**Credit Card Approval Prediction Using IBM Watson Machine Learning**

**1. INTRODUCTION**

* 1. **Overview**

Nowadays, banks receive a lot of applications for issuance of credit cards. Many of them are rejected for many reasons, like high-loan balances, low-income levels, or too many inquiries on an individual’s credit report. Manually analysing these applications is error-prone and a time-consuming process. Luckily, this task can be automated with the power of machine learning and pretty much every bank does so nowadays. In this project, we will build an automatic credit card approval predictor using machine learning techniques, just like the real banks do.

In this project, we will be using regression algorithms such as Decision tree, Random forest, KNN, and xgboost. We will train and test the data with these algorithms. From this the best model is selected and saved in pkl format. We will be doing flask integration and IBM deployment.

* 1. **Purpose**

The main purpose of this project is to predict credit card by analysing individual’s credit report

In this project, we will be using regression algorithms such as Decision tree, Random forest, KNN, and xgboost. We will train and test the data with these algorithms. From this the best model is selected and saved in pkl format. We will be doing flask integration and IBM deployment.

**2. LITERATURE SURVEY**

**2.1 Existing Problem**

In the existing system most of the works are done manually. And Credit cards have become a part and parcel of our financial routine. They not only bring in convenience but also help tide over short-term crunches. . There are multiple ways to [apply for a credit card](https://www.creditmantri.com/credit-card/). You can apply for one online or by visiting the nearest branch. Often, they can be applied through many promotional stalls set up at various events or through the field service officers of the bank. If you submit an online application, a representative of the bank will visit you to collect your documents.

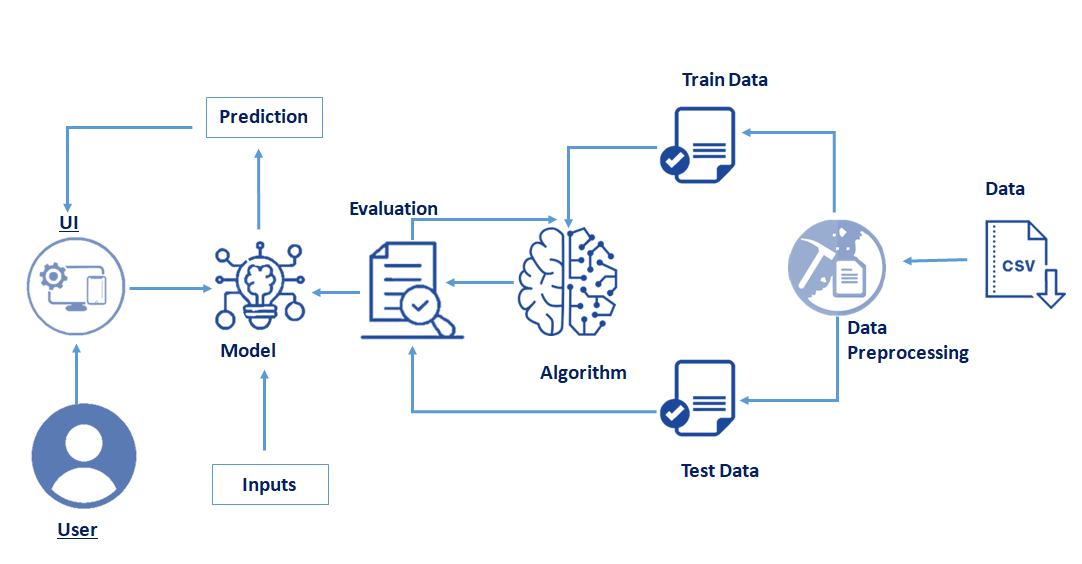
**2.2 Proposed Solution**

The aim of proposed system is to develop a system of improved facilities. The proposed system can overcome all the limitations of the existing system. The existing system has several disadvantages and many more difficulties to work well. The proposed system tries to eliminate the difficulties and it will be less time consuming. And Credit card approval can be beneficial for organisations that lend credit cards, and due to increase in a huge number of the applicant, there is need to automate the task and classify the applicants into if they are eligible for a credit card or not. This helps to avoid organisation losses by avoiding potential defaulters. Here we are not just looking into bank balance but into there personal attributes like gender, married, age, Occupation etc. We account for these personal attributes to evaluate if the given applicant is a good customer. This can also help cut down the weeks-long process into a few days. This gives benefit by cutting down costs on credit analysis and faster credit decisions. Here we are using Credit Approval Data Set to train the model.

