Ensemble Learning:-

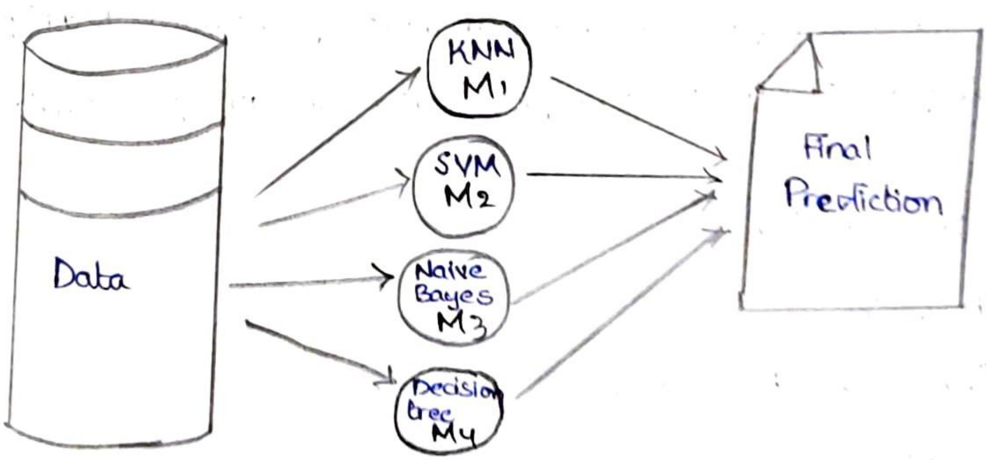
Ensemble Learning is a supervised learning technique used in machine learning to improve overall performance by combining the predictions from multiple models.

* A group of predictions is also an "Ensemble"; thus this technique is called "Ensemble Learning", and an ensemble alearning algorithm is called an "ensemble method."

Definition :-

* An ensemble method is technique that contains or combines the predictions from multiple machine learning algorithms toge-ther make more accurate predictions than individual model. A model comprised of many models is called an "Ensemble model."

⇒ Simply it combines the several base models in order to produce one optimal predictive model.



Base learners.

⇒ Ensemble learning Combines multiple machine learning models into one "predictive model".

⇒ Ensemble methods are classified into two groups.

1. Seqeuntial Ensemble methods.
2. Parallel Ensemble methods.

Ensemble Methods

Seqeuntial Ensemble method

Ex:- Basting

* Base learners are generated consecutively (one after other)
* Basic motivation is to use the dependence b/w the base learners.
* The overall performance of a model can be boosted.

Parellel Ensemble method

Ex:- Bagging

* Applied whatever the base learners are generated in parellel.
* Basic motivation is to use independence b/w the base learners.

Types of ensembles:-

1. Homogereous ensembles :- Ensemble methods contain the same type of learning algorithms | learning model.

2. Heterogeneous ensembles :- Methods that contain different types of learning algorithms | learning model.

* The most popular ensemble methods of ensemble techniques:-

1. Simple Ensemble Techniques - Use mean | average ensemble techniques:-

a) Max voting

b) Averaging.

c) Weighted Averaging.

2. Advanced ensemble techniques:-

1. Bagging (Bootstrap Aggregation)
2. Random Forest
3. Boosting
4. Stacking Generalization (Blending)
5. Voting classifiers.

1. Bagging (Bootstrap Aggregation):-

a. bagging:- it gets its name because it contains or combines bootstrapping and aggregation to turn one ensemble model.

Use the same training algorithms/ methods for every predictor but train them on different random subsets of training set. When sampling is preferred with replacement this method is called bagging.

