**BCIS5110 Project Part-2**

**By**

**Abhignu reddy Alety**

**Chavidisetty naga kalyan babu**



**Executive Summary:**

Kiva is a nonprofit organization which crowdfunds loans for the needy community and also developing the quality and reducing the price of financial services and also people who cannot access or afford this financial aid.

Our goal at hand is to understand the structure of the data in kiva website by examining every variable in the data set. In this project we have used kiva data set for creating models using distinct techniques and execute specific regression models using Jupyter IDE. Using these models one can analyze the loan amounts using any given parameters. These models will enable loan lenders and loan seekers to make inform decisions and to maximize kiva’s loan disbursement and its efficiency. It is highly helpful for students to pay for their tuition and people who can start their businesses and support families or individuals.

Using Jupyter IDE we developed the regression models such as bar charts, scatter plots, tables or data sets, line charts. We study and analyze three major data sets which are loan\_lenders, loans and lenders which contain all the data required to fulfill the needs. We also predicted if a loan can get funded based on the loan features.

**Project Motivation/background:**

The reason I choose kiva data set is that it deals with the prediction of loans where we get to know the status of the loan and our objective of the project is to join the kivas dataset with extra wellsprings of information so we can appraise the government assistance levels of borrowers in determined locales dependent on shared financial and segment qualities.

The reason us to choose this project is that we get to deal with scatter plot in the project where we use scatter plots to see the relationship between variables and used dots to represent the relationship between them.

Matplotlib library will be used to draw a scatter plot. Scatter plots are widely used to represent relation among variables and how change in one effect the other.

The background of the dataset is about the kiva which is a non-profit organization which deals with crowdfunding loans which helps a lot of borrowers in achieving their career or future goals. I have seen that kiva does not accept interest based on any loan they take and as per the details or facts among 96% of the loans given are returned on time. I mean seeing these numbers amazed me because now a days most of the financial services does not provide a loan without taking interest from the borrowers and on the other hand kiva does absolute best in giving out the loans without considering any interest or any kind of charges.

**Libraries Used:**

NumPy as np # linear algebra

pandas as pd # data processing, CSV file I/O (e.g., pd.read\_csv)

matplotlib.pyplot as plt # plotting graphs/visualizations

pandas\_profiling # quick EDA

plotly.graph\_objects

plotly.offline.

**Data Description:**

In this project we use these three data sets which are loan\_lenders, loans, and lenders to acquire and analyse the required information. In loan\_lenders we deal with two information sets which are LOAN\_ID and lenders i.e.., who gives out the loans. In lenders we have several information sets which are PERMANENT\_NAME, DISPLAY\_NAME, MAIN\_PIC\_ID, CITY, STATE, COUNTRY\_CODE, MEMBER\_SINCE, PERSONAL\_URL, OCCUPATION, LOAN\_BECAUSE, OTHER\_INFO, LOAN\_PURCHASE\_NUM, INVITED\_BY, NUM\_INVITED. These columns indicate the information of the lenders. This data set combines the information acquired from both LOAN\_LENDERS and lenders. Loan\_id acts as the primary key with the help of which we can access all the variables in the given three data sets. All the three data sets contain different data types varying from integer to character data formats. Usually in most of the cases integer data types indicates the loan amount, the LOAN\_ID etc. Character data types are used to represent the information such as name of the lender, seeker and the address of the lender and seeker.

Demographic information can also be accessed in the data set file which can help users to locate the information of the lender or seeker efficiently. The funded and borrowed amount can be easily attained through LOAN data set. The seekers who are availing loans can also be attained.

