Comparison between decision tree and random forest towards recommendation system or engine

Submitted as a partial fulfillment of Bachelor of Technology in Computer Science & Engineering
of
Maulana Abul Kalam Azad University of Technology
(Formerly known as West Bengal University of Technology)

Project Report

Submitted by

Name of Students	University Roll No.
Pritam Roy	**11600118037**
Rupak Pal	**11600118032**
Pritam Das	**11600118038**
Srijon Mallick	**11600118017**
Souvik Saha	**11600118020**

Under the supervision of

Dr. S. S. Thakur
Associate Professor, Department of Computer Science and Engineering

Department of Computer Science & Engineering,
MCKV Institute of Engineering
243, G.T. Road(N)
Liluah, Howrah - 711204

Department of Computer Science & Engineering
MCKV Institute of Engineering
243, G. T. Road (N),
Liluah, Howrah-711204

CERTIFICATE OF RECOMMENDATION

I hereby recommend that the thesis prepared under my supervision by Pritam Roy, Rupak Pal, Pritam Das, Srijon Mallick, Souvik Saha entitled Comparison between decision tree and random forest towards recommendation system or engine be accepted in partial fulfillment of the requirements for the degree of Bachelor of Technology in Computer Science & Engineering Department.

Mr. Avijit Bose
Assistant Professor & Head of the Department,
Computer Science & Engineering Department
MCKV Institute of Engineering, Howrah

Project guide
Dr. S. S. Thakur,
Associate Professor,
Computer Science &
Engineering Department

MCKV Institute of Engineering
243, G. T. Road (N), Liluah
Howrah-711204

Affiliated to
Maulana Abul Kalam Azad University of Technology
(Formerly known as West Bengal University of Technology)

CERTIFICATE

This is to certify that the project entitled Comparison between decision tree and random forest towards recommendation system or engine and submitted by

Name of students	University Roll No.
Pritam Roy	11600118037
Rupak Pal	11600118032
Pritam Das	11600118038
Srijon Mallick	11600118017
Souvik Saha	11600118020

has been carried out under the guidance of myself following the rules and regulations of the degree of Bachelor of Technology in Computer Science & Engineering of **Maulana Abul Kalam Azad University of Technology (**Formerly West Bengal University of Technology).

_____

(Signature of the project guide)
Dr. S. S. Thakur,
Associate Professor,
Computer Science & Engineering Department

1. Pritam Roy
2. Rupak Pal
3. Pritam Das,
4. Srijon Mallick
5. Souvik Saha

MCKV Institute of Engineering
243, G. T. Road (N), Liluah
Howrah-711204
Affiliated to
Maulana Abul Kalam Azad University of Technology
(Formerly known as West Bengal University of Technology)

CERTIFICATE OF APPROVAL

(B.Tech Degree in Computer Science & Engineering)

This project report is hereby approved as a creditable study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is to be understood that by this approval, the undersigned do not necessarily endorse or approve any statement made, opinion expressed and conclusion drawn therein but approve the project report only for the purpose for which it has been submitted

COMMITTEE ON FINAL	1.
EXAMINATION FOR	2.
EVALUATION OF	3.
PROJECT REPORT	4.
	5.

ACKNOWLEDGEMENT

We express our sincere gratitude to Dr. S. S. Thakur, Associate Professor, Department of Computer Science and Engineering, our project guide and Mr. Avijit Bose, Assistant Professor and Head of Department (CSE) for providing us their guidance and cooperation for the project. We also extend our sincere thanks to all other faculty members of Computer Science & Engineering Department and our friends for their support and encouragement. We will be failing in duty if we do not acknowledge with grateful thanks to the authors of the references and other literatures referred to in this project. Last but never the least we are very much thankful to our parents who guided and supported us in every step which we took.

CONTENTS

1. ABSTRACT

On the Internet, where the number of choices is overwhelming, there is need to filter, prioritize and efficiently deliver relevant information in order to alleviate the problem of information overload, which has created a potential problem to many Internet users. Recommendation system solve this problem by searching through large volume of dynamically generated information to provide users with personalized content and services. Different machine learning algorithms like decision tree classifier and random forest classifier plays a significant role in providing personalized content and services to all internet users. In this project we will explore the above to classifiers, comparing based on their performance and accuracy, which will help us towards suggesting a new recommendation system or engine.