In this project, we will be using regression algorithms such as Decision tree, Random forest, KNN, and xgboost. We will train and test the data with these algorithms. From this the best model is selected and saved in pkl format. We will be doing flask integration and IBM deployment.

**3. THEORITICAL ANALYSIS**

**3.1 Block Diagram**

**3.2 Hardware / Software Designing**

**Software**

**Anaconda Navigator** : Anaconda Navigator is a free and open-source distribution of the Python and R programming languages for data science and machine learning-related applications. It can be installed on Windows, Linux, and macOS.Conda is an open-source, cross-platform, package management system. Anaconda comes with so very nice tools like JupyterLab, Jupyter Notebook,QtConsole, Spyder, Glueviz, Orange, Rstudio, Visual Studio Code. For this project, we will be using a Jupyter notebook and Visual Studio Code.

**Python Packages**

**NumPy :** NumPy is a Python package that stands for 'Numerical Python. It is the core library for scientific computing, which contains a powerful n-dimensional array of objects.

**Pandas :** Pandas is a fast, powerful, flexible, and easy-to-use open-source data analysis and manipulation tool, built on top of the Python programming language[.](https://pandas.pydata.org/)

**scikit-learn :** Scikit-learn is an open source data analysis library, and the gold standard for Machine Learning (ML) in the Python ecosystem.

**Matplotlib :** Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible. Create publication quality plots. Make interactive figures that can zoom, pan, update.

**SciPy :** SciPy stands for Scientific Python. It provides more utility functions for optimization, stats and signal processing. Like NumPy, SciPy is open source so we can use it freely. SciPy was created by NumPy's creator Travis Olliphant.

**Seaborn :** Seaborn is a library that uses Matplotlib underneath to plot graphs. It will be used to visualize random distributions.

**Flask :** Web framework used for building Web applications.

**Hardware**

* **Device Name  :** LAPTOP-T9PK1LO2
* **Processor       :** AMD Ryzen 5 2500U with Radeon Vega Mobile Gfx 2.00 GHz
* **System Type  :** 64-bit operating system, x64-based processor

