Prediction of Modernized Loan Approval System Based on Machine Learning Approach

DISSERTATION

Submitted in partial fulfillment of the
Requirements for the award of the degree

of

Bachelor of Technology

in

Computer Science & Engineering

By:

Pradhumn Gupta (09113202720)
Shekhar Gupta (10513202720)
Prabhneet Singh (08813202720)
Prabhjot Singh (01013207221)

Under The Guidance of:
Dr. Aashish Bharadwaj

Department of Computer Science & Engineering
Guru Tegh Bahadur Institute of Technology

Guru Gobind Singh Indraprastha University
Dwarka, New Delhi
Year 2020-2024

DECLARATION

I hereby declare that the thesis work entitled "***Prediction of Modernized Loan Approval System Based on Machine Learning Approach***" which is being submitted To Guru Gobind Singh Indrapastha University, in partial fulfilment of requirements for the award of degree of Bachelors of Technology (Computer Science and Engineering) is a Bonafide report of Minor Project carried out by us. The material contained in the report has not been submitted to any university or institution for the award of any degree.

Place: New Delhi

Date:

Name-Pradhumn Gupta

Roll No.-031/CSE2/2020

Name-Shekhar Gupta

Roll No.-45/CSE2/2020

Name-Prabhneet Singh

Roll No.-028/CSE2/2020

Name-Prabhjot Singh

Roll No.-LE-010/CSE2/2020

CERTIFICATE

This is to certify that Project Report entitled "**Prediction of Modernized Loan Approval System Based on Machine Learning Approach**" submitted by **Mr. Pradhumn Gupta, Mr. Shekhar Gupta, Mr. Prabhneet Singh** and **Mr. Prabhjot Singh** in partial fulfilment of the requirement for the award of degree Bachelors of Technology (Computer Science and Engineering) is a record of the original work carried out by us under our supervision.

Dr. Aashish Bhardwaj

(Project Guide)

Dr .**Aashish Bhardwaj**

(HOD of CSE Department)

Date:

ACKNOWLEDGEMENT

I am very thankful to **Mr. Aashish Bhardwaj** (Head of Department, Computer Science and Engineering Department) and all the faculty members of the Computer Science and Engineering Department of Guru Tegh Bahadur Institute of Technology. They all provided us with immense support and guidance for the project.

I would also like to express my gratitude to the university for providing us with the laboratories, infrastructure, testing facilities and environment which allowed us to work without any obstructions.

I would also like to appreciate the support provided to us by our lab assistants, seniors and our peer group who aided us with all the knowledge they had regarding various topics.

Pradhumn Gupta (09113202720)

Shekhar Gupta (10513202720)

Prabhneet Singh (08813202720)

Prabhjot Singh (LE-010)

B. Tech. (CSE)

GGSIPU

ABSTRACT

Technology has boosted the existence of human kind the quality of life they live. Every day we are planning to create something new and different. We have a solution for every other problem we have machines to support our lives and make us somewhat complete in the banking sector candidate gets proofs/ backup before approval of the loan amount. The application approved or not approved depends upon the historical data of the candidate by the system. Every day lots of people applying for the loan in the banking sector but Bank would have limited funds. In this case, the right prediction would be very beneficial using some classes-function algorithm. An example the logistic regression, random forest classifier, support vector machine classifier, etc. A Bank's profit and loss depend on the amount of the loans that is whether the Client or customer is paying back the loan. Recovery of loans is the most important for the banking sector. The improvement process plays an important role in the banking sector. The historical data of candidates was used to build a machine learning model using different classification algorithms. The main objective of this paper is to predict whether a new applicant granted the loan or not using machine learning models trained on the historical data set

TABLE OF CONTENTS

Contents

Prediction of Modernized Loan Approval System Based on Machine Learning Approach 1
DECLARATION 2
CERTIFICATE 3
ACKNOWLEDGEMENT 4
ABSTRACT 5
TABLE OF CONTENTS 6
INTRODUCTION 8
SYSTEM ANALYSIS 9
REQUIREMENT SPECIFICATIONS 11
Introduction to Python 12
K-Nearest Neighbors 19
KNN algorithm 19
The KNN algorithm work 19
How do we choose the factor K? 20
Conclusion 23
Decision tree introduction 23
What is a Decision Tree? 24
How can an algorithm be used to represent a tree 24
Types of Decision Tree 26
Lists of Algorithms 26
What is a Decision Tree? 29
Introduction to Logistics. 33
How do you explain logistics? 33
What are the 3 types of logistics? 33
What is logistic cycle? 35
Design and Implementation Constraints 38
Architecture Diagram: 41
Sequence Diagram: 42
Use Case Diagram: 43
Activity Diagram: 45
Collaboration Diagram: 47
MODULES 48
Machine learning Algorithm: 48
Prediction: 48
CODING AND TESTING 50

IMPLIMENTATION AND RESULT 2
Training the model 4
Model Selection 5
Ridge Classifier Performs better 6
App design 8
References 10

CHAPTER 1

INTRODUCTION

Aim: To determine the loan approval system using machine learning algorithms.

.

Synopsis:

Loan approval is a very important process for banking organizations. The systems approved or reject the loan applications. Recovery of loans is a major contributing parameter in the financial statements of a bank. It is very difficult to predict the possibility of payment of loan by the customer. In recent years many researchers worked on loan approval prediction systems. Machine Learning (ML) techniques are very useful in predicting outcomes for large amount of data. In this paper different machine learning algorithms are applied to predict the loan approval of customers..In this paper, various machine learning algorithms that have been used in past are discussed and their accuracy is evaluated. The main focus of this paper is to determine whether the loan given to a particular person or an organization shall be approved or not.

CHAPTER 2

SYSTEM ANALYSIS

EXISTING SYSTEM

The enhancement in the banking sector lots of people are applying for bank loans but the bank has its limited assets which it has to grant to limited people only, so finding out to whom the loan can be granted which will be a safer option for the bank is a typical process. In existing process, they are use RF algorithm in loan approval system. But the efficiency and accuracy was pretty low. Already banks are provide online transaction system, online bank account opening system, etc,. But there is no loan approval system in the banking sector. Then now we create a new system for loan approval. So now we move on to the proposed system.

DisAdvantage

*To apply the loan we need to go to bank to apply it

PROPOSED SYSTEM

The proposed model focuses on predicting the credibility of customers for loan repayment by analyzing their details. The input to the model is the customer details collected. On the output from the classifier, decision on whether to approve or reject the customer request can be made. Using different data analytics tools loan prediction and there severity can be forecasted. In this process it is required to train the data using different algorithms and then compare user data with trained data to predict the nature of loan. The training data set is now supplied to machine learning model; on the basis of this data set the model is trained. Every new applicant details filled at the time of application form acts as a test data set. After the operation of testing,

model predict whether the new applicant is a fit case for approval of the loan or not based upon the inference it conclude on the basis of the training data sets. By providing real time input on the web app. In our project, Logistic Regression gives high accuracy level compared with other algorithms. Finally, we are predicting the result via data visualization and display the predicted output using web app using flask.

Advantage

*No need to go to bank We can do the transaction from house,

* we can consume the time doing from home

CHAPTER 3

REQUIREMENT SPECIFICATIONS

INTRODUCTION

Prediction of modernized loan approval system based on machine learning approach is a loan approval system from where we can know whether the loan will pass or not. In this system, we take some data from the user like his monthly income, marriage status, loan amount, loan duration, etc. Then the bank will decide according to its parameters whether the client will get the loan or not. So there is a classification system, in this system, a training set is employed to make the model and the classifier may classify the data items into their appropriate class. A test dataset is created that trains the data and gives the appropriate result that, is the client potential and can repay the loan. Prediction of a modernized loan approval system is incredibly helpful for banks and also the clients. This system checks the candidate on his priority basis. Customer can submit his application directly to the bank so the bank will do the whole process, no third party or stockholder will interfere in it. And finally, the bank will decide that the candidate is deserving or not on its priority basis. The only object of this research paper is that the deserving candidate gets straight forward and quick results.

HARDWARE AND SOFTWARE SPECIFICATION

HARDWARE REQUIREMENTS

Hard disk : 500 GB and above.

Processor : i3 and above.

Ram : 4GB and above.

SOFTWARE REQUIREMENTS

Operating System : Windows 10
Software : python
Tools :Anaconda (Jupyter Note Book IDE)

TECHNOLOGIES USED

Programming Language: Python.

Introduction to Python

Python is a widely used general-purpose, high level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code.

Python is a programming language that lets you work quickly and integrate systems more efficiently.

It is used for:

web development (server-side),
software development,
mathematics,
System scripting.

What can Python do?

Python can be used on a server to create web applications.

Python can be used alongside software to create workflows.

Python can connect to database systems. It can also read and modify files.

Python can be used to handle big data and perform complex mathematics.

Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).

Python has a simple syntax similar to the English language.

Python has syntax that allows developers to write programs with fewer lines than some other programming languages.

Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.

Python can be treated in a procedural way, an object-orientated way or a functional way.

Good to know

The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.

Python 2.0 was released in 2000, and the 2.x versions were the prevalent releases until December 2008. At that time, the development team made the decision to release version 3.0, which contained a few relatively small but significant changes that were not backward compatible with the 2.x versions. Python 2 and 3 are very similar, and some features of Python 3 have been backported to Python 2. But in general, they remain not quite compatible.

Both Python 2 and 3 have continued to be maintained and developed, with periodic release updates for both. As of this writing, the most recent versions available are 2.7.15 and 3.6.5. However, an official End Of Life date of January 1, 2020 has been established for Python 2, after which time it will no longer be maintained.

Python is still maintained by a core development team at the Institute, and Guido is still in charge, having been given the title of BDFL (Benevolent Dictator For Life) by the

Python community. The name Python, by the way, derives not from the snake, but from the British comedy troupe Monty Python's Flying Circus, of which Guido was, and presumably still is, a fan. It is common to find references to Monty Python sketches and movies scattered throughout the Python documentation.

It is possible to write Python in an Integrated Development Environment, such as Thonny, Pycharm, Netbeans or Eclipse which are particularly useful when managing larger collections of Python files.

Python Syntax compared to other programming languages

Python was designed to for readability, and has some similarities to the English language with influence from mathematics.

Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.

Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

Python is Interpreted

Many languages are compiled, meaning the source code you create needs to be translated into machine code, the language of your computer's processor, before it can be run. Programs written in an interpreted language are passed straight to an interpreter that runs them directly.

This makes for a quicker development cycle because you just type in your code and run it, without the intermediate compilation step.

One potential downside to interpreted languages is execution speed. Programs that are compiled into the native language of the computer processor tend to run more quickly than interpreted programs. For some applications that are particularly computationally intensive, like graphics processing or intense number crunching, this can be limiting.

In practice, however, for most programs, the difference in execution speed is measured in milliseconds, or seconds at most, and not appreciably noticeable to a human user. The

expediency of coding in an interpreted language is typically worth it for most applications.

