DRONACHARYA COLLEGE OF ENGINEERING GURUGRAM

Unit 2

Logistic regression is a technique used when the dependent variable is categorical (or nominal). Examples: 1) Consumers make a decision to buy or not to buy, 2) a product may pass or fail quality control, 3) there are good or poor credit risks, and 4) employee may be promoted or not.

Binary logistic regression - determines the impact of multiple independent variables presented simultaneously to predict membership of one or other of the two dependent variable categories.

Since the dependent variable is dichotomous we cannot predict a numerical value for it using logistic regression so the usual regression least squares deviations criteria for best fit approach of minimizing error around the line of best fit is inappropriate (It's impossible to calculate deviations using binary variables!).

Instead, logistic regression employs binomial probability theory in which there are only two values to predict: that probability (p) is 1 rather than 0, i.e. the event/person belongs to one group rather than the other.

Logistic regression forms a best fitting equation or function using the maximum likelihood (ML) method, which maximizes the probability of classifying the observed data into the appropriate category given the regression coefficients.

Like multiple regression, logistic regression provides a coefficient 'b', which measures each independent variable's partial contribution to variations in the dependent variable.

The goal is to correctly predict the category of outcome for individual cases using the most parsimonious model.

To accomplish this goal, a model (i.e. an equation) is created that includes all predictor variables that are useful in predicting the response variable.

10.2 The Purpose of Binary Logistic Regression

1. The logistic regression predicts group membership

- Since logistic regression calculates the probability of success over the probability of failure, the results of the analysis are in the form of an odds ratio.
- Logistic regression determines the impact of multiple independent variables presented simultaneously to predict membership of one or other of the two dependent variable categories.

2. The logistic regression also provides the relationships and strengths among the variables Assumptions of (Binary) Logistic Regression

- Logistic regression does not assume a linear relationship between the dependent and independent variables.
  - Logistic regression assumes linearity of independent variables and log odds of dependent variable.

- The independent variables need not be interval, nor normally distributed, nor linearly related, nor of equal variance within each group

- Homoscedasticity is not required. The error terms (residuals) do not need to be normally distributed.
- The dependent variable in logistic regression is not measured on an interval or ratio scale.

- The dependent variable must be a dichotomous (2 categories) for the binary logistic regression.
- The categories (groups) as a dependent variable must be mutually exclusive and exhaustive; a case can only be in one group and every case must be a member of one of the groups.

- Larger samples are needed than for linear regression because maximum coefficients using a ML method are large sample estimates. A minimum of 50 cases per predictor is recommended (Field, 2013)

- Hosmer, Lemeshow, and Sturdivant (2013) suggest a minimum sample of 10 observations per independent variable in the model, but caution that 20 observations per variable should be sought if possible.

- Leblanc and Fitzgerald (2000) suggest a minimum of 30 observations per independent variable.

10.3 Log Transformation

The log transformation is, arguably, the most popular among the different types of transformations used to transform skewed data to approximately conform to normality.

- Log transformations and sq. root transformations moved skewed distributions closer to normality. So what we are about to do is common.

This log transformation of the p values to a log distribution enables us to create a link with the normal regression equation. The log distribution (or logistic transformation of p) is also called the logit of p or logit(p).

In logistic regression, a logistic transformation of the odds (referred to as logit) serves as the depending variable:

$\log(\text{odds}) = \text{logit}(P) = \ln\left(\frac{P}{1-P}\right)$ If we take the above dependent variable and add a regression equation for the independent variables, we get a logistic regression:

$\text{logit}(p) = a + b_1x_1 + b_2x_2 + b_3x_3 + \ldots$ As in least-squares regression, the relationship between the logit(P) and X is assumed to be linear.

10.4 Equation

$P=\exp(a+b_1x_1+b_2x_2+b_3x_3+\ldots)1+\exp(a+b_1x_1+b_2x_2+b_3x_3+\ldots)P=\exp(a+b1x1+b2x2+b3x3+\ldots)1+\exp(a+b1x1+b2x2+b3x3+\ldots)$In the equation above: *P* can be calculated with the following formula
where :

P = the probability that a case is in a particular category,

exp = the exponential function (approx. 2.72),

a = the constant (or intercept) of the equation and,

b = the coefficient (or slope) of the predictor variables.

10.5 Hypothesis Test

In logistic regression, hypotheses are of interest:

- **the null hypothesis**, which is when all the coefficients in the regression equation take the value zero, and
- **the alternate hypothesis** that the model currently under consideration is accurate and differs significantly from the null of zero, i.e. gives significantly better than the chance or random prediction level of the null hypothesis.

10.6 Likelihood Ratio Test for Nested Models

The likelihood ratio test is based on -2LL ratio. It is a test of the significance of the difference between the likelihood ratio (-2LL) for the researcher's model with predictors (called model chi square) minus the likelihood ratio for baseline model with only a constant in it.

Significance at the .05 level or lower means the researcher's model with the predictors is significantly different from the one with the constant only (all 'b' coefficients being zero). It measures the improvement in fit that the explanatory variables make compared to the null model.

Chi square is used to assess significance of this ratio.

11.1 Introduction to Multinomial Logistic Regression

Logistic regression is a technique used when the dependent variable is categorical (or nominal). For Binary logistic regression the number of dependent variables is two, whereas the number of dependent variables for multinomial logistic regression is more than two. Examples: Consumers make a decision to buy or not to buy, a product may pass or fail quality control, there are good or poor credit risks, and employee may be promoted or not.

11.2 Equation

In logistic regression, a logistic transformation of the odds (referred to as logit) serves as the depending variable:

log(odds)=logit(P)=ln(P1−P)=a+b1x1+b2x2+b3x3+…log⁡(odds)=logit⁡(P)=ln⁡(P1−P)=a+b1x1+b2x2+b3x3+…
or

p=exp(a+b1X1+b2X2+b3X3+…)1+exp(a+b1X1+b2X2+b3X3+…)p=exp⁡(a+b1X1+b2X2+b3X3+…)1+exp⁡(a+b1X1+b2X2+b3X3+…)> Where:
p = the probability that a case is in a particular category,

exp = the exponential (approx. 2.72),

a = the constant of the equation and,

b = the coefficient of the predictor or independent variables.

Logits or Log Odds

- Odds value can range from 0 to infinity and tell you how much more likely it is that an observation is a member of the target group rather than a member of the other group.
  - Odds = p/(1-p)
- If the probability is 0.80, the odds are 4 to 1 or .80/.20; if the probability is 0.25, the odds are .33 (.25/.75).
- The odds ratio (OR), estimates the change in the odds of membership in the target group for a one unit increase in the predictor. It is calculated by using the regression coefficient of the predictor as the exponent or exp.
- Assume in the example earlier where we were predicting accountancy success by a maths competency predictor that b = 2.69. Thus the odds ratio is exp(2.69) or 14.73. Therefore the odds of passing are 14.73 times greater for a student for example who had a pre-test score of 5 than for a student whose pre-test score was 4.

11.3 Hypothesis Test of Coefficients

In logistic regression, hypotheses are of interest:

- The null hypothesis, which is when all the coefficients in the regression equation take the value zero, and
- The alternate hypothesis that the model currently under consideration is accurate and differs significantly from the null of zero, i.e. gives significantly better than the chance or random prediction level of the null hypothesis.

Evaluation of Hypothesis

We then work out the likelihood of observing the data we actually did observe under each of these hypotheses. The result is usually a very small number, and to make it easier to handle, the *natural logarithm* is used, producing a *log likelihood (LL)*. Probabilities are

always less than one, so LL's are always negative. Log likelihood is the basis for tests of a logistic model.

11.4 Likelihood Ratio Test

The likelihood ratio test is based on -2LL ratio. It is a test of the significance of the difference between the likelihood ratio (-2LL) for the researcher's model with predictors (called model chi square) minus the likelihood ratio for baseline model with only a constant in it.

Significance at the .05 level or lower means the researcher's model with the predictors is significantly different from the one with the constant only (all 'b' coefficients being zero). It measures the improvement in fit that the explanatory variables make compared to the null model.

Chi square is used to assess significance of this ratio (see Model Fitting Information in SPSS output).

- H0H0: There is no difference between null model and final model.
- H1H1: There is difference between null model and final model.

11.5 Checking AssumptionL: Multicollinearity

Just run "linear regression" after assuming categorical dependent variable as continuous variable

- If the largest VIF (Variance Inflation Factor) is greater than 10 then there is cause of concern (Bowerman & O'Connell, 1990)
- Tolerance below 0.1 indicates a serious problem.
- Tolerance below 0.2 indicates a potential problem (Menard,1995).
- If the Condition index is greater than 15 then the multicollinearity is assumed.

11.6 Features of Multinomial logistic regression

Multinomial logistic regression to predict membership of more than two categories. It (basically) works in the same way as binary logistic regression. The analysis breaks the outcome variable down into a series of comparisons between two categories.

E.g., if you have three outcome categories (A, B and C), then the analysis will consist of two comparisons that you choose:

- Compare everything against your first category (e.g. A vs. B and A vs. C),
- Or your last category (e.g. A vs. C and B vs. C),
- Or a custom category (e.g. B vs. A and B vs. C).

The important parts of the analysis and output are much the same as we have just seen for binary logistic regression.

