

FINANCIAL RISK MANAGEMENT

1. INTRODUCTION

1.1 OVERVIEW

Financial risk management is a practice where , by using financial instruments , the risk of a person in terms of his finance is calculated which in turn helps them to be more manageable with their financial status. Financial risks can be foreign exchange risk, business risk, legal risk, etc.

Financial risk management can be either qualitative or quantitative .Based on these, it helps people to manage their costly exposures to risk.

1.2 PURPOSE

The main purpose of this Financial risk management is , as the name suggests ,to be able to identify one's financial status and know the risk one might face regarding his finance which is based on various things like gender, age , the type of housing, profession , savings, checking accounts, etc.

All these details, when considered give the probability of the risk a person might face regarding his or her financial status.

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

People, these days face a lot of financial crisis. These can be either in the form of

healthcare costs, debts or loans from banks or individuals, college or academic expenses, renting a house , or high costs of living . Many don't know how to manage all these finance related problems and hence face financial risks be it in business, marketing, education, etc.

Learning to manage your money is like overcoming a big hurdle in your life. The younger generations also face these risks in forms of student loans, or taking other loans when just beginning their career. People don't realize and don't come to a conclusion about their financial status and hence fall into the risk.

2.2 PROPOSED SOLUTION

By applying Machine Learning algorithms to the financial risk management dataset, which takes the input values of age, gender, job, housing, credit amount, etc., the risk is calculated and hence concluded whether the risk is good or bad for that particular details of a person. This is done by using various algorithms such as Decision Tree Classifier, Naïve Bayes, Logistic Regression, Support Vector Machine. Out of these, SVM was proved to be the best to train the model, having an accuracy of 70%.

3. THEORITICAL ANALYSIS

3.1 BLOCK DIAGRAM

The following is the block diagram which summarizes the entire procedure involved in the Model Building of the Financial Risk Management. It basically shows the entire step-wise procedure from the beginning that is collection of the dataset to the building

of the web page and running the model.

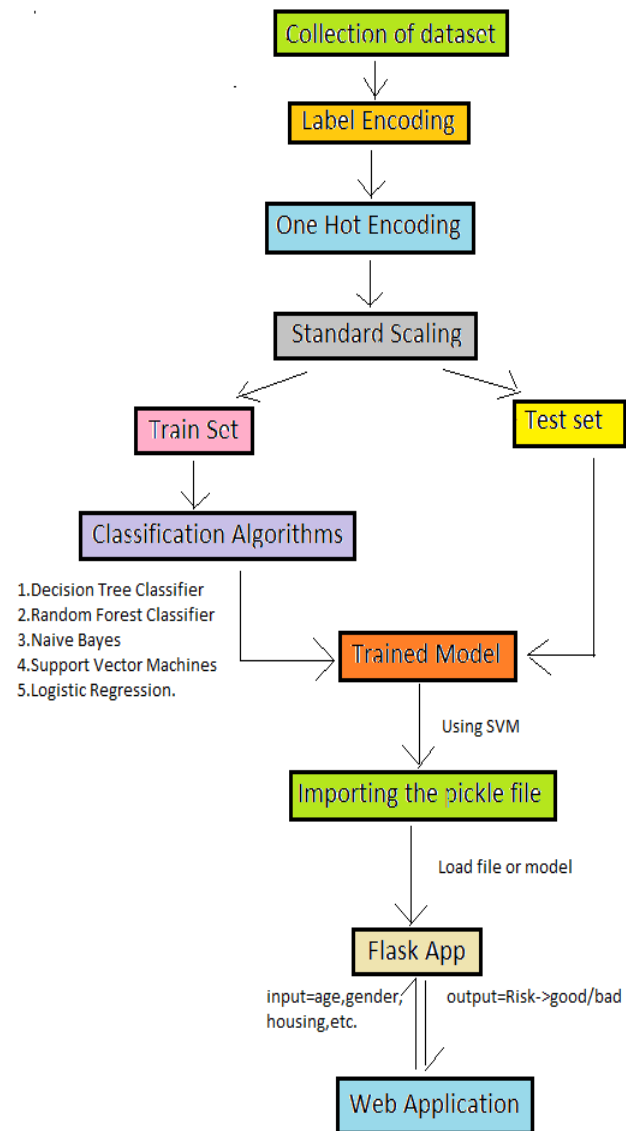


Fig: Block Diagram for Financial Risk Management

3.2 HARDWARE / SOFTWARE DESIGNING

- Hardware
 1. Laptop with 4GB RAM
 2. 64-bit operating system
 3. i7
 4. Hard disk

- Software
 1. Python 3.6
 2. Anaconda environment
 3. Jupyter Notebook
 4. Spider IDE

The machine learning algorithm used in this model of Financial Risk Management involves the use of Supervised Learning. In supervised learning, the model that is trained here is under Classification and Regression model.

This model uses Classification because the output has to get segregated into different classes and the inputs which are age, gender, etc. when entered give the desired output. Classification in machine learning is in which the program reads and learns the dataset given to convert it into new classifications or classes.

Given is an example which uses Classification model. Considering an example of the data which is about Heart Disease detection. It is a pure binary classification model

which will have only two classes i.e. either the person has a heart disease or not.

Similarly, our Financial Risk Management model works the same where it is trained with the dataset to understand the required inputs and then the classifier is trained and the one with the best accuracy is considered to give the precise output of a person having his or her financial risk or not.