2. INTRODUCTION

Machine Learning (ML) pertains to the ability of data-driven models to “learn” information about a system directly from observed data without predetermining mechanistic relationships that govern the system. ML algorithms are able to adaptively improve their performance with each new data sample and discover hidden patterns in complex heterogeneous and high dimensional data. ML has become the core technology for numerous real-world applications: from weather forecasting and DNA sequencing, to Internet search engines and image recognition. In different engineering domain ML offers predictive models, such as Decision Trees (DTs), Random Forests (RFs), Support Vector Machines (SVMs), ectara. which are able to map highly non-linear heterogeneous input and output patterns even when physiological relationships between model variables could not be determined due to complexity, pathologies, or lack of biological understanding. Nevertheless, ML models are rarely viewed in the context of small data, where insufficient number of training samples can compromise the learning success. DTs are easy to interpret by non-statistician and are intuitive to follow. They cope with missing values and are able to combine heterogeneous data types into a single model, whilst also performing an automatic principal feature selection. Combining multiple Decision Trees (DTs) in a Random Forests (RFs) maintains this interpretability, but offers state-of-the-art prediction accuracies.

Tomas Pranckevičius, and Virginijus Marcinkevicius [1] investigated Naïve Bayes, Random Forest, Decision Tree, Support Vector Machines, and Logistic Regression classifiers implemented in Apache Spark, i.e., the in-memory intensive computing platform. They focused on comparing these classifiers by evaluating the classification accuracy, based on the size of training data sets, and the number of n-grams. In experiments, short texts for product-review data from Amazon were analysed. A comparative analysis of decision tree algorithms and random forest was done by Shiju Sathyadevan and Remya R. Nair [2]. Their paper focuses on comparison of decision tree algorithms-ID3, C4.5 which was then compared with random forest. Random forest is found to be superior than other two based on accuracy. Apurva Datkhile et al. [3] defined a prototype using four different models- Naive Bayes, Random Forest, Logistic Regression and Decision Tree Algorithm, which can be used by organizations to make a correct or correct decision to approve or reject a consumer's request for a loan. Sebastian Buschjäger and Katharina Morik [4] investigated implementations of

decision trees and random forests for the classical von-Neumann computing architecture and custom circuits by the means of field programmable gate arrays.

The current study aims to address data applications of ML models for classification tasks. Specifically, the report considers Decision Trees (DTs) and Random Forests (RFs) for comparison between the two and propose a recommendation system.

This report is so structured that firstly we describe the different concepts of machine learning, decision trees and random forests followed by prerequisites of the project, then a brief explanation of code and output. Lastly concluding with conclusion at this current stage and next what can be expected from this project on completion.

3. Machine Learning:

3.1. What is Machine Learning?

In the real-life scenario, a human being can learn or can gain knowledge from his/her experiences with his/her learning ability. So, we can easily train a human being. We have computers or machines which work on our instructions. But like a human, can we train a machine? The answer is yes, we can train a machine. A machine can also learn from its past experience. So here comes the role of Machine Learning.

Machine learning is a branch of artificial intelligence, that enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed.

Two types of learning technique are there in machine learning:

1. Supervised Learning

2. Unsupervised Learning

3.2. Supervised Learning:

Before exploring Supervised Learning, we need to be familiar with the two terminologies:

1. **Train Dataset:** Datasets having some labelled training data. The labelled data means some input data is already tagged with the correct output. This dataset is used to train a machine.

2. **Test Dataset:** Datasets having some unlabelled test data on which the machine predicts the output.

In supervised learning, the training data provided to the machines work as the supervisor that teaches the machines to predict the output correctly on test datasets. It applies the same concept as a student learns in the supervision of the teacher.

In the real-world, supervised learning can be used for Risk Assessment, Image classification, Fraud Detection, spam filtering, etc.

Categories of Supervised Machine Learning

Supervised machine learning can be classified into two types of problems, which are given below:

- Classification
- Regression

a) Classification

Classification algorithms are used to solve the classification problems in which the output variable is categorical, such as "Yes" or No, Male or Female, Red or Blue, etc. The classification algorithms predict the categories present in the dataset. Some real-world examples of classification algorithms are Spam Detection, Email filtering, etc.

Some popular classification algorithms are given below:

- Random Forest Algorithm
- Decision Tree Algorithm
- Logistic Regression Algorithm
- Support Vector Machine Algorithm

b) Regression

Regression algorithms are used to solve regression problems in which there is a linear relationship between input and output variables. These are used to predict continuous output variables, such as market trends, weather prediction, etc.

