BCIS 5110 – PROGRAMMING LANGUAGES FOR BUSINESS ANALYTICS

PROJECT FINAL REPORT

LAKSHMI NARAYANA MUTHE (11616657)

Executive summary

This report presents an analysis of the data collected from Kiva, a global non-profit organization dedicated to providing financial services to underserved and marginalized communities. The analysis of the data focuses on understanding the characteristics of the borrowers, lenders and lending teams, as well as the impact of Kiva's partners on their lending activities.The process involves, data preprocessing, and data mining techniques to understand lender behaviors.

PROJECT MOTIVATION/BACKGROUND

The preparation and analysis of Kiva data is a critical step in understanding the impact and effectiveness of the organization's microfinance activities. Kiva's data is collected from a variety of sources, including its global network of field partners, its website, and other sources such as surveys. This data includes information about the borrowers, their businesses, and the loan terms. It also includes socio-economic indicators such as gender, location, and industry. Preparation of Kiva data involves collecting, cleaning, and organizing the data into a format that can be used for analysis. This includes removing errors and inconsistencies, standardizing data formats, and ensuring that the data is in a format that is suitable for analysis.

The analysis of Kiva data can provide valuable insights into the effectiveness of Kiva's microfinance activities (Pillai, 2015). This includes understanding the demographics of the borrowers, the types of businesses they are engaging in, and their repayment rates. It can also provide insights into the impact of Kiva loans on poverty reduction in the communities where they are provided. Analysis of Kiva data can be done using a variety of methods, including descriptive statistics, predictive analytics, machine learning algorithms, and other data mining techniques (Ghosh, 2017). These methods can be used to identify patterns and trends in the data, which can then be used to inform decisions about Kiva's microfinance operations.

The project motivation was to find if borrowers have plans to repay their loans, in the upcoming years, what the payback rate will be.

DATA DESCRIPTION

The loans data frame had 2087236 and 34 rows; it had the following columns:

```
: loans.columns
```

```
: Index(['LOAN_ID', 'LOAN_NAME', 'ORIGINAL_LANGUAGE', 'DESCRIPTION',
       'DESCRIPTION_TRANSLATED', 'FUNDED_AMOUNT', 'LOAN_AMOUNT', 'STATUS',
       'IMAGE_ID', 'VIDEO_ID', 'ACTIVITY_NAME', 'SECTOR_NAME', 'LOAN_USE',
       'COUNTRY_CODE', 'COUNTRY_NAME', 'TOWN_NAME', 'CURRENCY_POLICY',
       'CURRENCY_EXCHANGE_COVERAGE_RATE', 'CURRENCY', 'PARTNER_ID',
       'POSTED_TIME', 'PLANNED_EXPIRATION_TIME', 'DISBURSE_TIME',
       'RAISED_TIME', 'LENDER_TERM', 'NUM_LENDERS_TOTAL',
       'NUM_JOURNAL_ENTRIES', 'NUM_BULK_ENTRIES', 'TAGS', 'BORROWER_NAMES',
       'BORROWER_GENDERS', 'BORROWER_PICTURED', 'REPAYMENT_INTERVAL',
       'DISTRIBUTION_MODEL'],
      dtype='object')
```

The dataset variable the following datatypes:

```
 #   Column                            Dtype
---  ------                            -----
 0   LOAN_ID                           int64
 1   LOAN_NAME                         object
 2   ORIGINAL_LANGUAGE                 object
 3   DESCRIPTION                       object
 4   DESCRIPTION_TRANSLATED            object
 5   FUNDED_AMOUNT                     float64
 6   LOAN_AMOUNT                       float64
 7   STATUS                            object
 8   IMAGE_ID                          float64
 9   VIDEO_ID                          float64
 10  ACTIVITY_NAME                     object
 11  SECTOR_NAME                       object
 12  LOAN_USE                          object
 13  COUNTRY_CODE                      object
 14  COUNTRY_NAME                      object
 15  TOWN_NAME                         object
 16  CURRENCY_POLICY                   object
 17  CURRENCY_EXCHANGE_COVERAGE_RATE   float64
 18  CURRENCY                          object
 19  PARTNER_ID                        float64
 20  POSTED_TIME                       object
 21  PLANNED_EXPIRATION_TIME           object
 22  DISBURSE_TIME                     object
 23  RAISED_TIME                       object
 24  LENDER_TERM                       float64
 25  NUM_LENDERS_TOTAL                 int64
 26  NUM_JOURNAL_ENTRIES               int64
 27  NUM_BULK_ENTRIES                  int64
 28  TAGS                              object
 29  BORROWER_NAMES                    object
 30  BORROWER_GENDERS                  object
 31  BORROWER_PICTURED                 object
 32  REPAYMENT_INTERVAL                object
 33  DISTRIBUTION_MODEL                object
```

