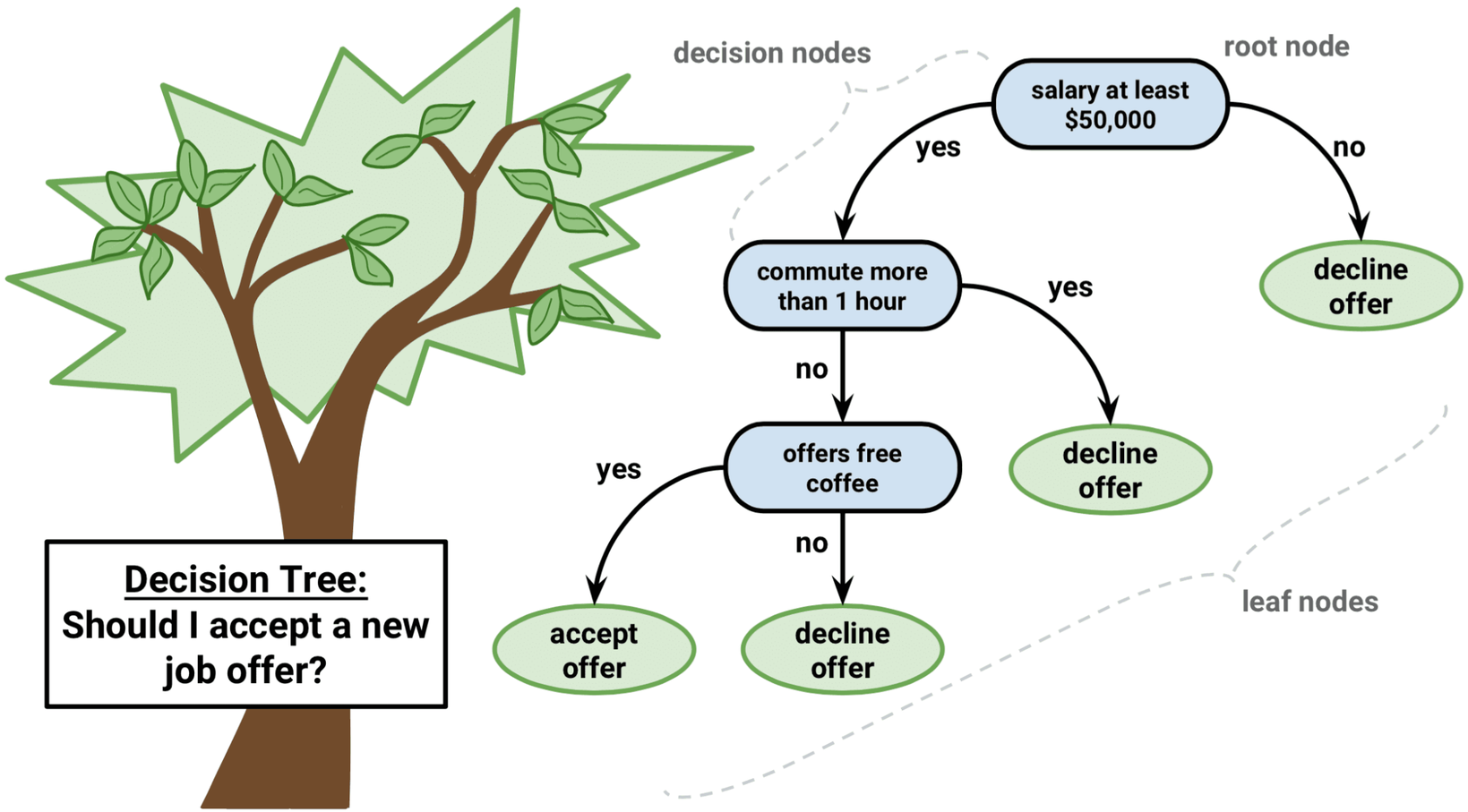