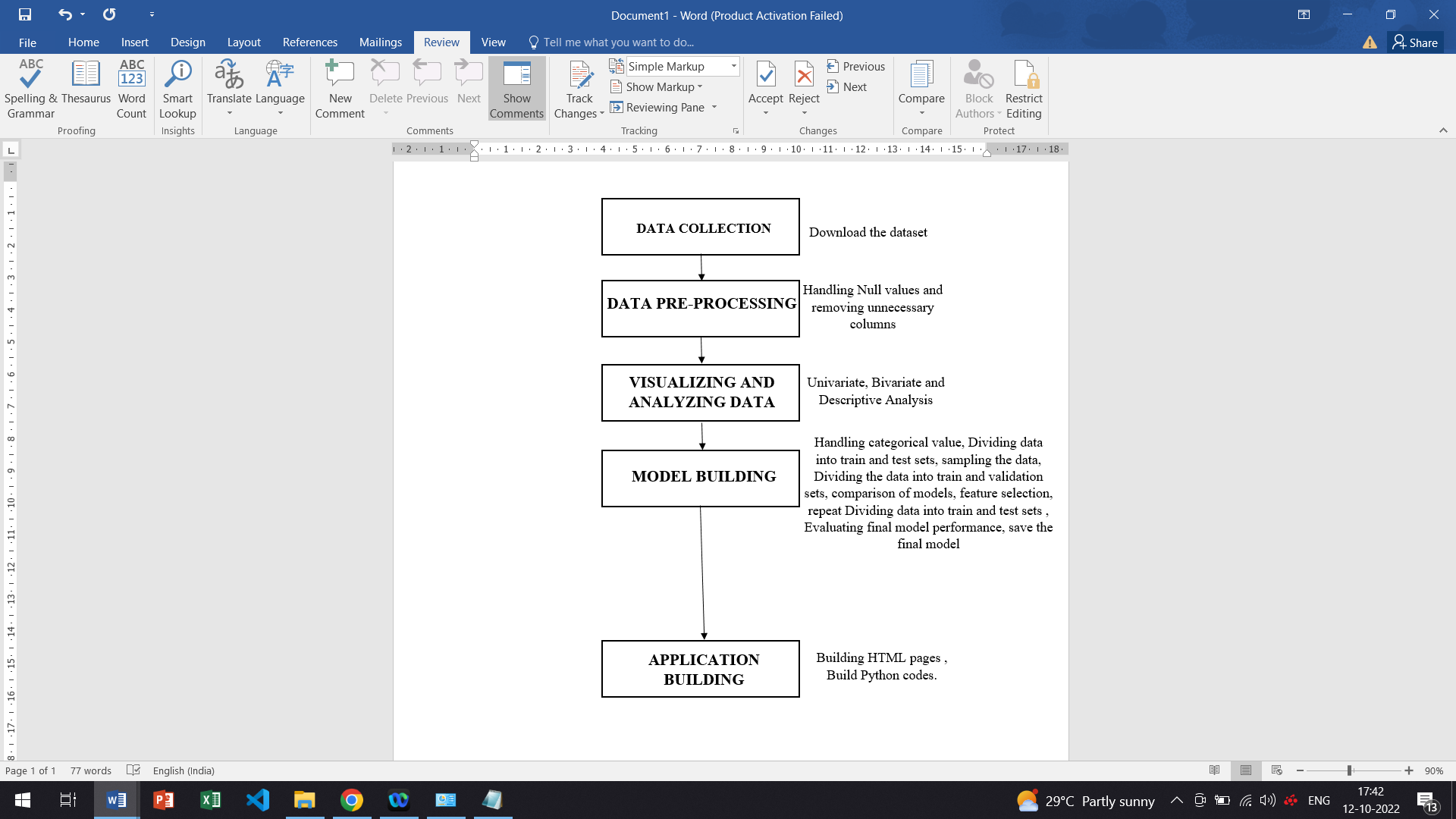
**4. EXPERIMENTAL INVESTIGATIONS**

There are many popular open sources for collecting the data. Eg: kaggle.com, As the dataset is downloaded. Let us read and understand the data properly with the help of some visualization techniques and some analysing techniques. The download data set is not suitable for training the machine learning model as it might have so much randomness so we need to clean the dataset properly in order to fetch good results. Generally, applicant ids are unique in nature. But in our dataset we found some of the ids are repeating multiple times. To handle this we have to remove the duplicate rows. Drop duplicates() function from pandas is used to remove the duplicate rows For checking the null values, df.isnull() function is used. To sum those null values we use sum() function to it. mean() function is used to find the impact of null values in features. In this process, we are going to combine two inter-related columns. Our dataset have some negative values. Those negative values are converted into absolute values. Feature mapping is used on some categorical columns. A function data\_cleaning() is defined. A column is created by adding number of family members with number of childrens.