Some popular Regression algorithms are given below:

- Simple Linear Regression Algorithm
- Multivariate Regression Algorithm
- Decision Tree Algorithm
- Lasso Regression

3.3. Unsupervised Learning:

In the previous topic, we learned supervised machine learning in which models are trained using labelled data under the supervision of training data. But there may be many cases in

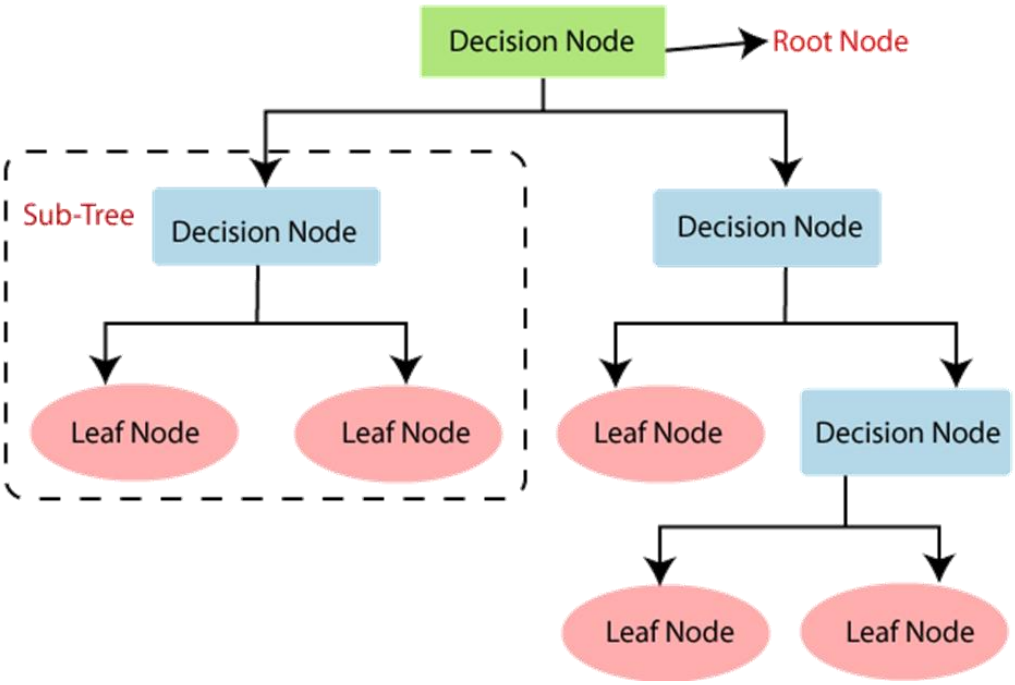
which we do not have labelled data and need to find the hidden patterns from the given dataset. So, to solve such types of cases in machine learning, we need unsupervised learning techniques.

Unsupervised learning is a type of machine learning in which models are trained using unlabelled dataset and are allowed to act on that data without any supervision.

4. Decision Tree Classifier:

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome. Decision tree consists of two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches. The decisions or the test are performed on the basis of features of the given dataset.

It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions. A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees. Below diagram explains the general structure of a decision tree:



4.1. Decision Tree Terminologies

- **Root Node:** Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- **Leaf Node**: Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- **Splitting:** Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- **Branch/Sub Tree:** A tree formed by splitting the tree.
- **Pruning:** Pruning is the process of removing the unwanted branches from the tree.
- **Parent/Child node:** The root node of the tree is called the parent node, and other nodes are called the child nodes.

4.2. Why use Decision tree?

There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision tree:

- Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
- The logic behind the decision tree can be easily understood because it shows a tree-like structure.

