BUSINESS INTELLIGENCE KIVA LOANS

project report

AGENDA

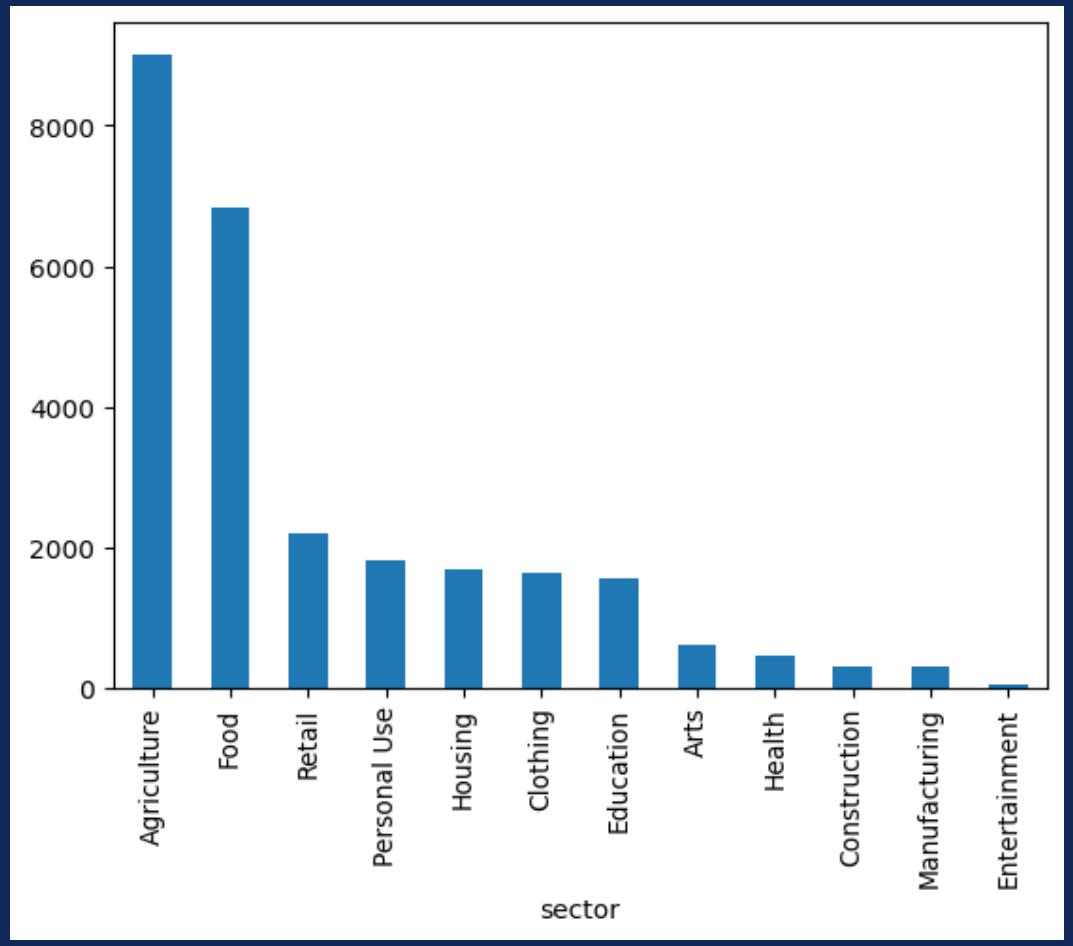
- Introduction
- Conclusion
- Code Break-down
- Summary

