**2. INTRODUCTION**

The term banking can be defined as receiving and protecting money that is deposited by the individual or the entities. This also includes lending money to the people which will be repaid within the given time. Banking sector is regulated in most of the countries as it is the important factor in determining the financial stability of the country. The provision of banking regulation act allows public to obtain loans. Loans are good sum of money borrowed for a period and expected to be paid back at given interest rate. The purpose of the loan can be anything based on the customer requirements. Loans are broadly divided as open-ended and close-ended loans. Open-ended loans are the loans for which the client has approval for a specific amount. Examples of open-end loans are credit cards and a Home equity line of credit. Close-ended loans decreases with each payment. In other words, it is a legal term that cannot be modified by the borrower. Personal loans, mortgages, auto payments, installment loan and student loans are the most common examples of close-ended loans.

Secured or collateral loan are those loans that are protected by an asset. Houses, Vehicles, Savings accounts are the personal properties used to secure the loan. Unsecured loans are also known as personal or signature loans. Here the lender believes that the borrower can repay the loan based on financial resources possessed by the borrower. Liquidity risk is the risk that arises from the lack of marketability of an investment that cannot be bought or sold quickly enough to prevent or minimize a loss

The primary objective of the bank is to provide their wealth in the safer hands. In recent times, banks approve loan after verifying and validating the documents provided by the customer. Yet there is no guarantee whether the applicant is deserving or not. This paper classifies the customers based on certain criteria. Machine learning is an approach to analyze the datasets that summarizes the main characteristics with different methodologies. The purpose of using Machine learning is to uncover the underlying structure of a relatively larger set of variables using various techniques.

**3. INSTALLATIONS**

To build our Machine learning project we require to install certain tools. For our project we had to install Jupyter Notebook, flask in which many other libraries have been installed.

**3.1 Jupyter Notebook**

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more. The Jupyter Notebook can be used to create and share documents that contain live code, equations, visualizations, and text. Jupyter Notebook is maintained by the people at [Project Jupyter](http://jupyter.org/).

Jupyter Notebooks are a spin-off project from the IPython project, which used to have an IPython Notebook project itself. The name, Jupyter, comes from the core supported programming languages that it supports: Julia, Python, and R. Jupyter ships with the IPython kernel, which allows you to write your programs in Python, but there are currently over 100 other kernels that you can also use.

pip3 install jupyter

We can install Jupyter notebook either by pip or else we can directly install by installing Anaconda. Before installing Jupyter we need to ensure that we have latest version of pip.

**3.2 Flask**

Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Extensions are updated far more frequently than the core Flask program.

To install flask firstly we need to do some certain steps:

* Make a Virtual Environment

Pip install virtualenv

* Connect our project with our environment

mkdir myproject

* Set project directory

cd myproject

* Install flask

Pip install flask

After installing the flask then we need to install certain libraries.

**3.2.1 Pandas**

**Pandas** is a Python package providing fast, flexible, and expressive data structures designed to make working with structured (tabular, multidimensional, potentially heterogeneous) and time series data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, **real world** data analysis in Python. Additionally, it has the broader goal of becoming **the most powerful and flexible open source data analysis / manipulation tool available in any language**.

**3.2.2 NumPy**

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

* a powerful N-dimensional array object
* sophisticated (broadcasting) functions
* tools for integrating C/C++ and Fortran code
* useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

**3.2.3 Sklearn**

Scikit-learn is an open source Python library that implements a range of machine learning, pre-processing, cross-validation and visualization algorithms using a unified interface.

**Important features of scikit-learn:**

* Simple and efficient tools for data mining and data analysis. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means, etc.
* Accessible to everybody and reusable in various contexts.
* Built on the top of NumPy, SciPy, and matplotlib.
* Open source, commercially usable.

**3.2.4 XGBoost**

XGBoost is an algorithm that has recently been dominating applied machine learning and Kaggle competitions for structured or tabular data. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance. XGBoost is a software library that you can download and install on your machine, then access from a variety of interfaces. Specifically, XGBoost supports the following main interfaces:

* Command Line Interface (CLI).
* C++ (the language in which the library is written).
* Python interface as well as a model in scikit-learn.
* R interface as well as a model in the caret package.
* Java and JVM languages like Scala and platforms like Hadoop.

