Trees and Random Forest

- Sonal Ghanshani

Decision Trees

Why Decision Tree?

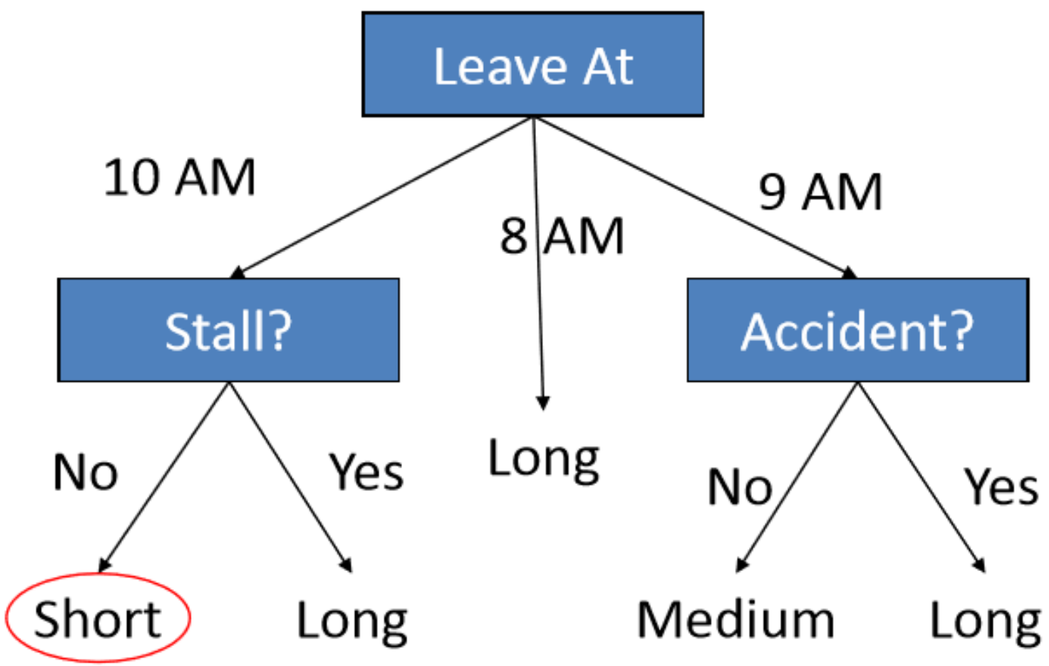
- Decision trees are powerful and popular tools for classification and prediction.
- Decision trees represent rules, which can be understood by humans
- Relatively fast compared to other classification models
- Obtain similar and sometimes better accuracy compared to other models

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?