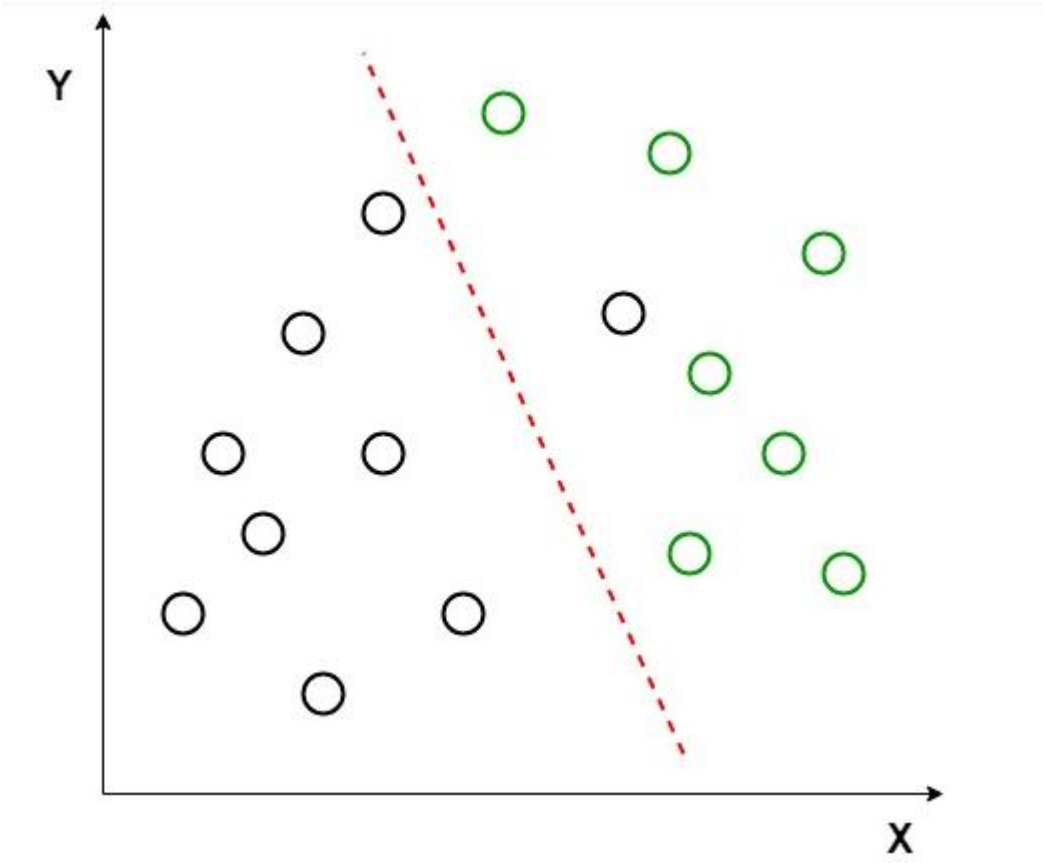
DRONACHARYA COLLEGE OF ENGINEERING GURUGRAM

Linear Discriminant Analysis or **Normal Discriminant Analysis** or **Discriminant Function Analysis** is a dimensionality reduction technique that is commonly used for supervised classification problems. It is used for modelling differences in groups i.e. separating two or more classes. It is used to project the features in higher dimension space into a lower dimension space.

For example, we have two classes and we need to separate them efficiently. Classes can have multiple features. Using only a single feature to classify them may result in some overlapping as shown in the below figure. So, we will keep on increasing the number of features for proper classification.

Example:

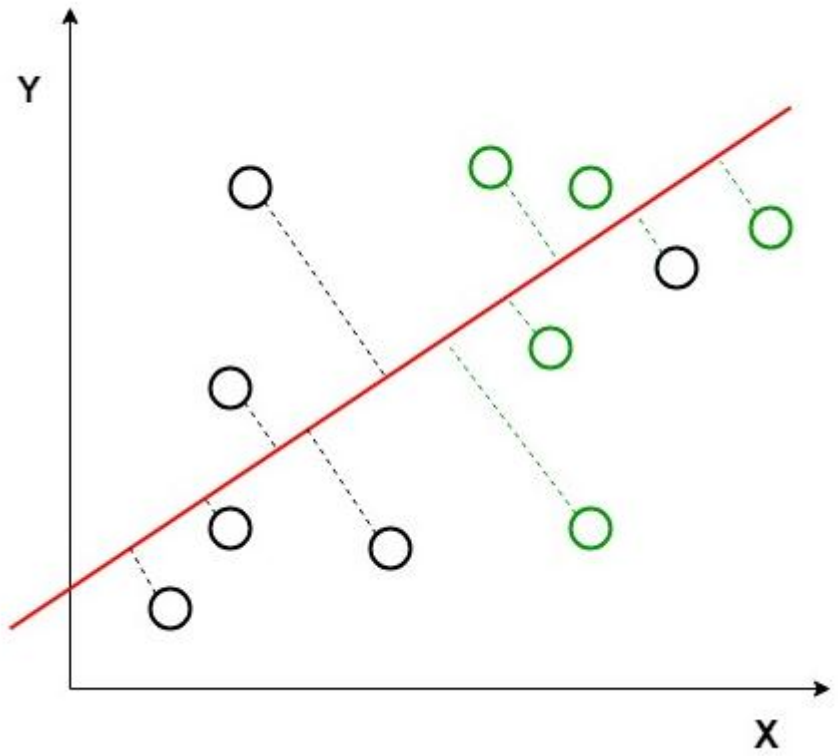
Suppose we have two sets of data points belonging to two different classes that we want to classify. As shown in the given 2D graph, when the data points are plotted on the 2D plane, there's no straight line that can separate the two classes of the data points completely. Hence, in this case, LDA (Linear Discriminant Analysis) is used which reduces the 2D graph into a 1D graph in order to maximize the separability between the two classes.



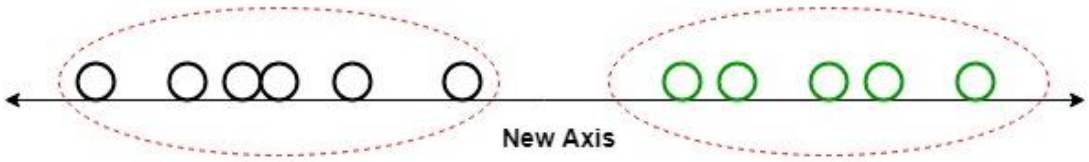
Here, Linear Discriminant Analysis uses both the axes (X and Y) to create a new axis and projects data onto a new axis in a way to maximize the separation of the two categories and hence, reducing the 2D graph into a 1D graph.

Two criteria are used by LDA to create a new axis:

1. Maximize the distance between means of the two classes.
2. Minimize the variation within each class.



In the above graph, it can be seen that a new axis (in red) is generated and plotted in the 2D graph such that it maximizes the distance between the means of the two classes and minimizes the variation within each class. In simple terms, this newly generated axis increases the separation between the data points of the two classes. After generating this new axis using the above-mentioned criteria, all the data points of the classes are plotted on this new axis and are shown in the figure given below.



But Linear Discriminant Analysis fails when the mean of the distributions are shared, as it becomes impossible for LDA to find a new axis that makes both the classes linearly separable. In such cases, we use non-linear discriminant analysis.

DRONACHARYA COLLEGE OF ENGINEERING GURUGRAM

Mathematics

Let's suppose we have two classes and a d- dimensional samples such as x1, x2 … xn, where:

- n1 samples coming from the class (c1) and n2 coming from the class (c2).

If xi is the data point, then its projection on the line represented by unit vector v can be written as vTxi

Let's consider u1 and u2 be the means of samples class c1 and c2 respectively before projection and u1hat denotes the mean of the samples of class after projection and it can be calculated by:

$$\widetilde{\mu_1} = \frac{1}{n_1} \sum_{x_i \in c_1}^{n_1} v^T x_i = v^T \mu_1$$

Similarly,

$$\widetilde{\mu_2} = v^T \mu_2$$

Now, In LDA we need to normalize |\widetilde{\mu_1} -\widetilde{\mu_2} |. Let y_i = v^{T}x_i be the projected samples, then scatter for the samples of c1 is:

$$\widetilde{s_1^2} = \sum_{y_i \in c_1} (y_i - \mu_1)^2$$

Similarly:

$$\widetilde{s_2^2} = \sum_{y_i \in c_1} (y_i - \mu_2)^2$$

Now, we need to project our data on the line having direction v which maximizes

$$J(v) = \frac{\widetilde{\mu_1} - \widetilde{\mu_2}}{\widetilde{s_1^2} + \widetilde{s_2^2}}$$

For maximizing the above equation we need to find a projection vector that maximizes the difference of means of reduces the scatters of both classes. Now, scatter matrix of s1 and s2 of classes c1 and c2 are:

$$s_1 = \sum_{x_i \in c_1} (x_i - \mu_1)(x_i - \mu_1)^T$$

and s2

$$s_2 = \sum_{x_i \in c_2} (x_i - \mu_2)(x_i - \mu_2)^T$$

After simplifying the above equation, we get:

Now, we define, scatter within the classes(s_w) and scatter b/w the classes(s_b):

$$s_w = s_1 + s_2$$

$$s_b = (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T$$

Now, we try to simplify the numerator part of J(v)

$$J(v) = \frac{|\widetilde{\mu_1} - \widetilde{\mu_2}|}{\widetilde{s_1^2} + \widetilde{s_2^2}} = \frac{v^T s_b v}{v^T s_w v}$$

Now, To maximize the above equation we need to calculate differentiation with respect to v

$$\frac{dJ(v)}{dv} = s_b v - \frac{v^t s_b v (s_w v)}{v^T s_w v}$$

$$= s_b v - \lambda s_w v = 0$$

$$s_b v = \lambda s_w v$$

$$s_w^{-1} s_b v = \lambda v$$

$$Mv = \lambda v$$

$where,$

$$\lambda = \frac{v^T s_b v}{v^T s_w v} and$$

$$M = s_w^{-1} s_b$$

Here, for the maximum value of J(v) we will use the value corresponding to the highest eigenvalue. This will provide us the best solution for LDA.

Extensions to LDA:

1. **Quadratic Discriminant Analysis (QDA):** Each class uses its own estimate of variance (or covariance when there are multiple input variables).
2. **Flexible Discriminant Analysis (FDA):** Where non-linear combinations of inputs are used such as splines.
3. **Regularized Discriminant Analysis (RDA):** Introduces regularization into the estimate of the variance (actually covariance), moderating the influence of different variables on LDA.

