بنك التنمية الاجتماعية
SOCIAL DEVELOPMENT BANK

SOCIAL DEVELOPMENT BANK

Table of contents

01.
INTRODUCTION
About Social Development Bank

02.
BUSINESS PROBLEM
Vision 2030 and Problem Statement

03.
DATA SET DESCRIPTION
Dataset preview

04.
EDA
Data Exploration

05.
DATA PREPROCESSING
Prepare and clean the data

06.
THE MODEL
Building Regression Models

07.
THE DASHBOARD
KPI Dashboard

01.

INTRODUCTION

About Social Development Bank

Social Development Bank

Vision

To be pioneers in empowering social development tools and enhancing the financial independence of individuals and families towards a vital and productive society.

Mission

Provide financial and non-financial services and targeted savings plans supported by qualified human resources to contribute in social development, building partnerships with multiple sectors, spreading financial awareness and promoting a culture of self-employment among all segments of society.

02.

BUSINESS PROBLEM

Vision 2030 and Problem Statement

Vision 2030 and Problem Statement

In order to achieve the Bank's goals in achieving the goals and programs of Vision 2030 by enabling social development tools and enhancing the financial independence of individuals and families towards a vibrant and productive society, We found it necessary to anticipate the future value of financing loans granted to the individual, because through this it is possible to predict how the individual will become financially independent, enhance financial sufficiency and raise economic productivity.

Our goal

Our goal is to forecast the ideal loan amount for a given client, one that will help him/her increase their quality of life and ensure financial stability.

The prediction will be made using data from the Social Development Bank dataset and after considering and exploring the citizen information that has been provided to the Bank.

03.

DATASET DESCRIPTION

Dataset preview

Dataset description

Social Development Bank dataset is an open-source data provided by the Open Data portal of Saudi Arabia initiative. The data was obtained in the period of 2019 as described in the official website but we took our dataset from Kaggle as it was translated into English.

It contains 15 columns and 11,176 rows.

Dataset description

Variable	Type	Definition
ID	float64	client ID
bank branch	object	The city of the client to whom the loan was disbursed
funding type	object	Loan type (social, project, transfers)
funding classification	object	Type of loan disbursed to the client
Client sector	object	The sector in which the client works
financing value	float64	Amount provided as a loan to the client
installment value	object	Monthly payment amount
cashing date	object	The month the funding was disbursed