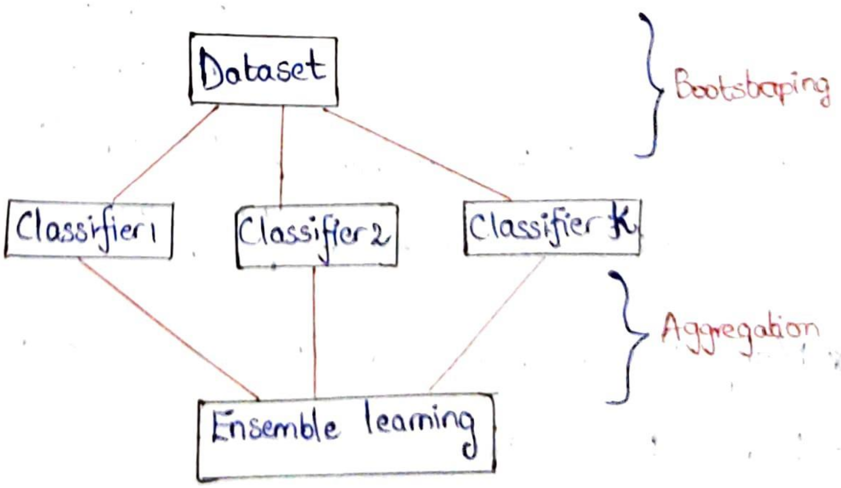
b. posting:- When sampling is performed without replacement, is called posting.

⇒ Bagging avoids overfilling of data & is used of both classification & Regression.

⇒ It is another approach to use the same training algorithm for each prediction, but to train them on different random sets of the training data.

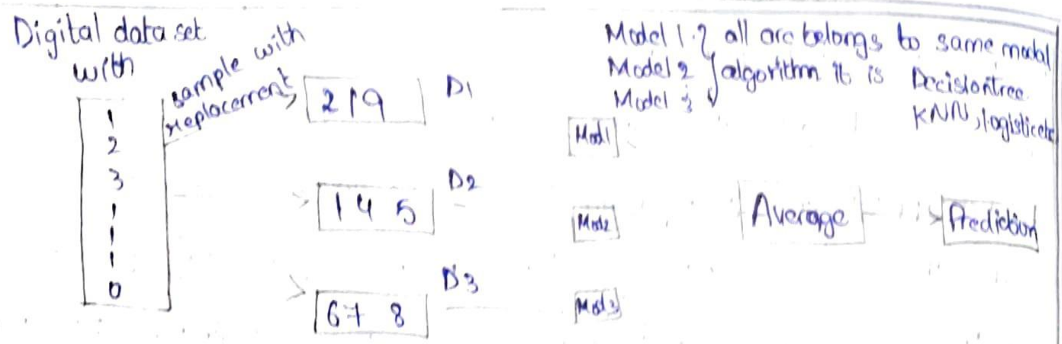
⇒ Here, we will use only, decision tree and it is under random forest Algorithm. (It will generate no. of decision treess to a model).

Aggregation:- The subsets of bootstraping data is given as training data & predict the aggregate output to the new data.



Bootstraping:-

It means randomly creating samples of data out of a dataset with replacement.



Definition:- It can be defined as selecting the random data from dataset & place it in a subset is called as "Bootstraping."

* Bootstraping is a sampling technique in which we create "sub-sets" of observations from the original dataset, "with replacement". The size of the original set not to be change.

* The primary goal of "bagging" or "bootstrap aggregating" ensemble method is to "minimize variance errors" in database decision treess

Sample Code:-

```
from sklearn.ensemble import Bagging Classifier
from sklearn.ensemble import Decision Tree classifier.
bag-classifier = Bagging Classifier (Decision Tree Classifier,
                                     n-estimator = 10,
                                     max-samples = 5,
                                     bootstrap = True)
bag-clf.fit (x-train, y-train)
y-pred = bag-clf.predict (x-test)
```

to fit the data

if bootstrap is false then it comes under pasting

* Many Popular ensemble algorithms are based on Bagging as follows:

⇒ Bagged Decision Tree

⇒ Random Forest

⇒ Extra Trees

⇒ Bagging Meta Estimator

Voting classifiers:- (Averaging)

