LOAN PREDICTION APPROVAL SYSTEM WITH WEB APPLICATION

A PROJECT REPORT

Submitted by

RAJKIRAN S S **312019104053**

KARTHIKEYAN P **312019104025**

in partial fulfilment for the award of the degreeof

BACHELOR IN ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

JEPPIAAR SRR ENGINEERING COLLEGE, PADUR

ANNA UNIVERSITY: CHENNAI 600 025

MARCH 2022

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **" LOAN PREDICTION APPROVAL SYSTEM WITH WEB APPLICATION "** is the bonafide work of **RAJKIRAN S S (312019104053), KARTHIKEYAN P (312019104025)** who carried out the project work under by supervision.

Mr.S. RAMAKRISHNAN MCA.,M.E.,(Ph.D) Mr.S. RAMAKRISHNAN MCA.,M.E.,(Ph.D)

HEAD OF THE DEPARTMENT

Assistant Professor,
Information Technology,
Jeppiaar SRR Engineering
College,
Old Mamallapuram Road,
Padur, Chennai – 603 103

INTERNAL GUIDE

Assistant Professor,
Information Technology,
Jeppiaar SRR Engineering
College,
Old Mamallapuram Road,
Padur, Chennai – 603 103

Submitted for the examination held on_____.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEGEMENT

We take this opportunity to express our profound gratitude and deep regard toour beloved **Founder Chairman (Late)Col. Dr. JEPPIAAR M.A., B.L., Ph.D.,** for enlightening our lives and showering heavenly blessings forever.

We also express our heartfelt thanks to our **Chairman and Managing Director Dr. REGEENA JEPPIAAR B.E., M.B.A., Ph.D.,** for her kind cooperation and keen interest for the success of the project.

We express our thanks to our **Head of the DepartmentMr. S. RAMAKRISHNAN MCA, M.E.,** for his valuable suggestions and guidance for the development and completion of this project.

We are immensely happy to accord the warmth of gratitude to our **Director Mr. MURLI SUBRAMANIAN** for being the beacon in all our endeavours.

We express our profound gratitude to our **Principal Dr. M. SASIKUMAR M.TECH., PH.D.,** for bringing out novelty in all executions.

We are highly thankful to our project **Internal Guide Mr. S. RAMAKRISHNAN MCA, M.E.** for guidance and encouragement in carrying outthis project work.

We are much obliged to all our teaching and non-teaching staff members for their valuable information and constructive criticism that immensely contributed to the development of the project.

Above all, we wish to avail this opportunity to express a sense of gratitude and love to our beloved parents and friends for their moral support and constant strength at various stages of our project

TABLE OF CONTENTS

ABSTRACT

When any financial institution lends the money to the person, it is always been a high risk. Today data is increasing with the rapid pace in the banks, therefore the bankersneed to evaluate the person's data before giving the loan. It can be a big headache to evaluate the data. This problem is solved by analyzing and training the data by using oneof the Machine Learning algorithms. For this, we have generated a model for theprediction that the person will get the loan or not. The primary objective of this paper is to check whether the person can get the loan or not by evaluating the data with the help of decision tree classifiers which can gives the accurate result for the prediction.

Keywords—Loan, Machine Learning, Random Forest Classifier, Data training

1. INTRODUCTION

1.1 GENERAL

The immense increase in capitalism, the fast-paced development and instantaneous changes in the lifestyle has us in awe. Emi, loans at nominal rate, housing loans, vehicle loans, these are some of the few words which have skyrocketed from the past few years. The needs, wants and demands have never been increased this before. People gets loan from banks; however, it may be baffling for the bankers to judge who can pay back the loan nevertheless the bank shouldn't be in loss. Banks earn most of their profits through the loan sanctioning. Generally, banks pass loan after completing the numerous verification processes despite all these, it is still not confirmed that the borrower will pay back the loan or not. To get over the dilemma, I have built up a prediction model which says if the loan has been assigned in the safe hands or not. Government agencies like keep under surveillance why one person got a loan and the other person could not. In Machine Learning techniques which include classification and prediction can be applied to conquer this to a brilliant extent. Machine learning has eased today's world bydeveloping these prediction models. Here we will be using the fine techniques of machine learning – Decision tree algorithm to build this prediction model for loan assessment. It is as so because decision tree gives accuracy in the prediction and is often used in the industry for these models.

1.2 MACHINE LEARNING

1.2.1 GENERAL

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would

expect.

Fig 1. Machine Learning

The term Machine Learning was coined by Arthur Samuel in 1959, an American pioneer in the field of computer gaming and artificial intelligence and stated that “it gives computers the ability to learn without being explicitly programmed”.

And in 1997, Tom Mitchell gave a “well-posed” mathematical and relational definition that “A computer program is said to learn from experience E with respectto some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

Machine learning involves a computer to be trained using a given data set, and use this training to predict the properties of a given new data. For example, we can train a computer by feeding it 1000 images of cats and 1000 more images which are not of a cat, and tell each time to the computer whether a picture is cat or not. Then if we show the computer a new image, then from the above training, the computer should be able to tell whether this new image is a cat or not.

Let’s try to understand Machine Learning in layman terms. Consider you are trying to toss a paper to a dustbin.

After first attempt, you realize that you have put too much force in it. After second attempt, you realize you are closer to target but you need to increase your throw angle.

What is happening here is basically after every throw we are learning something and improving the end result. We are programmed to learn from our experience.

This implies that the tasks in which machine learning is concerned offers a fundamentally operational definition rather than defining the field in cognitive terms. This follows Alan Turing's proposal in his paper "Computing Machinery and Intelligence", in which the question "Can machines think?" is replaced with the question "Can machines do what we (as thinking entities) can do?" Within the field of data analytics, machine learning is used to devise complex modelsand algorithms that lend themselves to prediction; in commercial use, this is known as predictive analytics. These analytical models allow researchers, data scientists,engineers, and analysts to "produce reliable, repeatable decisions and results" and uncover "hidden insights" through learning from historical relationships and trends inthe data set(input).

Suppose that you decide to check out that offer for a vacation . You browse through the travel agency website and search for a hotel. When you look at a specific hotel, just below the hotel description there is a section titled "You might also like these hotels". This is a common use case of Machine Learning called "Recommendation Engine". Again, many data points were used to train a model in order to predict what will be the best hotels to show you under that section, based on a lot of information they already know about you.

So if you want your program to predict, for example, traffic patterns at a busy intersection (task T), you can run it through a machine learning algorithm with data about past traffic patterns (experience E) and, if it has successfully "learned", it will then do better at predicting future traffic patterns (performance measure P).
The highly complex nature of many real-world problems, though, often means that inventing specialized algorithms that will solve them perfectly every time is impractical, if not impossible. Examples of machine learning problems include, "Is this cancer?", "Which of these people are good friends with each other?", "Will this person like this

movie?" such problems are excellent targets for Machine Learning, and in fact machine learning has been applied such problems with great success.

When do we need Machine Learning?

When do we need machine learning rather than directly program our computers to carry out the task at hand? Two aspects of a given problem may call for the use of programs that learn and improve on the basis of their "experience": the problem's complexity and the need for adaptivity.

Tasks That Are Too Complex to Program.

• Tasks Performed by Animals/Humans: There are numerous tasks that we human beings perform routinely, yet our introspection concerning how we do them is not sufficiently elaborate to extract a well-defined program. Examples of such tasks include driving, speech recognition, and image understanding. In all of these tasks, state of the art machine learning programs, programs that "learn from their experience," achieve quite satisfactory results, once exposed to sufficiently many training examples.