Dataset description

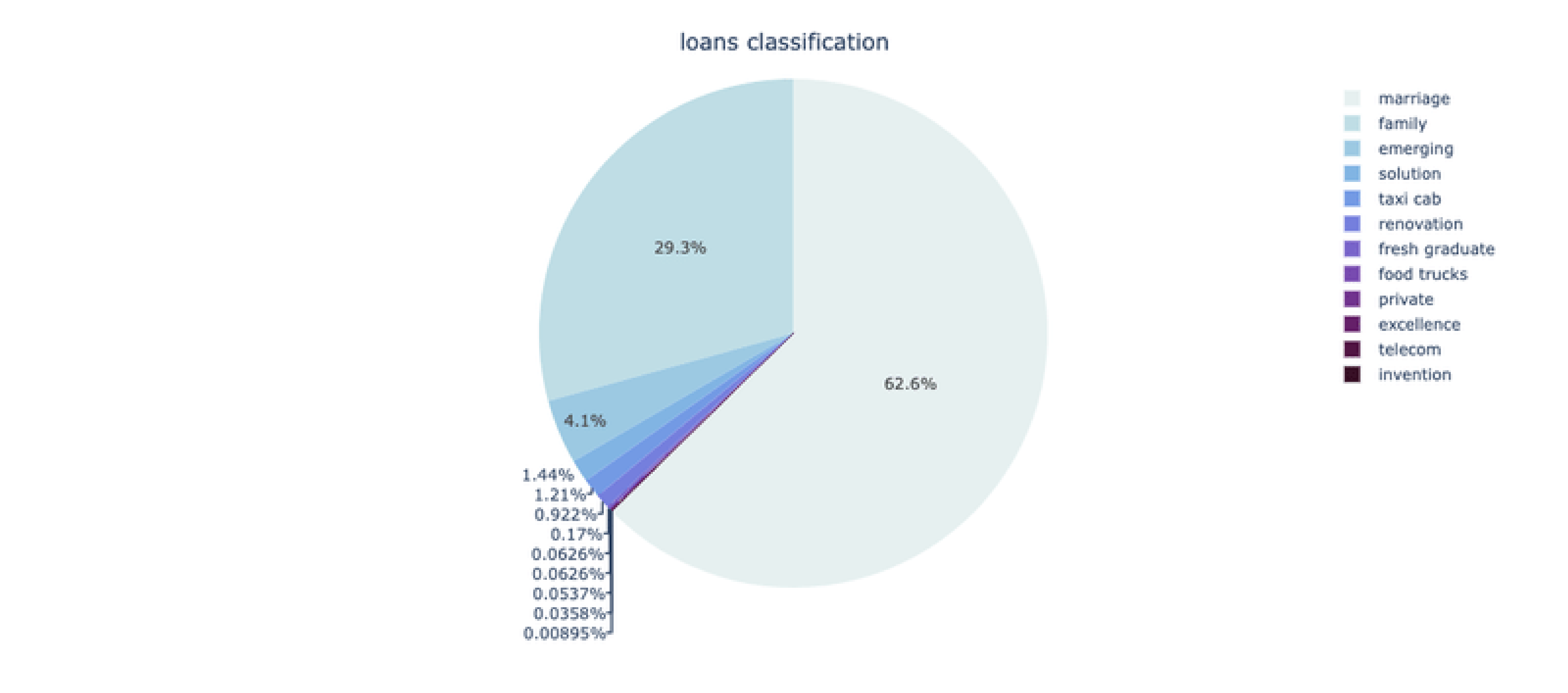
Variable	Type	Definition
sex	object	Male or female
Age	object	The age group to which the client belongs (youth, middle-aged adults, seniors, etc.)
Social status	object	Marital status or civil status of a person
Special needs	object	does the customer have special needs (yes, no)
Number of family members	object	Approximate number of members of the client's family
Savings loan	object	Is it saving loan? (Yes, no)
Income type	object	Categorize income into groups like (weak, medium, high, etc.)

04.

