MAHARASHTRA EDUCATION SOCIETY'S
(SINCE 1860)

INSTITUTE OF MANAGEMENT & CAREER COURSES (IMCC)

Approved by AICTE and Recognized by Savitribai Phule Pune University, Pune

IMCC Campus, 131, Mayur Colony, Kothrud, Pune 411038, Maharashtra, India | Ph.: 020-2546 6271 / 73 | e-mail: info.imcc@mespune.in | http://imcc.mespune.in

NAAC Accredited with Grade A+

Ref. No. MES IMCC / 198/ 2024 – 25 **Date:27/12/2024**

CERTIFICATE

This is to certify that the Project entitled **"Loan Check**" is completed by **"Pratyush Prakash Majumdar"** of M.C.A. Semester I for the Academic Year 2024-25 at MES' Institute of Management & Career Courses (IMCC), Pune - 411038.

To the best of our knowledge, this is an original study done by the said student and important sources used by him/her have been duly acknowledged in this report.

The report is submitted as a part of course ITP11 Mini Project for the Academic Year 2024-25 as per the rules and guidelines prescribed by the institute.

Ms. Kalpana Dhende Project Coordinator	Ms. Manasi Shirurkar Program Coordinator	Dr. Ravikant Zirmite Head, Dept. Of MCA

MAHARASHTRA EDUCATION SOCIETY'S
(SINCE 1860)

INSTITUTE OF MANAGEMENT & CAREER COURSES (IMCC)

Approved by AICTE and Recognized by Savitribai Phule Pune University, Pune

IMCC Campus, 131, Mayur Colony, Kothrud, Pune 411038, Maharashtra, India | Ph.: 020-2546 6271 / 73 | e-mail: info.imcc@mespune.in | http://imcc.mespune.in

NAAC Accredited with Grade A+

Ref. No. MES IMCC / 198/ 2024 – 25 **Date:27/12/2024**

CERTIFICATE

This is to certify that the Project entitled **"Loan Check**" is completed by **"Aditya Mahesh Pattar"** of M.C.A. Semester I for the Academic Year 2024-25 at MES' Institute of Management & Career Courses (IMCC), Pune - 411038.

To the best of our knowledge, this is an original study done by the said student and important sources used by him/her have been duly acknowledged in this report.