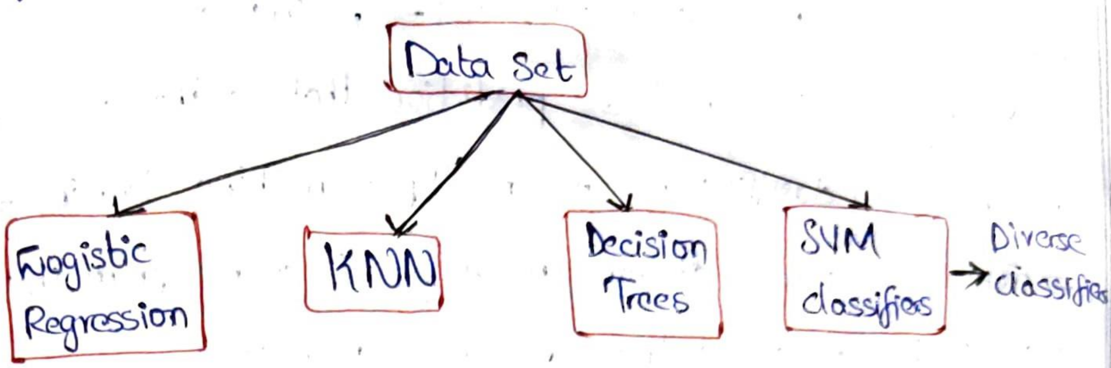


Fig:- Training on Diverse classifiers.

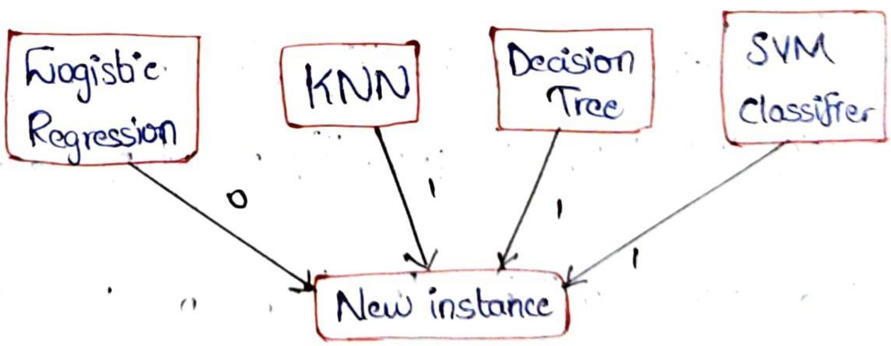
Diverse Classifiers:-

⇒ Giving training on various classifiers models on the same data-set is called as Diverse Classifiers.

⇒ After giving training to the dataset the model will predict the New data (or) unseen data.

⇒ Based on aggregate output we can predict for the new instance.

⇒ Aggregate output based on counting of all the models the majority of count will be as output for new data.



⇒ Suppose you have logistic regression, KNN, Decision Tree, & SVM classifier to be trained on the dataset.

⇒ A very simple way to create an better classifier is to aggregate the predictions of each classifier & predict the class that gets "Most votes."

⇒ The majority of votes is called as "Voting Classifier."

⇒ In classification, the target class contains only categorical data like +ve/-ve, +/-, 0/1 etc.

⇒ Voting classifier involves prediction that is the average(regression) on the sum(classification) of multiple machine learning models.

⇒ From the above diagram majority of count from 0, 1, 1, 1 → '1' will be option output for new data set.

⇒ There are 2 types of voting classifier.

a) Hard Voting Classifier.

b) Soft Voting Classifier.

a) Hard Voting Classifier:-

The predicted output class is a class with the highest majority of the votes. Suppose 4 classifiers predicted output as (0,1,1,1), So here the majority predicted 1 as output, Hence will be final output.

b) Soft Voting Classifier:-

Here calculating the probability for each & every instance and calculating the average of probability of getting Yes/No, +/-, 0/1, etc.

⇒ The Highest average will be the output of New data.

$$\boxed{\text{Avg} = \frac{\text{Sum of all events}}{\text{Total no of models.}}}$$

Ex:-

For class 1	For class 0
Model 1 = 0.9	Model 1 = 0.1
Model 2 = 0.8	Model 2 = 0.2
Model 3 = 0.7	Model 3 = 0.3
Model 4 = 0.6	Model 4 = 0.4

$$Avg = \frac{0.9+0.8+0.7+0.6}{4}$$
$$= 3/4$$
$$= 0.75$$

$$Avg = \frac{0.1+0.2+0.3+0.4}{4}$$
$$= 1/4$$
$$= 0.25.$$

⇒ Highest average is 0.75, So it belongs to class 1.