• Tasks beyond Human Capabilities: Another wide family of tasks that benefit from machine learning techniques are related to the analysis of very large and complex data sets: astronomical data, turning medical archives into medical knowledge, weather prediction, analysis of genomic data, Web search engines, and electronic commerce. With more and more available digitally recorded data, it becomes obvious that there are treasures of meaningful information buried in data archives that are way too large and too complex for humans to make sense of. Learning to detect meaningful patterns in large and complex data sets is a promising domain in which the combination of programs that learn with the almost unlimited memory capacity and ever increasing processing speed of computers opens up new horizons. Adaptivity. One limiting feature of programmed tools is their rigidity – once the program has been written down n and installed, it stays unchanged. However, many tasks change over time or from one user to another. Machine learning tools – programs whose behavior adapts to their input data – offer a solution to such issues; they are, by nature, adaptive to changes in the environment they interact with. Typical successful applications of machine learning to

such problems include programs that decode handwritten text, where a fixed program can adapt to variations between the handwriting of different users; spam detection programs, adapting automatically to changes in the nature of spam e-mails; and speech recognition programs.

Terminologies of Machine Learning

- **Model**

  A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called hypothesis.

- **Feature**

  A feature is an individual measurable property of our data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.

  Note: Choosing informative, discriminating and independent features is a crucial step for effective algorithms. We generally employ a feature extractor to extract the relevant features from the raw data.

- **Target (Label)**

  A target variable or label is the value to be predicted by our model. For the fruit example discussed in the features section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.

- **Training**

  The idea is to give a set of inputs(features) and it's expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.

- **Prediction**

  Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

The figure shown below clears the above concepts:

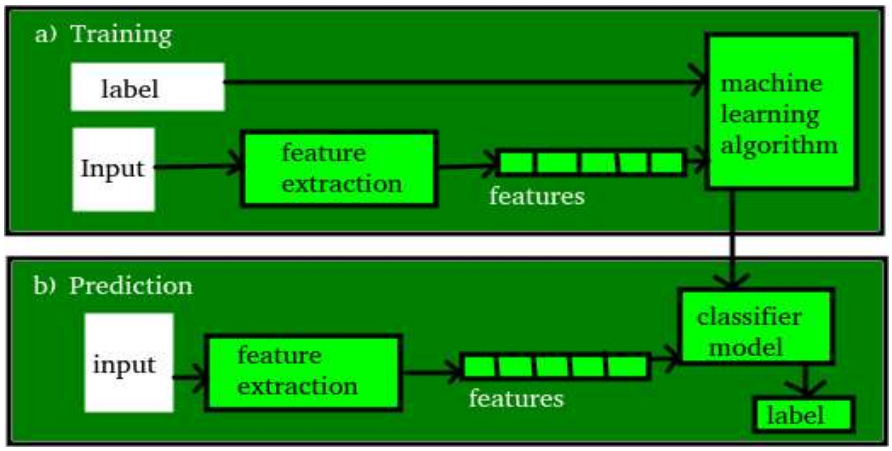


Fig-2 Training and Prediction

1.2.1.1 MACHINE LEARNING TYPES

1.2.2.1 **GENERAL** :

Learning is, of course, a very wide domain. Consequently, the field of machine learning has branched into several subfields dealing with different types of learning tasks. We give a rough taxonomy of learning paradigms, aiming to provide some perspective of where the content sits within the wide field of machine learning.

Terms frequently used are:

- Labeled data: Data consisting of a set of training examples, where each example is a pair consisting of an input and a desired output value (also called the supervisory signal, labels, etc)
- Classification**:** The goal is to predict discrete values, e.g. {1,0}, {True, False}, {spam, not spam}.
- Regression**:** The goal is to predict continuous values, e.g. home prices.

There some variations of how to define the types of Machine Learning Algorithms but commonly they can be divided into categories according to their purpose and the main categories are the following:

- Supervised learning

- Unsupervised Learning
- Semi-supervised Learning
- Reinforcement Learning

1.2.1.2 Supervised Learning

- I like to think of supervised learning with the concept of function approximation, where basically we train an algorithm and in the end of the process we pick the function that best describes the input
  data, the one that for a given X makes the best estimation of y (X -> y). Most of the time we are not able to figure out the true function that always make the correct predictions and other reason is that the algorithm rely upon an assumption made by humans about how the computer should learn and this assumptions introduce a bias.
- Here the human experts act as the teacher where we feed the computer with training data containing the input/predictors and we show it the correct answers (output) and from the data the computer should be able to learn the patterns.
- Supervised learning algorithms try to model relationships and dependencies between the target prediction output and the input features such that we can predict the output values for new data based on those relationships which it learned from the previous data sets.

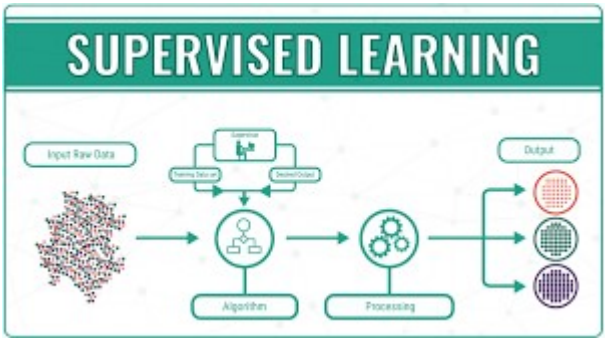


Fig 3: Supervised Learning

Draft

- Predictive Model
- we have labelled data
- The main types of supervised learning problems include regression and classification problems

List of Common Algorithms

- Nearest Neighbour
- Naive Bayes
- Decision Trees
- Linear Regression
- Support Vector Machines (SVM)
- Neural Networks

1.2.1.3 Unsupervised Learning

- The computer is trained with unlabelled data.
- Here there's no teacher at all, actually the computer might be able to teach you new things after it learns patterns in data, these algorithms a particularly useful in cases where the human expert doesn't know what to look for in the data.
- are the family of machine learning algorithms which are mainly used in pattern detection and descriptive modelling. However, there are no output categories or labels here based on which the algorithm can try to model relationships. These algorithms try to use techniques on the input data to mine for rules, detect patterns, and summarize and group the data points which help in deriving meaningful insights and describe the data better to the users.

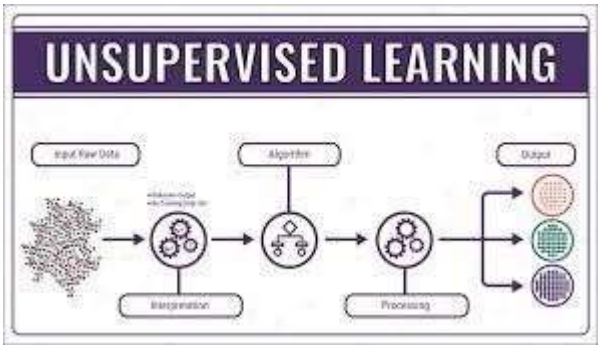


Fig 4: Unsupervised Learning

- Descriptive Model
- The main types of unsupervised learning algorithms include Clustering algorithms and Association rule learning algorithms.

List of Common Algorithms

- k-means clustering, Association Rules

1.2.1.4 Semi-Supervised Learning

In the previous two types, either there are no labels for all the observation in the dataset or labels are present for all the observations. Semi-supervised learning falls in between these two. In many practical situations, the cost to label is quite high, since it requires skilled human experts to do that. So, in the absence of labels in the majority of the observations but present in few, semi-supervised algorithms are the best candidates for the model building. These methods exploit the idea that even though the group memberships of the unlabeled data are unknown, this data carries important information about the group parameters.

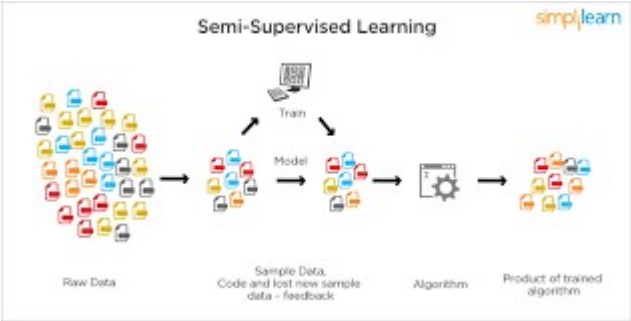


Fig 5: Semi-Supervised Learning

1.2.1.5 Reinforcement Learning

method aims at using observations gathered from the interaction with the environment to take actions that would maximize the reward or minimize the risk. Reinforcement learning algorithm (called the agent) continuously learns from the environment in an iterative fashion. In the process, the agent learns from its experiences of the environment until it explores the full range of possible states.