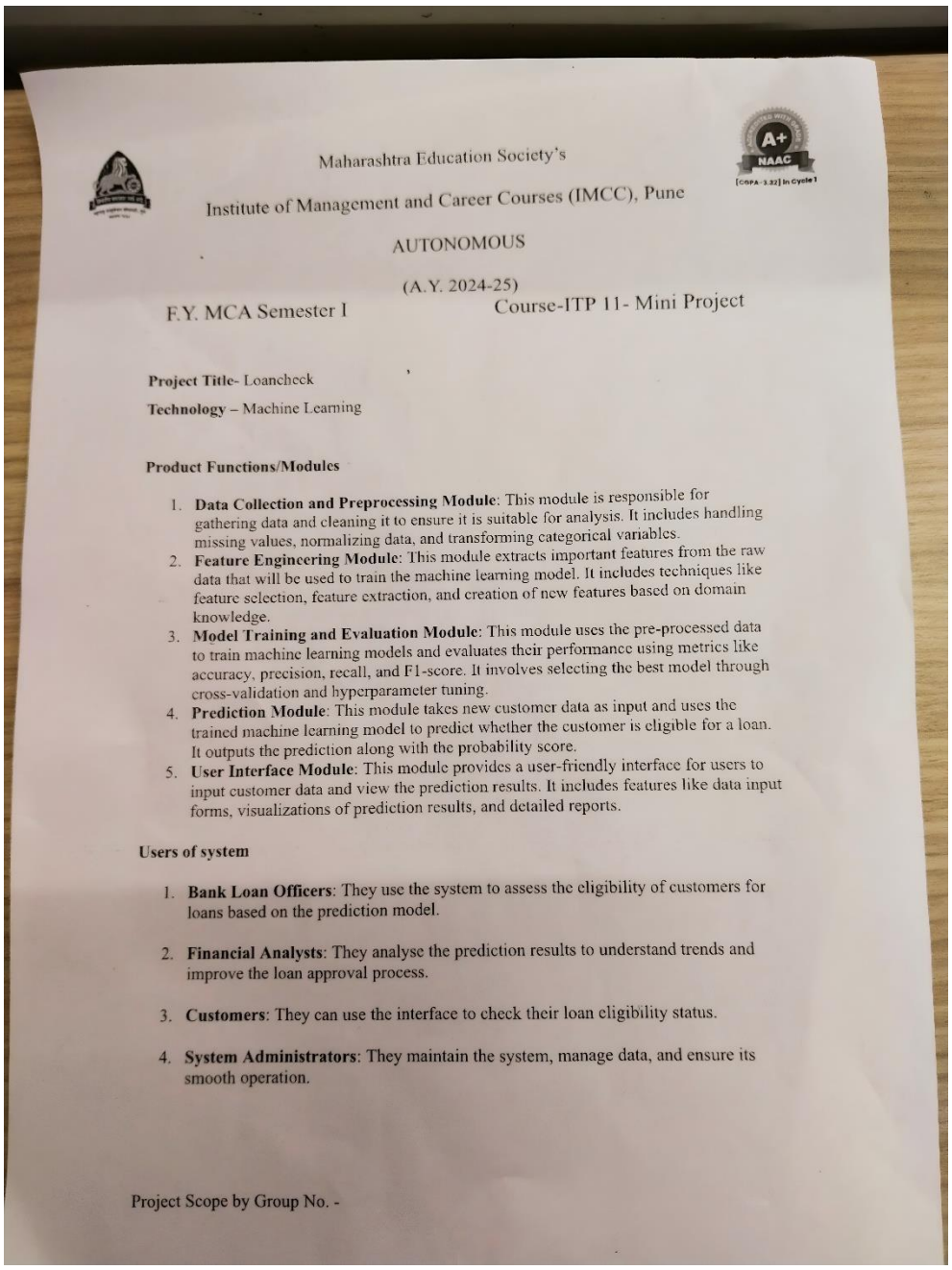
The report is submitted as a part of course ITP11 Mini Project for the Academic Year 2024-25 as per the rules and guidelines prescribed by the institute.

Ms. Kalpana Dhende
Project Coordinator

Ms. Manasi Shirurkar
Program Coordinator

Dr. Ravikant Zirmite
Head, Dept. Of MCA

Synopsis



Maharashtra Education Society's

Institute of Management and Career Courses (IMCC), Pune

AUTONOMOUS

(A.Y. 2024-25)