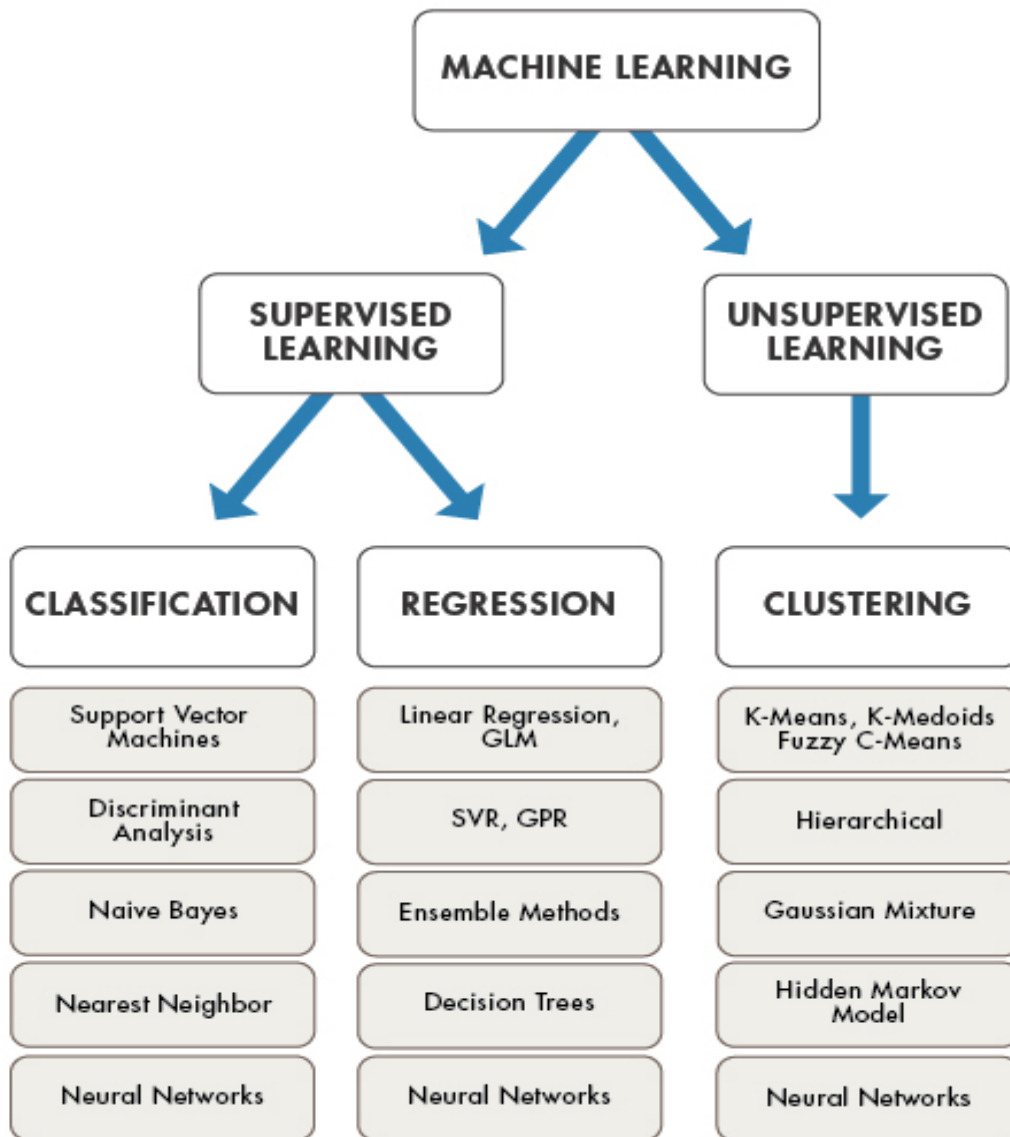


Fig: The classification of Machine Learning

4. EXPERIMENTAL INVESTIGATIONS

- Data Collection

The data for the given topic is collected from the dataset. The dataset contains various categories depending upon the topic. Out of which, some are to be chosen as inputs and the final one as an output i.e. the risk, whether it is good or bad.

- Data Pre-processing

1. Importing the libraries

The first and the foremost step for starting the code is by importing the libraries. The two libraries used here in the first step are pandas to read the dataset and convert it into a DataFrame and the second is NumPy to convert the dataset to an array.

```
In [1]: 1 import pandas as pd  
        2 import numpy as np
```

Fig: Importing the libraries

2. Importing the dataset

The given dataset is imported and then read by using pandas `read_csv()` (which reads the dataset).

The path of the downloaded dataset is given along with the following syntax and then the dataset can be read.

```
In [2]: 1 dataset=pd.read_csv(r"E:\KK\datasets_9109_12699_german_credit_data.csv")

In [3]: 1 dataset
        2
```

Out[3]:

	Unnamed: 0	Age	Sex	Job	Housing	Saving accounts	Checking account	Credit amount	Duration	Risk
0	0	67	male	2	own	NaN	little	1169	6	good
1	1	22	female	2	own	little	moderate	5951	48	bad
2	2	49	male	1	own	little	NaN	2096	12	good
3	3	45	male	2	free	little	little	7882	42	good
4	4	53	male	2	free	little	little	4870	24	bad
...
995	995	31	female	1	own	little	NaN	1736	12	good
996	996	40	male	3	own	little	little	3857	30	good
997	997	38	male	2	own	little	NaN	804	12	good
998	998	23	male	2	free	little	little	1845	45	bad
999	999	27	male	2	own	moderate	moderate	4576	45	good

Fig: Reading the dataset

3. Taking care of the missing values

The dataset is checked if it has any missing values in any of it's classes which are indicated by NaN (Not a Number).

```
In [4]: 1 dataset.isnull().any()

Out[4]: Unnamed: 0      False
Age      False
Sex      False
Job      False
Housing  False
Saving accounts    True
Checking account   True
Credit amount    False
Duration          False
Risk             False
dtype: bool
```

Fig: Checking for missing values

The places where there are True indicates that there are NaN values present in those particular columns.

