JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Savitribai Phule Pune University LOAN ELIGIBILITY PREDICTION

Under the guidance **ofDr. Manisha** Bharati

(PROFESSOR AT DEPARTMENT OF DATA SCIENCE AND ARTIFICIAL INTELLIGENCE,SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE)

By

RAHUL ASHOK DHOTE -(ROLL NO:PGD22DS94)

VAISHNAVI MADHAV PAWAR (ROLNO: PGD22DS30)

Submitted in partial fulfillment of the requirement for the degree of

Post Graduate Diploma in

(Data science and Artificial Intelligence (PGDDSAI))

Chapter 1 Introduction and library

Introduction

Loan prediction is a crucial aspect of the lending industry that utilizes data analysis and machine learning techniques to forecast the likelihood of a borrower defaulting on a loan. With the increasing availability of data and advancements in technology, lenders can now make more accurate predictions and informed decisions.

The process of loan prediction involves analyzing various factors such as credit history, income, employment status, loan amount, loan purpose, and other relevant data points. By examining historical loan data and identifying patterns, machine learning algorithms can be trained to recognize and predict potential risks associated with loan applications.

These predictive models enable lenders to evaluate the creditworthiness of borrowers and assess the probability of loan repayment. By understanding the potential risks, lenders can make informed decisions regarding loan approvals, interest rates, and loan terms. This not only helps lenders manage their risk exposure but also ensures responsible lending practices.

Loan prediction models are continuously refined and improved as more data becomes available and new techniques are developed. These models can be used for various types of loans, including personal loans, home loans, auto loans, and business loans. They provide valuable insights to lenders, allowing them to streamline their lending processes, reduce default rates, and enhance overall efficiency.

Import of Library

Pandas:

Pandas is a software library written for the Python programming language for data manipulation and analysis.

Pandas is mainly used for data analysis and associated manipulation of tabular data in Dataframes.

Pandas allows importing data from various file formats such as comma-separated values, JSON, Parquet, SQL database tables or queries, and Microsoft Excel.

Pandas allows various data manipulation operations such as merging, reshaping, selecting, as well as data cleaning, and data wrangling features.

Pandas is a Python library that allows you to easily manipulate data meant to be analyzed.

It is possible to manipulate data tables with labels of variables (columns) and individuals (rows). These arrays are called DataFrames, similar to dataframes under R. These data frames can be easily read and written from a tabular file or vice versa. Graphs can be easily drawn from these DataFrames using matplotlib.

Why chooses Pandas?

Using pandas, you can:

- Retrieve data from CSV files, Excel tables, web pages, HDF5, etc.
- Group, cut, lighten, move, write data; these data can be one or two dimensions,with gaps, or temporal with or without periodicity
- As long as the data is correctly formatted, pandas can get the job done even if the quantity exceeds your machine's capacity by treating the sources piece by piece. The development of this library is part of the problem of having tools to handle large volumes of data for the purpose of their scientific or commercial exploitation.

Pandas are generally used for data science but have you wondered why? This is because pandas are used in conjunction with other libraries that are used for data science. It is built on the top of the **NumPy** library which means that a lot of structures of NumPy are used or replicated in Pandas.

The data produced by Pandas are often used as input for plotting functions of **Matplotlib**, statistical analysis in **SciPy**, and machine learning algorithms in **Scikit-learn**. Panda's program can be run from any text editor but it is recommended to use Jupyter Notebook for this as Jupyter given the ability to execute code in a particular cell rather than executing the entire file. Jupyter also provides an easy way to visualize pandas data frames and plots.

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data.

In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data.

Prior to Pandas, Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, prepare, manipulate, model, and analyze.

Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Numpy:

Numpy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It is open-source software.

It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

The Python programming language was not originally designed for numerical computing, but attracted the attention of the scientific and engineering community early on. In 1995 the special interest group (SIG) matrix-sig was founded with the aim of defining an array computing package; among its members was Python designer and maintainer Guido van Rossum, who extended Python's syntax (in particular the indexing syntax) to make array computing easier.

An implementation of a matrix package was completed by Jim Fulton, then generalized[further explanation needed] by Jim Hugunin and called Numeric (also variously known as the "Numerical Python extensions" or "NumPy"). Hugunin, a graduatestudent at the Massachusetts Institute of Technology (MIT), joined the Corporation for National Research Initiatives (CNRI) in 1997 to work on JPython, leaving Paul Dubois of Lawrence Livermore National Laboratory (LLNL) to take over as maintainer. Other early contributors include David Ascher, Konrad Hinsen and Travis Oliphant.

A new package called Numarray was written as a more flexible replacement for Numeric. Like Numeric, it too is now deprecated. Numarray had faster operations for largearrays, but was slower than Numeric on small ones, so for a time both packages were usedin parallel for different use cases. The last version of Numeric (v24.2) was released on 11 November 2005, while the last version of num array (v1.5.2) was released on 24 August 2006.

There was a desire to get Numeric into the Python standard library, but Guido van Rossum decided that the code was not maintainable in its state then.

In early 2005, NumPy developer Travis Oliphant wanted to unify the community around a single array package and ported Num array's features to Numeric, releasing the result as NumPy 1.0 in 2006. This new project was part of SciPy. To avoid installing the large SciPy

package just to get an array object, this new package was separated and called NumPy. Support for Python 3 was added in 2011 with NumPy version 1.5.0.

In 2011, PyPy started development on an implementation of the NumPy API for PyPy. It is not yet fully compatible with NumPy.

NumPy targets the CPython reference implementation of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents due to the absence of compiler optimization. NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays; using these requires rewriting some code, mostly inner loops, using NumPy.

Using NumPy in Python gives functionality comparable to MATLAB since they are both interpreted,[21] and they both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars. In comparison, MATLAB boastsa large number of additional toolboxes, notably Simulink, whereas NumPy is intrinsically integrated with Python, a more modern and complete programming language. Moreover, complementary Python packages are available; SciPy is a library that adds more MATLAB-like functionality and Matplotlib is a plotting package that provides MATLAB-like plotting functionality. Internally, both MATLAB and NumPy relyon BLAS and LAPACK for efficient linear algebra computations.

Python bindings of the widely used computer vision library OpenCV utilize numpy arrays to store and operate on data. Since images with multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image. The NumPy array as universal data structure in OpenCV for images, extracted feature points, filter kernels and many more vastly simplifies the programming workflow and debugging.

Array Creation

```
>>> import numpy as np
>>> x = np.array([1, 2, 3])
>>> x
array([1, 2, 3])
>>> y = np.arange(10)  # like Python's list(range(10)), but returns an array
>>> y
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

Matplotlib:

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib.

Matplotlib was originally written by John D. Hunter. Since then, it has an active development community and is distributed under a BSD-style license. Michael Droettboom was nominated as matplotlib's lead developer shortly before John Hunter's death in August 2012 and was further joined by Thomas Caswell. Matplotlib is a NumFOCUS fiscally sponsored project.

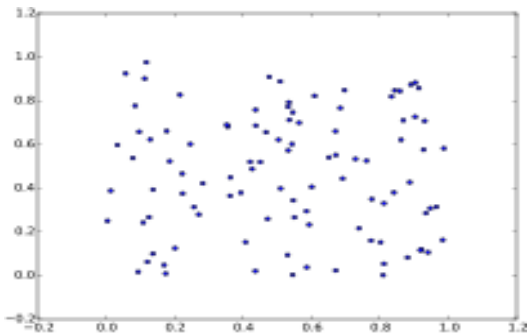
Pyplot is a Matplotlib module which provides a MATLAB-like interface. Matplotlib is designed to be as usable as MATLAB, with the ability to use Python, and the advantage of being free and open-source.

A Python matplotlib script is structured so that a few lines of code are all that is required in most instances to generate a visual data plot.

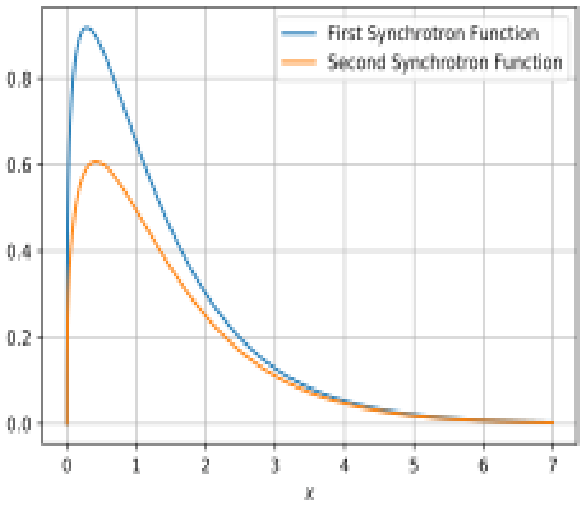
The matplotlib scripting layer overlays two APIs:

- The pyplot API is a hierarchy of Python code objects topped by matplotlib.pyplot
- An OO (Object-Oriented) API collection of objects that can be assembled with greater flexibility than pyplot. This API provides direct access to Matplotlib's backend layers.

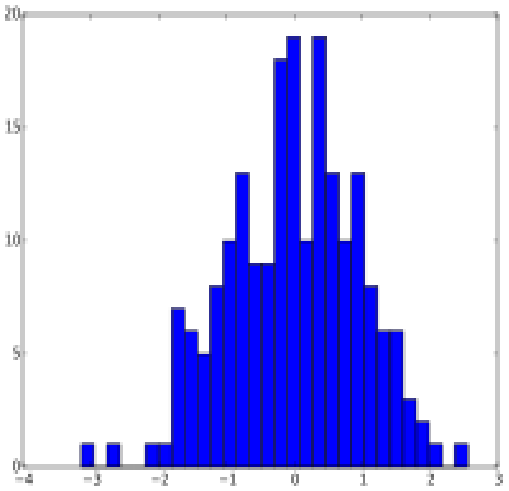
Example:



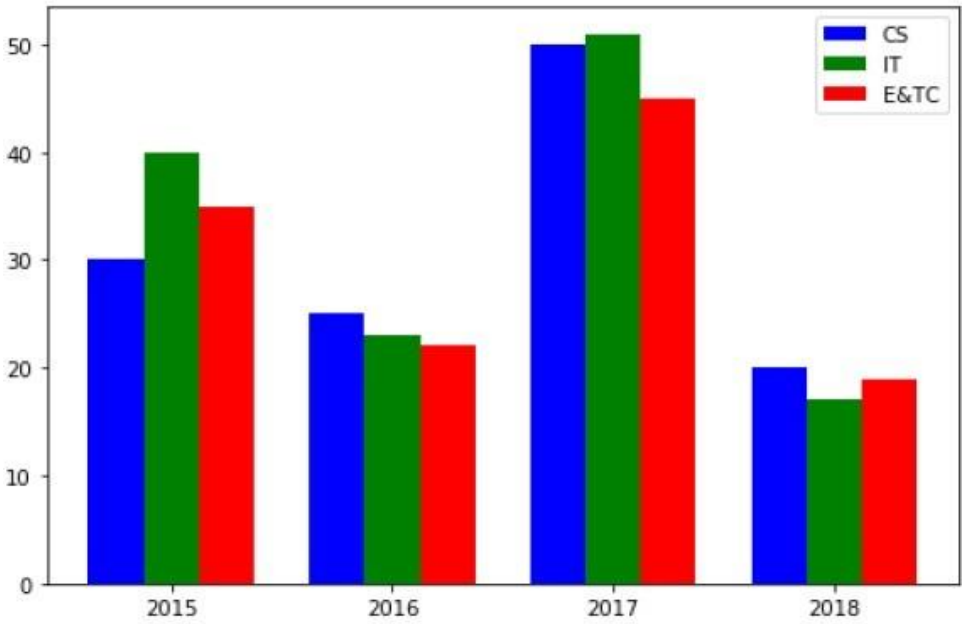
Scatter plot

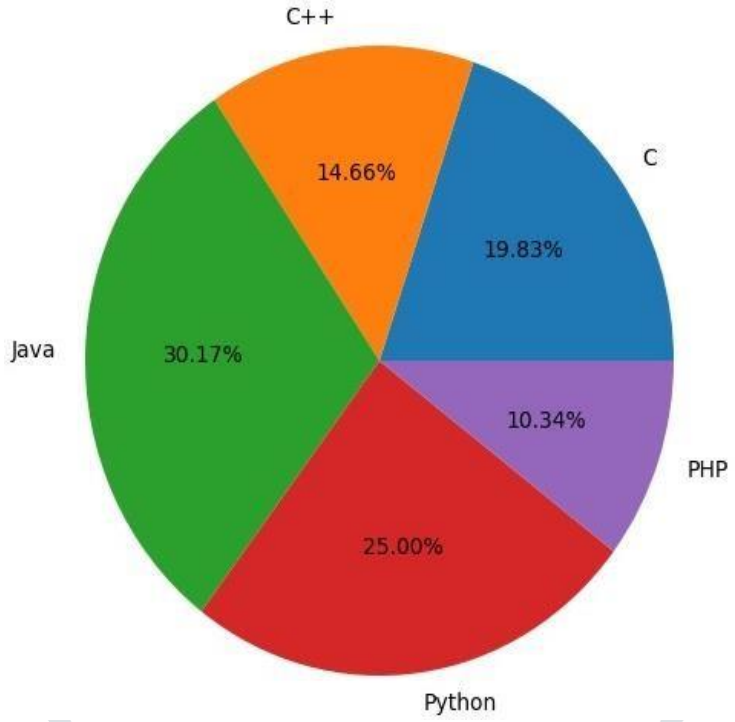


Line plot



Histogram





Seaborn:

Seaborn is a library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with pandas' data structures.

Seaborn helps you explore and understand your data. Its plotting functions operate on dataframes and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots.

Its dataset-oriented, declarative API lets you focus on what the different elements of your plots mean, rather than on the details of how to draw them.

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

For a brief introduction to the ideas behind the library, you can read the introductorynotes or the paper. Visit the installation page to see how you can download the package and get started with it. You can browse the example gallery to see some of the things that you can do with seaborn, and then check out the tutorial or API reference to find out how.

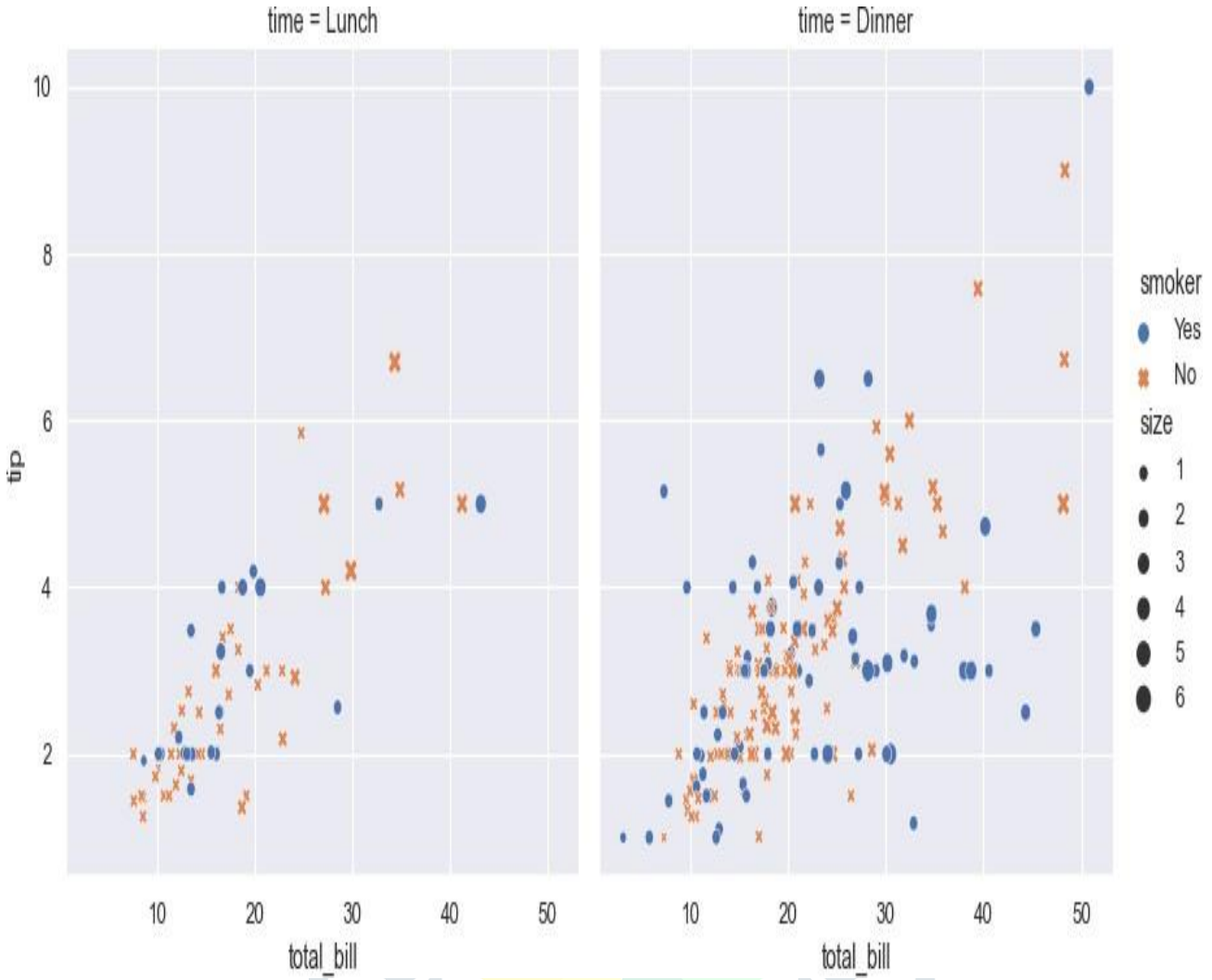
To see the code or report a bug, please visit the GitHub repository. General support questions are most at home on stackoverflow or discourse, which have dedicated channels for seaborn.

Seaborn is a library for making statistical graphics in Python. It is built on topof matplotlib and is tightly integrated with the PyData stack, including supportfor numpy and pandas' data structures, and statistical routines from scipy and statsmodels.

Here is some of the functionality that seaborn offers:

- A dataset-oriented API for examining relationships between multiple variables
- Convenient views onto the overall structure of complex datasets
- Specialized support for using categorical variables to show observations or aggregatestatistics
- Options for visualizing univariate or bivariate distributions and for comparing thembetween subsets of data
- Automatic estimation and plotting of linear regression models for different kinds ofdependent variables
- High-level abstractions for structuring multi-plot grids that let you easily build complexvisualizations
- Concise control over matplotlib figure styling with several built-in themes