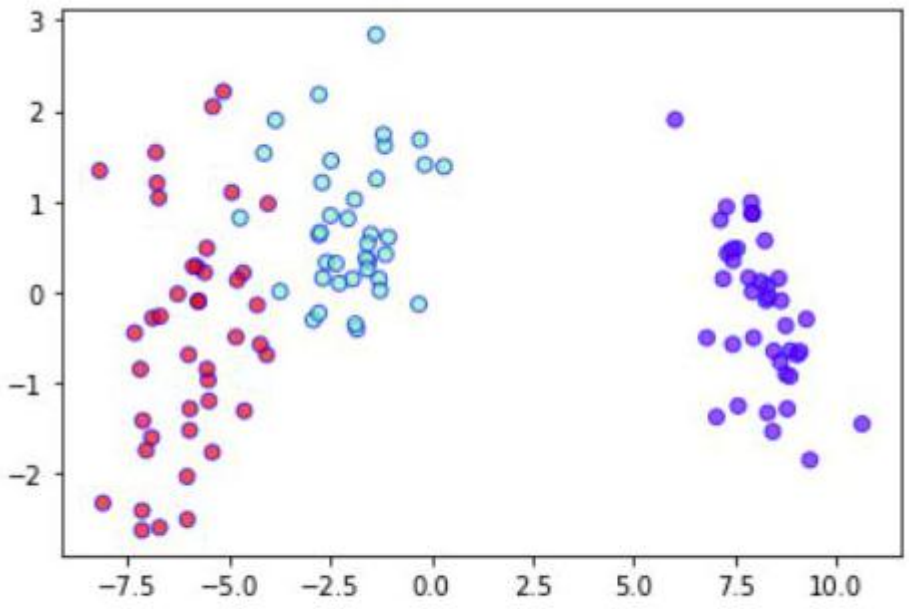
Implementation

- In this implementation, we will perform linear discriminant analysis using the Scikit-learn library on the Iris dataset.

```
# necessary import

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import sklearn
```

```
from sklearn.preprocessing import StandardScaler, LabelEncoder

from sklearn.model_selection import train_test_split

from sklearn.discriminant_analysis import LinearDiscriminantAnalysis

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy_score, confusion_matrix

 # read dataset from URL

url = "https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data"

cls = ['sepal-length', 'sepal-width', 'petal-length', 'petal-width', 'Class']

dataset = pd.read_csv(url, names=cls)

 # divide the dataset into class and target variable

X = dataset.iloc[:, 0:4].values

y = dataset.iloc[:, 4].values

# Preprocess the dataset and divide into train and test

sc = StandardScaler()

X = sc.fit_transform(X)

le = LabelEncoder()

y = le.fit_transform(y)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

# apply Linear Discriminant Analysis

lda = LinearDiscriminantAnalysis(n_components=2)

X_train = lda.fit_transform(X_train, y_train)

X_test = lda.transform(X_test)

# plot the scatterplot

plt.scatter(

    X_train[:,0],X_train[:,1],c=y_train,cmap='rainbow',

  alpha=0.7,edgecolors='b'

)

  # classify using random forest classifier

classifier = RandomForestClassifier(max_depth=2, random_state=0)

classifier.fit(X_train, y_train)

y_pred = classifier.predict(X_test)
```

```
# print the accuracy and confusion matrix

print('Accuracy : ' + str(accuracy_score(y_test, y_pred)))

conf_m = confusion_matrix(y_test, y_pred)

print(conf_m)
```



LDA 2 -variable plot

```
Accuracy : 0.9


[[10  0  0]
 [ 0  9  3]
 [ 0  0  8]]
```

Applications:

1. **Face Recognition:** In the field of Computer Vision, face recognition is a very popular application in which each face is represented by a very large number of pixel values. Linear discriminant analysis (LDA) is used here to reduce the number of features to a more manageable number before the process of classification. Each of the new dimensions generated is a linear combination of pixel values, which form a template. The linear combinations obtained using Fisher's linear discriminant are called Fisher's faces.
2. **Medical:** In this field, Linear discriminant analysis (LDA) is used to classify the patient disease state as mild, moderate, or severe based upon the patient's various parameters and the medical treatment he is going through. This helps the doctors to intensify or reduce the pace of their treatment.
3. **Customer Identification:** Suppose we want to identify the type of customers who are most likely to buy a particular product in a shopping mall. By doing a simple question and answers survey, we can gather all the features of the customers. Here, a Linear discriminant analysis will help us to identify and select the features which can describe the characteristics of the group of customers that are most likely to buy that particular product in the shopping mall.

Linear Discriminant Analysis

Now, Let's consider a classification problem represented by a Bayes Probability distribution P(Y=k | X=x), LDA does it differently by trying to model the distribution of X given the predictors class (I.e. the value of Y) P(X=x| Y=k):

$$P(Y = k|X = x) = \frac{P(X=x|Y=k)P(Y=k)}{P(X=x)}$$

$$= \frac{P(X=x|Y=k)P(Y=k)}{\sum_{j=1}^{K} P(X=x|Y=j)P(Y=j)}$$

In LDA, we assume that P(X | Y=k) can be estimated to the multivariate Normal distribution that is given by following equation:

$$f_k(x) = \frac{1}{(2\pi)^{p/2}|\mathbf{\Sigma}|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^T \mathbf{\Sigma}^{-1}(x-\mu_k)}$$

where, $\mu_k = mean\ of\ the\ examples\ of\ category\ k$
$\sum = covariance\ (we\ assume\ common\ covariance\ for\ all\ categories)$

and P(Y=k) =\pi_k. Now, we try to write the above equation with the assumptions:

$$P(Y = k|X = x) = \frac{\pi_k \frac{1}{(2\pi)^{p/2}|\mathbf{\Sigma}|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^T \mathbf{\Sigma}^{-1}(x-\mu_k)}}{\sum_{j=1}^{K} \frac{1}{(2\pi)^{p/2}|\mathbf{\Sigma}|^{1/2}} e^{-\frac{1}{2}(x-\mu_j)^T \mathbf{\Sigma}^{-1}(x-\mu_j)}}$$

Now, we take log both sides and maximizing the equation, we get the decision boundary:

$$\delta_k(x) = \log \pi_k - \tfrac{1}{2}\mu_k^T \Sigma^{-1} \mu_k + x^T \Sigma^{-1} \mu_k$$

For two classes, the decision boundary is a linear function of x where both classes give equal value, this linear function is given as:

$$\{x : \delta_k(x) = \delta_\ell(x)\}, 1 \leq j, \ell \leq K$$

For multi-class (K>2), we need to estimate the pK means, pK variance, K prior proportions and $\binom{p}{2}K = \left(\frac{p(p-1)}{2}\right)K$. Now, we discuss in more detail about Quadratic Discriminant Analysis.

Quadratic Discriminant Analysis

Quadratic discriminant analysis is quite similar to Linear discriminant analysis except we relaxed the assumption that the mean and covariance of all the classes were equal. Therefore, we required to calculate it separately.

Now, for each of the class y the covariance matrix is given by:

$$\Sigma_y = \frac{1}{N_y - 1} \sum_{y_i = y} (x_i - \mu_y)(x_i - \mu_y)^T$$

By adding the following term and solving (taking log both side and). The quadratic Discriminant function is given by:

$$\delta_k(x) = \log \pi_k - \tfrac{1}{2}\mu_k^T \mathbf{\Sigma}_k^{-1} \mu_k + x^T \mathbf{\Sigma}_k^{-1} \mu_k - \tfrac{1}{2} x^T \Sigma_k^{-1} x - \tfrac{1}{2} \log |\Sigma_k|$$