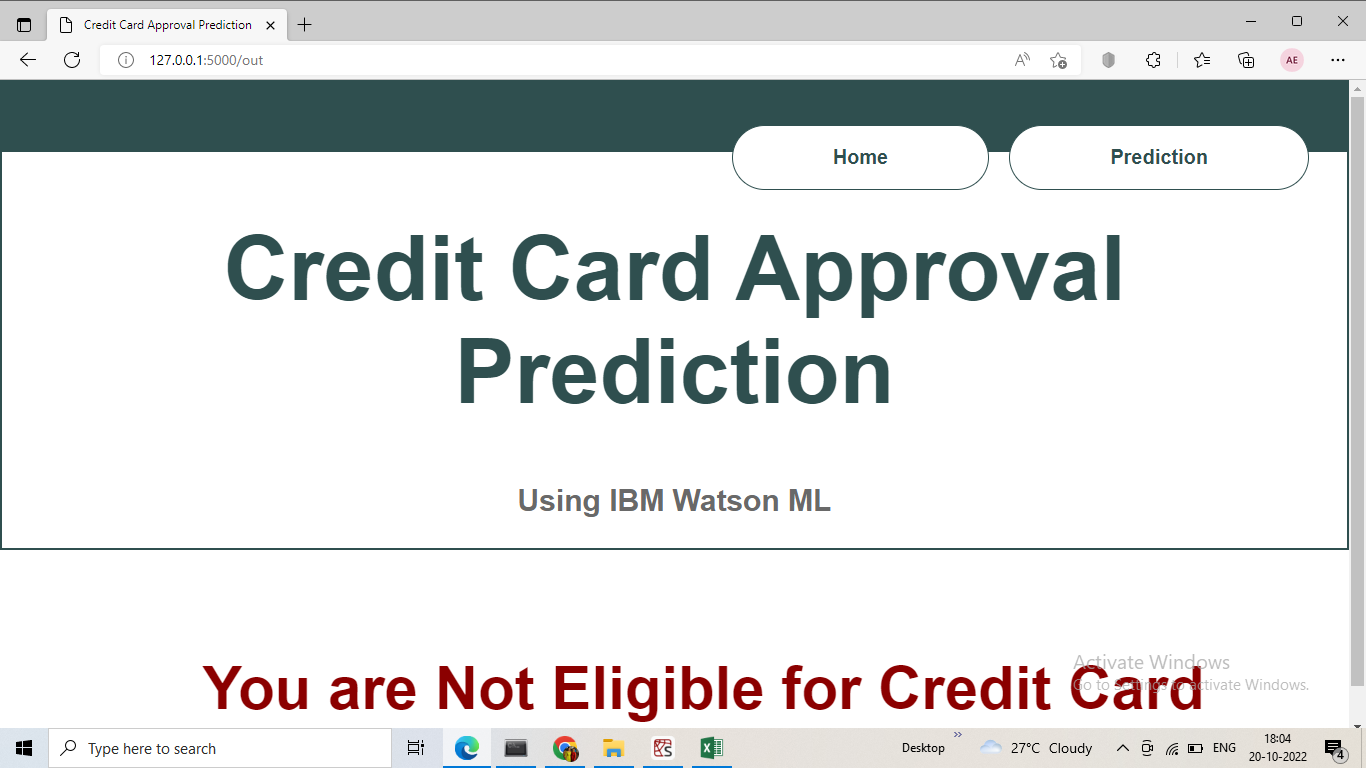
Six unwanted columns are dropped by drop() function. Refer the below image to know the columns name.Days birth and days employed columns have negative values. To convert the negative values to absolute values we use abs() function.