Then these missing values are filled using mean mode or median accordingly.

```
In [5]: 1 dataset['Saving accounts'].fillna(dataset['Saving accounts'].mode()[0],inplace = True)
        2 dataset['Checking account'].fillna(dataset['Checking account'].mode()[0],inplace = True)

In [6]: 1 dataset.isnull().any()

Out[6]: Unnamed: 0      False
        Age           False
        Sex           False
        Job           False
        Housing       False
        Saving accounts False
        Checking account False
        Credit amount  False
        Duration      False
        Risk          False
        dtype: bool
```

Fig: Rechecking the missing values

Then all the columns are checked for the missing values. All Falses show that there are no missing values left in the dataset.

4. Label Encoding

Label Encoding refers to the conversion of textual data into numerical data. All the columns having textual data like gender, housing, risk, etc are converted into numerical data.

```
In [14]: 1 from sklearn.preprocessing import LabelEncoder
        2 le=LabelEncoder()
        3 dataset["Sex"]=le.fit_transform(dataset["Sex"])
        4 dataset["Housing"]=le.fit_transform(dataset["Housing"])
        5 dataset["Saving accounts"]=le.fit_transform(dataset["Saving accounts"])
        6 dataset["Checking account"]=le.fit_transform(dataset["Checking account"])
        7 dataset["Risk"]=le.fit_transform(dataset["Risk"])
```

Fig: Label Encoding

5. Taking input and output from the dataset

From the dataset required columns are chosen as input which effect the output that is the financial risk. Then the selected inputs and output is converted in the form of arrays using '.values'.

```
In [18]: 1 x=dataset.iloc[:,0:8].values  
        2 y=dataset.iloc[:,8:9].values
```

Fig: Splitting into inputs and output

6. Testing and Training the model

Splitting the dataset into train and test. 20% of the dataset is kept for testing and 80% is kept for training.

```
In [26]: 1 from sklearn.model_selection import train_test_split  
        2 x_train,x_test,y_train,y_test= train_test_split(x,y,test_size=0.2,random_state=0)
```

Fig: Training and testing the dataset

7. Standard Scaling

Standard scaling is thus done to the executed data.

```
In [27]: 1 from sklearn.preprocessing import StandardScaler  
        2 sc=StandardScaler()  
        3 x_train=sc.fit_transform(x_train)  
        4 x_test=sc.transform(x_test)
```

Fig: Standard Scaling

- Model Building

1. Building and training the model

The model has to be built using the different types of Classifiers like Decision Tree Classifier, Logistic Regression, Random Forest Classifier, Support Vector Machines, etc. Among these the one with the highest accuracy is taken as the most precise classifier and hence the inputs are given under these.

One of the classifier used here is given below.

```
1 Decision Tree Classifier On our dataset

In [32]: 1 from sklearn.tree import DecisionTreeClassifier
          2 dtc=DecisionTreeClassifier(random_state=5,criterion="entropy")
          3 dtc.fit(x_train,y_train)

Out[32]: DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='entropy',
                                max_depth=None, max_features=None, max_leaf_nodes=None,
                                min_impurity_decrease=0.0, min_impurity_split=None,
                                min_samples_leaf=1, min_samples_split=2,
                                min_weight_fraction_leaf=0.0, presort='deprecated',
                                random_state=5, splitter='best')
```

Fig: Applying Decision Tree Classifier

The accuracy of this decision tree classifier is found

```
In [37]: 1 from sklearn.metrics import accuracy_score
          2 accuracy=accuracy_score(y_test,dtcpred)
          3 accuracy
```

```
Out[37]: 0.585
```

Fig: Finding the accuracy of Decision Tree Classifier

The ROC curve is found and the AUC is also checked using other libraries.

```
In [41]: 1 import matplotlib.pyplot as plt
2 plt.plot(fpr, tpr, "blue", label="AUC=%.2f"%roc_auc)
3 plt.legend(loc="lower right")
4 plt.title("ROC")
5 plt.xlabel("fpr")
6 plt.ylabel("tpr")
```

Out[41]: Text(0, 0.5, 'tpr')

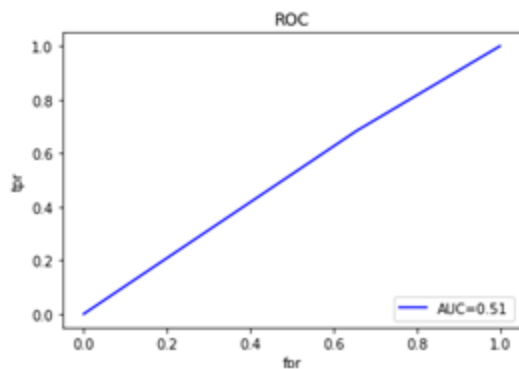


Fig: ROC curve

Since the accuracy of this classifier isn't satisfactory other classifiers are used and the one with best accuracy or AUC is chosen.

Using Support Vector Machines (SVM)

SVM

```
In [77]: 1 from sklearn.svm import SVC
2 svm=SVC(kernel="rbf",random_state=7)
3 svm.fit(x_train,y_train)
```

C:\Users\Acer\anaconda3\lib\site-packages\sklearn\utils\validation.py:760: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
y = column_or_1d(y, warn=True)

Out[77]: SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0, decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf', max_iter=-1, probability=False, random_state=7, shrinking=True, tol=0.001, verbose=False)

Fig: Applying SVM on to the model

Then the accuracy of SVM is obtained and the dataset is predicted using random values from the dataset.

```
In [80]: 1 svmacc=accuracy_score(y_test,svmpred)
          2
In [81]: 1 svmacc
Out[81]: 0.7
In [86]: 1 svm_p= svm.predict(sc.fit_transform([[61,1,1,1,3,0,3059,12]]))
In [87]: 1 svm_p
Out[87]: array([1])
```

Fig: Finding the accuracy of SVM and predicting

The ROC curve is also obtained using the respective libraries and AUC is also checked.