INTRODUCTION

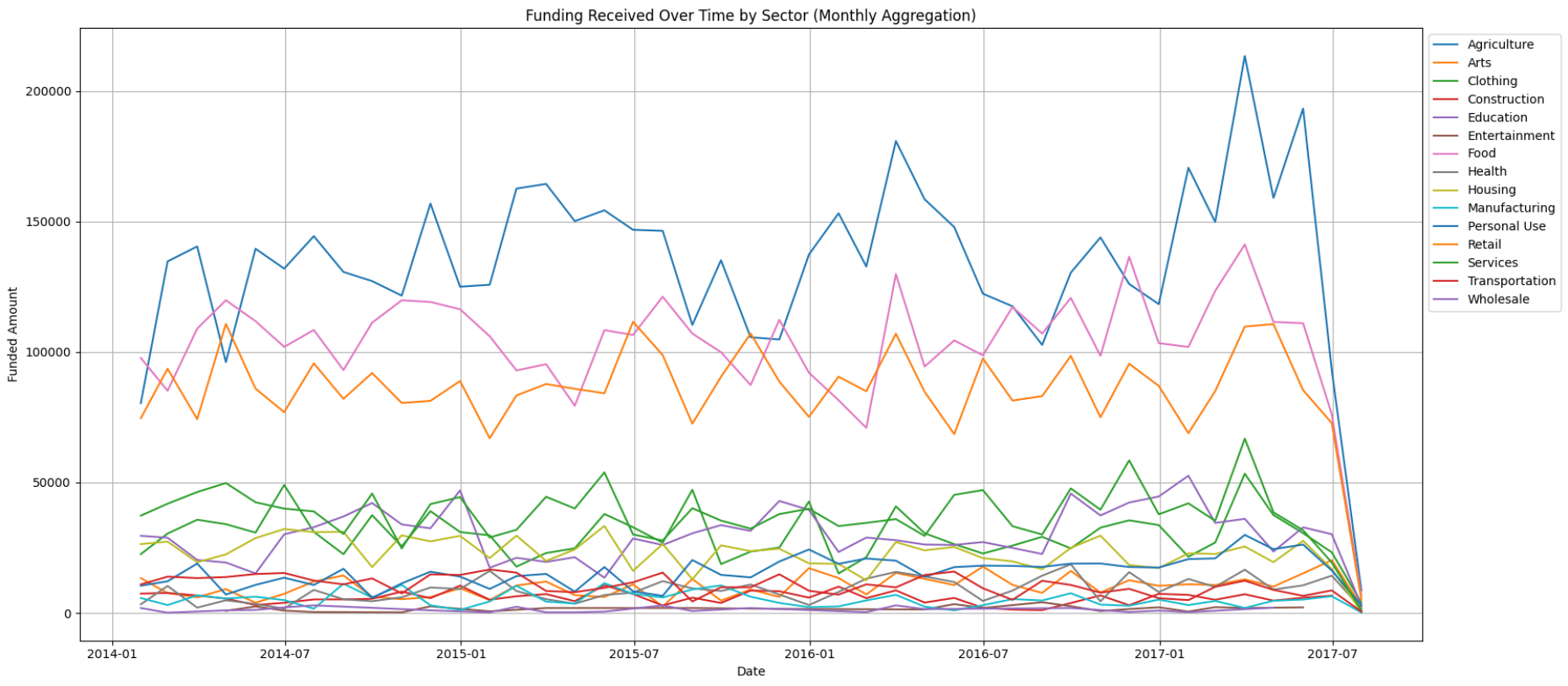
This documentation outlines the Business Intelligence project analyzing Kiva loans data to provide insights into funding patterns, lender behavior, and loan amount predictions.

CONCLUSION

SECTORS RECEIVED THE HIGHEST AMOUNT OF FUNDING

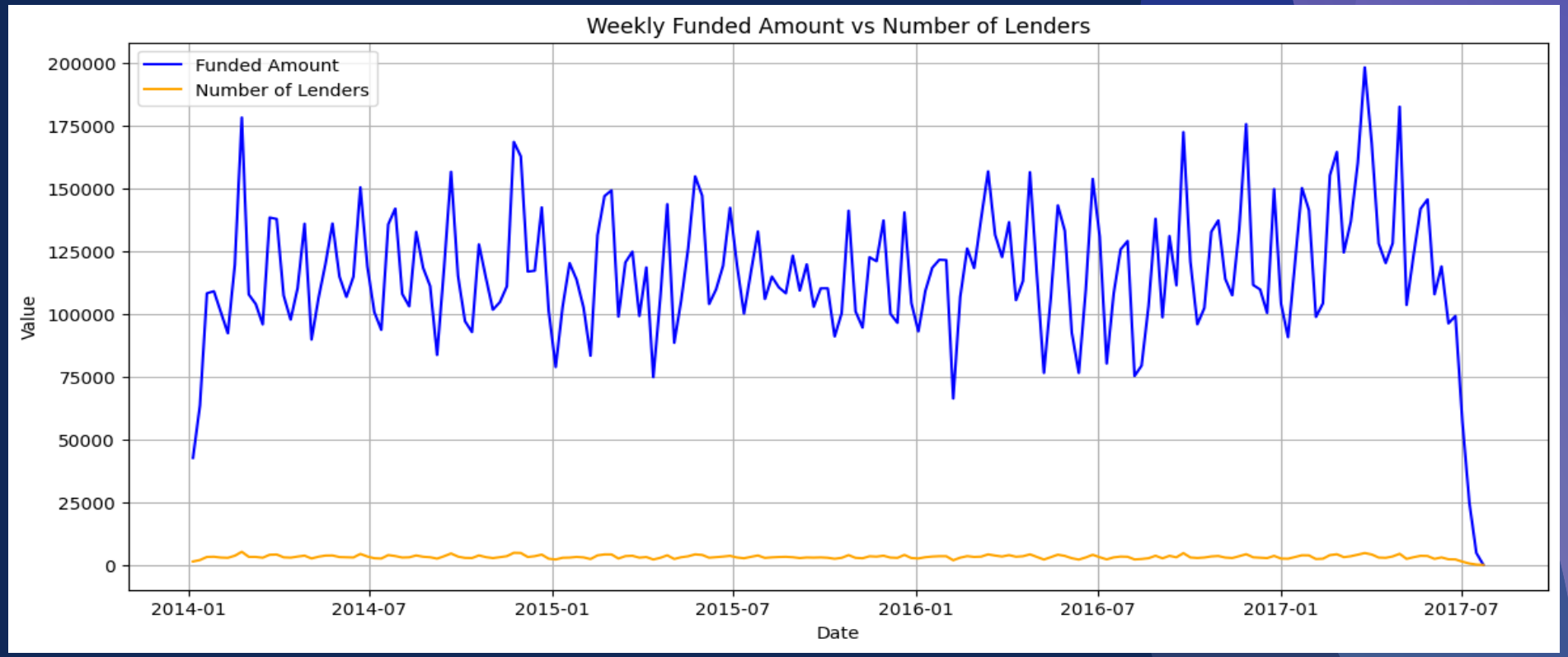


RECEIVED AMOUNT OF FUNDING CHANGE OVER TIME FOR EACH SECTOR(TIME SERIES)