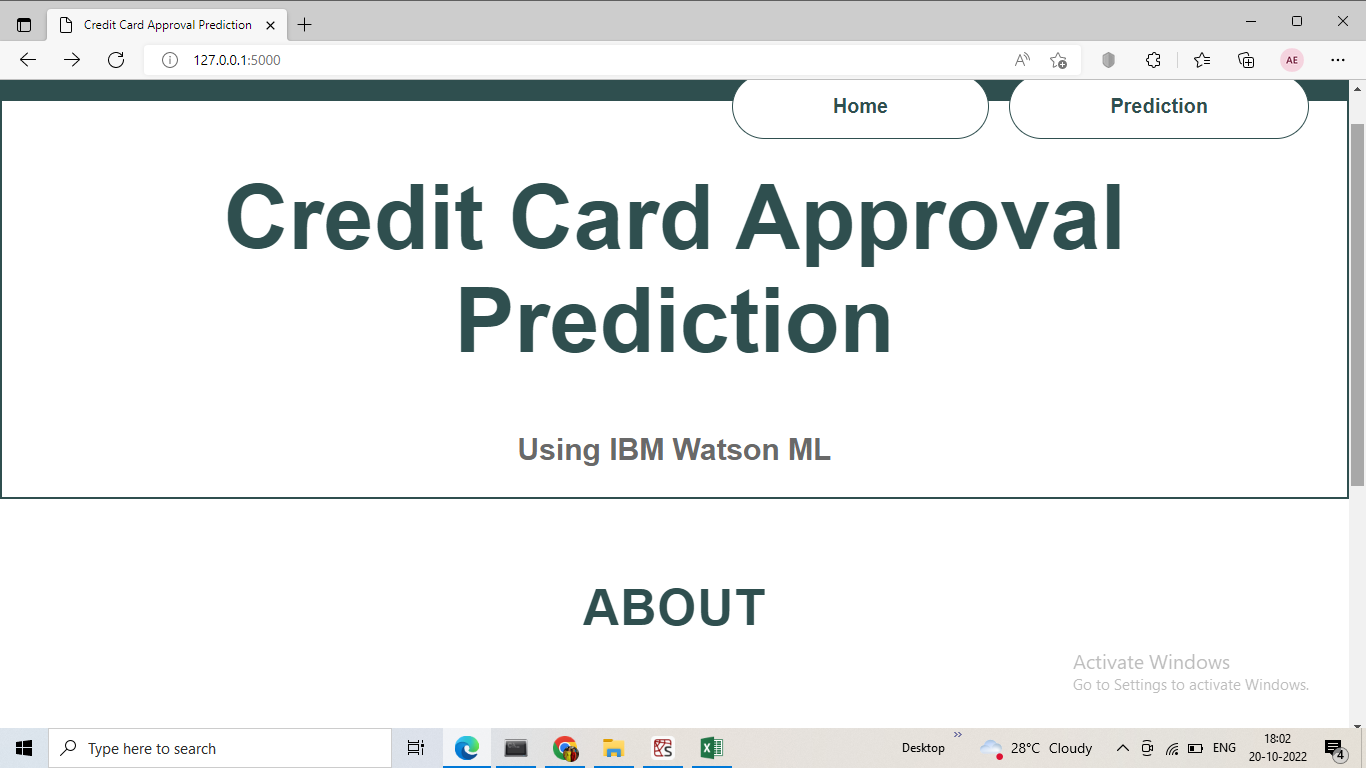
Feature mapping are done in housing type, income type, education type and family type columns. To evaluate the performance confusion matrix and classification report is used. A function named logistic\_reg is created and train and test data are passed as the parameters. Inside the function, LogisticRegression() algorithm is initialized and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, confusion matrix and classification report is done. A function named random\_forest is created and train and test data are passed as the parameters. Inside the function, RandomForestClassifier() algorithm is initialized and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, confusion matrix and classification report is done A function named g\_boosting is created and train and test data are passed as the parameters. Inside the function, GradientBoostingClassifier() algorithm is initialized and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, confusion matrix and classification report is done A function named d\_tree is created and train and test data are passed as the parameters. Inside the function, DecisionTreeClassifier() algorithm is initialized and training data is passed to the model with .fit() function. Test data is predicted with .predict() function and saved in new variable. For evaluating the model, confusion matrix and classification For comparing the above four models compareModel function is defined After calling the function, the results of models are displayed below as an output. From these four model we found decision tree model performs well. Accuracy of decision tree model is 99%. So, this model is saved and used on flask integration.Decision tree model is saved by pickle.dump() function. It saves the model as .pkl file. we will be building a web application that is integrated to the model we built. A UI is provided for the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI.

**5. FLOWCHART**



**6. RESULT**





The Project will predict whether the customer or user is eligible or not for credit card.

**7. ADVANTAGES AND DISADVANTAGES**

**Advantages**

* Helps to provide better quality of services to customers.
* Easy to access
* Easy to predict whether a customer is eligible or not

**Disadvantage**

* May not be that much familiar for normal people.
* Initial Implementation cost

**8. APPLICATIONS**

* Banking sector

**9. CONCLUSIONS**

The main purpose of this project is to predict whether a person who is eligible for credit card or not based on multiple factors.  This Project presents a comparative study ,Those techniques are Regression Algorithm, KNN, Decision tree, Random Forest, and XGBoost.