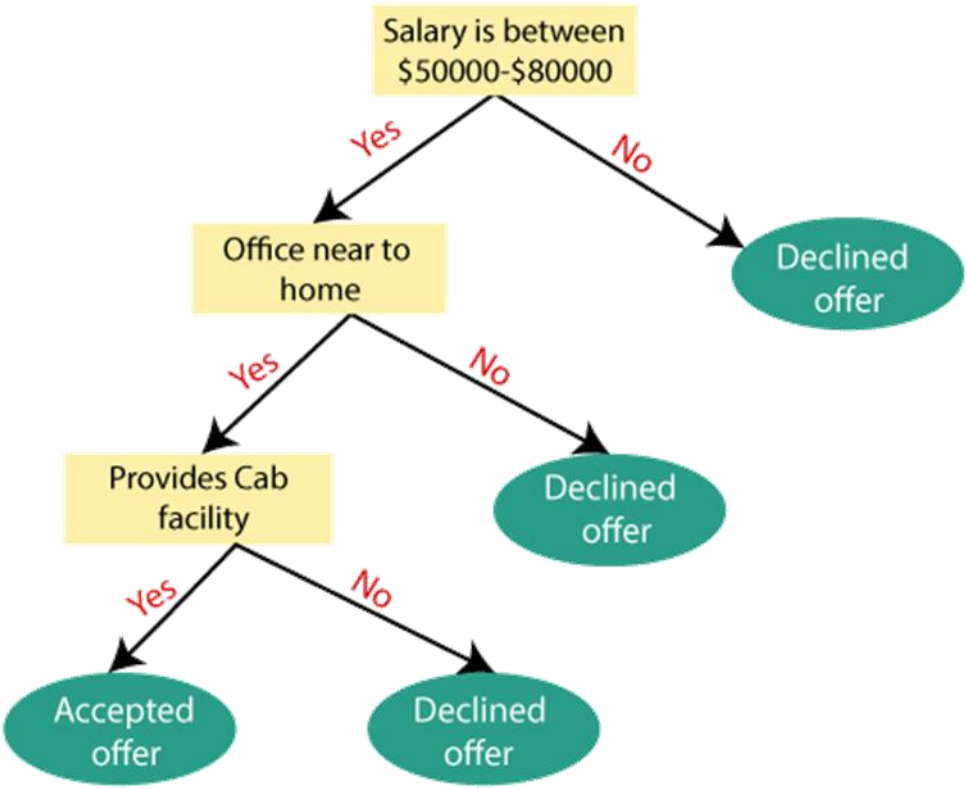
4.3. How does the Decision Tree algorithm Work?

In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

- **Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.
- **Step-2:** Find the best attribute in the dataset using Attribute Selection Measure (ASM).
- **Step-3:** Divide the S into subsets that contains possible values for the best attributes.
- **Step-4:** Generate the decision tree node, which contains the best attribute.
- **Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Example: Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or not. So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM). The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels. The next decision node further gets split into one decision node (Cab facility) and one leaf node. Finally, the decision node splits into two leaf nodes (Accepted offers and Declined offer). Consider the below diagram:



4.4. Attribute Selection Measures:

While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as Attribute selection measure or ASM. By this measurement, we can easily select the best attribute for the nodes of the tree. There are two popular techniques for ASM, which are:

- Information Gain
- Gini Index

1. Information Gain:

- Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
- It calculates how much information a feature provides us about a class.
- According to the value of information gain, we split the node and build the decision tree.
- A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

  Information Gain= Entropy(S) - [(Weighted Avg) *Entropy (each feature)

Entropy: Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

Entropy(s)= -P(yes)log2 P(yes)- P(no) log2 P(no)

Where,

- S= Total number of samples
- P(yes)= probability of yes
- P(no)= probability of no

2. Gini Index:

- Gini index is a measure of impurity or purity used while creating a decision tree in the CART (Classification and Regression Tree) algorithm.
- An attribute with the low Gini index should be preferred as compared to the high Gini index.
- It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.
- Gini index can be calculated using the below formula:

  Gini Index= $1- \sum_j P_j^2$

4.5. Advantages of the Decision Tree

- It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
- It can be very useful for solving decision-related problems.
- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.

4.6. Disadvantages of the Decision Tree

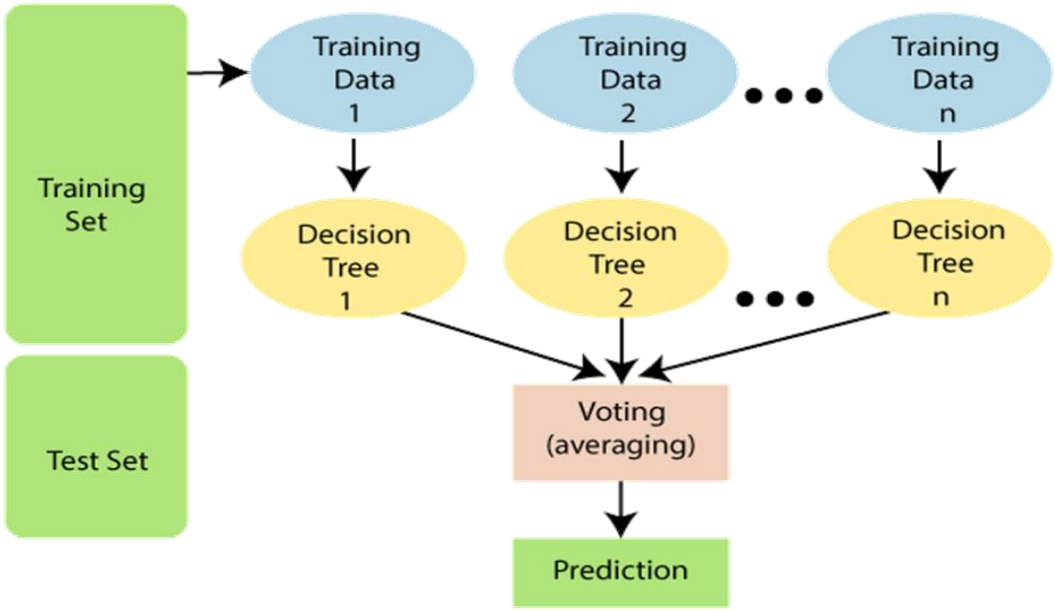
- The decision tree contains lots of layers, which makes it complex.
- It may have an overfitting issue, which can be resolved using the Random Forest algorithm.
- For more class labels, the computational complexity of the decision tree may increase.