```
import seaborn as sns
```


time = Lunch
time = Dinner
tip
total_bill
total_bill
smoker
Yes
No
size
1
2
3
4
5
6

Scikitlearn:

Scikit-learn (formerly **scikits.learn** and also known as **sklearn**) is a free software machine learning library for the Python programming language. It features,various classification, regression and clustering algorithms,including support-vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientificlibraries NumPy and SciPy. Scikit-learn is a NumFOCUS fiscally sponsored project.

Scikit-learn is probably the most useful library for machine learning in Python. The sklearn library contains a lot of efficient tools for machine learning and statistical modelling including classification, regression, clustering and dimensionality reduction.

Scikit-learn is an indispensable part of the Python machine learning toolkit at JPMorgan. It is very widely used across all parts of the bank for classification, The scikit-learn project started as scikits.learn, a Google Summer of Code project by French data scientist David Cournapeau. Its name stems from the notion that it is a "SciKit" (SciPy Toolkit), a separately-developed and distributed third-party extension to SciPy. The original codebase was later rewritten by other developers. In 2010 Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort and Vincent Michel, all from the French Institute for Research in Computer Science and Automation in Rocquencourt, France, took leadership of the project and made the first public release on February the 1st 2010. Of the various scikits, scikit-learn as well as scikit-image were described as "well-maintained and popular" in November 2012. Scikit-learn is one of the most popular machine learning libraries on GitHub.

Scikit-learn is largely written in Python, and uses NumPy extensively for high-performance linear algebra and array operations. Furthermore, some core algorithms are written in Cython to improve performance. Support vector machines are implemented by a Cython wrapper around LIBSVM; logistic regression and linear support vector machines by a similar wrapper around LIBLINEAR. In such cases, extending these methods with Python may not be possible.

Scikit-learn integrates well with many other Python libraries, suchas Matplotlib and plotly for plotting, NumPy for arrayvectorization, Pandas dataframes, SciPy, and many more.

Version History

Scikit-learn was initially developed by David Cournapeau as a Google summer of code project in 2007. Later Matthieu Brucher joined the project and started to use it as a part of his thesis work. In 2010 INRIA, the French Institute for Research in Computer Science and

Automation, got involved and the first public release (v0.1 beta) was published in late January 2010.

- August 2013. scikit-learn 0.14
- July 2014. scikit-learn 0.15.0
- March 2015. scikit-learn 0.16.0
- November 2015. scikit-learn 0.17.0
- September 2016. scikit-learn 0.18.0
- July 2017. scikit-learn 0.19.0
- September 2018. scikit-learn 0.20.0
- May 2019. scikit-learn 0.21.0
- December 2019. scikit-learn 0.22.0
- May 2020. scikit-learn 0.23.0
- Jan 2021. scikit-learn 0.24
- September 2021. scikit-learn 1.0

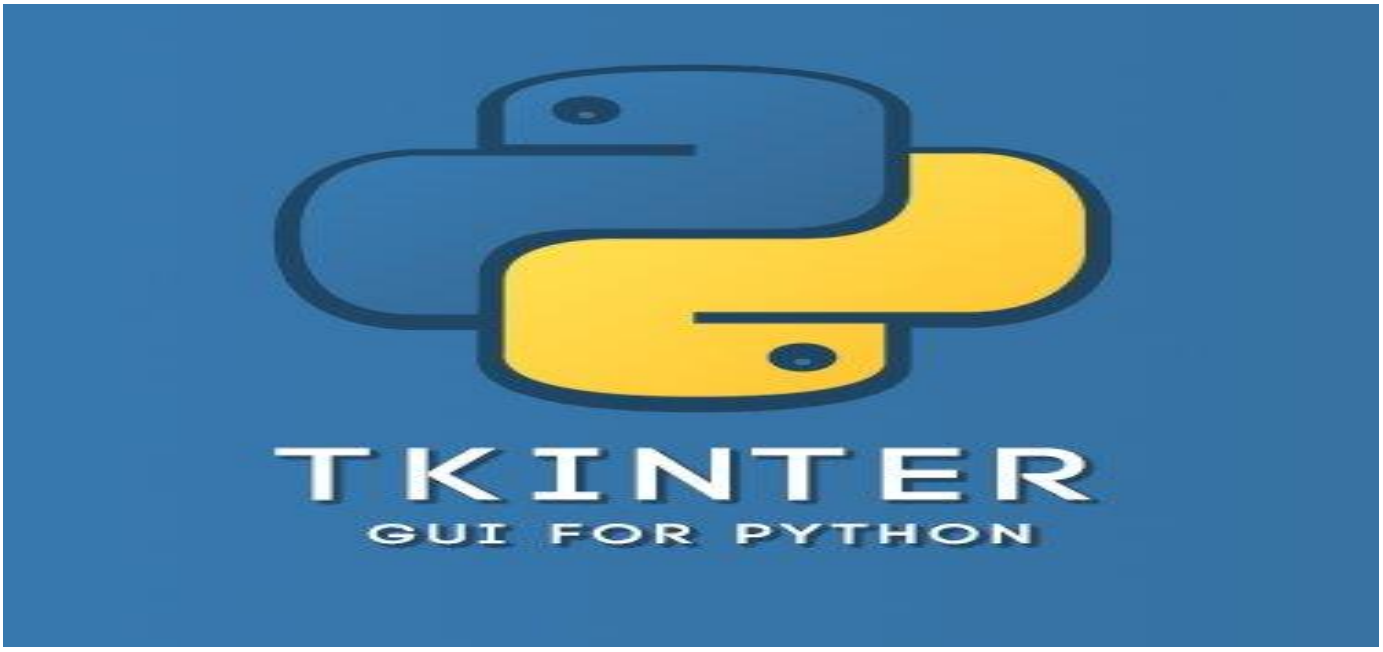
Tkinter :

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

To create a tkinter app:

1. Importing the module – tkinter
2. Create the main window (container)
3. Add any number of widgets to the main window
4. Apply the event Trigger on the widgets.

Importing tkinter is same as importing any other module in the Python code.



Pillow :

Python Pillow module is built on top of PIL (Python Image Library). It is the essential modules for image processing in Python. But it is not supported by Python 3. But, we can use this module with the Python 3.x versions as PIL. It supports the variability of images such as jpeg, png, bmp, gif, ppm, and tiff.

We can do anything on the digital images using the pillow module. In the upcoming section, we will learn various operations on the images such as filtering images, Creating thumbnail, merging images, cropping images, blur an image, resizing an image, creating a water mark and many other operations.

Operating System:

The OS module in Python provides functions for creating and removing a directory (folder), fetching its contents, changing and identifying the current directory, etc.

The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules.

This module provides a portable way of using operating system-dependent functionality.

The "os" and "os.path" modules include many functions to interact with the file system.

OS is a core Python module that allows exposure of OS-specific tasks.

Contains...

- os.error - an alias for the OSError exception
- os.name - the registered name of the OS. For example: 'posix', 'nt','win32', 'win64'.
- os.environ - the environment that Python interpreter currently sees. This object is likea dict. Setting will call 'putenv'.
- os.chdir(path), os.fchdir(fd), os.getcwd() - current directory functions. chdir CHangesDIRectory, getcwd GETs Current Working Directory and fchdir changes to the directthat the file that is specified by the fd is in.

It is possible to automatically perform many operating system tasks. The OS module in Python provides functions for creating and removing a directory (folder), fetching its contents, changing and identifying the current directory, etc.

You first need to import the os module to interact with the underlying operating system. So, import it using the import os statement before using its functions.

Getting Current Working Directory
The getcwd() function confirms returns the current working directory.

Example: Get Current Working Directory

```
>>> import os
>>> os.getcwd()
'C:\\Python37'
```

Creating a Directory

We can create a new directory using the os.mkdir() function, as shown below.

Example: Create a Physical Directory

```
>>> import os
>>> os.mkdir("C:\MyPythonProject")
```

A new directory corresponding to the path in the string argument of the function will be created. If you open the C:\ drive, then you will see the MyPythonProject folder has been created.

By default, if you don't specify the whole path in the mkdir() function, it will create the specified directory in the current working directory or drive. The following will create MyPythonProject in the C:\Python37 directory.

Example: Create a Physical Directory

```
>>> import os
>>> os.getcwd()
'C:\Python37'
>>> os.mkdir("MyPythonProject")
```

Changing the Current Working Directory

We must first change the current working directory to a newly created one before doing any operations in it. This is done using the chdir() function. The following change current working directory to C:\MyPythonProject.

Example: Change Working Directory

```
>>> import os
>>> os.chdir("C:\MyPythonProject") # changing current workign directory
>>> os.getcwd()
'C:\MyPythonProject'
```

You can change the current working directory to a drive. The following makesthe C:\ drive as the current working directory.

Example: Change Directory to Drive

```
>>> os.chdir("C:\\")
>>> os.getcwd()
'C:\\'
```

In order to set the current directory to the parent directory use ".." as the argument in the chdir() function.

Example: Change CWD to Parent

```
>>> os.chdir("C:\\MyPythonProject")
>>> os.getcwd()
'C:\\MyPythonProject'
>>> os.chdir("..")
>>> os.getcwd()
'C:\\'
```

Removing a Directory

The rmdir() function in the OS module removes the specified directory either with an absolute or relative path. Note that, for a directory to be removed, it should be empty.

Example: Remove Directory

```
>>> import os
>>> os.rmdir("C:\\MyPythonProject")
```

However, you cannot remove the current working directory. To remove it, you must change the current working directory, as shown below.