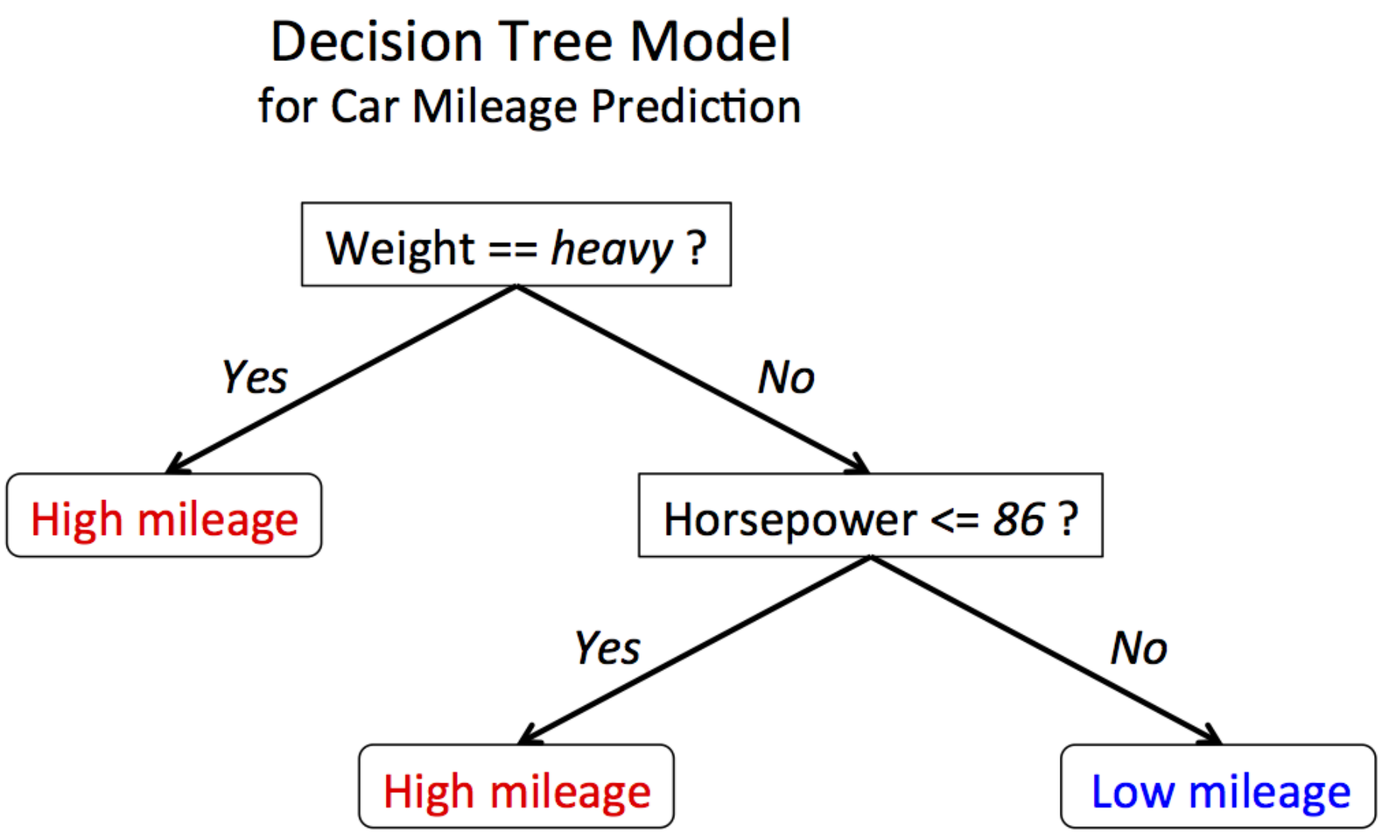
Trees and Random Forest

