Vangala Surya Teja Reddy **50465103**

Loan Default Prediction

Problem Statement
background of the problem

A finance company provides customers with housing loans at reasonable and economical interest rates. It suffered a lot due to loan defaults. The majority of the applicants are not repaying their loans in accordance with the promissory note.

The potential of your project to contribute to your problem domain.
So, we started a new system that identifies loan defaulters based on the data. This system will assist in identifying potential applicants and ensuring a smooth process. We are planning to build a new system to predict the probability of an applicant defaulting on a loan or not in the future.

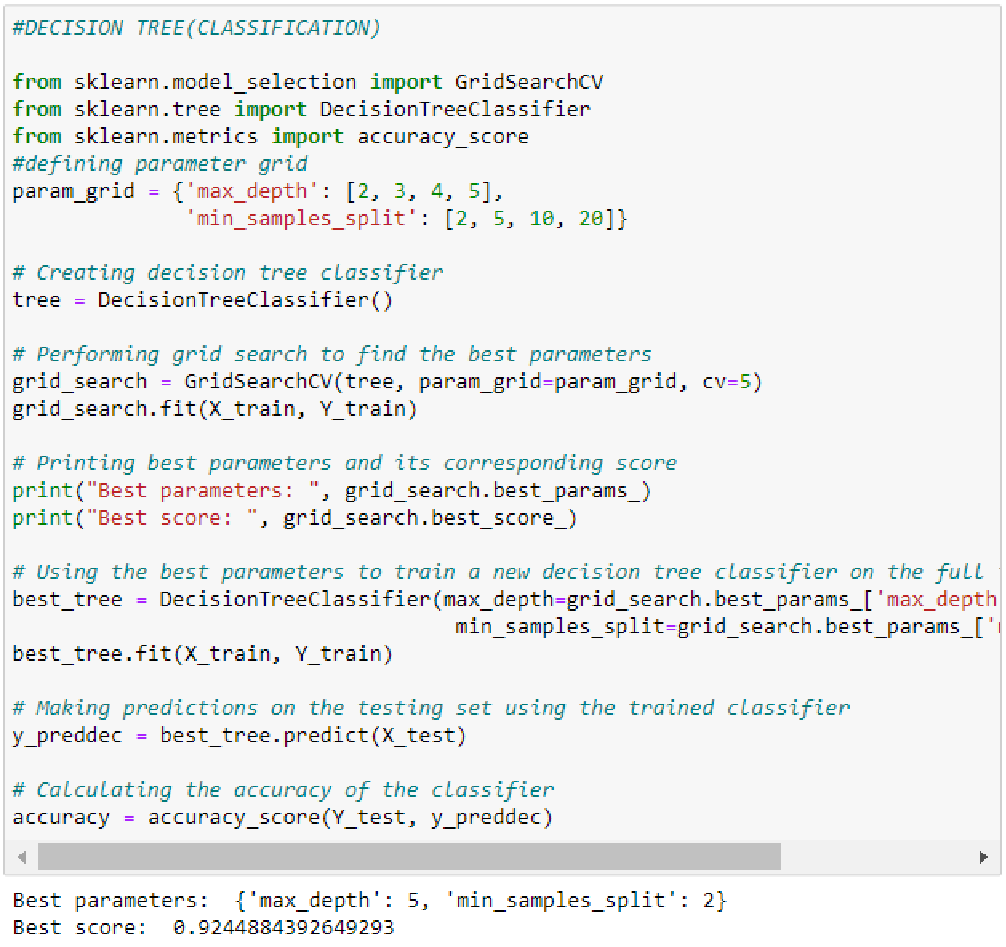
Why is this contribution crucial?
It is very important to build a system that helps to identify the potential applicants who are not paying the loans otherwise the company will incur huge losses.

Data Source
https://www.kaggle.com/datasets/gauravduttakiit/loan-defaultprediction?select=test_4zJg83n.csv

In the last 2 phases we have analyzed the data and made required changes in it and this data was trained and tested using multiple machine learning models. We have checked for the accuracy of these models and have concluded decision tree classification had highest accuracy. In this phase of the project I have built a web application based on the machine learning model which can be used by the bank to decide whether to sanction the loan or to reject it.