F.Y. MCA Semester I Course-ITP 11- Mini Project

Project Title- Loancheck

Technology – Machine Learning

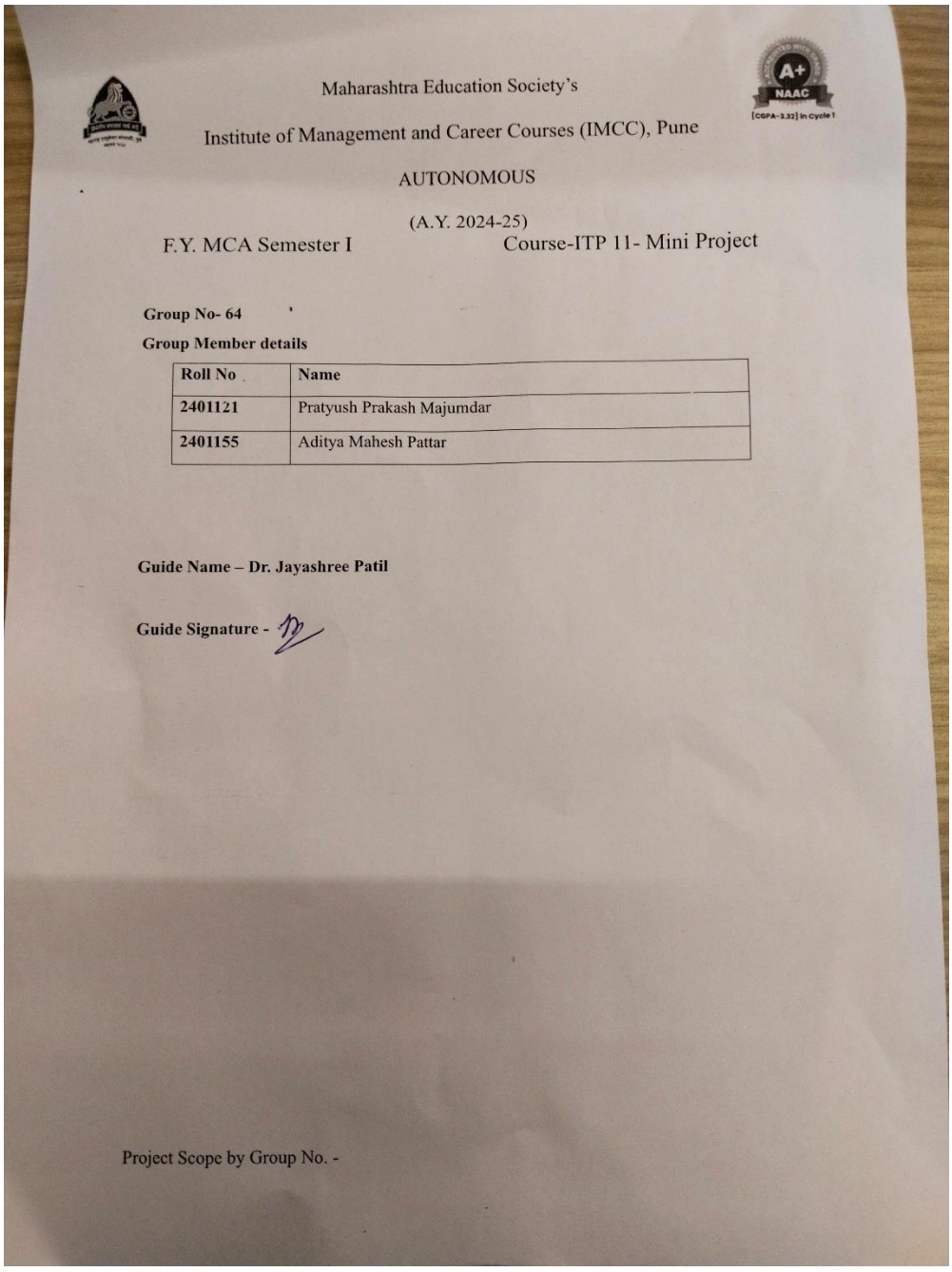
Product Functions/Modules

1. **Data Collection and Preprocessing Module**: This module is responsible for gathering data and cleaning it to ensure it is suitable for analysis. It includes handling missing values, normalizing data, and transforming categorical variables.
2. **Feature Engineering Module**: This module extracts important features from the raw data that will be used to train the machine learning model. It includes techniques like feature selection, feature extraction, and creation of new features based on domain knowledge.
3. **Model Training and Evaluation Module**: This module uses the pre-processed data to train machine learning models and evaluates their performance using metrics like accuracy, precision, recall, and F1-score. It involves selecting the best model through cross-validation and hyperparameter tuning.
4. **Prediction Module**: This module takes new customer data as input and uses the trained machine learning model to predict whether the customer is eligible for a loan. It outputs the prediction along with the probability score.
5. **User Interface Module**: This module provides a user-friendly interface for users to input customer data and view the prediction results. It includes features like data input forms, visualizations of prediction results, and detailed reports.

Users of system

1. **Bank Loan Officers**: They use the system to assess the eligibility of customers for loans based on the prediction model.
2. **Financial Analysts**: They analyse the prediction results to understand trends and improve the loan approval process.
3. **Customers**: They can use the interface to check their loan eligibility status.
4. **System Administrators**: They maintain the system, manage data, and ensure its smooth operation.

Project Scope by Group No. -



Maharashtra Education Society's

Institute of Management and Career Courses (IMCC), Pune

AUTONOMOUS

(A.Y. 2024-25)