**Column descriptions:**

Id: Unique ID for loan (Loan ID)

Loan theme ID: Unique ID for Loan theme

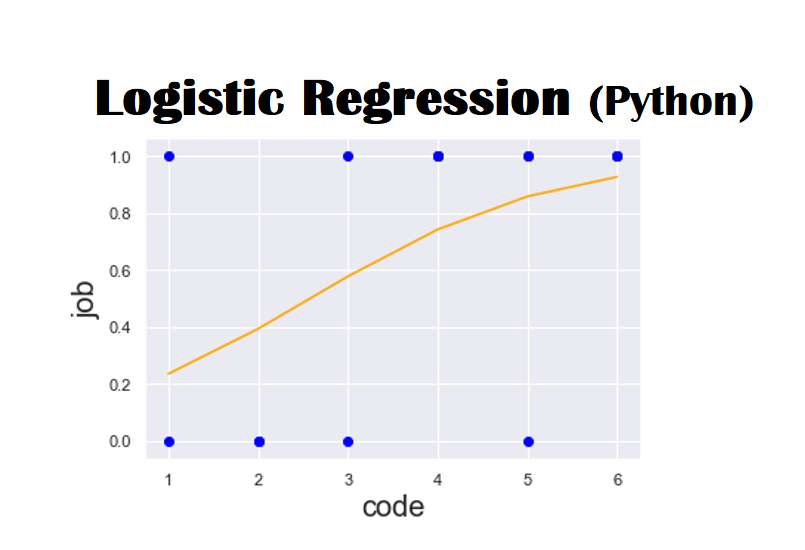
Loan theme type: General description for the loan theme category

Partner ID: Unique ID for field partners (Partner ID)

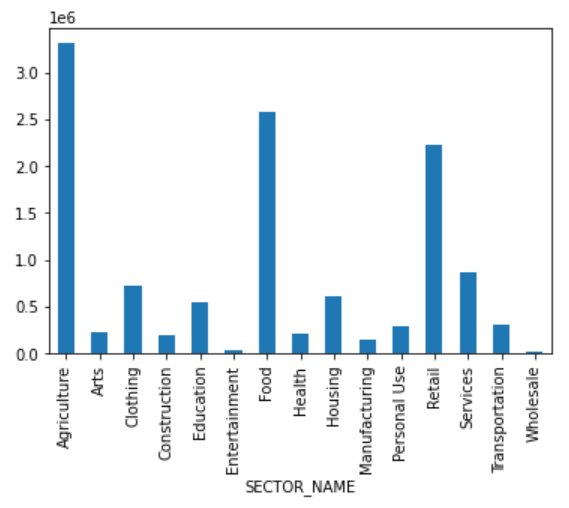
**Data transformation/ Exploratory data analysis:**

Data transformation is the most common way of changing over information from one arrangement or construction into another configuration or design. It is a principal part of most information integration and information the board undertakings, for example, information fighting, information warehousing, information reconciliation and application mix.

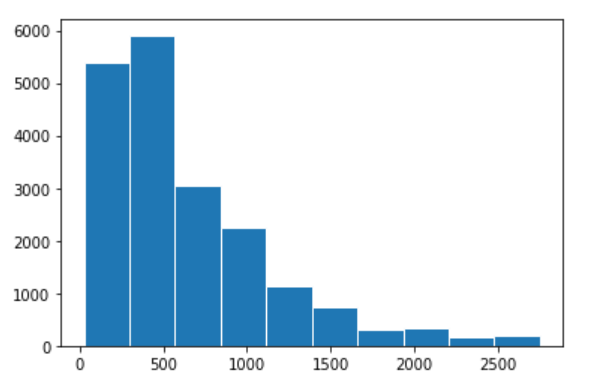
Exploratory information examination is a methodology of investigating informational indexes to sum up their primary qualities, regularly utilizing factual designs and different information representation strategies. Using exploratory data analysis, we distributed the data using different variables and we can figure out what would be the total number of lenders and borrowers in united states of America.