1. Machine Learning Model

I have used linear regression, logistic regression, naïve bayes classifier and support vector machine in phase 2. The regression models have not been much efficient for classification and in the rest of the models Decision tree has highest accuracy.
Decision tree classification has performed better for loan defaulters prediction compared to other machine learning models linear regression, logistic regression, support vector machine, and Naive Bayes classifier because Loan default prediction has non-linear relationships between features and target variable which better captures using decision trees. Linear and logistic regressions may not be able to capture the interactions between different features efficiently. Decision trees are robust to outliers in the data and are easy to interpret.



```
#DECISION TREE(CLASSIFICATION)

from sklearn.model_selection import GridSearchCV
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
#defining parameter grid
param_grid = {'max_depth': [2, 3, 4, 5],
              'min_samples_split': [2, 5, 10, 20]}

# Creating decision tree classifier
tree = DecisionTreeClassifier()

# Performing grid search to find the best parameters
grid_search = GridSearchCV(tree, param_grid=param_grid, cv=5)
grid_search.fit(X_train, Y_train)

# Printing best parameters and its corresponding score
print("Best parameters: ", grid_search.best_params_)
print("Best score: ", grid_search.best_score_)

# Using the best parameters to train a new decision tree classifier on the full
best_tree = DecisionTreeClassifier(max_depth=grid_search.best_params_['max_depth
                                    min_samples_split=grid_search.best_params_['
best_tree.fit(X_train, Y_train)

# Making predictions on the testing set using the trained classifier
y_preddec = best_tree.predict(X_test)

# Calculating the accuracy of the classifier
accuracy = accuracy_score(Y_test, y_preddec)
```

```
Best parameters:  {'max_depth': 5, 'min_samples_split': 2}
Best score:  0.9244884392649293
```

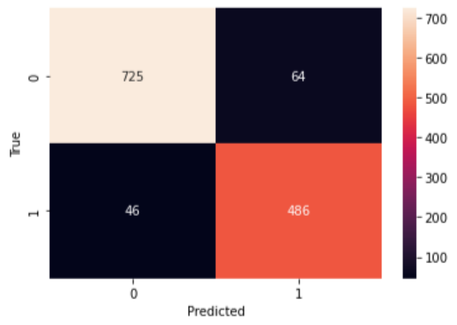
I have checked the accuracy for different depths and sample split sizes and the best accuracy of 92.448 has been for max depth of 5 and samples split value 2.

2. Working of Machine Learning Code

As I have already built the machine learning algorithm, I will be using it in the web application of predictor.

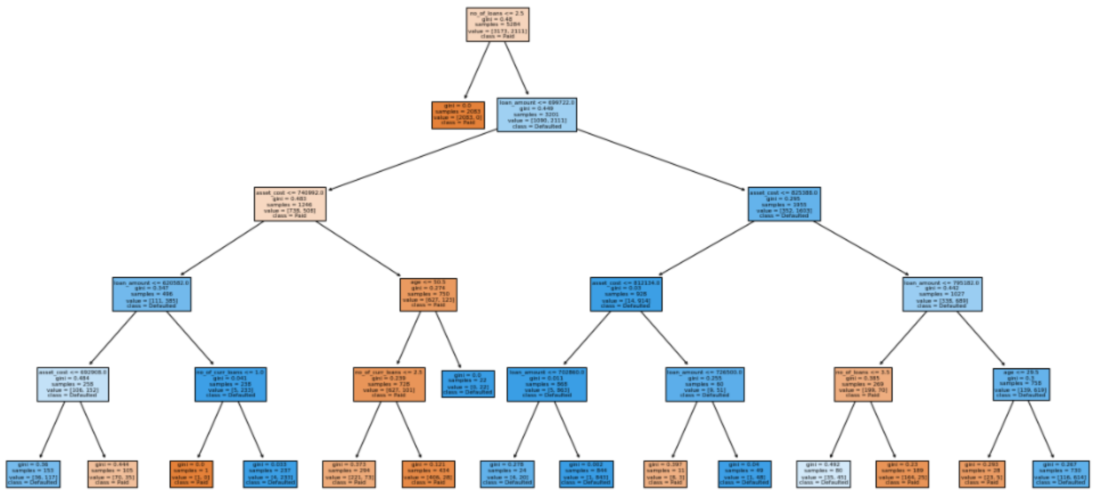
```
In [21]: #confusion matrix
         from sklearn.metrics import confusion_matrix
         cm_dt= confusion_matrix(Y_test, y_preddec)
         cm_dt

Out[21]: array([[725,  64],
                [ 46, 486]], dtype=int64)
```