Sample Coding for Voting Classifier:-

```
from sklearn.ensemble import voting classifier
from sklearn.linear.model import Logistic Regression.
from sklearn.SVM import SVC
from sklearn.tree. import Decision Tree classifier.
estimator = []
estimator.append ('DTC', Decision Tree Classifier())
estimator.append ('SVC', SVC(gammer='auto', probability = True)
#hard voting
  vot_hard = voting classifier (estimators = estimators, voting = 'hard')
  vot_hard.fit(x_train, y_train)      } fitting data into model
  y-pred = vot_hard.predict(x_test)
# soft voting
  vot_soft = voting classifier (estimators = estimator, voting = 'hard')
  vot_soft.fit (x_train, y_train)
  y-pred = vot_soft.predict (x_test)
```

Stacking:- (Stacked Generalization.)

→ Introduced by 'Wolpert'.

⇒ It is different from the bagging & boosting combination methods, the stack often combines heterogenous weak learners/ base learners.

⇒ Additionally, stacking learns to combine the base models using a meta model, where as bagging & boosting are some averaging process.

⇒ So in order to build stacking model, we need to define two things.

1. The Base algorithms for weak learners.

2. The meta-method that will combine weak learners.

⇒ Stacking is also known as a stacked generalization & it is an extended form of the model Averaging ensemble technique in which all sub models equally participate as per the performance weights and build a new model with better predictions.

⇒ Bagging & Boosting methods for handling bias & variance.

⇒ Stacking helps to improve model prediction accuracy.

⇒ Stacking is a ensemble technique to combine with multiple classifications models via "Meta Classifier".

⇒ Each model is different from another (Heterogeneous) and trained on training data.

⇒ Meta classifier will predict train on the outputs (or) probability of multiple classifiers which are involved.

⇒ Meta classifier will predict the final output.

How it works?

1) Lets say $D = \{x_1, x_2\}$ and y as output variable i.e, $y = \{0, 1\}$.

2) M_1, M_2, M_3, M_4 are different classification models.

3) Train the models with data set 'D'.

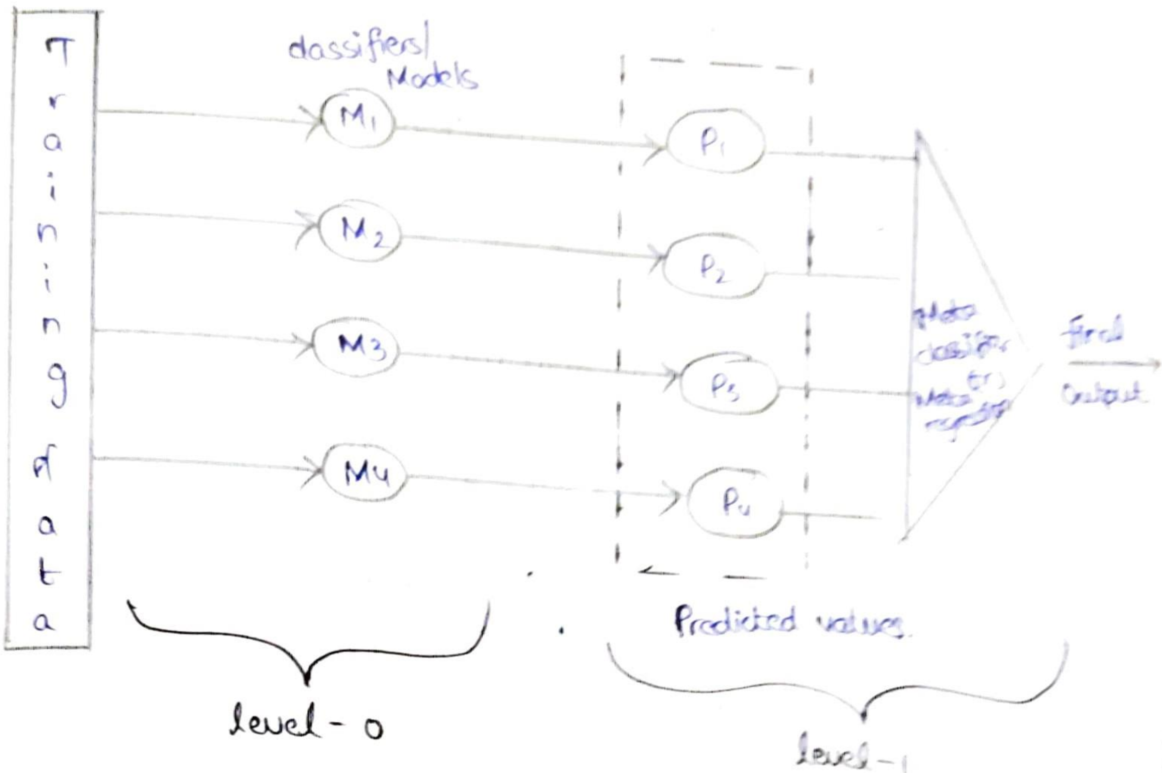
4) Assume 'x_2' as a New data instance we need to predict the class label to it.