- Kumar Sundram

Example 1: Salary



Example 2: Car Mileage



Decision Trees

Why Decision Tree?

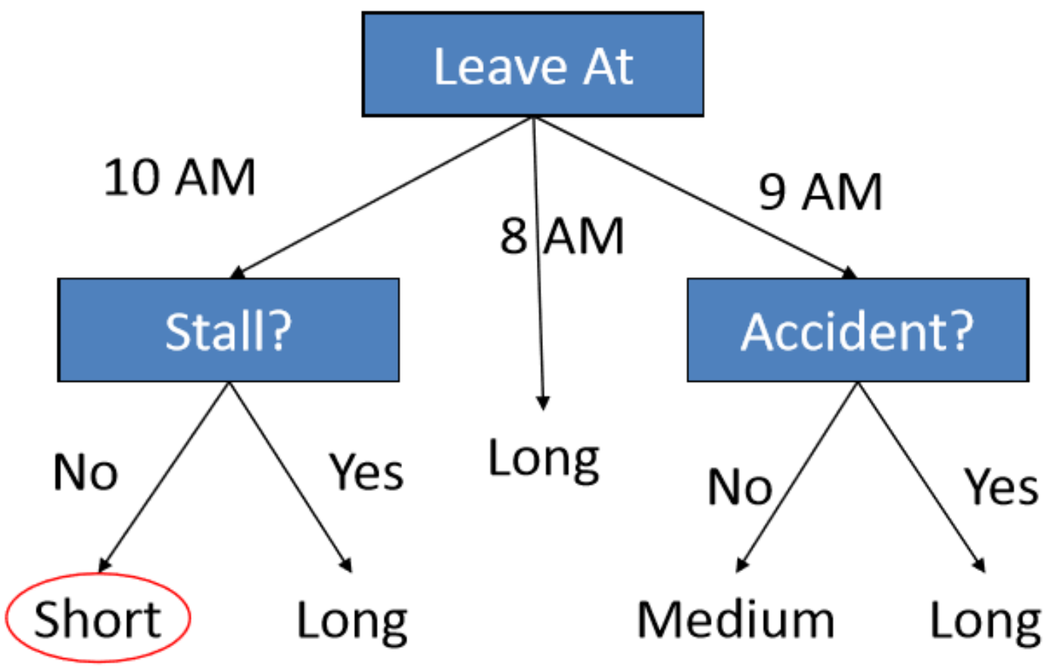
- Decision trees are powerful and popular tools for classification and prediction.
- Decision trees represent rules, which can be understood by humans
- Relatively fast compared to other classification models
- Obtain similar and sometimes better accuracy compared to other models

Key Requirements

- Attribute-value description
- Predefined classes (target values): the target function has discrete output values (Boolean or multiclass)
- Sufficient data: enough training cases should be provided to learn the model

Example

- Predicting Commute Time



If we leave at 10 AM and there are no cars stalled on the road, what will our commute time be?