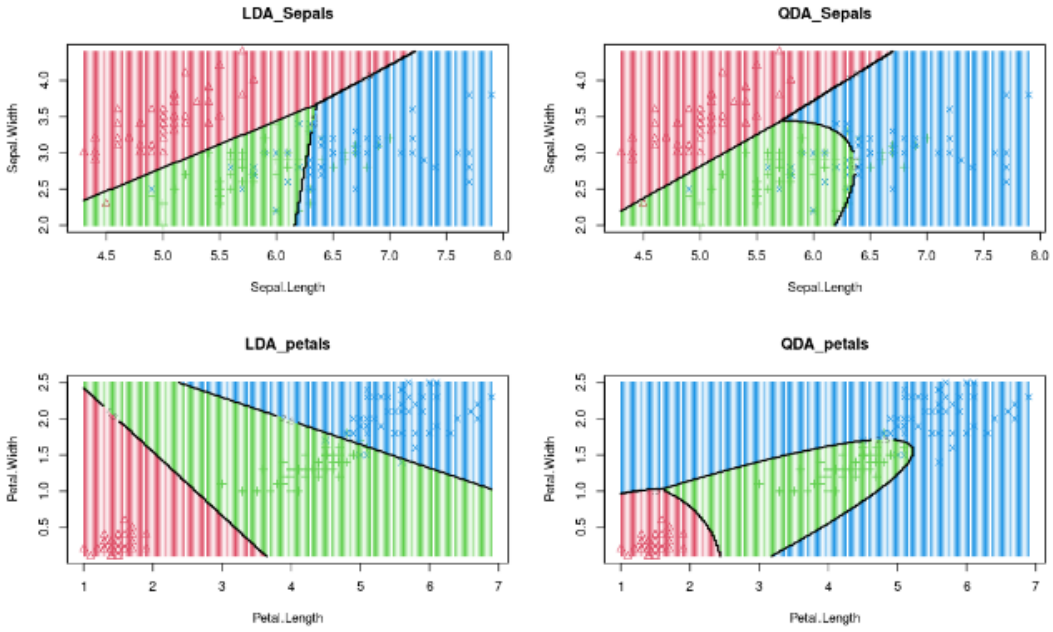
Implementation

- In this implementation, we will be using R and MASS library to plot the decision boundary of Linear Discriminant Analysis and Quadratic Discriminant Analysis. For this, we will use iris dataset:

```
# import libraries

library(caret)

library(MASS)

library(tidyverse)

# Code to plot decision plot

decision_boundary = function(model, data,vars, resolution = 200,...) {

  class='Species'

  labels_var = data[,class]

  k = length(unique(labels_var))

  # For sepals

  if (vars == 'sepal'){

  data = data %>% select(Sepal.Length, Sepal.Width)

  }

  else{

  data = data %>% select(Petal.Length, Petal.Width)

  }

  # plot with color labels

  int_labels = as.integer(labels_var)

  plot(data, col = int_labels+1L, pch = int_labels+1L, ...)

     # make grid

  r = sapply(data, range, na.rm = TRUE)

  xs = seq(r[1,1], r[2,1], length.out = resolution)

  ys = seq(r[1,2], r[2,2], length.out = resolution)

  dfs = cbind(rep(xs, each=resolution), rep(ys, time = resolution))
```

```
  colnames(dfs) = colnames(r)

  dfs = as.data.frame(dfs)

    p = predict(model, dfs, type ='class' )

  p = as.factor(p$class)

   points(dfs, col = as.integer(p)+1L, pch = ".")

    mats = matrix(as.integer(p), nrow = resolution, byrow = TRUE)

  contour(xs, ys, mats, add = TRUE, lwd = 2, levels = (1:(k-1))+.5)

  invisible(mats)

}

par(mfrow=c(2,2))

# run the linear discriminant analysis and plot the decision boundary with
Sepals variable

model = lda(Species ~ Sepal.Length + Sepal.Width, data=iris)

lda_sepals = decision_boundary(model, iris, vars= 'sepal' , main =
"LDA_Sepals")

# run the quadratic discriminant analysis and plot the decision boundary
with Sepals variable

model_qda = qda(Species ~ Sepal.Length + Sepal.Width, data=iris)

qda_sepals = decision_boundary(model_qda, iris, vars= 'sepal', main =
"QDA_Sepals")

 # run the linear discriminant analysis and plot the decision boundary with
Petals variable

model = lda(Species ~ Petal.Length + Petal.Width, data=iris)

lda_petal =decision_boundary(model, iris, vars='petal', main =
"LDA_petals")
```

```
# run the quadratic discriminant analysis and plot the decision boundary
with Petals variable


model_qda = qda(Species ~ Petal.Length + Petal.Width, data=iris)


    qda_petal =decision_boundary(model_qda, iris, vars='petal', main =
    "QDA_petals")
```



LDA and QDA visualization

Regression trees

A regression tree is basically a decision tree that is used for the task of regression which can be used to predict continuous valued outputs instead of discrete outputs.

In Decision Trees for Classification, we saw how the tree asks right questions at the right node in order to give accurate and efficient classifications. The way this is done in Classification Trees is by using 2 measures , namely Entropy and Information Gain. But since we are predicting continuous variables, we cannot calculate the entropy and go through the same process. We need a different measure now. A measure that tells us how much our predictions deviate from the original target and that's the entry-point of mean square error.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

fig 1.1: Mean Square Value

Y is the actual value and Y_hat is the prediction , we only care about how much the prediction varies from the target. Not in which direction. So, we square the difference and divide the entire sum by the total number of records.

In the Regression Tree algorithm, we do the same thing as the Classification trees. But, we try to reduce the Mean Square Error at each child rather than the entropy.

Building a Regression Tree

Let's consider a dataset where we have 2 variables, as shown below

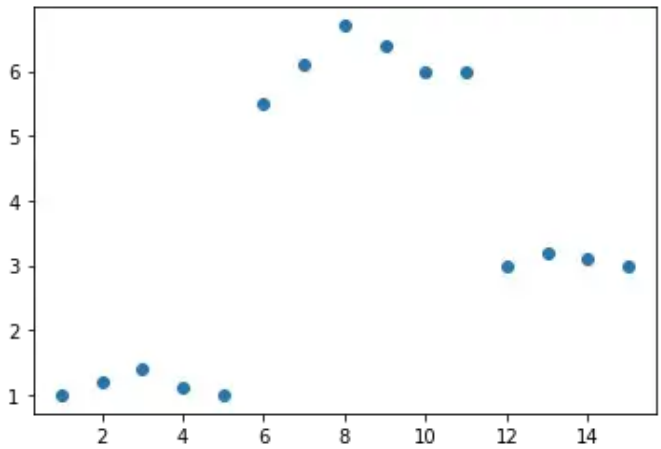


fig 2.1: Dataset, X is a continuous variable and Y is another continuous variable

DRONACHARYA COLLEGE OF ENGINEERING GURUGRAM

X	Y
1	1
2	1.2
3	1.4
4	1.1
5	1
6	5.5
7	6.1
8	6.7
9	6.4
10	6
11	6
12	3
13	3.2
14	3.1

we need to build a Regression tree that best predicts the Y given the X.

Step 1

The first step is to sort the data based on X (*In this case, it is already sorted*). Then, take the average of the first 2 rows in variable X (which is (1+2)/2 = 1.5 according to the given dataset). Divide the dataset into 2 parts *(Part A and Part B)* , separated by $x < 1.5$ and $X \geq 1.5$.

Now, Part A consist only of one point, which is the first row (1,1) and all the other points are in Part — B. Now, take the average of all the Y values in Part A and average of all Y values in Part B separately. These 2 values are the predicted output of the decision tree for $x < 1.5$ and $x \geq 1.5$ respectively. Using the predicted and original values, calculate the mean square error and note it down.

Step 2

In step 1, we calculated the average for the first 2 numbers of sorted X and split the dataset based on that and calculated the predictions. Then, we do the same process again but this time, we calculate the average for the second 2 numbers of sorted X ((2+3)/2 = 2.5). Then, we split the dataset again based on $X < 2.5$ and $X \geq 2.5$ into Part A and Part B again and predict outputs, find mean square error as shown in step 1. This process is repeated for the third 2 numbers, the fourth 2 numbers, the 5th, 6th, 7th till n-1th 2 numbers (*where n is the number of records or rows in the dataset*).

Step 3

Now that we have n-1 mean squared errors calculated , we need to choose the point at which we are going to split the dataset. and that point is the point, which resulted in the lowest mean squared error on splitting at it. In this case, the point is x=5.5. Hence the tree will be split into 2 parts. $x<5.5$ and $x\geq 5.5$. The Root node is selected this way and the data points that go towards the left child and right child of the root node are further recursively exposed to the same algorithm for further splitting.

Brief Explanation of What the algorithm is doing

The basic idea behind the algorithm is to find the point in the independent variable to split the data-set into 2 parts, so that the mean squared error is the minimised at that point. The algorithm does this in a repetitive fashion and forms a tree-like structure.

A regression tree for the above shown dataset would look like this

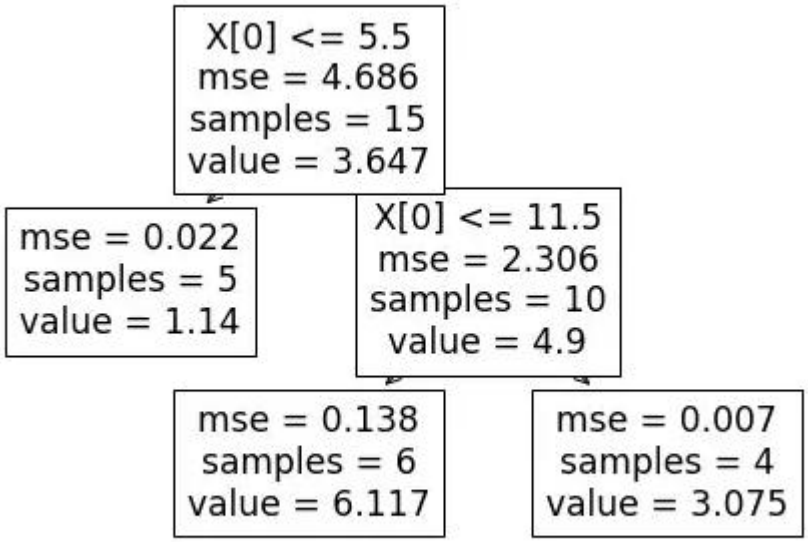


fig 3.1: The resultant Decision Tree

and the resultant prediction visualisation would be this

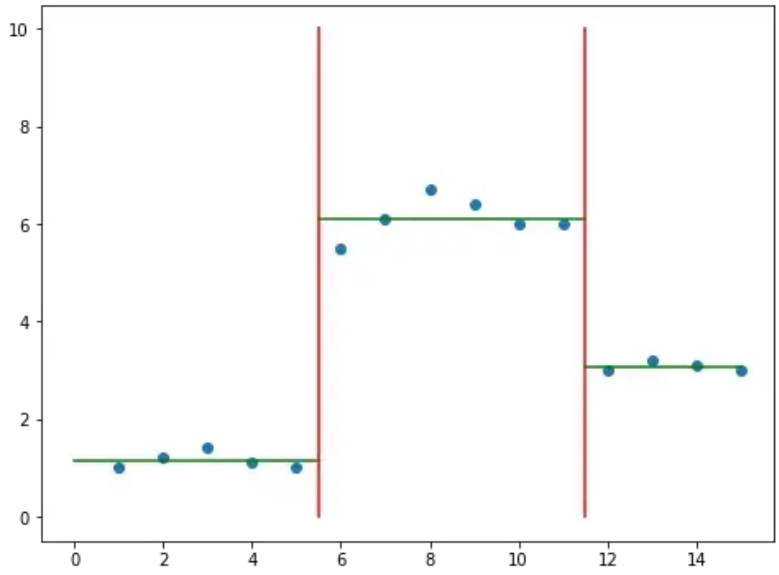


fig 3.2: The Decision Boundary

well, The logic behind the algorithm itself is not rocket science. All we are doing is splitting the data-set by selecting certain points that best splits the data-set and minimises the mean square error. And the way we are selecting these points is by going through an iterative process of calculating mean square error for all the splits and choosing the split that has the least value for the *mse*. So, It only natural this works.

