(https://www.facebook.com/AnalyticsVidhya) | (https://twitter.com/analyticsvidhya) | (https://plı

(https://www.linkedin.com/groups/Analytics-Vidhya-Learn-everything-about-5057165)

Home (https://www.analyticsvidhya.com/) | Blog (https://www.analyticsvidhya.com/blog/) | Jobs (https:

Trainings (https://www.analyticsvidhya.com/trainings/)

Learning Paths (https://www.analyticsvidhya.com/learning-paths-data-science-business-analytics-business-

DataHack (https://datahack.analyticsvidhya.com)

Analytics Vidhya
Learn Everything About Analytics (https://www.analyticsvidhya.com)

A Complete Tutorial on Tree Based Modeling froı Scratch (in R & Python)

MACHINE LEARNING (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/MACHINE-LEARNING/) PYTHON (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/PYTHON-2/) R (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/I

arer.php?u=https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-
20on%20Tree%20Based%20Modeling%20from%20Scratch%20(in%20R%20&%20Python)) (https://twitter.com/home?
n%20Tree%20Based%20Modeling%20from%20Scratch%20(in%20R%20&%20Python)+https://www.analyticsvidhya.com/blog/201
/thon/) (https://plus.google.com/share?url=https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeli
:reate/button/?url=https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-
lhya.com/wp-
:cription=A%20Complete%20Tutorial%20on%20Tree%20Based%20Modeling%20from%20Scratch%20(in%20R%20&%20Python))

Introduction

Tree based learning algorithms are considered to be one of the best and mostly used learning methods. Tree based methods empower predictive models with high accuracy, s ease of interpretation. Unlike linear models, they map non-linear relationships quite are adaptable at solving any kind of problem at hand (classification or regression).

Methods like decision trees, random forest, gradient boosting are being popularly used in data science problems. Hence, for every analyst (fresher also), it's important to learn these and use them for modeling.

This tutorial is meant to help beginners learn tree based modeling from scratch. After the completion of this tutorial, one is expected to become proficient at using tree based and build predictive models.

Note: This tutorial requires no prior knowledge of machine learning. However, elementary of R or Python will be helpful. To get started you can follow full tuto (https://www.analyticsvidhya.com/blog/2016/02/complete-tutorial-learn-data-science-s and full tutorial in Python (https://www.analyticsvidhya.com/blog/2016/01/complete-tut data-science-python-scratch-2/).

Table of Contents

(https://www.analyticsvidhya.com) HOME (HTTPS://WWW.ANAL

BLOG (HTTPS://WWW.ANALY

JOBS (HTTPS://WWW.ANALYTICSVIDHYA.COM/JOBS/) TRAININGS (HTTPS://WWW.ANALYTICSVIDHYA.CO

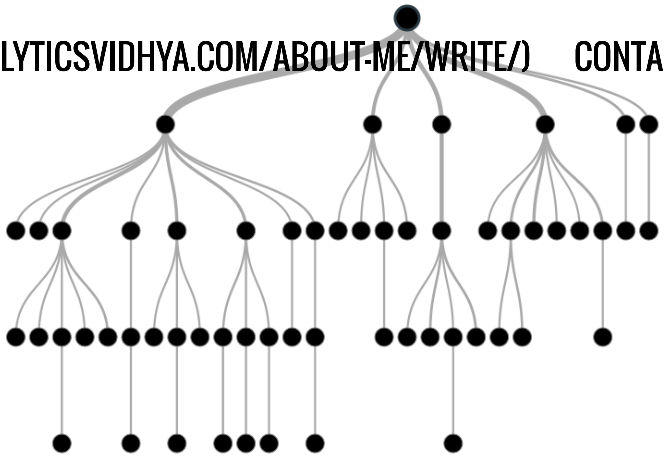
LEARNING PATHS (HTTPS://WWW.ANALYTICSVIDHYA.COM/LEARNING-PATHS-DATA-SCIENCE-BUSINESS-ANAL

DATAHACK (HTTPS://DATAHACK.ANALYTICSVIDHYA.COM) ⌄ STORIES (HTTPS://WWW.ANALYTICSVIDHYA.(

WRITE FOR US (HTTP://WWW.ANALYTICSVIDHYA.COM/ABOUT-ME/WRITE/) CONTACT US (HTTPS://WWW

1. What is a Decision Tree ? How does it work ?

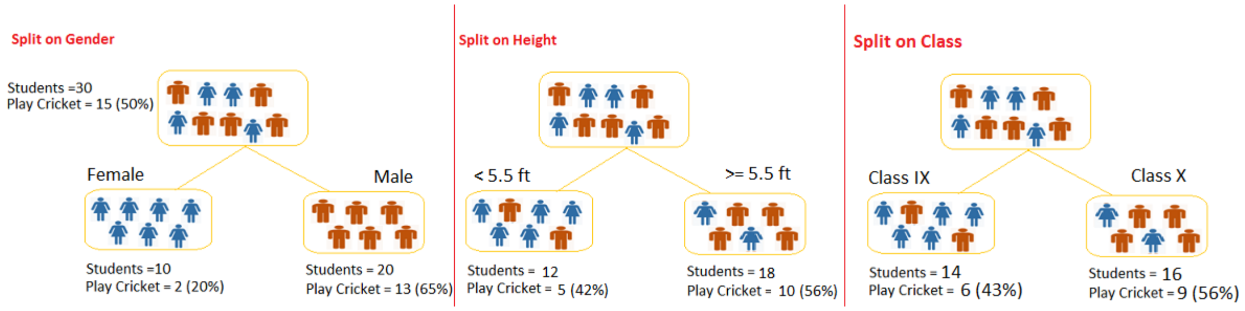
Decision tree is a type of supervised learning algorithm (having a pre-defined target varia mostly used in classification problems. It works for both categorical and continuous input variables. In this technique, we split the population or sample into two or more homogene sub-populations) based on most significant splitter / differentiator in input variables.



Example:-

Let's say we have a sample of 30 students with three variables Gender (Boy/ Girl), Class(Height (5 to 6 ft). 15 out of these 30 play cricket in leisure time. Now, I want to creat to predict who will play cricket during leisure period? In this problem, we need to segrega who play cricket in their leisure time based on highly significant input variable among all thr

This is where decision tree helps, it will segregate the students based on all values of thr and identify the variable, which creates the best homogeneous sets of students heterogeneous to each other). In the snapshot below, you can see that variable Gende identify best homogeneous sets compared to the other two variables.



(https://www.analyticsvidhya.com/blog/wp-content/uploads/2015/01/Test.png)

As mentioned above, decision tree identifies the most significant variable and it's value tha homogeneous sets of population. Now the question which arises is, how does it identify t and the split? To do this, decision tree uses various algorithms, which we will shall dis following section.

Types of Decision Trees

Types of decision tree is based on the type of target variable we have. It can be of two type

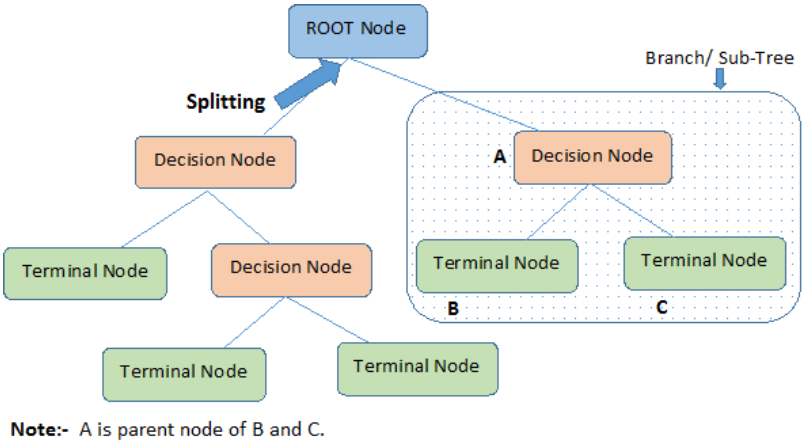
1. **Categorical Variable Decision Tree:** Decision Tree which has categorical target variable th as categorical variable decision tree. Example:- In above scenario of student problem, whe variable was "Student will play cricket or not" i.e. YES or NO.
2. **Continuous Variable Decision Tree:** Decision Tree has continuous target variable then it Continuous Variable Decision Tree.

Example:- Let's say we have a problem to predict whether a customer will pay his renew with an insurance company (yes/ no). Here we know that income of customer is a significa but insurance company does not have income details for all customers. Now, as we kno important variable, then we can build a decision tree to predict customer income occupation, product and various other variables. In this case, we are predicting values for variable.

Important Terminology related to Decision Trees

Let's look at the basic terminology used with Decision trees:

1. **Root Node:** It represents entire population or sample and this further gets divided into t homogeneous sets.
2. **Splitting:** It is a process of dividing a node into two or more sub-nodes.
3. **Decision Node:** When a sub-node splits into further sub-nodes, then it is called decision nc
4. **Leaf/ Terminal Node:** Nodes do not split is called Leaf or Terminal node.



5. (https://www.analyticsvidhya.com/blog/wp-content/uploads/2015/01/Decision_Tree_2.png)**Pruning:** When we remove sub-nodes o node, this process is called pruning. You can say opposite process of splitting.
6. **Branch / Sub-Tree:** A sub section of entire tree is called branch or sub-tree.
7. **Parent and Child Node:** A node, which is divided into sub-nodes is called parent node o where as sub-nodes are the child of parent node.

These are the terms commonly used for decision trees. As we know that every alg advantages and disadvantages, below are the important factors which one should know.

Advantages

1. **Easy to Understand**: Decision tree output is very easy to understand even for people analytical background. It does not require any statistical knowledge to read and interpr graphical representation is very intuitive and users can easily relate their hypothesis.
2. **Useful in Data exploration:** Decision tree is one of the fastest way to identify most significa and relation between two or more variables. With the help of decision trees, we can variables / features that has better power to predict target variable. You can refer arti enhance power of regression model (https://www.analyticsvidhya.com/blog/2013/10/tric power-regression-model-2/)) for one such trick. It can also be used in data exploratio

example, we are working on a problem where we have information available in hundreds there decision tree will help to identify most significant variable.

3. **Less data cleaning required:** It requires less data cleaning compared to some othe techniques. It is not influenced by outliers and missing values to a fair degree.
4. **Data type is not a constraint:** It can handle both numerical and categorical variables.
5. **Non Parametric Method:** Decision tree is considered to be a non-parametric method. This decision trees have no assumptions about the space distribution and the classifier structure

Disadvantages

1. **Over fitting:** Over fitting is one of the most practical difficulty for decision tree models. This gets solved by setting constraints on model parameters and pruning (discussed in detailed
2. **Not fit for continuous variables**: While working with continuous numerical variables, decisi looses information when it categorizes variables in different categories.