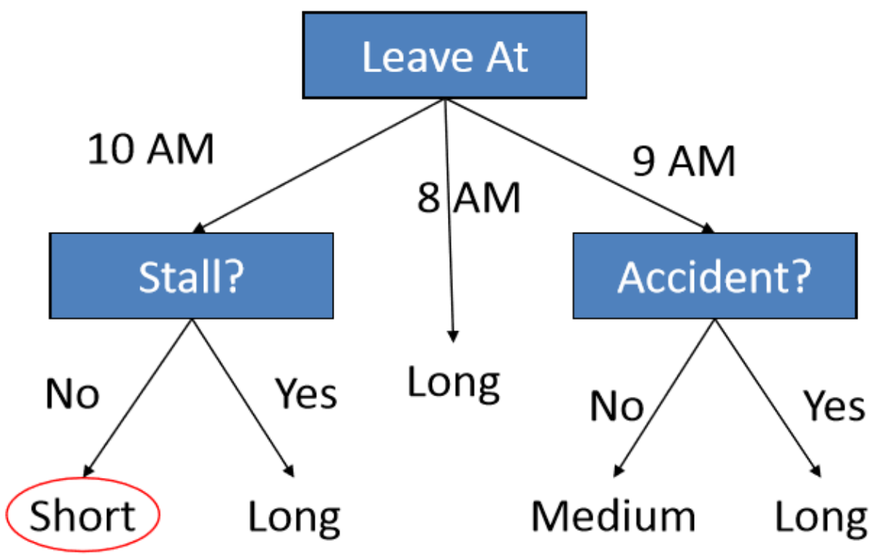
Inductive Learning

- In this decision tree, we make a series of Boolean decisions and follow the corresponding branch
  - Did we leave at 10 AM?
  - Did a car stall on the road?
  - Is there an accident on the road?
- By answering each of these yes/no questions, we then come to a conclusion on how long our commute might take

Splitting

Identifying the Best Attributes

- How did we decide to split on leave at and then on stall and accident and not weather?



Decision Tree Algorithms

- The basic idea behind any decision tree algorithm is as follows:
  - Choose the best attribute(s) to split the remaining instances and make that attribute a decision node
  - Repeat this process for recursively for each child
  - Stop when:
    - All the instances have the same target attribute value
    - There are no more attributes
    - There are no more instances
- ***Choose the best attribute to split***

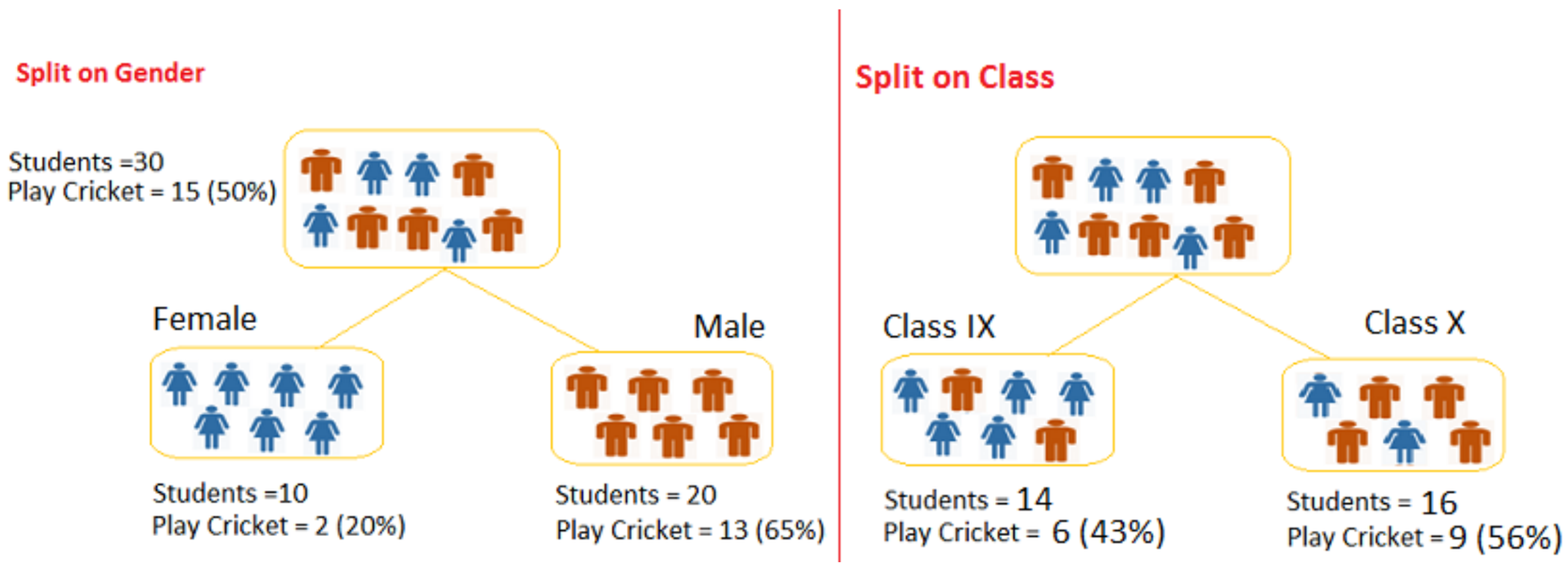
How does a tree decide where to split?