EDA

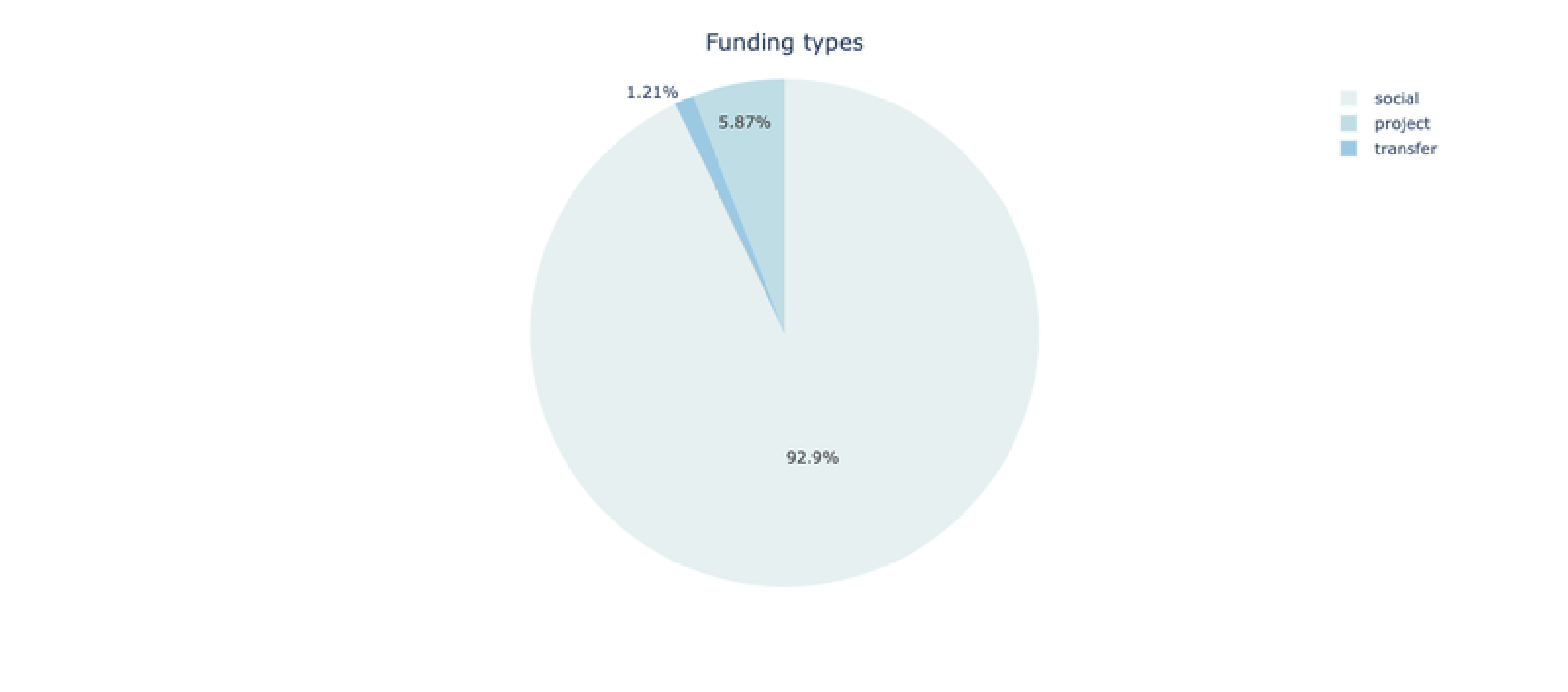
Data Exploration

EDA



loans classification
29.3%
62.6%
4.1%
1.44%
1.21%
0.922%
0.17%
0.0626%
0.0626%
0.0537%
0.0358%
0.00895%
marriage
family
emerging
solution
taxi cab
renovation
fresh graduate
food trucks
private
excellence
telecom
invention

