DECISION TREE AND RANDOM FOREST

What is Decision Tree

- Decision Trees (DTs) are a **non-parametric supervised learning method** used for classification and regression.
- The goal is to create a model that predicts the **value of a target variable by learning simple decision rules inferred from the data features**. A tree can be seen as a piecewise constant approximation.

Give an Example of Decision Tree

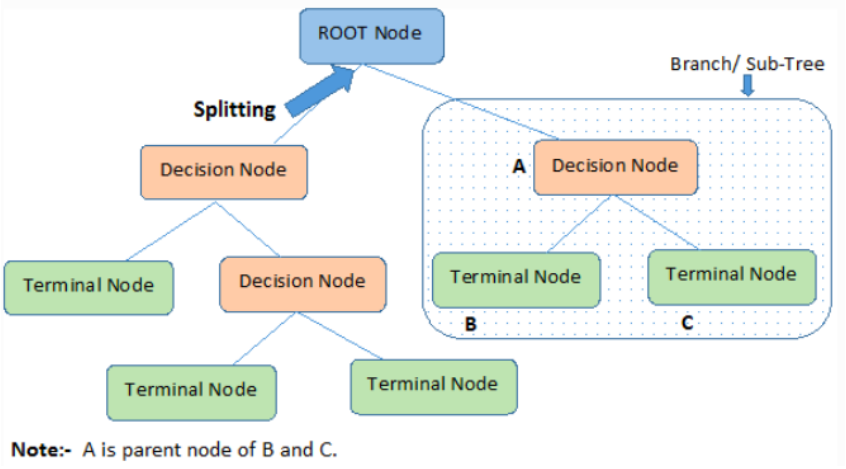
- Let's say we have a problem to predict whether a **customer will invest in Fixed Deposit (yes/ no).**
- Here we know that the income of customers is a significant variable, but the FD company does not have income details for all customers.
- Now, as we know this is an important variable, then we can build a decision tree to predict customer income based on occupation, product, and various other variables. In this case, we are predicting values for the categorical variable.

Types of Decision Trees

Types of decision trees are based on the type of target variable we have. It can be of two types:

1. **Categorical Variable Decision Tree**: Decision Tree which has a **categorical target variable**
2. **Continuous Variable Decision Tree:** Decision Tree has a **continuous target variable**

Overall structure of Decision Tree



Assumptions in Decision Tree –

- Normally, **non-parametric test (sometimes called a distribution free test) does not assume anything** about the underlying distribution.

How Decision Tree works

- In the **beginning**, the **whole training set** is considered **as the root**.
- **Feature values are preferred to be categorical**. If the **values are continuous** then they are **discretized** prior to building the model.
- Decision trees use multiple algorithms to decide to split a node into two or more sub-nodes.
- The **creation of sub-nodes increases the homogeneity of resultant sub-nodes.**
- The decision tree splits the nodes on all available variables and then **selects the split which results in most homogeneous sub-nodes.**

The algorithm selection is also based on the type of target variables. Few algorithms used in Decision Trees:

1. CART → (Classification and Regression Tree)
2. CHAID → (Chi-square automatic interaction detection Performs multi-level splits when computing classification trees)

What is Homogeneity.

The **more homogeneous the labels are in the dataset, the simpler your model** will be and simple the Decision Tree model is going to be.

- Always try to generate the partitions that result in homogeneous data points. For classification tasks, a **data set is completely homogeneous if it contains only a single class label**.
- If **all 100% of the data points belong to our class label, we call that dataset to be completely homogeneous**.
- The **ultimate aim of the Decision Tree splitting is to increase homogeneity.**
- More homogeneity means that most of the data points belong to same class and it will result in lesser errors.

Based on which attributes Decision Tree should choose the split (Homogeneity).

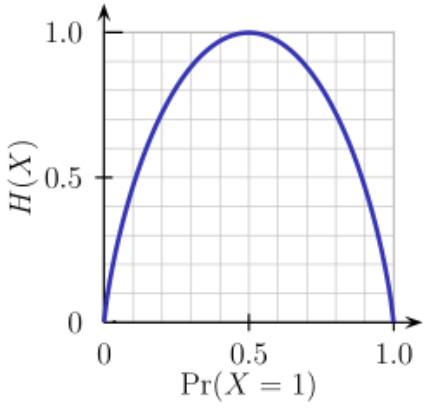
We normally check following criteria

1. **Entropy**
2. **Information gain** (using Information Gain as a criterion, we assume attributes to be categorical)
3. **Gini index** (for the Gini index, attributes are assumed to be continuous)
4. **Chi-Square**

The values are sorted, and attributes are placed in the tree by some order i.e, the attribute with a high value (in case of information gain) is placed at the root.

What is Entropy. **What is Entropy Curve.**

- Entropy is a **measure of the randomness in the information being processed**. The **higher the entropy, the harder it is to draw any conclusions from that information**.
- **Flipping a coin is an example of an action that provides information that is random.**



- From the graph, it is quite evident that the **entropy H(X) is zero when the probability is either 0 or 1.**
- The **Entropy is maximum when the probability is 0.5** because it projects perfect randomness in the data and there is no chance if perfectly determining the outcome.
- Normally, a **branch with an entropy of zero is a leaf node/terminal node** and A **branch with entropy more than zero needs further splitting**.