5. Random Forest:

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting. The below diagram explains the working of the Random Forest algorithm:



5.1. Assumptions for Random Forest:

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random Forest classifier:

- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

5.2. Why use Random Forest?

Below are some points that explain why we should use the Random Forest algorithm:

- It takes less training time as compared to other algorithms.

- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.

5.3. How does Random Forest algorithm work?

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

Step-1: Select random K data points from the training set.

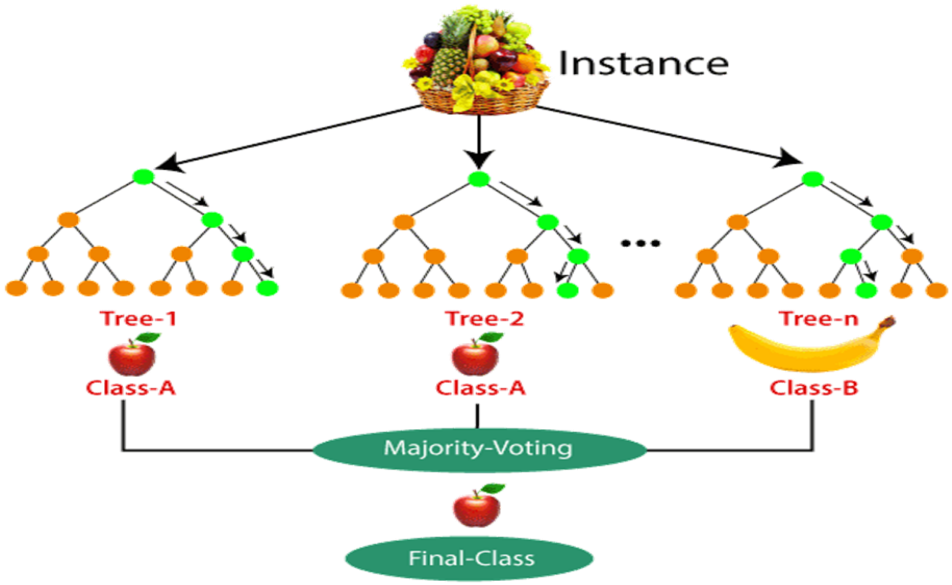
Step-2: Build the decision trees associated with the selected data points (Subsets).

Step-3: Choose the number N for decision trees that you want to build.

Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes. The working of the algorithm can be better understood by the below example:

Example: Suppose there is a dataset that contains multiple fruit images. So, this dataset is given to the Random Forest classifier. The dataset is divided into subsets and given to each decision tree. During the training phase, each decision tree produces a prediction result, and when a new data point occurs, then based on the majority of results, the Random Forest classifier predicts the final decision. Consider the below image:



5.4. Advantages of Random Forest

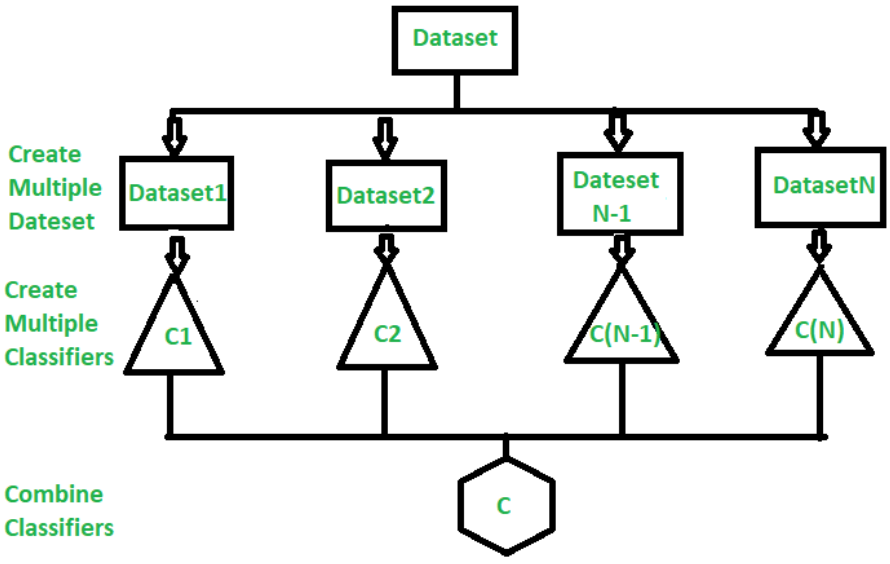
- Random Forest is capable of performing both Classification and Regression tasks.
- It is capable of handling large datasets with high dimensionality.
- It enhances the accuracy of the model and prevents the overfitting issue.

5.5. Disadvantages of Random Forest

- Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

6. Ensemble Learning:

Ensemble Learning is also a part of machine learning. It helps in improving machine learning results by combining several models. This approach allows the production of better predictive performance compared to a single model.



6.1. Benefits of Ensemble Learning:

Ensembles overcome three problems –

- **Statistical Problem –**
  The Statistical Problem arises when the hypothesis space is too large for the amount

of available data. Hence, there are many hypotheses with the same accuracy on the data and the learning algorithm chooses only one of them! There is a risk that the accuracy of the chosen hypothesis is low on unseen data!

- **Computational Problem –**
  The Computational Problem arises when the learning algorithm cannot guarantees finding the best hypothesis.

- **Representational Problem –**
  The Representational Problem arises when the hypothesis space does not contain any good approximation of the target class(es).

6.2. Methods for developing Ensemble:

The main challenge is not to obtain highly accurate base models, but rather to obtain base models which make different kinds of errors. For example, if ensembles are used for classification, high accuracies can be accomplished if different base models misclassify different training examples, even if the base classifier accuracy is low.

Methods for Independently Constructing Ensembles –

- Majority Vote
- Bagging and Random Forest
- Randomness Injection
- Feature-Selection Ensembles
- Error-Correcting Output Coding

Methods for Coordinated Construction of Ensembles –

- Boosting
- Stacking

6.3. Types of Ensemble Classifier:

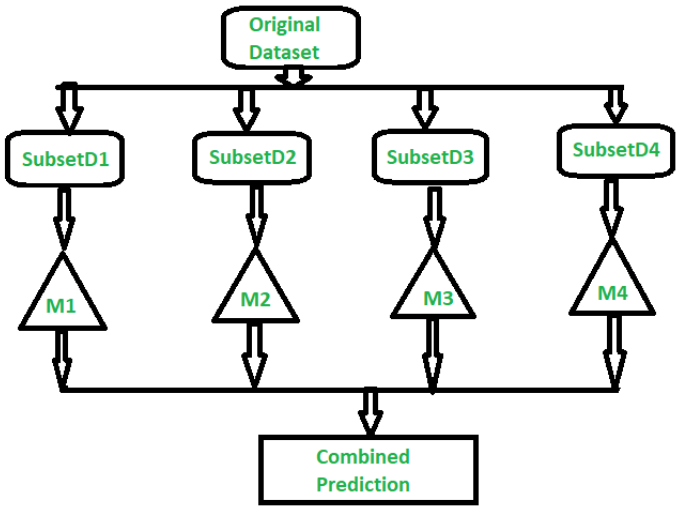
6.3.1. Bagging:

Bagging is used when our objective is to reduce the variance of a decision tree. Here the concept is to create a few subsets of data from the training sample, which is chosen randomly with replacement. Now each collection of subset data is used to prepare their decision trees thus, we end up with an ensemble of various models. The average of all the assumptions from numerous trees is used, which is more powerful than a single decision tree.

Suppose a set D of d tuples, at each iteration *i*, a training set D_i of d tuples is sampled with replacement from D (i.e., bootstrap). Then a classifier model M_i is learned for each training set D < i. Each classifier M_i returns its class prediction. The bagged classifier M* counts the votes and assigns the class with the most votes to X (unknown sample).