5) We will pass 'x_2' to each model for prediction class

6) M_1 predicts = 0 M_2 predicts = 1 M_3 Predicts = 1 M_4 predicts = 1.

7) These outputs are sent to final meta classifier (or) Blender (or) Blending

8) The metaclassifier chooses the final output.



Level-0 models (Base-models) :- Models fit on the training data and "whose preditions are computes."

Level-1-model:-(Meta-model) :- Model that learns how to best combine the preditions of the base models."

⇒ Base level models/algorithms are trained on bases of a complete training dataset using k-fold cross validation.

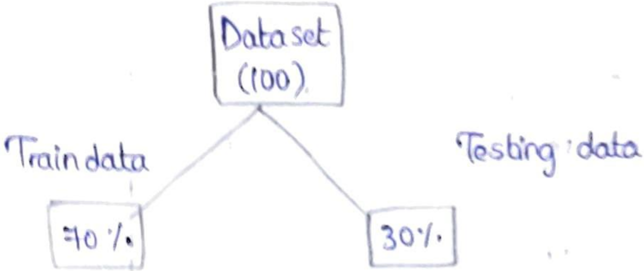
⇒ Stacking is useful when the results of the individual algorithms are very different.

Blending:-

It follows the same approach as stacking but uses only a holdout (validation) set from the train set to make predictions. The hold out set and the predictions are used to build a model which is run on the test set.

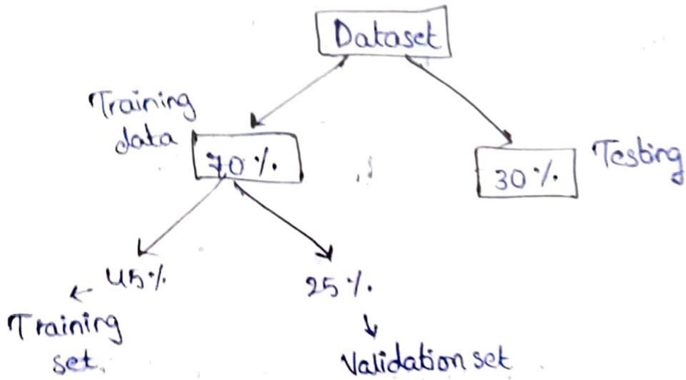
Step-1:-

First data is divided into train & train set.



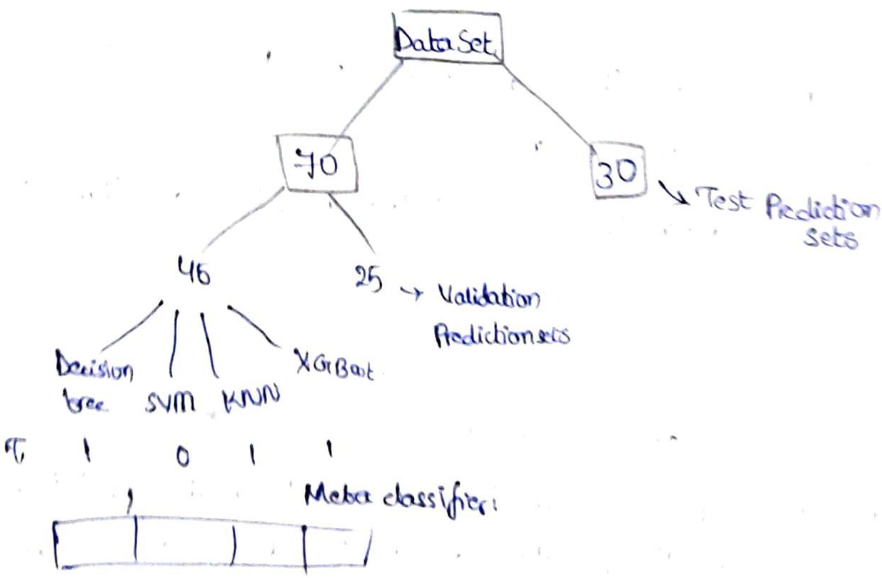
Step 2:-

The train set is split into into training & validation sets.