Reinforcement Learning is a type of Machine Learning, and thereby also a branch of Artificial Intelligence. It allows machines and software agents to automatically determine the ideal behaviour within a specific context, in order to maximize its performance. Simple reward feedback is required for the agent to learn its behaviour; this is known as the reinforcement signal.

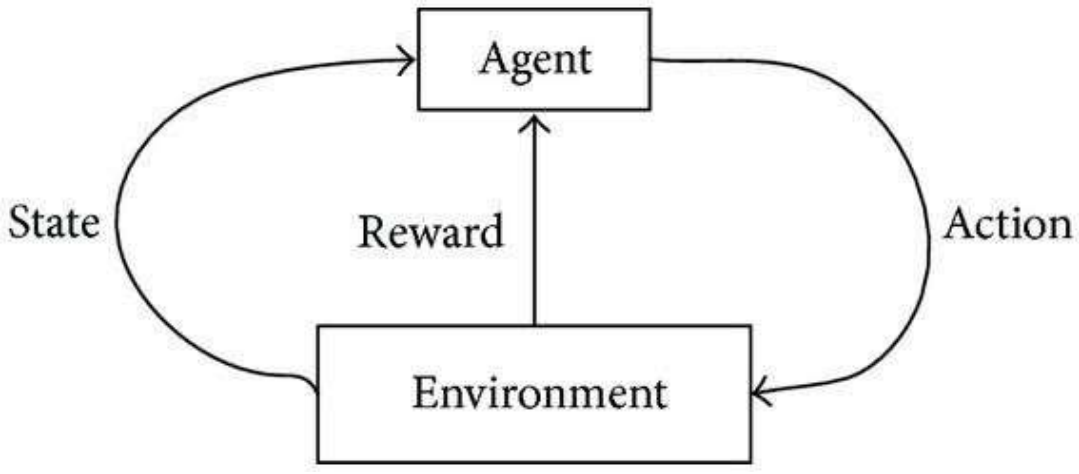


Fig 6: Reinforcement Learning

There are many different algorithms that tackle this issue. As a matter of fact, Reinforcement Learning is defined by a specific type of problem, and all its solutions are classed as Reinforcement Learning algorithms. In the problem, an agent is supposed decide the best action to select based on his current state. When this step is repeated, theproblem is known as a Markov Decision Process.

In order to produce intelligent programs (also called agents), reinforcement learning goes through the following steps:

1. Input state is observed by the agent.
2. Decision making function is used to make the agent perform an action.
3. After the action is performed, the agent receives reward or reinforcement from the environment.
4. The state-action pair information about the reward is stored.

List of Common Algorithms

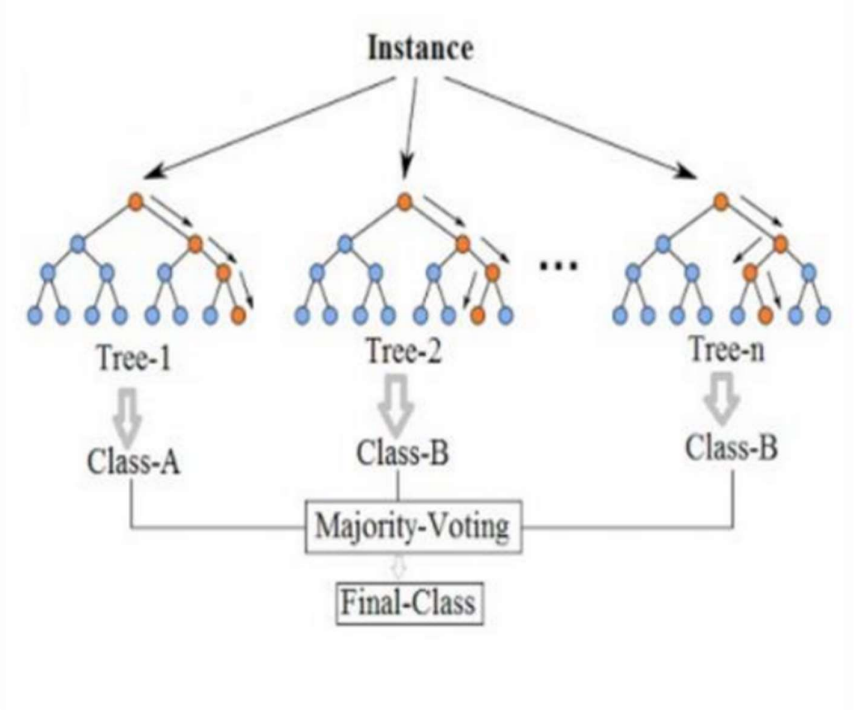
- Q-Learning
- Temporal Difference (TD)
- Deep Adversarial Networks

Use cases:

Some applications of the reinforcement learning algorithms are computer played board games (Chess, Go), robotic hands, and self-driving cars.

1.3 RANDOM FOREST CLASSIFIER

Random forests are an ensemble method used for classification. In Random Forest, we grow multiple trees as opposed to a single tree in Decision Tree model. But the question arises why to use multiple trees when the same work can be done by a single tree as well. One of the major problems of Decision Tree is overfitting which gives us a very bad predictive model and adding multiple trees in the random forest introduces randomness which in turn gets rid of overfitting and gives us a very superior predictive model. To classify a new object based on attributes, each tree gives a classification and we say the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest) and in case of regression, it takes the average of outputs by different trees.



Diagrammatic representation of the working of a simple random forest

The methodology includes construction of decision trees of the given training data and matching the test data with these. Random forests are used to rank the importance of variables in a classification problem. To measure the importance of a variable in a data setDn = {(Xi,Yi) }i=1 to n we fit a random forest to the data. During the fitting process the error for each data point is calculated and averaged over the forest. To measure the importance of the i-th feature after training, the values of the i-th feature are permuted among the training data and the error is again computed on this data set. The importance score for the i-th feature is computed by averaging the difference in error before and after the permutation for all the trees. Normalization of the score is done by the standard deviation of these differences. Features which produce large values for this score are more important than features which produce small values. Random forests provide information about the importance of a variable and also the proximity of the data points with one another. But, the question arises why we should choose Random Forest over all the other equally good or even better machine learning algorithms like Neural Networks, Support Vector machines, etc. available to us, the answer is quite simple random forest could be said to be the "worry free" approach. In case of Random Forest we almost need to do almost no hyperparameter tuning

(except the number of trees present in the forest, generally more the tree better is the algorithm and depth of the tree), while in case of neural networks and support vector machines there are lot of knobs to be tuned: choosing the "right" kernel, regularization penalties, slack variable, number of hidden layers in a neural network, and the list goes on. Both random forests and SVMs are non-parametric models (i.e., the complexity grows as the number of training samples increases). Training a non-parametric model can thus be more expensive, computationally, compared to a generalized linear model, for example. The more trees we have,

the more expensive it is to build a random forest. Also, we can end up with a lot of support vectors in SVMs; in the worst-case scenario, we have as many support vectors as we have samples in the training set. Although, there are multi-class SVMs, the typical implementation for multi-class classification is One-vs.-All; thus, we have to train an SVM for each class -- in contrast, decision trees or random forests, which can handle multiple classes out of the box.

METHODOLOGY:

Algorithm for Construction of Random Forest is:

Step 1: Let the number of training cases be "n" and let the number of variables included in the classifier be "m".

Step 2: Let the number of input variables used to make decision at the node of a tree be "p". We assume that p is always less than "m".

Step 3: Choose a training set for the decision tree by choosing k times with replacement from all "n" available training cases by taking a bootstrap sample. Bootstrapping computes for a given set of data the accuracy in terms of deviation from the mean data. It is usually used for hypothesis tests. Simple block bootstrap can be used when the data can be divided into non- overlapping blocks. But, moving block bootstrap is used when we divide the data into overlapping blocks where the portion "k" of overlap between first and second block is always equal to the "k" overlap between second and third overlap and so on. We use the remaining cases to estimate the error of the tree. Bootstrapping is also used for estimating the properties of the given training data.