- Gini Index
- Chi-Square
- Information Gain
- Reduction in Variance

How does a tree decide where to split?

- **Gini Index**
  - It works with categorical target variable “Success” or “Failure”
  - It performs only Binary splits
  - Higher the value of Gini higher the homogeneity
  - CART (Classification and Regression Tree) uses Gini method to create binary splits

Split by Gini Index



Split by Gini Index

- **Split on Gender:**
  - Calculate, Gini for sub-node Female = (0.2)*(0.2)+(0.8)*(0.8)=0.68
  - Gini for sub-node Male = (0.65)*(0.65)+(0.35)*(0.35)=0.55
  - Calculate weighted Gini for Split Gender = (10/30)*0.68+(20/30)*0.55 = **0.59**
- **Similar for Split on Class:**
  - Gini for sub-node Class IX = (0.43)*(0.43)+(0.57)*(0.57)=0.51
  - Gini for sub-node Class X = (0.56)*(0.56)+(0.44)*(0.44)=0.51
  - Calculate weighted Gini for Split Class = (14/30)*0.51+(16/30)*0.51 = **0.51**
- Above, you can see that Gini score for *Split on Gender* is higher than *Split on Class,* hence, the node split will take place on Gender.

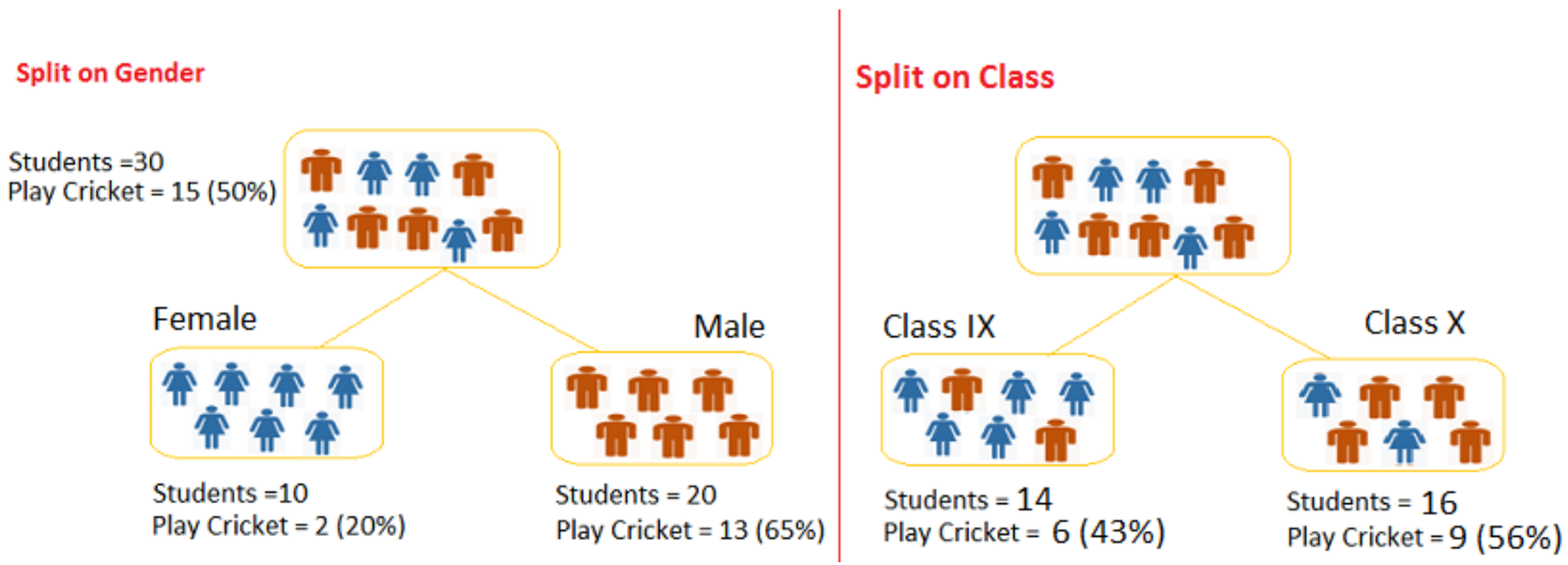
How does a tree decide where to split?