After saving the model, we pickle the file by importing the respective library into the precise Classifier with a high accuracy.

```
In [33]: 1 import pickle
          2 pickle.dump(svm,open("svmrisk.pkl",'wb'))
```

Fig: Dumping the model in pickle file

INTEGRATING A WEBSITE WITH THE MACHINE LEARNING MODEL:

1. Load your pkl file:

```
1 from flask import Flask , render_template , request
2 app = Flask(__name__) # interface between by server and my application wsgi
3 import pickle
4 from sklearn.preprocessing import StandardScaler
5 model = pickle.load(open('svmrisk.pkl','rb'))
6 scaler = pickle.load(open("scalar.pkl","rb"))
7 @app.route('/') # bind to an url
```

Fig: Loading pickle file

2. Build Flask App:

Create a folder and then create templates folder and save the HTML file in it as index.html

In the main folder we should have our app.py and pkl file

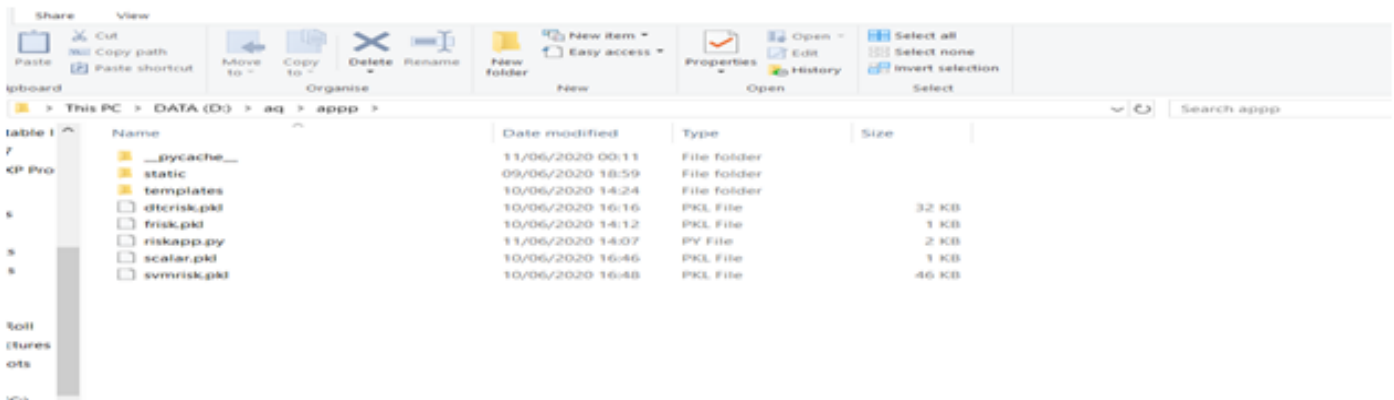


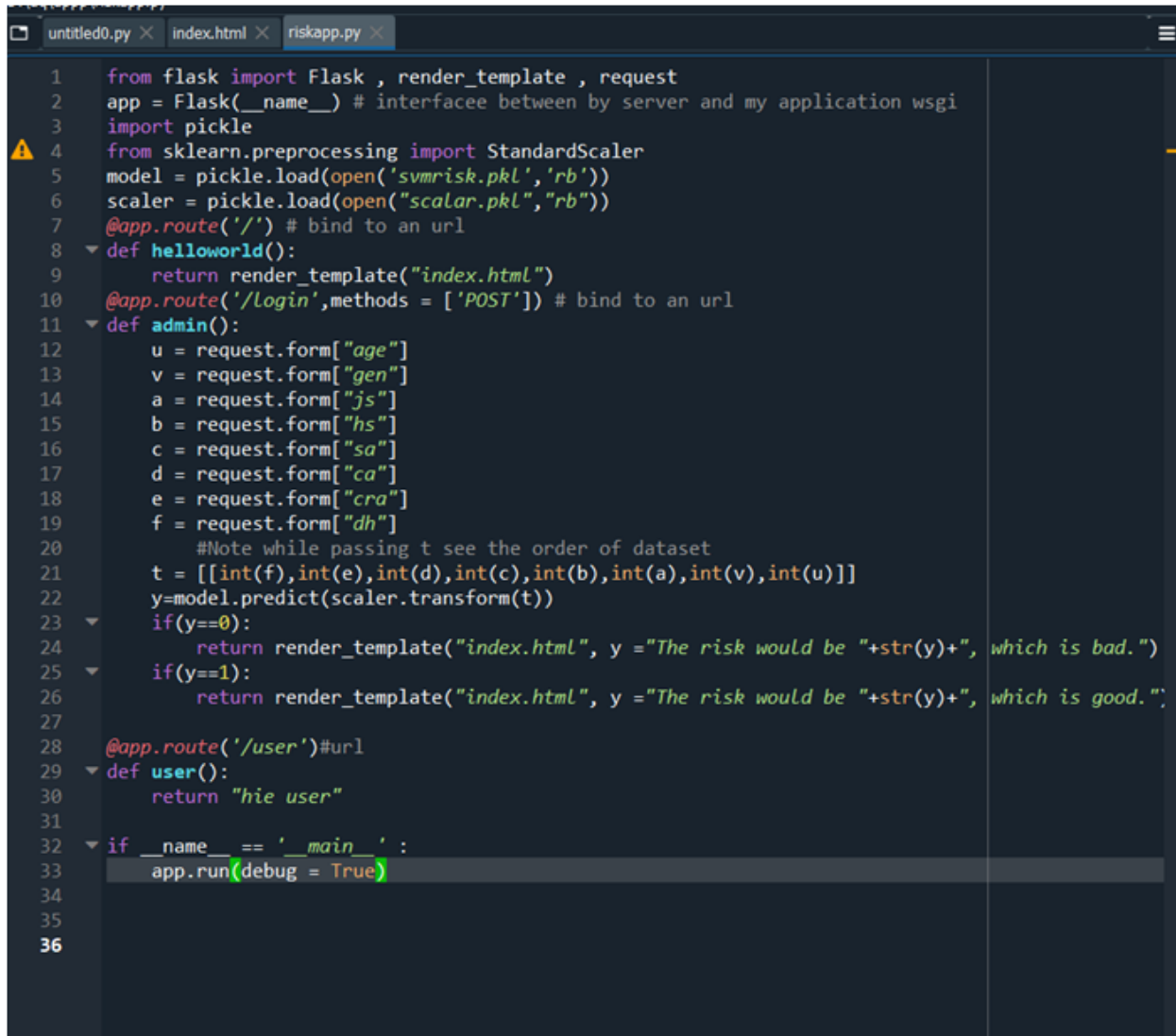
Fig: Folder containing the .pkl and .py file