Step-3:-

Base models are. lifted fited on the training set & predictors are made validation sets & test set.



Step-4:-

The validation set & its predictors are used as features to build a new model and this model is used to make final predictions on the test features.

Blending Ensembles:- Use of a linear models:

Linear regression (Regression Problem).

Logistic regression (Classification Problem).

Blending:- Stacking-type ensemble where the meta_model is training on predictions made on "a hold out (validation set) dataset."

Stacking:- Stacking-type ensemble where the meta model is trained on "out of fold" predictions made "during K-fold cross validation."

Implementation:-

```
from sklearn.linear model import Logistic Regression
from sklearn. SVM import Linear SVC
from sklearn. Ensemble import Random Forest Classifier.
from sklearn. Ensemble import stacking classifier
estimators = [
    ('rf', RandomForest classifier (n_estimators = 10, random_state=42)),
    ('svr', linear SVC (random_state=42))]
classifier = Stacking classifier (estimators = estimators, final-estimator
                                  = Logistic Regression()')
classifier. fit ( x-train, y_train).
```

What is cross validation?

Usually we take a data set, split it into train & test sets. But sometimes there is a chance to train data sample may be in testing samples. In real time production we need to solve this one, by using cross validation techniques.

Types of cross validations:-

1. K- fold validation.
2. Startified K fold
3. Time Series split
4. Repeated K-fold
5. Leave One Out Cross Validation (LOOCV).
6. Shuffle & Split.

K- fold Validation:-

⇒ In K-fold cross validation, the training set is randomly split into K (usually 5 to 10) subsets known as folds, where K-1 folds are used to train the model and the other fold is used to test the model.

⇒ The steps required to perform K-fold cross validation are given below.

Step-1:- Split the entire data randomly in k-folds.

Step-2:- Then fit the model with K-1 folds & test it with the remaining K^{th} fold. Record the performance metric.

Step-3:- Repeat step 2 untill every K-fold serves as testset.

Step-4:- Take the average of all the recorded scores. This will serve as the final performance metric of your model.

Implementation:-

from sklearn.model_selection import cross_val_score

accuracies = cross_val_score (estimator = classifier, x = x_train, y = y_train, cv = 10).

$\downarrow$ no of folds.

accuracies.main () (average of all accuracies).

Leave on out Cross Validation (LOOCV):-

In this model, usually we divide the data into train & test sets, But in this, instead of dividing the data into 2 subsets, we select a single observation as test data, and everything else is labeled as training data and the model is trained. Now the 2nd observation is selected as test data and the model is trained on the remaining data.

1 2 3 - - - - - - - - - - - - n

$\Downarrow$

$\boxed{1}$ 2 3 - - - - - - - - - - - - n

test data with remaining all data points are labelled data.

1 $\boxed{2}$ 3 - - - - - - - - - - - - n

1 2 $\boxed{3}$ - - - - - - - - - - - - n

1 2 3 - - - - - - - - - - - - $\boxed{n}$

⇒ This process continues 'n' times & the average of all these items is calculated and estimated as test error.

⇒ from sklearn.model_selection import leave one out.

Random Forest:-

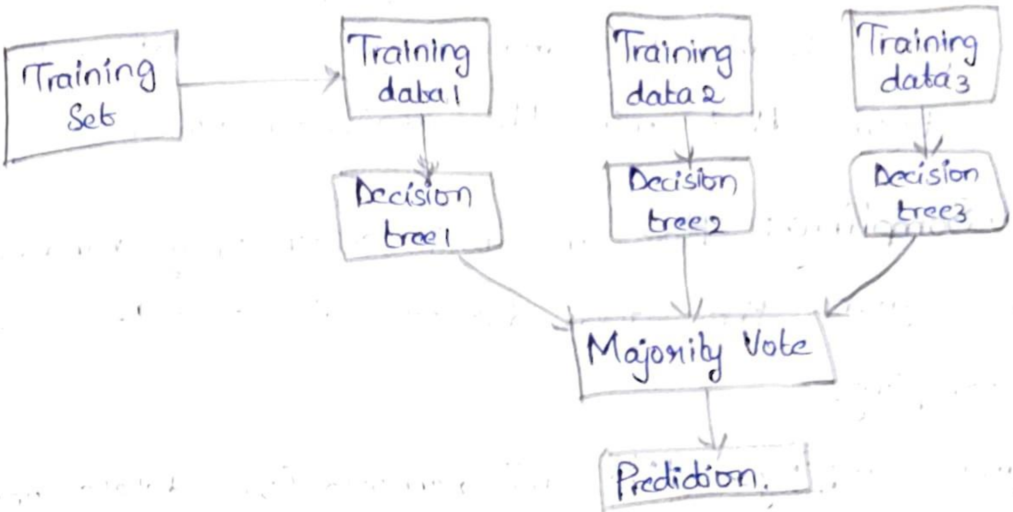
Random Forest is a good example of a ensemble learning technique which is followed Bagging.