- **Information Gain = 1 – Entropy**

$$\text{Entropy} = -p \log_2 p - q \log_2 q$$

  - Here, p and q is probability of success and failure respectively in that node.
  - Entropy is also used with categorical target variable.
  - It chooses the split which has lowest entropy compared to parent node and other splits.
  - The lesser the entropy, the better it is.
  - Calculate entropy of parent node
  - Calculate entropy of each individual node of split and calculate weighted average of all sub-nodes available in split.

Split by Information Gain



Split by Information Gain

- Entropy for Female node = -(2/10) log2 (2/10) – (8/10) log2 (8/10) = 0.72 and for male node, -(13/20) log2 (13/20) – (7/20) log2 (7/20) = **0.93**
- Entropy for split Gender = Weighted entropy of sub-nodes = (10/30)*0.72 + (20/30)*0.93 = **0.86**
- Entropy for Class IX node, -(6/14) log2 (6/14) – (8/14) log2 (8/14) = 0.99 and for Class X node, -(9/16) log2 (9/16) – (7/16) log2 (7/16) = 0.99.
- Entropy for split Class = (14/30)*0.99 + (16/30)*0.99 = **0.99**
- Above, you can see that entropy for *Split on Gender* is the lowest among all, so the tree will split on *Gender*.
- We can derive information gain from entropy as **1- Entropy.**

Overfitting and pruning

Preventing Overfitting

Setting constraints on tree size

- **Minimum samples for a node split**
  - Defines the minimum number of samples (or observations) which are required in a node to be considered for splitting.
- **Minimum samples for a terminal node (leaf)**
  - Defines the minimum samples (or observations) required in a terminal node or leaf.
- **Maximum depth of tree (vertical depth)**
  - Defines the maximum depth of a tree.
- **Maximum number of terminal nodes**
  - The maximum number of terminal nodes or leaves in a tree.

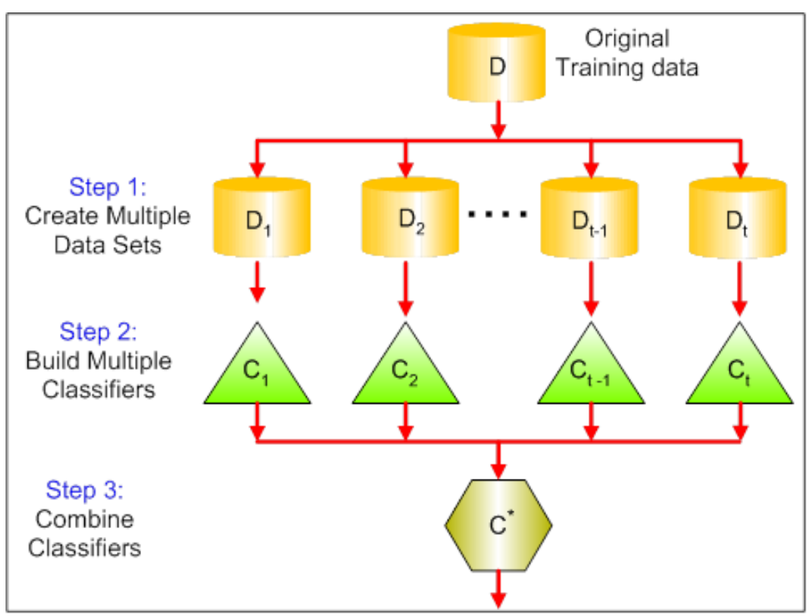
Ensemble Methods

What are ensemble methods in tree based modeling ?