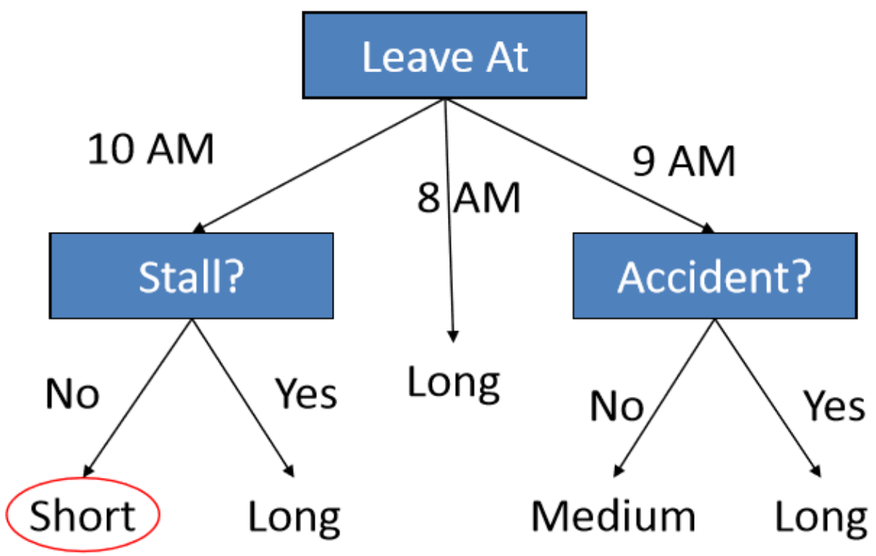
Inductive Learning

- 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***

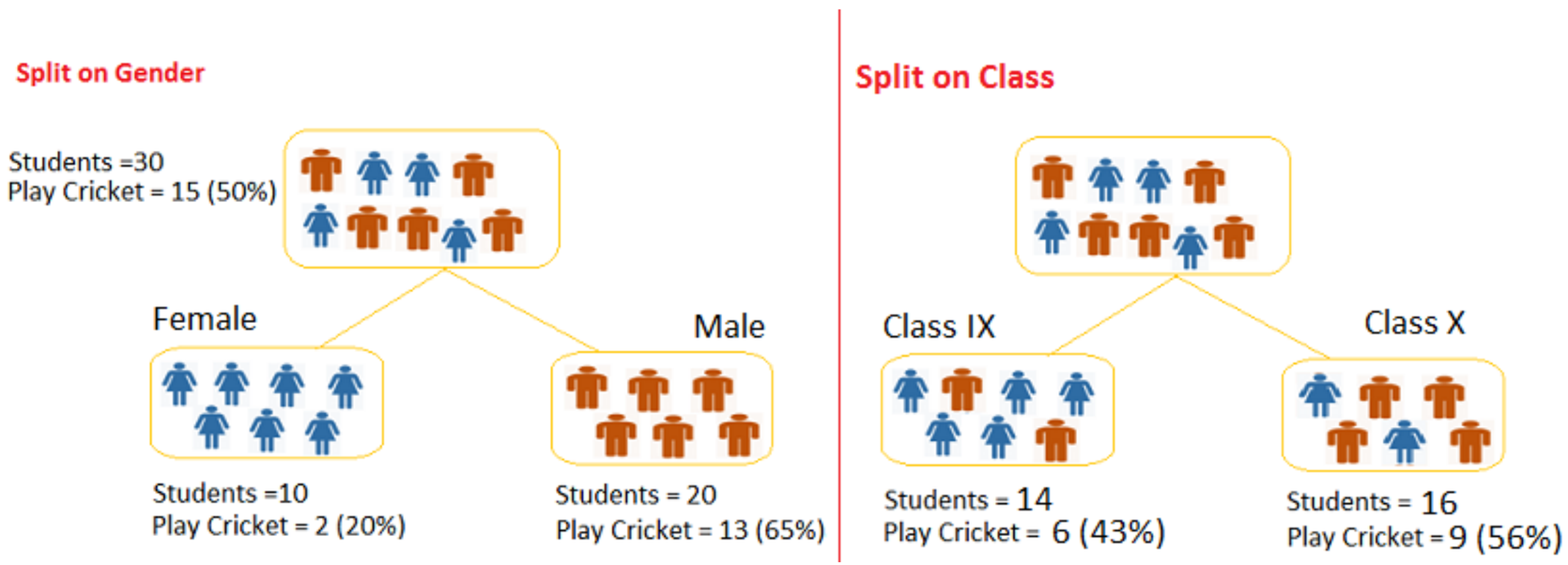
How does a tree decide where to split?

- 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?

- **Chi-Square**
  - It works with categorical target variable “Success” or “Failure”.
  - It can perform two or more splits.
  - Higher the value of Chi-Square higher the statistical significance of differences between sub-node and Parent node.
  - Chi-Square of each node is calculated using formula,
  - Chi-square = ((Actual – Expected)^2 / Expected)^1/2
  - It generates tree called CHAID (Chi-square Automatic Interaction Detector)

Split on Gender by Chi Square

Node	Play Cricket	Not Play Cricket	Total	Expected Play Cricket	Expected Not Play Cricket	Deviation Play Cricket	Deviation Not Play Cricket	Chi-Square	
								Play Cricket	Not Play Cricket
Female	2	8	10	5	5	-3	3	1.34	1.34
Male	13	7	20	10	10	3	-3	0.95	0.95
							Total Chi-Square	**4.58**	

- Calculate Chi-square of node for “**Play Cricket**” and “**Not Play Cricket**” using formula with formula, **= ((Actual – Expected)^2 / Expected)^1/2**.