F.Y. MCA Semester I Course-ITP 11- Mini Project

Group No- 64

Group Member details

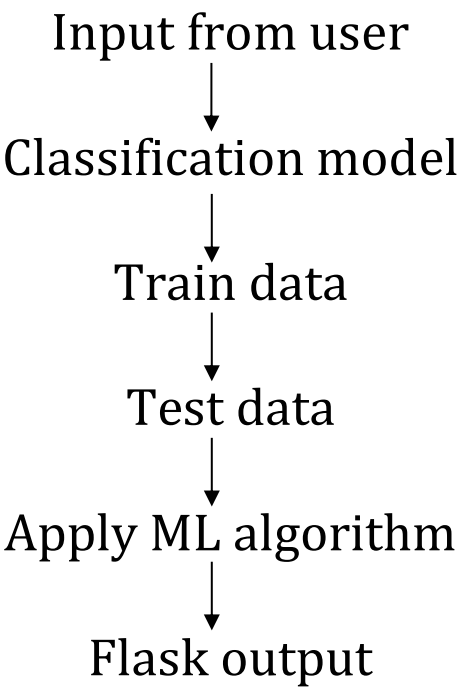
Roll No	Name
2401121	Pratyush Prakash Majumdar
2401155	Aditya Mahesh Pattar

Guide Name – Dr. Jayashree Patil

Guide Signature -

Project Scope by Group No. -

Use Case Diagram



MAHARASHTRA EDUCATION SOCIETY'S
(SINCE 1860)

INSTITUTE OF MANAGEMENT & CAREER COURSES (IMCC)

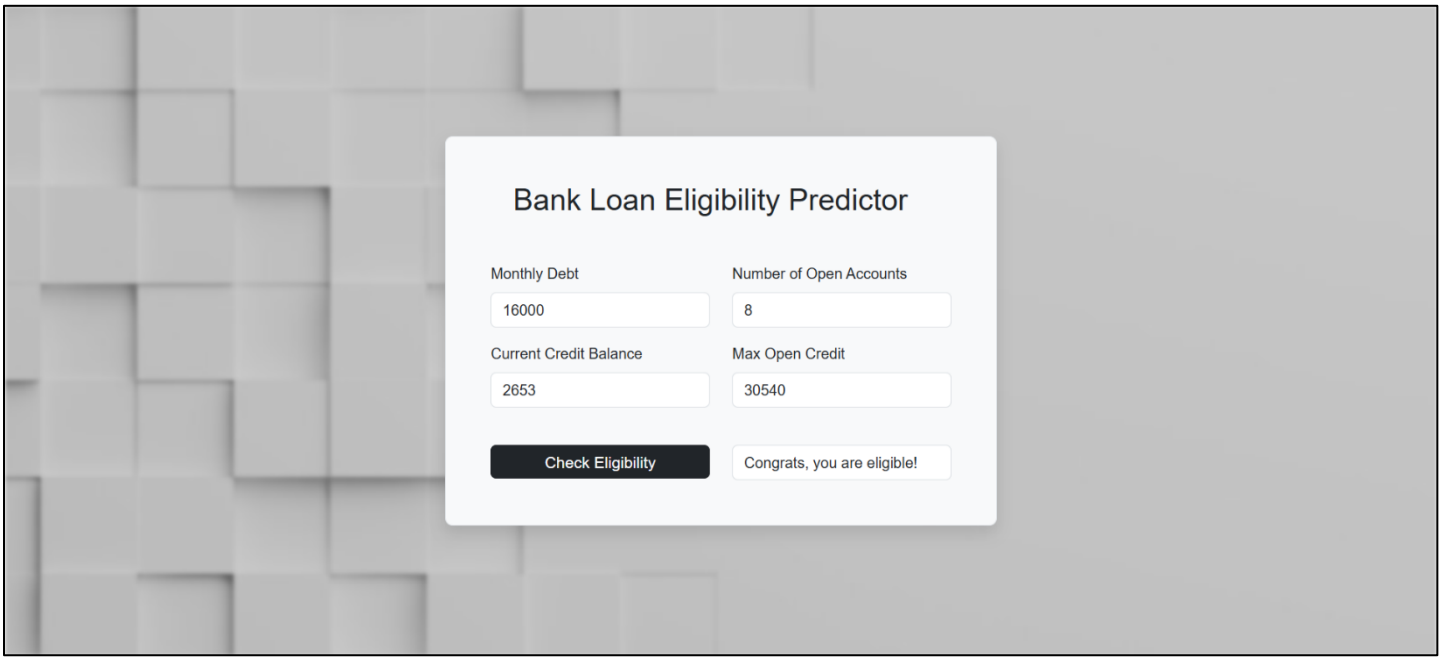
Approved by AICTE and Recognized by Savitribai Phule Pune University, Pune