Implementation steps of Bagging

1. Multiple subsets are created from the original data set with equal tuples, selecting observations with replacement.
2. A base model is created on each of these subsets.
3. Each model is learned in parallel from each training set and independent of each other.
4. The final predictions are determined by combining the predictions from all the models.



Random Forest is an expansion over bagging. It takes one additional step to predict a random subset of data. It also makes the random selection of features rather than using all features to develop trees. When we have numerous random trees, it is called the Random Forest.

These are the following steps which are taken to implement a Random Forest:

- Let us consider X observations Y features in the training data set. First, a model from the training data set is taken randomly with substitution.
- The tree is developed to the largest.
- The given steps are repeated, and prediction is given, which is based on the collection of predictions from n number of trees.

6.3.2. Boosting:

Boosting is another ensemble procedure to make a collection of predictors. In other words, we fit consecutive trees, usually random samples, and at each step, the objective is to solve net error from the prior trees.

If a given input is misclassified by theory, then its weight is increased so that the upcoming hypothesis is more likely to classify it correctly by consolidating the entire set at last converts weak learners into better performing models.

Gradient Boosting is an expansion of the boosting procedure.

Gradient Boosting = Gradient Descent + Boosting

It utilizes a gradient descent algorithm that can optimize any differentiable loss function. An ensemble of trees is constructed individually, and individual trees are summed successively. The next tree tries to restore the loss (It is the difference between actual and predicted values).

7. Methodology:

7.1. Dataset Used:

To create a machine learning model, the first thing we required is a dataset as a machine learning model completely works on data. The collected data for a particular problem in a

proper format is known as the dataset. Dataset may be of different formats for different purposes, such as, if we want to create a machine learning model for business purpose, then dataset will be different with the dataset required for a liver patient. Such as in our case a dataset from Dream Housing Finance company which lends home loans or approves customers for home loans based on criteria as follows

- Unique Loan ID
- Male/ Female
- Applicant married (Y/N)
- Number of dependents
- Applicant Education (Graduate/ Under Graduate)
- Self-employed (Y/N)
- Applicant income
- Co-applicant income
- Loan amount in thousands
- Term of loan in months
- credit history meets guidelines
- Urban/ Semi Urban/ Rural
- Loan approved (Y/N)

7.2. Data Pre-Processing:

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data pre-processing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

For achieving optimal and moreover better results from the applied model the format of the data has to be in a proper manner. Keeping this in mind all the null values present in the raw dataset is managed. Also, whenever a row with insufficient data is found where more than one attributes are missing, we eliminate those rows. Moreover, in case of duplicate rows we remove all the repeated entries keeping only one.

7.3. Technology Used:

This machine learning based project is implemented using python because of its simple syntax, modular architecture, rich text processing tools and the ability to work on multiple

operating systems. Python language is one of the most flexible languages and can be used for various purposes. The language is great to use when working with machine learning algorithms as it contains special libraries for machine learning.

Python packages:

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Numpy

NumPy is the fundamental package for scientific computing with Python. It contains among other things a powerful N-dimensional array-object, sophisticated(broadcasting) functions, tools for integrating C/C++ and Fortran code useful linear algebra, Fourier transform, and random number capabilities. Besides its obvious scientific uses, NumPy can also be used as an efficient multidimensional container of generic data. Arbitrary datatypes can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shell, the jupyter notebook, web application servers, and four graphical user interface toolkits.

Pandas

Pandas is an open source, BSD-licensed library providing high performance, easy- to-use data structures and data analysis tools for the Python programming language. Pandas library is well suited for data manipulation and analysis using python. In particular, it offers data structures and operations for manipulating numerical tables and time series.

Scikit-learn

Scikit-learn provides machine learning libraries for python some of the features of Scikit-learn includes:

- Simple and efficient tools for data mining and data analysis.

- ❖ Accessible to everybody, and reusable in various contexts.
- ❖ Built on NumPy, SciPy, and matplotlib.
- ❖ Open source, commercially usable –BSD license.

7.4. Software & Hardware requirements:

The experiments were conducted on Jupyter Notebook version 6.4.5 present within Anaconda-3 version 4.10.3 with Python 3.8.12 running on an Intel Core TM i5 PC with 2.42 GHz CPU with 8GB RAM.