Step 4: For each node of the tree, randomly choose variables on which to search for the best split. New data can be predicted by considering the majority votes in the tree. Predict data which is not in the bootstrap sample. And compute the aggregate.

Step 5: Calculate the best split based on these chosen variables in the training set. Base the decision at that node using the best split.

Step 6: Each tree is fully grown and not pruned. Pruning is used to cut of the leaf nodes so that the tree can grow further. Here the tree is completely retained.

Step 7: The best split is one with the least error i.e. the least deviation from the observed data set.

ADVANTAGES:

1. It provides accurate predictions for many types of applications
2. It can measure the importance of each feature with respect to the training data set.
3. Pairwise proximity between samples can be measured by the training data set.

DISADVANTAGES:

1. For data including categorical variables with different number of levels, random forests are biased in favour of those attributes with more levels.
2. If the data contain groups of correlated features of similar relevance for the output, then smaller groups are favoured over larger groups Factors on which the construction of decision tree depends are:
    1. The shape of the decision to use in each node.
    2. The type of predictor to use in each leaf.
    3. The splitting objective to optimize in each node.
    4. The method for injecting randomness into the trees.

APPLICATIONS:

1. Is used for image classification for pixel analysis.
2. Is used in the field of Bioinformatics for complex data analysis.
3. Is used for video segmentation (high dimensional data)

2 LITERATURE REVIEW

2.1 GENERAL

Machine learning helps to learn data from its own experience and in predictionand decision of data. It has brought the revolution in the field of computer science because data is the most important thing in the world. To analyze data, Machine Learning has given many solutions by its algorithms. I have gone through many papers before developing this model to collect information. In paper (1), authors have tried to reduce the efforts of banks by generating a model by various machine learning models and explained which of the methods can be accurate. The paper was divided into 4 sections- data collection, comparison of various machine learning models on data, training of data and testing. They have done by the mining the previous data. Author has also mentioned that this system can be integrated with the Automatic-processing system in future. In paper (2), authors basically want to know about the nature of client by analyzing various variables. In this paper, author compares various attributes by exploratory data analysis technique, seven graphs were generated and short-term loan was preferred in the paper. In paper (3), the authors have explained about whether it is safe or not to give loans to the person. They have used map function to predict the records. In paper (4), authors have used decision tree induction algorithm to implement a model and tried to review credit scoring of mortgage loans and criteria for the applicants. Credit score helps in sanctioning of the loan. They have developed a model to predict it is safe or not for loan sanctioning and it was concluded that mostly low-income applicants receive loan approval as they are likely to repay their loan back. Dataset is collected from the Kaggle. In paper (6), authors have tried to evaluate the credit risks and to identify the loan repayment prediction. They have made use of decision tree algorithm in the paper. A test set is utilized to

approve the form. In paper (7), authors have used data mining to develop a model and the system has 3 components- preprocessing, classification and databaseupdation. KNN, Binning and naïve-Bayesian algorithm was used for prediction model. A hybrid of naïve-Bayesian and K-means was used for improving the efficiency of the system. In paper (8), authors have used three types of machine algorithm which are Logistic Regression, Decision tree and Random Forest algorithm. After analyzing the data sets on these models, Random Forest algorithm come out with the highest accuracy of all the model. The main aim of this model was to sanction loan to applicants in a short span of time. In paper (9), an ensemble model is used by the authors which gives better accuracy and efficiency than the individual models for prediction. In paper (10), authors have used classifier technique in which combination of min-max normalization and KNN algorithm is used which gives 75.08% accuracy result. R programming language is used to build KNN soring credit model. This paper focuses on predicting the loan status in commercial bank.
It cannot be clear which model is best because each model has its own specification. Therefore, an efficient model is important to be developed whichcan give higher accuracy.

efficiency of the system. In paper (8), authors have used three types of machine algorithm which are Logistic Regression, Decision tree and Random Forest algorithm. After analyzing the data sets on these models, Random Forest algorithm come out with the highest accuracy of all the model. The main aim of this model was to sanction loan to applicants in a short span of time. In paper (9), an ensemble model is used by the authors which gives better accuracy and efficiency than the individual models for prediction. In paper (10), authors have used classifier technique in which combination of min-max normalization and KNN algorithm is used which gives 75.08% accuracy result. R programming language is used to build KNN soring credit model. This paper focuses on

predicting the loan status in commercial bank.

It cannot be clear which model is best because each model has its own specification. Therefore, an efficient model is important to be developed whichcan give higher accuracy.

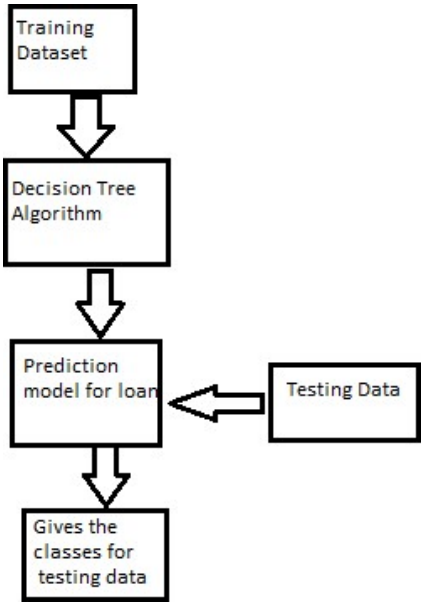
3 IMPLEMENTATION OF MODEL

3.1 EXISTING SYSTEM

Banks need to analyze for the person who applies for the loan will repay the loan or not. Sometime it happens that customer has provided partial data to the bank, in this case person may get the loan without proper verification and bank may end up with loss. Bankers cannot analyze the huge amounts of data manually, it may become a bigheadache to check whether a person will repay its loan or not. It is very much necessary to know the person getting loan is going in safe hand or not. So, it is pretty much important to have a automated model which should predict the customer getting the loan will repay the loan or not.

3.2 PROPOSED SYSTEM

I have developed a prediction model for Loan sanctioning which will predict whether theperson applying for loan will get loan or not. The major objective of this project is to derive patterns from the datasets which are used for the loan sanctioning process and create a model based on the patterns derived in the previous step. This model is developed by using the one of the machine



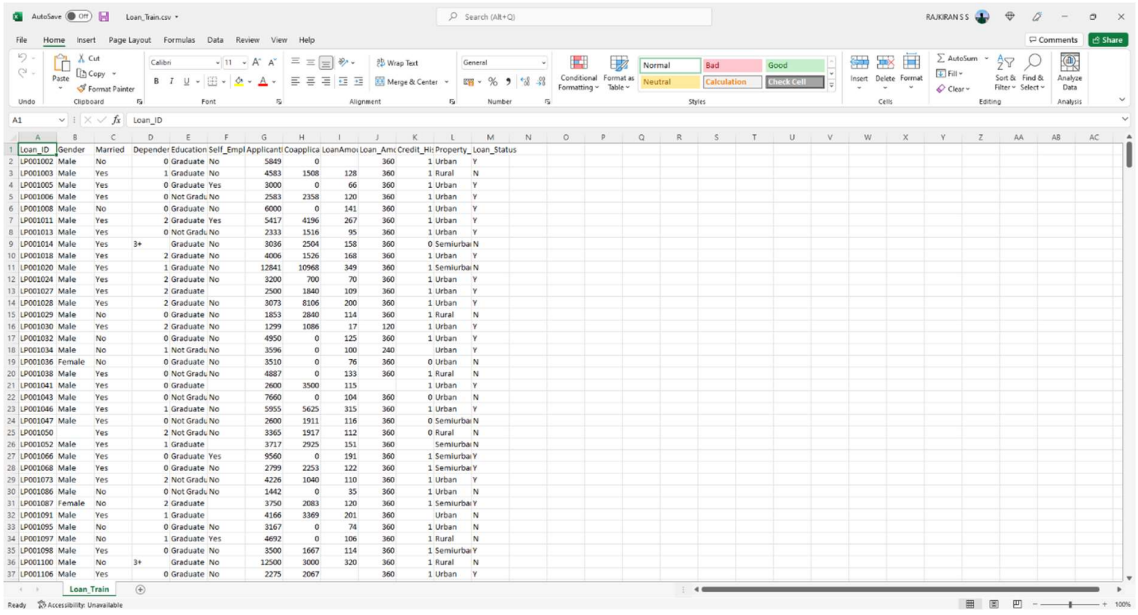
learning algorithms.