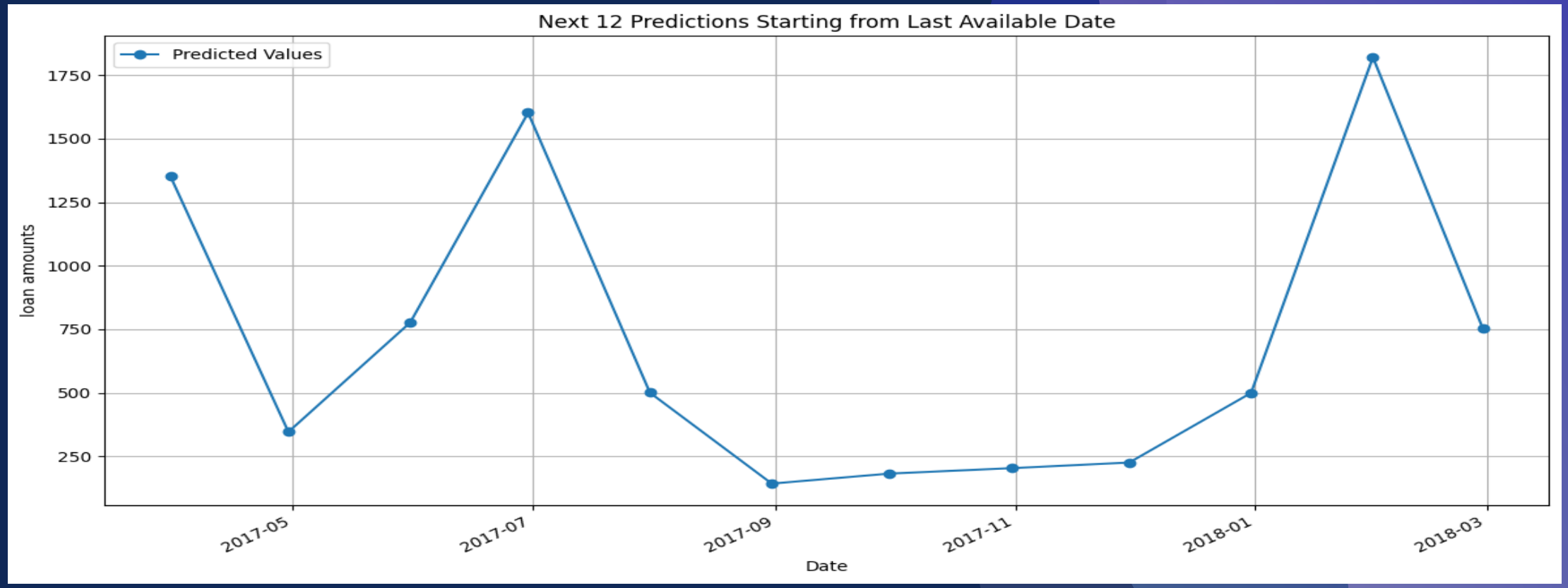


THE CORRELATION BETWEEN THE NUMBER OF LENDERS AND THE FUNDED LOAN AMOUNT (TIME SERIES)

Correlation between lender_count and funded_amount: 0.95



PREDICTION OF FUTURE LOAN AMOUNTS BASED ON HISTORICAL DATA (REGRESSION MODELS)



CODE BREAK-DOWN STEP BY STEP

exploration	Preprocessing	Visualization	Time Series	Prediction
Understand the dataset structure and initial characteristics	*Handle missing values, duplicates, and outliers*	*Visualize distributions, relationships, and trends Using Power BI*	*Analyze and forecast funded amount over time*	*Predict funded amount using ARIMA model*

DATA EXPLORATION

```
[5] df.head()
```

	id	funded_amount	loan_amount	sector	country	partner_id	term_in_months	lender_count	borrower_genders	repayment_interval	date
0	1242201	500	500	Agriculture	Pakistan	245.0	14	14	female	monthly	2/20/2017
1	1165778	325	325	Agriculture	Philippines	145.0	14	13	female	irregular	10/11/2016
2	1123052	800	800	Agriculture	Ecuador	159.0	14	29	female	bullet	7/25/2016
3	1312344	425	425	Agriculture	Philippines	136.0	8	1	female	irregular	6/2/2017
4	861422	275	275	Agriculture	Kenya	133.0	12	11	female	monthly	3/25/2015

```
[6] df.tail()
```

	id	funded_amount	loan_amount	sector	country	partner_id	term_in_months	lender_count	borrower_genders	repayment_interval	date
33556	886976	500	500	Wholesale	Pakistan	247.0	14	18	female	monthly	5/18/2015
33557	1017003	300	300	Wholesale	Pakistan	247.0	12	12	female	irregular	2/2/2016
33558	831649	225	225	Wholesale	Pakistan	421.0	14	9	female	monthly	1/23/2015
33559	920920	2000	2000	Wholesale	Palestine	80.0	27	54	male	monthly	7/21/2015
33560	931675	750	750	Wholesale	Pakistan	247.0	12	27	female, female	irregular	8/13/2015