What happens when there are multiple independent variables ?

Let us consider that there are 3 variables similar to the independent variable **X** from fig 2.2. At each node, All the 3 variables would go through the same process as what **X** went through in the above example. The data would be sorted based on the 3 variables separately. The points that minimises the mse are calculated for all the 3 variables. out of the 3 variables and the points calculated for them, the one that has the least mse would be chosen.

How are categorical variables handled ?

When we use the continuous variables as independent variables , we select the point with the least mse using an iterative algorithm as mentioned above. When given a categorical variable , we simply split it by asking a binary question (*usually*). For example, let's say we have a column specifying the size of the tumor in categorical terms. say **Small, Medium** and **Large.**

The tree would split the data-set based on whether **tumor size = small** or **tumor size = large** or **tumor size = Medium** or it can also combine multiple values in some cases, based on whichever question reduces the *mse* the most. and that becomes the top contender for this variable *(Tumor Size).* The Top contenders of the other variables are compared with this and the selection process is similar to the situation mentioned in *"What happens when there are multiple independent variables ?"*

Dealing with Over-Fitting and When to stop building the Tree ?

On reading the previous blogs, one might understand the problem of overfitting and how it affects machine learning models. Regression Trees are prone to this problem.

When we want to reduce the mean square error, the decision tree can recursively split the data-set into a large number of subsets to the the point where a set contains only one row or record. Even though this might reduce the *mse* to zero, this is obviously not a good thing.

This is the famous problem of overfitting and it is a topic of it's own. The basic takeaway is that the models fit to the existing data too perfectly that it fails to generalise with new data. We can use cross validation methods to avoid this.

One way to prevent this, with respect to Regression trees, is to specify the minimum number of records or rows, A leaf node can have, In advance.

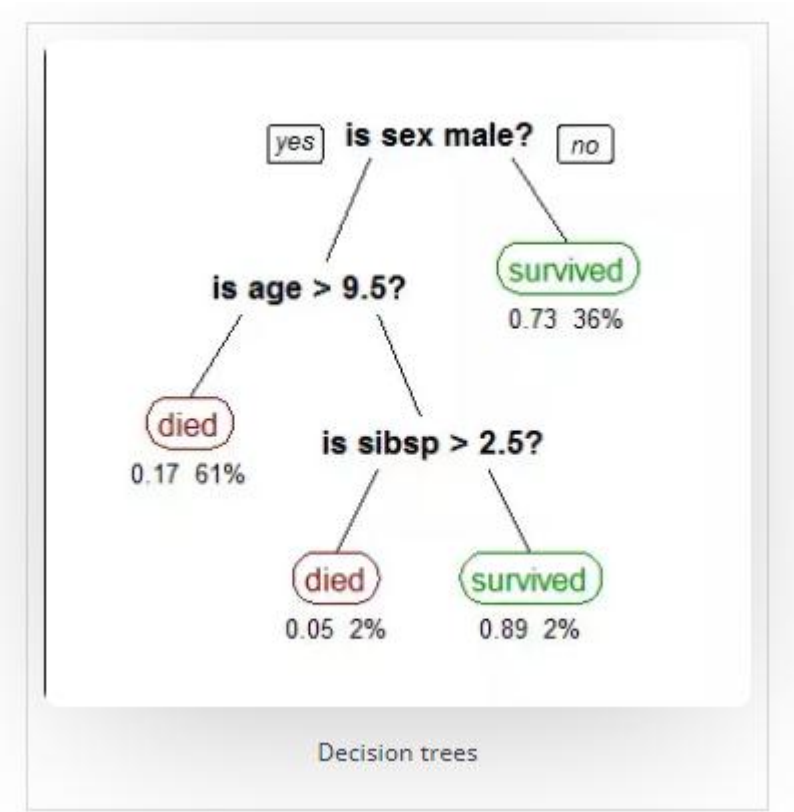
And the exact number is not easily known when it comes to large data-sets. But, cross-validation could be used for this purpose.

Classification and regression trees is a term used to describe decision tree algorithms that are used for classification and regression learning tasks.

The Classification and Regression Tree methodology, also known as the CART were introduced in 1984 by Leo Breiman, Jerome Friedman, Richard Olshen, and Charles Stone. In order to understand classification and regression trees better, we need to first understand decision trees and how they are used. While there are many classification and regression tree ppts and tutorials around, we need to start with the basics.

What are Decision Trees?

If you strip it down to the basics, decision tree algorithms are nothing but if-else statements that can be used to predict a result based on data. For instance, this is a simple decision tree that predicts whether a passenger on the Titanic survived.



Decision trees

Machine learning algorithms can be classified into two types- supervised and unsupervised. A decision tree is a supervised machine learning algorithm. It has a tree-like structure with its root node at the top.

Classification and Regression Trees Tutorial

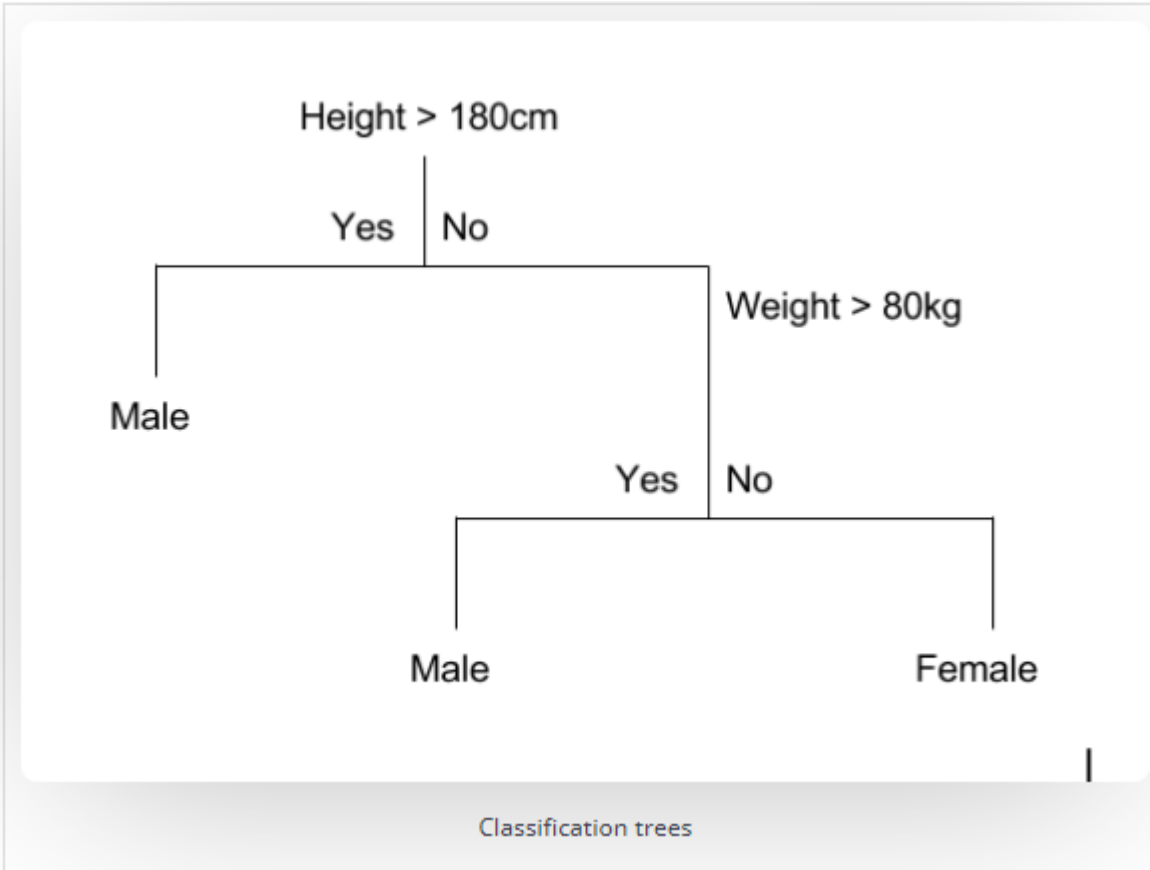
The CART or Classification & Regression Trees methodology refers to these two types of decision trees.

While there are many classification and regression trees tutorials and classification and regression trees ppts out there, here is a simple definition of the two kinds of decision trees. It also includes classification and regression tree examples.

(i) Classification Trees

A classification tree is an algorithm where the target variable is fixed or categorical. The algorithm is then used to identify the "class" within which a target variable would most likely fall.

An example of a classification-type problem would be determining who will or will not subscribe to a digital platform; or who will or will not graduate from high school. These are examples of simple binary classifications where the categorical dependent variable can assume only one of two, mutually exclusive values. In other cases, you might have to predict among a number of different variables. For instance, you may have to predict which type of smartphone a consumer may decide to purchase. In such cases, there are multiple values for the categorical dependent variable. Here's what a classic classification tree looks like.