Split on Class by Chi Square

Node	Play Cricket	Not Play Cricket	Total	Expected Play Cricket	Expected Not Play Cricket	Deviation Play Cricket	Deviation Not Play Cricket	Chi-Square	
								Play Cricket	Not Play Cricket
IX	6	8	14	7	7	-1	1	0.38	0.38
X	9	7	16	8	8	1	-1	0.35	0.35
							Total Chi-Square	1.46	

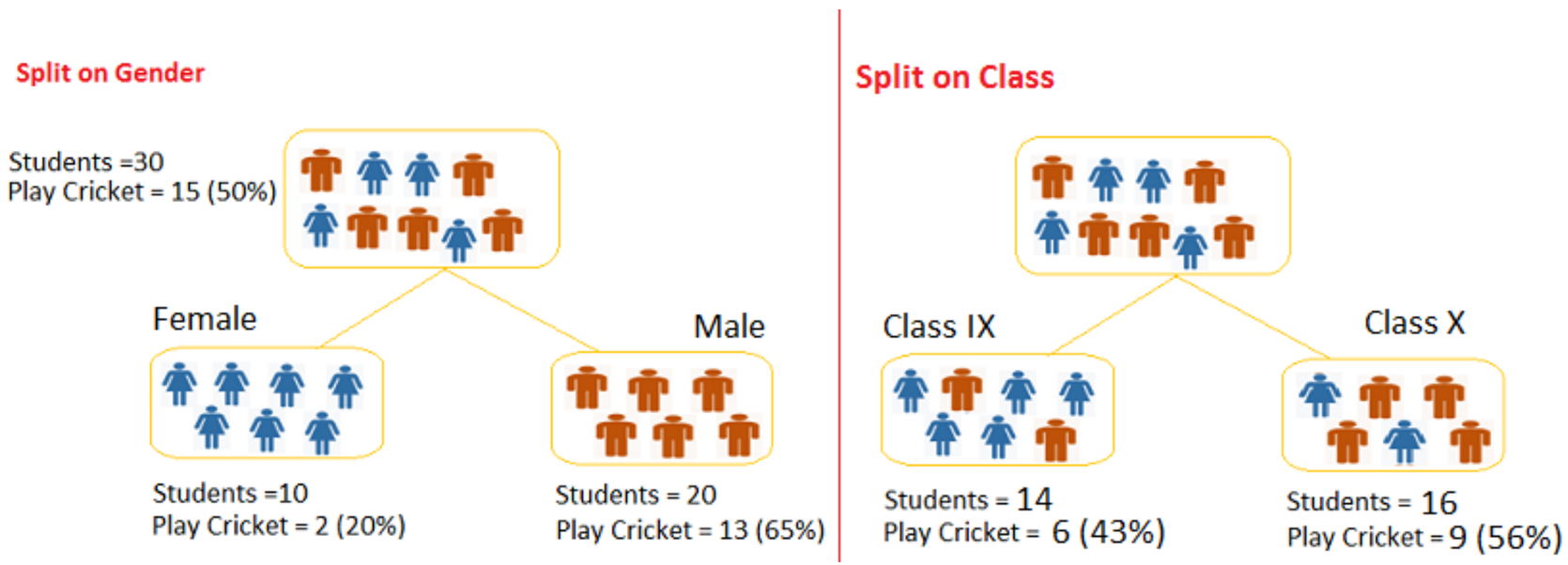
How does a tree decide where to split?

- **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.**

How does a tree decide where to split?

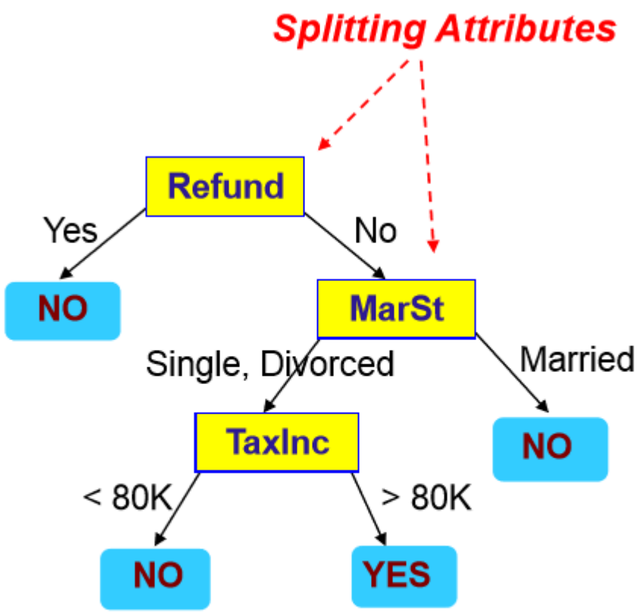
- Reduction in Variance
  - Calculate variance for each node

