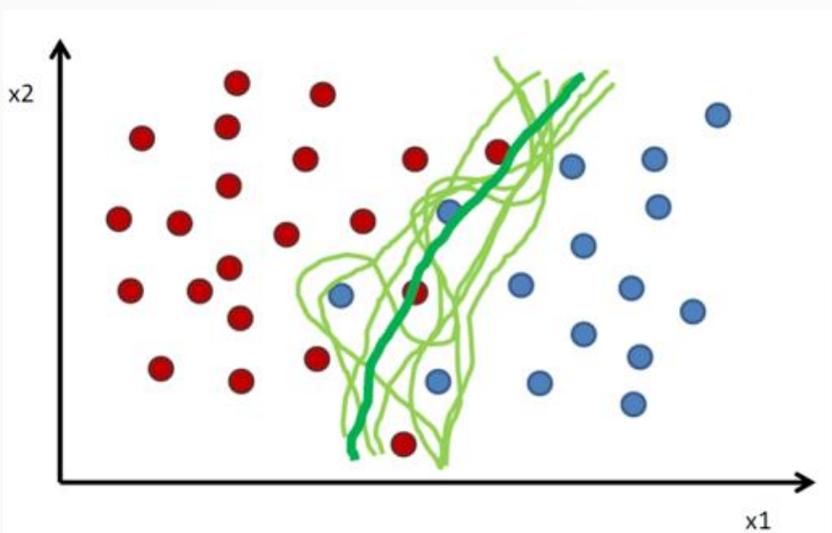
Bootstrap estimation

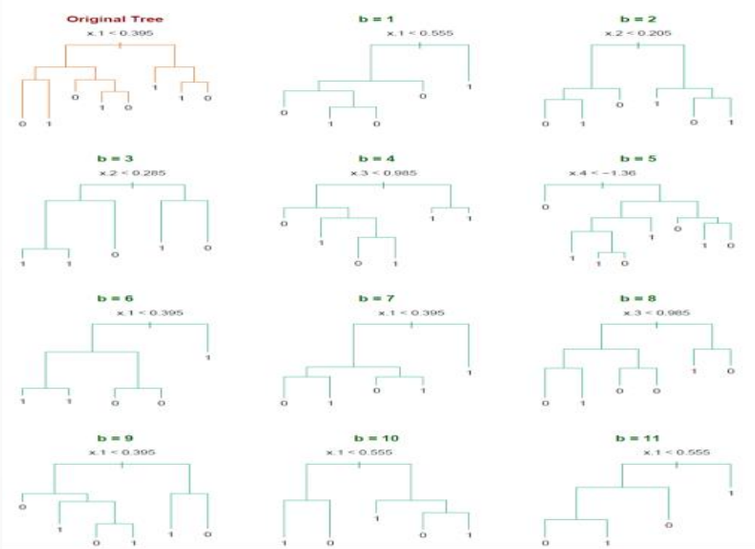
- A bootstrap sample is a smaller sample that is generated (bootstrapped) from a larger sample.
- It uses a resampling method found in statistics.
- Bootstrap in many cases can result in less variance and more accurate results.



Bagging

- Bootstrap aggregation or bagging
  - General-purpose procedure for reducing the variance of a statistical learning method.
    - Given a set of n independent estimates $Z_1, Z_2, \ldots, Z_n$ each with variance σ^2, the variance of their mean is n–times lower (σ^2/n).
      - When the estimates are not independent, reduction in variance is lower.
  - Uses multiple classifiers trained on different under-sampled subsets and then allow these classifiers to vote on a final decision.





Random forest algorithm

- Random forest classifier
  - Creates a set of decision trees from randomly selected subset of training set.
    - Each tree is built from a bootstrap sample of data
    - Form the tree based on the best feature from the subset
    - Repeat these steps T times, where T is the number of the trees
    - Impact on bias
      - Increases. Why?
        - Uses subsets of features in different independent trees
  - Aggregates the votes from different decision trees to decide the final class of the test object.

Random forest algorithm...