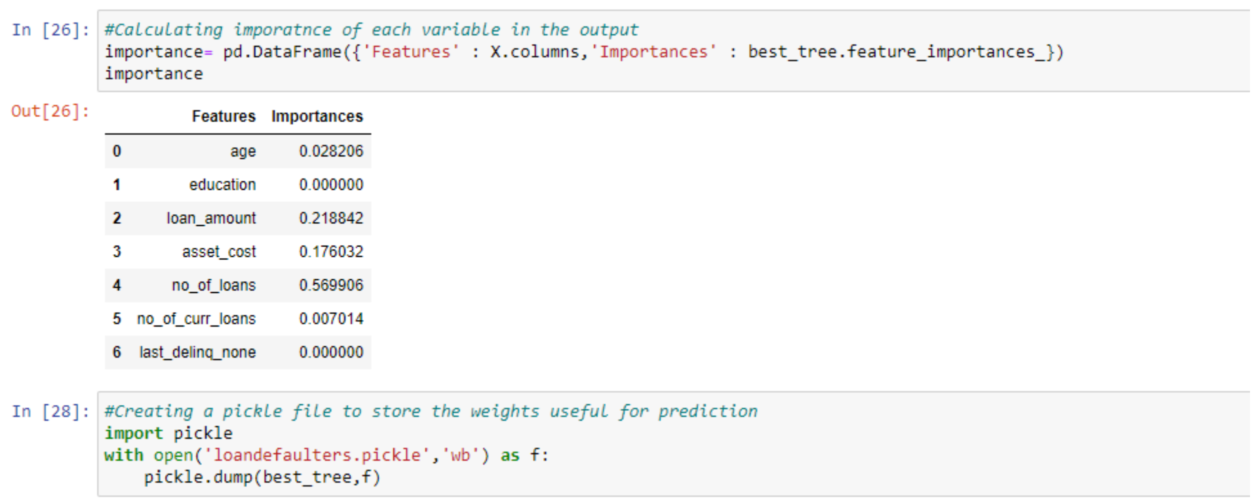
```
In [22]: import matplotlib.pyplot as plt
         sns.heatmap(cm_dt, annot=True, fmt='g')
         plt.xlabel('Predicted')
         plt.ylabel('True')
         plt.show()
```



Printing the diagram of decision tree

```
In [25]: #Tree diagram
         from sklearn.tree import plot_tree
         plt.figure(figsize=(20,10))
         plot_tree(best_tree, filled=True, feature_names=loan_data.columns[:-1], class_names=['Paid','Defaulted'])
         plt.show()
```





```
In [26]: #Calculating imporatnce of each variable in the output
         importance= pd.DataFrame({'Features' : X.columns,'Importances' : best_tree.feature_importances_})
         importance
```

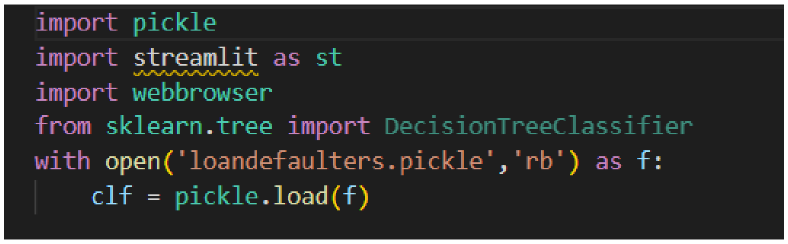
Out[26]:

	Features	Importances
0	age	0.028206
1	education	0.000000
2	loan_amount	0.218842
3	asset_cost	0.176032
4	no_of_loans	0.569906
5	no_of_curr_loans	0.007014
6	last_delinq_none	0.000000

```
In [28]: #Creating a pickle file to store the weights useful for prediction
         import pickle
         with open('loandefaulters.pickle','wb') as f:
             pickle.dump(best_tree,f)
```