IMCC Campus, 131, Mayur Colony, Kothrud, Pune 411038, Maharashtra, India | Ph.: 020-2546 6271 / 73 | e-mail: info.imcc@mespune.in | http://imcc.mespune.in

NAAC Accredited with Grade A+

Snapshots

1) Prediction Page



2) Import libraries and read the dataset

[1]:
```
# import the necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

[2]:
```
# read the dataset
df = pd.read_csv('C:/Users/ADMIN/MCA_Sem1Project/Bank_Dataset.csv', low_memory=False, header = 1, names=[
    'Loan_status', 'Curr_loan_amt', 'Term', 'Credit_score', 'Yrs_in_curr_job', 'Home_ownership', 'Annual_inc', 'Purpose', 'Monthly_debt',
    'Yrs_of_credit_hist', 'Months_since_last_delinquet', 'No_of_open_acc', 'No_of_credit_problems', 'Curr_credit_bal', 'Max_open_credit',
    'Bankruptcies', 'Tax_liens'])
df.head()
```

[2]:

	Loan_status	Curr_loan_amt	Term	Credit_score	Yrs_in_curr_job	Home_ownership	Annual_inc	Purpose	Monthly_debt	Yrs_of_credit_hist	Months_since_last_delinc
0	Loan Given	20032	Short Term	NaN	6 years	Home Mortgage	NaN	Debt Consolidation	81.25	20.0	
1	Loan Given	2962	Short Term	745.0	10+ years	Rent	29619.0	Debt Consolidation	81.25	20.0	
2	Loan Given	6722	Short Term	NaN	< 1 year	Home Mortgage	NaN	Debt Consolidation	904.07	24.5	
3	Loan Given	20845	Short Term	NaN	2 years	Home Mortgage	NaN	Debt Consolidation	923.16	21.3	
4	Loan Refused	14587	Short Term	728.0	1 year	Rent	42040.0	Debt Consolidation	923.16	21.3	

3) Check info and duplicate values

```
[4]: df.info()  # provides information about the columns in the dataset
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 111098 entries, 0 to 111097
Data columns (total 17 columns):
 #   Column                      Non-Null Count   Dtype
---  ------                      --------------   -----
 0   Loan_status                 111098 non-null  object
 1   Curr_loan_amt               111098 non-null  int64
 2   Term                        111098 non-null  object
 3   Credit_score                89761 non-null   float64
 4   Yrs_in_curr_job             106405 non-null  object
 5   Home_ownership              111098 non-null  object
 6   Annual_inc                  89761 non-null   float64
 7   Purpose                     111096 non-null  object
 8   Monthly_debt                111096 non-null  float64
 9   Yrs_of_credit_hist          111096 non-null  float64
 10  Months_since_last_delinquet 52096 non-null   float64
 11  No_of_open_acc              111096 non-null  float64
 12  No_of_credit_problems       111096 non-null  float64
 13  Curr_credit_bal             111096 non-null  float64
 14  Max_open_credit             111096 non-null  float64
 15  Bankruptcies                110869 non-null  float64
 16  Tax_liens                   111086 non-null  float64
dtypes: float64(11), int64(1), object(5)
memory usage: 14.4+ MB
```

```
[5]: df.duplicated().sum()  # checks for duplicate values
```

```
[5]: 3250
```

```
[6]: df = df.drop_duplicates()  # we drop the duplicate values
     df.shape
```