2. Regression Trees vs Classification Trees

We all know that the terminal nodes (or leaves) lies at the bottom of the decision tree. This decision trees are typically drawn upside down such that leaves are the the bottom & rc tops (shown below).

Both the trees work almost similar to each other, let's look at the primary diff similarity between classification and regression trees:

1. Regression trees are used when dependent variable is continuous. Classification trees are dependent variable is categorical.
2. In case of regression tree, the value obtained by terminal nodes in the training data i response of observation falling in that region. Thus, if an unseen data observation falls in we'll make its prediction with mean value.
3. In case of classification tree, the value (class) obtained by terminal node in the training mode of observations falling in that region. Thus, if an unseen data observation falls in that make its prediction with mode value.
4. Both the trees divide the predictor space (independent variables) into distinct and non- regions. For the sake of simplicity, you can think of these regions as high dimensional boxes
5. Both the trees follow a top-down greedy approach known as recursive binary splitting. \ 'top-down' because it begins from the top of tree when all the observations are available region and successively splits the predictor space into two new branches down the tree. It 'greedy' because, the algorithm cares (looks for best variable available) about only the c and not about future splits which will lead to a better tree.
6. This splitting process is continued until a user defined stopping criteria is reached. For exam tell the the algorithm to stop once the number of observations per node becomes less thar
7. In both the cases, the splitting process results in fully grown trees until the stopping criteria But, the fully grown tree is likely to overfit data, leading to poor accuracy on unseen data

'pruning'. Pruning is one of the technique used tackle overfitting. We'll learn more about it section.

3. How does a tree decide where to split?

The decision of making strategic splits heavily affects a tree's accuracy. The decisior different for classification and regression trees.

Decision trees use multiple algorithms to decide to split a node in two or more sub-r creation of sub-nodes increases the homogeneity of resultant sub-nodes. In other words, that purity of the node increases with respect to the target variable. Decision tree splits th all available variables and then selects the split which results in most homogeneous sub-n

The algorithm selection is also based on type of target variables. Let's look at the commonly used algorithms in decision tree:

Gini Index

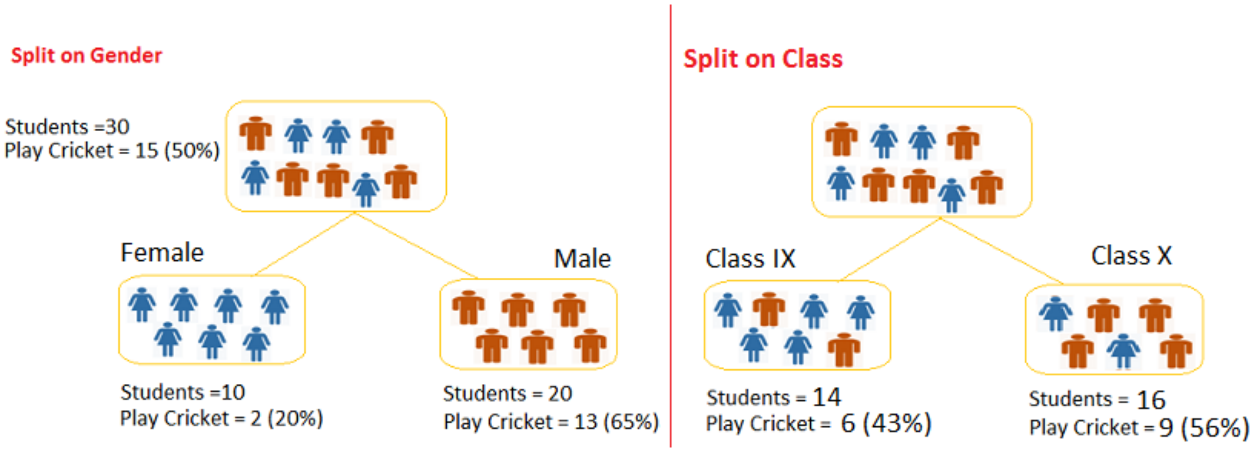
Gini index says, if we select two items from a population at random then they must be of and probability for this is 1 if population is pure.

1. It works with categorical target variable "Success" or "Failure".
2. It performs only Binary splits
3. Higher the value of Gini higher the homogeneity.
4. CART (Classification and Regression Tree) uses Gini method to create binary splits.

Steps to Calculate Gini for a split

1. Calculate Gini for sub-nodes, using formula sum of square of probability for success (p^2+q^2).
2. Calculate Gini for split using weighted Gini score of each node of that split

Example: – Referring to example used above, where we want to segregate the students target variable (playing cricket or not). In the snapshot below, we split the population usin variables Gender and Class. Now, I want to identify which split is producing more homoge nodes using Gini index.



(https://www.analyticsvidhya.com/blog/wp-content/uploads/2015/01/Decision_Tree_Algorithm1.png)**Split on Gender:**

1. Calculate, Gini for sub-node Female = (0.2)*(0.2)+(0.8)*(0.8)=0.68
2. Gini for sub-node Male = (0.65)*(0.65)+(0.35)*(0.35)=0.55
3. Calculate weighted Gini for Split Gender = (10/30)*0.68+(20/30)*0.55 = **0.59**

Similar for Split on Class:

1. Gini for sub-node Class IX = (0.43)*(0.43)+(0.57)*(0.57)=0.51
2. Gini for sub-node Class X = (0.56)*(0.56)+(0.44)*(0.44)=0.51
3. 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 split will take place on Gender.

Chi-Square

It is an algorithm to find out the statistical significance between the differences between and parent node. We measure it by sum of squares of standardized differences betwee and expected frequencies of target variable.

1. It works with categorical target variable “Success” or “Failure”.
2. It can perform two or more splits.
3. Higher the value of Chi-Square higher the statistical significance of differences between su Parent node.
4. Chi-Square of each node is calculated using formula,

5. Chi-square = ((Actual – Expected)^2 / Expected)^1/2
6. It generates tree called CHAID (Chi-square Automatic Interaction Detector)

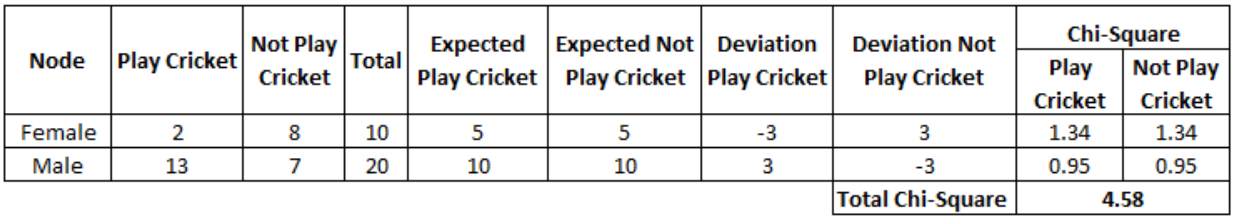
Steps to Calculate Chi-square for a split:

1. Calculate Chi-square for individual node by calculating the deviation for Success and Failur
2. Calculated Chi-square of Split using Sum of all Chi-square of success and Failure of each n split

Example: Let's work with above example that we have used to calculate Gini.

Split on Gender:

1. First we are populating for node Female, Populate the actual value for "**Play Cricket"** an **Cricket"**, here these are 2 and 8 respectively.
2. Calculate expected value for "**Play Cricket"** and "**Not Play Cricket"**, here it would be because parent node has probability of 50% and we have applied same probability count(10).
3. Calculate deviations by using formula, Actual – Expected. It is for "**Play Cricket"** (2 – 5 = -3) **play cricket"** (8 – 5 = 3).
4. Calculate Chi-square of node for "**Play Cricket**" and "**Not Play Cricket**" using formula w **= ((Actual – Expected)^2 / Expected)^1/2**. You can refer below table for calculation.
5. Follow similar steps for calculating Chi-square value for Male node.
6. Now add all Chi-square values to calculate Chi-square for split Gender.

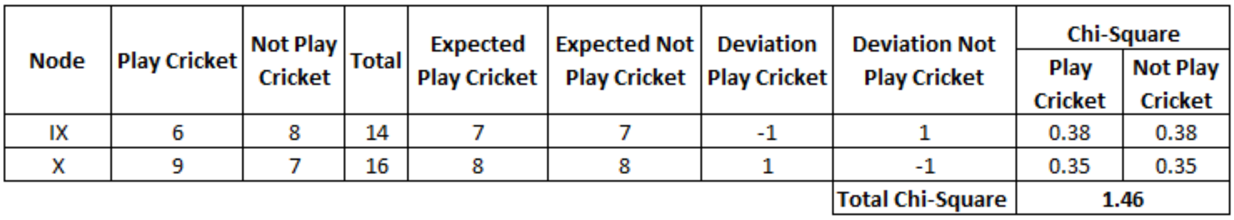


Node	Play Cricket	Not Play Cricket	Total	Expected Play Cricket	Expected Not Play Cricket	Deviation Play Cricket	Deviation Not Play Cricket	Chi-Square	
								Play Cricket	Not Play Cricket
Female	2	8	10	5	5	-3	3	1.34	1.34
Male	13	7	20	10	10	3	-3	0.95	0.95
							Total Chi-Square	4.58	

(https://www.analyticsvidhya.com/blog/wp-content/uploads/2015/01/Decision_Tree_Chi_Square1.png)

Split on Class:

Perform similar steps of calculation for split on Class and you will come up with below tabl

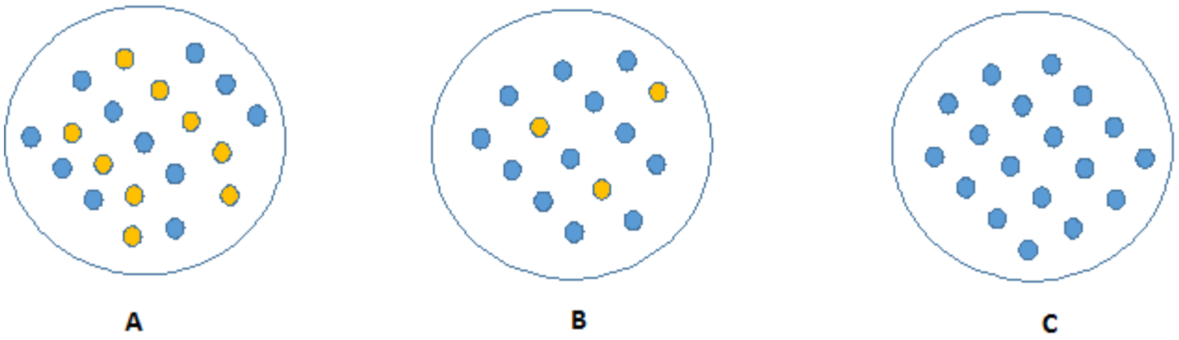


Node	Play Cricket	Not Play Cricket	Total	Expected Play Cricket	Expected Not Play Cricket	Deviation Play Cricket	Deviation Not Play Cricket	Chi-Square	
								Play Cricket	Not Play Cricket
IX	6	8	14	7	7	-1	1	0.38	0.38
X	9	7	16	8	8	1	-1	0.35	0.35
							Total Chi-Square	1.46	

(https://www.analyticsvidhya.com/blog/wp-content/uploads/2015/01/Decision_Tree_Chi_Square_2.png)Above, you can see that Chi-s identify the Gender split is more significant compare to Class.