DATA PREPARATION

We checked for number of missing values in loans data frame using is null sum function, below was the output of the function:

```
loans.isnull().sum()
```

LOAN_ID	0
LOAN_NAME	50238
ORIGINAL_LANGUAGE	45891
DESCRIPTION	45926
DESCRIPTION_TRANSLATED	455312
FUNDED_AMOUNT	0
LOAN_AMOUNT	0
STATUS	0
IMAGE_ID	45891
VIDEO_ID	2086058
ACTIVITY_NAME	0
SECTOR_NAME	0
LOAN_USE	45914
COUNTRY_CODE	34
COUNTRY_NAME	0
TOWN_NAME	176356
CURRENCY_POLICY	0
CURRENCY_EXCHANGE_COVERAGE_RATE	473884
CURRENCY	0
PARTNER_ID	21175
POSTED_TIME	0
PLANNED_EXPIRATION_TIME	371834
DISBURSE_TIME	4015
RAISED_TIME	102404
LENDER_TERM	24
NUM_LENDERS_TOTAL	0
NUM_JOURNAL_ENTRIES	0
NUM_BULK_ENTRIES	0
TAGS	943057
BORROWER_NAMES	50238
BORROWER_GENDERS	45891
BORROWER_PICTURED	45891
REPAYMENT_INTERVAL	0
DISTRIBUTION_MODEL	0

We determined number of missing values in VIDEO_ID column, percentage of missing values was determined and found to be more than 99%, and the column was dropped because of high number of missing values. Below is how missing values in VIDEO_ID column was determined:

Number of missing values in VIDEO_ID

```
]: x=loans[['VIDEO_ID']]#assigning VIDEO_ID to data frame
   yVIDEO_ID=x.isnull().sum().sum() #summing number of missing values in VIDEO_ID col
   yVIDEO_ID #display the sum
```

```
]: 2086058
```

What's the percentage?

```
]: percentage=(yVIDEO_ID/(len(loans['VIDEO_ID'])))*100
   print('The percentage is {}%'.format(percentage))
```

```
The percentage is 99.94356172469236%
```

```
]: # 2086058 missing; divided by total number of loans
   2086058/2087236 # More than 99.9% missing
   # We cannot use this variable. Delete it.
```

```
]: 0.9994356172469236
```

We cannot use VIDEO_ID attribute because it has high percentage of missing values

```
]: loans.drop('VIDEO_ID', axis = 1, inplace = True)
```

We dropped some of the columns to reduce the dimension of the data frame as shown below:

```
: loans.drop(['LOAN_NAME', 'LOAN_USE', 'COUNTRY_CODE', 'TOWN_NAME', 'PARTNER_ID', 'PLANNED_EXPIRATION_TIME', 'TAGS', 'BORROWER_NAME
  loans.columns
```

```
: Index(['LOAN_ID', 'ORIGINAL_LANGUAGE', 'DESCRIPTION', 'DESCRIPTION_TRANSLATED',
         'FUNDED_AMOUNT', 'LOAN_AMOUNT', 'STATUS', 'IMAGE_ID', 'ACTIVITY_NAME',
         'SECTOR_NAME', 'COUNTRY_NAME', 'CURRENCY_POLICY',
         'CURRENCY_EXCHANGE_COVERAGE_RATE', 'CURRENCY', 'POSTED_TIME',
         'DISBURSE_TIME', 'RAISED_TIME', 'LENDER_TERM', 'NUM_LENDERS_TOTAL',
         'NUM_JOURNAL_ENTRIES', 'NUM_BULK_ENTRIES', 'BORROWER_GENDERS',
         'BORROWER_PICTURED', 'REPAYMENT_INTERVAL', 'DISTRIBUTION_MODEL'],
        dtype='object')
```