- Overall, Entropy is used for **calculating the purity of a node**. It is the degree of disorder or randomness in the data. Its **values lies between 0 and 1.**
- **Lower the value of entropy, higher is the purity of the node**.
- The **entropy of a homogeneous node is zero.**

What is Information Gain

- **Constructing a decision tree is all about finding attribute that returns the highest information gain** (i.e., the most homogeneous branches).
- It is commonly used in the construction of decision trees from a training dataset, by evaluating the information gain for each variable, and **selecting the variable that maximizes the information gain**, which in turn **minimizes the entropy and best splits the dataset into groups for effective classification.**
- In order to **evaluate Information Gain**, the **difference in entropy before and after the split is calculated**.
- That is, first we calculate the entropy of the dataset before the split, and then we calculate the entropy for each subset after the split. Finally, the sum of the output entropies weighted by the size of the subsets is subtracted from the entropy of the dataset before the split. This difference measures the gain in information or the reduction in entropy.
- If the information gain is a positive number, this means that we move from a confused dataset to a number of pure subsets.

$$Information\ Gain\ =\ Entropy(before) - \sum_{j=1}^{K} Entropy(j,\ after)$$

- Where "before" is the dataset before the split, K is the number of subsets generated by the split, and (j, after) is subset j after the split.
- At each step, we would then choose to split the data on the feature with the highest value in information gain as this leads to the purest subsets.
- The **algorithm has the disadvantage of favoring features with a larger number of values, generating larger decision trees.**

What is Gini Index (CART)

- The Gini index is based on Gini impurity. Gini impurity is defined as 1 minus the sum of the squares of the class probabilities in a dataset.

$$Gini\ Impurity\ (p)\ =\ 1\ -\ \sum_{i=1}^{N} p_i^2$$

- Where p is the whole dataset, N is the number of classes, and pi is the frequency of class i in the same dataset.
- **For a dataset with two classes, the range of the Gini index is between 0 and 0.5: 0 if the dataset is pure and 0.5 if the two classes are distributed equally. Thus, the feature with the lowest Gini index is used as the next splitting feature.**

$$Gini\ Index\ =\ \sum_{j=1}^{K} w_j\ Gini\ Impurity\ (j,\ after)$$

$$w_j\ =\ \frac{\#\ data\ in\ in\ subset\ (j,\ after)}{\#\ data\ in\ dataset\ (before)}$$

Where K is the number of subsets generated by the split and (j, after) is subset j after the split.

What is Chi-Square (CHAID)

- The acronym **CHAID stands for Chi-squared Automatic Interaction Detector.**
- It finds out the statistical significance between the differences between sub-nodes and parent node. **We measure it by the sum of squares of standardized differences between observed and expected frequencies of the target variable.**
- It works with the 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**.
- It generates a tree called CHAID (Chi-square Automatic Interaction Detector).

$$\chi^2 = \sum \frac{(O-E)^2}{E}$$

Where:

χ^2 = Chi Square obtained
$\sum$ = the sum of
O = observed score
E = expected score

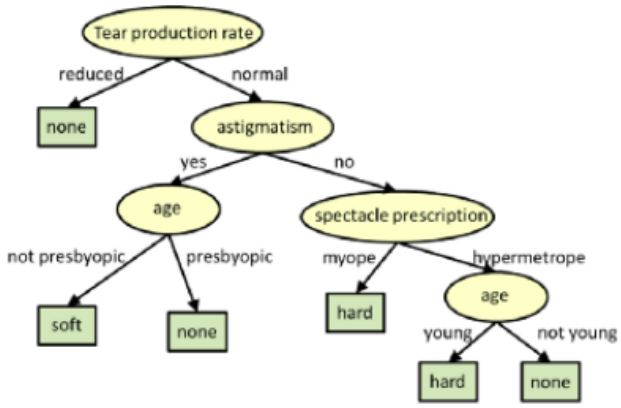
- Steps to Calculate Chi-square for a split:
  1. Calculate Chi-square for an individual node by calculating the deviation for Success and Failure both
  2. Calculated Chi-square of Split using Sum of all Chi-square of success and Failure of each node of the split

How to handle Size and Overfitting in Decision Tree.

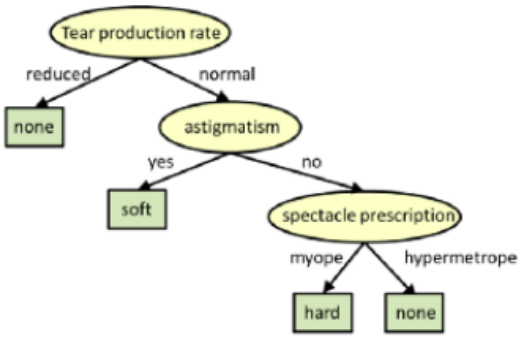
There are two ways to avoid an over-specialized tree: Pruning and/or Random Forest.