Fig 9: Architecture of propsed system

In the proposed model for loan prediction, Dataset is split into training and testing data. After then training datasets are trained using the decision tree algorithm and a predictionmodel is developed using the algorithm. Testing datasets are then given to model for theprediction of loan. The motive of this paper is to predict the defaults who will repay the loan or not. Various libraries like pandas, numpy have been used. After the loading of datasets, Data Preprocessing like missing value treatment of numerical and categorical is done by checking the values. Numerical and categorical values are segregated. Outliers and frequency analysis are done, outliers are checked by getting the boxplot diagram of attributes.

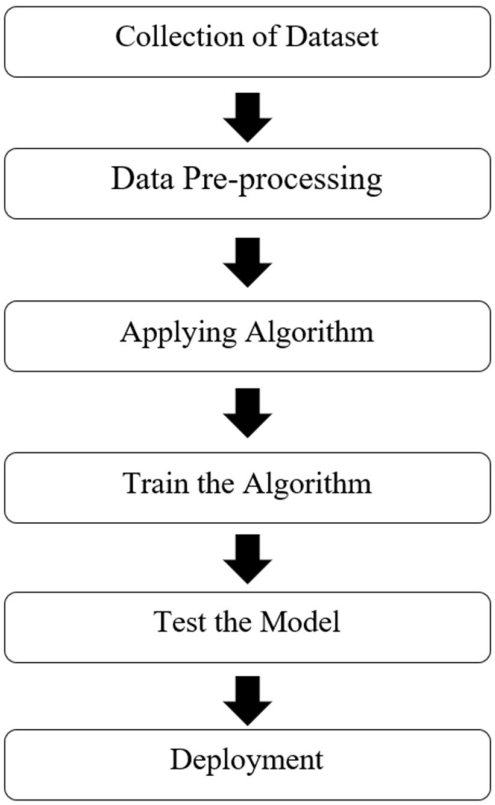
3.2.1 DATA SET

Datasets are gathered from Kaggle. Data set is now provided to Machine learningmodels on the basis of this facts this version is trained. Data sets are divided into Existing and New Customers. Every new applicant info act as a fact test set. After the operation of testing, model expect whether the brand-new applicant is in case for approval of the loan or now not primarily based upon the inference it concludes on the idea of training information sets.



Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
LP001002	Male	No	0	Graduate	No	5849	0		360	1	Urban	Y
LP001003	Male	Yes	1	Graduate	No	4583	1508	128	360	1	Rural	N
LP001005	Male	Yes	0	Graduate	Yes	3000	0	66	360	1	Urban	Y
LP001006	Male	Yes	0	Not Graduate	No	2583	2358	120	360	1	Urban	Y
LP001008	Male	No	0	Graduate	No	6000	0	141	360	1	Urban	Y
LP001011	Male	Yes	2	Graduate	Yes	5417	4196	267	360	1	Urban	Y
LP001013	Male	Yes	0	Not Graduate	No	2333	1516	95	360	1	Urban	Y
LP001014	Male	Yes	3+	Graduate	No	3036	2504	158	360	0	Semiurban	N
LP001018	Male	Yes	2	Graduate	No	4006	1526	168	360	1	Urban	Y
LP001020	Male	Yes	1	Graduate	No	12841	10968	349	360	1	Semiurban	N
LP001024	Male	Yes	2	Graduate	No	3200	700	70	360	1	Urban	Y
LP001027	Male	Yes	2	Graduate		2500	1840	109	360	1	Urban	Y
LP001028	Male	Yes	2	Graduate	No	3073	8106	200	360	1	Urban	Y
LP001029	Male	No	0	Graduate	No	1853	2840	114	360	1	Rural	N
LP001030	Male	Yes	2	Graduate	No	1299	1086	17	120	1	Urban	Y
LP001032	Male	No	0	Graduate	No	4950	0	125	360	1	Urban	Y
LP001034	Male	No	1	Not Graduate	No	3596	0	100	240		Urban	Y
LP001036	Female	No	0	Graduate	No	3510	0	76	360	0	Urban	N
LP001038	Male	Yes	0	Not Graduate	No	4887	0	133	360	1	Rural	N
LP001041	Male	Yes	0	Graduate		2600	3500	115		1	Urban	Y
LP001043	Male	Yes	0	Not Graduate	No	7660	0	104	360	0	Urban	N
LP001046	Male	Yes	1	Graduate	No	5955	5625	315	360	1	Urban	Y
LP001047	Male	Yes	0	Not Graduate	No	2600	1911	116	360	0	Semiurban	N
LP001050		Yes	2	Not Graduate	No	3365	1917	112	360	0	Rural	N
LP001052	Male	Yes	1	Graduate		3717	2925	151	360		Semiurban	N
LP001066	Male	Yes	0	Graduate	Yes	9560	0	191	360	1	Semiurban	Y
LP001068	Male	Yes	0	Graduate	No	2799	2253	122	360	1	Semiurban	Y
LP001073	Male	Yes	2	Not Graduate	No	4226	1040	110	360	1	Urban	Y
LP001086	Male	No	0	Not Graduate	No	1442	0	35	360	1	Urban	N
LP001087	Female	No	2	Graduate		3750	2083	120	360	1	Semiurban	Y
LP001091	Male	Yes	1	Graduate		4166	3369	201	360		Urban	N
LP001095	Male	No	0	Graduate	No	3167	0	74	360	1	Urban	N
LP001097	Male	No	1	Graduate	Yes	4692	0	106	360	1	Rural	N
LP001098	Male	Yes	0	Graduate	No	3500	1667	114	360	1	Semiurban	Y
LP001100	Male	No	3+	Graduate	No	12500	3000	320	360	1	Rural	N
LP001106	Male	Yes	0	Graduate	No	2275	2067		360	1	Urban	Y

3.3 IMPLEMENTATION



3.3.1 STEPS TO SOLVE MACHINE LEARNING PROJECTS

1. start
2. data selection
3. data description : A story of data is all abot and the features present in the data
4. performing both statistical and graphical data analysis
5. data tranformation and derivation of new attributes , if necessary
6. selection of machine learning algorithms based on the patterns observed in EDA
7. data standardization and normalization

8. creation of train and test data sets
9. model training using machine learning algorithms
10. calculation of model accuracy : both training and testing accuracy
11. hyper parameter tuning to achieve a better accuracy
12. saving the created model file
13. Deployment strategies for model
14. production deployment and testing.

3.3.2 DATA PREPROCESSING:

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model.

```
In [12]: df.isnull().sum()

Out[12]: Loan_ID              0
         Gender              26
         Married              6
         Dependents          30
         Education            0
         Self_Employed       64
         ApplicantIncome      0
         CoapplicantIncome    0
         LoanAmount          22
         Loan_Amount_Term    28
         Credit_History     100
         Property_Area        0
         Loan_Status          0
         dtype: int64
```

The next step of data preprocessing is to handle missing data in the datasets. If our dataset contains some missing data, then it may create a huge problem for our machine learning model. Hence it is necessary to handle missing values present in the dataset.

```
In [13]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1234 entries, 0 to 1233
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Loan_ID            1234 non-null   object
 1   Gender             1208 non-null   object
 2   Married            1228 non-null   object
 3   Dependents         1204 non-null   object
 4   Education          1234 non-null   object
 5   Self_Employed      1170 non-null   object
 6   ApplicantIncome    1234 non-null   int64
 7   CoapplicantIncome  1234 non-null   float64
 8   LoanAmount         1212 non-null   float64
 9   Loan_Amount_Term   1206 non-null   float64
 10  Credit_History     1134 non-null   float64
 11  Property_Area      1234 non-null   object
 12  Loan_Status        1234 non-null   object
dtypes: float64(4), int64(1), object(8)
memory usage: 125.5+ KB
```