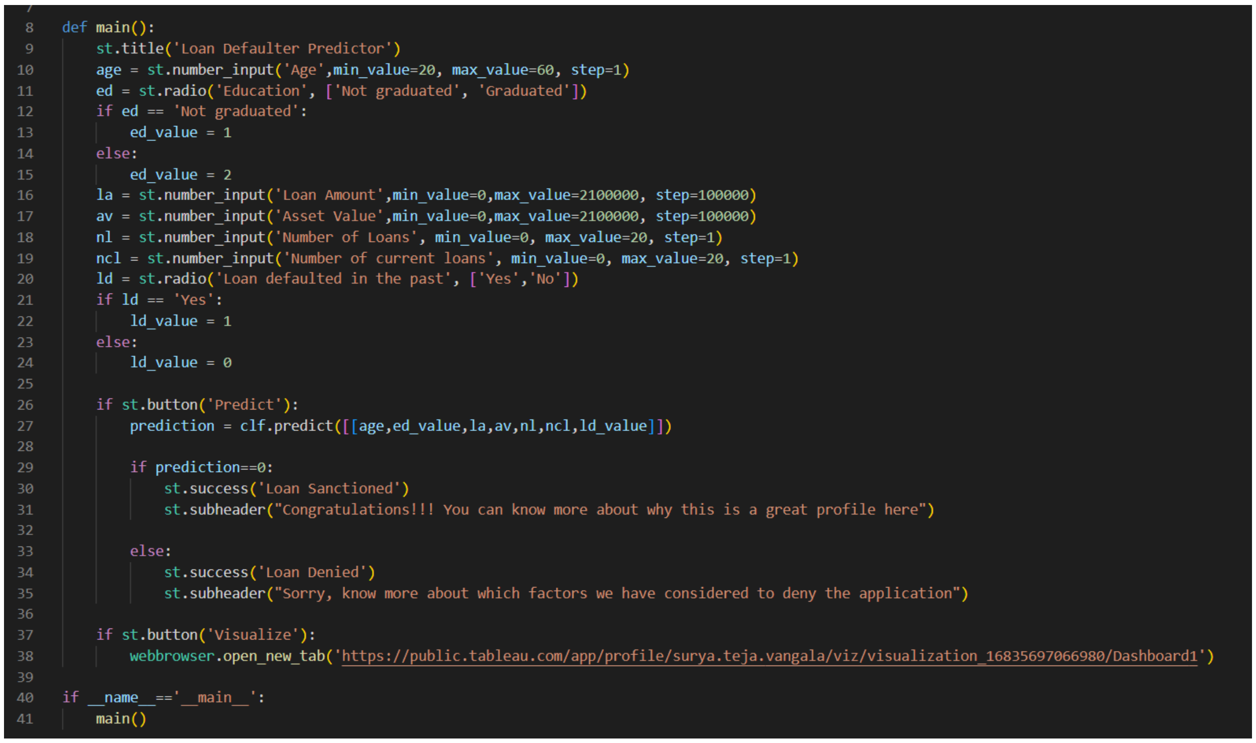
From the decision tree the weights have been printed for each variable and these weights are exported into a pickle file which will be used for the web application later.

3. Working of Web application



```
import pickle
import streamlit as st
import webbrowser
from sklearn.tree import DecisionTreeClassifier
with open('loandefaulters.pickle','rb') as f:
    clf = pickle.load(f)
```