```
[6]: (107848, 17)
```

4) Check and replace null values

```
[19]: df.isna().sum()  # checks for null values
```

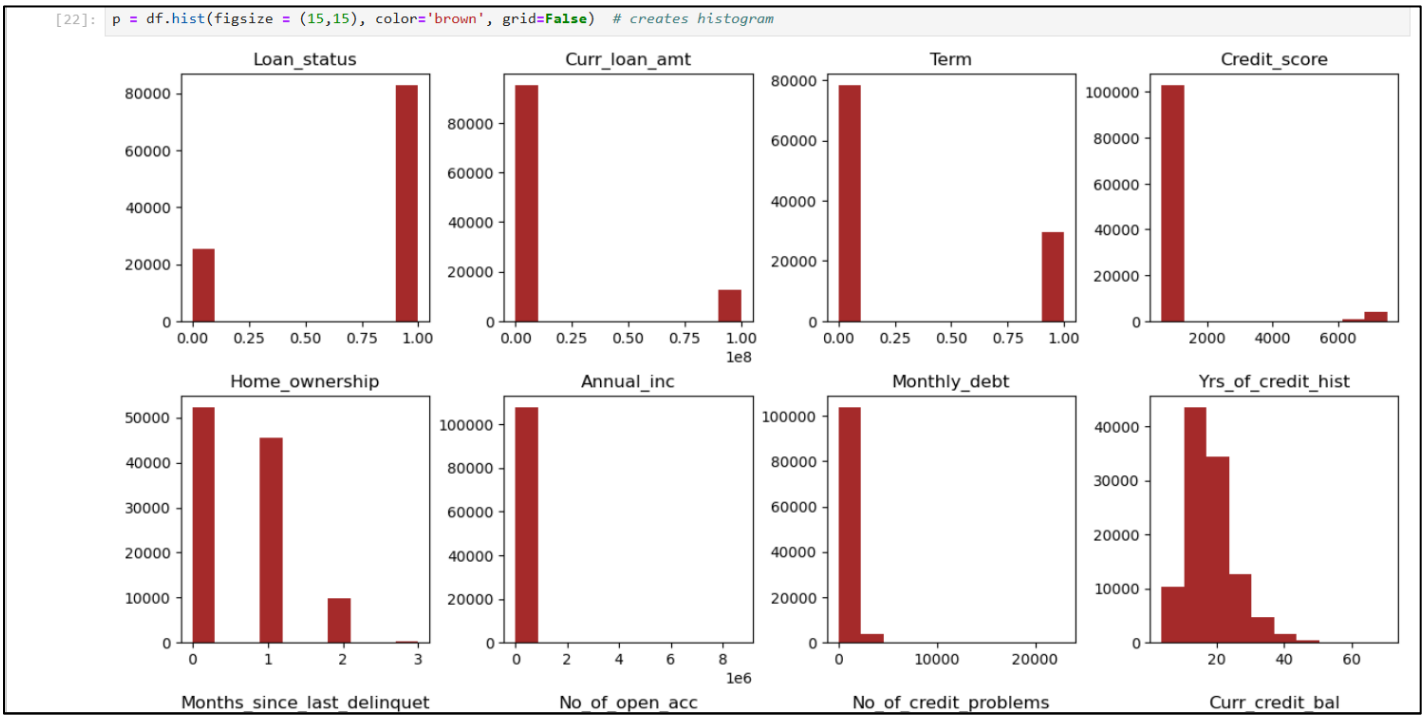
```
[19]: Loan_status                     0
      Curr_loan_amt                  0
      Term                           0
      Credit_score                21337
      Yrs_in_curr_job              4555
      Home_ownership                 0
      Annual_inc                 21337
      Purpose                        2
      Monthly_debt                   2
      Yrs_of_credit_hist             2
      Months_since_last_delinquet    57494
      No_of_open_acc                 2
      No_of_credit_problems          2
      Curr_credit_bal                2
      Max_open_credit                2
      Bankruptcies                 223
      Tax_liens                     12
      dtype: int64
```

```
[20]: # For numerical and discrete columns, we fill na values with its mode
      mode_cols = ['Credit_score', 'Yrs_in_curr_job', 'Months_since_last_delinquet', 'Bankruptcies', 'Tax_liens']
      for i in mode_cols:
          df[i].fillna(df[i].mode()[0],inplace=True)
      df.isna().sum()
```

```
[20]: Loan_status                     0
      Curr_loan_amt                  0
      Term                           0
      Credit_score                   0
      Yrs_in_curr_job                0
      Home_ownership                 0
      Annual_inc                 21337
```

5) Create histograms to check skewness of data