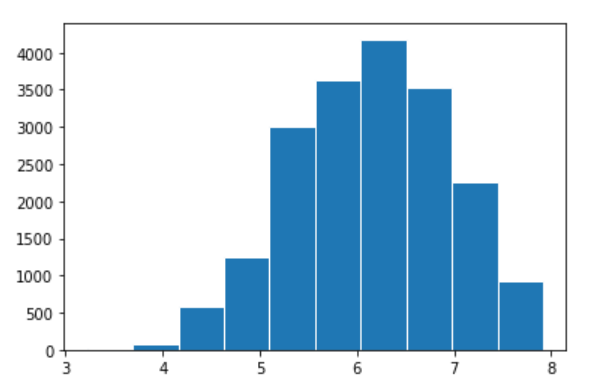


**Models and analysis:**



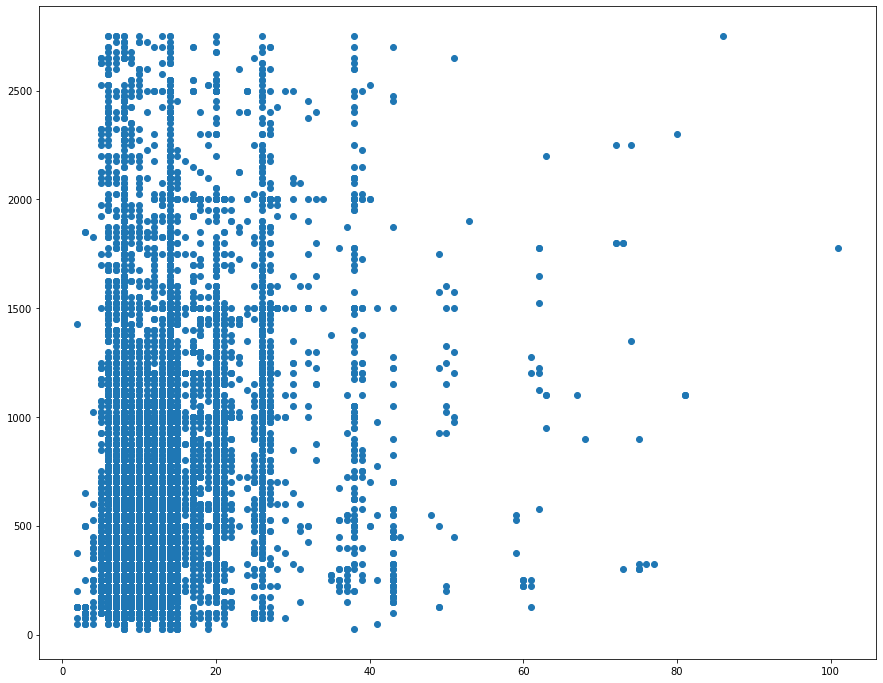
Over here in the above chart we can see which sector has the highest amount of loan in all of the sectors. Agriculture got the highest amount of loans when compared to all other sectors and food sector got the highest number of loans after agriculture.





If we observe Loan amount is skewed so we have used log to smooth it and log amount is normally distributed.

**Scatter Plot**



Over here we can see the scatter plot of ‘LOAN\_AMOUNT’ and ‘LENDER\_TERM

**Conclusions:**

**If we observe loan amount is skewed so we have used log to smooth it n log amount is normally distributed**

**Agriculture sector has the highest amount of loans**

**Philippines has the highest number of loans of all the countries**

**There are several categorical variables in the project**

**The no of loan borrowers decreased from time to time**

**Codes:**

1. Read in the three tables (the original dataset you downloaded, not the sample I provided for your initial report.)

**Code:** import pandas as pd

import numpy as np

#lenders=pd.read\_csv('C:/Users/LaptopCheckout/Downloads/kiva\_ds\_csv/lenders.csv')

df\_loans=pd.read\_csv('C:/Users/LaptopCheckout/Downloads/kiva\_ds\_csv/loans.csv')

#df\_loans\_lenders=pd.read\_csv('C:/Users/LaptopCheckout/Downloads/kiva\_ds\_csv/loans\_lenders.csv')