For all its syntactical simplicity, Python supports most constructs that would be expected in a very high-level language, including complex dynamic data types, structured and functional programming, and object-oriented programming.

Additionally, a very extensive library of classes and functions is available that provides capability well beyond what is built into the language, such as database manipulation or GUI programming.

Python accomplishes what many programming languages don't: the language itself is simply designed, but it is very versatile in terms of what you can accomplish with it.

Machine learning

Introduction:

Machine learning (ML) is the scientific study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to perform the task. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or infeasible to develop a conventional algorithm for effectively performing the task.

Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a field of study within machine learning, and focuses on exploratory data analysis through learning. In its application across business problems, machine learning is also referred to as predictive analytics.

Machine learning tasks:

Machine learning tasks are classified into several broad categories. In supervised learning, the algorithm builds a mathematical model from a set of data that contains both the inputs and the desired outputs. For example, if the task were determining whether an image contained a certain object, the training data for a supervised learning algorithm would include images with and without that object (the input), and each image would have a label (the output) designating whether it contained the object. In special cases, the input may be only partially available, or restricted to special feedback. Semi algorithms develop mathematical models from incomplete training data, where a portion of the sample input doesn't have labels.

Classification algorithms and regression algorithms are types of supervised learning. Classification algorithms are used when the outputs are restricted to a limited set of values. For a classification algorithm that filters emails, the input would be an incoming email, and the output would be the name of the folder in which to file the email. For an algorithm that identifies spam emails, the output would be the prediction of either "spam" or "not spam", represented by the Boolean values true and false. Regression algorithms are named for their continuous outputs, meaning they may have any value within a range. Examples of a continuous value are the temperature, length, or price of an object.

In unsupervised learning, the algorithm builds a mathematical model from a set of data that contains only inputs and no desired output labels. Unsupervised learning algorithms are used to find structure in the data, like grouping or clustering of data points. Unsupervised learning can discover patterns in the data, and can group the inputs into categories, as in feature learning. Dimensionality reduction is the process of reducing the number of "features", or inputs, in a set of data.

Active learning algorithms access the desired outputs (training labels) for a limited set of inputs based on a budget and optimize the choice of inputs for which it will acquire training labels. When used interactively, these can be presented to a human user for labeling. Reinforcement learning algorithms are given feedback in the form of positive or negative reinforcement in a dynamic environment and are used in autonomous vehicles or in learning to play a game against a human opponent. Other specialized algorithms in machine learning include topic modeling, where the computer program is given a set of natural language documents and finds other documents that cover similar topics. Machine learning algorithms can be used to find the

unobservable probability density function in density estimation problems. Meta learning algorithms learn their own inductive bias based on previous experience. In developmental robotics, robot learning algorithms generate their own sequences of learning experiences, also known as a curriculum, to cumulatively acquire new skills through self-guided exploration and social interaction with humans. These robots use guidance mechanisms such as active learning, maturation, motor synergies, and imitation.

Types of learning algorithms:

The types of machine learning algorithms differ in their approach, the type of data they input and output, and the type of task or problem that they are intended to solve.

Supervised learning:

Supervised learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs. The data is known as training data, and consists of a set of training examples. Each training example has one or more inputs and the desired output, also known as a supervisory signal. In the mathematical model, each training example is represented by an array or vector, sometimes called a feature vector, and the training data is represented by a matrix. Through iterative optimization of an objective function, supervised learning algorithms learn a function that can be used to predict the output associated with new inputs. An optimal function will allow the algorithm to correctly determine the output for inputs that were not a part of the training data. An algorithm that improves the accuracy of its outputs or predictions over time is said to have learned to perform that task.

Supervised learning algorithms include classification and regression. Classification algorithms are used when the outputs are restricted to a limited set of values, and regression algorithms are used when the outputs may have any numerical value within a range. Similarity learning is an area of supervised machine learning closely related to regression and classification, but the goal is to learn from examples using a similarity function that measures how similar or related two objects are. It has applications in ranking, recommendation systems, visual identity tracking, face verification, and speaker verification.

In the case of semi-supervised learning algorithms, some of the training examples are missing training labels, but they can nevertheless be used to improve the quality of a model. In weakly supervised learning, the training labels are noisy, limited, or imprecise; however, these labels are often cheaper to obtain, resulting in larger effective training sets.

Unsupervised learning:

Unsupervised learning algorithms take a set of data that contains only inputs, and find structure in the data, like grouping or clustering of data points. The algorithms, therefore, learn from test data that has not been labeled, classified or categorized. Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data. A central application of unsupervised learning is in the field of density estimation in statistics, though unsupervised learning encompasses other domains involving summarizing and explaining data features.

Cluster analysis is the assignment of a set of observations into subsets (called *clusters*) so that observations within the same cluster are similar according to one or more pre designated criteria, while observations drawn from different clusters are dissimilar. Different clustering techniques make different assumptions on the structure of the data, often defined by some *similarity metric* and evaluated, for example, by *internal compactness*, or the similarity between members of the same cluster, and *separation*, the difference between clusters. Other methods are based on *estimated density* and *graph connectivity*.

Semi-supervised learning:

Semi-supervised learning falls between unsupervised learning (without any labeled training data) and supervised learning (with completely labeled training data). Many machine-learning researchers have found that unlabeled data, when used in conjunction with a small amount of labeled data, can produce a considerable improvement in learning accuracy.

K-Nearest Neighbors

Introduction

In four years of the analytics built more than 80% of classification models and just 15- 20% regression models. These ratios can be more or less generalized throughout the industry. The reason of a bias towards classification models is that most analytical problem involves making a decision. For instance will a customer attrite or not, should we target customer X for digital campaigns, whether customer has a high potential or not etc. This analysis is more insightful and directly links to an implementation roadmap. In this article, we will talk about another widely used classification technique called K-nearest neighbors (KNN). Our focus will be primarily on how does the algorithm work and how does the input parameter effect the output/prediction.

KNN algorithm

KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry. To evaluate any technique we generally look at 3 important aspects:

Ease to interpret output

Calculation time

Predictive Power

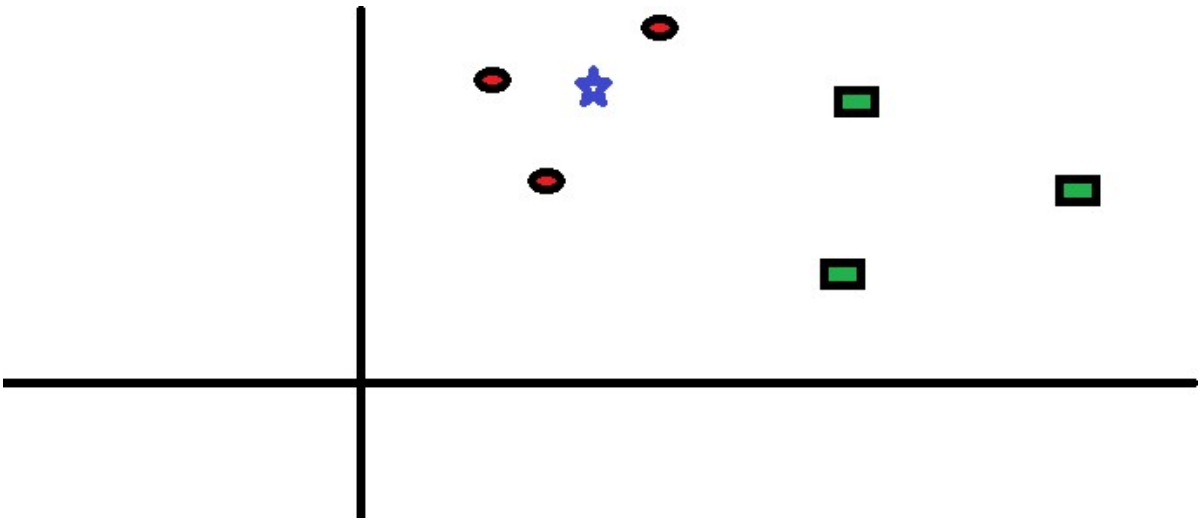
Let us take a few examples to place KNN in the scale:

	Logistic Regression	CART	Random Forest	KNN
1. Ease to interpret output	2	3	1	3
2. Calculation time	3	2	1	3
3. Predictive Power	2	2	3	2

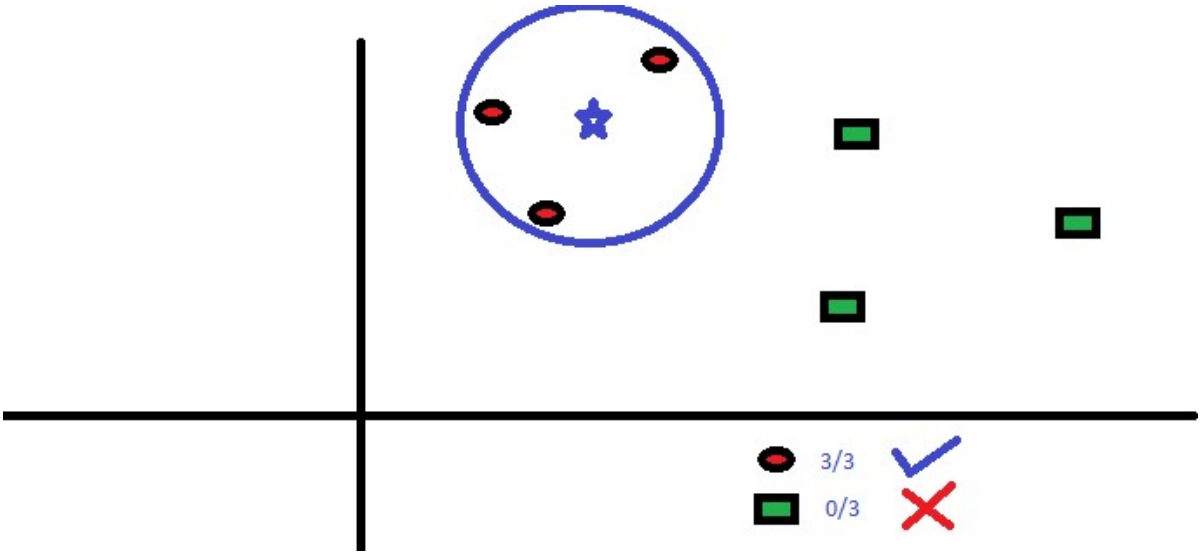
KNN algorithm fairs across all parameters of considerations. It is commonly used for its easy of interpretation and low calculation time.