```
>>> import os
>>> os.getcwd() 'C:\\MyPythonProject'
>>> os.rmdir("C:\\MyPythonProject")
PermissionError: [WinError 32] The process cannot access the file because it is being used by another process: 'd:\\MyPythonProject'
>>> os.chdir("..")
>>> os.rmdir("MyPythonProject")
```

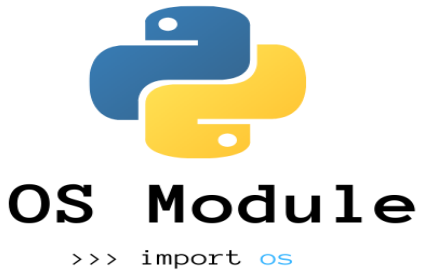
Above, the MyPythonProject will not be removed because it is the current directory. We changed the current working directory to the parent directory using os.chdir("..") andthen remove it using the rmdir() function.

List Files and Sub-directories

The listdir() function returns the list of all files and directories in the specified directory.

```
>>> import os
>>> os.listdir("c:\python37")
['DLLs', 'Doc', 'fantasy-1.py', 'fantasy.db', 'fantasy.py', 'frame.py',
'gridexample.py', 'include', 'Lib', 'libs', 'LICENSE.txt', 'listbox.py', 'NEWS.txt', 'place.py', 'players.db',
'python.exe', 'python3.dll', 'python36.dll', 'pythonw.exe','sclst.py', 'Scripts', 'tcl', 'test.py', 'Tools',
'tooltip.py', 'vcruntime140.dll', 'virat.jpg', 'virat.py']
```

If we don't specify any directory, then list of files and directories in the current working directory will be returned.



SQLite:

SQLite is a self-contained, file-based SQL database. SQLite comes bundled with Python and can be used in any of your Python applications without having to install any additional software. In this tutorial, we'll go through the sqlite3 module in Python 3.

SQLite is a database engine written in the python language. It is not a standalone app; rather, it is a library that software developers embed in their apps. As such, it belongs to thefamily of embedded databases. It is the most widely deployed database engine, as it is usedby several of the top web browsers, operating systems, mobile phones, and other embedded systems.

SQLite has bindings to many programming languages.It generally follows PostgreSQL syntax but does not enforce type checking. This means that one can, for example, insert a string into a column defined as an integer.

D. Richard Hipp designed SQLite in the spring of 2000 while working for General Dynamics on contract with the United States Navy. Hipp was designing software used fora damage-control system aboard guided-missile destroyers, which originally used HP-UX with an IBM Informix database back-end. SQLite began as a Tcl extension.

The design goals of SQLite were to allow the program to be operated without installing a database management system or requiring a database administrator. Hipp based the syntax and semantics on those of PostgreSQL 6.5. In August 2000, version 1.0 of SQLite was released, with storage based on gdbm (GNU Database Manager). In September 2001, SQLite 2.0 replaced gdbm with a custom B-tree implementation, adding transaction capability.In June 2004,SQLite 3.0 added internationalization, manifest typing, and other major improvements, partially funded by America Online.

In 2011, Hipp announced his plans to add a NoSQL interface (managing documents expressed in JSON) to SQLite databases and to develop UnQLite,an embeddable document-oriented database.

SQLite is one of four formats recommended for long-term storage of datasets approved for use by the Library of Congress.

SQLite implements most of the SQL-92 standard for SQL, but lacks some features. For example, it only partially provides triggers and cannot write to views (however, it provides INSTEAD OF triggers that provide this functionality). Its support of ALTER TABLE statements is limited.[20]

SQLite uses an unusual type system for a SQL-compatible DBMS: instead of assigning a type to a column as in most SQL database systems, types are assigned to individual values; in language terms it is dynamically typed. Moreover, it is weakly typed in some ofthe same ways that Perl is: one can insert a string into an integer column (although SQLitewill try to convert the string to an integer first, if the column's preferred type is integer). This adds flexibility to columns, especially when bound to a dynamically typed scripting language. However, the technique is not portable to other SQL products. A common

criticism is that SQLite's type system lacks the data integrity mechanism provided by statically typed columns in other products. The SQLite web site describes a "strict affinity" mode, but this feature has not yet been added.[19] However, it can be implemented with constraints like `CHECK(typeof(x)='integer')`.[9]

Tables normally include a hidden rowid index column, which gives faster access.[21] If a database includes an Integer Primary Key column, SQLite will typically optimize it by treating it as an alias for rowid, causing the contents to be stored as a strictly typed 64-bit signed integer and changing its behavior to be somewhat like an auto-incrementing column. Future[when?] versions of SQLite may include a command to introspect whether a column has behavior like that of rowid to differentiate these columns from weakly typed, non-autoincrementing Integer Primary Keys.

RE (regular expression):

A regular expression is a special sequence of characters that helps you match or find other strings or sets of strings, using a specialized syntax held in a pattern. Regular expressions are widely used in UNIX world. The Python module re provides full support for Perl-like regular expressions in Python.

A regular expression (shortened as regex or regexp; sometimes referred to as rational expression) is a sequence of characters that specifies a search pattern in text. Usually such patterns are used by string-searching algorithms for "find" or "find and replace" operations on strings, or for input validation. Regular expression techniques are developed in theoretical computer science and formal language theory.

The concept of regular expressions began in the 1950s, when the American mathematician Stephen Cole Kleene formalized the concept of a regular language. They came into common use with Unix text-processing utilities. Different syntaxes for writing regular expressions have existed since the 1980s, one being the POSIX standard and another, widely used, being the Perl syntax.

Regular expressions are used in search engines, in search and replace dialogs of word processors and text editors, in text processing utilities such as sed and AWK, and in lexical analysis. Most general-purpose programming languages support regex capabilities either natively or via libraries, including for example Python, C, C++, Java, and JavaScript.

Regular expressions originated in 1951, when mathematician Stephen Cole Kleene described regular languages using his mathematical notation called regular events. These arose in theoretical computer science, in the subfields of automata theory (models of computation) and the description and classification of formal languages. Other early implementations of pattern matching include the SNOBOL language, which did not use regular expressions, but instead its own pattern matching constructs.

Regular expressions entered popular use from 1968 in two uses: pattern matching in a text editor and lexical analysis in a compiler. Among the first appearances of regular expressions in program form was when Ken Thompson built Kleene's notation into the editor QED as a means to match patterns in text files. For speed, Thompson implemented regular expression matching by just-in-time compilation (JIT) to IBM 7094 code on the Compatible Time-Sharing System, an important early example of JIT compilation. He later added this capability to the Unix editor ed, which eventually led to the popular searchtool grep's use of regular expressions ("grep" is a word derived from the command for regular expression searching in the ed editor: g/re/p meaning "Global search for Regular Expression and Print matching lines"). Around the same time when Thompson developed QED, a group of researchers including Douglas T. Ross implemented a tool based on regular expressions that is used for lexical analysis in compiler design.

Many variations of these original forms of regular expressions were used in Unix[15] programs at Bell Labs in the 1970s, including vi, lex, sed, AWK, and expr, andin other programs such as Emacs. Regexes were subsequently adopted by a wide range of programs, with these early forms standardized in the POSIX.2 standard in 1992.

In the 1980s, the more complicated regexes arose in Perl, which originally derived from a regex library written by Henry Spencer (1986), who later wrote an implementation of Advanced Regular Expressions for Tcl.[18] The Tcl library is a hybrid NFA/DFA implementation with improved performance characteristics. Software projects that have adopted Spencer's Tcl regular expression implementation include PostgreSQL.[19] Perl later expanded on Spencer's original library to add many newfeatures.[20] Part of the effort in the design of Raku (formerly named Perl 6) is to improve Perl's regex integration, and to increase their scope and capabilities to allow the definition of parsing expression grammars.[21] The result is a mini-language called Raku rules, whichare used to define Raku grammar as well as provide a tool to programmers in the language.These rules maintain existing features of Perl 5.x regexes, but also allow BNF-style definition of a recursive descent parser via sub-rules.

Problem Statement

- The problem statement of loan prediction is to accurately assess the creditworthiness of borrowers and predict the likelihood of loan default. This is crucial for lenders to make informed decisions about loan approvals, interest rates, and loan terms.

- All over the world banks, housing finance companies deal in a multitude of types of personal loans,home loans and business loans. These companies validate the eligibility of the companies when they apply for the loans.

- To develop accurate, efficient, and automated methods for assessing creditworthiness, predicting loan default, and managing risk. This will enable lenders to make informed lending decisions, reduce the risk of default

- Our project will automate this process by analyzing the online form that customers will fill as a requirement by employing machine learning and calculate whether the customer is eligible for the loan or not.

Business Objective:

➢ **Risk Management:** Loan prediction helps lenders manage their risk exposure by identifying potential high-risk borrowers. By accurately assessing the creditworthiness of applicants, lenders can make informed decisions about loan approvals, interest rates, and loan terms. This reduces the chances of lending to individuals or businesses with a higher probability of default.