DATA EXPLORATION

CHECKING FOR MISSING VALUES

Noted : partner id is not unique & contains null values

Borrower genders contains null values

```
df.isna().sum()
```

	0
id	0
funded_amount	0
loan_amount	0
sector	0
country	0
partner_id	634
term_in_months	0
lender_count	0
borrower_genders	195
repayment_interval	0
date	0

DATA EXPLORATION

CHECKING FOR *OUTLIERS*

using IQR method columns with Outliers detected

```
for col in df.select_dtypes(include=['int64', 'float64']):
  q1 = np.percentile(df[col].dropna(), 25)
  q3 = np.percentile(df[col].dropna(), 75)
  norm_range = (q3 - q1) * 1.5
  lower_outliers = df[df[col] < (q1 - norm_range)]
  upper_outliers = df[df[col] > (q3 + norm_range)]
  outliers = len(lower_outliers) + len(upper_outliers)
  print(f"The number of outliers in {col} is: {outliers}")
```

```
The number of outliers in id is: 0
The number of outliers in funded_amount is: 2790
The number of outliers in loan_amount is: 2654
The number of outliers in partner_id is: 2954
The number of outliers in term_in_months is: 3187
The number of outliers in lender_count is: 2724
```

DATA CLEANING & PREPROCESSING

FILLING MISSING VALUES WITH MODE

```
[ ]
    x = df['borrower_genders'].mode()[0]
    df['borrower_genders'].fillna(x, inplace=True)
    x = df['partner_id'].mode()[0]
    df['partner_id'].fillna(x, inplace=True)
```

CHECK FOR MISSING VALUES AGAIN

```
[ ] df.isna().sum()
```

	0
id	0
funded_amount	0
loan_amount	0
sector	0
country	0
partner_id	0
term_in_months	0
lender_count	0
borrower_genders	0
repayment_interval	0
date	0

DATA CLEANING & PREPROCESSING

Handling outliers using IQR method

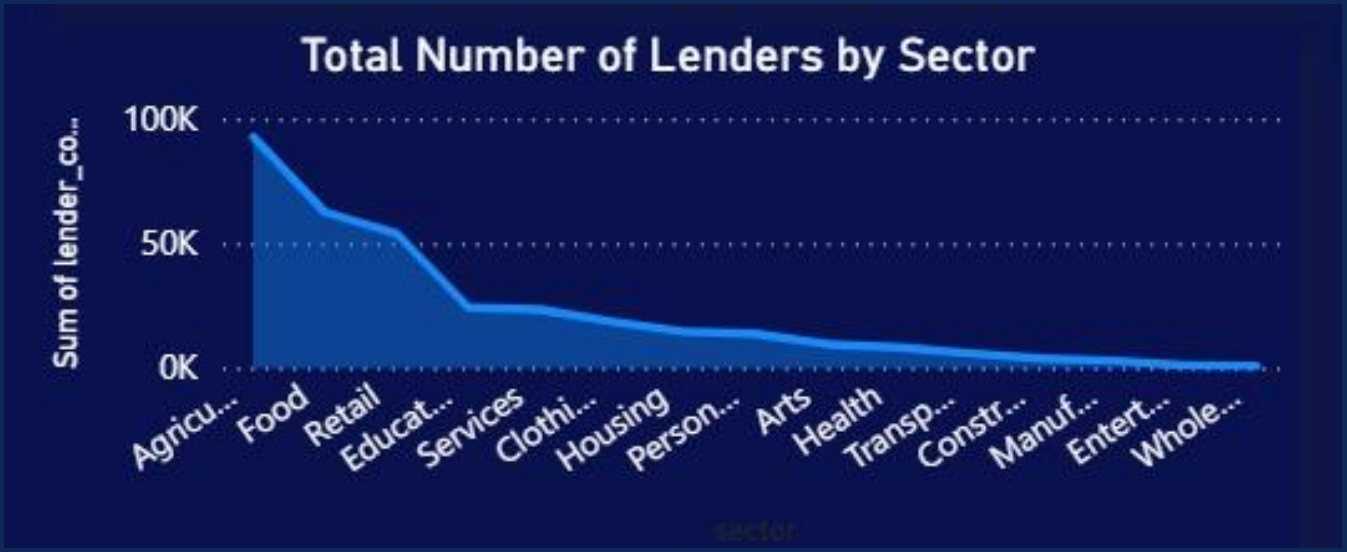
```
[ ]
    cols=[ 'funded_amount','loan_amount','partner_id','term_in_months','lender_count']
    for col in cols:
      q1 = np.percentile(df[col], 25)
      q3 = np.percentile(df[col], 75)
      norm_range = (q3 - q1) * 1.5
      df[col] = np.where(df[col] < (q1 - norm_range), q1 - norm_range, df[col])
      df[col] = np.where(df[col] > (q3 + norm_range), q3 + norm_range, df[col])
```

```
cols=[ 'funded_amount','loan_amount','partner_id','term_in_months','lender_count']
for col in cols:
  q1 = np.percentile(df[col], 25)
  q3 = np.percentile(df[col], 75)
  norm_range = (q3 - q1) * 1.5
  lower_outliers = df[df[col] < (q1 - norm_range)]
  upper_outliers = df[df[col] > (q3 + norm_range)]
  outliers = len(lower_outliers)+len(upper_outliers)
  print(f"The number of outliers in {col} is : {outliers}")
```

```
The number of outliers in funded_amount is : 0
The number of outliers in loan_amount is : 0
The number of outliers in partner_id is : 0
The number of outliers in term_in_months is : 0
The number of outliers in lender_count is : 0
```