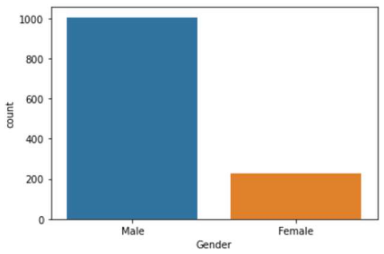
```
In [20]: df['Gender'] = df['Gender'].fillna(df['Gender'].mode()[0])
         df['Married'] = df['Married'].fillna(df['Married'].mode()[0])
         df['Dependents'] = df['Dependents'].fillna(df['Dependents'].mode()[0])
         df['Self_Employed'] = df['Self_Employed'].fillna(df['Self_Employed'].mode()[0])
```

```
In [21]: df.isnull().sum()

Out[21]: Loan_ID             0
         Gender              0
         Married             0
         Dependents          0
         Education           0
         Self_Employed       0
         ApplicantIncome     0
         CoapplicantIncome   0
         LoanAmount          0
         Loan_Amount_Term    0
         Credit_History      0
         Property_Area       0
         Loan_Status         0
         dtype: int64
```

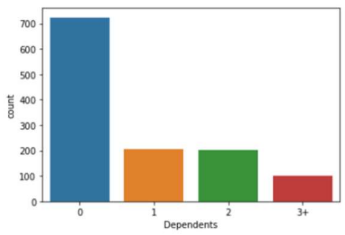
3.3.3 EXPLORATORY DATA ANALYSIS:

Exploratory Data Analysis refers to the critical process of performing initial investigations on data so as to discover patterns,to spot anomalies,to test hypothesis and to check assumptions with the help of summary statistics and graphical representations.

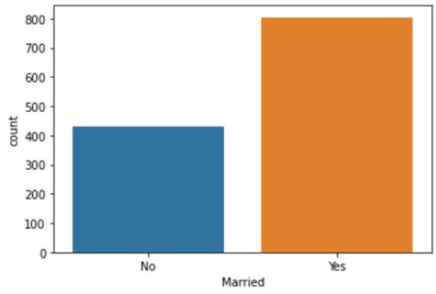
```
In [22]: # categorical data
         import seaborn as sns
         sns.countplot(df['Gender'])
Out[22]: <AxesSubplot:xlabel='Gender', ylabel='count'>
```




```
In [23]: sns.countplot(df.Dependents)
Out[23]: <AxesSubplot:xlabel='Dependents', ylabel='count'>
```




```
In [24]: sns.countplot(df.Married)
Out[24]: <AxesSubplot:xlabel='Married', ylabel='count'>
```



3.3.4 SPLIT DATASETS:

In machine learning data pre-processing, we divide our dataset into a training set and test set. This is one of the crucial steps of data pre-processing as by doing this, we can enhance the performance of our machine learning model.

Suppose, if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models.

If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:

Training Set: A subset of dataset to train the machine learning model, and we already know the output.

Test set: A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.30, random_state=42)
```

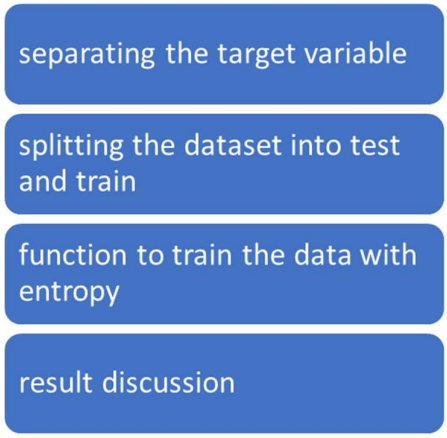
```
In [60]: x
Out[60]:
```

	Credit_History	ApplicantIncomeLog	LoanAmountLog	Loan_Amount_Term_Log	Total_Income_Log	Male	Yes	1	2	3+	Not Graduate	Yes	Semiurban	Urba
0	1.0	8.674026	5.131421	5.886104	8.674026	1	0	0	0	0	0	0	0	
1	1.0	8.430109	4.852030	5.886104	8.714568	1	1	1	0	0	0	0	0	
2	1.0	8.006368	4.189655	5.886104	8.006368	1	1	0	0	0	0	1	0	
3	1.0	7.856707	4.787492	5.886104	8.505323	1	1	0	0	0	1	0	0	
4	1.0	8.699515	4.948760	5.886104	8.699515	1	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1229	1.0	8.936956	5.468060	5.886104	8.936956	1	1	0	1	0	0	0	0	
1230	0.0	8.435549	5.209486	5.886104	8.435549	0	0	0	0	0	0	1	1	
1231	1.0	8.167636	5.407172	5.886104	8.703673	1	1	1	0	0	0	0	1	
1232	0.0	7.613325	5.209486	5.886104	7.613325	0	0	0	0	0	0	1	1	
1233	0.0	6.932448	5.209486	5.886104	6.932448	0	0	0	0	0	0	1	1	

1234 rows × 14 columns

```
In [61]: y

Out[61]: 0       Y
         1       N
         2       Y
         3       Y
         4       Y
                ..
         1229    Y
         1230    N
         1231    Y
         1232    N
         1233    N
         Name: Loan_Status, Length: 1234, dtype: object
```

3.3.5 Processes for Loan Prediction:




```
# specify input and output attributes
x = df.drop(columns=['Loan_Status'], axis=1)
y = df['Loan_Status']
```

3.3.6 MODEL TRAINING:

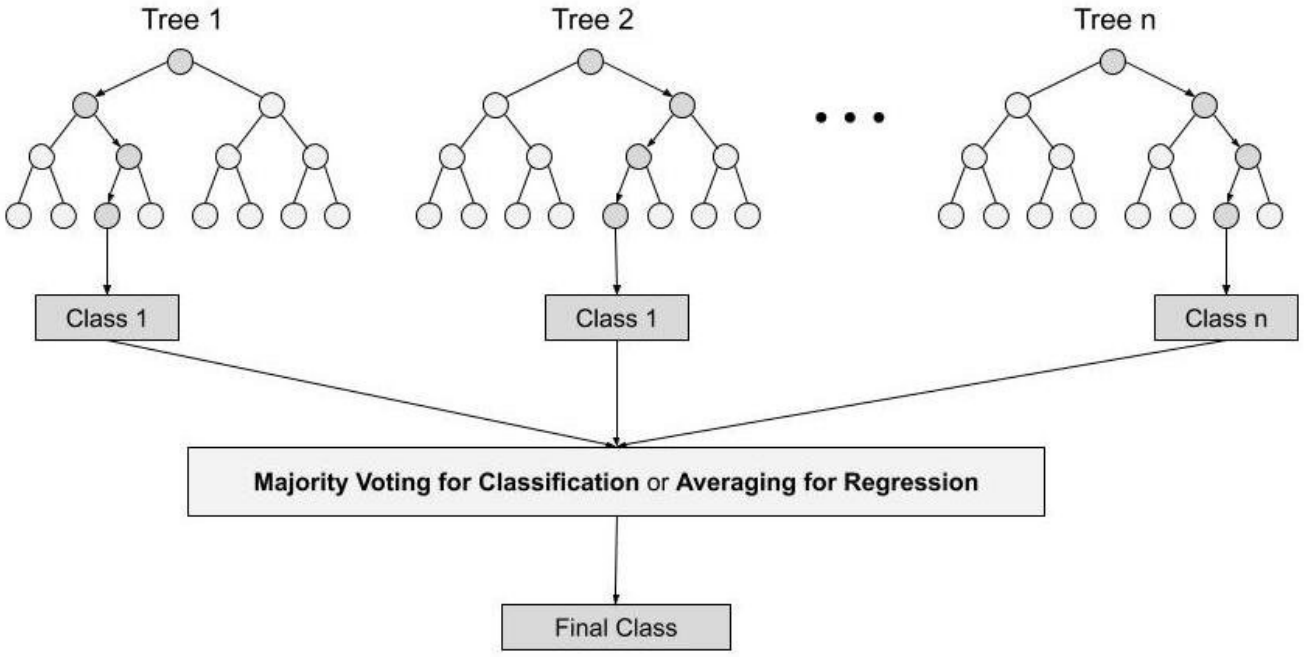
A training model is a dataset that is used to train an ML algorithm. It consists of the sample output data and the corresponding sets of input data that have an influence on the output. The training model is used to run the input data through the algorithm to correlate the processed output against the sample output. The result from this correlation is used to modify the model.

This iterative process is called “model fitting”. The accuracy of the training dataset or the validation dataset is critical for the precision of the model.

Model training in machine language is the process of feeding an ML algorithm with data to help identify and learn good values for all attributes involved. There are several types of machine learning models, of which the most common ones are supervised and unsupervised learning.

Supervised learning is possible when the training data contains both the input and output values. Each set of data that has the inputs and the expected output is called a supervisory signal. The training is done based on the deviation of the processed result from the documented result when the inputs are fed into the model.

Unsupervised learning involves determining patterns in the data. Additional data is then used to fit patterns or clusters. This is also an iterative process that improves the accuracy based on the correlation to the expected patterns or clusters. There is no reference output dataset in this method.



3.3.7 SOURCE CODE