```
[22]: p = df.hist(figsize = (15,15), color='brown', grid=False)  # creates histogram
```



6) Check and handle outliers

[24]:
```
# function to return the lower wisker and upper wisker of a column
def wiskers(col):
    q1,q3 = np.percentile(col,[25,75])  # calculates the 25 and 75 percentile respectively
    iqr = q3 - q1  # calculates the inter quartile range
    lw = q1 - 1.5 * iqr  # calculates the lower wisker
    uw = q3 + 1.5 * iqr  # calculates the upper wisker
    return lw, uw
```

[25]:
```
wiskers(df['Annual_inc'])  # returns the lower and upper wisker for Credit_score column
```

[25]:
```
(3678.0, 124910.0)
```

[26]:
```
# remove outliers using iqr method
original_df = df  # creates copy of original dataframe
outlier_col = ['Annual_inc']
for i in outlier_col:
    lw,uw = wiskers(original_df[i])
    df = original_df[(original_df[i] >= lw) & (original_df[i] <= uw)]  # For removing outliers
df.shape
```

[26]:
```
(100397, 17)
```

[27]:
```
# replace the rows in Curr_loan_amt having values as 99999999 to the median of its column (Refer dataset to udnerstand this)
curr_loan_amt_median = df[df['Curr_loan_amt'] < 99999999]['Curr_loan_amt'].median()
df.loc[df['Curr_loan_amt'] == 99999999, 'Curr_loan_amt'] = curr_loan_amt_median
df.shape
```

[27]:
```
(100397, 17)
```

[28]:
```
monthly_debt_thresh = 10000   # keeping threshold value as 10000
df = df[(df['Monthly_debt'] <= monthly_debt_thresh)]
df.shape   # 4 rows removed
```

7) Perform factor analysis to find potential factors

[38]:
```
# we perform factor analysis
from sklearn.decomposition import FactorAnalysis
n_factors = 4  # Choose the number of factors
fa = FactorAnalysis(n_components=n_factors, random_state=10, max_iter=1000)
fa.fit(X_train)
```

[38]:
```
FactorAnalysis(n_components=4, random_state=10)
```

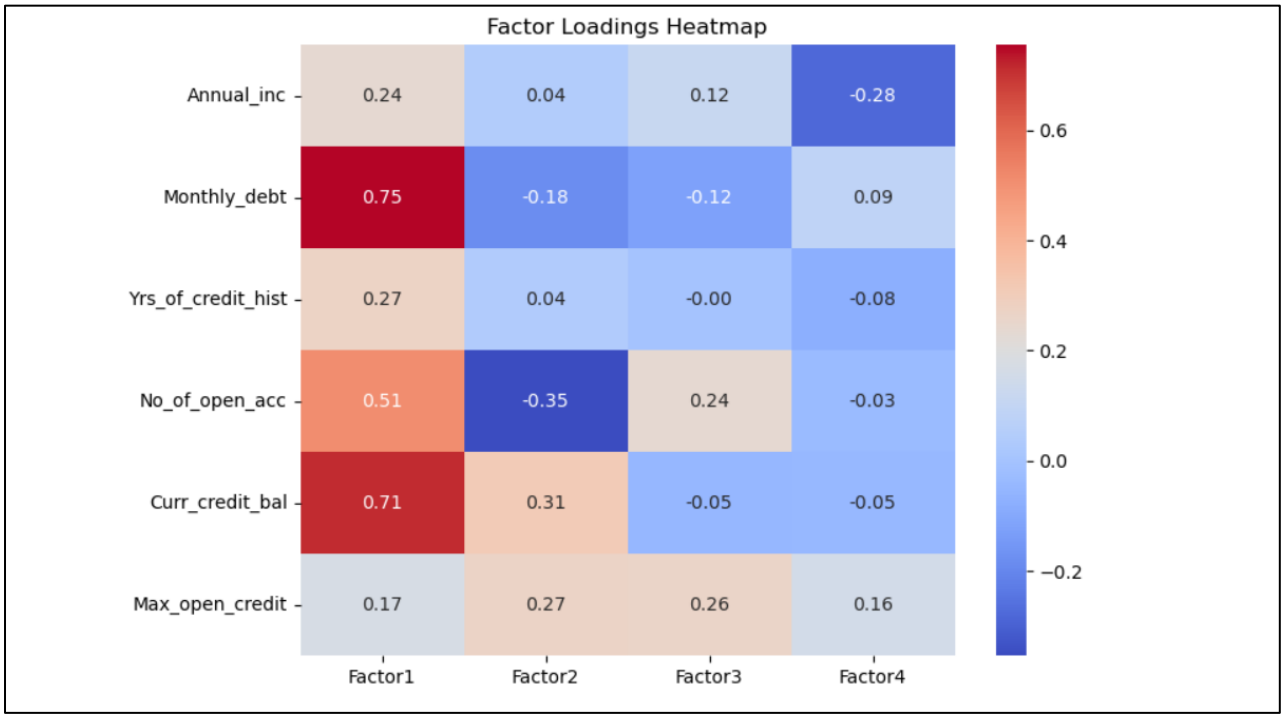
[39]:
```
factor_loadings = pd.DataFrame(fa.components_.T, columns=[f"Factor{i+1}" for i in range(n_factors)],
                               index=['Annual_inc', 'Monthly_debt', 'Yrs_of_credit_hist', 'No_of_open_acc', 'Curr_credit_bal', 'Max_open_credit'])
factor_loadings
```

[39]:

	Factor1	Factor2	Factor3	Factor4
Annual_inc	0.237213	0.043713	0.116593	-0.281577
Monthly_debt	0.754796	-0.181705	-0.120243	0.085491
Yrs_of_credit_hist	0.271385	0.037225	-0.003048	-0.079339
No_of_open_acc	0.508155	-0.353956	0.237291	-0.033769
Curr_credit_bal	0.713848	0.311501	-0.048266	-0.046095
Max_open_credit	0.172481	0.266602	0.264850	0.157076

[40]:
```
# Step 7: Heatmap of Factor Loadings for Visualization
plt.figure(figsize=(8, 6))
sns.heatmap(factor_loadings, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Factor Loadings Heatmap")
plt.show()
```

8) Heatmap to display the factor loadings



9) Find the best model for prediction