**10. FUTURE SCOPE**

Once you have submitted all the necessary documents, the bank starts the process of verifying the authenticity of your documents. They will check all the documents and may also call and talk to the references provided in your application. Your employer may also be approached to check for the correctness of the details that you have provided. Also, the verification may be expedited if you operate a salary or loan account of a fixed deposit with that bank.so in this project we focus about the prediction of credit card approval.

**11.BIBLIOGRAPHY**

* **Application of Machine Learning in credit card approval Prediction** [**https://bmcmedresmethodol.biomedcentral.com/articles/10.1186/s12874-021-01284-z**](https://bmcmedresmethodol.biomedcentral.com/articles/10.1186/s12874-021-01284-z)

* [**https://www.w3schools.com/css/css\_rwd\_templates.asp**](https://www.w3schools.com/css/css_rwd_templates.asp)

* [**https://smartinternz.com/**](https://smartinternz.com/)

**APPENDIX**

**app.py**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

dt = pd.read\_csv(r"C:\Users\jagad\Dropbox\PC\Downloads\DATASET\application\_record.csv")

df = pd.read\_csv(r"C:\Users\jagad\Dropbox\PC\Downloads\DATASET\credit\_record.csv")

dt.head()

sns.countplot('OCCUPATION\_TYPE',data=dt)

#handling categorical value

from sklearn.preprocessing import LabelEncoder

a = LabelEncoder()

m['CODE\_GENDER'] = a.fit\_transform(m['CODE\_GENDER'])

m['FLAG\_OWN\_CAR'] = a.fit\_transform(m['FLAG\_OWN\_CAR'])

m['FLAG\_OWN\_REALTY'] = a.fit\_transform(m['FLAG\_OWN\_REALTY'])

m['NAME\_INCOME\_TYPE'] = a.fit\_transform(m['NAME\_INCOME\_TYPE'])

m['NAME\_EDUCATION\_TYPE'] = a.fit\_transform(m['NAME\_EDUCATION\_TYPE'])

m['NAME\_FAMILY\_STATUS'] = a.fit\_transform(m['NAME\_FAMILY\_STATUS'])

m['NAME\_HOUSING\_TYPE'] = a.fit\_transform(m['NAME\_HOUSING\_TYPE'])

#spliting our data

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

x = m[m.drop('target',axis=1).columns]

y = m['target']

x\_train,x\_test,y\_train,y\_test = train\_test\_split(x,y,train\_size=0.75,random\_state=0)

#model building

#logisticRegression

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import classification\_report, confusion\_matrix

def logisticRegression(x\_train, x\_test, y\_train, y\_test):

# creating the model

model = LogisticRegression()

# feeding the training set into the model

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

# predicting the results for the test set

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

# calculating the training and testing accuracies

print('\*\*\*logisticRegression\*\*\*')

print("Training accuracy :", model.score(x\_train, y\_train))

print("Testing accuracy :", model.score(x\_test, y\_test))

# classification report

print(classification\_report(y\_test, y\_pred))

# confusion matrix

print(confusion\_matrix(y\_test, y\_pred))

from sklearn.linear\_model import SGDClassifier

def SGD(x\_train, x\_test, y\_train, y\_test):

# creating the model

model = SGDClassifier(penalty=None)

# feeding the training model into the model

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

# predicting the values for the test set

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

print('\*\*\*Stochastic Gradient Descent Classifier\*\*\*')

print("Training accuracy :", model.score(x\_train, y\_train))

print("Testing accuracy :", model.score(x\_test, y\_test))

# classification report

print(classification\_report(y\_test, y\_pred))

# confusion matrix

print(confusion\_matrix(y\_test, y\_pred))

from sklearn.svm import SVC

def SVClassifier(x\_train, x\_test, y\_train, y\_test):

# creating the model

model = SVC()