We checked dependent variable to determine unique values in the variable, the column had four unique values as shown below:

```
loans['STATUS'].unique()
# There are four unique categories.
```

```
array(['funded', 'expired', 'refunded', 'fundRaising'], dtype=object)
```

We explored the obsevations in each category of status column and the observations were as shown in the below figure; the function that was used to determine observations in each category of status column was value-counts function.

```
: loans['STATUS'].value_counts()
```

```
: funded         1978361
  expired          92416
  refunded          9182
  fundRaising       7277
  Name: STATUS, dtype: int64
```

funded, refunded, and expired categories suggested that loan was funded. We filtered out all the funRaising records from the data frame.

The missing values for IMAGE_ID column ment that image was not provided. We filled the missing values with 0 if image was not provided and 1 when image was provided. To confirm that missing values had been filled, We checked for unique values in the column as shown below:

We can consider recode this variable to indicate whether there is image provided. Replace the missing value with 0 and change all IMAGE_ID to 1.

```
4]: # image_id: we can recode this variable to indicate whether the loan has an image.
    # All missing values indicate there is no image, so we code them as 0.
    # All with image ID indicates there is image, so we code them as 1.
    # We can do it with a lambda function
    loans['IMAGE_ID'] = loans['IMAGE_ID'].apply(lambda x: 1 if x>0 else 0)
    loans['IMAGE_ID'].unique()

    # Graders: students may have used other methods.
```

```
4]: array([1, 0])
```

We converted POSTED_TIME, RAISED_TIME and DISBURSE_TIME to datetime data type as shown below:

```
loans['DISBURSE_TIME'] = pd.to_datetime(loans['DISBURSE_TIME'])
loans['DISBURSE_TIME'].dtype
```

```
datetime64[ns, UTC]
```

```
loans['RAISED_TIME'] = pd.to_datetime(loans['RAISED_TIME'])
loans['RAISED_TIME'].dtype
```

```
datetime64[ns, UTC]
```

```
loans['POSTED_TIME'] = pd.to_datetime(loans['POSTED_TIME'])
loans['POSTED_TIME'].dtype
```

```
datetime64[ns, UTC]
```

We assigned POST_TIME to year column as shown below:

```
loans['year'] = loans['POSTED_TIME'].dt.year
loans.tail(15)
```

	LOAN_ID	ORIGINAL_LANGUAGE	DESCRIPTION	DESCRIPTION_TRANSLATED	FUNDED_AMOUNT	LOAN_AMOUNT	STATUS	IMAGE_ID	ACTI
2087221	199046	French	Le groupe de solidarité KOUTEDIOMBOULOU fait p...	The Koutediomboulou solidarity group is one of...	950.0	950.0	funded	1	Prod
2087222	266714	English	Narcisa M. is from the village of Bangko,Pana-...	NaN	125.0	125.0	funded	1	
2087223	2071629	English	Ana is an artisan from Peru, who specializes i...	Ana is an artisan from Peru, who specializes i...	5000.0	5000.0	funded	1	
2087224	1572520	English	Abduljalal is married and lives with his famil...	Abduljalal is married and lives with his famil...	175.0	175.0	funded	1	
2087225	332330	English	Jayson is 29, married and has two infant-age c...	NaN	175.0	175.0	funded	1	
2087226	260163	Spanish	Carmen Beatriz C. P. tiene 47 años. Es casada,...	Carmen Beatriz C. P. is 47 years old. She is m...	1000.0	1000.0	funded	1	
2087227	1156700	Spanish	Maria es una mujer de 50 años de edad, emprend...	Maria is 50 years old. She is an entrepreneur ...	825.0	1050.0	notfunded	1	
2087228	1187832	English	Pictured above is Willimina, often described a...	Pictured above is Willimina, often described a...	550.0	550.0	funded	1	
2087229	928183	English	Dauda is a Babban Gona maize farmer from Kargo	Dauda is a Babban Gona maize farmer from Kargo...	75.0	75.0	funded	1	

I made a calculated attribute time_used from difference between RAISED_TIME and POSTED_TIME. The result was as shown below:

```
]: loans['time_used'] =  (loans['RAISED_TIME'] - loans['POSTED_TIME']).dt.days
   loans['time_used'].head()
```

```
]: 0     3.0
   1    19.0
   2     4.0
   3     2.0
   4     6.0
   Name: time_used, dtype: float64
```