$$\text{Variance} = \frac{\Sigma(X - \overline{X})^2}{n}$$

Example of a Decision Tree

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data

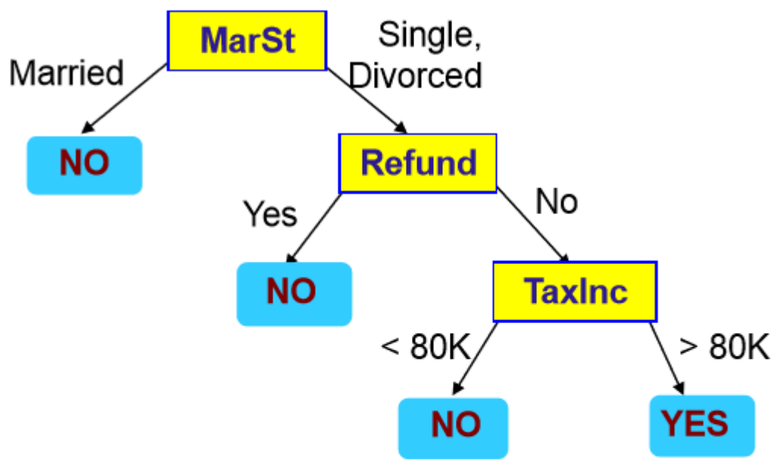


Model: Decision Tree

Example of a Decision Tree

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data



There could be more than one tree that fits the same data!

Types of Decision Trees

Types of Decision Trees

- Types of decision tree is based on the type of target variable we have. It can be of two types:
  - **Categorical Variable Decision Tree**: Decision Tree which has categorical target variable then it called as categorical variable decision tree. Example:- In above scenario of student problem, where the target variable was “Student will play cricket or not” i.e. YES or NO.
  - **Continuous Variable Decision Tree**: Decision Tree has continuous target variable then it is called as Continuous Variable Decision Tree.
- Example: Let’s say we have a problem to predict whether a customer will pay his renewal premium with an insurance company (yes/ no). Here we know that income of customer is a significant variable but insurance company does not have income details for all customers. Now, as we know this is an important variable, then we can build a decision tree to predict customer income based on occupation, product and various other variables. In this case, we are predicting values for continuous variable.

Regression Trees vs Classification Trees

- **Regression trees** are used when dependent variable is **continuous**. **Classification trees** are used when dependent variable is **categorical**.
- In case of **regression tree**, the value obtained by **terminal nodes** in the training data is the **mean response of observation** falling in that region. Thus, if an unseen data observation falls in that region, we'll make its prediction with mean value.
- In case of **classification tree**, the value (class) obtained by **terminal node** in the training data is the **mode of observations** falling in that region. Thus, if an unseen data observation falls in that region, we'll make its prediction with mode value.

Overfitting and pruning

Greedy Algorithm and Overfitting

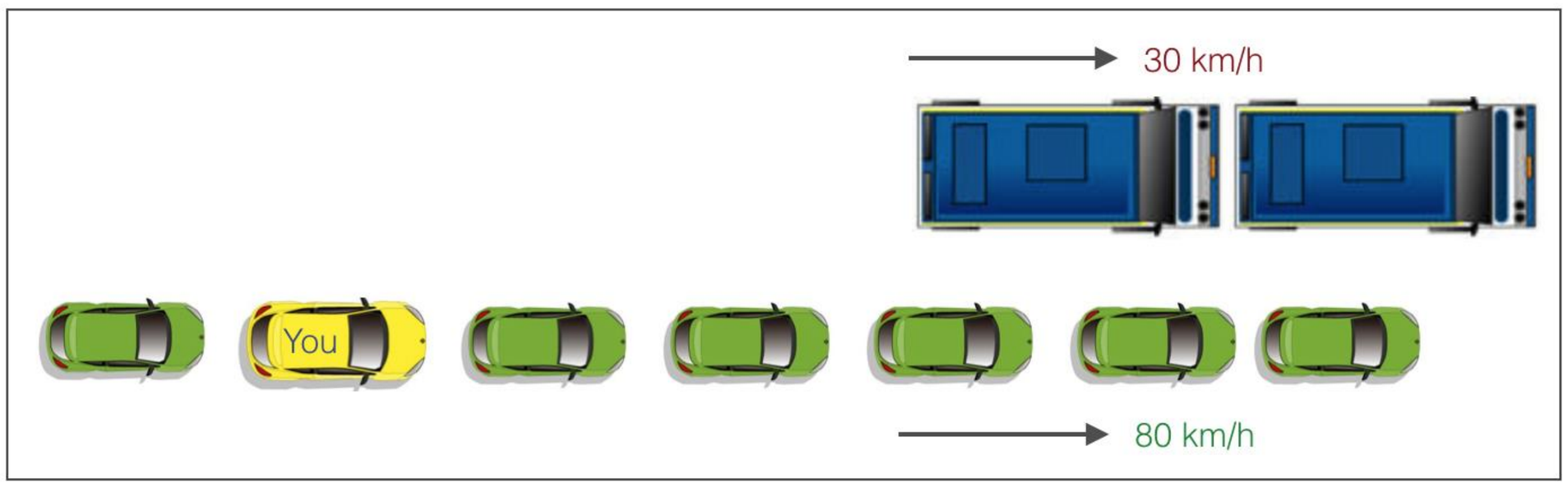
- Both the trees follow **a top-down greedy approach** known as recursive binary splitting. We call it as ‘top-down’ because it begins from the top of tree when all the observations are available in a single region and successively splits the predictor space into two new branches down the tree.
- It is known as ‘greedy’ because, the algorithm cares (looks for best variable available) about only the current split, and not about future splits which will lead to a better tree.
- **This splitting process is continued until a user defined stopping criteria is reached.** For example: we can tell the algorithm to stop once the number of observations per node becomes less than 50.
- In both the cases, the splitting process results in fully grown trees until the stopping criteria is reached. But, **the fully grown tree is likely to overfit data**, leading to poor accuracy on unseen data.
- This bring ‘pruning’. Pruning is one of the technique used tackle overfitting.
- We’ll learn more about it in following section.