#lenders.head()

df\_loans.head(100)

#df\_loans\_lenders.head()

2. Display the information about each dataframe.

**Code:** print(df\_loans.head())

print(df\_loans.head())

#print(df\_loans\_lenders.head())

3. Check for missing values of the loans dataframe. You can use sum() to find out the number of missing values.

**Code:** df\_loans\_miss = df\_loans.isnull().sum().sum()

df\_loans\_miss

4. First, we check the dependent variable 'STATUS' in the loans dataframe. How many unique values?

**Code:** counts = df\_loans.nunique()

counts

df\_loans['STATUS'].unique

5. How many observations for each category of 'STATUS'? (Which function we have learned can do that?)

**Code:** catag = df\_loans['STATUS'].value\_counts()

catag

6. Our purpose is to build a model that predicts whether a loan can get funded based on loan features. So, which suggests the loan is funded? Which not? Which should we ignore?<br>

**Code:** 1) Delete the records with the category we can ignore.<br>

2) Recode the three categories to two categories (funded, notfunded)<br>

The recoded variable will be our target variable.

df\_loans['STATUS']=df\_loans[df\_loans['STATUS']!="fundRaising"]

#print(test['STATUS'].unique())

df\_loans['STATUS'].replace(to\_replace={"expired":["notfunded"],"refunded":["notfunded"]},inplace=True)

print(df\_loans['STATUS'].unique())

#loans.unique()

new\_set=df\_loans['STATUS'].value\_counts()

new\_set

7. Examine the variable 'LOAN\_AMOUNT'. <br>

**Code:** 1) Check the distribution. What is the value of outliers if we use 3\*IQR to define? <br>

2) Remove the outliers. <br>

3) Create a new variable by taking log of this variable.

df\_loans['LOAN\_AMOUNT'].hist(bins=10)

import numpy as np

df\_loans\_out = df\_loans[np.abs(df\_loans.LOAN\_AMOUNT-df\_loans.LOAN\_AMOUNT.mean()) <= (3\*df\_loans.LOAN\_AMOUNT.std())]

# keep only the ones that are within +3 to -3 standard deviations in the column 'Data'.\

df\_loans\_out = df\_loans\_out[~(np.abs(df\_loans\_out.LOAN\_AMOUNT-df\_loans\_out.LOAN\_AMOUNT.mean()) > (3\*df\_loans\_out.LOAN\_AMOUNT.std()))]

df\_loans\_out['LOAN\_AMOUNT'].hist()

8. Observe the missing value results from Q3. What is the outcome for variables 'COUNTRY\_CODE' and 'COUNTRY\_NAME'. What would you do? Do it.

**Code:** df\_loans\_out["COUNTRY\_NAME"].replace("",np.nan,inplace=True)

#df\_loans\_out.at[0,'COUNTRY\_NAME']=""

df\_loans\_out["COUNTRY\_NAME"]

pip install pycountry

import pycountry

list\_alpha\_2 = [i.alpha\_2 for i in list(pycountry.countries)]

list\_alpha\_3 = [i.alpha\_3 for i in list(pycountry.countries)]

pycountry.countries.get(alpha\_3='USA').name

def country\_flag(df):

#print(df['COUNTRY\_CODE'])

if type(df['COUNTRY\_CODE']) == float:

df['COUNTRY\_CODE'] = 'n/a'

if len(df['COUNTRY\_CODE'])==2 and df['COUNTRY\_CODE'] in list\_alpha\_2:

return pycountry.countries.get(alpha\_2=df['COUNTRY\_CODE']).name

elif len(df['COUNTRY\_CODE'])==3 and df['COUNTRY\_CODE'] in list\_alpha\_3:

return pycountry.countries.get(alpha\_3=df['COUNTRY\_CODE']).name

else:

return 'Invalid Code'

df\_loans\_out['COUNTRY\_name']=df\_loans\_out.apply(country\_flag, axis = 1)

df\_loans\_out['COUNTRY\_name']