We filled all the missing values in CURRENCY_EXCHANGE_COVERAGE_RATE with 0 as shown below:

Replace the missing values of this variable with 0.

```
]: loans['CURRENCY_EXCHANGE_COVERAGE_RATE'].unique() # only several values
   # There are 473884 missing values, about 23% missing.
   loans['CURRENCY_EXCHANGE_COVERAGE_RATE'].fillna(0, inplace = True)
   print('\nConfirming if missing values have been fiiled, the number of missing values is {}'.format(loans['CURRENCY_EXCHANGE_COVER
```

```
]: array([0.1, 0.2, nan, 0. ])

   Confirming if missing values have been fiiled, the number of missing values is 0
```

EXPLORATORY DATA ANALYSIS

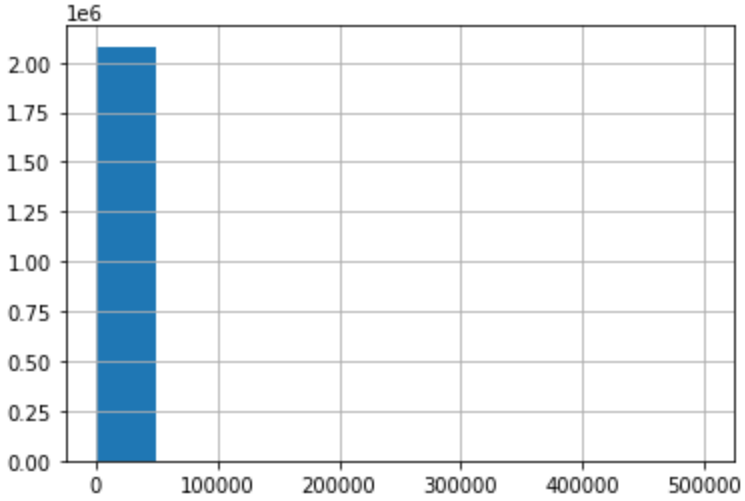
We explored statistical distribution on dataset using describe function. We determined the statisti cal distribution of LOAN_AMOUNT column and the minimum values was found to be 2.500000 e+01, maximum value was 5.000000e+05, and the median was 2.079959e + 06. The IQR was 7.0 00000e+02. The cut-of value for outliers was 3100.0. The result of the function was as shown bel ow:

```
: loans['LOAN_AMOUNT'].describe()

: count    2.079959e+06
  mean     8.151277e+02
  std      1.667618e+03
  min      2.500000e+01
  25%      2.750000e+02
  50%      5.000000e+02
  75%      9.750000e+02
  max      5.000000e+05
  Name: LOAN_AMOUNT, dtype: float64
```

We made a histogram for the variable LOAN_AMOUNT', y axis had frequency of the amount in the range, while the x had the amount range. Below was the display of the histogram:

```
loans['LOAN_AMOUNT'].hist()
```

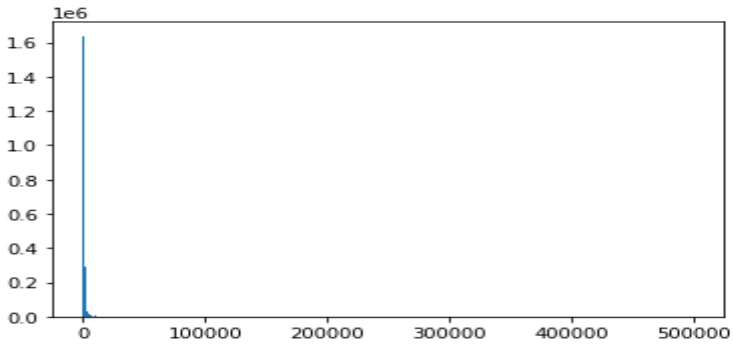
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fcdcddf1460>
```



We filtered loans of the amount less than or equal to 10000 and plot histogram with 500 bins. The observation from histogram result was that the bar ranged from 0 to 10000 and frequency ranged to 500. The result was as shown below:

```
import matplotlib.pyplot as plt
loans.loc[loans['LOAN_AMOUNT'] <= 10000]
plt.hist(loans['LOAN_AMOUNT'], bins=500)
```

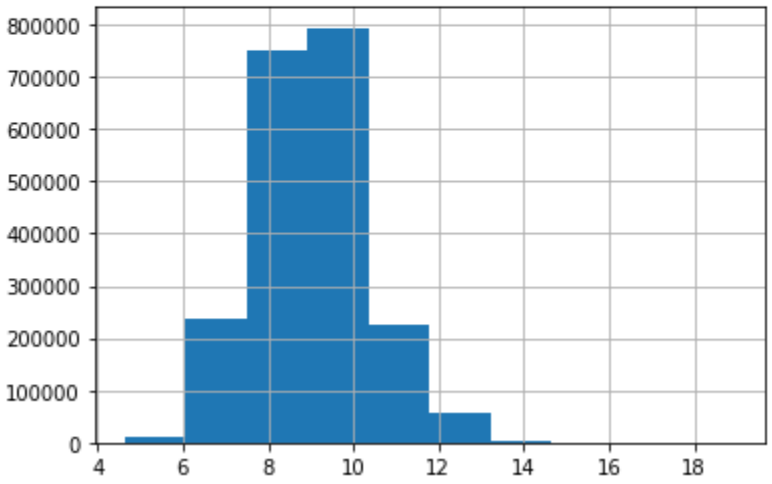
```
       5.0000000e+05]),
 <a list of 500 Patch objects>)
```



We determined the natural log of LOAN_AMOUNT variable and made its histogram plot, the plot was observed to have seven bars, with x axis bar range from 5 to 15. The result was as shown below:

```
: loans['logAmount'] = np.log2(loans['LOAN_AMOUNT'])
  loans['logAmount'].hist()
```

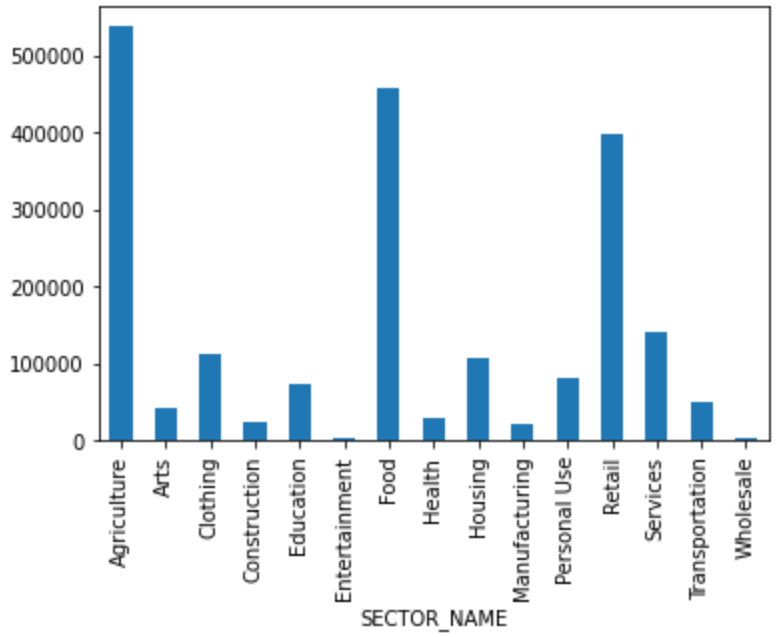
```
: <matplotlib.axes._subplots.AxesSubplot at 0x7fcdcd6411f0>
```



We used bar plot to visualize the number of LOAN_ID in each sector. the plot was as shown below:

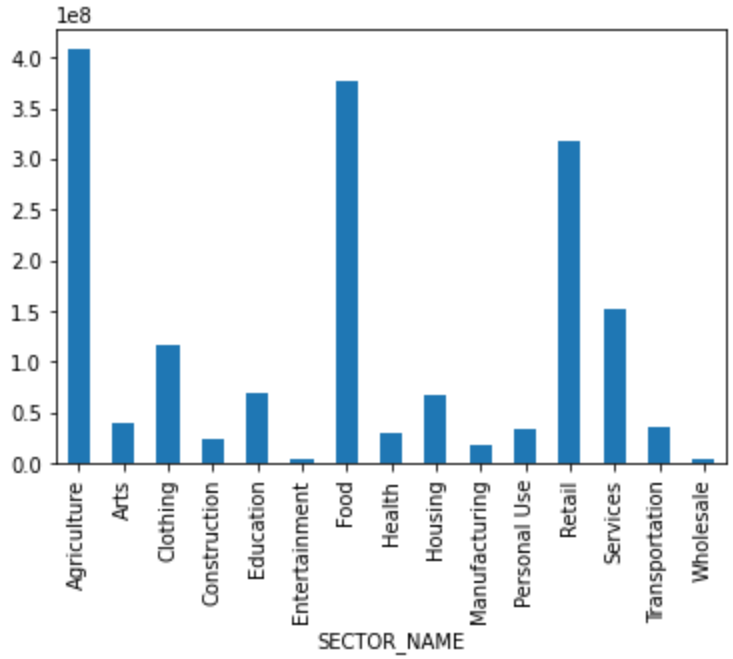
```
: loans.groupby('SECTOR_NAME')['LOAN_ID'].count().plot(kind='bar')
```