➢ **Responsible Lending:** Loan prediction promotes responsible lending practices. By using predictive models, lenders can ensure that loans are granted to borrowers who have a higher likelihood of repayment. This helps prevent borrowers from taking on loans they cannot afford and reduces the risk of financial hardship or default.

➢ **Efficiency and Cost Savings:** Loan prediction improves the efficiency of the lending process. By automating the assessment of creditworthiness, lenders can streamline their operations, reduce manual effort, and save time and resources. This allows lenders to focus on more complex cases and provide faster loan approvals to eligible borrowers.

➢ **Customer Experience:** Loan prediction can enhance the customer experience by providing quicker loan decisions. Borrowers benefit from a more efficient process, reduced paperwork, and faster access to funds. Additionally, borrowers with good credit profiles may receive more favorable loan terms, such as lower interest rates, based on the prediction of their creditworthiness

Chapter 2 : Literature Review

Index no.	Name of research paper	Name of author	Accuracy
1	An Approach for Prediction of Loan Approval using Machine Learning Algorithm	Ahmad Sheikh,Amit Kumar Goel	
2	Bank Loan Prediction System using Machine Learning	Anshika Gupta1 , Vinay Pant	
3	Loan Eligibility Prediction System	Chandan Singh Palhania, Amit Kumar Jaiswal	
4	Recognizing and Predicting the Non-Performing Loans of Commercial Banks	Zhang Yu[1,2] , Guan Yongsheng	Decision Tree : 94% SVM :99.28%
5	Loan Prediction System Using SVM, DT and RF	Atharva K. Ingale1 , Laksh R. Bhamare	Support Vector Machine :78.04% Random Forest : 73.98% Decision Tree : 70.73%
6	DEVELOPING PREDICTION MODEL OF LOAN RISK IN BANKS USING DATA MINING	Aboobyda Jafar Hamid1 and Tarig Mohammed Ahmed	j48 : 78.3784 % bayesNet: 77.4775 % Naive Bayes:73.8739 %
7	Loan Prediction System Using Machine Learning	Anant Shinde1 , Yash Patil2 , Ishan Kotian3	Logistic Regression: 82%
8	Predicting Bank Loan Eligibility Using Machine Learning Models and Comparison Analysis	Miraz Al Mamun	Logistic Regression:

			92% Random forest:91.88%
9	Machine Learning Models for Predicting Bank Loan Eligibility	Ugochukwu .E. Orji Chikodili .H. Ugwuishiwu	Random forest :95.55% Logistic regression :80%.
10	Predicting acceptance of the bank loan offers by using support vector machines	Mehmet Furkan Akça a, and Onur Sevli b	SVM : 97.2%

Explanation of research paper:

1. An Approach for Prediction of Loan Approval using Machine Learning Algorithm

In our banking system, banks have many products to sell but main source of income of any banks is on its credit line. So they can earn from interest of those loans which they credits.A bank's profit or a loss depends to a large extent on loans i.e. whether the customers are paying back the loan or defaulting. By predicting the loan defaulters, the bank can reduce its Non Performing Assets.This makes the study of this phenomenon very important. Pre vious research in this era has shown that there are so many methods to study the problem of controlling loan default. But as the right predictions are very important for the maximization of profits, it is essential to study the nature of the different methods and their comparison. A very important approach in predictive analytics is used to study the problem of predicting loan defaulters: The Logistic regression model.

2 . Bank Loan Prediction System using Machine Learning

With the advancement in technology, there are so many enhancements in the banking sector also. The number of applications is increasing every day for loan approval. There are some bank policies that they have to consider while selecting an applicant for loan approval. Based on some parameters, the bank has to decide which one is best for approval. It is tough and risky to check out manually every person and then recommended for loan approval.

In this work, we use a machine learning technique that will predict the person who is reliable for a loan, based on the previous record of the person whom the loan amount is accredited before. This work's primary objective is to predict whether the loan approval to a specific individual is safe or not.

3. Loan Eligibility Prediction System

In the modern financial system, banks give firms or people looking to buy anything the necessary initial investment. to assess a borrower's creditworthiness and forecast the possibility that they will be granted a loan. For lenders, banks, and financial organisations, a loan eligibility prediction system can be helpful in automating the loan application process and determining the risk of giving money to a certain applicant. It is a piece of software that uses techniques for data analysis and machine learning.a loan eligibility prediction system will be a useful tool for banks, financial institutions, and lenders to automate the application process and determine the risk involved in giving money to a certain borrower. The system entails gathering, pre-processing, and manipulating data; extracting pertinent features; choosing an appropriate machine learning model; training the model; and implementing it in a lending and banking application

4. Recognizing and Predicting the Non-Performing Loans of Commercial Banks

As the reform and opening up going into depth over the past three decades and more, the market economic system has been gradually established. The banking industry grows steadily in the process of the reform. It supports economic development, reduces and defends many financial risks in the processof the reform. However, there are many kinds of risks inside of banks, one of which is that the non-performing loans (NPLs) ratio is too high. Therefore, people should focus on how to accurately classify the banking loans into performing and non-performing ones and how to control and prevent the resulting crisis. This paper deeply analyses China's NPLs problem for the current period, recognizes and classifies loans types by adopting decision trees, Naïve Bayes and support vector machine (SVM) methods.

5 . Loan Prediction System Using SVM, DT and RF

Technology has brought significant advancements to the banking industry. With loan applications flooding in every day, it has become more challenging to approve loans. Banks must adhere to strict policies when selecting a candidate for loan approval, considering several criteria to find the most suitable candidate. Manually checking each application for loan approval is arduous and risky. To overcome this, we employ machine learning to predict the loan candidate's
creditworthiness based on their prior performance. Loans generate a substantial proportion of bank profits, but identifying legitimate applicants who will repay the loan is difficult.

6 . DEVELOPING PREDICTION MODEL OF LOAN RISK IN BANKS USING DATA MINING

Nowadays, There are many risks related to bank loans, for the bank and for those who get the loans. The analysis of risk in bank loans need understanding what is the meaning of risk. In addition, the number of transactions in banking sector is rapidly growing and huge data volumes are available which represent the customers behavior and the risks around loan are increased. Data Mining is one of the most motivating and vital area of research with the aim of extracting information from tremendous amount of accumulated data sets. In this paper a new model for classifying loan risk in banking sector by using data mining. The model has been built using data form banking sector to predict the status of loans.

7. Loan Prediction System Using Machine Learning

As the needs of people are increasing, the demand for loans in banks is also frequently getting higher every day. Banks typically process an applicant's loan after screening and verifying the applicant's eligibility, which is a difficult and time-consuming process. In some cases, some applicants default and banks lose capital. The machine learning approach is ideal for reducing human effort and effective decision making in the loan approval process by implementing machine learning tools that use classification algorithms to predict eligible loan applicants.

8. Predicting Bank Loan Eligibility Using Machine Learning Models and Comparison Analysis

As people's demands grow, so does the need for bank loans. Every day, banks get many loan applications from customers and other individuals but not every applicant is accepted. Typically, banks execute a loan application after verifying and evaluating the applicant's

eligibility, which is a time-consuming and challenging process. When examining loan applications and making credit approval decisions, most banks use their credit score and risk assessment systems. Despite this, some applicants fail to pay their bills each year, causing financial institutions to lose a substantial amount of money. In this study, Machine Learning (ML) algorithms are employed to extract patterns from a common loan-approved dataset and predict deserving loan applicants. Customers' previous data will be used to undertake the study, including their age, income type, loan annuity, last credit bureau report, Type of organization they
work for, and length of employment. ML methods such as Random Forest, XGBoost, Adaboost, Lightgbm, Decision tree, and K-Nearest Neighbor were used to discover the maximum relevant features, i.e., the elements that have the most impact on the prediction output.

9. Machine Learning Models for Predicting Bank Loan Eligibility

This paper presents six (6) machine learning algorithms (Random Forest, Gradient Boost, Decision Tree, Support Vector Machine, K-Nearest Neighbor, and Logistic Regression) for predicting loan eligibility. The models were trained on the historical dataset 'Loan Eligible Dataset,' available on Kaggle and licensed under Database Contents License (DbCL) v1.0. The dataset was processed and analyzed using Python programming libraries on Kaggle's Jupyter Notebook cloud environment.

10.Predicting acceptance of the bank loan offers by using support vector machines

Predicting acceptance of the bank loan offers by using support vector machines Loans are one of the main profit sources in banking system. Banks try to select reliable customers and offer them personal loans, but customers can sometimes reject bank loan offers. Prediction of this problem is an extra work for banks, but if they can predict which customers will accept personal loan offers, they can make a better profit. Therefore, at this point, the aim of this study is to predict acceptance of the bank loan offers using the Support Vector Machine (SVM) algorithm. In this context, SVM was used to predict results with four kernels of SVM, with a grid search algorithm for better prediction and cross validation for much more reliable results.