9. Find the number of missing values in variables 'IMAGE\_ID', 'VIDEO\_ID' and 'LOAN\_USE'. What would you do? Do it.

**Code:** #df\_loans\_out = df\_loans\_out.drop(columns=['VIDEO\_ID'])

df\_loans\_out['IMAGE\_ID'].replace('', np.nan, inplace=True)

df\_loans\_out.dropna(subset=['IMAGE\_ID'], inplace=True)

df\_loans\_out['LOAN\_USE'].replace('', np.nan, inplace=True)

df\_loans\_out.dropna(subset=['LOAN\_USE'], inplace=True)

df\_loans\_out.info()

10. Create a new variable 'TIME\_LENGTH' that is the difference between 'POSTED\_TIME' and 'RAISED\_TIME'.

**Code:** df\_loans\_out['POSTED\_TIME'] = pd.to\_datetime(df\_loans\_out.POSTED\_TIME, format='%Y/%m/%d %H:%M:%S')

df\_loans\_out['RAISED\_TIME'] = pd.to\_datetime(df\_loans\_out.RAISED\_TIME, format='%Y/%m/%d %H:%M:%S')

df\_loans\_out['TIME\_LENGTH'] = df\_loans\_out['RAISED\_TIME']-df\_loans\_out['POSTED\_TIME']

df\_loans\_out['TIME\_LENGTH'].head()

11. Find the different between 'POSTED\_TIME' and 'DISBURSE\_TIME'. Create a new variable 'PREDISBURSE' if the 'DISBURSE\_TIME' is before 'POSTED\_TIME'.

**Code:** df\_loans\_out['POSTED\_TIME'] = pd.to\_datetime(df\_loans\_out.POSTED\_TIME, format='%Y/%m/%d %H:%M:%S')

df\_loans\_out['DISBURSE\_TIME'] = pd.to\_datetime(df\_loans\_out.DISBURSE\_TIME, format='%Y/%m/%d %H:%M:%S')

df\_loans\_out['PREDISBURSE'] = df\_loans\_out['POSTED\_TIME']-df\_loans\_out['DISBURSE\_TIME']

df\_loans\_out['PREDISBURSE'].head()

12. Exame variable 'CURRENCY\_EXCHANGE\_COVERAGE\_RATE'. How many unique values for this variable? How many missing?<br>

Before we rush to replace missing values with mode, we ask the question: is the missing value actually missing or is it something else? We then read the policies on Kiva about exchange coverage. For instance, a lender gives 25 in US dollars and it changes to 250 in another currency. But when the money is repaid at 250 in the other currency, it only exchanges to 20 US dollars. The CURRENCY\_EXCHANGE\_COVERAGE\_RATE measures how much such loss will be covered by the field partners. Some would cover 10% or 20%. So, it is more likely that the missing values indicating the loan does not have any coverage. So, it's more likely means 0.<br>

Replace the missing value with 0.

**Code:** catag1 = df\_loans\_out['CURRENCY\_EXCHANGE\_COVERAGE\_RATE'].value\_counts()

df\_loans\_out['CURRENCY\_EXCHANGE\_COVERAGE\_RATE'] = df\_loans\_out['CURRENCY\_EXCHANGE\_COVERAGE\_RATE'].fillna(0)

catag2 = df\_loans\_out['CURRENCY\_EXCHANGE\_COVERAGE\_RATE'].value\_counts()

print(catag1,catag2)

13. Take a look at the data after we have done the 12 steps above. We will continue the preparation.

Check the unique values of 'ACTIVITY\_NAME' and 'SECTOR\_NAME'. You may notice that there are many more activities than sectors and Sector is higher-level categorization of loans. We can choose one of them. To keep the results easier to read. We can keep sector name.