```
: <matplotlib.axes._subplots.AxesSubplot at 0x7fcdcd634880>
```



We used bar plot to visualize total amount in each sector as shown below:

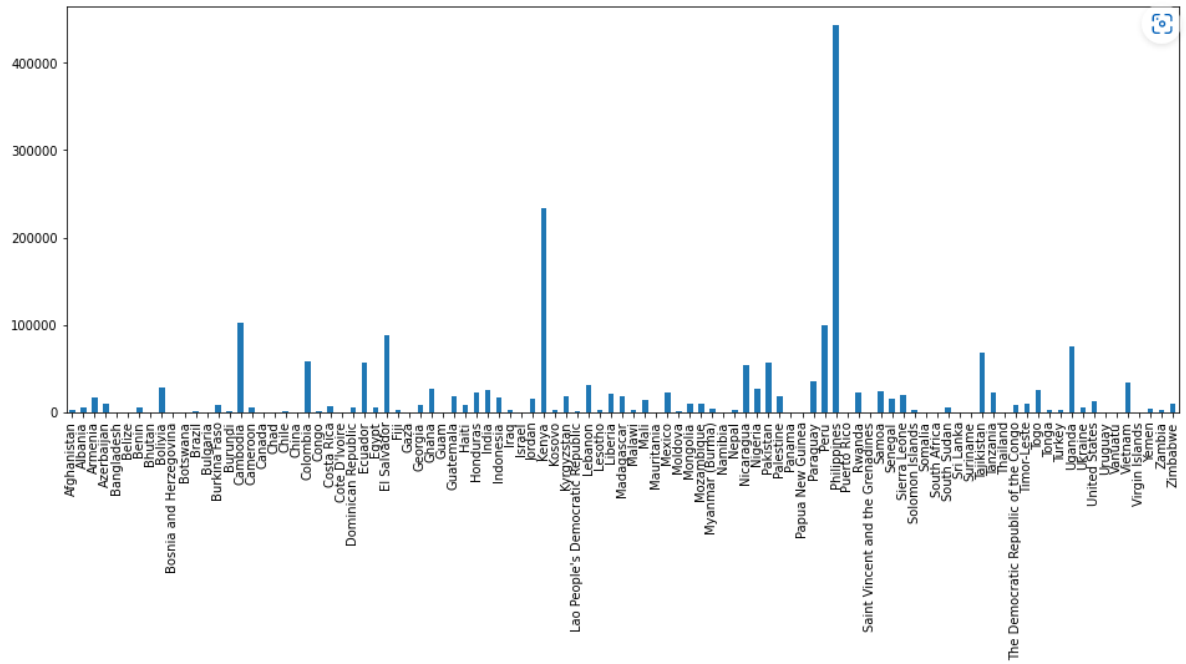
```
: loans.groupby('SECTOR_NAME')['LOAN_AMOUNT'].sum().plot(kind='bar')

: <matplotlib.axes._subplots.AxesSubplot at 0x7fcdcd5c5880>
```



We visualized number of loans in each country using bar plot as shown below:

```
fig = plt.figure(figsize = (16, 6))
loans.groupby('COUNTRY_NAME')['LOAN_ID'].count().plot(kind = 'bar')
```

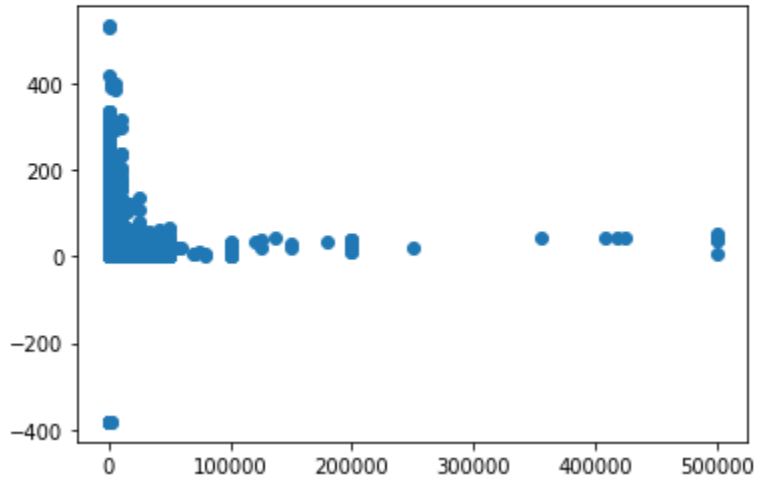
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fcdcd3c5b50>
```



We created a scatter plot for time_used variable and LOAN_AMOUNT and observed loan amount between 0 to 100000 was mostly take for a period of 0 to 400 days. Below was the result of the plot:

```
plt.scatter(loans['LOAN_AMOUNT'], loans['time_used'])
plt.show()
```

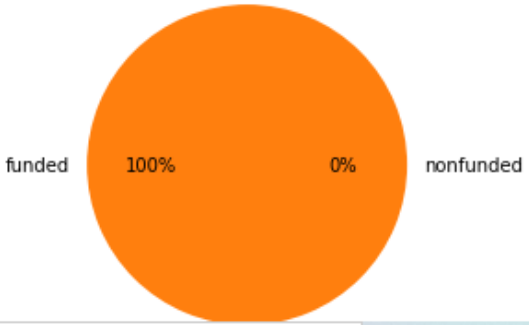
```
<matplotlib.collections.PathCollection at 0x7fcdcd1cddf0>
```



We determined the average amount of funded and non-refunded loans and plotted them on a pie chart. It was observed that funded loans had 100% average while non-refunded had 0% as shown below:

```
sumnonfunded=0
nonfundedf=loans.loc[loans['STATUS'] == 'nonfunded']
for x in nonfundedf['LOAN_AMOUNT']:
  sumnonfunded+=x
avgnf=sumnonfunded/len(loans['LOAN_AMOUNT'])
fundedf=loans.loc[loans['STATUS'] == 'funded']
sumfunded=0
for x in fundedf['LOAN_AMOUNT']:
  sumfunded+=x
avgf=sumfunded/len(loans['LOAN_AMOUNT'])
y = np.array([avgnf, avgf])

plt.pie(y, labels=['nonfunded', 'funded'], autopct='%.0f%%')
plt.show()
```

```
([<matplotlib.patches.Wedge at 0x7fcdcd0a92e0>,
  <matplotlib.patches.Wedge at 0x7fcdcd0a9790>],
 [Text(1.1, 0.0, 'nonfunded'), Text(-1.1, 1.3471114790620887e-16, 'funded')],
 [Text(0.6, 0.0, '0%'), Text(-0.6, 7.347880794884119e-17, '100%')])
```



MODELS AND ANALYSIS

To train the model, we had to first extract dependent and independent variables as y and X. We encoded categorical values using get dummy function. We split the data frame into train and test set using train test split function. We import Decision tree classifier and initialized it. We fitted the model with training set and made prediction using test. We compared the predicted values and actual values, we evaluated the model performance using metric score and confusion matrix, the model had accuracy score of 95%. The process was as shown below:

28. Prepare y (target) and X (features) using the dataset we get from Q27.

```
y=modeldf['STATUS']
```

```
X= modeldf[['IMAGE_ID','SECTOR_NAME','COUNTRY_NAME','CURRENCY_EXCHANGE_COVERAGE_RATE', 'LENDER_TERM','NUM_JOURNAL_ENTRIES', 'NUM_
```