The KNN algorithm work

Let's take a simple case to understand this algorithm. Following is a spread of red circles (RC) and green squares (GS):



You intend to find out the class of the blue star (BS). BS can either be RC or GS and nothing else. The "K" is KNN algorithm is the nearest neighbors we wish to take vote from. Let's say K = 3. Hence, we will now make a circle with BS as center just as big as to enclose only three data data points on the plane. Refer to following diagram for more details:

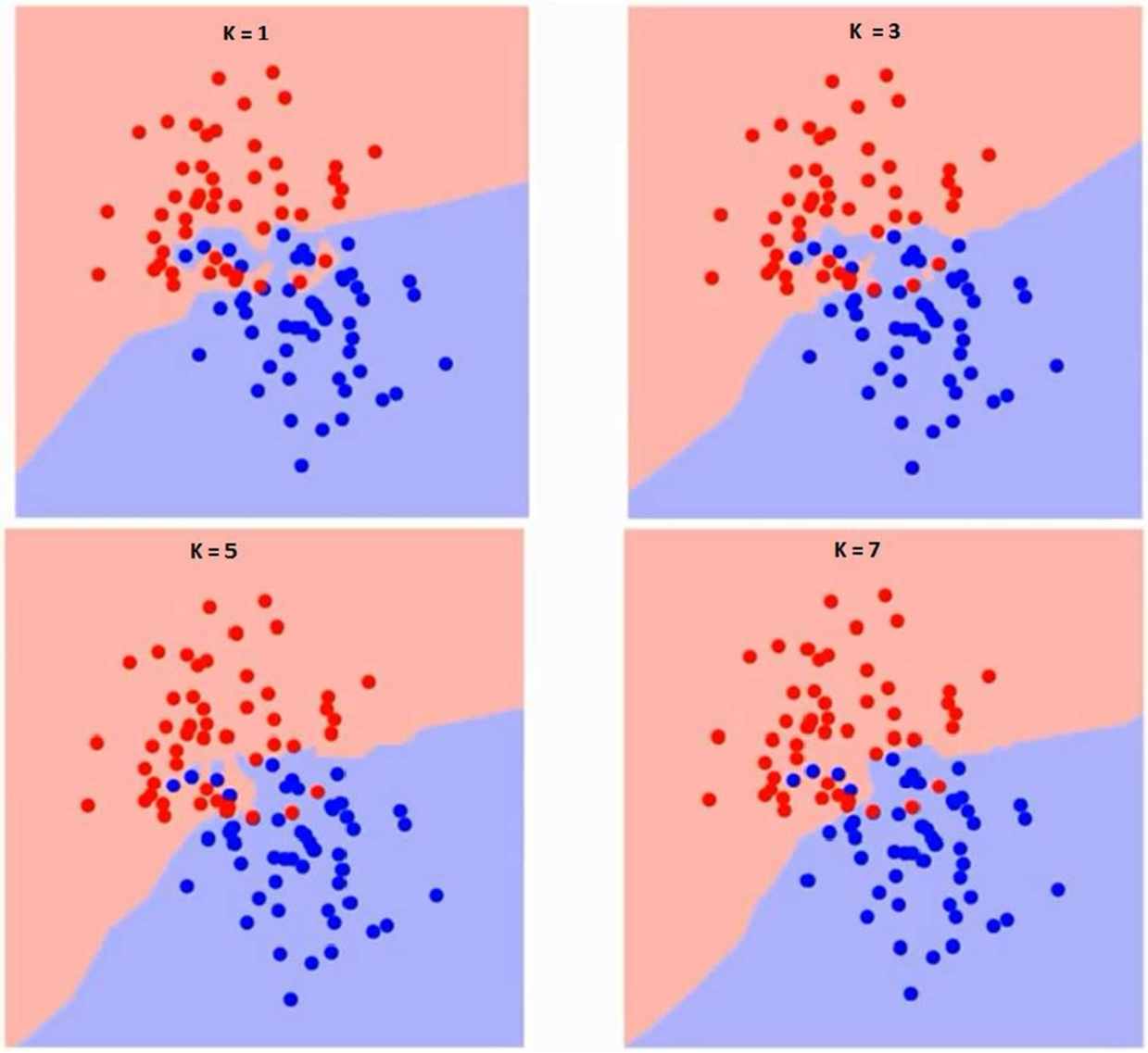


The three closest points to BS is all RC. Hence, with good confidence level we can say that the BS should belong to the class RC. Here, the choice became very obvious as all three votes from the closest neighbor went to RC. The choice of the parameter K is very crucial in this algorithm.

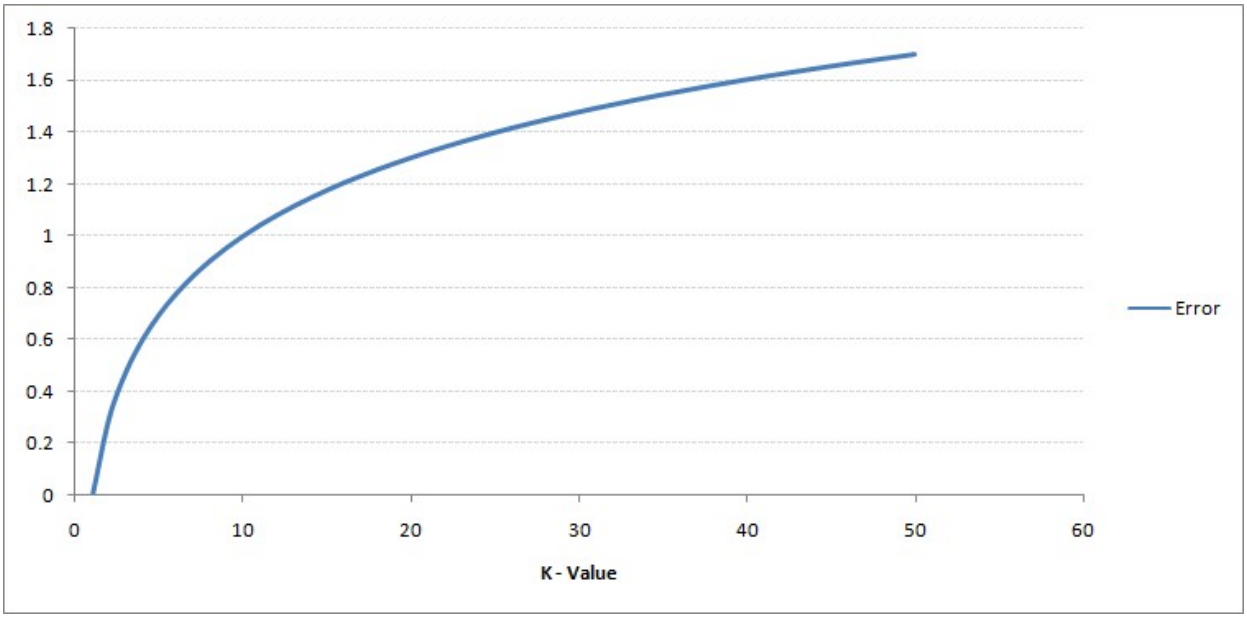
How do we choose the factor K?

First let us try to understand what exactly does K influence in the algorithm. If we see the last example, given that all the 6 training observation remain constant, with a given K value we can make boundaries of each class. These boundaries will segregate RC from GS. The same way,

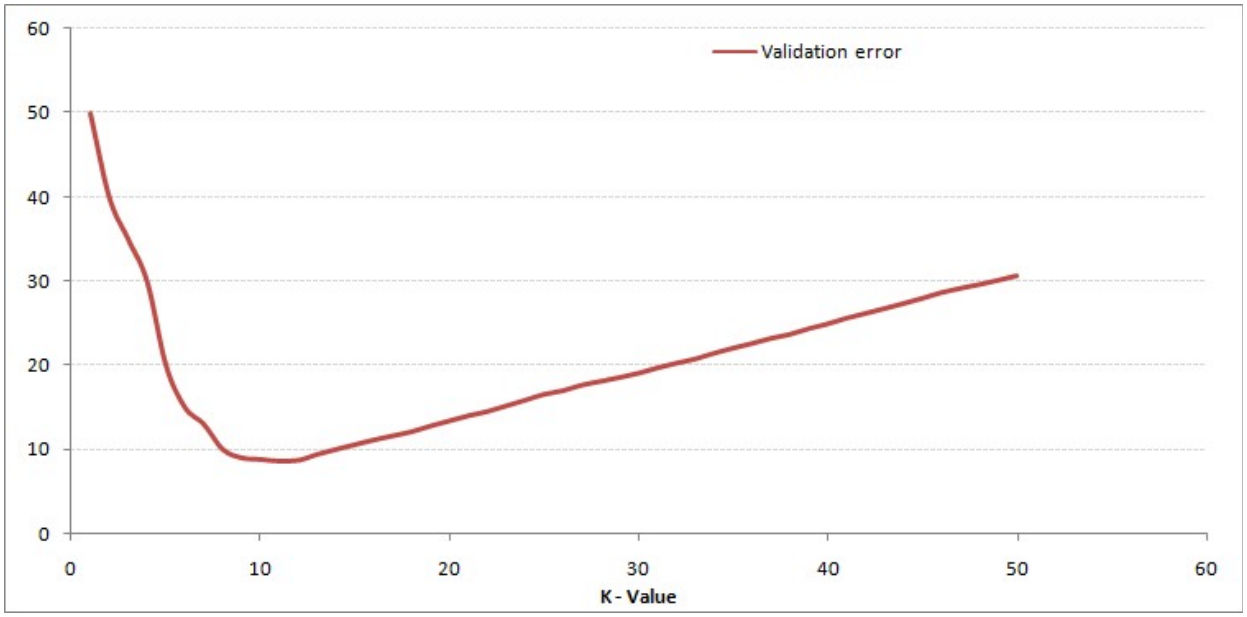
let's try to see the effect of value "K" on the class boundaries. Following are the different boundaries separating the two classes with different values of K.



If you watch carefully, you can see that the boundary becomes smoother with increasing value of K. With K increasing to infinity it finally becomes all blue or all red depending on the total majority. The training error rate and the validation error rate are two parameters we need to access on different K-value. Following is the curve for the training error rate with varying value of K:



As you can see, the error rate at K=1 is always zero for the training sample. This is because the closest point to any training data point is itself. Hence the prediction is always accurate with K=1. If validation error curve would have been similar, our choice of K would have been 1. Following is the validation error curve with varying value of K:



This makes the story more clear. At K=1, we were over fitting the boundaries. Hence, error rate initially decreases and reaches a minimal. After the minima point, it then increases with increasing K. To get the optimal value of K, you can segregate the training and validation from the initial dataset. Now plot the validation error curve to get the optimal value of K. This value of K should be used for all predictions.

Breaking it down – Pseudo Code of KNN

We can implement a KNN model by following the below steps:

Load the data
Initialize the value of k
For getting the predicted class, iterate from 1 to total number of training data points
Calculate the distance between test data and each row of training data. Here we will use Euclidean distance as our distance metric since it's the most popular method. The other metrics that can be used are Chebyshev, cosine, etc.
Sort the calculated distances in ascending order based on distance values
Get top k rows from the sorted array
Get the most frequent class of these rows
Return the predicted class

Conclusion

KNN algorithm is one of the simplest classification algorithms. Even with such simplicity, it can give highly competitive results. KNN algorithm can also be used for regression problems. The only difference from the discussed methodology will be using averages of nearest neighbors rather than voting from nearest neighbors.