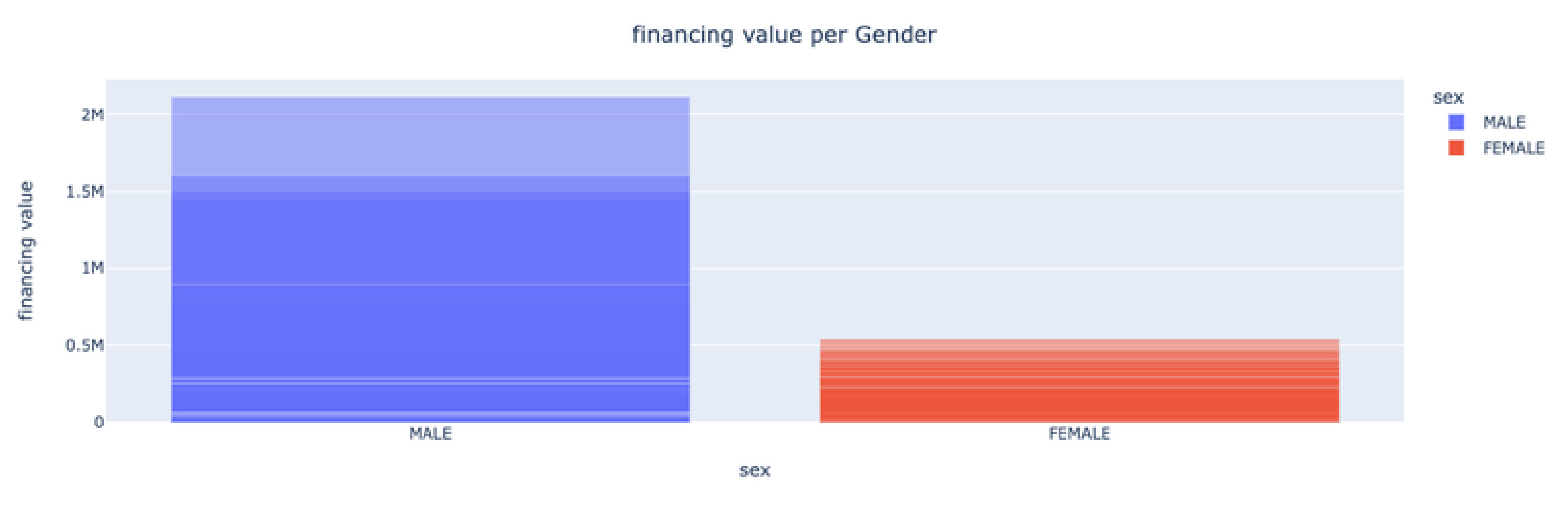
EDA



EDA

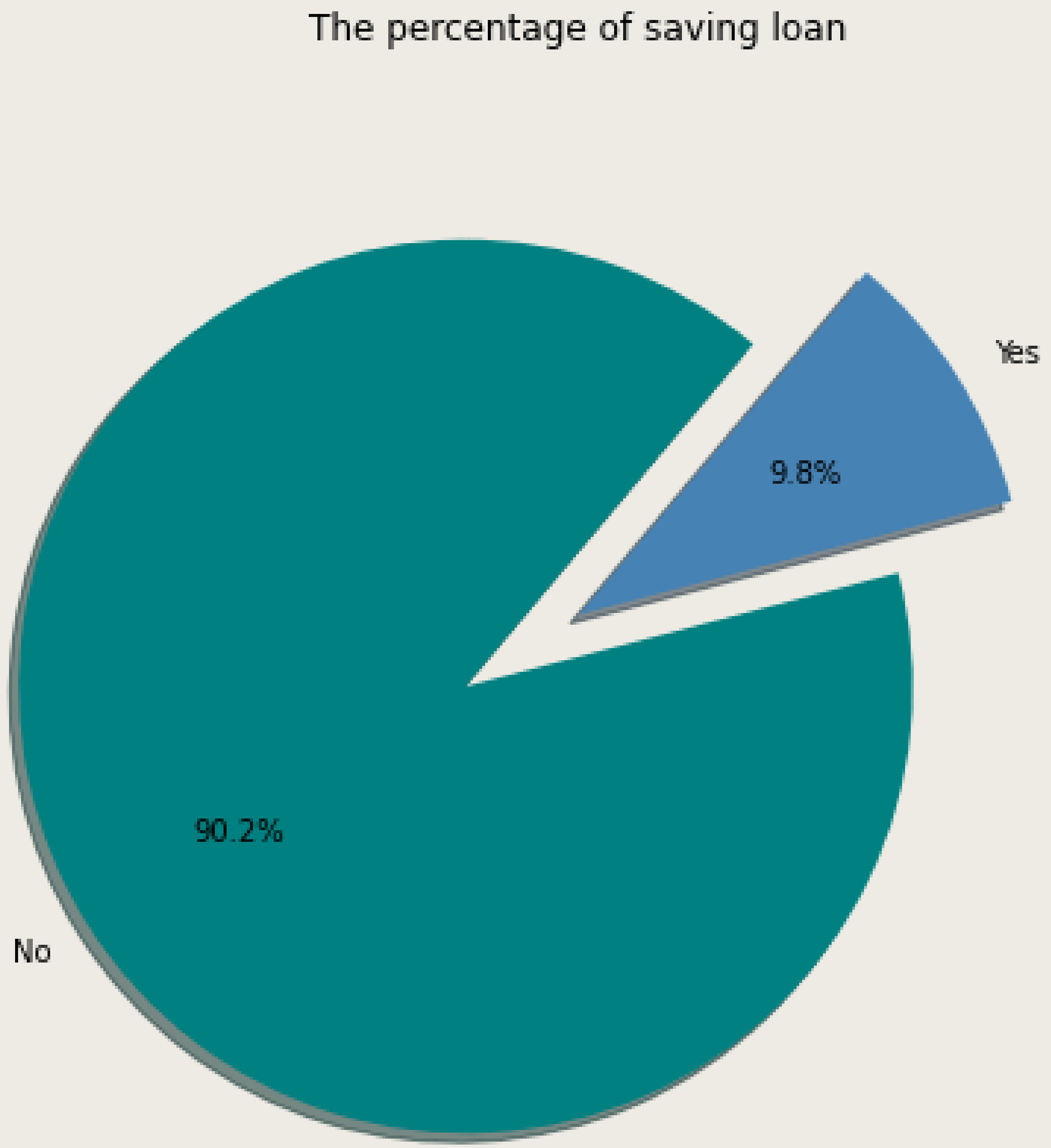
Total number of financing value for each funding producet
Total number of financing value
20M
15M
10M
5M
0
2019/02
2019/01
2019/03
Months
funding classification
family
solution
marriage
emerging
renovation
taxi cab
telecom
food trucks
excellence
fresh graduate
private
invention