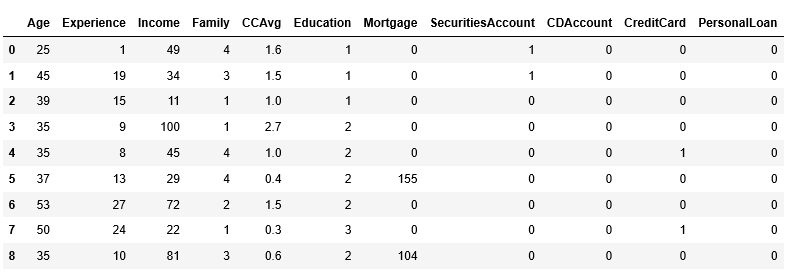
**4. THEORETICAL ANALYSIS**

**4.1 Data preprocessing**

**Data preprocessing** is a **data** mining technique that involves transforming raw **data** into an understandable format. Real-world **data** is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors. **Data preprocessing** is a proven method of resolving such issues.

In Real world **data** are generally incomplete: lacking attribute values, lacking certain attributes of interest, or containing only aggregate **data**. Noisy: containing errors or outliers. Inconsistent: containing discrepancies in codes or names.

Our dataset at the beginning before normalization looks like the one in the figure 4.1.



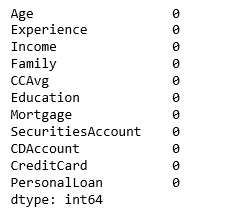
**Figure 4.1: Dataset Before Preprocessing**

**4.1.1 Finding missing values**

The concept of missing values is important to understand in order to successfully [manage](http://www.statisticssolutions.com/academic-solutions/resources/dissertation-resources/data-entry-and-management/multiple-imputation-for-missing-data/) data. If the missing values are not handled properly by the researcher, then he/she may end up drawing an inaccurate inference about the data. Due to improper handling, the result obtained by the researcher will differ from ones where the missing values are present.

dataset.isnull().sum()

Since we have no null values in our dataset, we get figure 4.1.1 as the output.



**Figure 4.1.1: Missing values output**

**4.1.2 See the Categorical Values**

To prevent the categorical variables we use the concept of dummy variables.

What is a dummy variable?

Dummy Variables is one that takes the value 0 or 1 to indicate the absence or presence of some categorical effect that may be expected to shift the outcome.

Instead of having one column here above, we are going to have three column.

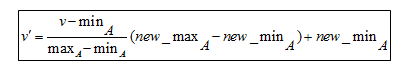
**4.1.3 Normalization**

Normalization is a technique often applied as part of data preparation for machine learning. The goal of normalization is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values. For machine learning, every dataset does not require normalization. It is required only when features have different ranges.

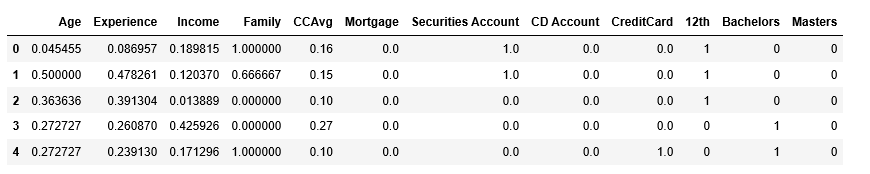
There are different types of normalization techniques **Z Normalization(Standardization), Min-Max Normalization, Unit Vector Normalization. The technique we used is Min-Max Normalization.**

**Min-Max Normalization**

Min-max normalization is one of the most common ways to normalize data. For every feature, the minimum value of that feature gets transformed into a 0, the maximum value gets transformed into a 1, and every other value gets transformed into a decimal between 0 and 1.

****

Our dataset after preprocessing looks like the one in the figure 4.1.3



**Figure 4.1.3: Dataset After Preprocessing**

**4.2 Methodologies**

To train and test our model we used three different types of algorithms. The algorithms we used to train our model are Logistic Regression, Decision Tree Classifier and Random Forest Classifier.

**4.2.1 Logistic regression**

Logistic Regression is a Machine Learning algorithm which is used for the classification problems, it is a predictive analysis algorithm and based on the concept of probability. A Logistic Regression a Linear Regression model but the Logistic Regression uses a more complex cost function, this cost function can be defined as the ‘**Sigmoid function’** or also known as the ‘logistic function’.

**Logistic function**

The [logistic function](https://en.wikipedia.org/wiki/Logistic_function), also called the sigmoid function was developed by statisticians to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment. It’s an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits.