Information Gain:

Look at the image below and think which node can be described easily. I am sure, your a because it requires less information as all values are similar. On the other hand, B req information to describe it and A requires the maximum information. In other words, we car is a Pure node, B is less Impure and A is more impure.



(https://www.analyticsvidhya.com/blog/wp-content/uploads/2015/01/Information_Gain_Decision_Tree2.png)

Now, we can build a conclusion that less impure node requires less information to descr more impure node requires more information. Information theory is a measure to define thi disorganization in a system known as Entropy. If the sample is completely homogeneou entropy is zero and if the sample is an equally divided (50% – 50%), it has entropy of one.

Entropy can be calculated using formula:-

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

Here p and q is probability of success and failure respectively in that node. Entropy is alsc categorical target variable. It chooses the split which has lowest entropy compared to p and other splits. The lesser the entropy, the better it is.

Steps to calculate entropy for a split:

1. Calculate entropy of parent node
2. Calculate entropy of each individual node of split and calculate weighted average of al available in split.

Example: Let's use this method to identify best split for student example.

1. Entropy for parent node = -(15/30) log2 (15/30) – (15/30) log2 (15/30) = **1**. Here 1 shows that i impure node.
2. Entropy for Female node = -(2/10) log2 (2/10) – (8/10) log2 (8/10) = 0.72 and for male node, log2 (13/20) – (7/20) log2 (7/20) = **0.93**
3. Entropy for split Gender = Weighted entropy of sub-nodes = (10/30)*0.72 + (20/30)*0.93 = **0.8**
4. Entropy for Class IX node, -(6/14) log2 (6/14) – (8/14) log2 (8/14) = 0.99 and for Class X node log2 (9/16) – (7/16) log2 (7/16) = 0.99.
5. 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 wil on *Gender*. We can derive information gain from entropy as **1- Entropy.**

Reduction in Variance

Till now, we have discussed the algorithms for categorical target variable. Reduction in va algorithm used for continuous target variables (regression problems). This algorithm standard formula of variance to choose the best split. The split with lower variance is s the criteria to split the population:

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

(https://www.analyticsvidhya.com/blog/wp-content/uploads/2015/01/Varince.png)

Above X-bar is mean of the values, X is actual and n is number of values.

Steps to calculate Variance:

1. Calculate variance for each node.
2. Calculate variance for each split as weighted average of each node variance.

Example:- Let's assign numerical value 1 for play cricket and 0 for not playing cricket. Now steps to identify the right split:

1. Variance for Root node, here mean value is (15*1 + 15*0)/30 = 0.5 and we have 15 one and 1 variance would be ((1-0.5)^2+(1-0.5)^2+....15 times+(0-0.5)^2+(0-0.5)^2+...15 times) / 30, this ca as (15*(1-0.5)^2+15*(0-0.5)^2) / 30 = **0.25**
2. Mean of Female node = (2*1+8*0)/10=0.2 and Variance = (2*(1-0.2)^2+8*(0-0.2)^2) / 10 = 0.16
3. Mean of Male Node = (13*1+7*0)/20=0.65 and Variance = (13*(1-0.65)^2+7*(0-0.65)^2) / 20 = 0.2
4. Variance for Split Gender = Weighted Variance of Sub-nodes = (10/30)*0.16 + (20/30) *0.23 =
5. Mean of Class IX node = (6*1+8*0)/14=0.43 and Variance = (6*(1-0.43)^2+8*(0-0.43)^2) / 14= 0.2
6. Mean of Class X node = (9*1+7*0)/16=0.56 and Variance = (9*(1-0.56)^2+7*(0-0.56)^2) / 16 = 0.2
7. Variance for Split Gender = (14/30)*0.24 + (16/30) *0.25 = **0.25**

Above, you can see that Gender split has lower variance compare to parent node, so the take place on *Gender* variable.

Until here, we learnt about the basics of decision trees and the decision making process choose the best splits in building a tree model. As I said, decision tree can be applie regression and classification problems. Let's understand these aspects in detail.

4. What are the key parameters of tree modeling and h we avoid over-fitting in decision trees?

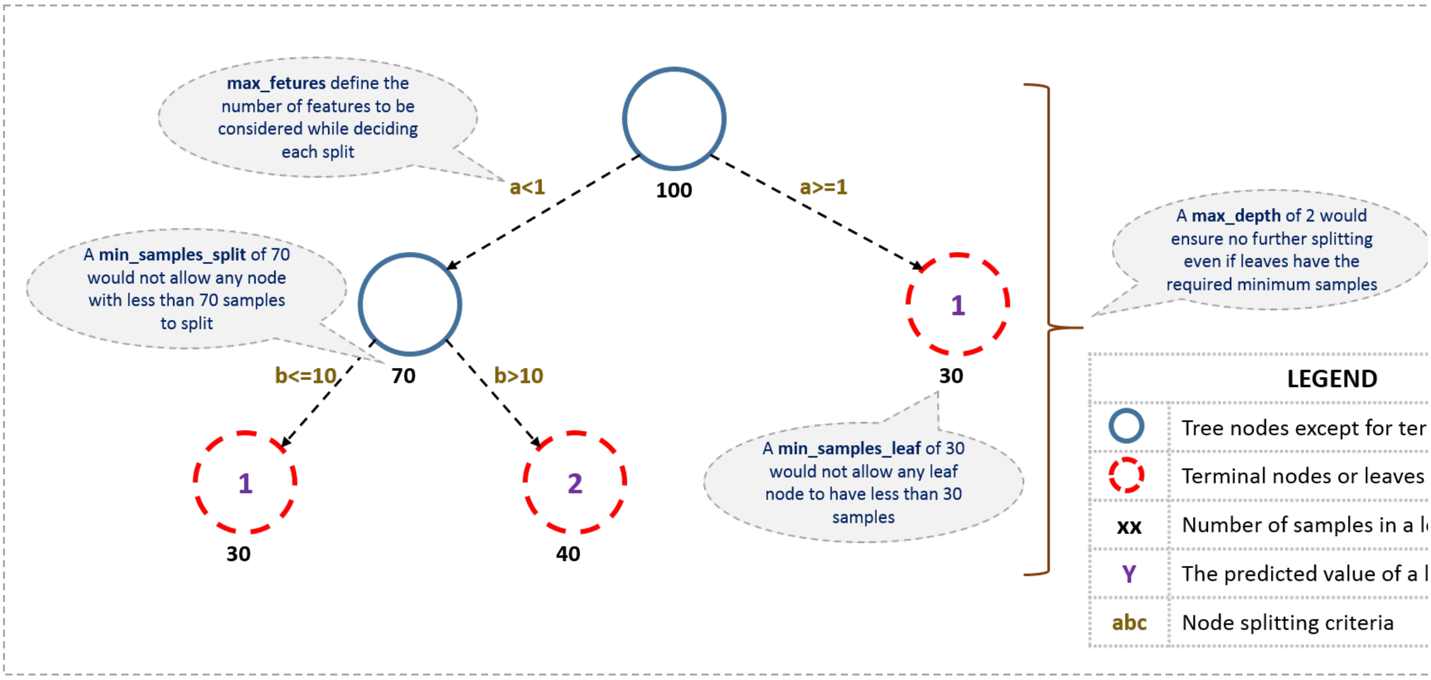
Overfitting is one of the key challenges faced while modeling decision trees. If there is no li decision tree, it will give you 100% accuracy on training set because in the worse case it making 1 leaf for each observation. Thus, preventing overfitting is pivotal while modeling tree and it can be done in 2 ways:

1. Setting constraints on tree size
2. Tree pruning

Lets discuss both of these briefly.

Setting Constraints on Tree Size

This can be done by using various parameters which are used to define a tree. First, lets loc
general structure of a decision tree:


max_fetures define the number of features to be considered while deciding each split
a<1
100
a>=1
A min_samples_split of 70 would not allow any node with less than 70 samples to split
1
b<=10
70
b>10
30
1
2
30
40
A min_samples_leaf of 30 would not allow any leaf node to have less than 30 samples
A max_depth of 2 would ensure no further splitting even if leaves have the required minimum samples
LEGEND
Tree nodes except for ter
Terminal nodes or leaves
xx Number of samples in a l
Y The predicted value of a l
abc Node splitting criteria

(https://www.analyticsvidhya.com/wp-content/uploads/2016/02/tree-infographic.png)

The parameters used for defining a tree are further explained below. The parameters below are irrespective of tool. It is important to understand the role of parameters us modeling. These parameters are available in R & Python.