8. Results & Discussions:

Decision Tree:

Initially we have processed the dataset and put it up for the test with decision tree classifier. Firstly, we import all the required packages and read in the CSV file as shown below.

```
In [1]: # Load libraries
        import pandas as pd
        from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier
        from sklearn.model_selection import train_test_split # Import train_test_split function
        from sklearn import metrics #Import scikit-learn metrics module for accuracy calculation
        from sklearn.datasets import load_iris
```

```
In [3]: # Reading the data By converting Normal String DataSet to Raw String DataSet
        data = pd.read_csv(r"D:\Final-Project\Data\dataset2.csv")

        # Printing First 5 Rows

        data.head()
```

Out[3]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
0	LP001002	1	0	0	1	0	5849	0.0	0	360	1
1	LP001003	1	1	1	1	0	4583	1508.0	128	360	1
2	LP001005	1	1	0	1	1	3000	0.0	66	360	1
3	LP001006	1	1	0	0	0	2583	2358.0	120	360	1
4	LP001008	1	0	0	1	0	6000	0.0	141	360	1

Now we will test the above data, in order to do that we split the data in 80/20 ratio, where 80% of the data will be fed into the classifier to train it and 20% is kept aside to test its accuracy. The following snippets of code shows the same.

```
In [6]: X=data.iloc[:,1:12]
        Y=data.iloc[:,12]
        Y[0:4]

Out[6]: 0    Y
        1    N
        2    Y
        3    Y
        Name: Loan_Status, dtype: object
```

```
In [7]: X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.25, random_state=100)
        clf = DecisionTreeClassifier()
        clf.fit(X_train,Y_train)
        Y_pred = clf.predict(X_test)
        print("Accuracy:",metrics.accuracy_score(Y_test, Y_pred))

        Accuracy: 0.6558441558441559
```

It is evident from the above image the decision tree classifier managed to get an accuracy rate of **66% (0.65744)** based on comparing actual test sets output values and predicted values.

Random Forest:

Initially we have processed the dataset and put it up for the test with random forest classifier. Firstly, we import all the required packages and read in the CSV file as shown below.

```
In [9]: import pandas as pd
        import numpy as np
        from sklearn.ensemble import RandomForestClassifier
        from sklearn.model_selection import train_test_split
        from sklearn import metrics
```

```
In [14]: data = pd.read_csv(r"D:\Final-Project\Data\dataset2.csv")
         df=data.iloc[:,1:13]
         df.head()
```

Out[14]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_A
0	1	0	0	1	0	5849	0.0	0	360	1	
1	1	1	1	1	0	4583	1508.0	128	360	1	
2	1	1	0	1	1	3000	0.0	66	360	1	
3	1	1	0	0	0	2583	2358.0	120	360	1	
4	1	0	0	1	0	6000	0.0	141	360	1	

Similarly, here also we split the data in 80/20 ratio, where 80% of the data will be fed into the classifier to train it and 20% is kept aside to test its accuracy. The following snippets of code shows the same.

```
In [17]: X = np.array(df.drop('Loan_Status', axis=1))
         y = np.array(df['Loan_Status'])
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)
         clf = RandomForestClassifier(n_estimators=100, random_state=42)
         clf.fit(X_train, y_train)
         accuracy = clf.score(X_test, y_test)
         print('accuracy', accuracy)

         accuracy 0.7154471544715447
```

Comparing actual test sets output values and predicted values Random Forest classifier managed to get an accuracy rate of **71% (0.71544).**

9. Conclusions

At this preliminary stage of our study, we have compared the performance of two important machine learning classification techniques based on datasets regarding house loans approval. Based on this current stage we come to a conclusion that random forest classifier produces more accurate predictions than decision tree classifier.

10. Future Scope

Till now the work done only reflects a part of what is actually intended, we have compared the performance of the two algorithms and found out that the performance of random forest is much better to that of the decision tree with respect to our datasets. Next part of the intended work i.e., recommendation of a better system or engine is to be done in the future.

11. References:

[1] Pranckevičius, Tomas and Virginijus Marcinkevicius. “Comparison of Naive Bayes, Random Forest, Decision Tree, Support Vector Machines, and Logistic Regression Classifiers for Text Reviews Classification.” *Balt. J. Mod. Comput.* 5 (2017): n. pag.

[2] Sathyadevan, Shiju and Remya R. Nair. “Comparative Analysis of Decision Tree Algorithms: ID3, C4.5 and Random Forest.” *CI 2015* (2015).

[3] Datkhile, Apurva, et al. "Statistical Modelling on Loan Default Prediction Using Different Models." *IJRESM* 3.3 (2020): 3-5.

[4] Buschjäger, Sebastian, and Katharina Morik. "Decision tree and random forest implementations for fast filtering of sensor data." *IEEE Transactions on Circuits and Systems I: Regular Papers* 65.1 (2017): 209-222.