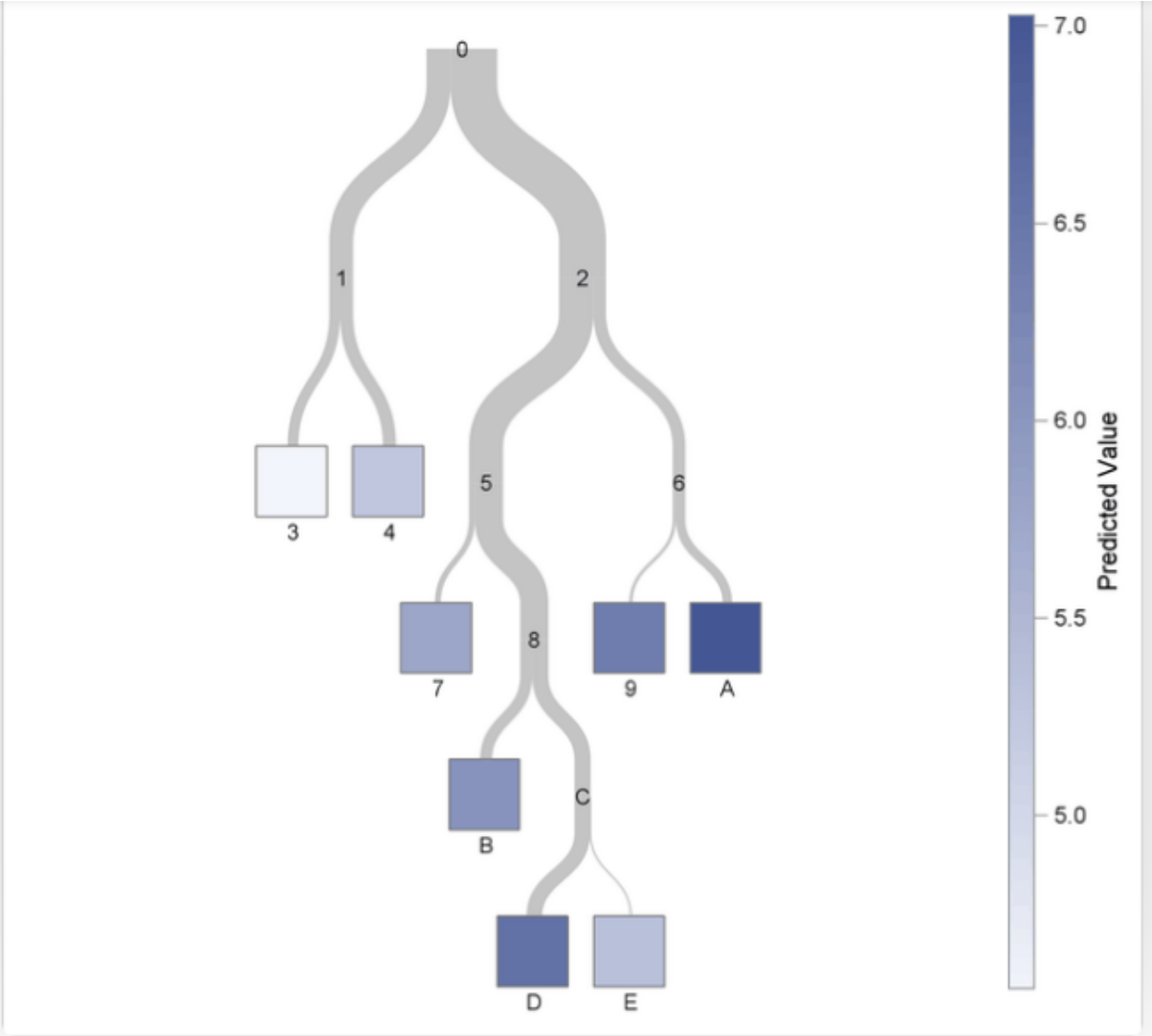


Classification trees

(ii) Regression Trees

A regression tree refers to an algorithm where the target variable is and the algorithm is used to predict its value. As an example of a regression type problem, you may want to predict the selling prices of a residential house, which is a continuous dependent variable.

This will depend on both continuous factors like square footage as well as categorical factors like the style of home, area in which the property is located, and so on.



Regression trees

Difference Between Classification and Regression Trees

Decision trees are easily understood and there are several classification and regression trees ppts to make things even simpler. However, it's important to understand that there are some fundamental differences between classification and regression trees.

When to use Classification and Regression Trees

Classification trees are used when the dataset needs to be split into classes that belong to the response variable. In many cases, the classes Yes or No.

In other words, they are just two and mutually exclusive. In some cases, there may be more than two classes in which case a variant of the classification tree algorithm is used.

Regression trees, on the other hand, are used when the response variable is continuous. For instance, if the response variable is something like the price of a property or the temperature of the day, a regression tree is used.

In other words, regression trees are used for prediction-type problems while classification trees are used for classification-type problems

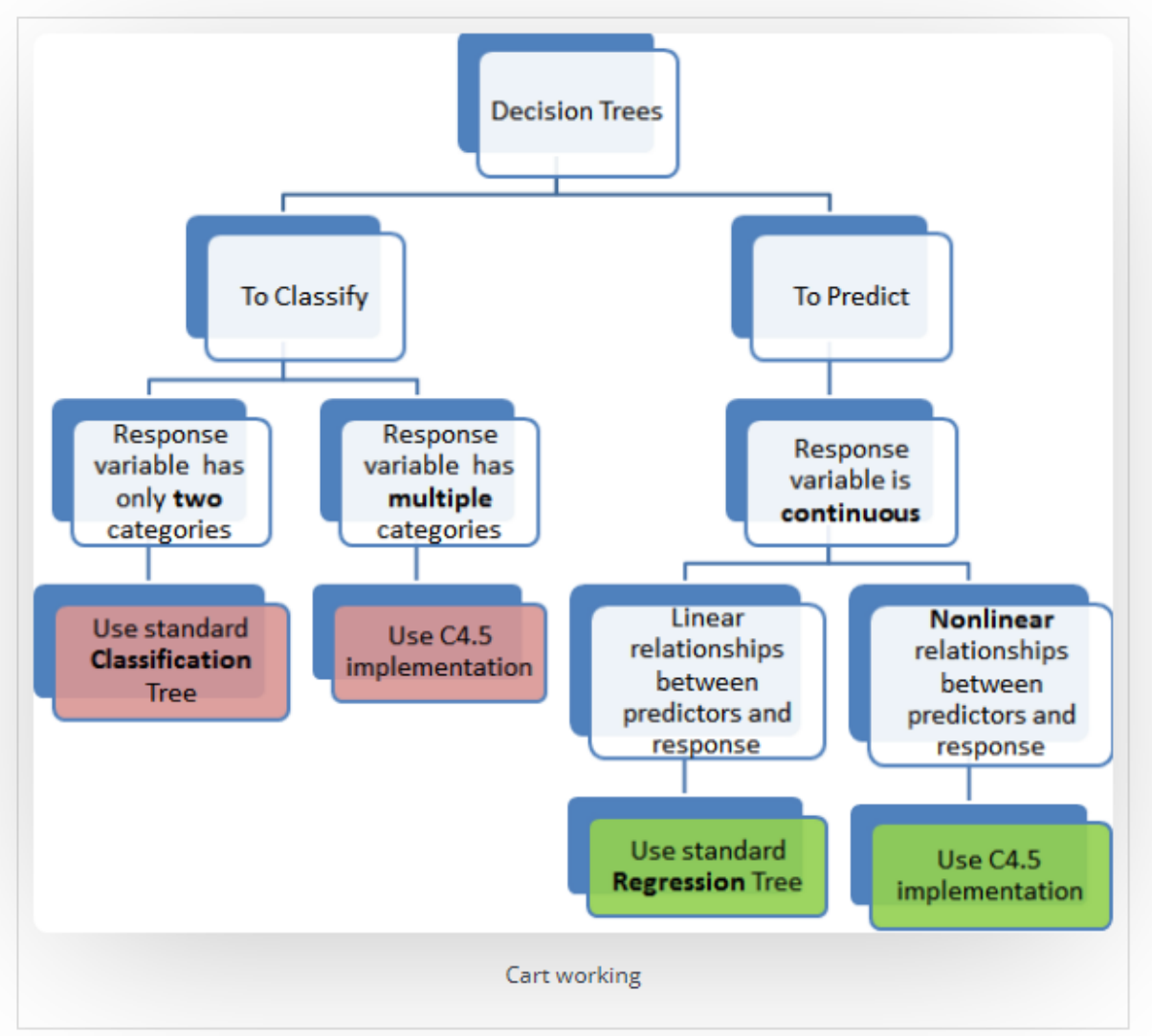
How Classification and Regression Trees Work

A classification tree splits the dataset based on the homogeneity of data. Say, for instance, there are two variables; income and age; which determine whether or not a consumer will buy a particular kind of phone.

If the training data shows that 95% of people who are older than 30 bought the phone, the data gets split there and age becomes a top node in the tree. This split makes the data “95% pure”. Measures of impurity like entropy or Gini index are used to quantify the homogeneity of the data when it comes to classification trees.

In a regression tree, a regression model is fit to the target variable using each of the independent variables. After this, the data is split at several points for each independent variable.

At each such point, the error between the predicted values and actual values is squared to get “A Sum of Squared Errors”(SSE). The SSE is compared across the variables and the variable or point which has the lowest SSE is chosen as the split point. This process is continued recursively.



Advantages of Classification and Regression Trees

The purpose of the analysis conducted by any classification or regression tree is to create a set of if-else conditions that allow for the accurate prediction or classification of a case.

Classification and regression trees work to produce accurate predictions or predicted classifications, based on the set of if-else conditions. They usually have several advantages over regular decision trees.

(i) The Results are Simplistic

The interpretation of results summarized in classification or regression trees is usually fairly simple. The simplicity of results helps in the following ways.

1. It allows for the rapid classification of new observations. That's because it is much simpler to evaluate just one or two logical conditions than to compute scores using complex nonlinear equations for each group.

2. It can often result in a simpler model which explains why the observations are either classified or predicted in a certain way. For instance, business problems are much easier to explain with if-then statements than with complex nonlinear equations.

(ii) Classification and Regression Trees are Nonparametric & Nonlinear

The results from classification and regression trees can be summarized in simplistic if-then conditions. This negates the need for the following implicit assumptions.

1. The predictor variables and the dependent variable are linear.
2. The predictor variables and the dependent variable follow some specific nonlinear link functions.
3. The predictor variables and the dependent variable are monotonic.

Since there is no need for such implicit assumptions, classification and regression tree methods are well suited to data mining. This is because there is very little knowledge or assumptions that can be made beforehand about how the different variables are related.

As a result, classification and regression trees can actually reveal relationships between these variables that would not have been possible using other techniques.