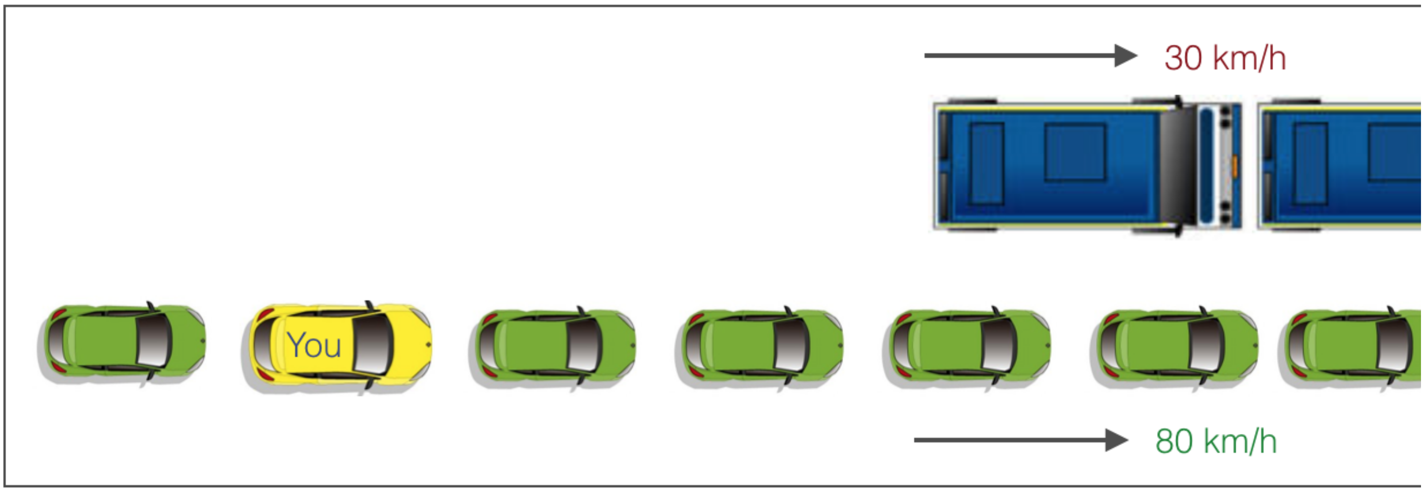
1. **Minimum samples for a node split**
    - Defines the minimum number of samples (or observations) which are required in a no considered for splitting.
    - Used to control over-fitting. Higher values prevent a model from learning relations w be highly specific to the particular sample selected for a tree.
    - Too high values can lead to under-fitting hence, it should be tuned using CV.
2. **Minimum samples for a terminal node (leaf)**
    - Defines the minimum samples (or observations) required in a terminal node or leaf.
    - Used to control over-fitting similar to min_samples_split.
    - Generally lower values should be chosen for imbalanced class problems because th which the minority class will be in majority will be very small.
3. **Maximum depth of tree (vertical depth)**

- The maximum depth of a tree.
- Used to control over-fitting as higher depth will allow model to learn relations very s particular sample.
- Should be tuned using CV.

4. **Maximum number of terminal nodes**
   - The maximum number of terminal nodes or leaves in a tree.
   - Can be defined in place of max_depth. Since binary trees are created, a depth produce a maximum of 2^n leaves.
5. **Maximum features to consider for split**
   - The number of features to consider while searching for a best split. These will be ranc selected.
   - As a thumb-rule, square root of the total number of features works great but we sh upto 30-40% of the total number of features.
   - Higher values can lead to over-fitting but depends on case to case.

Tree Pruning

As discussed earlier, the technique of setting constraint is a greedy-approach. In other w check for the best split instantaneously and move forward until one of the specifie condition is reached. Let's consider the following case when you're driving:



(https://www.analyticsvidhya.com/wp-content/uploads/2016/04/graphic.png)

There are 2 lanes:

1. A lane with cars moving at 80km/h
2. A lane with trucks moving at 30km/h

At this instant, you are the yellow car and you have 2 choices:

1. Take a left and overtake the other 2 cars quickly
2. Keep moving in the present lane

Lets analyze these choice. In the former choice, you'll immediately overtake the car reach behind the truck and start moving at 30 km/h, looking for an opportunity to move ba cars originally behind you move ahead in the meanwhile. This would be the optimum ch objective is to maximize the distance covered in next say 10 seconds. In the later choic through at same speed, cross trucks and then overtake maybe depending on situation ahe you!

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. On the other hand if we use pruning, we in effect look at a few steps ahead and make a choice.

So we know pruning is better. But how to implement it in decision tree? The idea is simple.

1. We first make the decision tree to a large depth.
2. Then we start at the bottom and start removing leaves which are giving us negative returns when compared from the top.
3. Suppose a split is giving us a gain of say -10 (loss of 10) and then the next split on that gives us a gain of 20. A simple decision tree will stop at step 1 but in pruning, we will see that the overall gain is +10 and keep both leaves.

Note that sklearn's decision tree classifier does not currently support pruning. Advancec like xgboost have adopted tree pruning in their implementation. But the library *rpart* in R, function to prune. Good for R users!

5. Are tree based models better than linear models?

"If I can use logistic regression for classification problems and linear regression for regressic problems, why is there a need to use trees"? Many of us have this question. And, this is a va

Actually, you can use any algorithm. It is dependent on the type of problem you are solvin at some key factors which will help you to decide which algorithm to use:

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

6. Working with Decision Trees in R and Python

For R users and Python users, decision tree is quite easy to implement. Let's quickly look a codes which can get you started with this algorithm. For ease of use, I've shared stan where you'll need to replace your data set name and variables to get started.

For R users, there are multiple packages available to implement decision tree such as ctre tree etc.

```
> library(rpart)
> x <- cbind(x_train,y_train)
# grow tree
> fit <- rpart(y_train ~ ., data = x,method="class")
> summary(fit)
#Predict Output
> predicted= predict(fit,x_test)
```

In the code above:

- y_train – represents dependent variable.
- x_train – represents independent variable
- x – represents training data.

For Python users, below is the code:

```
#Import Library
#Import other necessary libraries like pandas, numpy...
from sklearn import tree
#Assumed you have, X (predictor) and Y (target) for training data set and x_test(pre
f test_dataset
# Create tree object
model = tree.DecisionTreeClassifier(criterion='gini') # for classification, here you
ge the algorithm as gini or entropy (information gain) by default it is gini
# model = tree.DecisionTreeRegressor() for regression
# Train the model using the training sets and check score
model.fit(X, y)
model.score(X, y)
#Predict Output
predicted= model.predict(x_test)
```

7. What are ensemble methods in tree based modeling ?

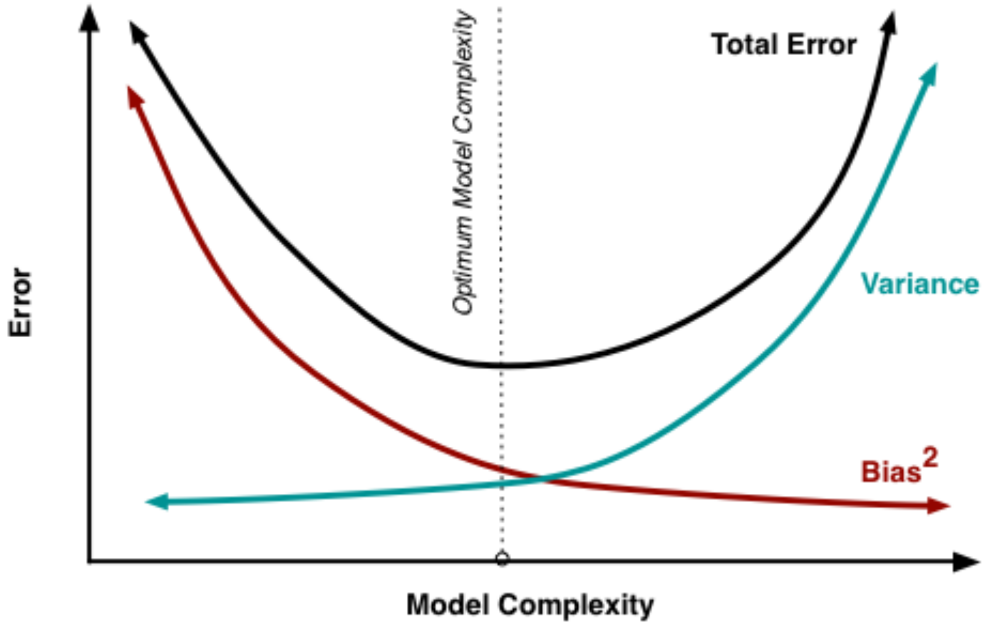
The literary meaning of word 'ensemble' is *group*. Ensemble methods involve group o models to achieve a better accuracy and model stability. Ensemble methods are know supreme boost to tree based models.

Like every other model, a tree based model also suffers from the plague of bias and va means, 'how much on an average are the predicted values different from the actual valu means, 'how different will the predictions of the model be at the same point if differe 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 y to balance the trade off between bias and variance ?

Normally, as you increase the complexity of your model, you will see a reduction in pred due to lower bias in the model. As you continue to make your model more complex, you eı fitting your model and your model will start suffering from high variance.

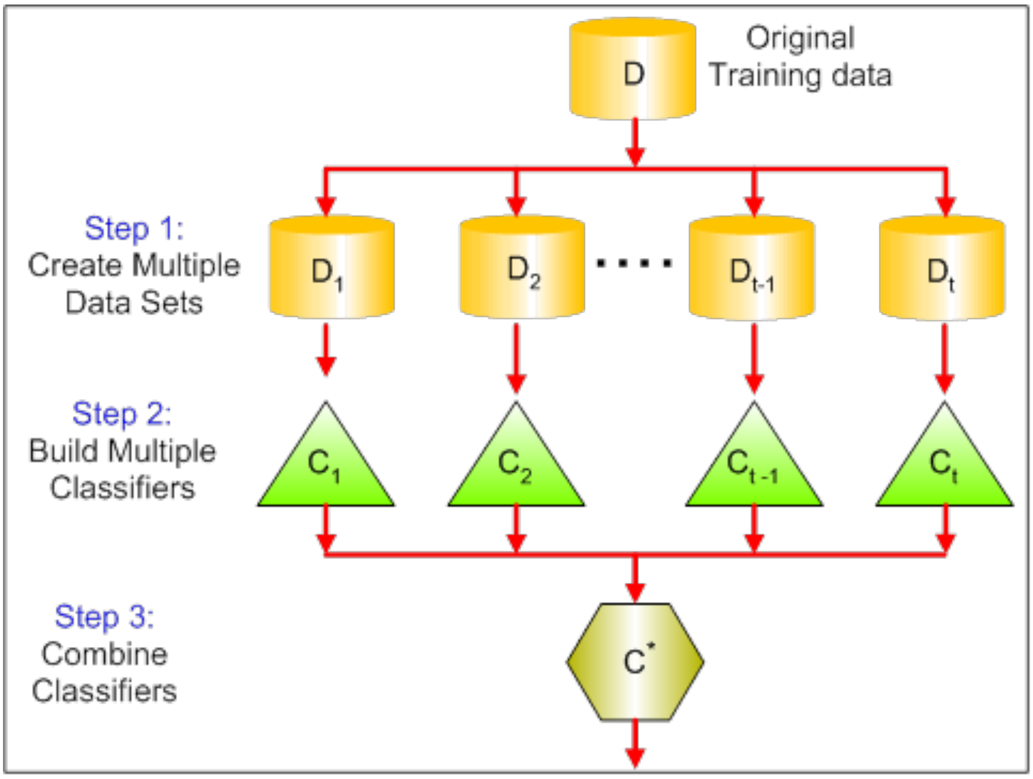
A champion model should maintain a balance between these two types of errors. This is kr **trade-off management** of bias-variance errors. Ensemble learning is one way to execute tł analysis.