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.

Tree Pruning



- This is exactly the difference between normal decision tree & pruning. A decision tree with constraints won’t see the truck ahead and adopt a greedy approach by taking a left.

Trees v/s Linear Models

Are tree based models better than linear models?

- If the relationship between dependent & independent variable is well approximated by a linear model, linear regression will outperform tree based model.
- If there is a high non-linearity & complex relationship between dependent & independent variables, a tree model will outperform a classical regression method.
- If you need to build a model which is easy to explain to people, a decision tree model will always do better than a linear model. Decision tree models are even simpler to interpret than linear regression!

Trees – Advantages and Disadvantages

Advantages

- **Easy to Understand**: Decision tree output is very easy to understand even for people from non-analytical background. It does not require any statistical knowledge to read and interpret them. Its graphical representation is very intuitive and users can easily relate their hypothesis.
- **Useful in Data exploration:** Decision tree is one of the fastest way to identify most significant variables and relation between two or more variables. It can also be used in data exploration stage. For example, we are working on a problem where we have information available in hundreds of variables, there decision tree will help to identify most significant variable.
- **Less data cleaning required:** It requires less data cleaning compared to some other modelling techniques. It is not influenced by outliers and missing values to a fair degree.
- **Data type is not a constraint:** It can handle both numerical and categorical variables.
- **Non Parametric Method:** Decision tree is considered to be a non-parametric method. This means that decision trees have no assumptions about the space distribution and the classifier structure.

Disadvantages

- **Over fitting:** Over fitting is one of the most practical difficulty for decision tree models. This problem gets solved by setting constraints on model parameters and pruning (discussed in detailed below).

- **Not fit for continuous variables**: While working with continuous numerical variables, decision tree looses information when it categorizes variables in different categories.