Decision tree introduction

in a decision tree, the algorithm starts with a root node of a tree then compares the value of different attributes and follows the next branch until it reaches the end leaf node. It uses different algorithms to check about the split and variable that allow the best homogeneous sets of population.

ecision trees are considered to be widely used in data science. It is a key proven tool for making decisions in complex scenarios. In Machine learning, ensemble methods like decision tree, random forest are widely used. Decision trees are a type of supervised learning algorithm where data will continuously be divided into different categories according to certain parameters.

So in this blog, I will explain the Decision tree algorithm. How is it used? How it functions will be covering everything that is related to the decision tree.

What is a Decision Tree?

Decision tree as the name suggests it is a flow like a tree structure that works on the principle of conditions. It is efficient and has strong algorithms used for predictive analysis. It has mainly attributes that include internal nodes, branches and a terminal node.

Every internal node holds a “test” on an attribute, branches hold the conclusion of the test and every leaf node means the class label. This is the most used algorithm when it comes to supervised learning techniques.

It is used for both classifications as well as regression. It is often termed as “CART” that means Classification and Regression Tree. Tree algorithms are always preferred due to stability and reliability.

How can an algorithm be used to represent a tree

Let us see an example of a basic decision tree where it is to be decided in what conditions to play cricket and in what conditions not to play. You might have got a fair idea about the conditions on which decision trees work with the above example. Let us now see the common terms used in Decision Tree that is stated below:

Branches - Division of the whole tree is called branches.

Root Node - Represent the whole sample that is further divided.
Splitting - Division of nodes is called splitting.
Terminal Node - Node that does not split further is called a terminal node.
Decision Node - It is a node that also gets further divided into different sub-nodes being a sub node.
Pruning - Removal of subnodes from a decision node.
Parent and Child Node - When a node gets divided further then that node is termed as parent node whereas the divided nodes or the sub-nodes are termed as a child node of the parent node.

How Does Decision Tree Algorithm Work

It works on both the type of input & output that is categorical and continuous. In classification problems, the decision tree asks questions, and based on their answers (yes/no) it splits data into further sub branches.

It can also be used as a binary classification problem like to predict whether a bank customer will churn or not, whether an individual who has requested a loan from the bank will default or not and can even work for multiclass classifications problems. But how does it do these tasks?

In a decision tree, the algorithm starts with a root node of a tree then compares the value of different attributes and follows the next branch until it reaches the end leaf node. It uses different algorithms to check about the split and variable that allow the best homogeneous sets of population.

> "Decision trees create a tree-like structure by computing the relationship between independent features and a target. This is done by making use of functions that are based on comparison operators on the independent features

Types of Decision Tree

Type of decision tree depends upon the type of input we have that is categorical or numerical :

If the input is a categorical variable like whether the loan contender will defaulter or not, that is either yes/no. This type of decision tree is called a Categorical variable decision tree, also called classification trees.
If the input is numeric types and or is continuous in nature like when we have to predict a house price. Then the used decision tree is called a Continuous variable decision tree, also called Regression trees.

Lists of Algorithms

ID3 (Iterative Dicotomizer3) – This DT algorithm was developed by Ross Quinlan that uses greedy algorithms to generate multiple branch trees. Trees extend to maximum size before pruning.
C4.5 flourished ID3 by overcoming restrictions of features that are required to be categorical. It effectively defines distinct attributes for numerical features. Using if-then condition it converts the trained trees.
C5.0 uses less space and creates smaller rulesets than C4.5.
The CART classification and regression tree are similar to C4.5 but it braces numerical target variables and does not calculate the rule sets. It generates a binary tree.

Why do we use Decision Trees?

Decision trees provide an effective method of Decision Making because they: **Clearly lay out the problem so that all options can be challenged**. Allow us to analyze fully the possible

consequences of a decision. Provide a framework to quantify the values of outcomes and the probabilities of achieving them.

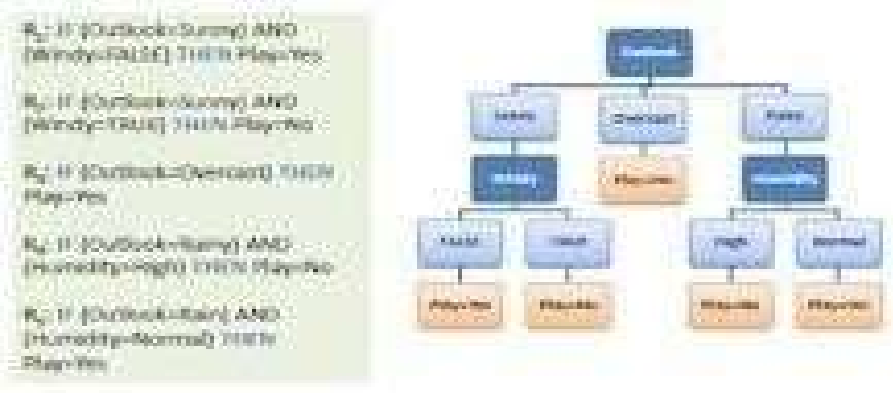
What is decision tree in interview explain?

A Decision Tree is a supervised machine learning algorithm that can be used for both Regression and Classification problem statements. It divides the complete dataset into smaller subsets while at the same time an associated Decision Tree is incrementally developed.

Where are Decision Trees used?

Decision trees are commonly used in **operations research**, specifically in decision analysis, to help identify a strategy most likely to reach a goal, but are also a popular tool in machine learning.

What is a decision tree classification model?



Decision Tree - Classification. Decision tree **builds classification or regression models in the form of a tree structure**. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. ... Decision trees can handle both categorical and numerical data

What is the final objective of decision tree?

As the goal of a decision tree is that **it makes the optimal choice at the end of each node it needs an algorithm that is capable of doing just that**. That algorithm is known as Hunt's algorithm, which is both greedy, and recursive

Decision tree introduction

in a decision tree, the algorithm starts with a root node of a tree then compares the value of different attributes and follows the next branch until it reaches the end leaf node. It uses different algorithms to check about the split and variable that allow the best homogeneous sets of population.

ecision trees are considered to be widely used in data science. It is a key proven tool for making decisions in complex scenarios. In Machine learning, ensemble methods like decision tree, random forest are widely used. Decision trees are a type of supervised learning algorithm where data will continuously be divided into different categories according to certain parameters.

So in this blog, I will explain the Decision tree algorithm. How is it used? How it functions will be covering everything that is related to the decision tree.

What is a Decision Tree?

Decision tree as the name suggests it is a flow like a tree structure that works on the principle of conditions. It is efficient and has strong algorithms used for predictive analysis. It has mainly attributes that include internal nodes, branches and a terminal node.

Every internal node holds a "test" on an attribute, branches hold the conclusion of the test and every leaf node means the class label. This is the most used algorithm when it comes to supervised learning techniques.

It is used for both classifications as well as regression. It is often termed as "CART" that means Classification and Regression Tree. Tree algorithms are always preferred due to stability and reliability.

How can an algorithm be used to represent a tree

Let us see an example of a basic decision tree where it is to be decided in what conditions to play cricket and in what conditions not to play. You might have got a fair idea about the conditions on which decision trees work with the above example. Let us now see the common terms used in Decision Tree that is stated below:

Branches - Division of the whole tree is called branches.
Root Node - Represent the whole sample that is further divided.
Splitting - Division of nodes is called splitting.
Terminal Node - Node that does not split further is called a terminal node.
Decision Node - It is a node that also gets further divided into different sub-nodes being a sub node.

Pruning - Removal of subnodes from a decision node.

Parent and Child Node - When a node gets divided further then that node is termed as parent node whereas the divided nodes or the sub-nodes are termed as a child node of the parent node.

How Does Decision Tree Algorithm Work

It works on both the type of input & output that is categorical and continuous. In classification problems, the decision tree asks questions, and based on their answers (yes/no) it splits data into further sub branches.

It can also be used as a binary classification problem like to predict whether a bank customer will churn or not, whether an individual who has requested a loan from the bank will default or not and can even work for multiclass classifications problems. But how does it do these tasks?

In a decision tree, the algorithm starts with a root node of a tree then compares the value of different attributes and follows the next branch until it reaches the end leaf node. It uses different algorithms to check about the split and variable that allow the best homogeneous sets of population.

> **"Decision trees create a tree-like structure by computing the relationship between independent features and a target. This is done by making use of functions that are based on comparison operators on the independent features**

Types of Decision Tree

Type of decision tree depends upon the type of input we have that is categorical or numerical :

If the input is a categorical variable like whether the loan contender will defaulter or not, that is either yes/no. This type of decision tree is called a Categorical variable decision tree, also called classification trees.

If the input is numeric types and or is continuous in nature like when we have to predict a house price. Then the used decision tree is called a Continuous variable decision tree, also called Regression trees.

Lists of Algorithms

ID3 (Iterative Dicotomizer3) – This DT algorithm was developed by Ross Quinlan that uses greedy algorithms to generate multiple branch trees. Trees extend to maximum size before pruning.

C4.5 flourished ID3 by overcoming restrictions of features that are required to be categorical. It effectively defines distinct attributes for numerical features. Using if-then condition it converts the trained trees.

C5.0 uses less space and creates smaller rulesets than C4.5.

The CART classification and regression tree are similar to C4.5 but it braces numerical target variables and does not calculate the rule sets. It generates a binary tree.

Why do we use Decision Trees?

Decision trees provide an effective method of Decision Making because they: **Clearly lay out the problem so that all options can be challenged**. Allow us to analyze fully the possible consequences of a decision. Provide a framework to quantify the values of outcomes and the probabilities of achieving them.

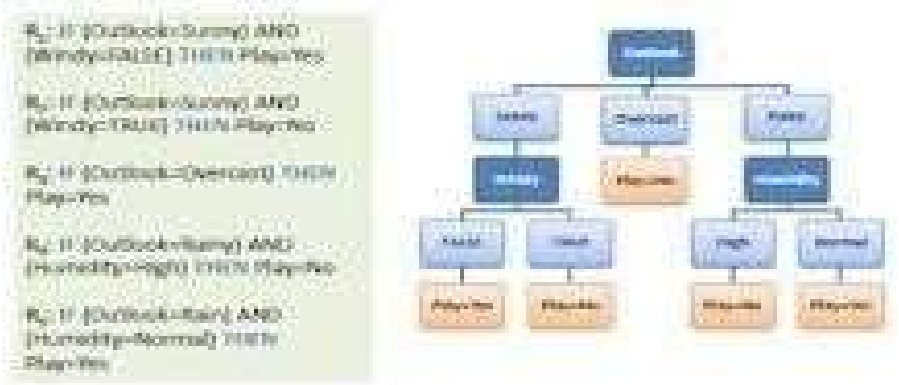
What is decision tree in interview explain?