In the spyder app we are applying css3 to our HTML file, along with the HTML code for our UI as shown:

```
7   font-family:verdana;
8   margin: 0 auto;
9   width: 280px;
10  margin-left:75px;
11  margin-top:10px;
12  }
13  input[type=text]{background-color:rgba(250, 250, 210,0.5);font-size:25px;border-bottom: 1px
14  label{font-size:25px;font-family='Times New Roman';font-weight:bold;margin-bottom:40px;}
15  select{font-size:20px;font-family='Times New Roman';margin-top:10px;}
16 </style>
17 <body>
18 <div style='margin:25% 30%;'>
19 <head>
20 <div style='color:white;font-size:20px;text-align:center;'>
21 <p><h2>Financial Risk Management</h2></p>
22 </div>
23 <form action = "/Login" method = "post" style='background-color:rgba(255,255,255,0.7);padding:10px;'>
24 <p><label style='color:Crimson;'>Enter Age: </label><br>
25 <input type = "text" name = "age" /></p>
26 <p><label style='color:Crimson;'>Gender: </label><br>
27 <input type = "text" name = "gen" /></p>
28 <p><label style='color:DarkGreen;'>Enter your Job Status:</label> <br>
29 <input type = "text" name = "js" /></p>
30 <p><label style='color:Crimson;'>Housing Type: </label><br>
31 <input type = "text" name = "hs" /></p>
32 <p><label style='color:DarkBlue;'>Saving Account:</label><br>
33 <input type = "text" name = "sa" /></p>
34 <p><label style='color:Purple;'>Checking Account:</label><br>
35 <input type = "text" name = "ca" /></p>
36 <p><label style='color:DarkBlue;'>Credit Amount:</label><br>
37 <input type = "text" name = "cra" /></p>
38 <p><label style='color:DarkBlue;'>Duration In Hours:</label><br>
39 <input type = "text" name = "dh" /></p>
40 <p> <input type = "submit" value = "Submit" style='background-color:rgba(255,255,255,0.7);font-size:20px;' /> </p>
41 <b>{{y}}</b>
42 </form>
43 <div style='color:Navy;font-size:20px;text-align:center;'>
44 </div>
```

Fig: HTML code embedded with CSS

And we are using the following python code:



```
1  from flask import Flask , render_template , request
2  app = Flask(__name__) # interface between by server and my application wsgi
3  import pickle
4  from sklearn.preprocessing import StandardScaler
5  model = pickle.load(open('svmrisk.pkl','rb'))
6  scaler = pickle.load(open("scalar.pkl","rb"))
7  @app.route('/') # bind to an url
8  def helloworld():
9      return render_template("index.html")
10 @app.route('/login',methods = ['POST']) # bind to an url
11 def admin():
12     u = request.form["age"]
13     v = request.form["gen"]
14     a = request.form["js"]
15     b = request.form["hs"]
16     c = request.form["sa"]
17     d = request.form["ca"]
18     e = request.form["cra"]
19     f = request.form["dh"]
20     #Note while passing t see the order of dataset
21     t = [[int(f),int(e),int(d),int(c),int(b),int(a),int(v),int(u)]]
22     y=model.predict(scaler.transform(t))
23     if(y==0):
24         return render_template("index.html", y ="The risk would be "+str(y)+", which is bad.")
25     if(y==1):
26         return render_template("index.html", y ="The risk would be "+str(y)+", which is good.")
27
28 @app.route('/user')#url
29 def user():
30     return "hie user"
31
32 if __name__ == '__main__':
33     app.run(debug = True)
34
35
36
```