(https://www.analyticsvidhya.com/wp-content/uploads/2015/07/model_complexity.png) the commonly used ensemble methods include: Bagging, Boosting and Stacking. In this tu focus on Bagging and Boosting in detail.

8. What is Bagging? How does it work?

Bagging is a technique used to reduce the variance of our predictions by combining tł multiple classifiers modeled on different sub-samples of the same data set. The following make it clearer:



(https://www.analyticsvidhya.com/wp-content/uploads/2015/07/bagging.png)

The steps followed in bagging are:

1. **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 gene parameters in a bagging model
   - Taking row and column fractions less than 1 helps in making robust models, le overfitting
2. **Build Multiple Classifiers:**
   - Classifiers are built on each data set.
   - Generally the same classifier is modeled on each data set and predictions are made.
3. **Combine Classifiers:**
   - The predictions of all the classifiers are combined using a mean, median or mode valu depending on the problem at hand.
   - The combined values are generally more robust than a single model.

Note that, here the number of models built is not a hyper-parameters. Higher number of always better or may give similar performance than lower numbers. It can be theoretically the variance of the combined predictions are reduced to 1/n (n: number of classifiers) of variance, under some assumptions.

There are various implementations of bagging models. Random forest is one of them and v discuss it next.

9. What is Random Forest ? How does it work?

Random Forest is considered to be a *panacea* of all data science problems. On a funny you can't think of any algorithm (irrespective of situation), use random forest!

Random Forest is a versatile machine learning method capable of performing both regr classification tasks. It also undertakes dimensional reduction methods, treats missing val values and other essential steps of data (https://www.analyticsvidhya.com/blog/2015/02/data-exploration-preparation-model/), a fairly good job. It is a type of ensemble learning method, where a group of weak models c form a powerful model.

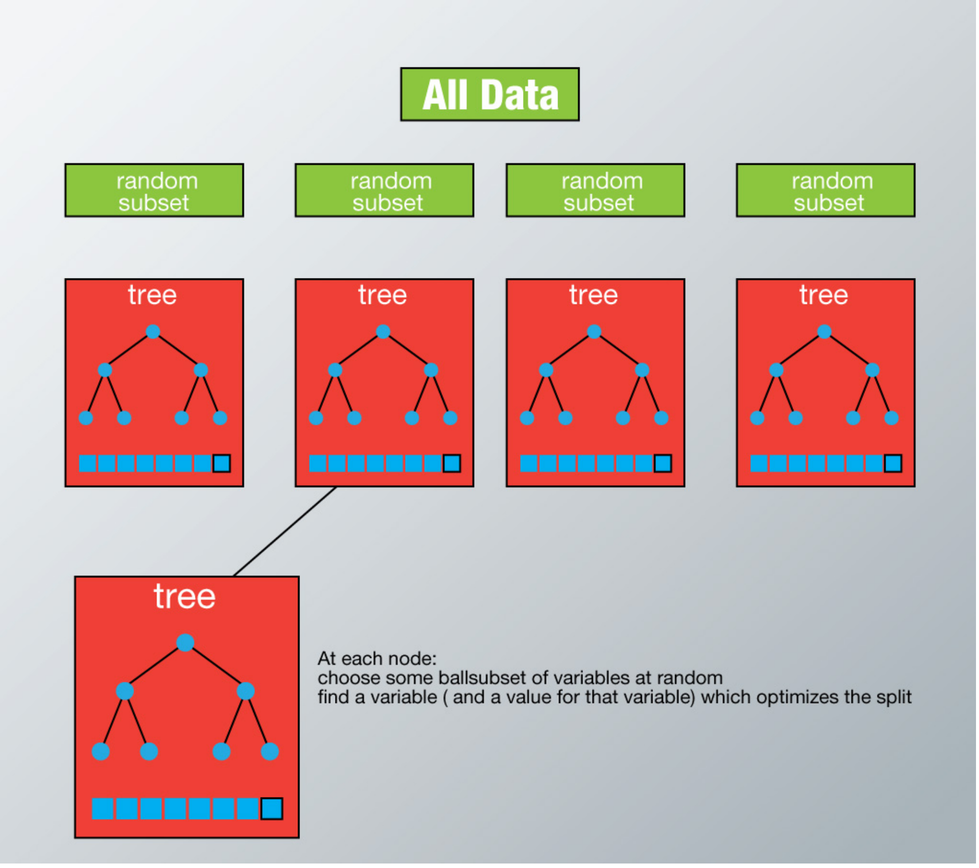
How does it work?

In Random Forest, we grow multiple trees as opposed to a single tree in CART r comparison between CART and Random Forest here, (https://www.analyticsvidhya.com/blog/2014/06/comparing-cart-random-forest-1/) a (https://www.analyticsvidhya.com/blog/2014/06/comparing-random-forest-simple-cart-model/)). To classify a new object based on attributes, each tree gives a classification and tree "votes" for that class. The forest chooses the classification having the most votes (c trees in the forest) and in case of regression, it takes the average of outputs by different tree

(https://www.analyticsvidhya.com/wp-content/uploads/2015/09/Forest-Canopy1.png)

It works in the following manner. Each tree is planted & grown as follows:

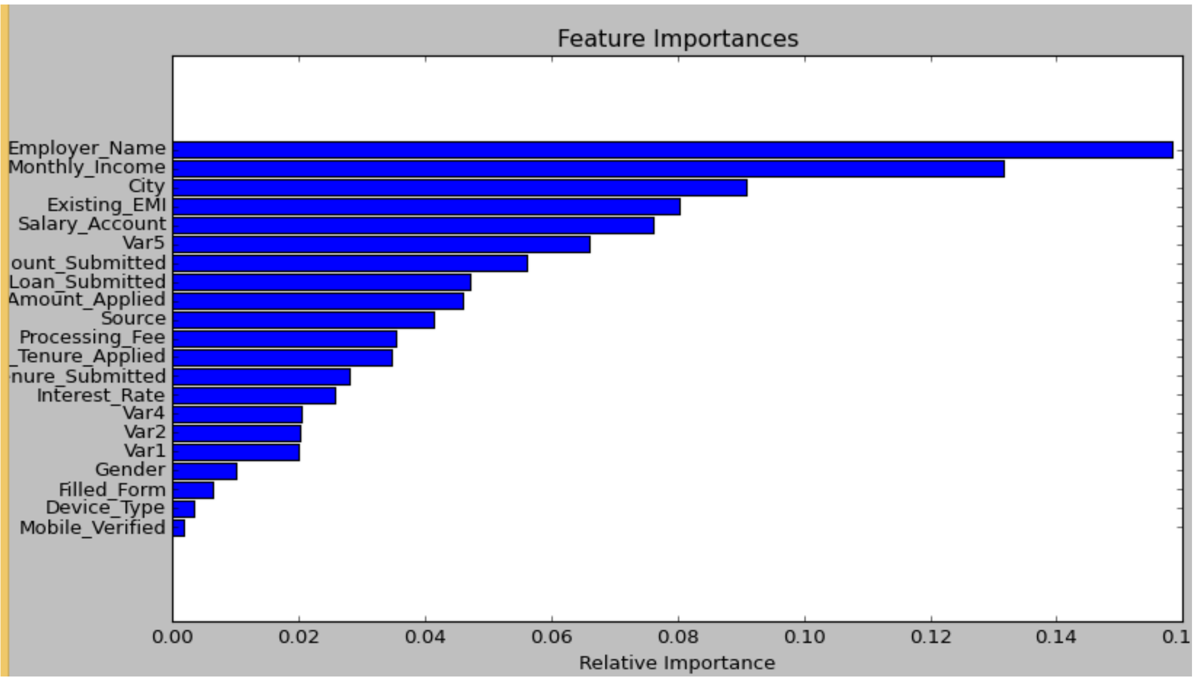
1. Assume number of cases in the training set is N. Then, sample of these N cases is taken at *with replacement*. This sample will be the training set for growing the tree.
2. If there are M input variables, a number m<M is specified such that at each node, m v selected at random out of the M. The best split on these m is used to split the node. The v held constant while we grow the forest.
3. Each tree is grown to the largest extent possible and there is no pruning.
4. Predict new data by aggregating the predictions of the ntree trees (i.e., majority votes for c average for regression).


All Data
random subset
random subset
random subset
random subset
tree
tree
tree
tree
tree
At each node:
choose some ballsubset of variables at random
find a variable (and a value for that variable) which optimizes the split

To understand more in detail about this algorithm using a case study, please read "Introduction to Random forest – (https://www.analyticsvidhya.com/blog/2014/06/introduction-random-forest-simplified/)

Advantages of Random Forest

- This algorithm can solve both type of problems i.e. classification and regression and do estimation at both fronts.
- One of benefits of Random forest which excites me most is, the power of handle large d higher dimensionality. It can handle thousands of input variables and identify most significa so it is considered as one of the dimensionality reduction methods. Further, outputs **Importance of variable,** which can be a very handy feature (on some random data



- It has an effective method for estimating missing data and maintains accuracy when a larg of the data are missing.
- It has methods for balancing errors in data sets where classes are imbalanced.
- The capabilities of the above can be extended to unlabeled data, leading to unsupervise data views and outlier detection.
- Random Forest involves sampling of the input data with replacement called as bootstra Here one third of the data is not used for training and can be used to testing. These are ca **of bag** samples. Error estimated on these out of bag samples is known as *out of bag er* error estimates by Out of bag, gives evidence to show that the out-of-bag estimate is as using a test set of the same size as the training set. Therefore, using the out-of-bag er removes the need for a set aside test set.

Disadvantages of Random Forest

- It surely does a good job at classification but not as good as for regression problem as it d precise continuous nature predictions. In case of regression, it doesn't predict beyond 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 on what the model does. You can at best – try different parameters and random seeds!

Python & R implementation

Random forests have commonly known implementations in R packages and Python scikit-
look at the code of loading random forest model in R and Python below:

Python

```
#Import Library
from sklearn.ensemble import RandomForestClassifier #use RandomForestRegressor for r
problem
#Assumed you have, X (predictor) and Y (target) for training data set and x_test(pre
f test_dataset
# Create Random Forest object
model= RandomForestClassifier(n_estimators=1000)
# Train the model using the training sets and check score
model.fit(X, y)
#Predict Output
predicted= model.predict(x_test)
```

R Code

```
> library(randomForest)
> x <- cbind(x_train,y_train)
# Fitting model
> fit <- randomForest(Species ~ ., x,ntree=500)
> summary(fit)
#Predict Output
> predicted= predict(fit,x_test)
```

10. What is Boosting ? How does it work?

Definition: The term 'Boosting' refers to a family of algorithms which converts weak learne learners.