(iii) Classification and Regression Trees Implicitly Perform Feature Selection

Feature selection or variable screening is an important part of analytics. When we use decision trees, the top few nodes on which the tree is split are the most important variables within the set. As a result, feature selection gets performed automatically and we don’t need to do it again.

Limitations of Classification and Regression Trees

Classification and regression tree tutorials, as well as classification and regression tree ppts, exist in abundance. This is a testament to the

classification and regression tree examples where the use of a decision tree has not led to the optimal result. Here are some of the limitations of popularity of these decision trees and how frequently they are used. However, these decision trees are not without their disadvantages.

There are many classification and regression trees.

(i) Overfitting

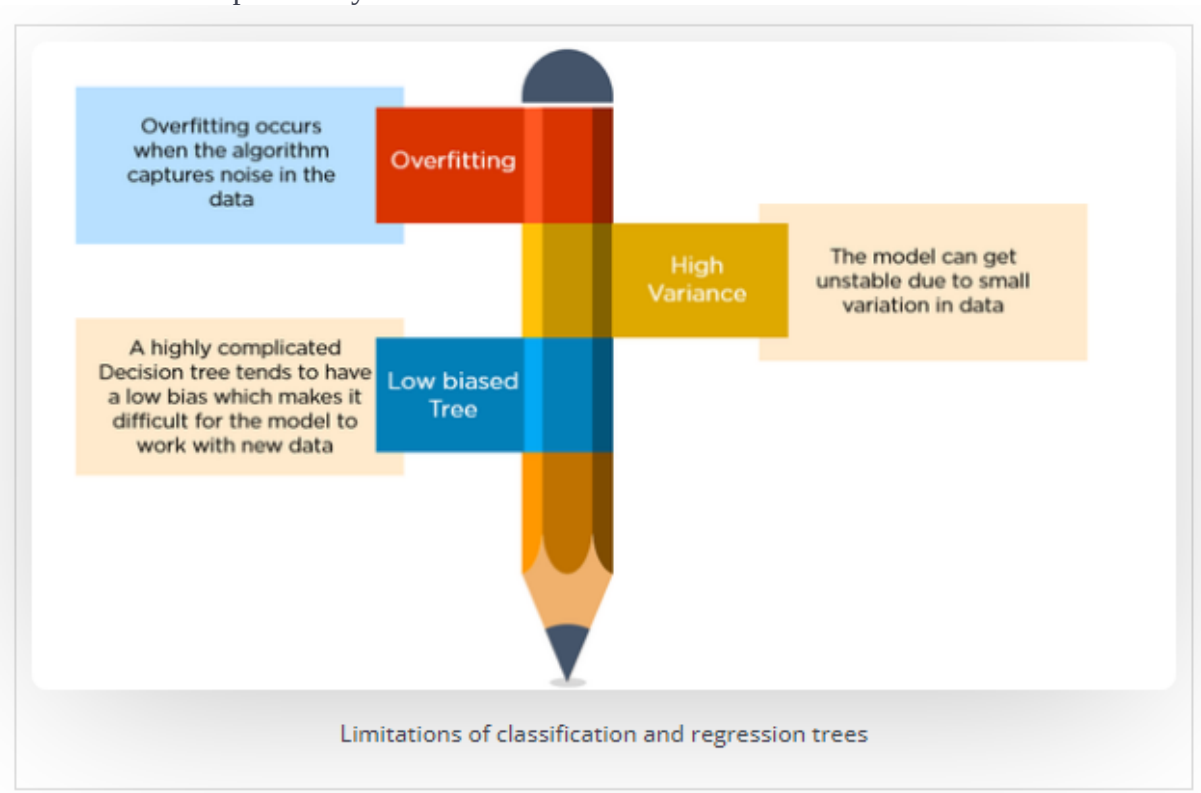
Overfitting occurs when the tree takes into account a lot of noise that exists in the data and comes up with an inaccurate result.

(ii) High variance

In this case, a small variance in the data can lead to a very high variance in the prediction, thereby affecting the stability of the outcome.

(iii) Low bias

A decision tree that is very complex usually has a low bias. This makes it very difficult for the model to incorporate any new data.



Limitations of classification and regression trees

What is a CART in Machine Learning?

A Classification and Regression Tree(CART) is a predictive algorithm used in machine learning. It explains how a target variable's values can be predicted based on other values. It is a decision tree where each fork is split in a predictor variable and each node at the end has a prediction for the target variable. The CART algorithm is an important decision tree algorithm that lies at the foundation of machine learning. Moreover, it is also the basis for other powerful machine learning algorithms like bagged decision trees, random forest, and boosted decision trees.

Pruning Trees

Pruning is a data compression technique in machine learning and search algorithms that reduces the size of decision trees by removing sections of the tree that are non-critical and redundant to classify instances.

> The tree package contains functions prune.tree and cv.tree for pruning trees by cross-validation.
>
> The function prune.tree takes a tree you fit by tree, and evaluates the error of the tree and various prunings of the tree, all the way down to the stump.
>
> The evaluation can be done either on new data, if supplied, or on the training data (the default).
>
> If you ask it for a particular size of tree, it gives you the best pruning of that size.
>
> If you don't ask it for the best tree, it gives an object which shows the number of leaves in the pruned trees, and the error of each one.
>
> This object can be plotted.

Bootstrap Aggregation (Bagging)

Methods such as Decision Trees, can be prone to overfitting on the training set which can lead to wrong predictions on new data. Bootstrap Aggregation (bagging) is a ensembling method that attempts to resolve overfitting for classification or regression problems. Bagging aims to improve the accuracy and performance of machine learning algorithms. It does this by taking random subsets of an original dataset, with replacement, and fits either a classifier (for classification) or regressor (for regression) to each subset. The predictions for each subset are then aggregated through majority vote for classification or averaging for regression, increasing prediction accuracy.

Random forest is a **Supervised Machine Learning Algorithm** that is **used widely in Classification and Regression problems**. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression. One of the most important features of the Random Forest Algorithm is that it can handle the data set containing **continuous variables** as in the case of regression and **categorical variables** as in the case of classification. It performs better results for classification problems.

Real Life Analogy

Let's dive into a real-life analogy to understand this concept further. A student named X wants to choose a course after his 10+2, and he is confused about the choice of course based on his skill set. So he decides to consult various people like his cousins, teachers, parents, degree students, and working people. He asks them varied questions like why he should choose, job opportunities with that course, course fee, etc. Finally, after consulting various people about the course he decides to take the course suggested by most of the people.

Working of Random Forest Algorithm

Before understanding the working of the random forest algorithm in machine learning, we must look into the ensemble technique. ***Ensemble*** simply means combining multiple models. Thus a collection of models is used to make predictions rather than an individual model.

Ensemble uses two types of methods:

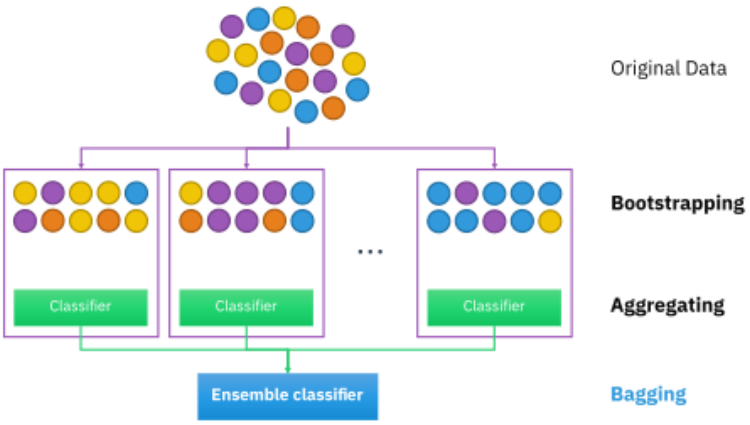
1. **Bagging**– It creates a different training subset from sample training data with replacement & the final output is based on majority voting. For example, Random Forest.

2. **Boosting**– It combines weak learners into strong learners by creating sequential models such that the final model has the highest accuracy. For example, ADA BOOST, XG BOOST