'LOAN\_NAME' seems irrelevant to whether a loan is funded. We have not learned how to process 'DESCRIPTION' and 'DESCRIPTION\_TRANSLATED'. 'FUNDED\_AMOUNT' is the amount that actually raised at the end. Most times it is the same as 'LOAN\_AMOUNT'. After doing similar analysis with every variable, we decide to keep only the following variables in the final analysis:

LOAN\_ID

ORIGINAL\_LANGUAGE<br>

LOAN\_AMOUNT (and the logAmount variable we created)<br>

STATUS<br>

IMAGE\_ID (the one we coded to 0/1)<br>

SECTOR\_NAME

COUNTRY\_NAME

CURRENCY\_POLICY

CURRENCY\_EXCHANGE\_COVERAGE\_RATE (the one we tranfromed)<br>

POSTED\_TIME (and the TIME\_LENGTH we created)<br>

DISBURSE\_TIME (and the PREDISBURSE we created)<br>

RAISED\_TIME

LENDER\_TERM

NUM\_JOURNAL\_ENTRIES

NUM\_BULK\_ENTRIES

REPAYMENT\_INTERVAL

DISTRIBUTION\_MODEL

Create a copy of the dataset that only contains the variables listed above. You may use the list of variable names to slice the dataframe we have from Step 12 in Assignment 8&9.

**Code:** df\_loans\_out['logAmount'] = np.log(df\_loans\_out['LOAN\_AMOUNT'])

variables=['LOAN\_ID', 'ORIGINAL\_LANGUAGE', 'LOAN\_AMOUNT', 'logAmount', 'STATUS', 'IMAGE\_ID', 'SECTOR\_NAME', 'COUNTRY\_NAME', 'CURRENCY\_POLICY', 'CURRENCY\_EXCHANGE\_COVERAGE\_RATE', 'POSTED\_TIME','DISBURSE\_TIME','PREDISBURSE','RAISED\_TIME','LENDER\_TERM','NUM\_JOURNAL\_ENTRIES','NUM\_BULK\_ENTRIES','REPAYMENT\_INTERVAL','TIME\_LENGTH','DISTRIBUTION\_MODEL']

loansAnalysis = df\_loans\_out[variables]

loansAnalysis.head()

14. Next we start to explore the data. You need to import matplotlib. First, make two histograms for 'LOAN\_AMOUNT' and 'logAMount'. How are they different?

**Code:** import matplotlib.pyplot as plt

plt.hist(df\_loans\_out['LOAN\_AMOUNT'],edgecolor = 'white')

plt.hist(df\_loans\_out['logAmount'],edgecolor = 'white')

15. What is the total loan amount in each sector? Create a bar chart to demonstrate.

**Code:** #group by sector and sum the Loan Amount

sectoramount=loansAnalysis.groupby('SECTOR\_NAME')['LOAN\_AMOUNT'].sum()

# **Bar chart**:

sectoramount.plot(kind='bar')

#Agriculture sector has the highest amount.

16. What is the total number of loans in each country? Create a bar chart to demonstrate.

**Code:** #Group by country and count Loan IDs

loanscountry = loansAnalysis.groupby('COUNTRY\_NAME')['LOAN\_ID'].count()

#plt.figure(figsize=(8,10))

loanscountry.plot(kind='bar',figsize = (15,12))

# Country with the highest number of loans is phillipines

17. Change the POSTED\_TIME to DateTimeIndex and create a new variable 'Year' using the year of POSTED\_TIME. What is the total number of loans in each year? Create a line graph to demonstrate.

**Code:** loansAnalysis['Year']=pd.DatetimeIndex(loansAnalysis['POSTED\_TIME']).year

loansAnalysis.head()

# Group by and county the number of loans

yearAmount = loansAnalysis.groupby('Year')['LOAN\_ID'].count()

yearAmount.plot()

18. Create a pivot table to show the average loan amount in each sector each year.

**Code:** loansAnalysis.pivot\_table('LOAN\_AMOUNT',index='SECTOR\_NAME',columns='Year')

19. Get the number of days in the TIME\_LENGTH variable we created and create a new variable 'Days'using it. Create a scatterplot of 'LOAN\_AMOUNT' and 'Days'. What's your observation?