VISUALIZATION

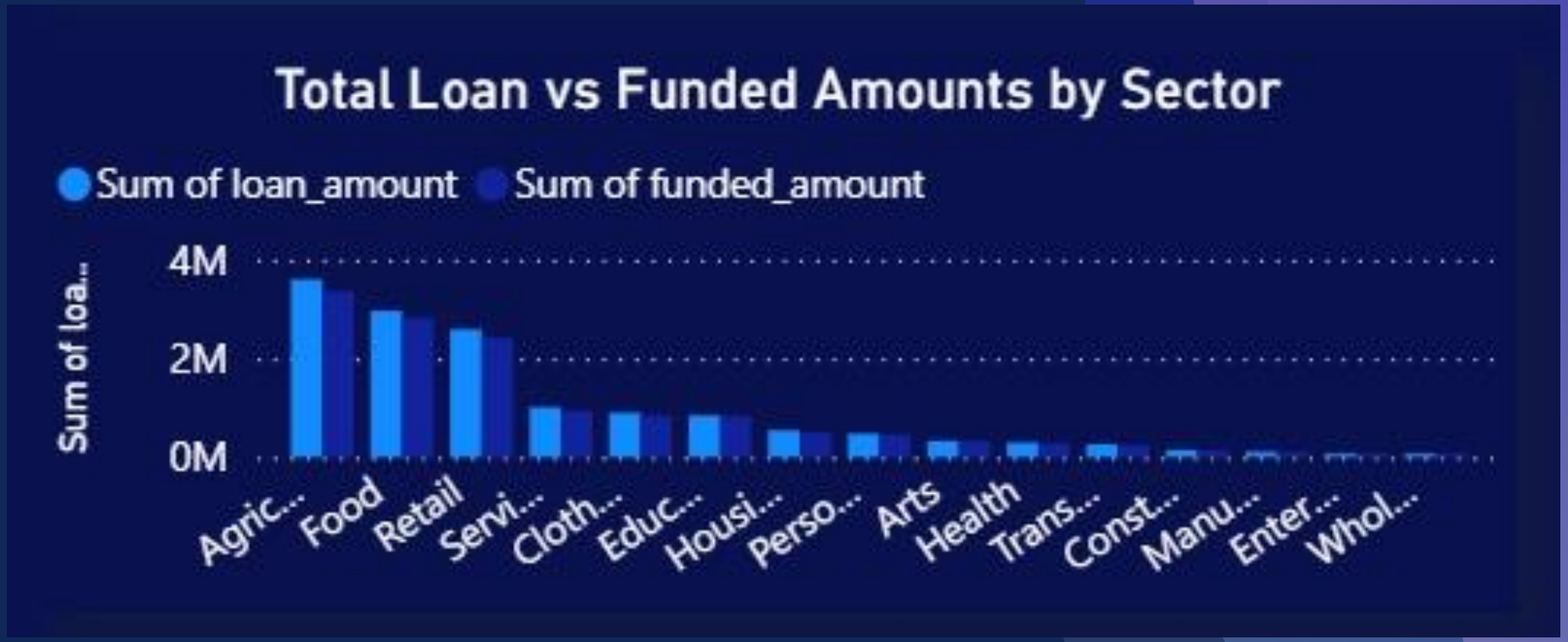
SECTORS WITH HIGHEST NUMBER OF LENDERS



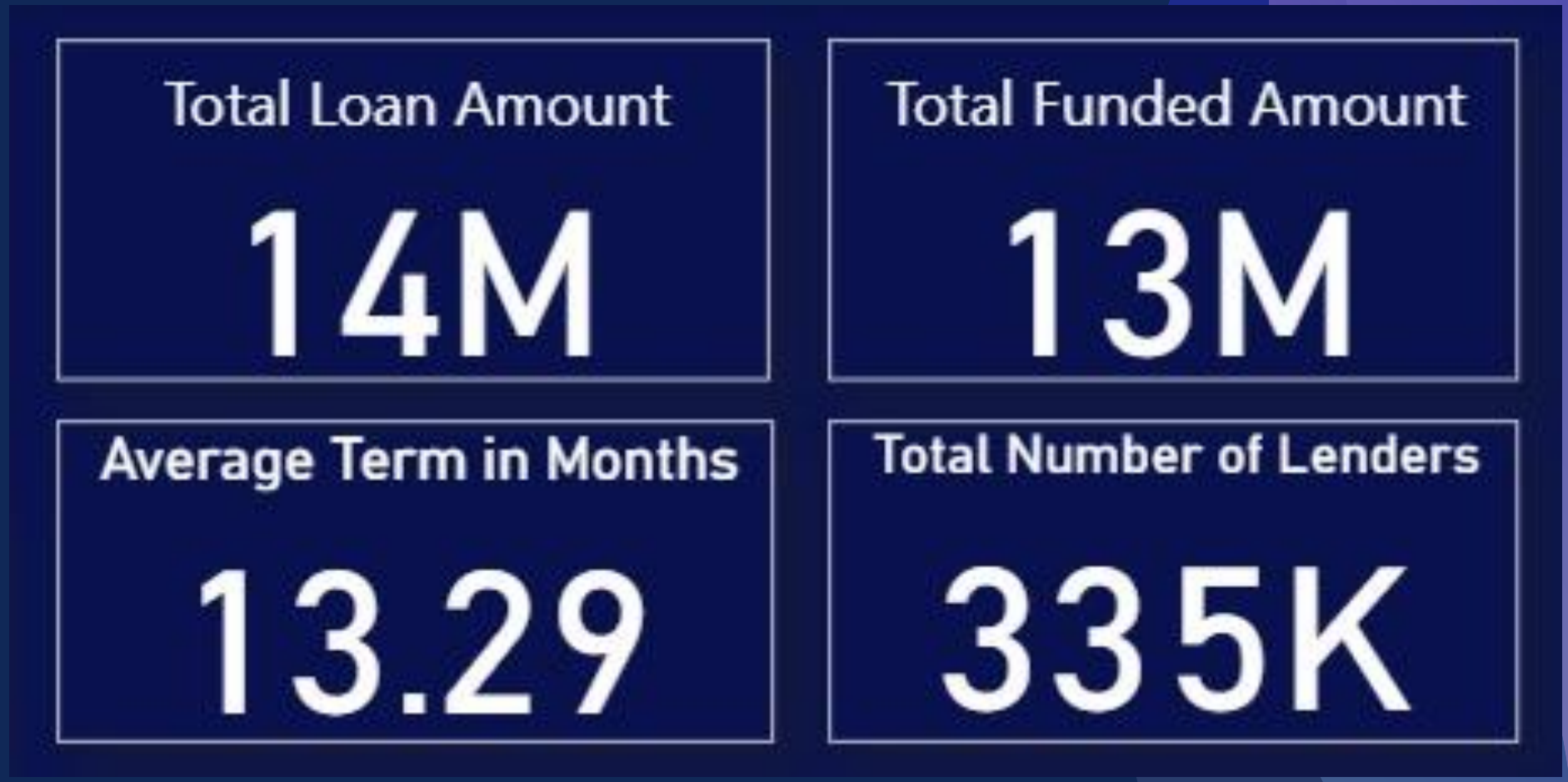
VISUALIZATION

GEOGRAPHICAL DISTRIBUTION TOP RECIPIENT COUNTRIES

VISUALIZATION



VISUALIZATION



TIME SERIES

First Make sure that the "Date" column is converted to Date Format.

Then , set the date column to be the index.