Pruning

- In pruning, you **trim off the branches of the tree**, i.e., remove the decision nodes starting from the leaf node such that the overall accuracy is not disturbed. This is done by segregating the actual training set into two sets: training data set, D and validation data set, V. Prepare the decision tree using the segregated training data set, D. Then continue trimming the tree accordingly to optimize the accuracy of the validation data set, V.



Original Tree



Pruned Tree

In the above diagram, the 'Age' attribute in the left-hand side of the tree has been pruned as it has more importance on the right-hand side of the tree, hence removing overfitting.

Random Forest

Random Forest is an example of ensemble learning, in which we combine multiple machine learning algorithms to obtain better predictive performance.

Why the name "Random"?

Two key concepts that give it the name random:

- A **random sampling of training data set** when building trees.
- **Random subsets of features considered** when splitting nodes.

A technique known as bagging is used to create an ensemble of trees where multiple training sets are generated with replacement.

In the bagging technique, a data set is divided into N samples using randomized sampling. Then, using a single learning algorithm a model is built on all samples. Later, the resultant predictions are combined using voting or averaging in parallel.

Advantages of Decision Tree

- Trees can be **visualised**.
- **Requires little data preparation**. Other techniques often require data normalisation, dummy variables need to be created and blank values to be removed.
- Able to **handle multi-output** problems.
- Uses a white box model. If a given situation is observable in a model, the **explanation for the condition is easily explained** by boolean logic.

Issues faced in Decision Trees

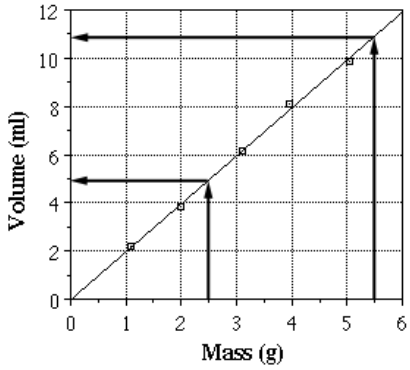
- Decision-tree learners can create over-complex trees that **do not generalise the data well**. This is called **overfitting**. Mechanisms such as pruning, setting the minimum number of samples required at a leaf node or setting the maximum depth of the tree are necessary to avoid this problem.
- Decision trees **can be unstable because small variations in the data might result in a completely different tree being generated**. This problem is mitigated by using decision trees within an ensemble.
- Predictions of decision trees are neither smooth nor continuous, but piecewise constant approximations. Therefore, they are **not good at extrapolation.**
- The problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts. Consequently, **practical decision-tree learning algorithms are based on heuristic algorithms such as the greedy algorithm where locally optimal decisions are made at each node. Such algorithms cannot guarantee to return the globally optimal decision tree. This can be mitigated by training multiple trees in an ensemble learner, where the features and samples are randomly sampled with replacement.**
- Decision tree learners **create biased trees if some classes dominate**. It is therefore recommended to balance the dataset prior to fitting with the decision tree.

What is Interpolation and Extrapolation.

- Besides being able to show trends between variables, plotting data on a graph allows us to predict values for which we have taken no data.
- When we predict values that fall within the range of data points taken it is called **interpolation**.

- When we predict values for points outside the range of data taken it is called **extrapolation**.
- Extrapolation over too far a range can be dangerous unless it is certain that the relationship between the variables continues over the entire range.

  Consider these examples based on the volume/mass data from the previous page. We could use our graph to interpolate the volume for a sample with a mass of 2.5 g. This is done by drawing a vertical line from the x-axis at a value of 2.5 g until it crosses our best fit line, and then drawing a horizontal line to the y-axis. The y-value at this point, 4.9 ml, is equal to the volume of 2.5 g sample. The same process is used for extrapolation. A sample with a mass of 5.5 g, will have a volume of 10.8 ml.



These values could also be determined using the equation for the best fit line determined previously.

volume = 1.982 (mass) + 0.002

If 2.5 g is substituted for the mass, the calculated volume will be 4.96 ml. If a mass of 5.5 g is used, the volume will be 10.90 ml. Both of these values are close to those read off the graph.

Note that while in both of the above examples we determined the volume from a given mass, the opposite could also be performed. If a sample has a volume of 7.0 ml, we could determine that its mass should be 3.53 g.

Assumptions in Interpolation and Extrapolation.

- Use of aggregate data, generally across time (population, employment, etc.)
- Future movement of the data series is determined by past patterns embedded in the series
- The essential information about the future of the data series is contained in the history of the series
- Past trends will continue into the future

Disadvantages in Interpolation and Extrapolation.

- Does not account for underlying causes / structural conditions
- Current trend often does not continue Excludes any external considerations
- Lots of possible way to extend/decrease the line.

What is Local Optima and Global Optima. Give an example.