Let's understand this definition in detail by solving a problem of spam email identification:

How would you classify an email as SPAM or not? Like everyone else, our initial approach v identify 'spam' and 'not spam' emails using following criteria. If:

1. Email has only one image file (promotional image), It's a SPAM
2. Email has only link(s), It's a SPAM
3. Email body consist of sentence like "You won a prize money of $ xxxxxx", It's a SPAM
4. Email from our official domain "Analyticsvidhya.com (http://analyticsvidhya.com/)" , Not a S
5. Email from known source, Not a SPAM

Above, we've defined multiple rules to classify an email into 'spam' or 'not spam'. But, d these rules individually are strong enough to successfully classify an email? No.

Individually, these rules are not powerful enough to classify an email into 'spa spam'. Therefore, these rules are called as **weak learner**.

To convert weak learner to strong learner, we'll combine the prediction of each weak learne methods like:

- Using average/ weighted average
- Considering prediction has higher vote

For example: Above, we have defined 5 weak learners. Out of these 5, 3 are voted as 'SPAN voted as 'Not a SPAM'. In this case, by default, we'll consider an email as SPAM becaus higher(3) vote for 'SPAM'.

How does it work?

Now we know that, boosting combines weak learner a.k.a. base learner to form a stro immediate question which should pop in your mind is, '*How boosting identify weak rules?*'

To find weak rule, we apply base learning (ML) algorithms with a different distribution. Eacl learning algorithm is applied, it generates a new weak prediction rule. This is an iterati After many iterations, the boosting algorithm combines these weak rules into a sir prediction rule.

Here's another question which might haunt you, '*How do we choose different distributic round?*'

For choosing the right distribution, here are the following steps:

Step 1: The base learner takes all the distributions and assign equal weight or attenti observation.

Step 2: If there is any prediction error caused by first base learning algorithm, then we attention to observations having prediction error. Then, we apply the next base learning alg

Step 3: Iterate Step 2 till the limit of base learning algorithm is reached or higher accuracy is

Finally, it combines the outputs from weak learner and creates a strong learner which improves the prediction power of the model. Boosting pays higher focus on examples whi classified or have higher errors by preceding weak rules.
There are many boosting algorithms which impart additional boost to model's accura tutorial, we'll learn about the two most commonly used algorithms i.e. Gradient Boosting XGboost.

11. Which is more powerful: GBM or Xgboost?

I've always admired the boosting capabilities that xgboost algorithm. At times, I've fo provides better result compared to GBM implementation, but at times you might find tha are just marginal. When I explored more about its performance and science behind its high discovered many advantages of Xgboost over GBM:

1. **Regularization:**
   - Standard GBM implementation has no re (https://www.analyticsvidhya.com/blog/2015/02/avoid-over-fitting-regularization/) like XGBoost, therefore it also helps to reduce overfitting.
   - In fact, XGBoost is also known as '**regularized boosting**' technique.

2. **Parallel Processing:**
    - XGBoost implements parallel processing and is **blazingly faster** as compared to GBM
    - But hang on, we know that boosting (https://www.analyticsvidhya.com/blog/201 introduction-boosting-algorithms-machine-learning/) is sequential process so hov parallelized? We know that each tree can be built only after the previous one, so wl from making a tree using all cores? I hope you get where I'm coming from. Che (http://zhanpengfang.github.io/418home.html) out to explore further.
    - XGBoost also supports implementation on Hadoop.
3. **High Flexibility**
    - XGBoost allow users to define **custom optimization objectives and evaluation criter**
    - This adds a whole new dimension to the model and there is no limit to what we can d
4. **Handling Missing Values**
    - XGBoost has an in-built routine to handle missing values.
    - User is required to supply a different value than other observations and pass that as a XGBoost tries different things as it encounters a missing value on each node and l path to take for missing values in future.
5. **Tree Pruning:**
    - A GBM would stop splitting a node when it encounters a negative loss in the split. Th of a **greedy algorithm**.
    - XGBoost on the other hand make **splits upto the max_depth** specified and then st the tree backwards and remove splits beyond which there is no positive gain.
    - Another advantage is that sometimes a split of negative loss say -2 may be followed positive loss +10. GBM would stop as it encounters -2. But XGBoost will go deeper ar a combined effect of +8 of the split and keep both.
6. **Built-in Cross-Validation**
    - XGBoost allows user to run a **cross-validation at each iteration** of the boosting proce it is easy to get the exact optimum number of boosting iterations in a single run.
    - This is unlike GBM where we have to run a grid-search and only a limited values can b
7. **Continue on Existing Model**
    - User can start training an XGBoost model from its last iteration of previous run. Thi significant advantage in certain specific applications.
    - GBM implementation of sklearn also has this feature so they are even on this point.

12. Working with GBM in R and Python

Before we start working, let's quickly understand the important parameters and the this algorithm. This will be helpful for both R and Python users. Below is the overall pseu GBM algorithm for 2 classes:

```
1. Initialize the outcome
2. Iterate from 1 to total number of trees
  2.1 Update the weights for targets based on previous run (higher for the ones mis-
d)
  2.2 Fit the model on selected subsample of data
  2.3 Make predictions on the full set of observations
  2.4 Update the output with current results taking into account the learning rate
3. Return the final output.
```

This is an extremely simplified (probably naive) explanation of GBM's working. But, it will beginners to understand this algorithm.

Lets consider the important GBM parameters used to improve model performance in Pytho

1. **learning_rate**
   - This determines the impact of each tree on the final outcome (step 2.4). GBM works with an initial estimate which is updated using the output of each tree. The learning controls the magnitude of this change in the estimates.
   - Lower values are generally preferred as they make the model robust to t characteristics of tree and thus allowing it to generalize well.
   - Lower values would require higher number of trees to model all the relations computationally expensive.
2. **n_estimators**
   - The number of sequential trees to be modeled (step 2)
   - Though GBM is fairly robust at higher number of trees but it can still overfit at a point. should be tuned using CV for a particular learning rate.
3. **subsample**
   - The fraction of observations to be selected for each tree. Selection is done by randon
   - Values slightly less than 1 make the model robust by reducing the variance.
   - Typical values ~0.8 generally work fine but can be fine-tuned further.

Apart from these, there are certain miscellaneous parameters which affect overall functiona

1. **loss**
   - It refers to the loss function to be minimized in each split.

- It can have various values for classification and regression case. Generally the de work fine. Other values should be chosen only if you understand their impact on the r

2. **init**
   - This affects initialization of the output.
   - This can be used if we have made another model whose outcome is to be used as th estimates for GBM.
3. **random_state**
   - The random number seed so that same random numbers are generated every time.
   - This is important for parameter tuning. If we don't fix the random number, ther different outcomes for subsequent runs on the same parameters and it become compare models.
   - It can potentially result in overfitting to a particular random sample selected. We car models for different random samples, which is computationally expensive and g used.
4. **verbose**
   - The type of output to be printed when the model fits. The different values can be:
     - 0: no output generated (default)
     - 1: output generated for trees in certain intervals
     - >1: output generated for all trees
5. **warm_start**
   - This parameter has an interesting application and can help a lot if used judicially.
   - Using this, we can fit additional trees on previous fits of a model. It can save a lot of ti should explore this option for advanced applications
6. **presort**
   - Select whether to presort data for faster splits.
   - It makes the selection automatically by default but it can be changed if needed.

I know its a long list of parameters but I have simplified it for you in an excel file whic download from this GitHub (https://github.com/aarshayj/Analytics_Vidhya/tree/master/Articles/Parameter_Tuning_(

For R users, using caret package, there are 3 main tuning parameters:

1. *n.trees* – It refers to number of iterations i.e. tree which will be taken to grow the trees
2. *interaction.depth* – It determines the complexity of the tree i.e. total number of splits it has on a tree (starting from a single node)
3. *shrinkage* – It refers to the learning rate. This is similar to learning_rate in python (shown ab
4. n.minobsinnode – It refers to minimum number of training samples required in a node splitting

GBM in R (with cross validation)

I've shared the standard codes in R and Python. At your end, you'll be required to change t dependent variable and data set name used in the codes below. Considering th implementing GBM in R, one can easily perform tasks like cross validation and grid searc package.

```
> library(caret)
```

```
> fitControl <- trainControl(method = "cv",
                             number = 10, #5folds)
```

```
> tune_Grid <-  expand.grid(interaction.depth = 2,
                             n.trees = 500,
                             shrinkage = 0.1,
                             n.minobsinnode = 10)
```

```
> set.seed(825)
> fit <- train(y_train ~ ., data = train,
                 method = "gbm",
                 trControl = fitControl,
                 verbose = FALSE,
                 tuneGrid = gbmGrid)
```

```
> predicted= predict(fit,test,type= "prob")[,2]
```

GBM in Python

```
#import libraries
```

```
from sklearn.ensemble import GradientBoostingClassifier #For Classification (http://
arn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html#sk
emble.GradientBoostingClassifier)
from sklearn.ensemble import GradientBoostingRegressor #For Regression (http://sciki
rg/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html#sklearn.
GradientBoostingRegressor)
```

```
#use GBM function
```

```
clf = GradientBoostingClassifier(n_estimators=100, learning_rate=1.0, max_depth=1)
clf.fit(X_train, y_train)
```

13. Working with XGBoost in R and Python

XGBoost (eXtreme Gradient Boosting) is an advanced implementation of gradien algorithm. It's feature to implement parallel computing makes it at least **10 times faster** th gradient boosting implementations. It supports various objective functions, including classification and ranking.

R Tutorial: For R users, this is a complete tutorial on XGboost which explains the parame with codes in R. Check Tutorial (https://www.analyticsvidhya.com/blog/2016/01/xgboost- easy-steps/).

Python Tutorial: For Python users, this is a comprehensive tutorial on XGBoost, good started. Check Tutorial (https://www.analyticsvidhya.com/blog/2016/03/comp parameter-tuning-xgboost-with-codes-python/).

14. Where to practice ?

Practice is the one and true method of mastering any concept. Hence, you need to *start p* you wish to master these algorithms.

Till here, you've got gained significant knowledge on tree based models along with thes implementation. It's time that you start working on them. Here are open practice problems can participate and check your live rankings on leaderboard:

For Regression: Big Mart Sales Prediction (http://datahack.analyticsvidhya.com/contest/pra problem-bigmart-sales-prediction)

For Classification: Loan Prediction (http://datahack.analyticsvidhya.com/contest/practice-p loan-prediction)

End Notes

Tree based algorithm are important for every data scientist to learn. In fact, tree models ar provide the best model performance in the family of whole machine learning algorith tutorial, we learnt until GBM and XGBoost. And with this, we come to the end of this tutorial

We discussed about tree based modeling from scratch. We learnt the important of decisio how that simplistic concept is being used in boosting algorithms. For better understandi suggest you to continue practicing these algorithms practically. Also, do keep note of the associated with boosting algorithms. I'm hoping that this tutorial would enrich you wit knowledge on tree based modeling.