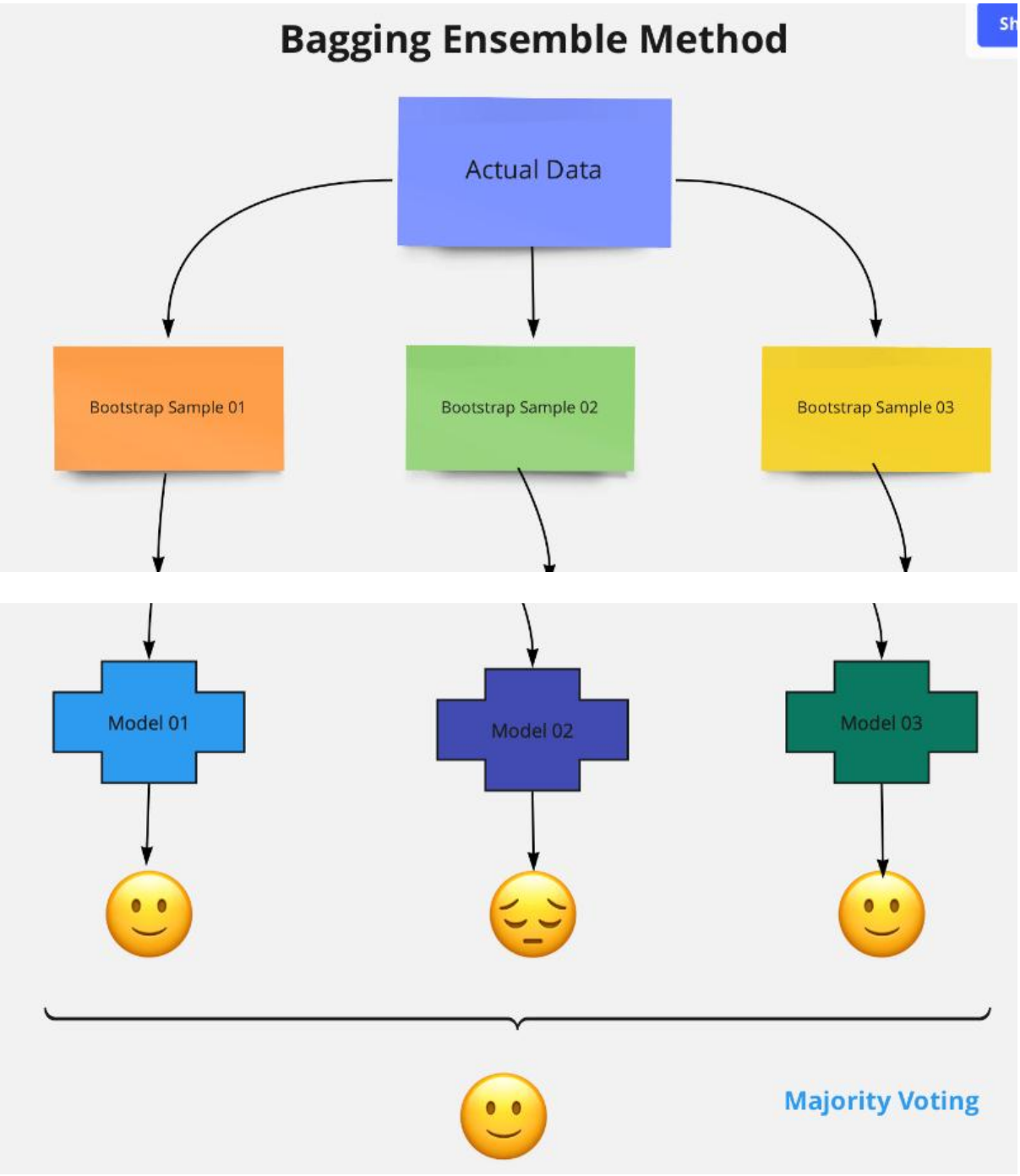
Bagging

Bagging, also known as ***Bootstrap Aggregation*** is the ensemble technique used by random forest. Bagging chooses a random sample from the data set. Hence each model is generated from the samples (Bootstrap Samples) provided by the Original Data with replacement known as ***row sampling***. This step of row sampling with replacement is called ***bootstrap***.

Now each model is trained independently which generates results. The final output is based on majority voting after combining the results of all models. This step which involves combining all the results and generating output based on majority voting is known as ***aggregation***.



Now let's look at an example by breaking it down with the help of the following figure. Here the bootstrap sample is taken from actual data (Bootstrap sample 01, Bootstrap sample 02, and Bootstrap sample 03) with a replacement which means there is a high possibility that each sample won't contain unique data. Now the model (Model 01, Model 02, and Model 03) obtained from this bootstrap sample is trained independently. Each model generates results as shown. Now Happy emoji is having a majority when compared to sad emoji. Thus based on majority voting final output is obtained as Happy emoji.


Bagging Ensemble Method
Actual Data
Bootstrap Sample 01
Bootstrap Sample 02
Bootstrap Sample 03
Model 01
Model 02
Model 03
Majority Voting

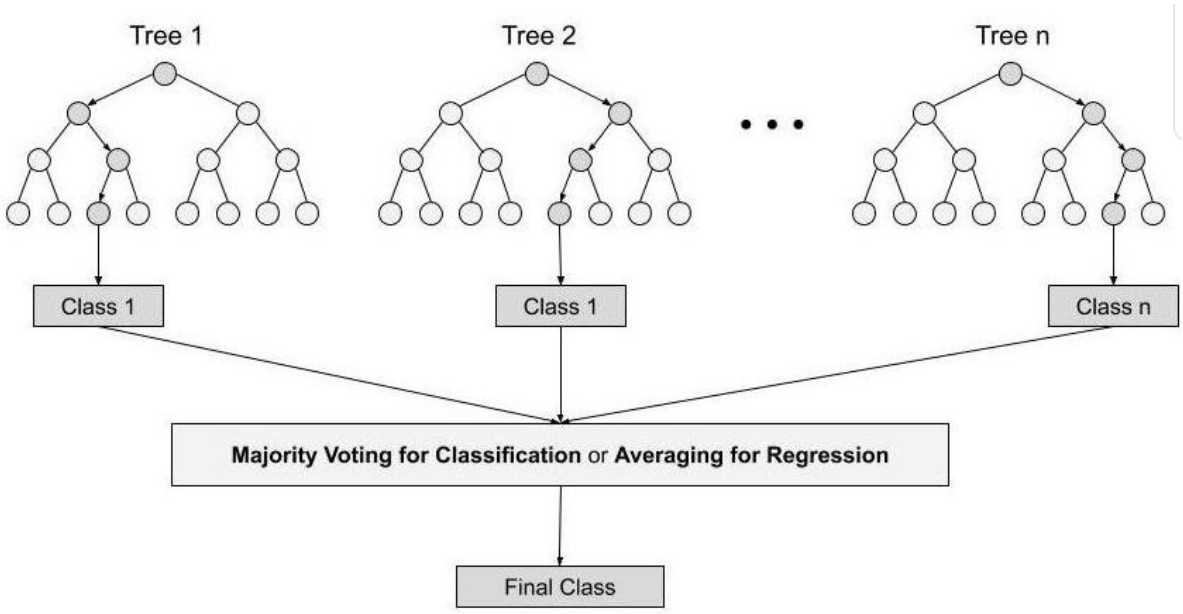
Steps involved in random forest algorithm:

Step 1: In Random forest n number of random records are taken from the data set having k number of records.

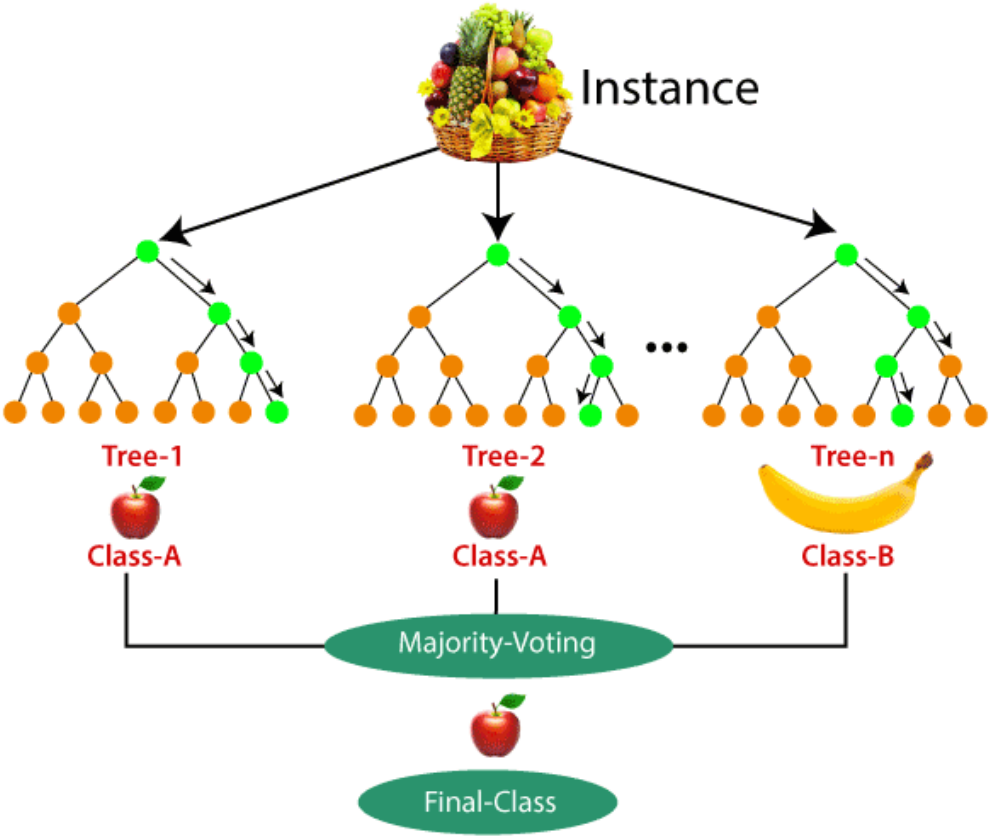
Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on ***Majority Voting or Averaging*** for Classification and regression respectively.



For example: consider the fruit basket as the data as shown in the figure below. Now n numbers of samples are taken from the fruit basket and an individual decision tree is constructed for each sample. Each decision tree will generate an output as shown in the figure. The final output is considered based on majority voting. In the below figure you can see that the majority decision tree gives output as an apple when compared to a banana, so the final output is taken as an apple.



Important Features of Random Forest

1. Diversity- Not all attributes/variables/features are considered while making an individual tree, each tree is different.

2. Immune to the curse of dimensionality- Since each tree does not consider all the features, the feature space is reduced.

3. Parallelization-Each tree is created independently out of different data and attributes. This means that we can make full use of the CPU to build random forests.

4. Train-Test split- In a random forest we don't have to segregate the data for train and test as there will always be 30% of the data which is not seen by the decision tree.

5. Stability- Stability arises because the result is based on majority voting/ averaging.

Difference Between Decision Tree & Random Forest

Random forest is a collection of decision trees; still, there are a lot of differences in their behavior.

Decision trees	Random Forest
1. Decision trees normally suffer from the problem of overfitting if it's allowed to grow without any control.	1. Random forests are created from subsets of data and the final output is based on average or majority ranking and hence the problem of overfitting is taken care of.
2. A single decision tree is faster in computation.	2. It is comparatively slower.
3. When a data set with features is taken as input by a decision tree it will formulate some set of rules to do prediction.	3. Random forest randomly selects observations, builds a decision tree and the average result is taken. It doesn't use any set of formulas.

Thus random forests are much more successful than decision trees only if the trees are diverse and acceptable.