Chapter 3 Data Validation

Loan Prediction Data Validation:

In Loan Prediction Project we have Person Background data. We get 3 Excel files;

1. In Home Loan Dataset CSV files have 615 rows and 12 columns.
2. In Person Loan Dataset CSV files have 5001 rows and 12 columns.
3. In Automobile Loan Dataset CSV files have 420 rows and 10 columns.

Chapter 4 Exploratory Data Analysis

Import csv files of EDA

The shape of new csv file is :

1. In Home Loan Dataset CSV files have 615 rows and 12 columns.
2. In Person Loan Dataset CSV files have 5001 rows and 12 columns.
3. In Automobile Loan Dataset CSV files have 420 rows and 10 columns.

1.Personal Loan :

	Precision	Recall	F1-score	Support
0	0.96	0.99	0.97	907
1	0.84	0.61	0.71	93
accuracy			0.95	1000

Macro avg	0.90	0.80	0.84	1000
Weighted avg	0.95	0.95	0.95	1000

Accuracy :95.3%

2.Home Loan :

	Precision	Recall	F1-score	Support
0	0.88	0.44	0.58	32
1	0.83	0.98	0.90	91
accuracy			0.84	123
Macro avg	0.85	0.71	0.74	123
Weighted avg	0.84	0.84	0.82	123

Accuracy :83.73983739837398%

3. Automobile Accuracy :

	Precision	Recall	F1-score	Support
0	0.94	0.97	0.95	64
1	0.00	0.00	0.00	4
accuracy			0.91	68
Macro avg	0.47	0.48	0.48	68
Weighted avg	0.88	0.91	0.90	68

Accuracy :91.17647058823529%

Chapter 6

Technology and Algorithms

Machine Learning:

Machine Learning tutorial provides basic and advanced concepts of machine learning. Our machine learning tutorial is designed for students and working professionals.

Machine learning is a growing technology which enables computers to learn automatically from past data. Machine learning uses various algorithms for building mathematical models and making predictions using historical data or information. Currently, it is being used for various tasks such as image recognition, speech recognition, email filtering, Facebook auto-tagging, recommender system, and many more.

This machine learning tutorial gives you an introduction to machine learning along with the wide range of machine learning techniques such as Supervised, Unsupervised, and Reinforcement learning. You will learn about regression and classification models, clustering methods, hidden Markov models, and various sequential models.

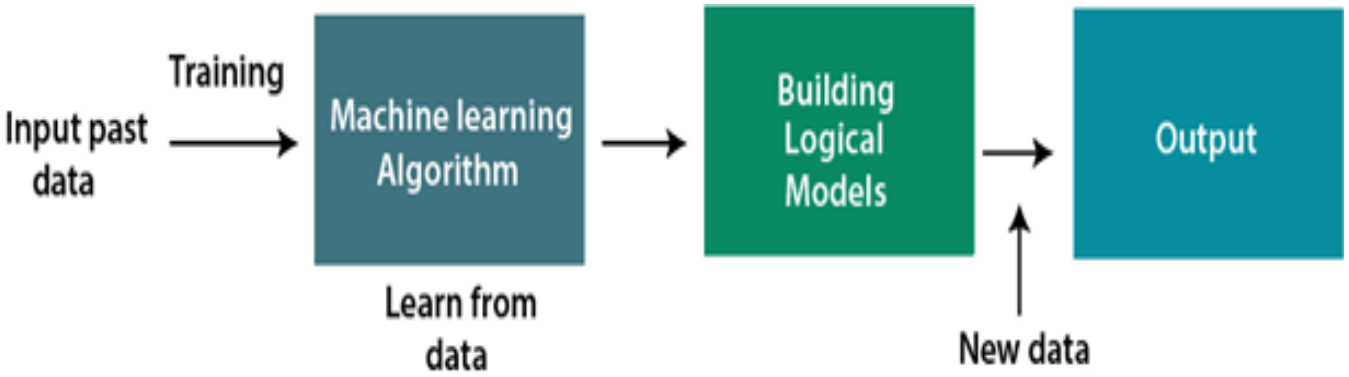
How does Machine Learning work:

A Machine Learning system learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it. The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.

Suppose we have a complex problem, where we need to perform some predictions, so instead of writing a code for it, we just need to feed the data to generic algorithms, and with the help of these algorithms, machine builds the logic as per the data and predict the output. Machine learning has changed our way of thinking about the problem. The below block diagr

am explains the working of Machine Learning algorithm:

Features of Machine Learning:



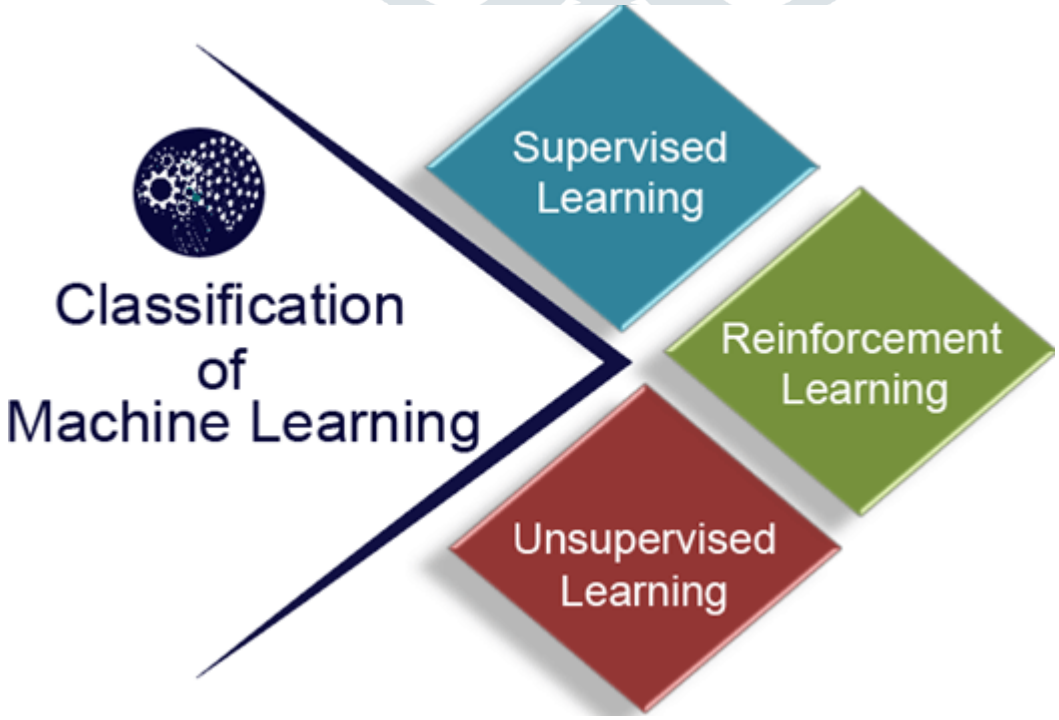
Machine learning uses data to detect various patterns in a given dataset.It can learn from past data and improve automatically.

It is a data-driven technology.

Machine learning is much similar to data mining as it also deals with the huge amount of the data.

classification of Machine Learning :

1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning



1) Supervised Learning :

- Supervised learning is a type of machine learning method in which we provide sample labeled data to the machine learning system in order to train it, and on that basis, it predicts the output.
- The system creates a model using labeled data to understand the datasets and learn about each data, once the training and processing are done then we test the model by providing a sample data to check whether it is predicting the exact output or not.
- The goal of supervised learning is to map input data with the output data.

Types:-

- Support Vector Machine
- Naive Bayes Classifiers

Advantages:-

1.Supervised learning allows collecting data and produces data output from previous experiences.
2.Helps to optimize performance criteria with the help of experience.
3.Supervised machine learning helps to solve various types of real-world computation problems.
4.It performs classification and regression tasks.

Disadvantages:-

1. Classifying big data can be challenging.
2. Supervised learning cannot handle all complex tasks in Machine Learning.
3. Computation time is vast for supervised learning.
4. It requires a labelled data set.

2. Unsupervised Learning :

- The training is provided to the machine with the set of data that has not been labeled, classified, or categorized, and the algorithm needs to act on that data without any supervision.
- The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns.

Advantages of Unsupervised Learning

- Unsupervised learning is used for more complex tasks as compared to supervised learning because, in unsupervised learning, we don't have labeled input data.

- Unsupervised learning is preferable as it is easy to get unlabeled data in comparison to labeled data.

Disadvantages of Unsupervised Learning

- Unsupervised learning is intrinsically more difficult than supervised learning as it does not have corresponding output.
- The result of the unsupervised learning algorithm might be less accurate as input data is not labeled, and algorithms do not know the exact output in advance.

3. Reinforcement Learning :

- Reinforcement learning is a feedback-based learning method, in which a learning agent gets a reward for each right action and gets a penalty for each wrong action.
- The agent learns automatically with these feedbacks and improves its performance. In reinforcement learning, the agent interacts with the environment and explores it.
- The goal of an agent is to get the most reward points, and hence, it improves its performance.

Advantages of Reinforcement Learning :

- It helps to solve very complex problems that conventional techniques fail to solve
- It gives long-term results that are very difficult to accomplish.
- This model works like human learning pattern and hence, demonstrates perfection in every action.

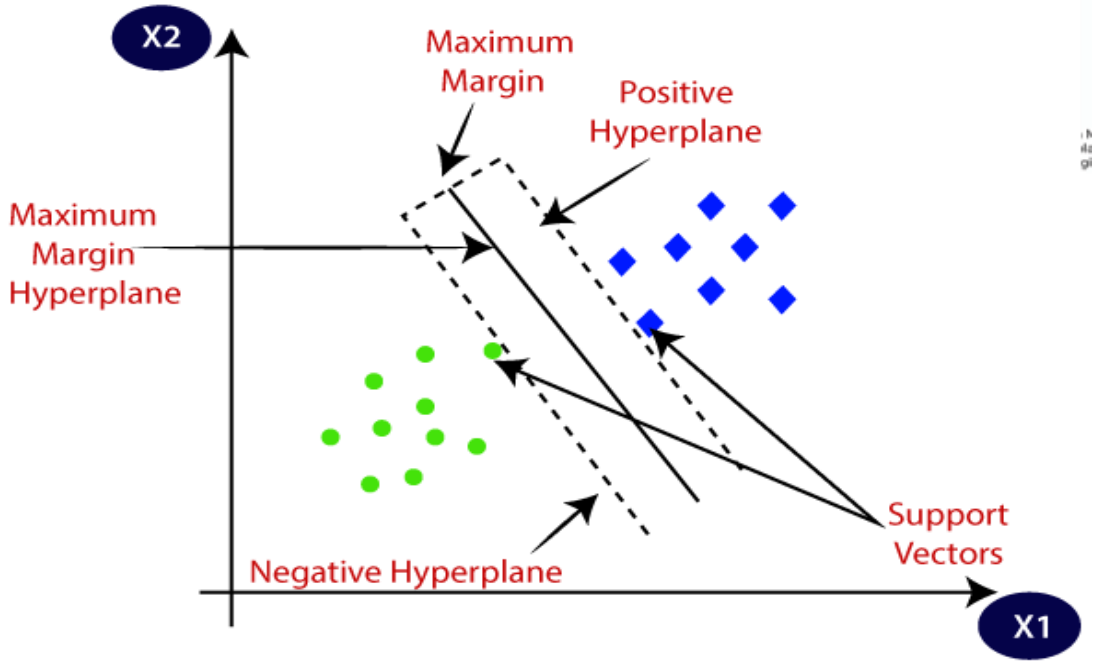
Disadvantages of Reinforcement Learning :

- Too much of reinforcement may cause an overload which could weaken the results.
- Reinforcement learning is preferred for solving complex problems, not simple ones.
- It requires plenty of data and involves a lot of computation.

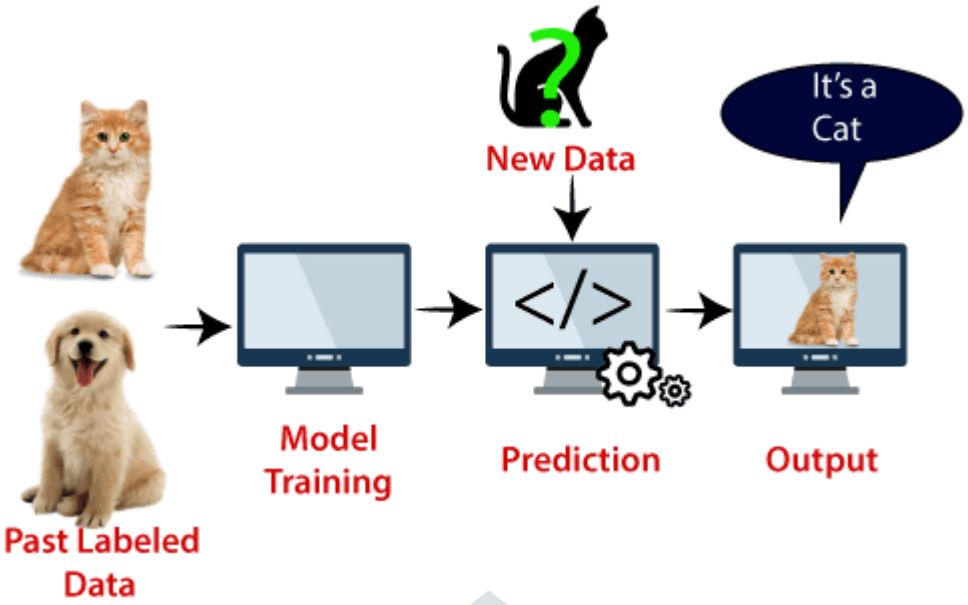
From Supervised Learning we have used Support Vector Machine and Naive bayes algorithm to implement our project :

1. Support Vector Machine :

- Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.
- The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.
- SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.



Example: SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat. Consider the below diagram:

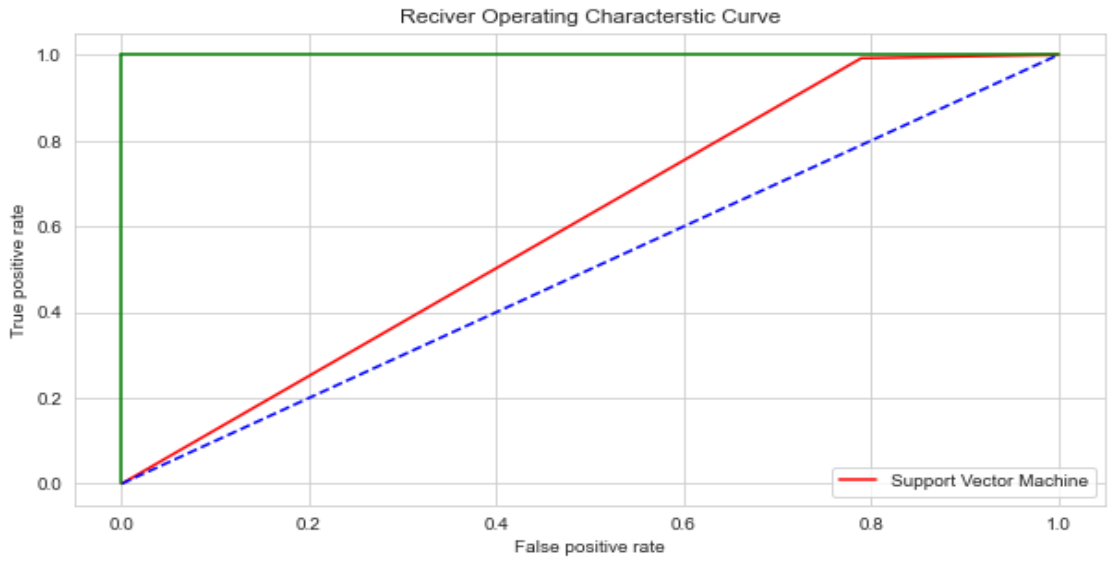


Types of SVM :

SVM can be of two types:

- **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
- **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

In SVM



B

Fig : Graphical Representation of Receiver operating characteristic curve of Support vector Machine

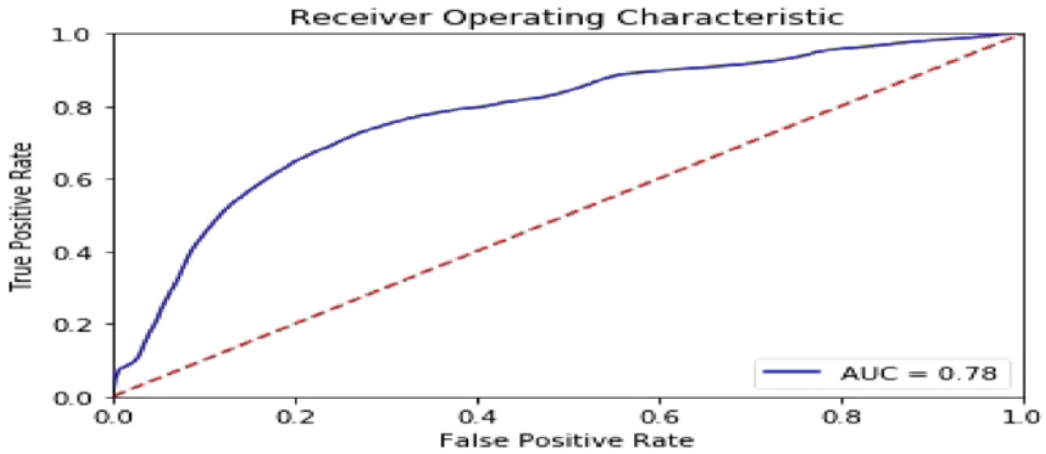
2. Naive Bayes Algorithm :

- Naive Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- It is mainly used in text classification that includes a high-dimensional training dataset.
- Naive Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
- Some popular examples of Naive Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

Here's a brief explanation of how it works:

1. Training: The algorithm learns from a labeled dataset by calculating the probabilities of each feature given each class.
2. Feature Independence: Naive Bayes assumes that the features are independent of each other, which simplifies the calculations.
3. Prior Probability: It calculates the prior probability of each class based on the frequency of occurrence in the training data.
4. Likelihood: It calculates the likelihood of observing the features for each class.
5. Posterior Probability: Using Bayes' theorem, it calculates the posterior probability of each class given the observed features.
6. Classification: The algorithm assigns the class with the highest posterior probability as the predicted class.

Naive Bayes is known for its simplicity, efficiency, and effectiveness in text classification and spam filtering tasks.



Graphical Representation of Receiver operating characteristic curve of naive bayes

Confusion matrix

The confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data. It can only be determined if the true values for test data are known. The matrix itself can be easily understood, but the related terminologies may be confusing. Since it shows the errors in the model performance in the form of a matrix, hence also known as an error matrix.

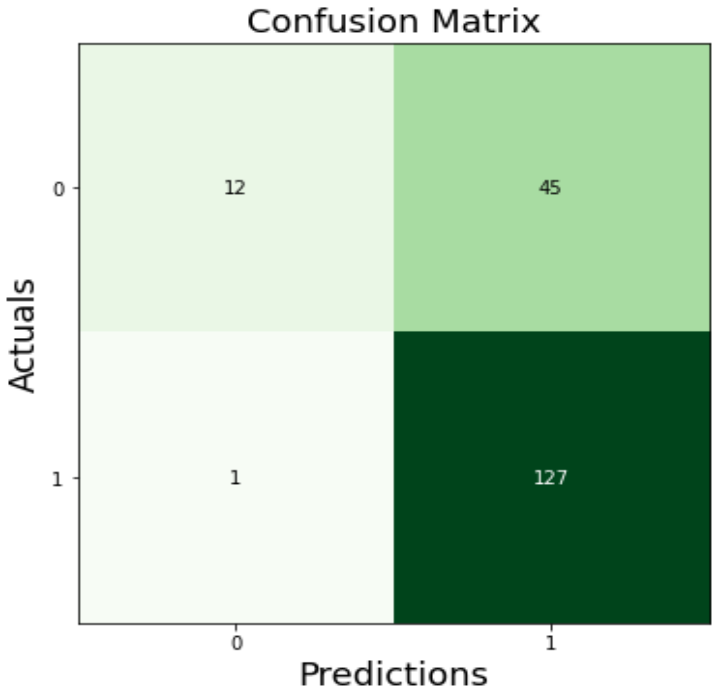
Some features of Confusion matrix are given below:

- The matrix is divided into two dimensions, that are predicted values and actual valuesalong with the total number of predictions.
- Predicted values are those values, which are predicted by the model, and actualvalues are the true values for the given observations.

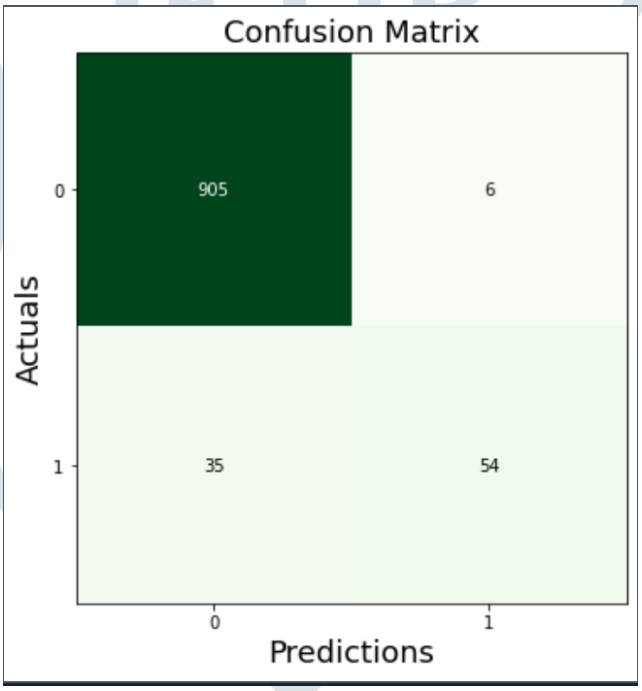
The confusion matrix for an Machine Learning algorithm provides a detailed breakdown of the model's performance. It consists of four components:

1. True Positives (TP): The number of positive instances that the model correctly predicted.
2. True Negatives (TN): The number of negative instances that the model correctly predicted.
3. False Positives (FP): The number of negative instances that the model incorrectly predicted as positive.
4. False Negatives (FN): The number of positive instances that the model incorrectly predicted as negative.
5. By analyzing these values, we can calculate various evaluation metrics such as accuracy, precision, recall, and F1 score to assess the effectiveness of the Machine Learning algorithm.

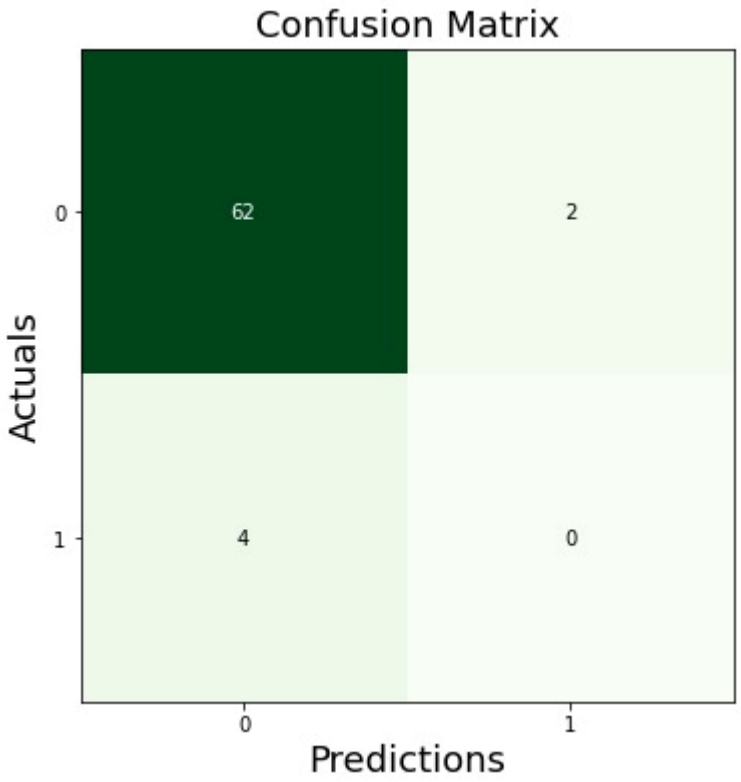
In SVM Confusion Matrix For Home Loan Dataset is :



In SVM Confusion Matrix For Personal Loan Dataset is :



In Naive Bayes Confusion matrix is :



Calculations using Confusion Matrix:

We can perform various calculations for the model, such as the model's accuracy, using this matrix. These calculations are given below:

```
print("=" * 40)
print("==========")
print("Classification Report : ",(classification_report(y_test, y_pred)))
print("Accuracy : ",accuracy_score(y_test,y_pred)*100)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy: %.2f%%" % (accuracy * 100.0))
ACC = (accuracy_score(y_test, y_pred) * 100)
repo = (classification_report(y_test, y_pred))
print("Classification Report :\n")
repo = (classification_report(y_test, y_pred))
print(repo)
print("Confusion Matrix :")
cm = confusion_matrix(y_test,y_pred)
print(cm)
print("\n")
from mlxtend.plotting import plot_confusion_matrix

fig, ax = plot_confusion_matrix(conf_mat=cm, figsize=(6, 6), cmap=plt.cm.Greens)
plt.xlabel('Predictions', fontsize=18)
plt.ylabel('Actuals', fontsize=18)
plt.title('Confusion Matrix', fontsize=18)
plt.show()
end = time.time()
```

- **Classification Accuracy:** It is one of the important parameters to determine the accuracy of the classification problems. It defines how often the model predicts the correct output. It can be calculated as the ratio of the number of correct predictions

made by the classifier to all number of predictions made by the classifiers. The formula is given below:

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

- **Misclassification rate:** It is also termed as Error rate, and it defines how often the model gives the wrong predictions. The value of error rate can be calculated as the number of incorrect predictions to all number of the predictions made by the classifier. The formula is given below:

$$\text{Error rate} = \frac{FP+FN}{TP+FP+FN+TN}$$

- **Precision:** It can be defined as the number of correct outputs provided by the modelor out of all positive classes that have predicted correctly by the model, how many of them were actually true. It can be calculated using the below formula:

$$\text{Precision} = \frac{TP}{TP+FP}$$

- **Recall:** It is defined as the out of total positive classes, how our model predicted correctly. The recall must be as high as possible.

$$\text{Recall} = \frac{TP}{TP+FN}$$

- **F-measure:** If two models have low precision and high recall or vice versa, it is difficult to compare these models. So, for this purpose, we can use F-score. This score helps us to evaluate the recall and precision at the same time. The F-score is maximum if the recall is equal to the precision. It can be calculated using the below formula:

$$\text{F-measure} = \frac{2*Recall*Precision}{Recall+Precision}$$

Other important terms used in Confusion Matrix:

- Null Error rate: It defines how often our model would be incorrect if it always

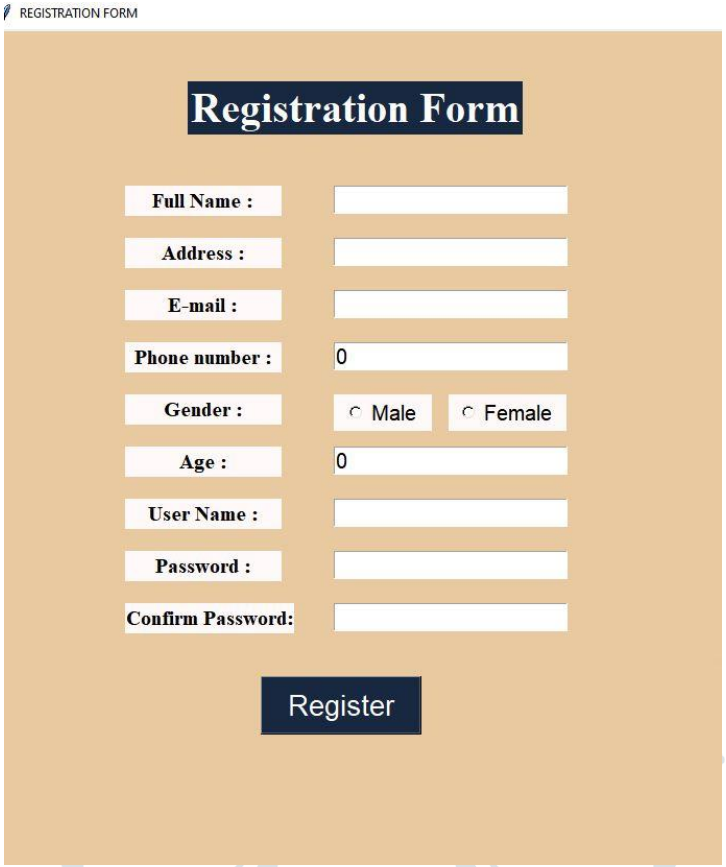
predicted the majority class. As per the accuracy paradox, it is said that "the best classifier has a higher error rate than the null error rate."

- ROC Curve: The ROC is a graph displaying a classifier's performance for all possiblethresholds. The graph is plotted between the true positive rate (on the Y-axis) and the false Positive rate (on the x-axis).

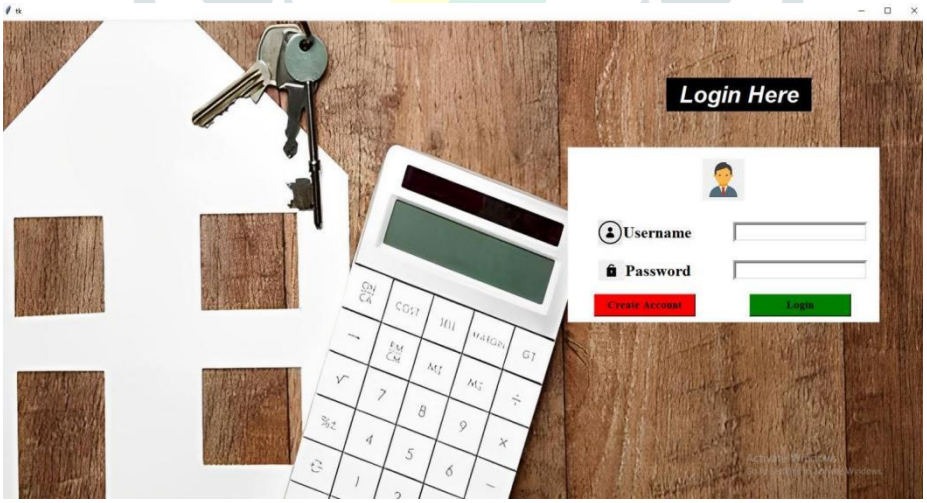
Chapter 7: Model Interface

1. Main GUI :

2. Registration Page :



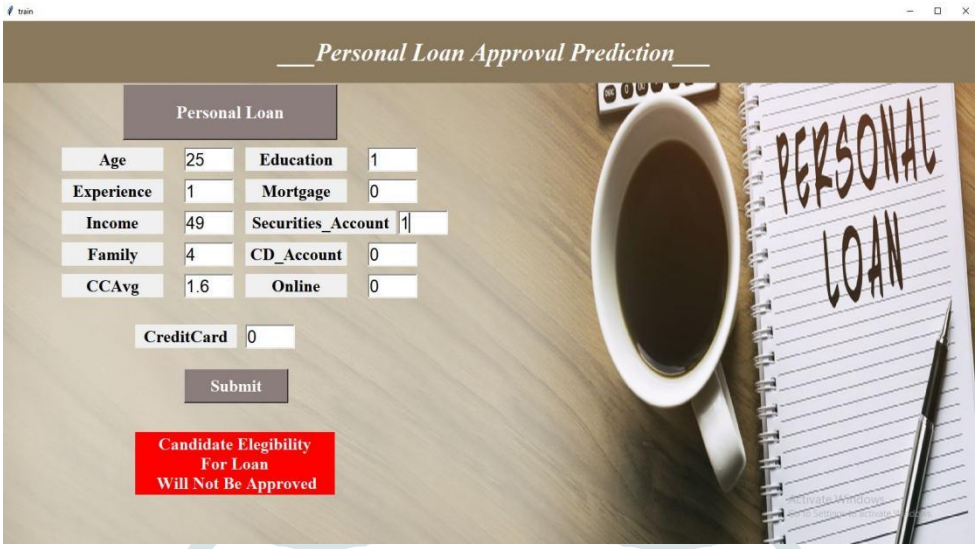
3. Login Page :



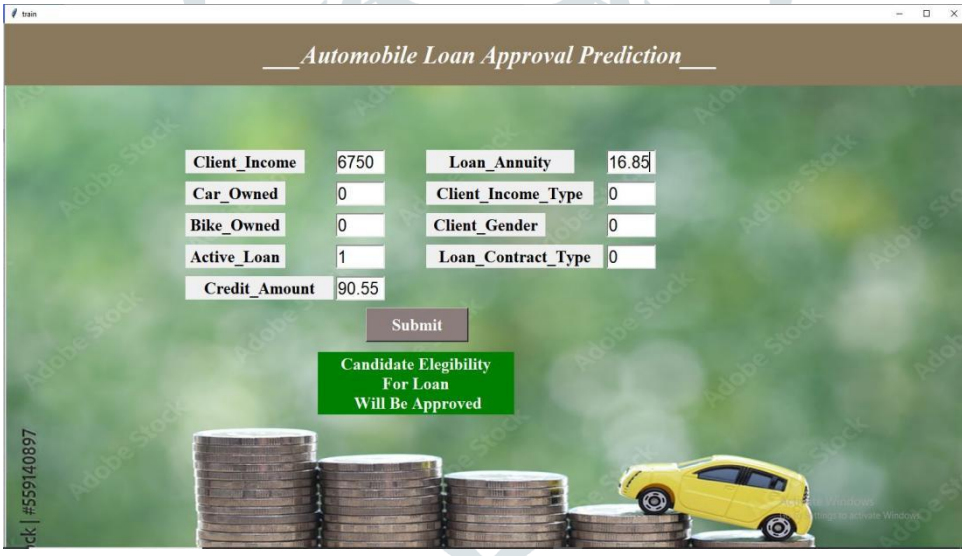
4. Home Loan Page:


train
___Home Loan Approval Prediction___
Home Loan
Gender
Male
Female
Married
No
Yes
Dependents
0
Education
Graduate
Non Graduate
Self_Employed
No
Yes
ApplicantIncome
5720
CoapplicantIncome
0
LoanAmount
110
Loan_Amount_Term
360
Credit_History
1
Property_Area
Urban
Rural
Semi Urban
Submit
Candidate Elegibility
For Loan
Will Be Approved
Activate Windows
Go to Settings to activate Windows.

5. Person Loan page :



6. Automobile Loan page :



Chapter 8
Research Paper

Fraud Detection In credit Card Data Using UnsupervisedMachine Learning Algorithm

ABSTARCT: We are living in the technology world in which Development of communication and e-commerce has made credit card as the most common technique of payment for both online and offline mode of purchases. As the e-commerce has increased, the buying and selling the product online is becoming very easy and comfortable to everyone in the daily life.

Due to this, the online payment and online banking with credit card is increased. The fraud also happens when we lost our credit card or it get stole. Recently due to COVID-19 everything has become contactless so the use of credit card has increased. The transaction is done on online shopping and online payment is done from many places so it become difficult to recognise the real transaction and the fraud transaction. So, it becomes difficulty for the bank to stop fraud detection.

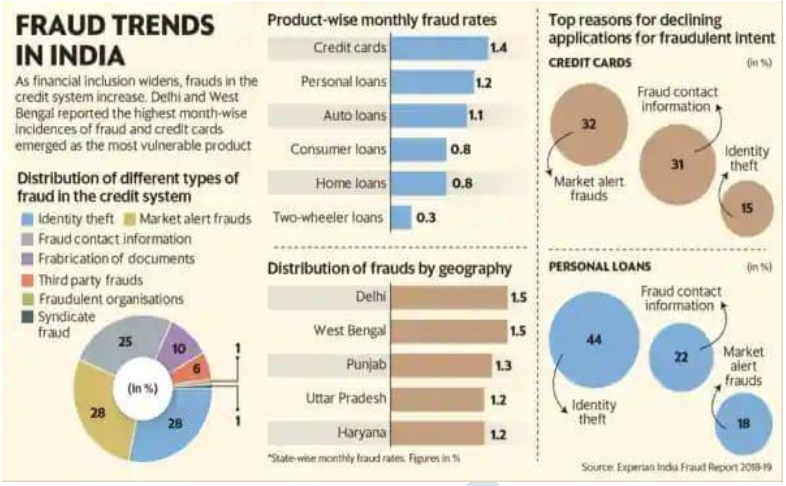
In this paper it clearly explains about how fraud can be detected by using the Unsupervised Machine Learning using the algorithm and by using the algorithm technique like Isolated Forest, Local Outlier Factor and One class SVM.

Keywords: Introduction, Machine learning, Supervised learning, Unsupervised learning.

I. Introduction

From the movement the e-commerce payment system it has found that there have always been a people who will find new ways to access someone's finances illegally. This has become major problem now a days because all the online transaction is being through the credit card. The fraud also happens by sharing the card number and the CVV of the card. In the Covid-19 situation the fraud through the online mode has become more by sharing the card detail and the phone number. So, it become difficult for bank to recognise the fraud happen.

Fraud trends in India:



Credit Card Fraud Detection with Machine Learning is a process of data investigation by a Data Science team and the development of a model that will provide the best results in revealing and preventing fraudulent transactions.

II. Machine Learning

Machine learning (ML) is a type of artificial intelligence (AI) that allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so. Machine learning algorithms use historical data as input to predict new output values.

There are two mostly used method in Machine LearningThey are:

1. Supervised Learning
2. Unsupervised Learning

III. How machine learning helps with fraud detection?

The key objective of any credit card fraud detection system is to identify suspicious events and report them to an analyst while letting normal transactions be automatically processed.

1. Higher accuracy of fraud detection: Compared to rule-based solutions, machine learning tools have higher precision and return more relevant results as they consider multiple additional factors. This is because ML technologies can consider many more data points, including the tiniest details of behaviour patterns associated with a particular account.

2. Less manual work needed for additional verification: Enhanced accuracy leads reduces the burden on analysts. People are unable to check all transactions manually.

3. Fewer false declines: False declines or false positives happen when a system identifies a legitimate transaction as suspicious and wrongly cancels it.

Ability to identify new patterns and adapt to changes: Unlike rule-based systems, ML algorithms are aligned with a constantly changing environment and financial conditions. They enable analysts to identify new suspicious patterns and create new rules to prevent

4. new types of scams.

Rule-based vs ML-based fraud detection:

Rule-based	ML-based
Catching obvious fraudulent scenarios.	Finding hidden and implicit correlations in data.
Requires much manual work to enumerate all possible detection rules.	Automatic detectionof possible fraud scenarios.
Multiple verification steps that harm user experience.	The reduce number of verification measure.
Long-term processing.	Real-time processing.

IV. Anomaly Detection:

An outlier is nothing but a data point that differs significantly from other data points in the given dataset. Anomaly detection is the process of finding the outlier in the data, i.e., points that are significantly different from the majority of the other data points.