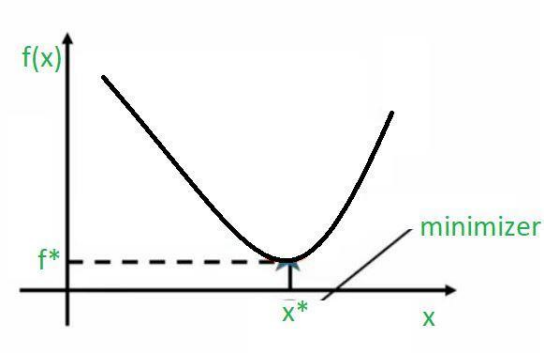
- Uni-variate optimization is a simple case of a non-linear optimization problem with an unconstrained case that is there is no constraint.
- Uni-variate optimization may be defined as a non-linear optimization with no constraint and there is only one decision variable in this optimization that we are trying to find a value for.

min f(x) such that x ∈ R where, f(x) = Objective function **and** x = Decision variable

- So, when you look at this optimization problem you typically write it in this above form where you say you are going to minimize f(x), and this **function is called the objective function**.
- And the variable that **you can use to minimize this function which is called the decision variable** is written below like this w.r.t x here and you also say x is continuous that is it could take any value in the real number line.
- And since this is a uni-variate optimization problem x is a scalar variable and not a vector variable.

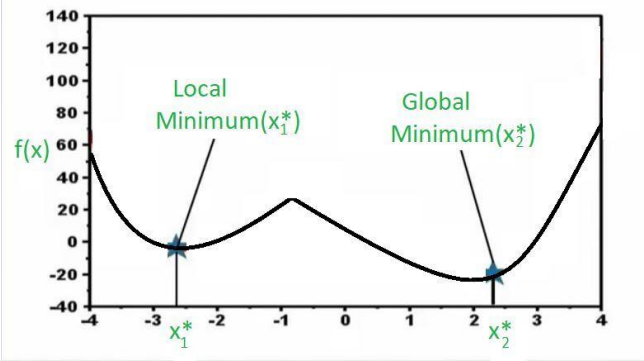
Convex Function:

- Whenever we talk about uni-variate optimization problems it is easy to visualize that in a 2D picture like this.



- So, what we have here is on the x-axis, we have different values for the decision variable x and in the y-axis, we have the function value.
- And when you plot this you can quite easily notice in the graph that marked the point at which this function attains its minimum value.
- So, the point at which this function attains minimum value can be found by dropping a perpendicular onto the x-axis.
- So, you can say x* is the actual value of x at which this function takes a minimum value and the value that the function takes at its minimum point can be identified by dropping this perpendicular onto the y-axis and this f* is the best value this function could possibly take.
- So, the functions of this type are called convex functions because there is only one minimum here.
- So, there is no question of multiple minima to choose from there is only one minimum here, and that is marked in the graph.
- So, in this case, we would say that this minimum is both a local minimum and also a global minimum.
- In fact, we can say it is a local minimum because in the vicinity of this point this is the best solution that you can get. And if the solution that we get in the vicinity of this point is also the best solution globally then we also call it the global minimum.

Non-convex Function:



- Now, take a look at the above graph. Here I have a function and again it is a univariate optimization problem. So, on the x-axis, I have different values of the decision variable and on the y-axis, we plot the function.
- Now, you may notice that there are two points where the function attains a minimum and you can see that when we say minimum we automatically actually only mean locally minimum because if you notice this x1* point in the graph, in the vicinity of this point, this function cannot take any better value from a minimization viewpoint.
- In other words, if I am at x1* and the function is taking this value, if I move to the right, the function value will increase which basically is not good for us because we are trying to find minimum value, and if I move to my left the function value will again increase which is not good because we are finding the minimum for this function.
- This says that in a local vicinity you can never find a point which is better than this.
- However, if you go far away then you will get to this point(x2*) here which again from a local viewpoint is the best because if we go in the right direction the function increases and if we go in the left direction also the function increases, and in this particular example it also turns out that globally this is the best solution.
- So, while both are local minimum in the sense that in the vicinity, they are the best but this local minimum(x2*) is also global minimum because if you take the whole region you still can't beat this solution.
- So, when you have a solution which is the lowest in the whole region then we call that as a global minimum. And these are types of functions that we call as non-convex functions where there are multiple local optima and the job of an optimizer is to find out the best solution from the many optimum solutions that are possible.