A Decision Tree is a supervised machine learning algorithm that can be used for both Regression and Classification problem statements. It divides the complete dataset into smaller subsets while at the same time an associated Decision Tree is incrementally developed.

Where are Decision Trees used?

Decision trees are commonly used in **operations research**, specifically in decision analysis, to help identify a strategy most likely to reach a goal, but are also a popular tool in machine learning.

What is a decision tree classification model?



Decision Tree - Classification. Decision tree **builds classification or regression models in the form of a tree structure**. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. ... Decision trees can handle both categorical and numerical data

What is the final objective of decision tree?

As the goal of a decision tree is that **it makes the optimal choice at the end of each node it needs an algorithm that is capable of doing just that**. That algorithm is known as Hunt's algorithm, which is both greedy, and recursive

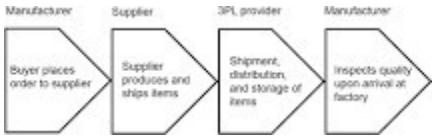
Introduction to Logistics.

LOGISTICS IS THE ART AND SCIENCE OF MANAGEMENT, ENGINEERING AND TECHNICAL ACTIVITIES CONCERNED WITH REQUIREMENTS, DESIGN AND SUPPLYING, MAINTAINING RESOURCES TO SUPPORT OBJECTIVES, PLANS AND OPERATION.

How do you explain logistics?

Logistics refers to the **overall process of managing how resources are acquired, stored, and transported to their final destination**. Logistics management involves identifying prospective distributors and suppliers and determining their effectiveness and accessibility.

What are the 3 types of logistics?



Logistics has three types; **inbound, outbound, and reverse logistics**.

What are the 7 R's of logistics?

So, what are the 7 Rs? The Chartered Institute of Logistics & Transport UK (2019) defines them as: **Getting the Right product, in the Right quantity, in the Right condition, at the Right place, at the Right time, to the Right customer, at the Right price**.

What are the importance of logistics?

Logistics is an important element of a successful supply chain that **helps increase the sales and profits of businesses that deal with the production, shipment, warehousing and delivery of products**. Moreover, a reliable logistics service can boost a business' value and help in maintaining a positive public image.

What is logistics in real life?

Logistics is **the strategic vision of how you will create and deliver your product or service to your end customer**. If you take the city, town or village that you live in, you can see a very clear example of what the logistical strategy was when they were designing it.

What are the 3 main activities of logistics systems?

Logistics activities or Functions of Logistics

Order processing. The Logistics activities start from the order processing which might be the work of the commercial department in an organization. ...

Materials handling. ...

Warehousing. ...

Inventory control. ...

Transportation. ...

Packaging.

What is 3PL and 4PL in logistics?

A 3PL (third-party logistics) provider manages all aspects of fulfillment, from warehousing to shipping. A 4PL (fourth-party logistics) provider manages a 3PL on behalf of the customer and other aspects of the supply chain.

What are the five major components of logistics?

There are five elements of logistics:

Storage, warehousing and materials handling.

Packaging and unitisation.

Inventory.

Transport.

Information and control.

What is logistic cycle?

Logistics management cycle includes key activities such as **product selection, quantification and procurement, inventory management, storage, and distribution**. Other activities that

help drive the logistics cycle and are also at the heart of logistics are organisation and staffing, budget, supervision, and evaluation.

Why did you choose logistics?

We choose logistics because **it is one of the most important career sectors in the globe and be more excited about it** I prefer my profession to work in logistics and it can be a challenging field, and with working in it I want to make up an important level of satisfaction in their jobs.

What is logistics and SCM?

The basic difference between Logistics and Supply Chain Management is that Logistics management is the process of integration and maintenance (flow and storage) of goods in an organization whereas Supply Chain Management is the coordination and management (movement) of supply chains of an organization

Here are 6 steps logistics companies should follow to develop a sound logistics marketing plan.

Define your service offer. ...

Determine your primary and secondary markets. ...

Identify your competition. ...

Articulate your value proposition. ...

Allocate a marketing budget. ...

Develop a tactical marketing plan

CHAPTER 4

Design and Implementation Constraints

Constraints in Analysis

Constraints as Informal Text

Constraints as Operational Restrictions

Constraints Integrated in Existing Model Concepts

Constraints as a Separate Concept

Constraints Implied by the Model Structure

Constraints in Design

Determination of the Involved Classes

Determination of the Involved Objects

Determination of the Involved Actions

Determination of the Require Clauses

Global actions and Constraint Realization

Constraints in Implementation

A hierarchical structuring of relations may result in more classes and a more complicated structure to implement. Therefore it is advisable to transform the hierarchical relation structure to a simpler structure such as a classical flat one. It is rather straightforward to transform the developed hierarchical model into a bipartite, flat model, consisting of classes on the one hand and flat relations on the other. Flat relations are preferred at the design level for reasons of simplicity and implementation ease. There is no identity or functionality associated with a flat relation. A flat relation corresponds with the relation concept of entity-relationship modeling and many object oriented methods.

Other Nonfunctional Requirements

Performance Requirements

The application at this side controls and communicates with the following three main general components.

embedded browser in charge of the navigation and accessing to the web service;

Server Tier: The server side contains the main parts of the functionality of the proposed architecture. The components at this tier are the following.

Web Server, Security Module, Server-Side Capturing Engine, Preprocessing Engine, Database System, Verification Engine, Output Module.

Safety Requirements

The software may be safety-critical. If so, there are issues associated with its integrity level

The software may not be safety-critical although it forms part of a safety-critical system. For example, software may simply log transactions.

If a system must be of a high integrity level and if the software is shown to be of that integrity level, then the hardware must be at least of the same integrity level.

There is little point in producing 'perfect' code in some language if hardware and system software (in widest sense) are not reliable.

If a computer system is to run software of a high integrity level then that system should not at the same time accommodate software of a lower integrity level.

Systems with different requirements for safety levels must be separate

Otherwise, the highest level of integrity required must be applied to all systems in the same environment.

CHAPTER 5

Architecture Diagram:

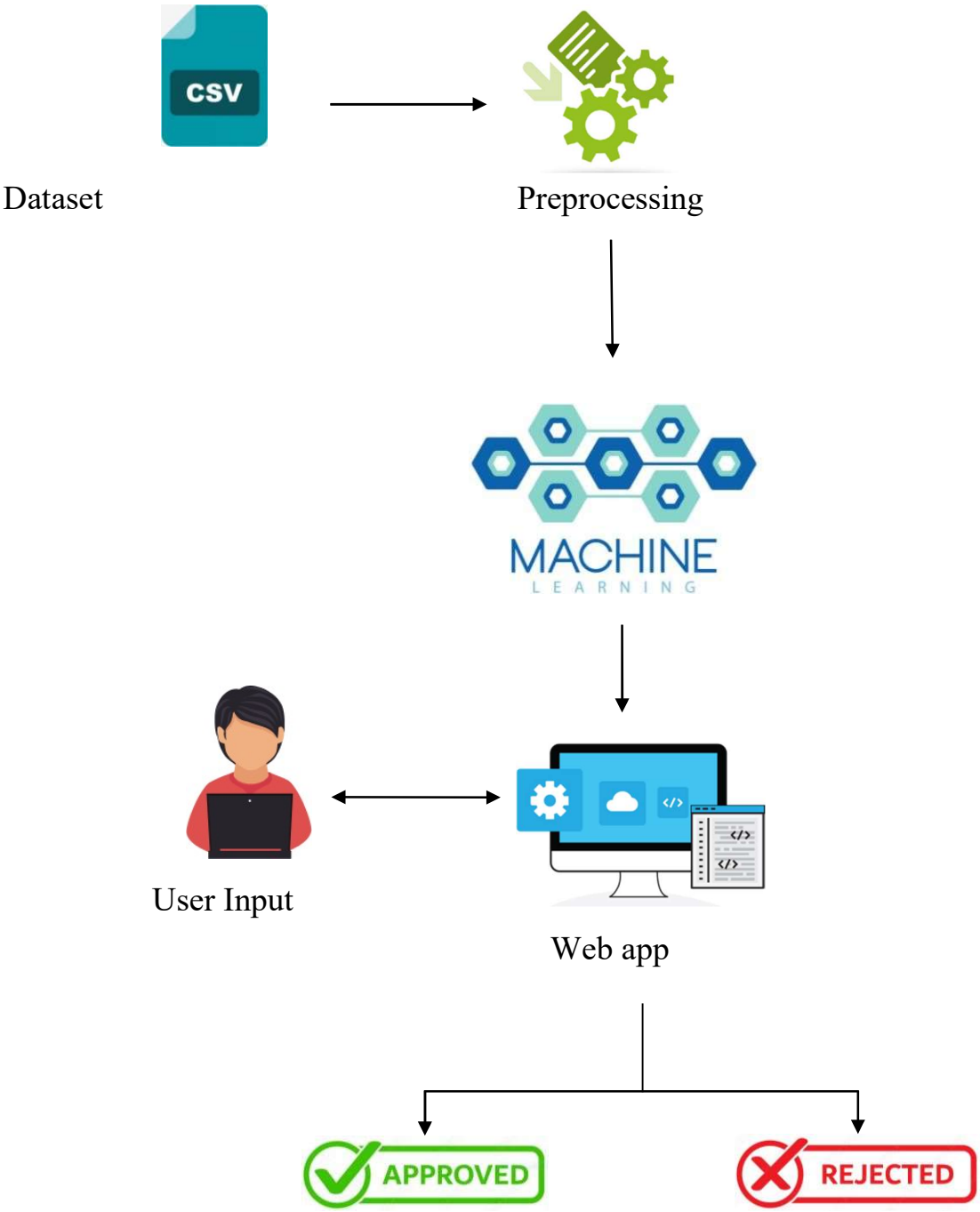
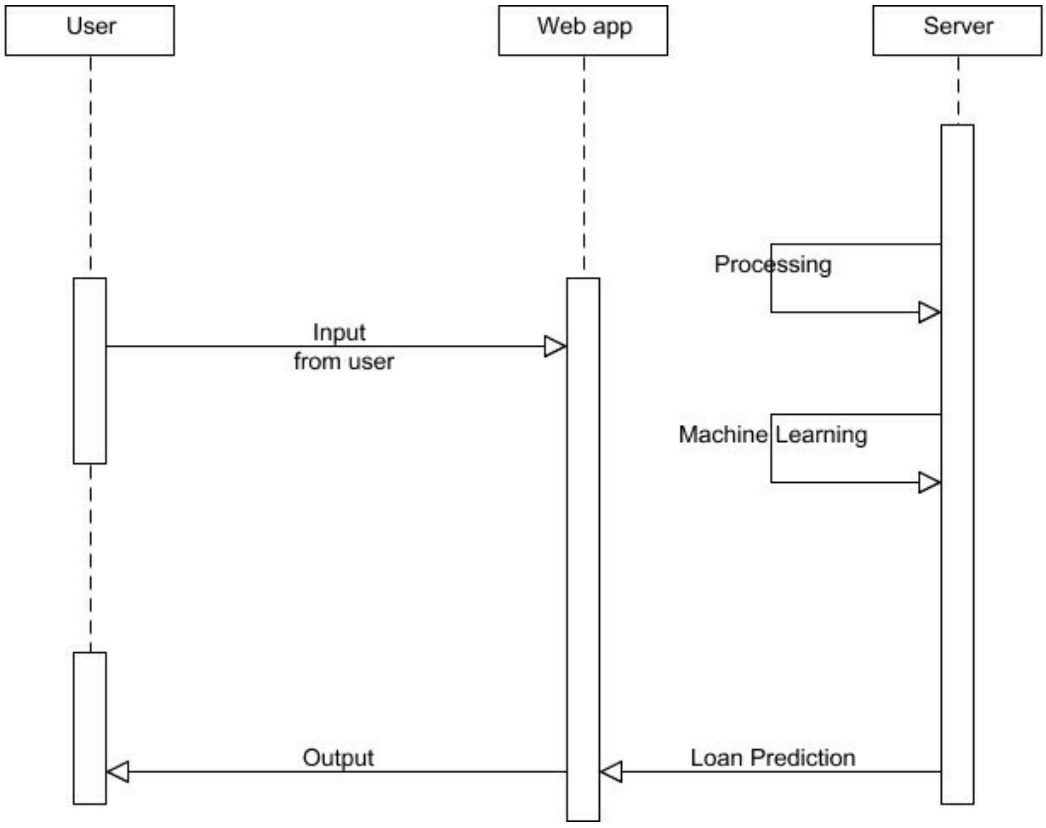