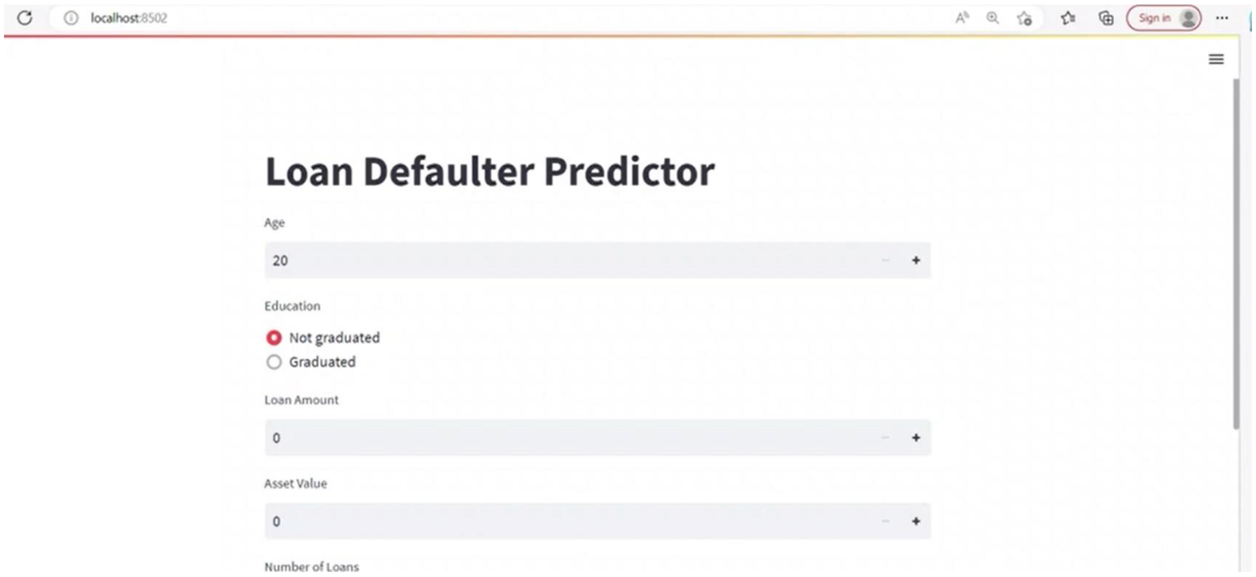
I have imported the pickle file to the python file and have imported streamlit to build the web application.



```python
def main():
    st.title('Loan Defaulter Predictor')
    age = st.number_input('Age',min_value=20, max_value=60, step=1)
    ed = st.radio('Education', ['Not graduated', 'Graduated'])
    if ed == 'Not graduated':
        ed_value = 1
    else:
        ed_value = 2
    la = st.number_input('Loan Amount',min_value=0,max_value=2100000, step=100000)
    av = st.number_input('Asset Value',min_value=0,max_value=2100000, step=100000)
    nl = st.number_input('Number of Loans', min_value=0, max_value=20, step=1)
    ncl = st.number_input('Number of current loans', min_value=0, max_value=20, step=1)
    ld = st.radio('Loan defaulted in the past', ['Yes','No'])
    if ld == 'Yes':
        ld_value = 1
    else:
        ld_value = 0

    if st.button('Predict'):
        prediction = clf.predict([[age,ed_value,la,av,nl,ncl,ld_value]])

        if prediction==0:
            st.success('Loan Sanctioned')
            st.subheader("Congratulations!!! You can know more about why this is a great profile here")

        else:
            st.success('Loan Denied')
            st.subheader("Sorry, know more about which factors we have considered to deny the application")

    if st.button('Visualize'):
        webbrowser.open_new_tab('https://public.tableau.com/app/profile/surya.teja.vangala/viz/visualization_16835697066980/Dashboard1')

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

The screenshot above is the code related the design of web application and its function.
https://public.tableau.com/app/profile/surya.teja.vangala/viz/visualization_16835697066980/Dashboard1
This is the link where I have hosted the visualization part of the web application.

Command to run the web application using streamlit
streamlit run dectree.py



Loan Defaulter Predictor

Age
20

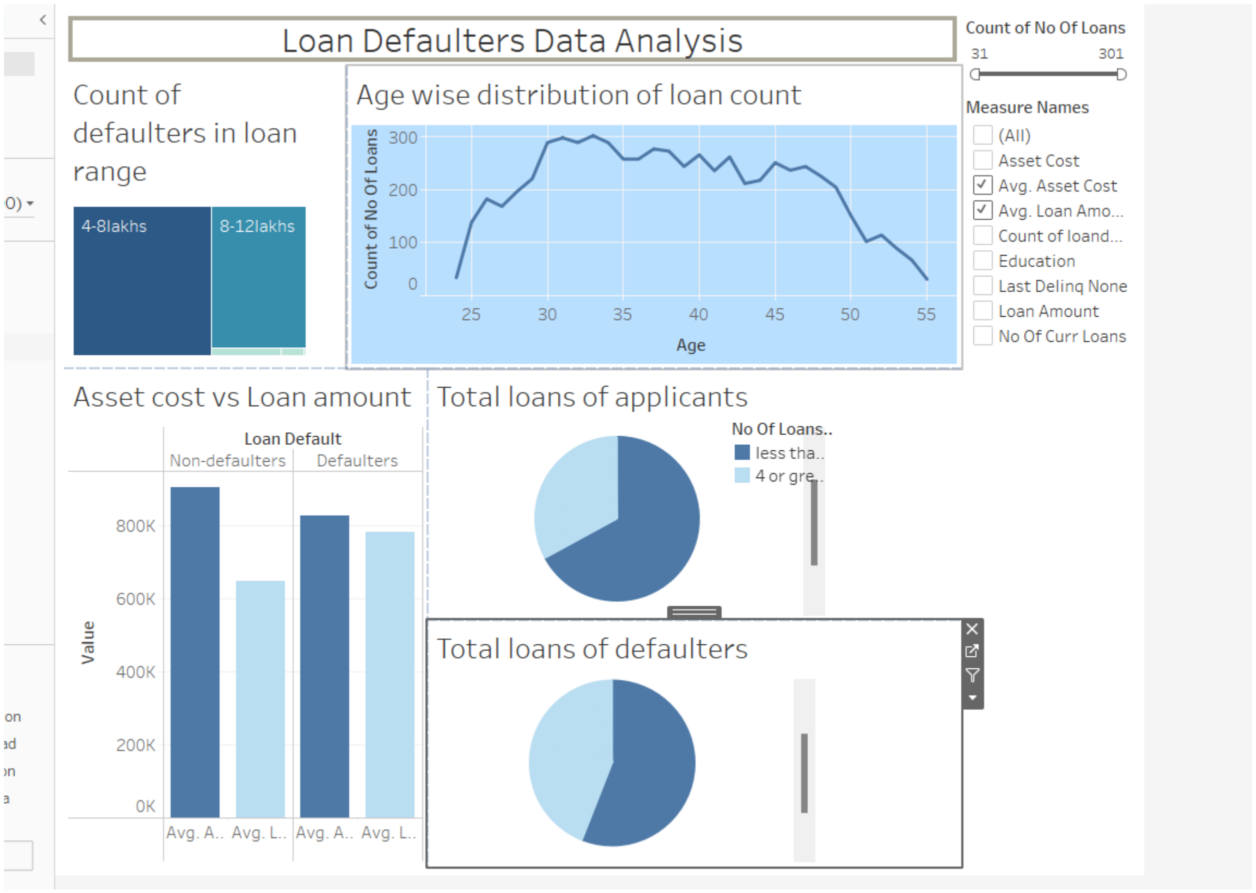
Education
[✓ Not graduated]
Graduated

Loan Amount
0

Asset Value
0

Number of Loans

4. Visualization



I have made my visualization dashboard part using tableau software.

Count of defaulters in loan range- I have divided the loan amount into 5 equal ranges from 0-20 lakhs (1 lakh= 100,000) and have made a graph to represent the count of loans in that category which shows that majority of the loans are in the ranges of 4-8 lakhs and 8-12 lakhs.