How to check variable importance in Decision tree

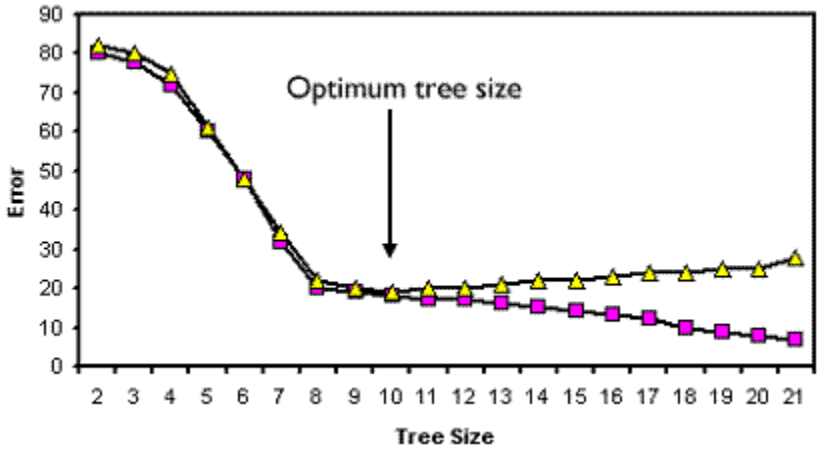
- A Decision Tree crawls through your data, one variable at a time, and attempts to determine how it can split the data into smaller, more homogeneous buckets. It can be used for either numeric or categorical prediction.
- The basic decision trees use Gini Index or Information Gain to help determine which variables are most important. That most important variable is then put at the top of your tree.
- However, there are some drawbacks to using a decision tree to help with variable importance.
    1. **Correlation not Causation:** The Gini Index or Information Gain is optimized but that doesn't imply that it's the cause of your response variables.
    2. **Choosing Gini or Information Gain:** Using the Gini Index or Information Gain metrics can lead to different trees and root nodes.
    3. **Binary Splits:** Most software packages are only going to split the data into two buckets at each node. Sometimes your problems won't be that clean.
    4. **Non-Linear Patterns:** Decision Trees are designed to look at linear separations of data, real-world data is often not so simple.
    5. **Biased Results:** Your data might be biased – one type of customer is more prevalent than any other. One decision tree isn't going to be able to get past this bias.

To **address the last two**, we can turn to **Random Forests**

- A Random Forest model **combines many decision trees but introduces some randomness** into the models built. A subset of variables are randomly selected.
- A **bootstrap sample is taken** which effectively takes about 63.2% of the data into training the model.
- Now **each model built is looking at a different subset of the data** and isn't always using the same variables. Now you won't be second guessing yourself on the importance of your root node.

When to stop the Decision Tree

- For best accuracy, minimum error pruning without early stopping is usually a good choice. (most common pruning methods used to improve generalization is k-fold cross validation)
- For a compromise between accuracy and an interpretable tree, try smallest tree pruning without early stopping.
- To produce an even smaller tree or reduce the running time while allowing accuracy to decrease, you can turn on early stopping.



What is Ensemble Techniques

Ensemble methods are techniques that create multiple models and then combine them to produce improved results. Ensemble methods usually produces more accurate solutions than a single model would.

Different types of Ensemble Learning techniques

Simple: 1. Max Voting 2. Averaging 3. Weighted Averaging

Advanced: 1. **Bagging** 2. **Boosting**

In bagging and Boosting, there are other popular models like Gradient Boosting, Random Forest, XGBoost, etc.

Do we use Ensemble Learning techniques only for classification or regression or both?

We can **use ensemble learning for both types of machine learning problems**: Classification and Regression. While techniques like **Max Voting are used for classification**, techniques **like Random Forest, Gradient Boosting, etc can be used for both Classification and Regression problems.**

What is the intuition behind Bagging?

The idea behind **bagging is combining the results of multiple models** (for instance, all decision trees) to get a generalized result. Bagging (or Bootstrap Aggregating) technique uses subsets (bags) to get a fair idea of the distribution (complete set).

What is the intuition behind Boosting?

Boosting is a **sequential process**, where each subsequent model **attempts to correct the errors of the previous model on subsets of the data**. The **succeeding models are dependent on the previous model**. The boosting algorithm **combines a number of weak learners to form a strong learner and boost your overall results.**

What is the difference between Bagging and Boosting?

While both bagging and boosting involve creating subsets, **bagging makes these subsets randomly, while boosting prioritizes misclassified subsets**. Additionally, at the **final step in bagging, the weighted average is used, while boosting uses majority weighted voting.**

What is Max Voting.

- The max voting method is generally used for classification problems.
- In this technique, multiple models are used to make predictions for each data point.
- The predictions by each model are considered as a 'vote'. The predictions which we get from the majority of the models are used as the final prediction.
  For example, when you asked 5 of your colleagues to rate your movie (out of 5); we'll assume three of them rated it as 4 while two of them gave it a 5. Since the majority gave a rating of 4, the final rating will be taken as 4. You can consider this as taking the mode of all the predictions.

What is Averaging.

- Similar to the max voting technique, multiple predictions are made for each data point in averaging.
- In this method, we take an average of predictions from all the models and use it to make the final prediction. Averaging can be used for making predictions in regression problems or while calculating probabilities for classification problems.
  For example, in the below case, the averaging method would take the average of all the values.
  i.e. (5+4+5+4+4)/5 = 4.4

What is Weighted Average

- This is an extension of the averaging method.
- All models are assigned different weights defining the importance of each model for prediction.
- For instance, if two of your colleagues are critics, while others have no prior experience in this field, then the answers by these two friends are given more importance as compared to the other people.