Fig: 5.1

Sequence Diagram:

A Sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of Message Sequence diagrams are sometimes called event diagrams, event sceneries and timing diagram.



Use Case Diagram:

Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of software engineering. The standard is managed and was created by the Object Management Group. UML includes a set of graphic notation techniques to create visual models of software intensive systems. This language is used to specify, visualize, modify, construct and document the artifacts of an object oriented software intensive system under development.

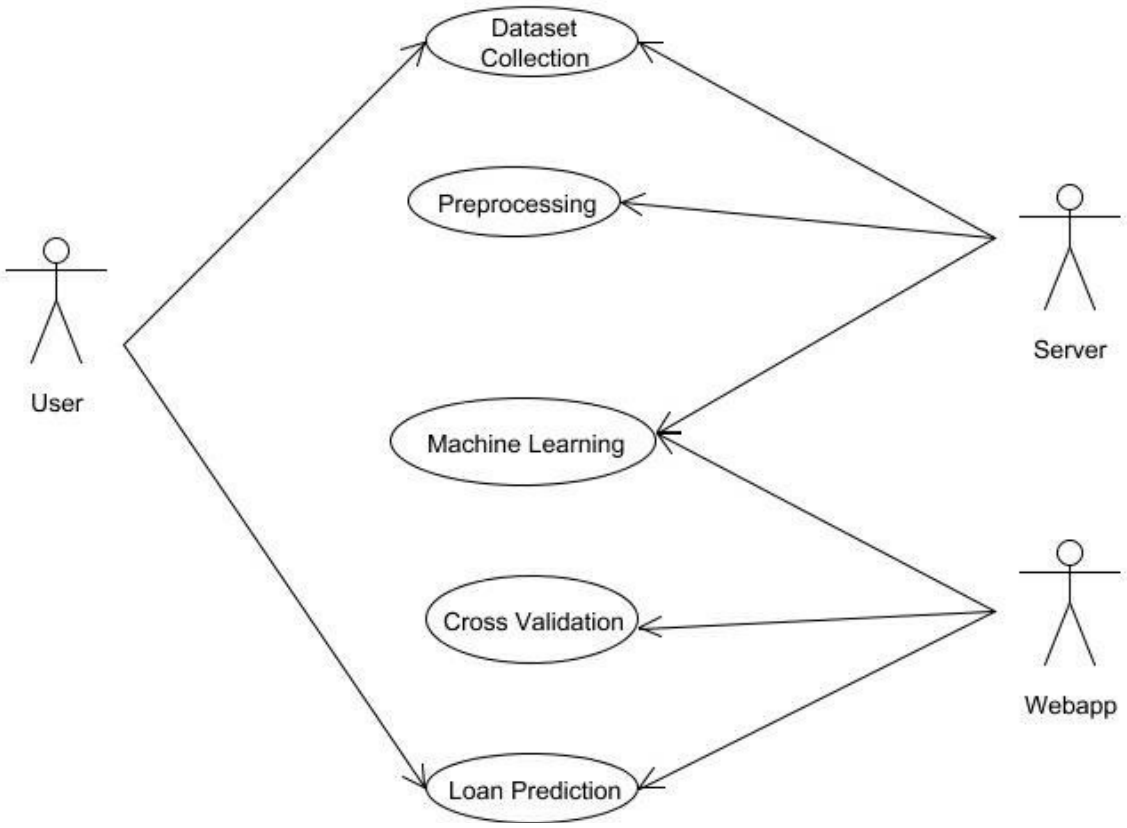
USECASE DIAGRAM

A Use case Diagram is used to present a graphical overview of the functionality provided by a system in terms of actors, their goals and any dependencies between those use cases.

Use case diagram consists of two parts:

Use case: A use case describes a sequence of actions that provided something of measurable value to an actor and is drawn as a horizontal ellipse.

Actor: An actor is a person, organization or external system that plays a role in one or more interaction with the system.

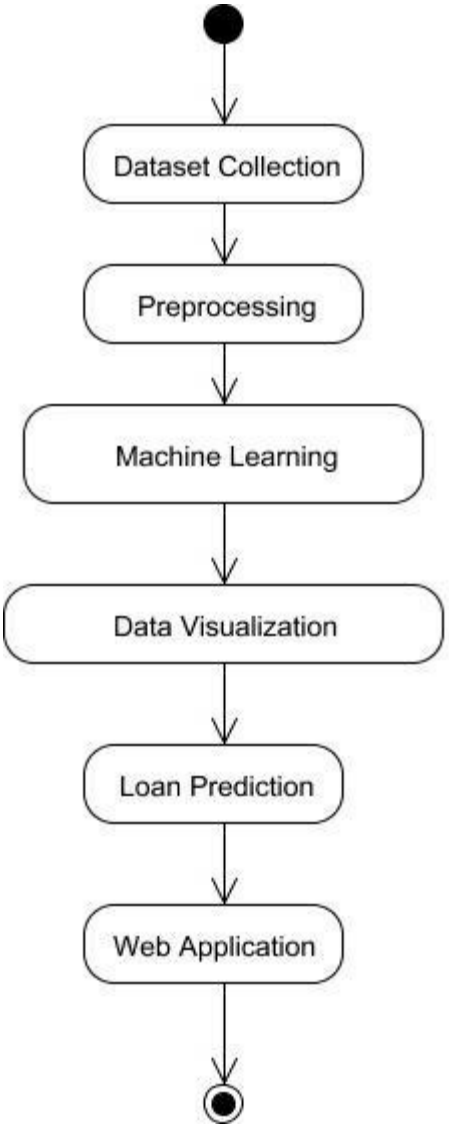

Dataset Collection
Preprocessing
Machine Learning
Cross Validation
Loan Prediction
User
Server
Webapp

Activity Diagram:

Activity diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control.

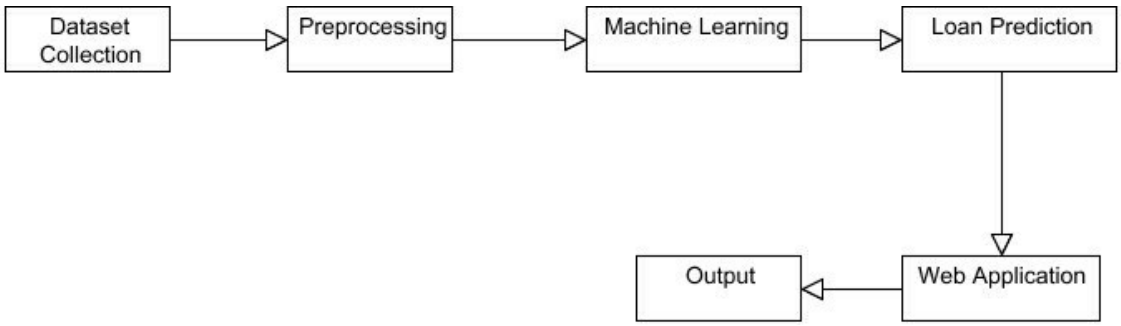
The most important shape types:

Rounded rectangles represent activities.
Diamonds represent decisions.
Bars represent the start or end of concurrent activities.
A black circle represents the start of the workflow.
An encircled circle represents the end of the workflow.


Dataset Collection
Preprocessing
Machine Learning
Data Visualization
Loan Prediction
Web Application

Collaboration Diagram:

UML Collaboration Diagrams illustrate the relationship and interaction between software objects. They require use cases, system operation contracts and domain model to already exist. The collaboration diagram illustrates messages being sent between classes and objects.



CHAPTER 6

MODULES

Dataset collection

Machine Learning Algorithm

Prediction

MODULE EXPLANATION:

6.2.1 Dataset collection:

Dataset is collected from the kaggle.com. That dataset have some value like gender, marital status, self-employed or not, monthly income, etc,. Dataset has the information, whether the previous loan is approved or not depends up on the customer information. That data well be preprocessed and proceed to the next step.

Machine learning Algorithm:

In this stage, the collected data will be given to the machine algorithm for training process. We use multiple algorithms to get high accuracy range of prediction. A preprocessed dataset are processed in different machine learning algorithms. Each algorithm gives some accuracy level. Each one is undergoes for the comparison.

- ✔ **Logistic Regression**
- ✔ **K-Nearest Neighbors**
- ✔ **Decision Tree Classifier**

Prediction:

Preprocessed data are trained and input given by the user goes to the trained dataset. The Logistic Regression trained model is used to predict and determine whether the loan given to a particular person shall be approved or not.

CHAPTER 7

CODING AND TESTING

CODING

Once the design aspect of the system is finalizes the system enters into the coding and testing phase. The coding phase brings the actual system into action by converting the design of the system into the code in a given programming language. Therefore, a good coding style has to be taken whenever changes are required it easily screwed into the system.

CODING STANDARDS

Coding standards are guidelines to programming that focuses on the physical structure and appearance of the program. They make the code easier to read, understand and maintain. This phase of the system actually implements the blueprint developed during the design phase. The coding specification should be in such a way that any programmer must be able to understand the code and can bring about changes whenever felt necessary. Some of the standard needed to achieve the above-mentioned objectives are as follows:

Program should be simple, clear and easy to understand. Naming

conventions

Value conventions

Script and comment procedure

Message box format Exception and

error handling

NAMING CONVENTIONS

Naming conventions of classes, data member, member functions, procedures etc., should be **self-descriptive**. One should even get the meaning and scope of the variable by its name. The conventions are adopted for **easy understanding** of the intended message by the user. So it is customary to follow the conventions. These conventions are as follows:

Class names

Class names are problem domain equivalence and begin with capital letter and have mixed cases.