Did you find this tutorial useful ? If you have experienced, what's the best trick you've using tree based models ? Feel free to share your tricks, suggestions and opinions in the section below.

You can test your skills and knowledge. Check out Live Comp (http://datahack.analyticsvidhya.com/contest/all) and compete with b Scientists from all over the world.

Share this:

(https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-python/?share=linkedin

(https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-python/?share=faceboo

(https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-python/?share=google-1&nb=1)

(https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-python/?share=twitter&

(https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-python/?share=pocket8

(https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-python/?share=reddit&r

RELATED

(https://www.analyticsvidhya.com/blog/2017/01/the-most-comprehensive-data-science-learning-plan-for-2017/)
The most comprehensive Data Science learning plan for 2017 (https://www.analyticsvidhya.com/blog/2017/01/the-most-comprehensive-data-science-learning-plan-for-2017/)
January 16, 2017
In "Business Analytics"

(https://www.analyticsvidhya.com/blog/2016/12/top-35-articles-resources-analytics-vidhya-year-2016/)
Top 35 Articles and Resources from Analytics Vidhya for the year 2016 (https://www.analyticsvidhya.com/blog/2016/12/top-35-articles-resources-analytics-vidhya-year-2016/)
December 29, 2016
In "Machine Learning"

(https://www.analyticsvi
blog/2015/06/hackathor
guide-analytics-vidhya/)
The Hackathon Practice (
Analytics Vidhya
(https://www.analyticsvi
blog/2015/06/hackathor
guide-analytics-vidhya/)
June 5, 2015
In "Analytics Vidhya"

TAGS: BAGGING (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/BAGGING/), BIAS (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/BIAS/) (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/BOOSTING/), CLASSIFICATION TREES (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/CLA TREES/), DECISION TREE (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/DECISION-TREE/), ENSEMBLE MODELING

(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/ENSEMBLE-MODELING/), GBM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/GBM/), GR
(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/GRADIENT-BOOSTING/), LINEAR MODELS (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/
MODELS/), LINEAR-REGRESSION (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/LINEAR-REGRESSION/), LOGISTIC REGRESSION
(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/LOGISTIC-REGRESSION/), MACHINE LEARNING (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/
LEARNING/), OVER FITTING (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/OVER-FITTING/), RANDOM FOREST
(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/RANDOM-FOREST/), REGRESSION TREES (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/I
TREES/), UNDER FITTING (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/UNDER-FITTING/), VARIANCE
(HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/VARIANCE/), XGBOOST (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/TAG/XGBOOST/)

Previous Article

Senior Hadoop Developer - Delhi NCR/Bangalore (6 - 8 years of experience) (https://www.analyticsvidhya.com/blog/2016/04/hadoop-developer-delhi-ncrbangalore-2-years-experience/)

Next Articl

Case Study For Freshers (Level : Medium) Call Center Optimizatio (https://www.analyticsvidhya.com/blog/201 study-level-medium-call-center-optimization/

(https://www.analyticsvidhya.com/blog/author/avcontentteam/)

Author

Analytics Vidhya Content Team (https://www.analyticsvidhya.com/blog/author/avcon

Analytics Vidhya Content team

62 COMMENTS

Trinadh Bylipudi says:

APRIL 12, 2016 AT 4:17 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED-

SCRATCH-IN-PYTHON/#COMMENT-109318)

Lovely Manish!

Very inspiring. Your articles are very helpful.

Looking forward to your next,

Trinadh

Analytics Vidhya Content Team says:

APRIL 12, 2016 AT 4:44 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIA MODELING-SCRATCH-IN-PYTHON/#COMMENT-109324)

Glad you found it helpful.
Thanks Trinadh!

Venky says:

APRIL 12, 2016 AT 4:58 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED- SCRATCH-IN-PYTHON/#COMMENT-109326)

Excellent Manish

Analytics Vidhya Content Team says:

APRIL 12, 2016 AT 12:46 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIA MODELING-SCRATCH-IN-PYTHON/#COMMENT-109354)

Thanks Venky

Hulisani says:

APRIL 12, 2016 AT 5:00 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED SCRATCH-IN-PYTHON/#COMMENT-109328)

Can I please have it in pdf or rather can you please make all your tutorials available in p

Analytics Vidhya Content Team says:

APRIL 12, 2016 AT 12:46 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORI
MODELING-SCRATCH-IN-PYTHON/#COMMENT-109353)

Hi Hulisani
I'll soon upload the pdf version of this article. Do keep a check.

SRK says:
APRIL 12, 2016 AT 6:00 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED
SCRATCH-IN-PYTHON/#COMMENT-109333)

Very detailed one Manish. Thank you.!

Analytics Vidhya Content Team says:
APRIL 12, 2016 AT 12:44 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORI
MODELING-SCRATCH-IN-PYTHON/#COMMENT-109352)

Welcome 🙂

Dr.D.K.Samuel says:
APRIL 12, 2016 AT 6:28 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED-
SCRATCH-IN-PYTHON/#COMMENT-109334)

Nice writeup

DR Venugopala Rao Manneni says:
APRIL 12, 2016 AT 7:13 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED-
SCRATCH-IN-PYTHON/#COMMENT-109337)

Very clearly explained .. Good Job

Gianni says:
APRIL 12, 2016 AT 8:46 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED
SCRATCH-IN-PYTHON/#COMMENT-109339)

Good job Manish, thank you.

Darshit Dani says:

APRIL 12, 2016 AT 12:04 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED SCRATCH-IN-PYTHON/#COMMENT-109350)

It was nice

Analytics Vidhya Content Team says:

APRIL 12, 2016 AT 12:44 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORI MODELING-SCRATCH-IN-PYTHON/#COMMENT-109351)

Thanks Darshit.

Dummy says:

APRIL 12, 2016 AT 2:10 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED- SCRATCH-IN-PYTHON/#COMMENT-109359)

Very nice

Joe says:

APRIL 12, 2016 AT 4:49 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED SCRATCH-IN-PYTHON/#COMMENT-109363)

Very clear explanations and examples. I have learned a lot from this. Thankyou.
Do you plan to write something similar on Conditional Logistic Regression, which is an a
find interesting?

Analytics Vidhya Content Team says:

APRIL 13, 2016 AT 1:00 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL MODELING-SCRATCH-IN-PYTHON/#COMMENT-109377)

Welcome Joe. And, thanks for your suggestion. I guess I need to check this to

kishore kumar says:

APRIL 12, 2016 AT 7:26 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED- SCRATCH-IN-PYTHON/#COMMENT-109370)

Amazing teacher you are..thanks for the great work

Analytics Vidhya Content Team says:

APRIL 13, 2016 AT 12:59 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORI
MODELING-SCRATCH-IN-PYTHON/#COMMENT-109376)

Thanks Kishore!

Jermaine says:

APRIL 13, 2016 AT 3:00 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED
SCRATCH-IN-PYTHON/#COMMENT-109382)

You are a really great teacher. Keep up the great work!

Analytics Vidhya Content Team says:

APRIL 13, 2016 AT 1:45 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL
MODELING-SCRATCH-IN-PYTHON/#COMMENT-109403)

means a lot. Thank You!

Veeramani says:

APRIL 13, 2016 AT 7:18 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED-
SCRATCH-IN-PYTHON/#COMMENT-109392)

Good Job Manish

Krishna says:

APRIL 13, 2016 AT 2:38 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED-
SCRATCH-IN-PYTHON/#COMMENT-109406)

Awesome post, Manish! Kudos to you! You are doing such a great service by imparting knowledge to so many!

james says:

APRIL 13, 2016 AT 11:55 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED SCRATCH-IN-PYTHON/#COMMENT-109426)

well described.

Perhaps you wish to tell us how many YEARS of experiment learning that you have that summarize in a few liners …

joe says:

APRIL 14, 2016 AT 12:56 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASEI SCRATCH-IN-PYTHON/#COMMENT-109464)

For your 30 students example it gives a best tree for the data from that particular schoc It is not clear how you test that fixed best tree for other data from other schools or wher of playing cricket, or not, is not known. How do you then establish how good the model

It seems that trees are biased towards correlating data, rather than establishing causes results for a country, say USA, that did not play much cricket or a school without a crick equipments would give completely misleading answers. So the example tree has really correlated data for a particualr Indian school but not investigated any cause of playing

Choukha Ram Choudhary (https://www.facebook.com/app_scoped_user_id/42077175143614

APRIL 17, 2016 AT 1:20 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED- SCRATCH-IN-PYTHON/#COMMENT-109613)

It is an All in One tutorial. Really helpful. Thanks a lot.

Shanu Agrawal says:

APRIL 19, 2016 AT 6:20 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED SCRATCH-IN-PYTHON/#COMMENT-109666)

Hi Manish,

This article is very informative. I have doubt in calculation of Gini Index.
You said "1. Calculate Gini for sub-nodes, using formula sum of square of probability for and failure (p^2+q^2)."
But in Rpart related pdf in R , formula for Gini index = p(1-p).
Please correct me if anything wrong in my understanding.

Venkatesh says:

APRIL 19, 2016 AT 9:35 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED SCRATCH-IN-PYTHON/#COMMENT-109688)

Excellent introduction and explanation. You are very good at explaining things and shar
Appreciate your hard work.
Venkatesh

Zhongkai Lv says:

APRIL 20, 2016 AT 12:57 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASEI SCRATCH-IN-PYTHON/#COMMENT-109714)

Good job

Rahul Manchanda says:

APRIL 21, 2016 AT 9:29 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED SCRATCH-IN-PYTHON/#COMMENT-109768)

Awesome!! Makes life so much easier for all of us.

Rajesh Pandit says:

MAY 6, 2016 AT 10:31 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED-M SCRATCH-IN-PYTHON/#COMMENT-110511)

Hi Manish,

Very detailed (both theory and examples). Really appreciate your work on this.
Keep up the good work.

Rajesh

Varun Sharma says:

MAY 9, 2016 AT 8:26 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED-M SCRATCH-IN-PYTHON/#COMMENT-110674)

Hi Manish, Thanks for the awesome post…

Please provide pdf version of this.