EDA

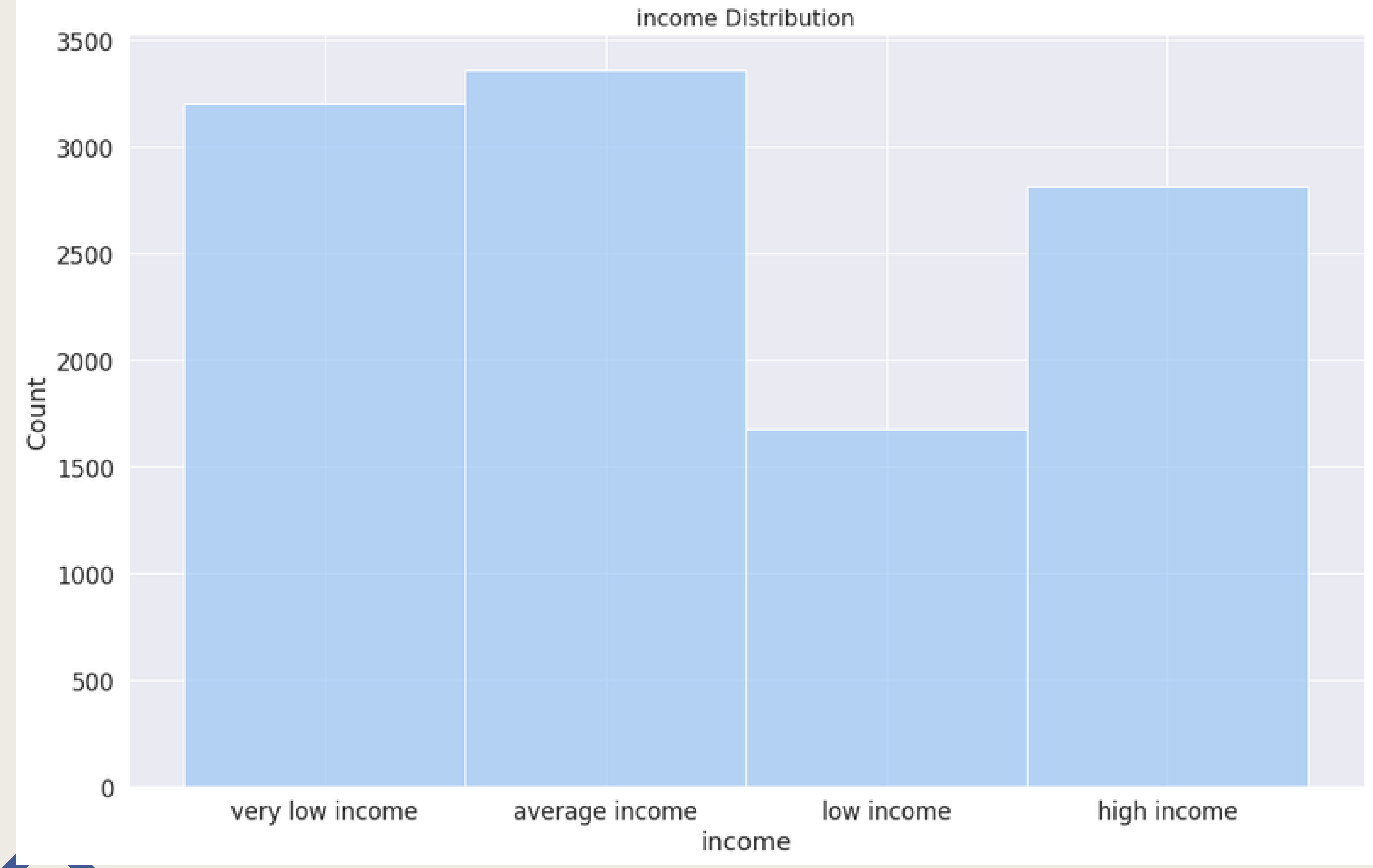


financing value per Gender
financing value
2M
1.5M
1M
0.5M
0
MALE
FEMALE
sex
sex
MALE
FEMALE