1 / (1 + e^-value)

Where e is the [base of the natural logarithms](https://en.wikipedia.org/wiki/E_(mathematical_constant)) (Euler’s number or the EXP() function in your spreadsheet) and value is the actual numerical value that you want to transform.

Logistic regression uses an equation as the representation, very much like linear regression. Input values (x) are combined linearly using weights or coefficient values (referred to as the Greek capital letter Beta) to predict an output value (y). A key difference from linear regression is that the output value being modeled is a binary values (0 or 1) rather than a numeric value. Below is an example logistic regression equation:

y = e^(b0 + b1\*x) / (1 + e^(b0 + b1\*x))

Where y is the predicted output, b0 is the bias or intercept term and b1 is the coefficient for the single input value (x). Each column in your input data has an associated b coefficient (a constant real value) that must be learned from your training data.

**4.2.2 Decision Tree Classifier**

Decision Tree Analysis is a general, predictive modelling tool that has applications spanning a number of different areas. In general, decision trees are constructed via an algorithmic approach that identifies ways to split a data set based on different conditions. It is one of the most widely used and practical methods for supervised learning. Decision Trees are a non-parametric supervised learning method used for both classification and regression tasks. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

In Decision Tree the major challenge is to identification of the attribute for the root node in each level. This process is known as attribute selection. We have two popular attribute selection measures:

1. Information Gain
2. Gini Index

**InformationGain**  
When we use a node in a decision tree to partition the training instances into smaller subsets the entropy changes. Information gain is a measure of this change in entropy.

**Gini Index**

* Gini Index is a metric to measure how often a randomly chosen element would be incorrectly identified.
* It means an attribute with lower Gini index should be preferred.
* Sklearn supports “Gini” criteria for Gini Index and by default, it takes “gini” value.
  + 1. **Random Forest Classifier**

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes or mean prediction of the individual trees. It creates a set of decision trees from randomly selected subset of training set. It then aggregates the votes from different decision trees to decide the final class of the test object.

**5. IMPLEMENTATION**

**5.1 Machine Learning**

import flask as fk

import pandas as pd

import numpy as np

import sklearn as sk

import xgboost as xg

url = "<https://raw.githubusercontent.com/AddalaTejaswini/DataSet/master/smartbees-loan.csv?token=AJS5HNV2DETXHHISLLW42GS5UDLRI>"

dataset = pd.read\_csv(url)

dataset.head(9)

dataset.isnull().sum()

y=dataset['PersonalLoan']

array = dataset.values

x = array[:,:-1]

y = array[:,10]

from sklearn.preprocessing import MinMaxScaler

x1=array[:,:-4]

scaler = MinMaxScaler(feature\_range=(0, 1))

rescaledX = scaler.fit\_transform(x1)

np.set\_printoptions(precision=3)

x3=x[:,7:]

dummy=pd.get\_dummies(dataset['Education'])

x5=pd.DataFrame(rescaledX,columns=['Age', 'Experience', 'Income', 'Family', 'CCAvg', 'Education', 'Mortgage'])

x6=pd.DataFrame(x3,columns=['Securities Account', 'CD Account', 'CreditCard'])

x6.head()

x4 = pd.merge(x5,x6,right\_index=True, left\_index=True)

x4=x4.drop(['Education'],axis=1)

x7=pd.merge(x4,dummy,right\_index=True, left\_index=True)

dummy.columns=['12th','Bachelors','Masters']

dummy.head()

x7=pd.merge(x4,dummy,right\_index=True, left\_index=True)

x7.head()

arrayRes = x7.values

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(arrayRes,y,test\_size=0.8, random\_state=0)

from sklearn.linear\_model import LogisticRegression

model = LogisticRegression(solver='liblinear')

model.fit(x\_train,y\_train)

predictions = model.predict(x\_test)

from sklearn.metrics import accuracy\_score

print(accuracy\_score(y\_test, predictions))

from sklearn.tree import DecisionTreeClassifier

clf = DecisionTreeClassifier(random\_state = 0)

clf.fit(x\_train, y\_train)

y\_pred = clf.predict(x\_test)

print(accuracy\_score(y\_test, y\_pred))

from sklearn.ensemble import RandomForestClassifier

clf1=RandomForestClassifier(n\_estimators=100)

clf1.fit(x\_train,y\_train)

y\_pred=clf1.predict(x\_test)

print(accuracy\_score(y\_test, y\_pred))

import pickle

with open('loan\_model\_xgboost.pkl', 'wb') as file:

pickle.dump(model, file)