Member Function and Data Member name

Member function and data member name begins with a lowercase letter with each subsequent letters of the new words in uppercase and the rest of letters in lowercase.

VALUE CONVENTIONS

Value conventions ensure values for variable at any point of time. This involves the following:

Proper default values for the variables.

Proper validation of values in the field.

Proper documentation of flag values.

SCRIPT WRITING AND COMMENTING STANDARD

Script writing is an art in which indentation is utmost important. Conditional and looping statements are to be properly aligned to facilitate easy understanding. Comments are included to minimize the number of surprises that could occur when going through the code.

MESSAGE BOX FORMAT

When something has to be prompted to the user, he must be able to understand it properly. To achieve this, a specific format has been adopted in displaying messages to the user. They are as follows:

X – User has performed illegal operation.

! – Information to the user.

TEST PROCEDURE
SYSTEM TESTING

Testing is performed to identify errors. It is used for quality assurance. Testing is an integral part of the entire development and maintenance process. The goal of the testing during phase is to verify that the specification has been accurately and completely incorporated into the design, as well as to ensure the correctness of the design itself. For example the design must not have any logic faults in the design is detected before coding commences, otherwise the cost of fixing the faults will be considerably higher as reflected. Detection of design faults can be achieved by means of inspection as well as walkthrough.

Testing is one of the important steps in the software development phase. Testing checks for the errors, as a whole of the project testing involves the following test cases:

Static analysis is used to investigate the structural properties of the Source code.

Dynamic testing is used to investigate the behavior of the source code by executing the program on the test data.

TEST DATA AND OUTPUT

UNIT TESTING

Unit testing is conducted to verify the functional performance of each modular component of the software. Unit testing focuses on the smallest unit of the software design (i.e.), the module. The white-box testing techniques were heavily employed for unit testing.

FUNCTIONAL TESTS

Functional test cases involved exercising the code with nominal input values for which the expected results are known, as well as boundary values and special values, such as logically related inputs, files of identical elements, and empty files.

Three types of tests in Functional test:

Performance Test

Stress Test

Structure Test

PERFORMANCE TEST

It determines the amount of execution time spent in various parts of the unit, program throughput, and response time and device utilization by the program unit.

STRESS TEST

Stress Test is those test designed to intentionally break the unit. A Great deal can be learned about the strength and limitations of a program by examining the manner in which a programmer in which a program unit breaks.

STRUCTURED TEST

Structure Tests are concerned with exercising the internal logic of a program and traversing particular execution paths. The way in which White-Box test strategy was employed to ensure that the test cases could Guarantee that all independent paths within a module have been have been exercised at least once.

Exercise all logical decisions on their true or false sides.

Execute all loops at their boundaries and within their operational bounds.

Exercise internal data structures to assure their validity.

Checking attributes for their correctness.

Handling end of file condition, I/O errors, buffer problems and textual errors in output information

INTEGRATION TESTING

Integration testing is a systematic technique for construction the program structure while at the same time conducting tests to uncover errors associated with interfacing. i.e., integration testing is the complete testing of the set of modules which makes up the product. The objective is to take untested modules and build a program structure tester should identify critical modules. Critical modules should be tested as early as possible. One approach is to wait until all

the units have passed testing, and then combine them and then tested. This approach is evolved from unstructured testing of small programs. Another strategy is to construct the product in increments of tested units. A small set of modules are integrated together and tested, to which another module is added and tested in combination. And so on. The advantages of this approach are that, interface dispenses can be easily found and corrected.

The major error that was faced during the project is linking error. When all the modules are combined the link is not set properly with all support files. Then we checked out for interconnection and the links. Errors are localized to the new module and its intercommunications. The product development can be staged, and modules integrated in as they complete unit testing. Testing is completed when the last module is integrated and tested.

TESTING TECHNIQUES / TESTING STRATERGIES

TESTING

Testing is a process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an as-yet –undiscovered error. A successful test is one that uncovers an as-yet- undiscovered error. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently as expected before live operation commences. It verifies that the whole set of programs hang together. System testing requires a test consists of several key activities and steps for run program, string, system and is important in adopting a successful new system. This is the last chance to detect and correct errors before the system is installed for user acceptance testing.

The software testing process commences once the program is created and the documentation and related data structures are designed. Software testing is essential for

correcting errors. Otherwise the program or the project is not said to be complete. Software testing is the critical element of software quality assurance and represents the ultimate the review of specification design and coding. Testing is the process of executing the program with the intent of finding the error. A good test case design is one that as a probability of finding an yet undiscovered error. A successful test is one that uncovers an yet undiscovered error. Any engineering product can be tested in one of the two ways:

WHITE BOX TESTING

This testing is also called as Glass box testing. In this testing, by knowing the specific functions that a product has been design to perform test can be conducted that demonstrate each function is fully operational at the same time searching for errors in each function. It is a test case design method that uses the control structure of the procedural design to derive test cases. Basis path testing is a white box testing.

Basis path testing:

Flow graph notation

Cyclometric complexity

Deriving test cases

Graph matrices Control

BLACK BOX TESTING

In this testing by knowing the internal operation of a product, test can be conducted to ensure that “all gears mesh”, that is the internal operation performs according to

specification and all internal components have been adequately exercised. It fundamentally focuses on the functional requirements of the software.

The steps involved in black box test case design are:

Graph based testing methods

Equivalence partitioning

Boundary value analysis

Comparison testing

SOFTWARE TESTING STRATEGIES:

A software testing strategy provides a road map for the software developer. Testing is a set activity that can be planned in advance and conducted systematically. For this reason a template for software testing a set of steps into which we can place specific test case design methods should be strategy should have the following characteristics:

Testing begins at the module level and works "outward" toward the integration of the entire computer based system.

Different testing techniques are appropriate at different points in time.

The developer of the software and an independent test group conducts testing.

Testing and Debugging are different activities but debugging must be accommodated in any testing strategy.

INTEGRATION TESTING:

Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with. Individual modules, which are highly prone to interface errors, should not be assumed to work instantly when we put them together. The problem of course, is "putting them together"- interfacing. There may be the chances of data lost across on another's sub functions, when combined may not produce the desired

major function; individually acceptable impression may be magnified to unacceptable levels; global data structures can present problems.

PROGRAM TESTING:

The logical and syntax errors have been pointed out by program testing. A syntax error is an error in a program statement that in violates one or more rules of the language in which it is written. An improperly defined field dimension or omitted keywords are common syntax error. These errors are shown through error messages generated by the computer. A logic error on the other hand deals with the incorrect data fields, out-off-range items and invalid combinations. Since the compiler s will not deduct logical error, the programmer must examine the output. Condition testing exercises the logical conditions contained in a module. The possible types of elements in a condition include a Boolean operator, Boolean variable, a pair of Boolean parentheses A relational operator or on arithmetic expression. Condition testing method focuses on testing each condition in the program the purpose of condition test is to deduct not only errors in the condition of a program but also other a errors in the program.

SECURITY TESTING:

Security testing attempts to verify the protection mechanisms built in to a system well, in fact, protect it from improper penetration. The system security must be tested for invulnerability from frontal attack must also be tested for invulnerability from rear attack. During security, the tester places the role of individual who desires to penetrate system.

VALIDATION TESTING

At the culmination of integration testing, software is completely assembled as a package. Interfacing errors have been uncovered and corrected and a final series of software test- validation testing begins. Validation testing can be defined in many ways, but a simple definition is that validation succeeds when the software functions in manner that is reasonably expected by the customer. Software validation is achieved through a series of black box tests that demonstrate conformity with requirement. After validation test has been conducted, one of two conditions exists.

The function or performance characteristics confirm to specifications and are accepted.
A validation from specification is uncovered and a deficiency created.

Deviation or errors discovered at this step in this project is corrected prior to completion of the project with the help of the user by negotiating to establish a method for resolving deficiencies. Thus the proposed system under consideration has been tested by using validation testing and found to be working satisfactorily. Though there were deficiencies in the system they were not catastrophic

USER ACCEPTANCE TESTING

User acceptance of the system is key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system and user at the time of developing and making changes whenever required. This is done in regarding to the following points.

CHAPTER 8

IMPLIMENTATION AND RESULT

In [1]:

```
1  import pandas as pd
```

In [2]:

```
1  df = pd.read_csv('LoanApprovalPrediction.csv')
```

In [3]:

```
1  df.head()
```

Out[3]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credi
0	LP001002	Male	No	0.0	Graduate	No	5849	0.0	NaN	360.0	
1	LP001003	Male	Yes	1.0	Graduate	No	4583	1508.0	128.0	360.0	
2	LP001005	Male	Yes	0.0	Graduate	Yes	3000	0.0	66.0	360.0	
3	LP001006	Male	Yes	0.0	Not Graduate	No	2583	2358.0	120.0	360.0	
4	LP001008	Male	No	0.0	Graduate	No	6000	0.0	141.0	360.0	

In [4]:

```
1  df.shape
```

Out[4]:

```
(598, 13)
```

In [5]:

```
1  df.info
```

Out[5]:

```
<bound method DataFrame.info of          Loan_ID Gender Married Dependents     Education Self_Employed \
0     LP001002    Male      No        0.0      Graduate            No
1     LP001003    Male     Yes        1.0      Graduate            No
2     LP001005    Male     Yes        0.0      Graduate           Yes
3     LP001006    Male     Yes        0.0  Not Graduate            No
4     LP001008    Male      No        0.0      Graduate            No
..         ...     ...     ...        ...           ...           ...
593   LP002978  Female      No        0.0      Graduate            No
594   LP002979    Male     Yes        3.0      Graduate            No
595   LP002983    Male     Yes        1.0      Graduate            No
596   LP002984    Male     Yes        2.0      Graduate            No
597   LP002990  Female      No        0.0      Graduate           Yes

     ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
0               5849                0.0         NaN             360.0
1               4583             1508.0       128.0             360.0
2               3000                0.0        66.0             360.0
3               2583             2358.0       120.0             360.0
4               6000                0.0       141.0             360.0
..               ...                ...         ...               ...
593             2900                0.0        71.0             360.0
594             4106                0.0        40.0             180.0
595             8072              240.0       253.0             360.0
596             7583                0.0       187.0             360.0
597             4583                0.0       133.0             360.0

     Credit_History Property_Area Loan_Status
0               1.0         Urban           Y
1               1.0         Rural           N
2               1.0         Urban           Y
3               1.0         Urban           Y
4               1.0         Urban           Y
..              ...           ...         ...
593             1.0         Rural           Y
594             1.0         Rural           Y
```

```
595             1.0         Urban   Y
596             1.0         Urban   Y
597             0.0     Semiurban   N

[598 rows x 13 columns]>
```