- The literary meaning of word ‘ensemble’ is *group*. Ensemble methods involve group of predictive models to achieve a better accuracy and model stability. Ensemble methods are known to impart supreme boost to tree based models.
- Like every other model, a tree based model also suffers from the plague of bias and variance. Bias means, ‘how much on an average are the predicted values different from the actual value.’ Variance means, ‘how different will the predictions of the model be at the same point if different samples are taken from the same population’.
- You build a small tree and you will get a model with low variance and high bias. How do you manage to balance the trade off between bias and variance ?
- Normally, as you increase the complexity of your model, you will see a reduction in prediction error due to lower bias in the model. As you continue to make your model more complex, you end up over-fitting your model and your model will start suffering from high variance.
- A champion model should maintain a balance between these two types of errors. This is known as the **trade-off management** of bias-variance errors. Ensemble learning is one way to execute this trade off analysis.

Bagging

- One of the commonly used ensemble methods include is Bagging.
- Bagging is a technique used to reduce the variance of our predictions by combining the result of multiple classifiers modelled on different sub-samples of the same data set. The following figure will make it clearer:



Bagging

- The steps followed in bagging are:
  - **Create Multiple DataSets**:
    - Sampling is done *with replacement* on the original data and new datasets are formed.
    - The new data sets can have a fraction of the columns as well as rows, which are generally hyper-parameters in a bagging model
    - Taking row and column fractions less than 1 helps in making robust models, less prone to overfitting
  - **Build Multiple Classifiers:**
    - Classifiers are built on each data set.
    - Generally the same classifier is modelled on each data set and predictions are made.
  - **Combine Classifiers:**
    - The predictions of all the classifiers are combined using a mean, median or mode value depending on the problem at hand.
    - The combined values are generally more robust than a single model.

Bagging

- Higher number of models are always better or may give similar performance than lower numbers.
- There are various implementations of bagging models.
- Random forest is one of them and we'll discuss it next.

Random Forest

Random Forest - How does it work?

- Random Forest is considered to be a panacea of all data science problems. On a funny note, when you can’t think of any algorithm (irrespective of situation), use random forest!
- Each tree is planted & grown as follows:
  - Assume number of cases in the training set is N. Then, sample of these N cases is taken at random but with replacement. This sample will be the training set for growing the tree.
  - If there are M input variables, a number m<M is specified such that at each node, m variables are selected at random out of the M. The best split on these m is used to split the node. The value of m is held constant while we grow the forest.
  - Each tree is grown to the largest extent possible and there is no pruning.
  - Predict new data by aggregating the predictions of the ntree trees (i.e., majority votes for classification, average for regression).

Random Forest - How does it work?

RF – Advantages and Disadvantages

Advantages

- This algorithm can solve both type of problems i.e. classification and regression and does a decent estimation at both fronts.

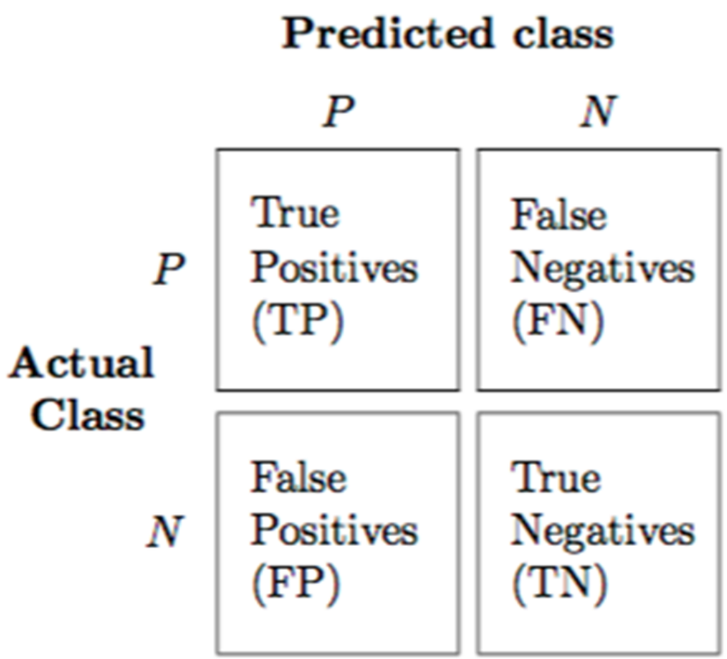
- One of benefits of Random forest which excites me most is, the power of handle large data set with higher dimensionality. It can handle thousands of input variables and identify most significant variables so it is considered as one of the dimensionality reduction methods. Further, the model outputs **Importance of variable,** which can be a very handy feature (on some random data set).