# feeding the training set into the model

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

# predicting the results for the test set

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

# calculating the training and testing accuracies

print('\*\*\*Support Vector Classifier\*\*\*')

print("Training accuracy :", model.score(x\_train, y\_train))

print("Testing accuracy :", model.score(x\_test, y\_test))

# classification report

print(classification\_report(y\_test, y\_pred))

# confusion matrix

print(confusion\_matrix(y\_test, y\_pred))

dt=DecisionTreeClassifier()

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

yPred = dt.predict(x\_test)

import pickle

pickle.dump(dt,open("model.pkl","wb"))

**cred.html**

<!DOCTYPE html>

<html>

<head>

<title> Analysis</title>

<!-- Latest compiled and minified CSS -->

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

<style type="text/css">

.result{

color:black;

margin-top:30px;

margin-bottom:20px;

font-size:25px;

color:red;

}

</style>

</head>

<body>

<div class="container">

<div class="row">

<div class="col-md-3"></div>

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

<div class="page-header">

<h1 style="color:red;">Credit Card Approval Prediction</h1>

</div>

</div>

</div>

</div>

<div class="container">

<div class="row">

<div class="col-md-3"></div>

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

<form id="form" action="/data\_predict" method="POST" enctype="multipart/form-data">

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

<div class="form-group">

<label for="gender" style="color:red;">Gender</label>

<select name="gender" class="form-control" id="gender">

<option value="0">FEMALE</option>

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

</select>

</div>

</div>

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

<div class="form-group">

<label for="car" style="color:red;">Own Car Or NOT</label>

<select name="car" class="form-control" id="car">

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

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

</select>

</div>

</div>

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

<div class="form-group">

<label for="estate" style="color:red;">Own Realstate</label>

<select name="estate" class="form-control" id="estate">

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

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

</select>

</div>

</div>

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

<div class="form-group">

<label for="a\_income" style="color:red;">Total Anual Income</label>

<input type="number" class="form-control" id="a\_income" name="a\_income">

</div>

</div>

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

<div class="form-group">

<label for="income" style="color:red;">Type of Income</label>

<select name="income" class="form-control" id="income">

<option value="0">Pensioner</option>

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

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

</select>

</div>

</div>

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

<div class="form-group">

<label for="edu" style="color:red;">Education</label>

<select name="edu" class="form-control" id="edu">

<option value="1">Higher Education</option>

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

</select>

</div>

</div>

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

<div class="form-group">

<label for="family" style="color:red;">Family Status</label>

<select name="family" class="form-control" id="family">

<option value="0">Married</option>

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

</select>

</div>

</div>

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

<div class="form-group">

<label for="house" style="color:red;">Type of Housing</label>

<select name="house" class="form-control" id="house">

<option value="0">House / apartment</option>

<option value="1">With parents</option>

</select>

</div>

</div>

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

<div class="form-group">

<label for="birth" style="color:red;">Days Birth</label>

<input type="number" class="form-control" id="birth" name="birth" >

</div>

</div>

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

<div class="form-group">

<label for="emp" style="color:red;">Days Employed</label>

<input type="number" class="form-control" id="emp" name="emp" >

</div>

</div>

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

<div class="form-group">

<label for="m\_family" style="color:red;">Family Members</label>

<input type="number" class="form-control" id="m\_family" name="m\_family" >

</div>

</div>

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

<div class="form-group">

<label for="paid" style="color:red;">Emi Paid Off</label>

<input type="number" class="form-control" id="paid" name="paid" >

</div>

</div>

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

<div class="form-group">

<label for="past" style="color:red;">Emi Of Pastdues</label>

<input type="number" class="form-control" id="past" name="past" >

</div>

</div>

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

<div class="form-group">

<label for="loan" style="color:red;">No of Loans</label>

<input type="number" class="form-control" id="loan" name="loan" >

</div>

</div>

<center>

<button type="submit" class="btn btn-default" style="color:red;">Predict</button>

</center>

</form>

</div>

</div>

</div>

<div class="result" id="data\_predict">

<center>

<h3>

{{predictiont}}

</h3></center>

</div>

<!-- Latest compiled and minified JavaScript -->

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

</body>

</html>