**5.2 HTML and Bootstrap**

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Personal Loan Predictor</title>

<script src="<https://ajax.googleapis.com/ajax/libs/angularjs/1.6.4/angular.min.js>"></script>

<meta name = "viewport" content = "width:device-width, initial-scale = 1">

<link rel = "stylesheet" href = "<http://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css>">

<meta name="viewport"content="width=device-width, initial-scale=1">

 <link rel="[stylesheet"href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css](stylesheet%22href=%22https:/maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css)">

<script src="<https://ajax.googleapis.com/ajax/libs/jquery/3.2.0/jquery.min.js>"></script>

<script src="<https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js>"></script>

</head>

<style>

body {

background:url( <https://images.livemint.com/rf/Image621x414/LiveMint/Period2/2018/05/29/Photos/Processed/pl-kfOI--621x414@LiveMint-kvoC--621x414@LiveMint.jpg>) no-repeat center center fixed;

}

.panel-default {

opacity: 0.9;

margin-top:80px;

}

.form-group.last {

margin-bottom:30px;

}

</style>

<body>

<div class="container">

<div class="row">

<div class="col-md-6 col-md-offset-3">

<div class="panel panel-default">

<div class="panel-heading">

<strong>Personal Loan Prediction Form</strong>

</div>

<div class="panel-body">

<form action="{{ url\_for('main') }}" method="POST" class="form-horizontal" role="form">

<div class="form-group">

<label for="Age" class="col-sm-3 control-label">Age</label>

<div class="col-sm-9">

<input class="form-control" id="Age" placeholder=" Enter Age" required="" type="number" name="Age">

</div>

</div>

<div class="form-group">

<label for="Experience" class="col-sm-3 control-label">Experience</label>

<div class="col-sm-9">

<input class="form-control" id="Experience" placeholder=" Enter Experience" required="" type="number" name="Experience">

</div>

</div>

<div class="form-group">

<label for="Income" class="col-sm-3 control-label">Income</label>

<div class="col-sm-9">

<input class="form-control" id="Income" placeholder=" Enter Income" required="" type="number" name="Income">

</div>

</div>

<div class="form-group">

<label for="Family" class="col-sm-3 control-label">Family</label>

<div class="col-sm-9">

<input class="form-control" id="Family" placeholder=" Enter Family Members" required="" type="number" name="Family">

</div>

</div>

<div class="form-group">

<label for="CCAvg" class="col-sm-3 control-label">CC Average</label>

<div class="col-sm-9">

<input class="form-control" id="CCAvg" placeholder=" Enter CC Average" required="" type="number" name="CCAvg">

</div>

</div>

<div class="form-group">

<label for="Mortgage" class="col-sm-3 control-label">Mortgage</label>

<div class="col-sm-9">

<input class="form-control" id="Mortgage" placeholder=" Enter Mortgage" required="" type="number" name="Mortgage">

</div>

</div>

<div class="form-group">

<label for="SecuritiesAccount" class="col-sm-3 control-label">SecuritiesAccount</label>

<div class="col-sm-9">

<select id="SecuritiesAccount" name="SecuritiesAccount">

<option value="0">No</option>

<option value="1">Yes</option>

</select>

</div>

</div>

<div class="form-group">

<label for="CDAccount" class="col-sm-3 control-label">CDAccount</label>

<div class="col-sm-9">

<select id="CDAccount" name="CDAccount">

<option value="0">No</option>

<option value="1">Yes</option>

</select>

</div>

</div>

<div class="form-group">

<label for="CreditCard" class="col-sm-3 control-label">Credit Card</label>

<div class="col-sm-9">

<select id="CreditCard" name="CreditCard">

<option value="0">No</option>

<option value="1">Yes</option>

</select>

</div>

</div>

<div class="form-group">

<label for="Education" class="col-sm-3 control-label">Education</label>

<div class="col-sm-9">

<select id="Education" name="Education">

<option value="1">12th</option>

<option value="2">Bachelors</option>

<option value="3">Masters</option>

</select>

</div>

</div>

<div class="form-group last">

<div class="col-sm-offset-3 col-sm-9">

<button type="submit" class="btn btn-success btn-sm">

Submit</button>

</div>

</div>

</form>

</div>

</div>

</div>

</div>

<div class="row">

<div class="result" align="center">

{% if result == 1 or result == 0%}

{% for variable, value in original\_input.items() %}

<b>{{ variable }}</b> : {{ value }}

{% endfor %}

<br> Personal Loan Prediction Value:

<p style="font-size:50px">{{ result }}</p>

{% endif %}

­­­­</div>

</div>

</div>

</body>

</html>

**5.3 Flask**

import flask

import pickle

import pandas as pd

import numpy as np

from sklearn.preprocessing import MinMaxScaler

with open('model/loan\_model\_xgboost.pkl', 'rb') as f:

model = pickle.load(f)

app = flask.Flask(\_\_name\_\_, template\_folder='templates')

@app.route('/', methods=['GET', 'POST'])

def main():

x=0

y=0

z=0

if flask.request.method == 'GET':

return(flask.render\_template('main.html'))

if flask.request.method == 'POST':

Age = flask.request.form['Age']

Experience = flask.request.form['Experience']

Income = flask.request.form['Income']

Family = flask.request.form['Family']

CCAvg = flask.request.form['CCAvg']

Mortgage = flask.request.form['Mortgage']

SecuritiesAccount = flask.request.form['SecuritiesAccount']

CDAccount = flask.request.form['CDAccount']

CreditCard = flask.request.form['CreditCard']

Education = flask.request.form['Education']

if(Education=="1"):

x=1

if(Education=="2"):

y=1

if(Education=="3"):

z=1

input\_variables =pd.DataFrame([[Age, Experience, Income, Family, CCAvg, Mortgage, SecuritiesAccount, CDAccount, CreditCard, x, y, z]], columns=['Age', 'Experience', 'Income', 'Family', 'CCAvg', 'Mortgage', 'SecuritiesAccount', 'CDAccount', 'CreditCard', '12th', 'Bachelors', 'Masters'], dtype=float)

prediction = model.predict(input\_variables)[0]

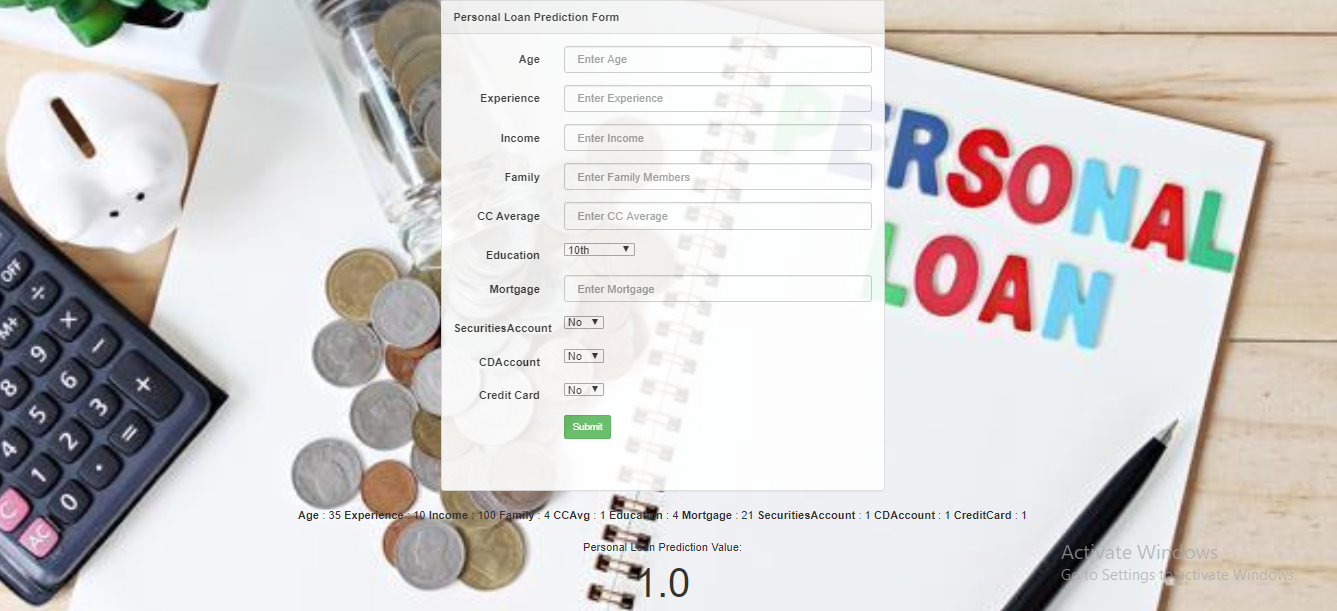
return flask.render\_template('main.html', original\_input={'Age':Age,'Experience':Experience,'Income':Income,'Family':Family,'CCAvg':CCAvg,'Mortgage':Mortgage,'SecuritiesAccount':SecuritiesAccount,'CDAccount':CDAccount,'CreditCard':CreditCard, '12th':x, 'Bachelors':y, 'Masters':z},result=prediction,)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