Fig: Python code for running the model

5. FLOWCHART

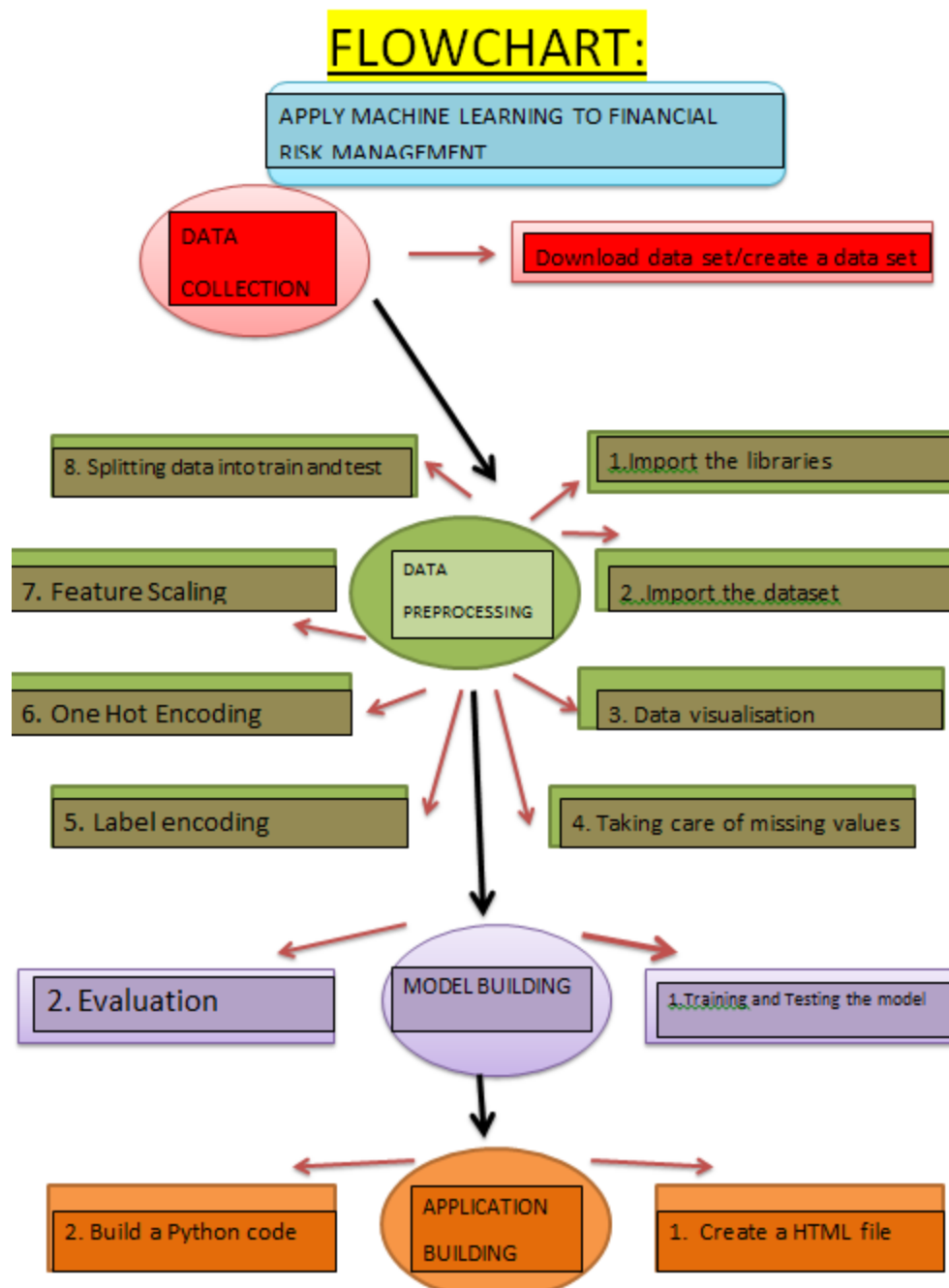


Fig: Flow chart showing the model building process

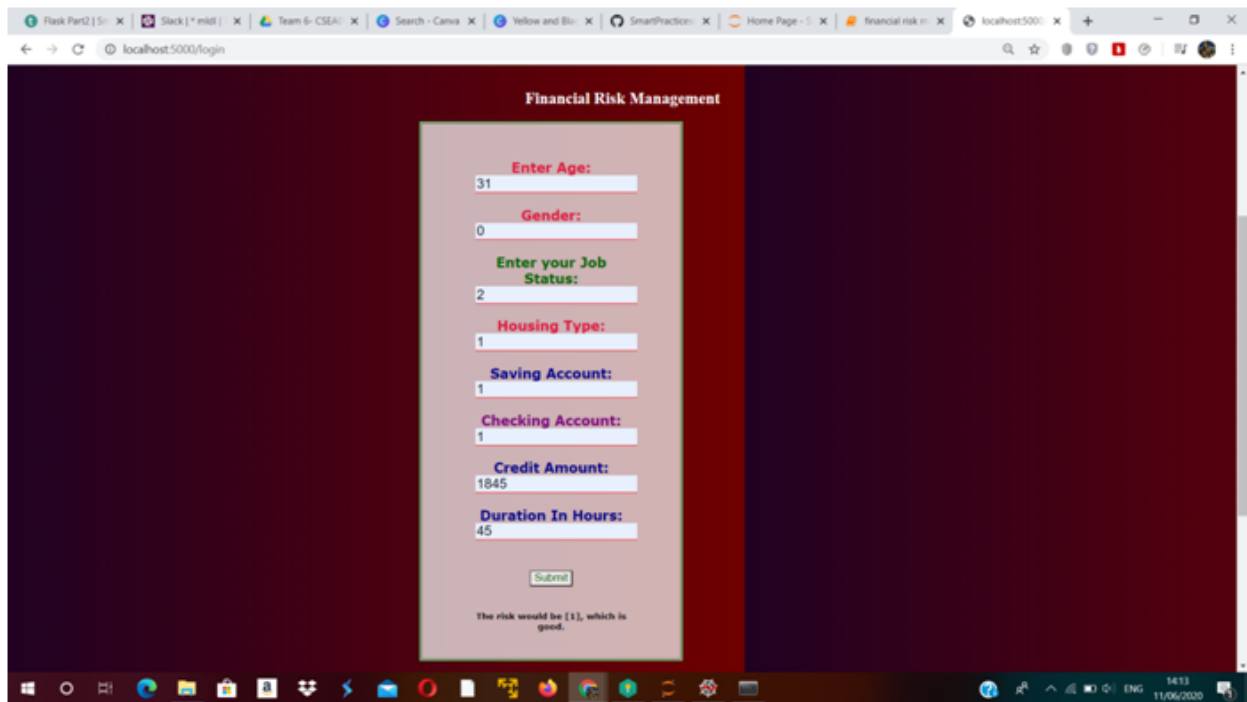
6. RESULT

This project collects the data of around 1000 people and gives out to losses or risk and to protect the value of its assets.