- In random forest:
  - all trees are fully grown and no pruning
  - we are dealing with two parameters:
    - number of trees (T);
      - Increasing the number can result in overfitting problem.
    - number of features (m_{try})

$$m_{try} = \sqrt{Number\ of\ \ features}$$

Random forest algorithm… Training

- For each of T iterations (T is the number of trees you may like to build):
  - Select a new bootstrap sample from the training set
  - Build an un-pruned tree on this bootstrap sample
  - At each internal node of the tree, randomly select m_{try} features and determine the best split using only these features.
    - Increasing number of features (m_{try}) for each split:
      - Increases correlation
      - Increases strength of single trees

Random forest algorithm…
Testing

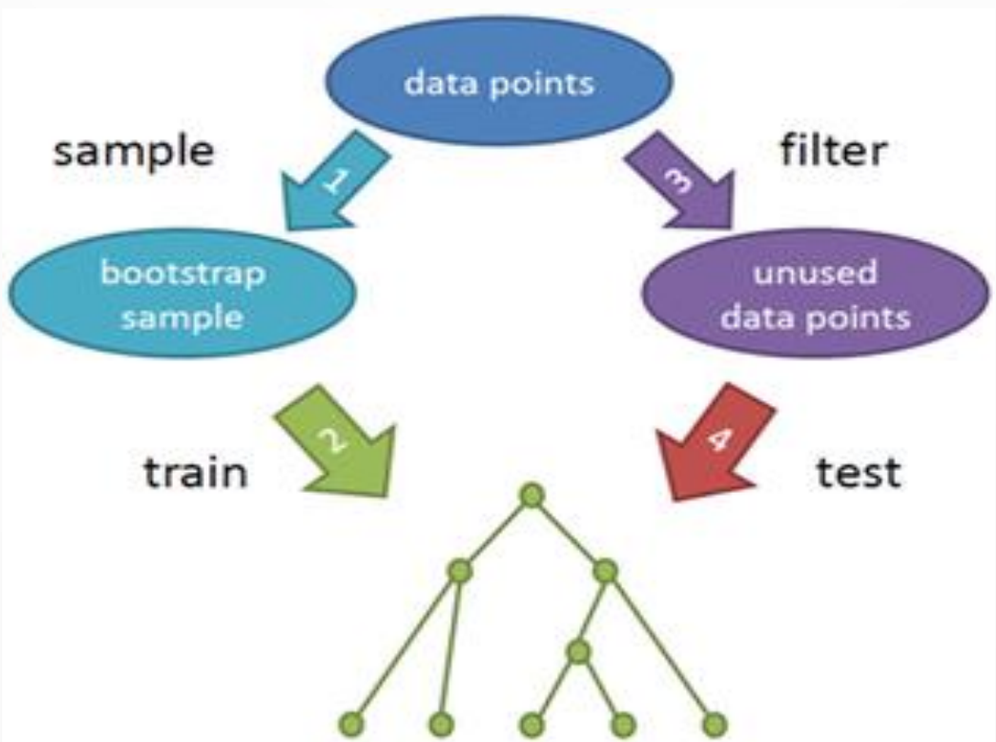
- Output overall prediction as a mean (or majority vote) from all individually trained trees.
- In random forest, the error rate depends on:
  - Correlation between trees (lower is better)
  - Strength of single trees (higher is better)

Out of bag error

- Estimate the goodness of fit of a bagged model
  - Out of Bag has been introduced which is equivalent to validation or test data.
- Each tree in a random forest is trained on a bootstrapped sample.
  - On average, each bagged tree makes use of 2/3 of the training instances.
- The remaining 1/3 of the instances are referred to as the out-of-bag (OOB) instances.

Out of bag error

- We can predict the response for the i^{th} observation using each of the trees in which that observation was OOB.
  - Average them to find the out of bag error



Feature importance

- Each node in the tree (single feature based split)
  - How much each feature decreases the weighted impurity (Gini or Entropy) of the tree
  - This provides rank of all features used in a tree
- In Random Forest
  - Multiple trees
    - Multiple rank values of single feature
  - Average impurity decreasing scores across all trees for getting overall score and rank of the feature

Advantages/Disadvantages of Random Forest

- Fast to build and even faster to predict!
  - Fully parallelizable since you can parallelly run the trees to go even faster!
  - Decision Tree complexity is $O(d \times n \times logn)$.
  - A random forest with T trees would have $O(T \times d \times n \times logn)$, where d is the number of features and n is the number of data points
- Ability to handle data without pre-processing
  - Not always required to normalize your dataset before running this method
  - data does not need to be rescaled, transformed, or modified! (Resistant to outliers)
  - automatic handling of missing values (a property of decision trees)
- Less interpretable results than a single decision tree.

Thank You.