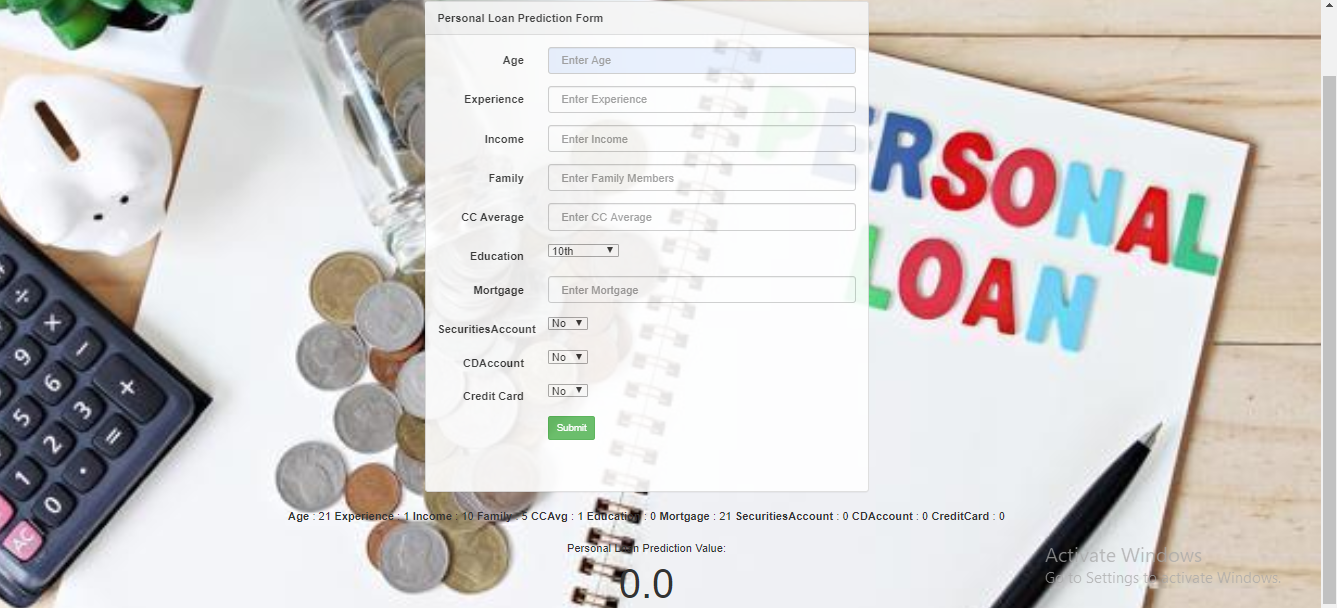
**6. OUTPUT**

The figure 6.1.1 is the output when the loan request gets approved.



**Figure 6.1.1: Loan request is approved**

The figure 6.1.2 is the output when the loan request gets rejected.



**Figure 6.1.2: Loan request is rejected**

**7. CONCLUSION AND FUTURE SCOPE**

The main purpose of the project is to classify and analyze the nature of the loan applicants. From a proper analysis of data set and constraints of the banking sector, three different algorithms have been used to train the model and the most accurate one has been used. This will be of a great use for the banks and also every individual. It will reduce the errors and will make the banks business more effective and will also simultaneously reduce our time.

The present model predicts the output as 1 or 0 further it can be implemented in such a way that the model will also predict the required amount that a bank can provide as loan to an individual based on certain criteria. A new option of uploading required certificates can also be added.

**REFERENCES**

1. <https://towardsdatascience.com/designing-a-machine-learning-model-and-deploying-it-using-flask-on-heroku-9558ce6bde7b>
2. <https://blog.cambridgespark.com/deploying-a-machine-learning-model-to-the-web-725688b851c7>
3. <https://t4tutorials.com/min-max-normalization-of-data-in-data-mining/>
4. <https://analyticsindiamag.com/7-types-classification-algorithms/>
5. <https://towardsdatascience.com/data-pre-processing-techniques-you-should-know-8954662716d6>