```
# importing libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
df = pd.read_csv("loan_data_set.csv")
df.head()
df.tail()
df.shape
df.info()
dups = df.duplicated()
print('No of rows duplicated = %d' %(dups.sum()))
df['ApplicantIncome']
df[['ApplicantIncome', 'LoanAmount']]
df.columns
df.isnull().sum()
df.info()
# handle numerical missing data
df['LoanAmount'] = df['LoanAmount'].fillna(df['LoanAmount'].mean())
df['Loan_Amount_Term'] =
df['Loan_Amount_Term'].fillna(df['Loan_Amount_Term'].mean())
df['Credit_History'] = df['Credit_History'].fillna(df['Credit_History'].mean())
df.isnull().sum()
# handle categorical missing data
df['Gender'].mode()[0]
```

```
# handle categorical missing data
df['Married'].mode()[0]
# handle categorical missing data
df['Dependents'].mode()[0]
# handle categorical missing data
df['Self_Employed'].mode()[0]
df['Gender'] = df['Gender'].fillna(df['Gender'].mode()[0])
df['Married'] = df['Married'].fillna(df['Married'].mode()[0])
df['Dependents'] = df['Dependents'].fillna(df['Dependents'].mode()[0])
df['Self_Employed'] =
df['Self_Employed'].fillna(df['Self_Employed'].mode()[0])
df.isnull().sum()
# categorical data
import seaborn as sns
sns.countplot(df['Gender'])
sns.countplot(df.Dependents)
sns.countplot(df.Married)
df.columns
# numerical data
sns.distplot(df.CoapplicantIncome)
sns.distplot(df.Credit_History)
df.head()
df['Total_income'] = df['ApplicantIncome']+df['CoapplicantIncome']
df.head()
df['ApplicantIncomeLog'] = np.log(df['ApplicantIncome'])
sns.distplot(df.ApplicantIncomeLog)
df['CoapplicantIncomeLog'] = np.log(df['CoapplicantIncome'])
sns.distplot(df["ApplicantIncomeLog"])
df['LoanAmountLog'] = np.log(df['LoanAmount'])
```

```
sns.distplot(df["LoanAmountLog"])
df['Loan_Amount_Term_Log'] = np.log(df['Loan_Amount_Term'])
sns.distplot(df["Loan_Amount_Term_Log"])
df['Total_Income_Log'] = np.log(df['Total_income'])
sns.distplot(df["Total_Income_Log"])
df.Loan_Status.value_counts()
df.Education.value_counts()
df.info()
d1 = pd.get_dummies(df['Gender'], drop_first= True)
d2 = pd.get_dummies(df['Married'], drop_first= True)
d3 = pd.get_dummies(df['Dependents'], drop_first= True)
d4 = pd.get_dummies(df['Education'], drop_first= True)
d5 = pd.get_dummies(df['Self_Employed'], drop_first= True)
d6 = pd.get_dummies(df['Property_Area'], drop_first= True)

df1 = pd.concat([df, d1, d2, d3, d4, d5, d6], axis = 1)
df=df1
cols = ['Gender', 'Married', "Dependents", "Education", "Self_Employed",
'Property_Area']
df = df.drop(columns=cols, axis=1)
#skewness - measure of how much the probability distribution of a random
variable deviates from the normal distribution.
test = pd.read_csv("test.csv")
# filling numerical missing data
test['LoanAmount']=test['LoanAmount'].fillna(test['LoanAmount'].mean())
test['Loan_Amount_Term']=test['Loan_Amount_Term'].fillna(test['Loan_Amou
nt_Term'].mean())
test['Credit_History']=test['Credit_History'].fillna(test['Credit_History'].mean())
# filling categorical missing data
```

```
test['Gender']=test['Gender'].fillna(test['Gender'].mode()[0])
test['Married']=test['Married'].fillna(test['Married'].mode()[0])
test['Dependents']=test['Dependents'].fillna(test['Dependents'].mode()[0])
test['Self_Employed']=test['Self_Employed'].fillna(test['Self_Employed'].mode(
)[0])
test['Total_income'] = test['ApplicantIncome']+test['CoapplicantIncome']
# apply log transformation to the attribute
test['ApplicantIncomeLog'] = np.log(test['ApplicantIncome'])
test['CoapplicantIncomeLog'] = np.log(test['CoapplicantIncome'])
test['LoanAmountLog'] = np.log(test['LoanAmount'])
test['Loan_Amount_Term_Log'] = np.log(test['Loan_Amount_Term'])
test['Total_Income_Log'] = np.log(test['Total_income'])
cols = ['ApplicantIncome', 'CoapplicantIncome', "LoanAmount",
"Loan_Amount_Term", "Total_income", 'Loan_ID', 'CoapplicantIncomeLog']
test = test.drop(columns=cols, axis=1)
t1 = pd.get_dummies(test['Gender'], drop_first= True)
t2 = pd.get_dummies(test['Married'], drop_first= True)
t3 = pd.get_dummies(test['Dependents'], drop_first= True)
t4 = pd.get_dummies(test['Education'], drop_first= True)
t5 = pd.get_dummies(test['Self_Employed'], drop_first= True)
t6 = pd.get_dummies(test['Property_Area'], drop_first= True)
df1 = pd.concat([test, t1, t2, t3, t4, t5, t6], axis = 1)
test=df1
cols = ['Gender', 'Married', "Dependents", "Education", "Self_Employed",
'Property_Area']
test = test.drop(columns=cols, axis=1)
# specify input and output attributes
x = df.drop(columns=['Loan_Status'], axis=1)
y = df['Loan_Status']
```

```
# randomforest classifier
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
model.fit(x_train, y_train)
print("Accuracy is", model.score(x_test, y_test)*100)
#Random_Forest_Classifier
from sklearn.metrics import confusion_matrix,ConfusionMatrixDisplay
y_pred = model.predict(x_test)
cm = confusion_matrix(y_test, y_pred)
disp =
ConfusionMatrixDisplay(confusion_matrix=cm,display_labels=model.classes_)
disp.plot()
plt.show()
from sklearn.metrics import classification_report
y_pred = model.predict(x_test)
print(classification_report(y_test,y_pred))
```

WEB PAGE SOURCE CODE:

```
<!doctype html>
<html lang="en">

<head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-
```

```
eOJMYsd53ii+scO/bJGFsiCZc+5NDVN2yr8+0RDqr0Ql0h+rP48ckxlpbzKgwr
a6" crossorigin="anonymous">
   <link href="https://unpkg.com/tailwindcss@^2/dist/tailwind.min.css"
rel="stylesheet">
   <title>prediction</title>
</head>

<body>
   <section class="text-gray-600 body-font">
      <div class="container px-5 py-24 mx-auto">
         <div class="flex flex-col text-center w-full mb-20">

            <h1 class="sm:text-3xl text-2xl font-medium title-font mb-4 text-
gray-900">Loan prediction project</h1>
            <p class="lg:w-2/3 mx-auto leading-relaxed text-base">fill the form
for prediction</p>
            <br><br>
            <h2><b>{{prediction_text}}</b></h2>
         </div>
         <div>