29. Transform all categorical variables into dummy variables.

```
X= pd.get_dummies(X)
X
```

	IMAGE_ID	CURRENCY_EXCHANGE_COVERAGE_RATE	LENDER_TERM	NUM_JOURNAL_ENTRIES	NUM_BULK_ENTRIES	predisburse	femalePer	logAm
0	1	0.1	14.0	5	3	True	1.000000	8.22
1	1	0.1	13.0	2	1	True	1.000000	7.96
2	1	0.1	14.0	3	2	True	1.000000	7.64
3	1	0.1	8.0	2	1	True	1.000000	8.22
4	1	0.2	11.0	2	1	True	1.000000	8.55
...	...	...	...	...	...	...	...	
205012	1	0.2	8.0	2	1	True	0.022727	11.85
205013	1	0.0	10.0	1	1	False	1.000000	9.10
205014	1	0.0	14.0	1	1	True	0.000000	8.45
205015	1	0.0	7.0	1	1	True	1.000000	7.45
205016	1	0.1	21.0	1	1	True	1.000000	10.31

30. Prepare the training and test datasets

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test  = train_test_split(X,y, test_size = 0.2, random_state = 0)
```

30. Fit a decision tree classifier.

```
from sklearn.tree import DecisionTreeClassifier
# Create Decision Tree classifer object
clf = DecisionTreeClassifier()

# Train Decision Tree Classifer
clf = clf.fit(x_train,y_train)
```

31. Get the predicted results for the test dataset

```
#Predict the response for test dataset
y_pred = clf.predict(x_test)
pred = pd.DataFrame({'Actual':y_test.tolist(),'Predicted':y_pred.tolist()}).head(25)
pred.head(10)
```

	Actual	Predicted
0	funded	funded
1	funded	funded
2	funded	funded
3	funded	funded
4	funded	funded
5	funded	funded
6	funded	funded
7	funded	funded
8	funded	notfunded
9	notfunded	funded

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

```
: from sklearn import metrics
  print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.950675
```

```
: #confusion matrix
  metrics.confusion_matrix(y_test, y_pred)
```

```
: array([[37327,   775],
         [ 1198,   700]])
```

FINDINGS AND MANAGERIAL IMPLICATIONS

We found out that there was no loan amount with non-funded status. We found out that loans were highly taken in the between the year 2016 and 2020. We found out that Philippines had highest number of loan takers. We found out that loan takers from agriculture sector took higher amount than any other sector.

Data preparation and analysis is an essential process for any organization, and Kiva is no exception. Kiva's data preparation and analysis processes are of utmost importance in order to ensure that their loan operations run smoothly and effectively (Çelik, Baykal, Memur. 2020). The managerial implications of Kiva's data preparation and analysis processes are significant. By understanding and analyzing the data, Kiva can make informed decisions about how to allocate resources and which areas to target for loan operations. This can help Kiva to ensure that its loan operations are efficient and effective. Furthermore, by understanding the trends and insights from the data, Kiva can develop strategies to improve its operations and better serve its customers.

CONCLUSION

Preparation and analysis of Kiva data is a vital part of understanding the impact of micro-finance and its effectiveness in alleviating poverty. The preparation and analysis of Kiva data is an important part of understanding the impact and effectiveness of micro-finance. By analyzing the data, researchers can identify trends and patterns in the usage of Kiva loans and the effectiveness of the micro-finance system. For example, researchers can identify which countries are receiving the most Kiva loans, which entrepreneurs are most successful at paying back their loans, and which types of businesses are receiving the most funding. This data can then be used to inform policy decisions and strategies for expanding access to micro-finance.

APPENDIX: ATTACH THE ANALYSIS NOTEBOOK:

BCIS5110_Assignment_10_11_(1) (1).ipynb

REFERENCES

Debamita Ghosh. (2017). ANALYSIS OF DATA. https://www.researchgate.net/publication/325484846_ANALYSIS_OF_DATA

Hilal Çelik, Nur Başer Baykal, Hale Nur Kılıç Memur. (2020). Qualitative Data Analysis and Fundamental Principles. **https://www.researchgate.net/publication/338923329_Qualitative_Data_Analysis_and_Fundamental_Principles**

Vijayamohanan Pillai N. (2015). Data Analysis and Interpretation. **https://www.researchgate.net/publication/282286137_Data_Analysis_and_Interpretation**