Anomalies can be detected in many ways. In this paper we are using the unsupervised machine learning for finding the anomalies on the fraud transaction of the credit card.

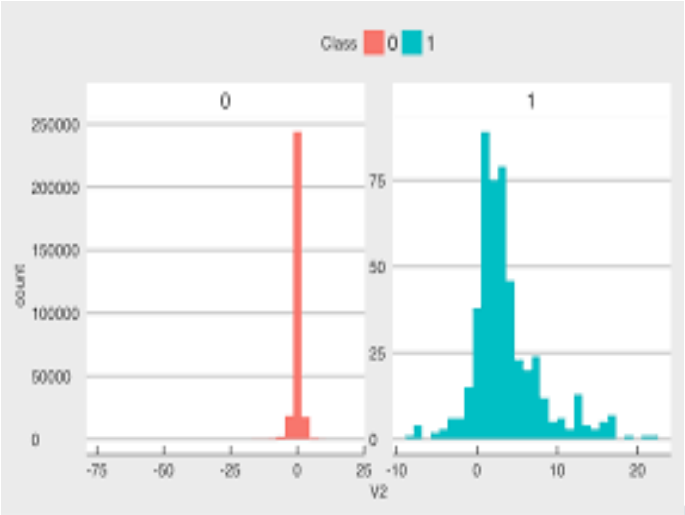


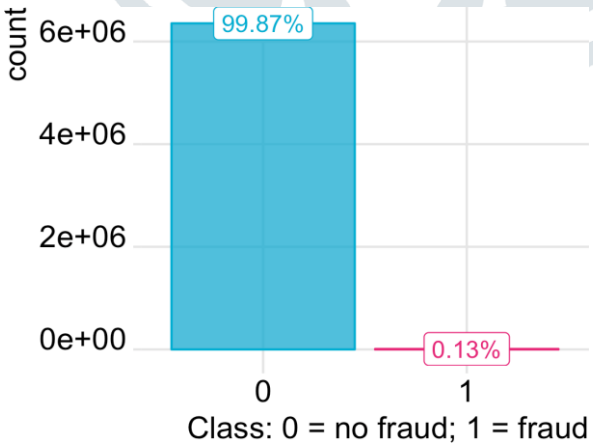
Fig 1: Anomalies detection in credit card Transaction.

In Fraud case implementation as we can see that data is imbalanced with very fee positive Fraud Cases

V. Implementation:

In, this paper we are using the Kaggle data set of the credit card to find the fraudulent transaction by using Unsupervised algorithms. The data set we used is not trained with variable it is directly trained to the actual dataset without any labels.

The datasets contain transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have



492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.132% of all transactions.

To predict the dataset, we had used the three unsupervised algorithm so to test the accuracy and compare the best algorithm. The reason to using the unsupervised in this paper because of the real world, there won't be any labelled variable and for credit card fraudulent unsupervised is the best approach.

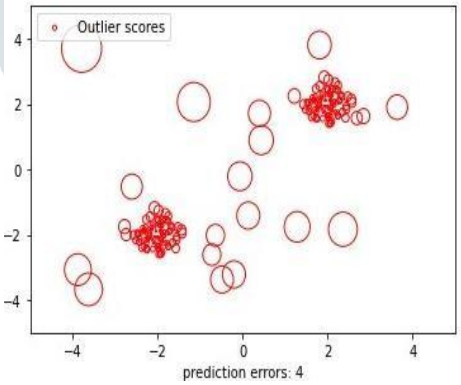
The algorithms we are using is:

I. Local Outlier Factor
II. Isolated Forest
III. One class SVM

I. Local Outlier Factor:

The LOF algorithm is an unsupervised outlier detection method which computes the local density deviation of a given data point with respect to its neighbours. It considers as outlier point that have a substantially lower density than their neighbours.

The parameter for number of neighbours is typically chosen to be greater than the minimum number of objects a cluster has to contain, so that other objects can be local outliers relative



to this cluster, and smaller than the maximum number of close by objects that can potentially be local outliers.

Fig 2. Local Outlier Factor

In the Fig 1 we can see the data points and the outlier scores. As the lower the density from the neighbour points are the outliers.

In the fig 2 we have used data set and showed Local Outlier Factor.

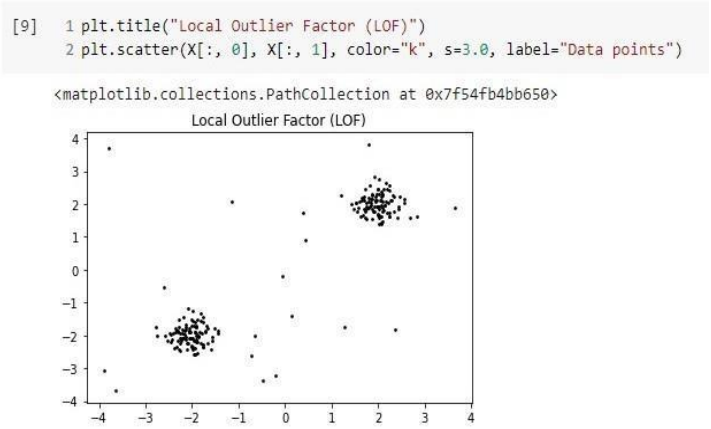


Fig. 3 Local Outlier Factor (Using Dataset)

To understand Local Outlier Factor, we should know about LRD is the Local Reachability Density it is the average reachability distance from its neighbours.

LRD is the Local Reachability Density it is the average reachability distance from its neighbours. Accordingly, to LRD formula, more the reachability distance, less density of points is present around a particular point.

This tells us how the point is far from the nearest cluster points. Low value of LRD tells us the closest cluster is far from the point.

LRD formula:

$$LRD_k(x) = \frac{1}{\frac{\sum d(x,0)}{\sum^{k} |N_k(x)|}}$$

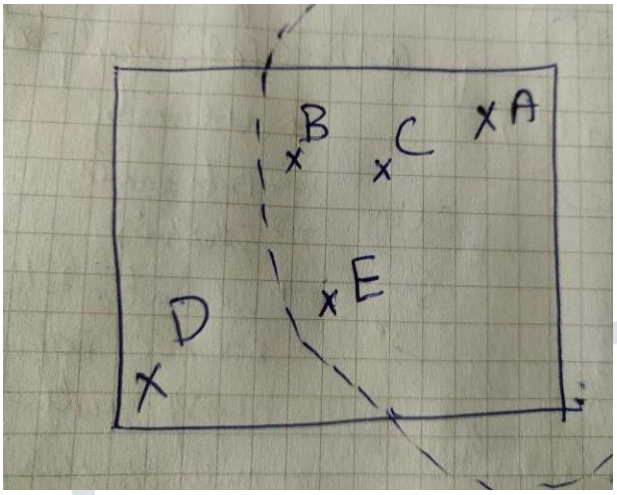
LOF Formula:

$$LOF(x) = \frac{\sum \frac{LRD_k(o)}{LRD_k(x)}}{N_k(x)}$$

LRD of each point is used to compare with the average LRD of its K neighbours. LOF is the ratio of the average LRD of the K neighbours of x to the LRD of x.

K-distance, it is the distance of a point to its k_th neighbour Let us assume k =3 and we calculate LOF of A. From below

Fig k-distance means 3rd closet distance from A i.e., E (consider it has greater distance among all other).



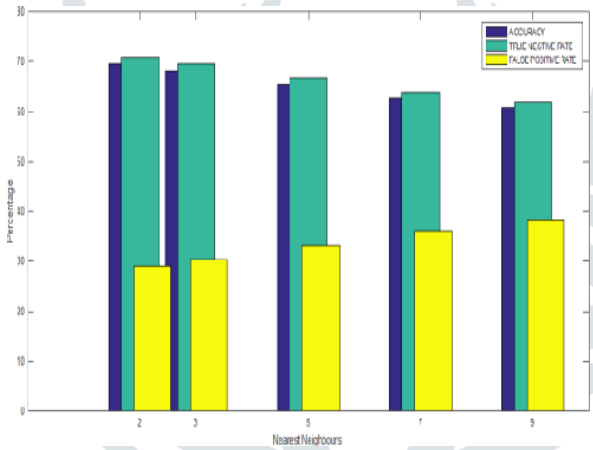


Fig 3: The graph of Accuracy, True Negative Rate, False Positive Rate

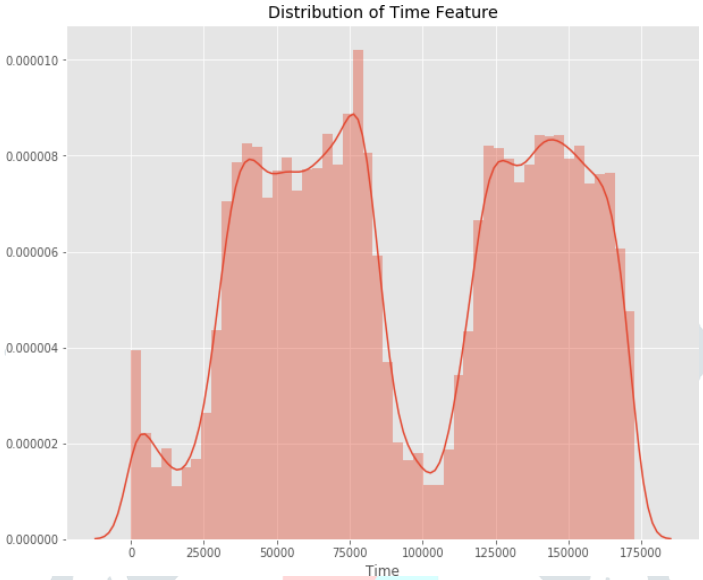
II. Isolated Forest:

A newer technique used to detect anomalies is called Isolation Forests. The algorithm works because anomalies are data points that are far from regular ones. As a result, anomalies are susceptible to a mechanism called isolation.

The algorithm isolates points by randomly selecting a feature and then split on values between the maximum and minimum values of the feature until the points are isolated from each other. Isolating anomalous observations is easier because only a few generations are needed to separate those cases from the normal observations. On the other hand, isolating normal observations require many more generations. Therefore, an anomaly score is calculated as the path length required to separate a given observation.

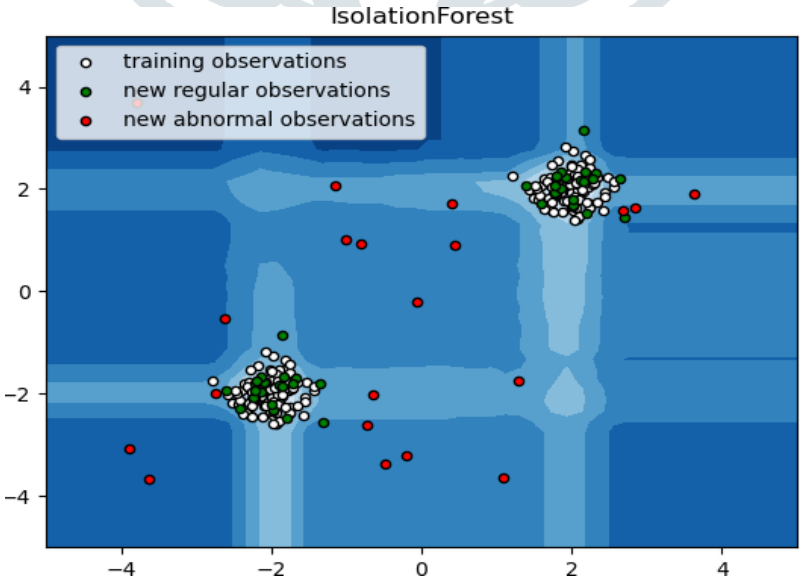
This algorithm has low computation complexity and use less memory. It builds a good performing model with a small number of trees using fixed sub-sample sizes, regardless of the size of a data set

In the Fig4 here the anomalies are highlighted by the red colour points and the normal points are highlighted by the green colour points. The red colour points are the outliers.

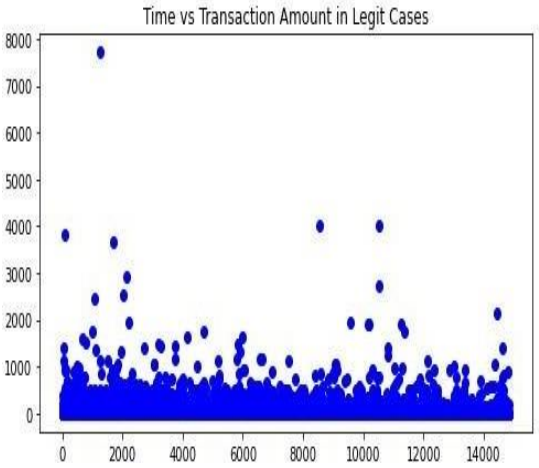


Graph of Distribution of Time Feature:

Fig 4: Isolated Forest



Bubble chart of Isolated Forest*:*



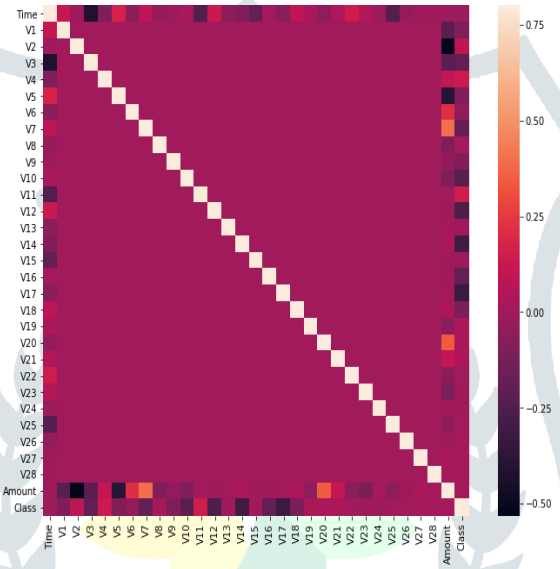


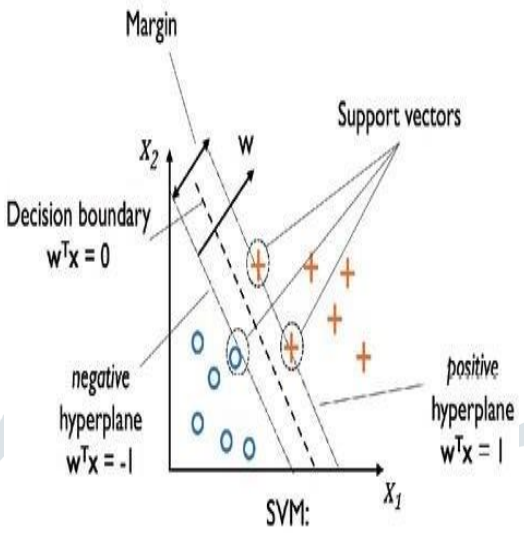
Fig 5: Correlation of Fraud Detection

III. One class SVM:

One Class SVM is unsupervised learning algorithms designed for outlier detection. The model is trained on 'healthy' data. The algorithm learns on the normal transactions and creates a model that contains a representation of the data. When introduced to observations that are far away, it will be labels as outlier and return a negative number. When introduced to observations that are close, it will be labelled as inlier, a positive number.

SVM is a supervised machine learning algorithm which can be used for classification or regression problems. It uses a technique called the kernel trick to transform your data and

then based on these transformations it finds an optimal boundary between the possible outputs.



Working of Support Vector Machine:

Support vector machines focus only on the points that are the most difficult to tell apart, whereas other classifiers pay attention to all of the points. Support vector machines focus only on the points that are the most difficult to tell apart, whereas other classifiers pay attention to all of the points.

In the Fig 4 One class SVM separate's the outlier. The yellow points in the diagram are consider as the anomalies.

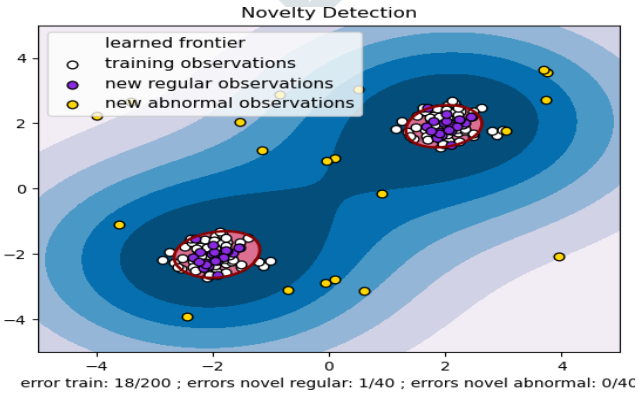


Fig 6. One class SVM

VI. RESULT

We have used the credit card data set for prediction by using the unsupervised algorithms viz, is Isolated Forest, One class SVM, Local Outlier Factor

Table of Algorithms:

Algorithms	Accuracy
Local outlier Factor	87.50%
Isolated Forest	95.76%
One class SVM	70.09%

VII. CONCLUSION

From the table we can see the accuracy of the Isolated Forest has accuracy 95.76%, One class SVM has the accuracy 70.09%, Local Outlier Factor has the accuracy 87.50%.

So, from the all three algorithms Isolated and Local Outlier Factor has the good accuracy than the One class SVM. The performance of Once class SVM is very poor compare to the Isolated Forest and Local outlier Factor. In this paper we have concentrated on the Unsupervised learning algorithm so that we can detect anomalies without the knowing the variables.

Unsupervised algorithm is very useful we can detect the anomalies. We can use Unsupervised algorithms to detect the anomalies to any field. We can detect the outlier or

Comparative chart

Algorithm		Precision	recall	F1 score	support	Accuracy Score	Fraud outlier
Local Outlier Factor	0	1	1	1	28432	87.15%	97
	1	0.02	0.02	0.02	49		
Support Vector Machine	0	1	0.46	0.63	28432	70.09%	15425
	1	0	0.46	0	49		
Isolation Forest	0	1	1	1	4987	95.76%	77
	1	0.22	0.22	0.22	49		

the fraudulent transaction for any field of trans

Result

To predict the eligibility for home, personal, and automobile loans using machine learning, we have used SVM Algorithm to train home loan and personal loan dataset and To train Automobile loan dataset we have used Naive Bayes algorithm. These algorithms learn patterns from historical loan data to make predictions on new loan applications. The result of the prediction system is binary classification indicating whether the applicant is likely to be eligible or not based on their input features.