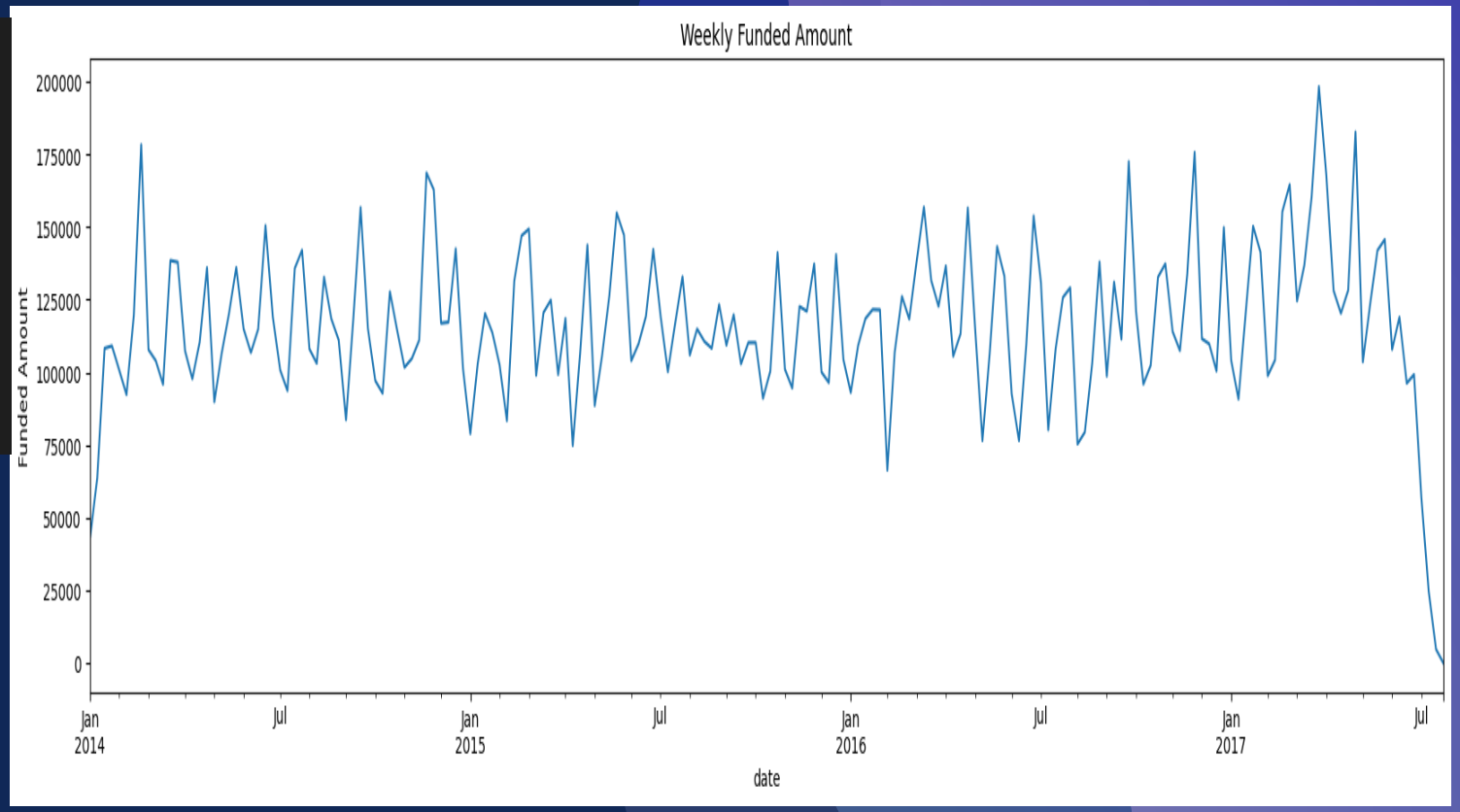
```
df['date'] = pd.to_datetime(df['date'])

# Then set this as the index
df = df.set_index('date')
ts = df['funded_amount'].resample('W').sum()
df = df.fillna(method='bfill')
plt.figure(figsize=(18,5))
ts.plot(title="Week Funded Amount")
plt.ylabel("Funded Amount")
plt.show()
```