Varun

Himanshu Tripathi says:

MAY 13, 2016 AT 1:21 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED-M SCRATCH-IN-PYTHON/#COMMENT-110937)

Very well drafted article on Decision tree for starters… Its indeed helped me. Thanks Ma look for more 🙂

Analytics Vidhya Content Team says:

MAY 21, 2016 AT 3:43 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL- MODELING-SCRATCH-IN-PYTHON/#COMMENT-111253)

Good to know. Thanks Himanshu!

Amol M says:

MAY 28, 2016 AT 6:36 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED-I SCRATCH-IN-PYTHON/#COMMENT-111542)

hi Manish… very effective and simple explanation on Tree Based Modeling. can you prov with pdf version please ?

Balaji says:

JUNE 19, 2016 AT 1:45 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED-I SCRATCH-IN-PYTHON/#COMMENT-112382)

Thanks for the article! Can someone help me to how to address the below scenario!

Is it advisable to use Classification Tree techniques (CHAID / CART) when the class pro highly skewed.
For e.g. Class A is 98% of the base and Class B is only 2% of the population.

Jhonny says:

JULY 14, 2016 AT 3:05 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED-I SCRATCH-IN-PYTHON/#COMMENT-113427)

Awesome post, thank you!
I would like to know why some people use a tree to caterorize varibles and then with th
categorized variables build a logistic regression?

---

jorge del rio says:

OCTOBER 13, 2016 AT 4:03 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTO BASED-MODELING-SCRATCH-IN-PYTHON/#COMMENT-117145)

Because in some way, a chaid tree defines best/optimal breaks in continuos
using points of break where chi test is more significant.

---

sagar (http://sa) says:

JULY 16, 2016 AT 8:09 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED-I SCRATCH-IN-PYTHON/#COMMENT-113532)

```
from scipy.stats import mode
mode(df['Gender'])

C:\Anaconda3\lib\site-packages\scipy\stats\stats.py:257: RuntimeWarning: The input
not be properly checked for nan values. nan values will be ignored.
"values. nan values will be ignored.", RuntimeWarning)

——————————————————————————
TypeError Traceback (most recent call last)
in ()
—-> 1 mode(df['Gender'])

C:\Anaconda3\lib\site-packages\scipy\stats\stats.py in mode(a, axis, nan_policy)
642 return mstats_basic.mode(a, axis)
643
–> 644 scores = np.unique(np.ravel(a)) # get ALL unique values
645 testshape = list(a.shape)
646 testshape[axis] = 1

C:\Anaconda3\lib\site-packages\numpy\lib\arraysetops.py in unique(ar, return_index
return_inverse, return_counts)
196 aux = ar[perm]
```

```
197 else:
–> 198 ar.sort()
199 aux = ar
200 flag = np.concatenate(([True], aux[1:] != aux[:-1]))

TypeError: unorderable types: str() > float()
```

can anybody help me on python..new in python..what should I do for this error

Rahul Suman says:

JULY 21, 2016 AT 11:58 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED-I SCRATCH-IN-PYTHON/#COMMENT-113777)

Please reply back as soon as possible. Thanks!!

Rajat Agarwal says:

JULY 27, 2016 AT 5:44 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED-I SCRATCH-IN-PYTHON/#COMMENT-114105)

Hi, is the formula ((p^2+q^2).) that you have for calculation of Gini Indx correct? Can you provide reference of a published paper or standard book.

Suyog says:

JULY 27, 2016 AT 10:09 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASED SCRATCH-IN-PYTHON/#COMMENT-114161)

I am trying to use MLLIB on spark to implement decision tree. How do I determine the I without using sklearn ?

Aditya Kumar Singh says:

AUGUST 3, 2016 AT 10:23 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BAS SCRATCH-IN-PYTHON/#COMMENT-114399)

Sir, how to decide the number of trees to get a good result from random forest ??

Nandita says:

AUGUST 9, 2016 AT 8:41 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASE SCRATCH-IN-PYTHON/#COMMENT-114597)

THis is one of the nest explanation. I came across. Thanks a ton.

Is it possible if you coyld talk about the M5 rule based algorithm

Ravi says:

AUGUST 9, 2016 AT 5:56 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASE SCRATCH-IN-PYTHON/#COMMENT-114613)

hi manish, i want to learn more practical approach in R with some example on control c bias , variance and pruning, can u please suggest .
it was nice and beautiful article. i learned a lot as i am new to machine learning. it cleare my confusions on decision tree and RandomForest.

Thank you

shudhan says:

AUGUST 25, 2016 AT 10:55 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTO BASED-MODELING-SCRATCH-IN-PYTHON/#COMMENT-115128)

you can refer to ISLR book for R code..

Thank you

Jochen says:

AUGUST 19, 2016 AT 7:44 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BAS SCRATCH-IN-PYTHON/#COMMENT-114947)

Hi Manish,
your article is one of the best explanation of decisions trees I have read so far. Very goo which make clear the gains of different approaches.
Now some things are clearer for me. Thanks a lot!

Vinay J says:

AUGUST 20, 2016 AT 11:40 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BAS
SCRATCH-IN-PYTHON/#COMMENT-114985)

The fact that I am reading this article at 4 AM and not feeling sleepy even a bit (in fact I
somewhere in the middle) and getting ready to execute code fir my own dataset, show
of this article. Hats off. Looking forward to read all your articles. Thanks a lot

WILSON POWLOUS says:

AUGUST 21, 2016 AT 11:23 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BAS
SCRATCH-IN-PYTHON/#COMMENT-114989)

Hi Manish,

Nicely written, good job

kushal wadhwani says:

AUGUST 24, 2016 AT 11:15 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BAS
SCRATCH-IN-PYTHON/#COMMENT-115089)

Is there a way to get sample to root node mapping

kishore90.raj@gmail.com says:

AUGUST 27, 2016 AT 11:33 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BAS
SCRATCH-IN-PYTHON/#COMMENT-115193)

Manish,
Very well written comprehensively. Thanks for your efforts. So random forest is special
Bagging Ensemble method with classifier as Decision Tree?
Thanks
Kishore

Ashish Yelkar says:

AUGUST 28, 2016 AT 4:41 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BAS
SCRATCH-IN-PYTHON/#COMMENT-115222)

Very simple and nicely written..Good job..

Adithya says:

SEPTEMBER 1, 2016 AT 9:26 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BA
SCRATCH-IN-PYTHON/#COMMENT-115424)

What does CV mean?

xiao ma says:

SEPTEMBER 12, 2016 AT 1:56 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUT
BASED-MODELING-SCRATCH-IN-PYTHON/#COMMENT-115979)

sorray for the wrong ,the meaning of cv is Cross-validation

xiaoma says:

SEPTEMBER 14, 2016 AT 12:12 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TU
BASED-MODELING-SCRATCH-IN-PYTHON/#COMMENT-116061)

Cross-validation

Colleen says:

OCTOBER 4, 2016 AT 6:04 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTOR
BASED-MODELING-SCRATCH-IN-PYTHON/#COMMENT-116780)

CV = cross-validation. It took me a while to figure that one out too.

HARI G B (http://newsydroozy.com) says:

SEPTEMBER 2, 2016 AT 9:26 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BA
MODELING-SCRATCH-IN-PYTHON/#COMMENT-115443)

Spectacular article….Keep it up. Manish

harigalla.indian@gmail.com says:

SEPTEMBER 17, 2016 AT 5:01 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-B
MODELING-SCRATCH-IN-PYTHON/#COMMENT-116152)

Thanks a lot Manish for sharing, I have started learning journey with your site, gradually
confidence
Appreciated your efforts for enhancing knowledge across world

---

Shia talesara says:

OCTOBER 5, 2016 AT 9:17 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BASE
SCRATCH-IN-PYTHON/#COMMENT-116835)

How can you tell if the GBM or Random forest did a good job in predicting the response
What if i have a low Rsquare but an AUC of .70 . Can i assume my model is good in expl
variability of my response categories???

Abdul says:

OCTOBER 15, 2016 AT 8:48 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BAS
SCRATCH-IN-PYTHON/#COMMENT-117178)

Yes, indeed very informative.

Malcolm Dmello says:

OCTOBER 18, 2016 AT 6:46 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BAS
SCRATCH-IN-PYTHON/#COMMENT-117258)

Hi Manish,

Thanks for a wonderful tutorial. Is there a way to get the scored probability per student
state that a particular student has X% of playing cricket.

Zunqiu says:

OCTOBER 19, 2016 AT 9:18 PM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BAS
SCRATCH-IN-PYTHON/#COMMENT-117316)

This is a great article! very detailed and understandable compared to other introductio
methods. Please post more like this! appreciate!

SAI ATTALURI says:

DECEMBER 18, 2016 AT 7:27 AM (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/2016/04/COMPLETE-TUTORIAL-TREE-BA
SCRATCH-IN-PYTHON/#COMMENT-119904)

Excellent

ABOUT US

For those of you, who are wondering what is "Analytics Vidhya", "Analytics" can be defined as the science of extracting insights from raw data. The spectrum of analytics starts from capturing data and evolves into using insights / trends from this data to make informed decisions. Read More (http://www.analyticsvidhya.com/about-me/)

STAY CONNECTED

8,028
FOLLOWERS
(http://www.twitter.com/analyticsvidhya)

26,148
FOLLOWERS
(http://www.facebook.com/An[illegible]a)

1,579
FOLLOWERS
(https://plus.google.com/+Analyticsvidhya)

Email
SUBSCRIBE
(https://feedburner.google.com/fb/a/mailverify?uri=analyticsvidhya)

LATEST POSTS

(https://www.analyticsvidhya.com/blog.
plan-2017-beginners-data-science/)

Infographic - Learning Plan 2 beginners in data science (https://www.analyticsvidhy 2017/01/learning-plan-2017-b data-science/)

KUNAL JAIN , JANUARY 28, 2017

(https://www.analyticsvidhya.com/blog.
data-science-plan-for-2017/)

Infographic - Learning Plan 2 Transitioners in data science (https://www.analyticsvidhy 2017/01/transitioners-data-s for-2017/)

KUNAL JAIN , JANUARY 28, 2017

Infographic - Learning Plan 2

(https://www.analyticsvidhya.com/blog.
plan-2017-intermediates-data-science/)

Intermediates in data scienc
(https://www.analyticsvidhy
2017/01/learning-plan-2017-
intermediates-data-science/

KUNAL JAIN , JANUARY 28, 2017

(https://www.analyticsvidhya.com/blog.
to-structuring-customer-complaints/)

Introduction to Structuring (
complaints explained with e:
(https://www.analyticsvidhy
2017/01/introduction-to-stru
customer-complaints/)

YOGESH KULKARNI , JANUARY 27, 20