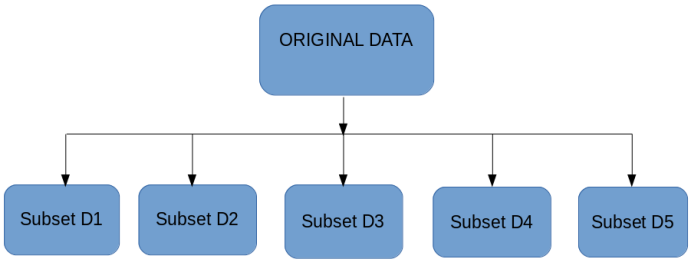
  The result is calculated as [(5*0.23) + (4*0.23) + (5*0.18) + (4*0.18) + (4*0.18)] = 4.41.

What is Out of Bag Sampling (OOB)?

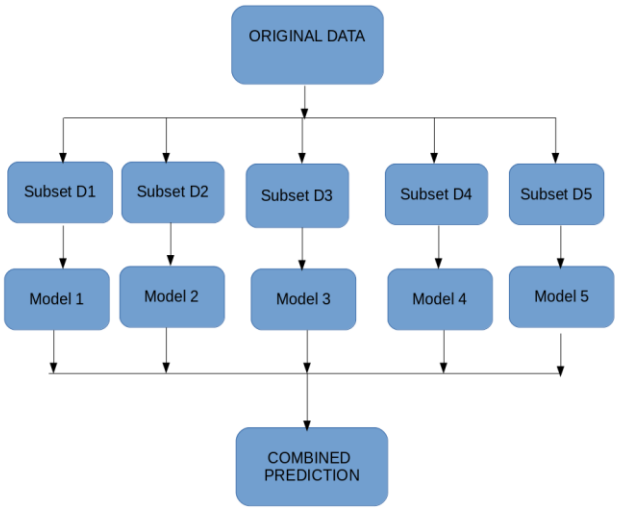
OOB which is a random forest cross-validation method. In this sampling, about **one-third of the data is not used to train the model and can be used to evaluate its performance**. These samples are called the **out-of-bag samples**. It's **very similar to the leave-one-out-cross-validation method**, but almost no additional computational burden goes along with it.

What is Bagging

- The idea **behind bagging is combining the results of multiple models** (for instance, all decision trees) to **get a generalized result**.
- Here's a question: If you **create all the models on the same set of data and combine it, will it be useful?** There is a high chance that these models will give the same result since they are getting the same input. So how can we solve this problem? **One of the techniques is bootstrapping**.
- **Bootstrapping is a sampling technique in which we create subsets of observations from the original dataset, with replacement.** The size of the subsets is the same as the size of the original set.
- Bagging (or Bootstrap Aggregating) technique uses these subsets (bags) to get a fair idea of the distribution (complete set). The size of subsets created for bagging may be less than the original set.



1. Multiple subsets are created from the original dataset, selecting observations with replacement.
2. A base model (weak model) is created on each of these subsets.
3. The models run in parallel and are independent of each other.
4. The final predictions are determined by combining the predictions from all the models.



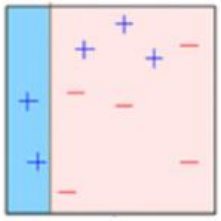
What is Boosting

If a data point is incorrectly predicted by the first model, and then the next (probably all models), will combining the predictions provide better results? Such situations are taken care of by boosting.

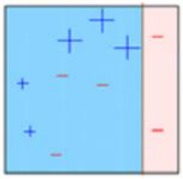
Boosting is a sequential process, where each subsequent model attempts to correct the errors of the previous model. The succeeding models are dependent on the previous model. Let's understand the way boosting works in the below steps.

1. A subset is created from the original dataset.
2. Initially, all data points are given equal weights.

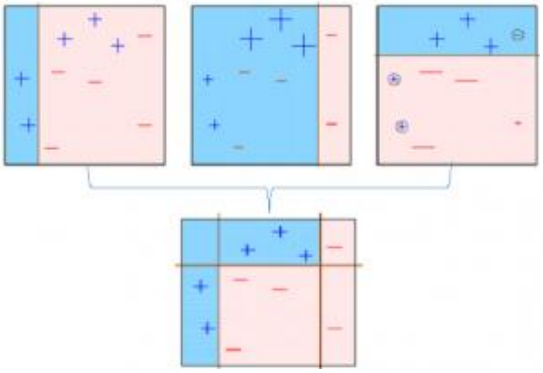
3. A base model is created on this subset.
4. This model is used to make predictions on the whole dataset.



5. Errors are calculated using the actual values and predicted values.
6. The observations which are incorrectly predicted, are given higher weights. (Here, the three misclassified blue-plus points will be given higher weights)
7. Another model is created, and predictions are made on the dataset. (This model tries to correct the errors from the previous model)



8. Similarly, multiple models are created, each correcting the errors of the previous model.
9. The final model (strong learner) is the weighted mean of all the models (weak learners).



Thus, the boosting algorithm combines a number of weak learners to form a strong learner. The individual models would not perform well on the entire dataset, but they work well for some part of the dataset. Thus, each model actually boosts the performance of the ensemble.