Age wise distribution of loan count- I have made a line graph that shows the number of loans distributed across all ages by the bank which shows that people between the ages 30-35 have highest approval rate and ages below 25 and above 50 have a very low approval rate.

Asset cost vs Loan amount- A bar graph comparing values of avg asset cost and avg loan amount between defaulters and non defaulters which shows defaulters have higher loan amount but lesser asset cost and it shows by how much in average.

Total loan of applicants and defaulters- here in the pie graph there are 2 categories that is less than 4 loans or greater than 4 loans which indicates whether applicants have more than 4 or less than 4 loans previously. It is shown that approximately one third of total applicants have 4 or more loans previously whereas if the graph was made on only defaulters we could see that more than 40 percent defaulters had 4 more previous loans.

5. Recommendations

Analysis on the data has clearly shown which factors have to be considered more. Total number of loans that the applicant has previously taken played a crucial role to decide the probability of whether they might default the loan or not. The trends show that on with higher number of loans have the tendency to default more loans. Similarly loan amount , asset cost have also showed a clear disparity on the trends of defaulters when the loan amount is high and the asset cost which they were using as a collateral is low there is a higher chance that the person could default the loan. Age of the person and existing loans of the person had little effect this might be because the applicants were only trying to get a new loan when they have cleared the previous ones as they would be aware that otherwise their application would be rejected.

This application is for the use of the bank to check whether a loan application has to be sanctioned or not and to check the risk involved in sanctioning the loan based on the data collected and the factors involved. They can know more about the red flags in an application and make changes in their loan policy accordingly.

The application could do a lot better if there would other crucial factors available such as credit score, income and purpose of the loan etc. So it would be very useful for the bank to collect this information which then could be added to the application. The application can also be extended to the customers where they can check their profile and know how to improve their profile if incase they would apply for a loan and how much loan are they eligible for.