- Firstly, we did validation of a train dataset. .
- Then we did validation with columns.
- We have import dataset using Pandas library.
- In EDA Part, we saw the shape of dataset for Person Loan (5001,12),Home Loan (615,12) and Automobile Loan(420,10) and No null values in our dataset.
- The result of loan prediction is the assessment or prediction of the likelihood of loan default for a given borrower.
- This result is typically presented as a probability or a binary classification (e.g., "high risk" or "low risk").
- The specific result will depend on the machine learning model and algorithm used for loan prediction.
- The final result of loan prediction aims to minimize the risk of default, ensure responsible lending practices, and make efficient and timely loan decisions.

We performed Algorithms, to predict the accuracy of model

- **Support Vector Machine**

➢ Using SVM algorithm Predict Person Loan Accuracy is : 95.3%.

➢ Using SVM algorithm Predict Home Loan Accuracy is : 83.73%.

- **Naive Bayes**

➢ Using Naive Bayes algorithm Predict Automobile Accuracy is : 91.17%.

Conclusion

➢ By leveraging advanced data analysis techniques and machine learning algorithms, lenders can make informed decisions about loan approvals, interest rates, and loan terms.

➢ This project helped us to learn about the complicated system of the loan prediction system and the best model that can work with this particular project.

➢ However, by developing accurate, efficient, and automated methods for credit assessment and risk prediction, lenders can minimize the risk of default and promote responsible lending practices.

➢ Loan prediction not only benefits lenders by enabling them to make informed lending decisions, but it also benefits borrowers by ensuring fair and transparent loan evaluations.

➢ This system properly and accurately calculates the result.It predicts the loan is approve or reject to loan applicant or customer very accurately.

BiBliography:

1. Amruta S. Aphale and R. Prof. Dr. Sandeep. R Shinde, "Predict Loan Approval in Banking System Machine Learning Approach for Cooperative Banks Loan Approval", International Journal of Engineering Trends and Applications (IJETA), vol. 9, issue 8 2020) .

2.Adyan Nur Alfiyatin, Hilman Taufiq, Ruth Ema Febrita, Wayan Firdaus Mahmudy, 'Modeling House Price Prediction using Regression Analysis and Particle Swarm Optimization': International Journal of Advanced Computer Science and Applications (Vol. 8, No. 10, 2017)

3.Mohamed El Mohadab, Belaid Bouikhalene, Said Safi, 'Predicting rank for scientific research papers using supervised learning'Applied Computing and Informatics 15 (2019) 182–190.

4 . G. Arutjothi, C. Senthamarai, "Prediction of loan status in commercial bank using machine learning classifier", International Conference on Intelligent Sustainable Systems (ICISS), 2017.

5 . Mohamed El Mohadab, Belaid Bouikhalene, Said Safi, "Predicting rank for scientific research papers using supervised learning", Applied Computing and Informatics 15 (2019) 182–190.

6 . Xingyun Lia,Daji Ergub,Di Zhangb,Dafeng Qiub,Ying Caib,Bo Mab, "Prediction of loan default based on multi-model fusion", ITQM 2020 & 2021.

7 . A. Khan, E. Bhadola, A. Kumar and N. Singh, "Loan Approval Prediction Model A Comparative Analysis", AAMS, 2021.

8 . J.M. Chambers. Computational methods for data analysis. Applied Statistics, Wiley, 1(2):1–10, 1077.

9 . Vishal Singh and Ayushman Yadav , "Prediction of Modernized Loan Approval System Based on Machine Learning Approach" IEEE, 2021.

10 . Abhishek Shivanna and Dharma P Agarwal,"Prediction of Defaulters using Machine Learning on Azure ML", International Research Journal Of Engineering And Technology, 2021

Result

To predict the eligibility for home, personal, and automobile loans using machine learning, we have used SVM Algorithm to train home loan and personal loan dataset and To train Automobile loan dataset we have used Naive Bayes algorithm. These algorithms learn patterns from historical loan data to make predictions on new loan applications. The result of the prediction system is binary classification indicating whether the applicant is likely to be eligible or not based on their input features.

- Firstly, we did validation of a train dataset. .
- Then we did validation with columns.
- We have import dataset using Pandas library.
- In EDA Part, we saw the shape of dataset for Person Loan (5001,12),Home Loan (615,12) and Automobile Loan(420,10) and No null values in our dataset.
- The result of loan prediction is the assessment or prediction of the likelihood of loan default for a given borrower.
- This result is typically presented as a probability or a binary classification (e.g., "high risk" or "low risk").
- The specific result will depend on the machine learning model and algorithm used for loan prediction.
- The final result of loan prediction aims to minimize the risk of default, ensure responsible lending practices, and make efficient and timely loan decisions.

We performed Algorithms, to predict the accuracy of model

- **Support Vector Machine**

- Using SVM algorithm Predict Person Loan Accuracy is : 95.3%.

- Using SVM algorithm Predict Home Loan Accuracy is : 83.73%.

- **Naive Bayes**

- Using Naive Bayes algorithm Predict Automobile Accuracy is : 91.17%.

Conclusion

➢ By leveraging advanced data analysis techniques and machine learning algorithms, lenders can make informed decisions about loan approvals, interest rates, and loan terms.

➢ This project helped us to learn about the complicated system of the loan prediction system and the best model that can work with this particular project.

➢ However, by developing accurate, efficient, and automated methods for credit assessment and risk prediction, lenders can minimize the risk of default and promote responsible lending practices.

➢ Loan prediction not only benefits lenders by enabling them to make informed lending decisions, but it also benefits borrowers by ensuring fair and transparent loan evaluations.

➢ This system properly and accurately calculates the result.It predicts the loan is approve or reject to loan applicant or customer very accurately.

BiBliography:

2. Amruta S. Aphale and R. Prof. Dr. Sandeep. R Shinde, "Predict Loan Approval in Banking System Machine Learning Approach for Cooperative Banks Loan Approval", International Journal of Engineering Trends and Applications (IJETA), vol. 9, issue 8 2020) .

2.Adyan Nur Alfiyatin, Hilman Taufiq, Ruth Ema Febrita, Wayan Firdaus Mahmudy, 'Modeling House Price Prediction using Regression Analysis and Particle Swarm Optimization': International Journal of Advanced Computer Science and Applications (Vol. 8, No. 10, 2017)

3.Mohamed El Mohadab, Belaid Bouikhalene, Said Safi, 'Predicting rank for scientific research papers using supervised learning'Applied Computing and Informatics 15 (2019) 182–190.

4 . G. Arutjothi, C. Senthamarai, "Prediction of loan status in commercial bank using machine learning classifier", International Conference on Intelligent Sustainable Systems (ICISS), 2017.

5 . Mohamed El Mohadab, Belaid Bouikhalene, Said Safi, "Predicting rank for scientific research papers using supervised learning", Applied Computing and Informatics 15 (2019) 182–190.

6 . Xingyun Lia,Daji Ergub,Di Zhangb,Dafeng Qiub,Ying Caib,Bo Mab, "Prediction of loan default based on multi-model fusion", ITQM 2020 & 2021.

7 . A. Khan, E. Bhadola, A. Kumar and N. Singh, "Loan Approval Prediction Model A Comparative Analysis", AAMS, 2021.

8 . J.M. Chambers. Computational methods for data analysis. Applied Statistics, Wiley, 1(2):1–10, 1077.

9 . Vishal Singh and Ayushman Yadav , "Prediction of Modernized Loan Approval System Based on Machine Learning Approach" IEEE, 2021.

10 . Abhishek Shivanna and Dharma P Agarwal,"Prediction of Defaulters using Machine Learning on Azure ML", International Research Journal Of Engineering And Technology, 2021