In [6]:

```
df.describe()
```

Out[6]:

	Dependents	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	586.000000	598.000000	598.000000	577.000000	584.000000	549.000000
mean	0.755973	5292.252508	1631.499866	144.968804	341.917808	0.843352
std	1.007751	5807.265364	2953.315785	82.704182	65.205994	0.363800
min	0.000000	150.000000	0.000000	9.000000	12.000000	0.000000
25%	0.000000	2877.500000	0.000000	100.000000	360.000000	1.000000
50%	0.000000	3806.000000	1211.500000	127.000000	360.000000	1.000000
75%	1.750000	5746.000000	2324.000000	167.000000	360.000000	1.000000
max	3.000000	81000.000000	41667.000000	650.000000	480.000000	1.000000

In [7]:

```
df.isna().sum()
```

Out[7]:

```
Loan_ID              0
Gender               0
Married              0
Dependents          12
Education            0
Self_Employed        0
ApplicantIncome      0
CoapplicantIncome    0
LoanAmount          21
Loan_Amount_Term    14
Credit_History      49
Property_Area        0
Loan_Status          0
dtype: int64
```

In [8]:

```
#check the uniqueness of Loan Id column
df.Loan_ID.nunique()
```

Out[8]:

```
598
```

In [9]:

```
#drop the Loan_ID column as it is not required (no dupllicates)
df.drop(['Loan_ID'], axis = 1, inplace = True)
```

In [10]:

```
df.head()
```

Out[10]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
0	Male	No	0.0	Graduate	No	5849	0.0	NaN	360.0	1.0
1	Male	Yes	1.0	Graduate	No	4583	1508.0	128.0	360.0	1.0
2	Male	Yes	0.0	Graduate	Yes	3000	0.0	66.0	360.0	1.0
3	Male	Yes	0.0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0
4	Male	No	0.0	Graduate	No	6000	0.0	141.0	360.0	1.0

In [11]:

```
#Total missing value cells
df.isna().sum().sum()
```

Out[11]:

In [ ]:

```
l
```

Out[12]:

```
Gender                  object
Married                 object
Dependents             float64
Education               object
Self_Employed           object
ApplicantIncome              int64
CoapplicantIncome float64 LoanAmount
                       float64
Loan_Amount_Term           float64
Credit_History      float64
Property_Area        object
Loan_Status             object
dtype: object
```

In [13]:

```
#convert Gender column from object to int datatype
df.Gender = df.Gender.map({'Male' : 0, 'Female' : 1})
```

In [14]:

```
df.head()
```

Out[14]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
0	0	No	0.0	Graduate	No	5849	0.0	NaN	360.0	1.0
1	0	Yes	1.0	Graduate	No	4583	1508.0	128.0	360.0	1.0
2	0	Yes	0.0	Graduate	Yes	3000	0.0	66.0	360.0	1.0
3	0	Yes	0.0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0
4	0	No	0.0	Graduate	No	6000	0.0	141.0	360.0	1.0

```
#Convert all categorical(object) columns using LabelEncoder
from sklearn.preprocessing import LabelEncoderlabel_encoder
= LabelEncoder()
obj = (df.dtypes == 'object')
print(list(obj[obj].index))#list of categorical objects
for col in list(obj[obj].index):
df[col] = label_encoder.fit_transform(df[col])
```

In [20]:

```
['Married', 'Education', 'Self_Employed', 'Property_Area', 'Loan_Status']
```

In [21]:

```
df.head()
```

Out[21]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
0	0	0	0.0	0	0	5849	0.0	NaN	360.0	1.0
1	0	1	1.0	0	0	4583	1508.0	128.0	360.0	1.0
2	0	1	0.0	0	1	3000	0.0	66.0	360.0	1.0
3	0	1	0.0	1	0	2583	2358.0	120.0	360.0	1.0
4	0	0	0.0	0	0	6000	0.0	141.0	360.0	1.0

In [22]:

```
df.dtypes
```

Out[22]:

```
Gender                    int64
Married                   int32
Dependents              float64
Education                 int32
Self_Employed             int32
ApplicantIncome               int64
CoapplicantIncome float64 LoanAmount
                        float64
Loan_Amount_Term              float64
Credit_History      float64
Property_Area         int32
Loan_Status               int32
dtype: object
```

In [23]:

```
df.columns
```

Out[23]:

```
Index(['Gender', 'Married', 'Dependents', 'Education', 'Self_Employed', 'ApplicantIncome',
'CoapplicantIncome', 'LoanAmount',
'Loan_Amount_Term', 'Credit_History', 'Property_Area', 'Loan_Status'], dtype='object')
```

In [24]:

```
#fill in the missing rows with mean
for i in df.columns:
df[i] = df[i].fillna(df[i].mean())
```

In [25]:

```
df.isna().sum()
```

Out[25]:

```
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
```

Training the model

In [26]:

```

X= dt.drop('Loan_Status',axis=1)
y = df['Loan_Status']
```

```
X
```

Out[27]:

	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
0	0	0	0.0	0	0	5849	0.0	144.968804	360.0	1.0
1	0	1	1.0	0	0	4583	1508.0	128.000000	360.0	1.0
2	0	1	0.0	0	1	3000	0.0	66.000000	360.0	1.0
3	0	1	0.0	1	0	2583	2358.0	120.000000	360.0	1.0
4	0	0	0.0	0	0	6000	0.0	141.000000	360.0	1.0
...	...	...	...	...	...	...	...	...	...	...
593	1	0	0.0	0	0	2900	0.0	71.000000	360.0	1.0
594	0	1	3.0	0	0	4106	0.0	40.000000	180.0	1.0
595	0	1	1.0	0	0	8072	240.0	253.000000	360.0	1.0
596	0	1	2.0	0	0	7583	0.0	187.000000	360.0	1.0
597	1	0	0.0	0	1	4583	0.0	133.000000	360.0	0.0

598 rows × 11 columns

```
y
```

In [28]:

Out[28]:

```
0      1
1      0
2      1
3      1
4      1
      ..
593    1
594    1
595    1
596    1
597    0
Name: Loan_Status, Length: 598, dtype: int32
```

In [48]:

```
#Split the data into train and test
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X, y ,test_size=0.3, random_state=7)
```

Model Selection

In [49]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import RidgeClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

In [50]:

```
models = []
models.append(('Logistic Regression', LogisticRegression()))
models.append(('Ridge Classifier', RidgeClassifier()))
models.append(('Decision Tree Classifier', DecisionTreeClassifier()))
models.append(('K-Neighbors Classifier', KNeighborsClassifier()))
models.append(('Random Forest Classifier', RandomForestClassifier()))
```

In [51]:

```
def model_selection(model):
model.fit(X_train,y_train)
y_pred=model.predict(X_test)
return  accuracy_score(y_test,y_pred)*100
```

```
    for name,model in models:
        #print(name,model)
            print(f'{name}  :  {model_selection(model)}')
```

```
Logistic Regression :
81.66666666666667 Ridge
Classifier : 82.22222222222221
Decision Tree Classifier :
75.55555555555556 K-Neighbors
Classifier : 66.11111111111111
Random Forest Classifier : 80.55555555555556
```

Ridge Classifier Performs better

In [ ]:

```

```

Modal design

```
import pandas as pd

from sklearn.preprocessing import LabelEncoder

from sklearn.model_selection import train_test_split

from sklearn.linear_model import RidgeClassifier

from sklearn.metrics import accuracy_score

import pickle


#load the dataset

df = pd.read_csv('LoanApprovalPrediction.csv')


#drop Load_ID column

df.drop(['Loan_ID'], axis = 1, inplace = True)


#convert categorical to int objects

label_encoder = LabelEncoder()

obj = (df.dtypes == 'object')

#print(list(obj[obj].index))#list of categorical objects

for col in list(obj[obj].index):
```

```
    df[col] = label_encoder.fit_transform(df[col])


#fill in the missing rows
for i in df.columns:
    df[i] = df[i].fillna(df[i].mean())


#divide the model into features and target variable
X= df.drop(['Loan_Status'],axis=1)
y = df['Loan_Status']


#split into training and testing data
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test =  train_test_split(X, y ,test_size=0.3, random_state=7)


#define the model
model = RidgeClassifier()


#fit the model on the training data
model.fit(X_train,y_train)


#save the train model
with open('train_model.pkl',mode ='wb') as pkl:
    pickle.dump(model,pkl)
```

App design

```
import streamlit as st
import pickle

def main():
    bg = """<div style ='backgroung-color:blue; padding:15px'>
          <h1 style='color:white'> Loan Eligibility Prediction Streamlit App</h1>
          </div>"""
    st.markdown("<h1 style='text-align: center; color:Blue;'>Loan Eligibility Prediction </h1>",
                unsafe_allow_html=True)
    #st.markdown(bg, unsafe_allow_html=True)

    left, right = st.columns((2,2))
    gender = left.selectbox('Gender', ('Male','Female'))
    married = right.selectbox('Married', ('Yes', 'No'))
    dependent = left.selectbox('Dependents', ('None','One','Two','Three') )
    education = right.selectbox('Education', ('Graduate', 'Not Graduate'))
    self_employed = left.selectbox('Self-Employed', ('Yes', 'No'))
    applicant_income = right.number_input('Applicant Income')
    coApplicantIncome = left.number_input('Coapplicant Income')
    loanAmount = right.number_input('Loan Amount')
    loan_amount_term = left.number_input('Loan Tenor (in months)')
    creditHistory = right.number_input('Credit History', 0.0, 1.0)
    propertyArea = st.selectbox('Property Area', ('Semiurban', 'Urban', 'Rural'))
    button = st.button('Predict')

    #if button is clicked make prediction
    if button:
        result = predict(gender, married, dependent, education, self_employed, applicant_income,
                    coApplicantIncome, loanAmount, loan_amount_term, creditHistory, propertyArea)
        st.success(f"You are {result} for the loan.")

#Load the train model
with open('train_model.pkl','rb') as pkl:
    train_model = pickle.load(pkl)

def predict(gender, married, dependent, education, self_employed, applicant_income,
                    coApplicantIncome, loanAmount, loan_amount_term, creditHistory, propertyArea):

        #processing user input
        gen = 0 if gender == 'Male' else 1
        mar = 0 if married == 'Yes' else 1
        dep = float(0 if dependent == 'None' else 1 if dependent == 'One' else 2 if dependent == 'Two' else
3)
        edu = 0 if education == 'Graduate' else 1
        sem = 0 if self_employed == 'Yes' else 1
        pro = 0 if propertyArea == 'Semiurban' else 1
        loAm = loanAmount / 1000
        cap = coApplicantIncome / 1000
        #making predictions
        prediction =
```

```
train_model.predict([[gen,mar,dep,edu,sem,applicant_income,cap,loAm,loan_amount_term,creditHistor
y,pro]])
        verdict = 'Not Eligible' if prediction == 0 else 'Eligible'
        return verdict

if __name__ == '__main__':
    main()
```