Algorithms based on Bagging and Boosting

Bagging algorithms: 1. Random forest

Boosting algorithms: 1. AdaBoost 2. GBM 3. XGBoost 4.Light GBM 5. CatBoost

What is Random Forest

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is controlled with the max_samples parameter if bootstrap=True (default), otherwise the whole dataset is used to build each tree.

Feature Importance in Random Forest

Impurity-based feature importance. The higher, the more important the feature. The importance of a feature is computed as the (normalized) total reduction of the criterion brought by that feature. It is also known as the Gini importance.

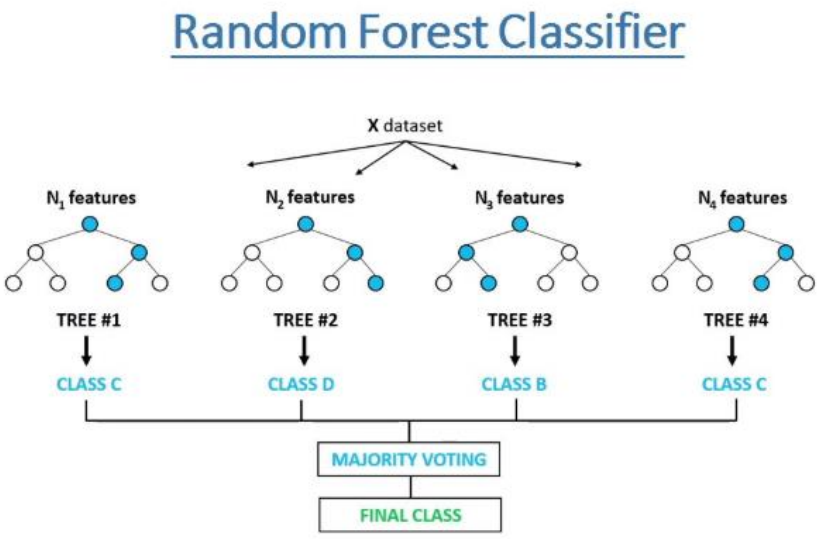
How does the Random Forest algorithm work?

Step 1: The algorithm **select random samples** from the dataset provided.

Step 2: The algorithm will **create a decision tree for each sample selected**. Then it **will get a prediction result** from each decision tree created.

Step 3: **Voting will then be performed** for every predicted result. For a **classification problem**, it will **use mode**, and for a **regression problem**, it will **use mean**.

Step 4: And finally, the algorithm will select the **most voted prediction result as the final prediction**.



Limitations in Random Forest

- For any data, that a Random Forest has not seen before, at best, it can predict an average of training values that it has seen before.
- It the Validation set consists of data points that are greater or less than the training data points, a Random Forest will provide us with Average results as it is not able to Extrapolate and understand the growing/decreasing trend in our data.

Therefore, a Random Forest model does not scale very well for time-series data and might need to be constantly updated in Production or trained with some Random Data that lies outside our range of Training set.

Example of Limitations of Random Forest

Answering questions like “What would the Sales be for next Year?”, “What would the population of China be after 5 years?”, “What would the global temperature be in 50 years from now?” or “How many units am I expected to sell for gloves in the next three months?” becomes really difficult when using Random Forests.

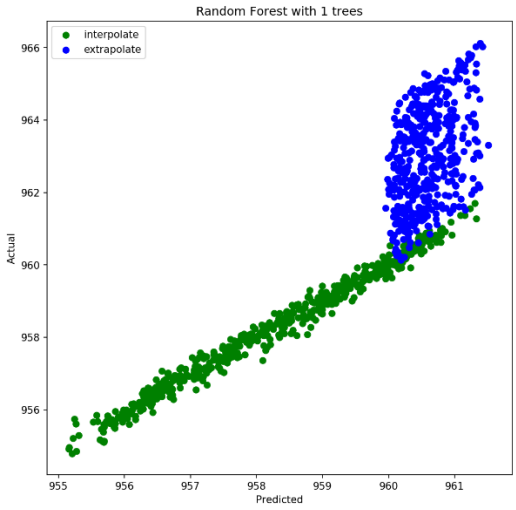
Features like Age Year Made and Sale Day of year which are all time dependent.

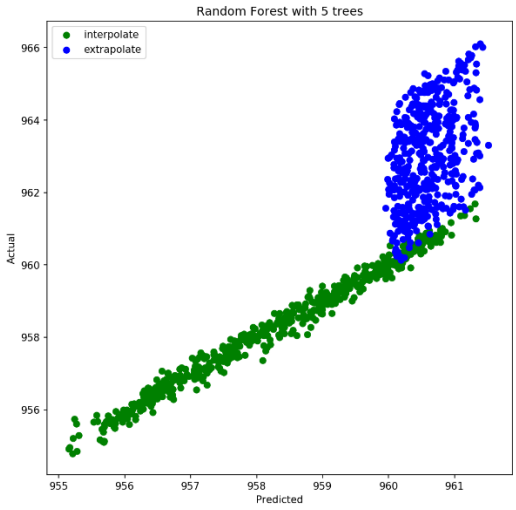
More Information regarding Limitations of Random Forest