Machine learning is applied to calculate the risk the company might have to face due to any particular person.

For this, the project uses various technical entities such as the anaconda prompt, jupyter notebook, spyder.

Here we show you one output that we derived on the basis of the details entered by the customer:



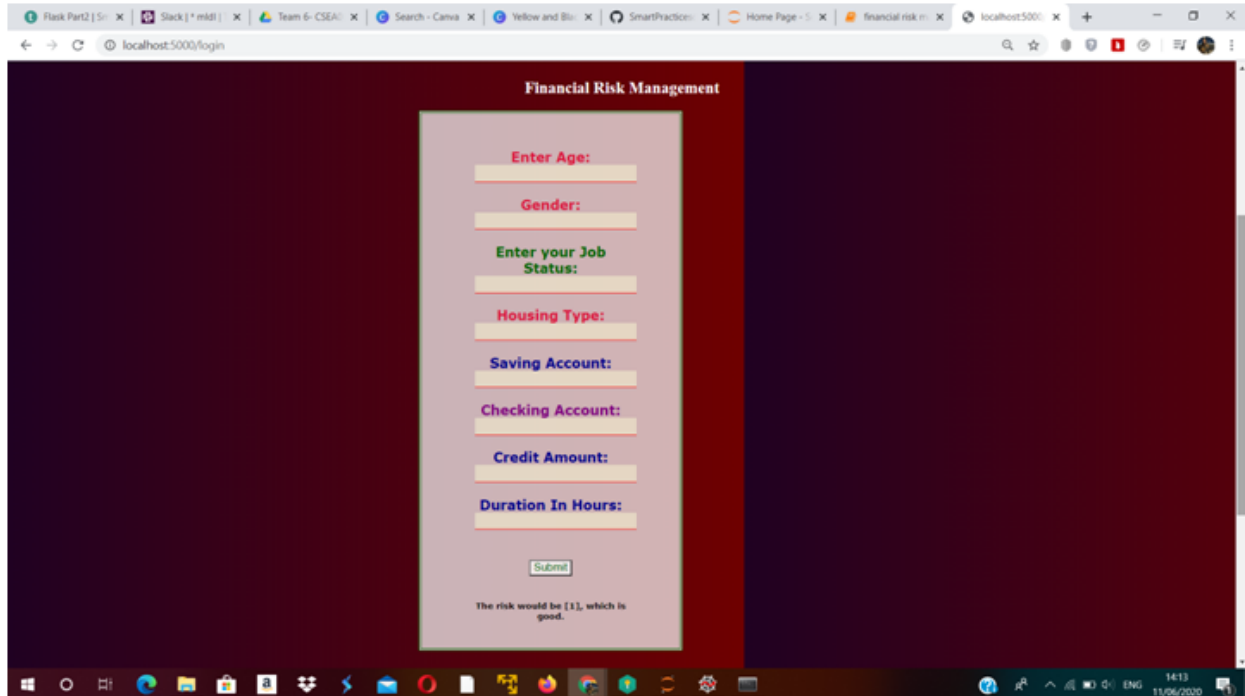
The screenshot displays a web browser window with the URL 'localhost:5000/login'. The page has a dark red background and a central white form titled 'Financial Risk Management'. The form contains the following input fields and values:

- Enter Age: 31
- Gender: 0
- Enter your Job Status: 2
- Housing Type: 1
- Saving Account: 1
- Checking Account: 1
- Credit Amount: 1845
- Duration In Hours: 45

Below the input fields is a 'Submit' button. At the bottom of the form, the output is displayed: 'The risk would be [1], which is good.'

Fig: Input given to the model

The output of this information was calculated and given as:



The screenshot displays a web browser window with the URL 'localhost:5000/login'. The page has a dark red background. In the center, there is a light pink rectangular form titled 'Financial Risk Management'. The form contains several input fields with labels: 'Enter Age:', 'Gender:', 'Enter your Job Status:', 'Housing Type:', 'Saving Account:', 'Checking Account:', 'Credit Amount:', and 'Duration In Hours:'. Each label is followed by a text input field. Below these fields is a green 'Submit' button. At the bottom of the form, it says 'The risk would be [1], which is good.' The browser's taskbar at the bottom shows various icons and the system clock indicating 14:13 on 11/06/2020.

Fig: Output obtained

The output is given as 1, which means the risk is good. Once the basic empirical results have been obtained, decisions have to be made about the appropriate risk-management techniques to use for managing the risks.

7. ADVANTAGES

- 1.This model helps people to know their financial status and also helps them know if they are at a risk or not
2. It takes in age, gender, housing etc. and helps customers to know if they are financially stable or not
3. It secures the financial position of the company by helping them know which

customer may pose a risk to the company.

DISADVANTAGES:

1. This model only takes age, gender, job and housing etc as an input. This may serve as a disadvantage as detailed information is not being inserted
2. The age has to be changed into 1 for male and 0 for female. Customers might not know this and hence it may cause difficulties.