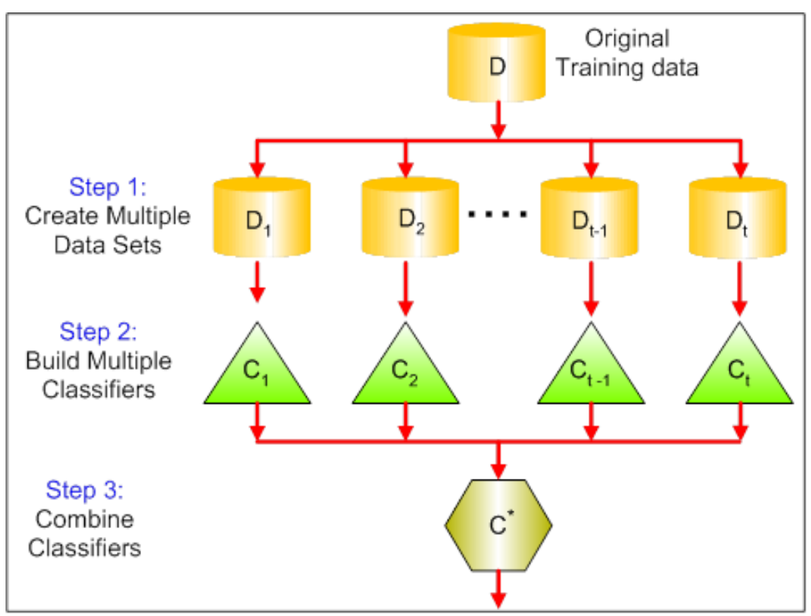
Ensemble Methods

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

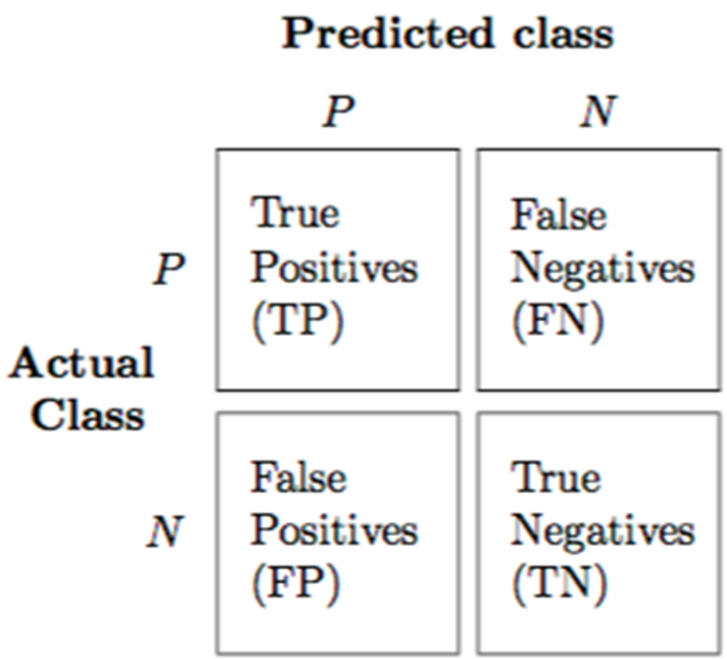
- This algorithm can solve both type of problems i.e. classification and regression and does a decent estimation at both fronts.
- One of benefits of Random forest which excites me most is, the power of handle large data set with higher dimensionality. It can handle thousands of input variables and identify 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).
- It has an effective method for estimating missing data and maintains accuracy when a large proportion of the data are missing. Categorical: Mode. Non categorical : Median

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



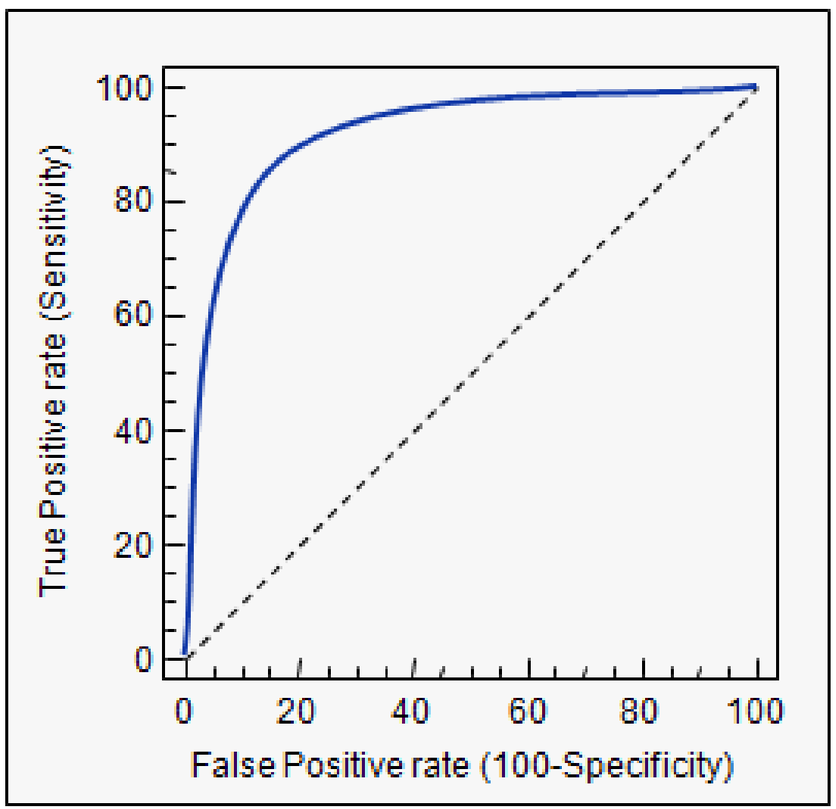
Sensitivity, recall, hit rate, or true positive rate (TPR)

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

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)