Disadvantages

- It surely does a good job at classification but not as good as for regression problem as it does not give precise continuous nature predictions. In case of regression, it doesn’t predict beyond the range in the training data, and that they may over-fit data sets that are particularly noisy.
- Random Forest can feel like a **black box approach** for statistical modelers – you have very little control on what the model does. You can at best – try different parameters and random seeds!

Model Validation

Accuracy Matrix



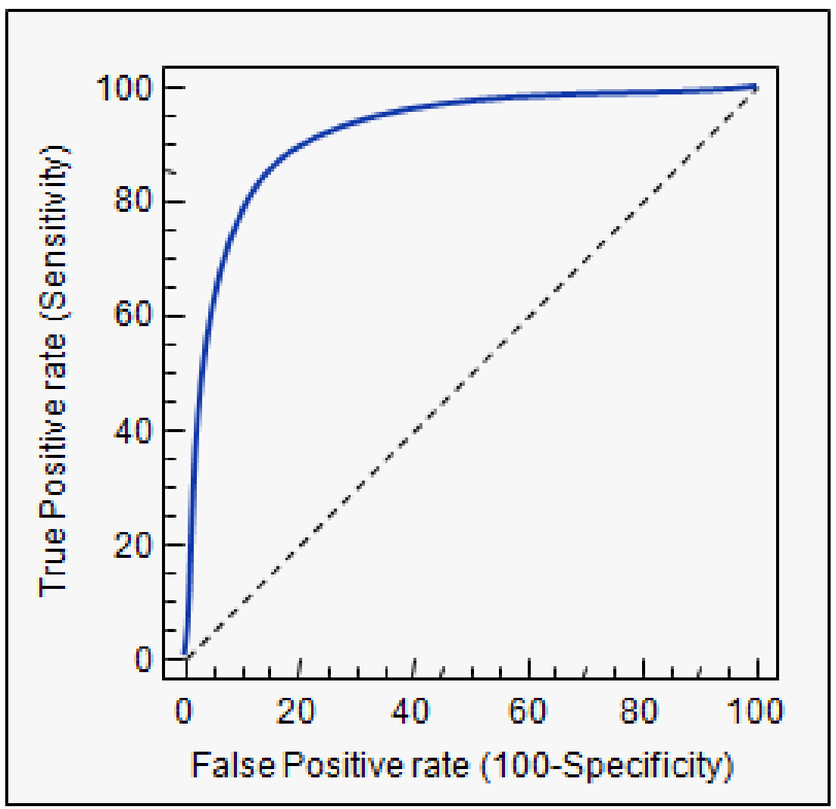
Sensitivity, recall, hit rate, or true positive rate (TPR)

$$\mathrm{TPR} = \frac{\mathrm{TP}}{P} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

Specificity or true negative rate (TNR)

$$\mathrm{TNR} = \frac{\mathrm{TN}}{N} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}$$

ROC - AUC


True Positive rate (Sensitivity)
100
80
60
40
20
0
0
20
40
60
80
100
False Positive rate (100-Specificity)