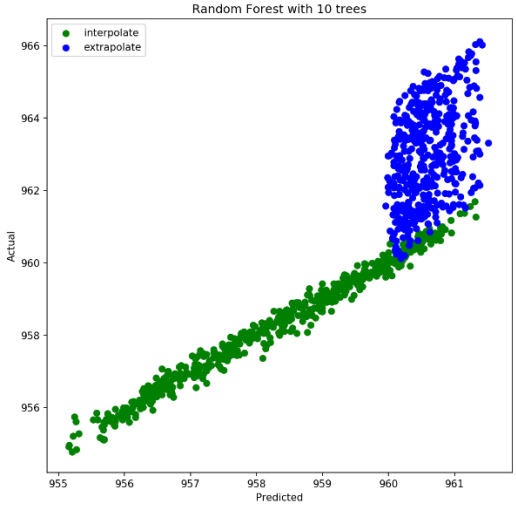
Random Forest can’t extrapolate. It can only make a prediction that is an average of previously observed labels.

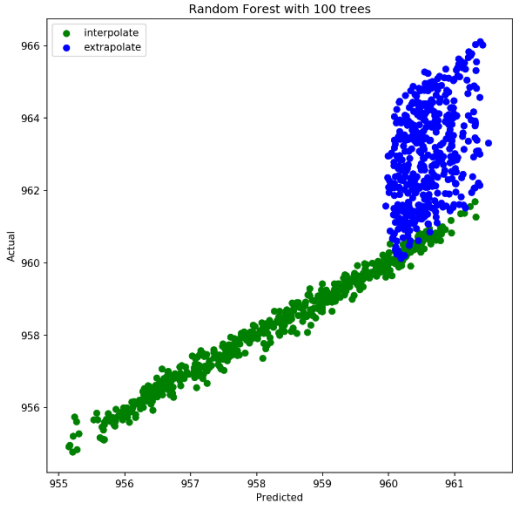
In other words, in a regression problem, the range of predictions a Random Forest can make is bound by the highest and lowest labels in the training data. This behavior becomes problematic in situations where the training and prediction inputs differ in their range and/or distributions. It is difficult for most models to handle but especially for Random Forest, because it can’t extrapolate.

Random Forest Output

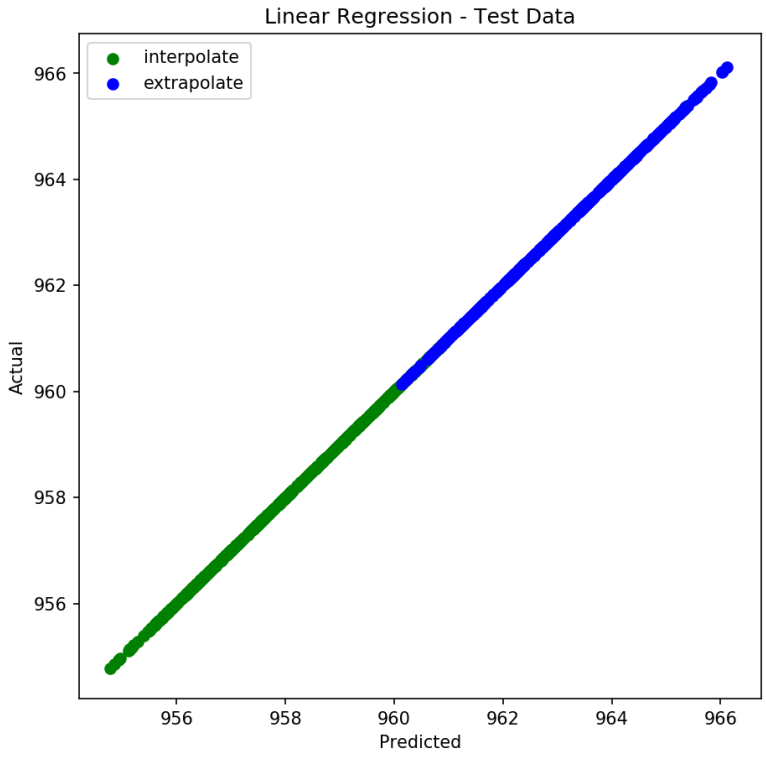








This plot makes it clear that the highest value the model can predict is around 961, while the underlying trend in the data pushes more recent values as high as 966. Unfortunately, the Random Forest can't extrapolate the linear trend and accurately predict



A model like Linear Regression would be a better choice and will have no problem detecting the trend in the data and making accurate predictions for data outside of the time ranges in the training data.

How to solve this Problem of Extrapolation?

Look for other options

Therefore, fitting a Linear Model or a Neural Net, in this case, might be sufficient to predict data which has an increasing/decreasing trend.

Ignore the time-series components of data while training the Random Forest

If we do not use the time-related features for our predictions, our Random Forest model will be able to generalise really well!