         </div>
         <a class="btn btn-primary" href="./index" role="button">Back</a>
         <form action='/predict' method='POST'>
            <div class="mb-3">
               <label for="exampleFormControlInput1" class="form-label">
gender</label>
               <select class="form-select" id="gender" name="gender" aria-
label="Default select example">
```

```
  <option selected>-- select gender --</option>
  <option value="Male">Male</option>
  <option value="Female">Female</option>
</select>
            </div>
            <div class="mb-3">
               <label for="exampleFormControlInput1" class="form-label">
married status</label>
               <select class="form-select" id="married" name="married" aria-
label="Default select example">
  <option selected>-- select married status --</option>
  <option value="Yes">Yes</option>
  <option value="No">No</option>
</select>
            </div>
            <div class="mb-3">
               <label for="exampleFormControlInput1" class="form-
label">Dependents</label>
               <select class="form-select" id="dependents" name="dependents"
aria-label="Default select example">
  <option selected>-- select dependents --</option>
  <option value="0">0</option>
  <option value="1">1</option>
  <option value="2">2</option>
  <option value="3+">3+</option>
</select>
            </div>
            <div class="mb-3">
               <label for="exampleFormControlInput1" class="form-
```

```
label">Education</label>
                <select class="form-select" id="education" name="education" aria-
label="Default select example">
  <option selected>-- select education --</option>
  <option value="Graduate">Graduate</option>
  <option value="Not Graduate">Not Graduate</option>
</select>
            </div>
            <div class="mb-3">
               <label for="exampleFormControlInput1" class="form-
label">Self_Employed</label>
                <select class="form-select" id="employed" name="employed" aria-
label="Default select example">
  <option selected>-- select Self_Employed --</option>
  <option value="Yes">Yes</option>
  <option value="No">No</option>
</select>
            </div>
            <div class="mb-3">
               <label for="exampleFormControlInput1" class="form-
label">Credit_History</label>
                <select class="form-select" id="credit" name="credit" aria-
label="Default select example">
  <option selected>-- select Credit_History --</option>
  <option value="1.000000">1.000000</option>
  <option value="0.000000">0.000000</option>
  <option value="0.842199">0.842199</option>
</select>
            </div>
```

```
            <div class="mb-3">
                <label for="exampleFormControlInput1" class="form-
label">Property_Area</label>
                <select class="form-select" id="area" name="area" aria-
label="Default select example">
  <option selected>-- select Property_Area --</option>
  <option value="Semiurban">Semiurban</option>
  <option value="Urban">Urban</option>
  <option value="Rural">Rural</option>
</select>
            </div>
            <div class="mb-3">
                <label for="exampleFormControlInput1" class="form-label">Enter
ApplicantIncome</label>
                <input type="text" class="form-control" id="ApplicantIncome"
name="ApplicantIncome" placeholder="ApplicantIncome">
            </div>
            <div class="mb-3">
                <label for="exampleFormControlInput1" class="form-label">Enter
CoapplicantIncome</label>
                <input type="text" class="form-control" id="CoapplicantIncome"
name="CoapplicantIncome" placeholder="CoapplicantIncome">
            </div>
            <div class="mb-3">
                <label for="exampleFormControlInput1" class="form-label">Enter
LoanAmount</label>
                <input type="text" class="form-control" id="LoanAmount"
name="LoanAmount" placeholder="LoanAmount">
            </div>
```

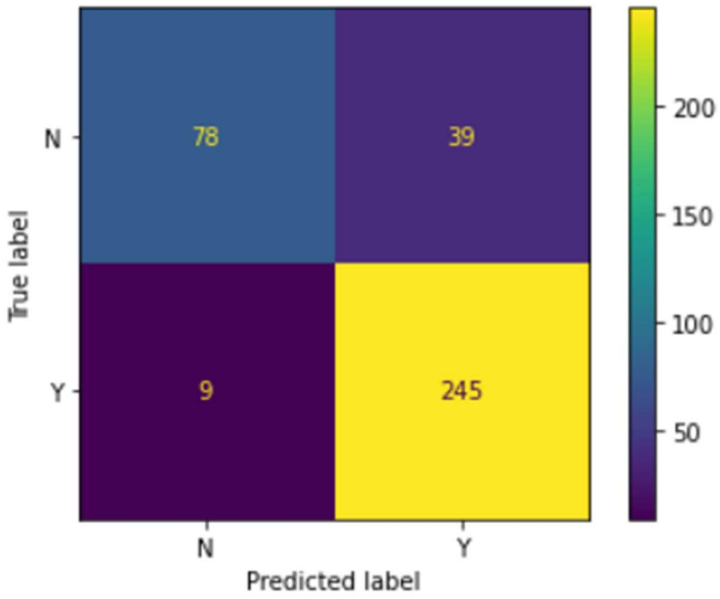
```
            <div class="mb-3">
              <label for="exampleFormControlInput1" class="form-label">Enter
Loan_Amount_Term</label>
              <input type="text" class="form-control" id="Loan_Amount_Term"
name="Loan_Amount_Term" placeholder="Loan_Amount_Term">
            </div>

            <button type="submit" class="btn btn-primary">Predict</button>
          </form>



      </div>
    </section>


    <!-- Optional JavaScript; choose one of the two! -->

    <!-- Option 1: Bootstrap Bundle with Popper -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/js/bootstrap.bundle.min.js" integrity="sha384-
JEW9xMcG8R+pH31jmWH6WWP0WintQrMb4s7ZOdauHnUtxwoG2vI5DkL
tS3qm9Ekf" crossorigin="anonymous"></script>

    <!-- Option 2: Separate Popper and Bootstrap JS -->
    <!--
    <script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.9.1/dist/umd/popper.min.j
s" integrity="sha384-
SR1sx49pcuLnqZUnnPwx6FCym0wLsk5JZuNx2bPPENzswTNFaQU1RDvt3
```

```
wT4gWFG" crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta3/dist/js/bootstrap.min.js" integrity="sha384-
j0CNLUeiqtyaRmlzUHCPZ+Gy5fQu0dQ6eZ/xAww941Ai1SxSY+0EQqNXN
E6DZiVc" crossorigin="anonymous"></script>
    -->
</body>
</html>
```

4. RESULTS

The Confusion Matrix (CM) is used to analyze and determine the performance of the proposed loanprediction model.

- True Positive (TP), when both the actual and predicted values are positive (1)
- True Negative (TN), when both the actual and predicted values are negative (0)
- False Positive (FP), when the actual value is negative and the predicted value is positive (1)
- False Negative (FN), when the actual value is positive (1) and the predicted value isnegative(0)




```
In [ ]:

In [72]: from sklearn.metrics import classification_report
         y_pred = model.predict(x_test)
         print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
N	0.90	0.67	0.76	117
Y	0.86	0.96	0.91	254
accuracy			0.87	371
macro avg	0.88	0.82	0.84	371
weighted avg	0.87	0.87	0.86	371

Confusion Matrix is given for the test data after the prediction of the model for loan. It shows the performance ofthe model developed for the prediction.

Accuracy for the model:

$$\textbf{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

After the model is built, best score for the model is 87.06.

5. CONCLUSION

5.1 Future Work

In future, this model can be used to compare various machine learning algorithm generated prediction models and the model which will give higher accuracy will be chosen as the prediction model.

5.2 Conclusion

After this work, we are able to conclude that Decision tree version is extraordinary efficient and gives ahigher end result. We have developed a model which can easily predict that the person will repay its loan or not. we can see our model has reduced the efforts of bankers. Machine learning has helped a lot in developing this model which gives precise results.

6 REFERENCES

[1] Dataset Source, https://www.kaggle.com

[2] Scikit Learn – scikitlearn.org

[3] https://gist.github.com/UpasanaYadav/1ea9051c849c4331afdc3f9818068022

[4] https://github.com/ASHOKKUMAR-K/Machine-Learning-Projects/tree/master/02.Loan_Status_Prediction

[5] https://www.youtube.com/watch?v=QIUxPv5PJOY (Computer Science)