TIME SERIES

- Showing Weekly Founded Amount Over Time

```
plt.figure(figsize=(20,5))
ts.plot(title="Weekly Funded Amount")
plt.ylabel("Funded Amount")
plt.show()
```



TIME SERIES

- Making the ADF Test to make sure that the Time series is stationary or not .
- Applying differencing Two Times to reduce the value of the P .
- The Time Series become more stationary .

```
result = adfuller(ts.dropna())
print("ADF Statistic:", result[0])
print("p-value:", result[1])
ts_diff = ts.diff().diff().dropna()
result = adfuller(ts_diff)
print("ADF Statistic after differencing:", result[0])
print("p-value:", result[1])
```

```
ADF Statistic: -2.9539090237812387
p-value: 0.039428700063465535
ADF Statistic after differencing: -8.162098896394967
p-value: 9.085718073454935e-13
```

TIME SERIES

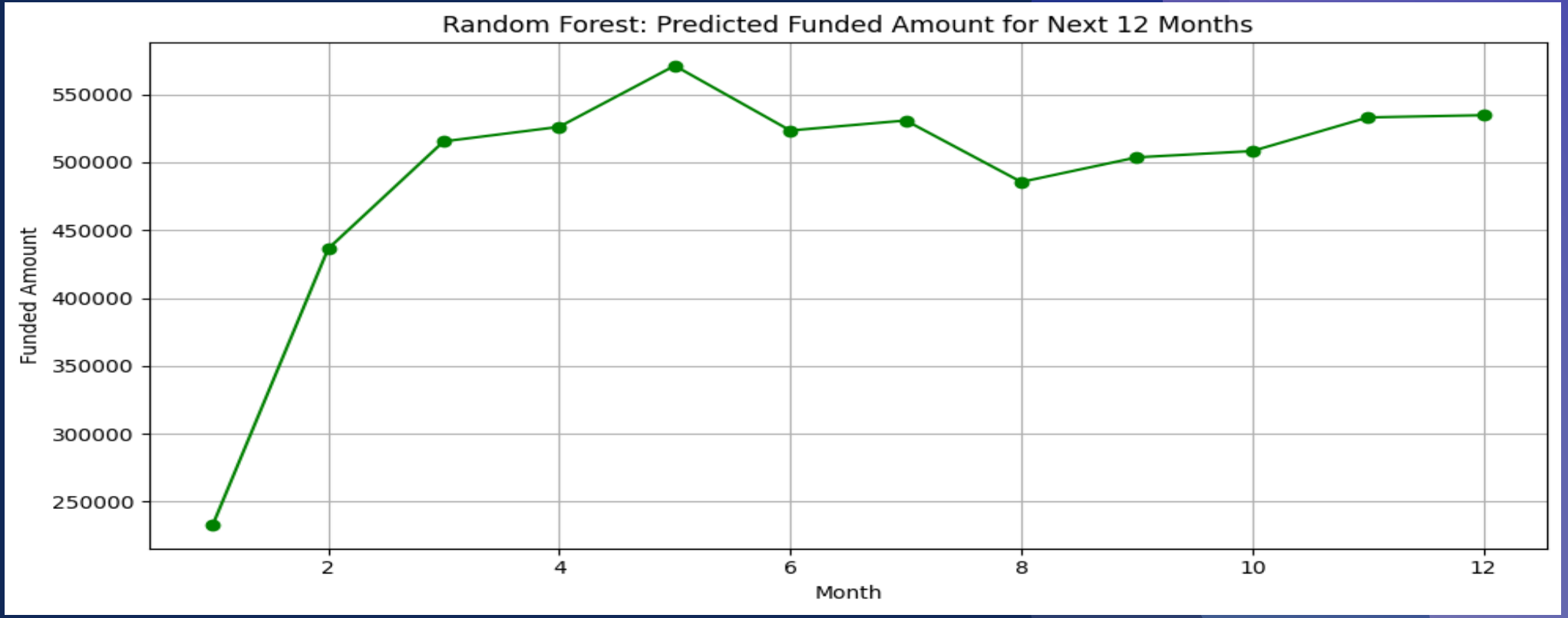
- Splitting Data into Train and Test.