```
[43]: # import some models to find the best model
      from sklearn.model_selection import cross_val_score
      from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
      from sklearn.tree import DecisionTreeClassifier
      from sklearn.linear_model import LogisticRegression
      from xgboost import XGBClassifier
      from sklearn.neighbors import KNeighborsClassifier

[44]: cross_val_score(LogisticRegression(),X_train, Y_train, cv=3)

[44]: array([0.76164506, 0.76164506, 0.76163616])

[45]: cross_val_score(GradientBoostingClassifier(),X_train, Y_train, cv=3)

[45]: array([0.76164506, 0.76138359, 0.76152409])

[46]: cross_val_score(RandomForestClassifier(),X_train, Y_train, cv=3)

[46]: array([0.73340555, 0.73699152, 0.73078072])

[47]: cross_val_score(DecisionTreeClassifier(),X_train, Y_train, cv=3)

[47]: array([0.62317433, 0.62399612, 0.61998506])

[48]: cross_val_score(XGBClassifier(),X_train, Y_train, cv=3)

[48]: array([0.75876882, 0.75925442, 0.75752708])

[49]: cross_val_score(KNeighborsClassifier(),X_train, Y_train, cv=3)

[49]: array([0.71480333, 0.71685779, 0.71154277])
```

10) Calculate metrics.

[52]:
```
from sklearn.metrics import precision_score, f1_score, accuracy_score, classification_report

# Make predictions
y_pred = model.predict(X_test)

# # Calculate metrics
accuracy = accuracy_score(Y_test, y_pred)
precision = precision_score(Y_test, y_pred)
f1 = f1_score(Y_test, y_pred)

# # Display metrics
print(f"Accuracy Score: {round(accuracy, 2)}")
print(f"Precision Score: {round(precision, 2)}")
print(f"F1 Score: {round(f1, 2)}")
```

```
Accuracy Score: 0.76
Precision Score: 0.76
F1 Score: 0.86
```

11) Store predictions in excel and pickle file

[53]:
```
# Creating the DataFrame
predictions_df = pd.DataFrame({
    "Predicted_Loan_Status": y_pred,
    "Monthly_debt": X_test['Monthly_debt'],
    "No_of_open_acc": X_test['No_of_open_acc'],
    "Curr_credit_bal": X_test['Curr_credit_bal'],
    "Max_open_credit": X_test['Max_open_credit'],
})

# Exporting to Excel
file_name = "Loan_Predictions.xlsx"
predictions_df.to_excel(file_name, index=False)

print(f"Predictions have been saved to {file_name}")
```

```
Predictions have been saved to Loan_Predictions.xlsx
```

[54]:
```
model.predict([[1600, 8, 2683, 3053]])
```

[54]:
```
array([0])
```

[55]:
```
import pickle as p
with open('loan_eligibility_predictor_pickle', 'wb') as f:
    p.dump(model, f)
```