8. APPLICATIONS

These days everything is being managed on a digital scale. From education to shopping to jobs, the internet has opened vast opportunities for everybody. A lot of business transactions are also made online. This creates a need to manage the finances technically on a very large scale in a short period of time. Hence the management of finances using machine learning is the most convenient way to do so. These methods are used in banks to calculate what risk a customer may pose to the bank's financial management. It is used in large, small and multi business companies to manage their finances easily and safely. They are used at almost every central and private sectors. Financial management for any start up is necessary. It is necessary. Companies must always keep a track of where all the money is being spent and that is where machine learning comes into picture. Financial management through machine learning opens many windows of opportunities to the world and helps small start ups flourish.

9. CONCLUSIONS

To conclude, our model correctly and precisely predicts the risk the company may have to face in investing in any particular member. The model decides if the risk will be good or bad for the company.

When we type in the inputs of age , gender(0=female,1=male),job, housing, saving

account, checking account , credit amount and duration is hours it will calculate and give us the output as good or bad.

10. FUTURE SCOPE

Artificial Intelligence is a game-changer for risk management in finance as it provides banks and credit unions with tools and AI solutions to identify potential risks and fraud.

The financial crisis of the previous decade gave financial services firms a lot of problems with credit-challenged consumers. Before the digital revolution in financial services industry customer intelligence was based on some relatively simple heuristics, the customer value data was gained through focus groups and surveys of consumer behavior the results of which didn't always correspond the reality.

Today new technologies give businesses access to really tremendous amounts of data about consumers' behavior and needs.

Risk management in banks should use cognitive technologies to gain competitive advantage and use risk to power their organizations' performance.

Artificial intelligence and risk management perfectly align when there is a need for handling and evaluating unstructured data. It is estimated that risk managers of financial institutions will focus on analytics and stopping losses in a proactive manner based on AI findings, rather than spending time in managing the risks inherent in the operational processes.

AI solutions are able to fuel financial institutions with trusted and timely data for building competence around their customer intelligence and successful implementation

of their strategies.

11. BIBLIOGRAPHY

[1] https://en.wikipedia.org/wiki/Financial_risk_management

[2] <https://www.edureka.co/blog/classification-in-machine-learning/>

[3] <https://archer-soft.com/blog/how-ai-changing-risk-management>

APPENDIX

A. CODE

Flask code:

```
from flask import Flask , render_template , request

app = Flask(__name__) # interface between by server and my application wsgi

import pickle

from sklearn.preprocessing import StandardScaler

model=pickle.load(open('svmrisk.pkl','rb'))

scaler=pickle.load(open("scalar.pkl","rb"))

@app.route('/') # bind to an url

def helloworld():

    return render_template("index.html")

@app.route('/login',methods = ['POST']) # bind to an url

def admin():
```

```

u = request.form["age"]

v = request.form["gen"]

a = request.form["js"]

b = request.form["hs"]

c = request.form["sa"]

d = request.form["ca"]

e = request.form["cra"]

f = request.form["dh"]

    #Note while passing t see the order of dataset

t = [[int(f),int(e),int(d),int(c),int(b),int(a),int(v),int(u)]]

y=model.predict(scaler.transform(t))

if(y==0):

    return render_template("index.html", y ="The risk would be "+str(y)+", which is
bad.")

if(y==1):

    return render_template("index.html", y ="The risk would be "+str(y)+", which is
good.")

@app.route('/user')#url

def user():

    return "hie user"

```

```
if __name__ == '__main__':  
    app.run(debug = True)
```

Homepage HTML code:

```
<html>
```

```
<style>
```

```
body{background-image:url('/static/Love.jpg');}
```

```
form{
```

```
    width: 300px;
```

```
    text-align: center;
```

```
font-family:verdana;
```

```
margin: 0 auto;
```

```
width: 280px;
```

```
margin-left:75px;
```

```
margin-top:10px;
```

```
}
```

```
input[type=text]{background-color:rgba(250, 250, 210,0.5);font-size:25px;border-bottom: 1px solid red;border-top:none;border-left:none;border-right:none;margin-bottom:10px}
```

```
label{font-size:25px;font-family='Times New Roman';font-weight:bold;margin-bottom:40px;}
```

```
select{font-size:20px;font-family='Times New Roman';margin-top:10px;}
```

```
</style>
```

```
<body>
```

```
<div style='margin:25% 30%;'>
```

```
<head>
```

```
<div style='color:white;font-size:20px;text-align:center;'>
```

```
<p><h2>Financial Risk Management</h2></p>
```

```
</div>
```

```
<form action = "/login" method = "post" style='background-color:rgba(255,255,255,0.7);padding:50px 100px;border:5px double DarkGreen'>
```

```
<p><label style='color:Crimson;'>Enter Age: </label><br>
```


<input type = "text" name = "age" /></p>

<p><label style='color:Crimson;*>Gender: </label>

<input type = "text" name = "gen" /></p>

<p><label style='color:DarkGreen;*>Enter your Job Status:</label>

<input type = "text" name = "js" /></p>

<p><label style='color:Crimson;*>Housing Type: </label>

<input type = "text" name = "hs" /></p>

<p><label style='color:DarkBlue;*>Saving Account:</label>

<input type = "text" name = "sa" /></p>

<p><label style='color:Purple;*>Checking Account:</label>

<input type = "text" name = "ca" /></p>

<p><label style='color:DarkBlue;*>Credit Amount:</label>

<input type = "text" name = "cra" /></p>

<p><label style='color:DarkBlue;*>Duration In Hours:</label>

<input type = "text" name = "dh" /></p>

<p> <input type = "submit" value = "Submit"

style='background-color:rgba(255,255,255,0.7);font-size:20px;color:DarkGreen;margin:30px;'/></p>

{{y}}

</form>

<div style='color:Navy;font-size:20px;text-align:center;'>

</div>

</div>

<body/>

</html>