- Ploting the Forecast with the Test using the ARIMA model .

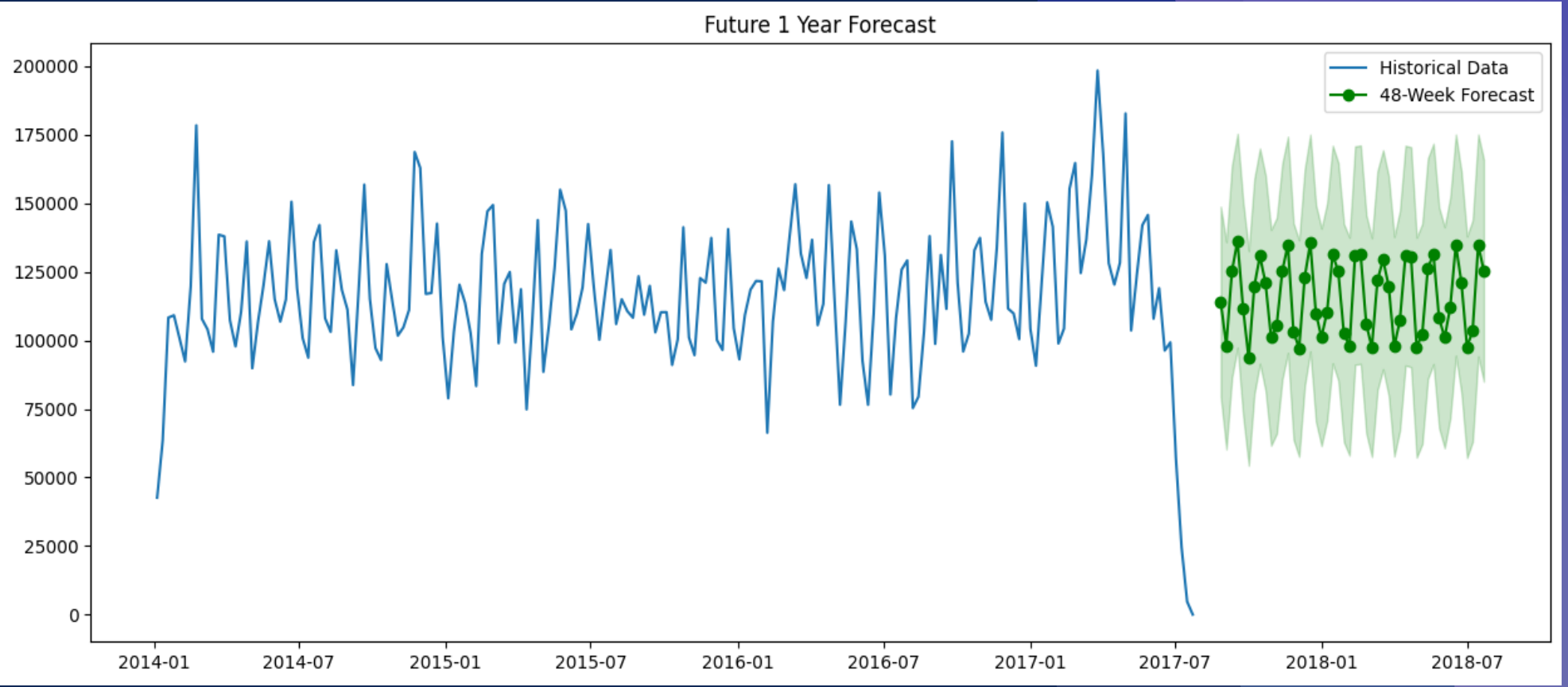
```
training_Data = int(len(ts) * 0.8)
train = ts[:training_Data]
test = ts[training_Data:]

model = ARIMA(train, order=(7, 1, 7))
model_fit = model.fit()
print(model_fit.summary())
forecast = model_fit.forecast(steps=len(test))
forecast.index = test.index
```

PREDICTION:
TOTAL FUNDED AMOUNT NEXT YEAR USING REGRESSION MODELS (TIME UNIT: 1 MONTH)



PREDICTION: TOTAL FUNDED AMOUNT NEXT YEAR USING TIME SERIES (TIME UNIT: 1 MONTH)



THANK YOU

Kamal Eldin
Omar Wafik
Rokaya
Nada Tarek
Salma
Siham