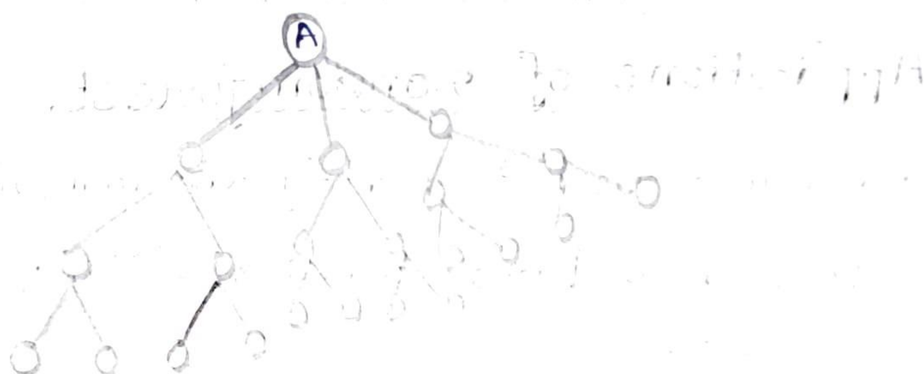
This algorithm builds multiple decision trees and merge them together to get a more accurate and stable predictions. Then it gets predictions from each tree and by means of majority voting is select the decision which gets. Majority vote.

Random Forest creates random subsets of the features.

The tree in random forests are run parallel.

There is no interaction between these trees while building the trees.





Tree1 ↓ Class 'A'

Tree 2 ↓ class 'A'

Tree 3 ↓ class 'B'

Prediction → On majority vote, O/P is class 'A'.

Steps for Building Random Forests:-

1. Picks at random 'K' data points (Row sampling + Column Sampling) from the training data set.
2. Building the decision tree with these 'K' data points.
3. Choose the number 'N' trees you want to build and repeat steps 1 & 2.

4. For a new data point, pass the data point to every decision tree to predict the category to which the data point belongs to and assign new data point to the category that wins the "majority vote."

Row Sampling:- Here we will select only random rows from the given datasets it is called as rowsampling.

Column Sampling:- We will remove (or) delete unwanted columns from the given data set.

Applications of randomforest:-

1. **Banking:-** used to identify the loan risk.
2. **Medicine:-** Disease trends & risks of disease can be idetifies.
3. **Marketing:-** Here marketing trends can be identified

Implementation:-

```
from sklearn.ensemble import RandomForest classifier
classifier = Random Forest Classifier (n_estimators = 10,
                         criteraion = 'entropy')
rf classifier . fit (x-train, y train)
y.pred = rf classifier.Predict (x text)
print ( y.pred).
```

How many decisiontrees we want

by default 10.

Boosting:-

→ Boosting is an ensemble modelling technique that attempts to build a strong classifier from the no. of work classifiers.

⇒ Boosting is a "sequential" process where each subsequent model attempts to correct the errors of previous model until and unless "error value is minimized."

→ The principal idea behind boosting technique is we first build a model on the training dataset and build a 2nd model to rectify the errors in 1st model.

⇒ This procedure continuous untill and unless "errors are minimized".

⇒ Here in Boosting,

* Model is developed - Incrementlly
* Execution of model is done - Sequentially.

⇒ Here in boosting technique we cannot assume that all the models will give correct output.

→ To calculate the performance (or) accuracy (or) error rate we can construct the decision true classifier.