**Code:** loansAnalysis['Days']=loansAnalysis['TIME\_LENGTH'].dt.days

20. Create a scatterplot of 'LOAN\_AMOUNT' and 'LENDER\_TERM'. What's your observation?

**Code:** plt.figure(figsize=(15,12.))

plt.scatter(loansAnalysis['LENDER\_TERM'],loansAnalysis['LOAN\_AMOUNT'])

21. Prepare the data for model analysis. The target variable is 'STATUS'. The predictor variables we want to include:<br>

**Code:** ORIGINAL\_LANGUAGE<br>

the logAmount variable we created<br>

IMAGE\_ID (the one we coded to 0/1)<br>

SECTOR\_NAME

COUNTRY\_NAME

CURRENCY\_POLICY

CURRENCY\_EXCHANGE\_COVERAGE\_RATE (the one we tranfromed)<br>

Days we created based on the TIME\_LENGTH <br>

PREDISBURSE, we created<br>

LENDER\_TERM

NUM\_JOURNAL\_ENTRIES

NUM\_BULK\_ENTRIES

REPAYMENT\_INTERVAL

DISTRIBUTION\_MODEL<br>

Now we can drop all records with missing values at this point.

**Code:** loansAnalysis=loansAnalysis.dropna()

target=loansAnalysis['STATUS']

features = ['ORIGINAL\_LANGUAGE','logAmount','IMAGE\_ID','SECTOR\_NAME','COUNTRY\_NAME','CURRENCY\_POLICY','CURRENCY\_EXCHANGE\_COVERAGE\_RATE','Days','PREDISBURSE','LENDER\_TERM','NUM\_JOURNAL\_ENTRIES','NUM\_BULK\_ENTRIES','REPAYMENT\_INTERVAL','DISTRIBUTION\_MODEL']

**Code:** x=loansAnalysis[features]

22. Generate dummies for all categorical variables

**Code:** x.head()

# There are several categorical variables

cat\_vars=['ORIGINAL\_LANGUAGE','SECTOR\_NAME','COUNTRY\_NAME','CURRENCY\_POLICY','PREDISBURSE','REPAYMENT\_INTERVAL','DISTRIBUTION\_MODEL']

x=pd.get\_dummies(x,columns=cat\_vars,drop\_first= True)

x.head()

23. Prepare training and test datasets

**Code:** from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,target,test\_size=0.33,random\_state=1)

24. Fit a logistic regression model.

**Code:** from sklearn.linear\_model import LogisticRegression

logRegressor = LogisticRegression()

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

25. Get the predicted results for the test dataset

**Code:** y\_pred = logRegressor.predict(x\_test)

26. Evaluate the model using accuracy rate and confusion matrix.

**Code:** from sklearn.metrics import accuracy\_score

accuracy\_score(y\_true=y\_test,y\_pred=y\_pred)

from sklearn.metrics import confusion\_matrix

confusion\_matrix(y\_true=y\_test,y\_pred=y\_pred)

27. Repeat Step 24-26 using another classification method. Which method has better results?

**Code:** from sklearn.neighbors import KNeighborsClassifier

Extra question: In your initial project report, you have proposed several descriptive questions. Pick one to explore.

What is the total count of lenders and borrowers in the USA?

**Code:** test = df\_loans\_out[df\_loans\_out['COUNTRY\_name'] == 'United States']

test

#df\_loans\_out['COUNTRY\_name']

sum(test.NUM\_LENDERS\_TOTAL)

sum(test.NUM\_JOURNAL\_ENTRIES)

**References:**

<https://www.kiva.org/build/data-snapshots>

<https://www.kaggle.com/kiva/data-science-for-good-kiva-crowdfunding>