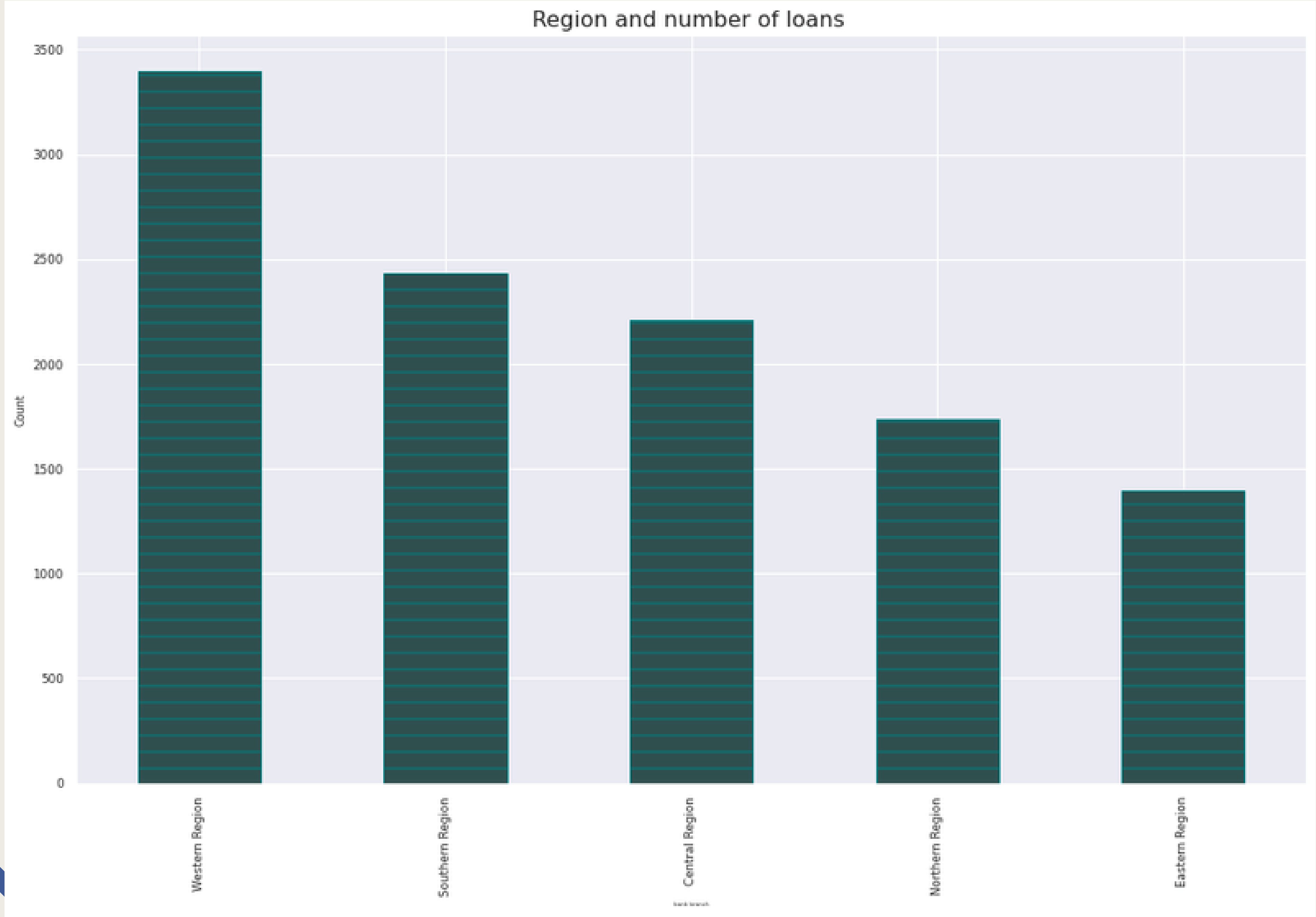
EDA



EDA

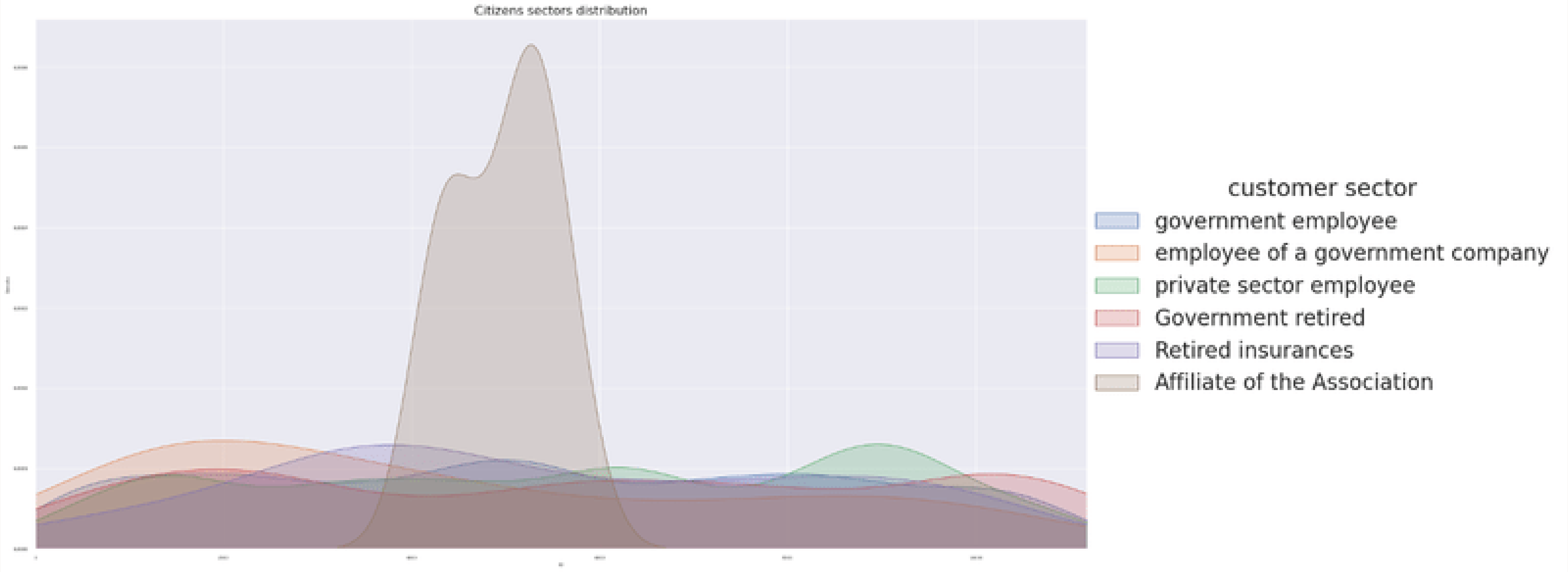

income Distribution
3500
3000
2500
2000
1500
1000
500
0
Count
very low income
average income
low income
high income
income

EDA

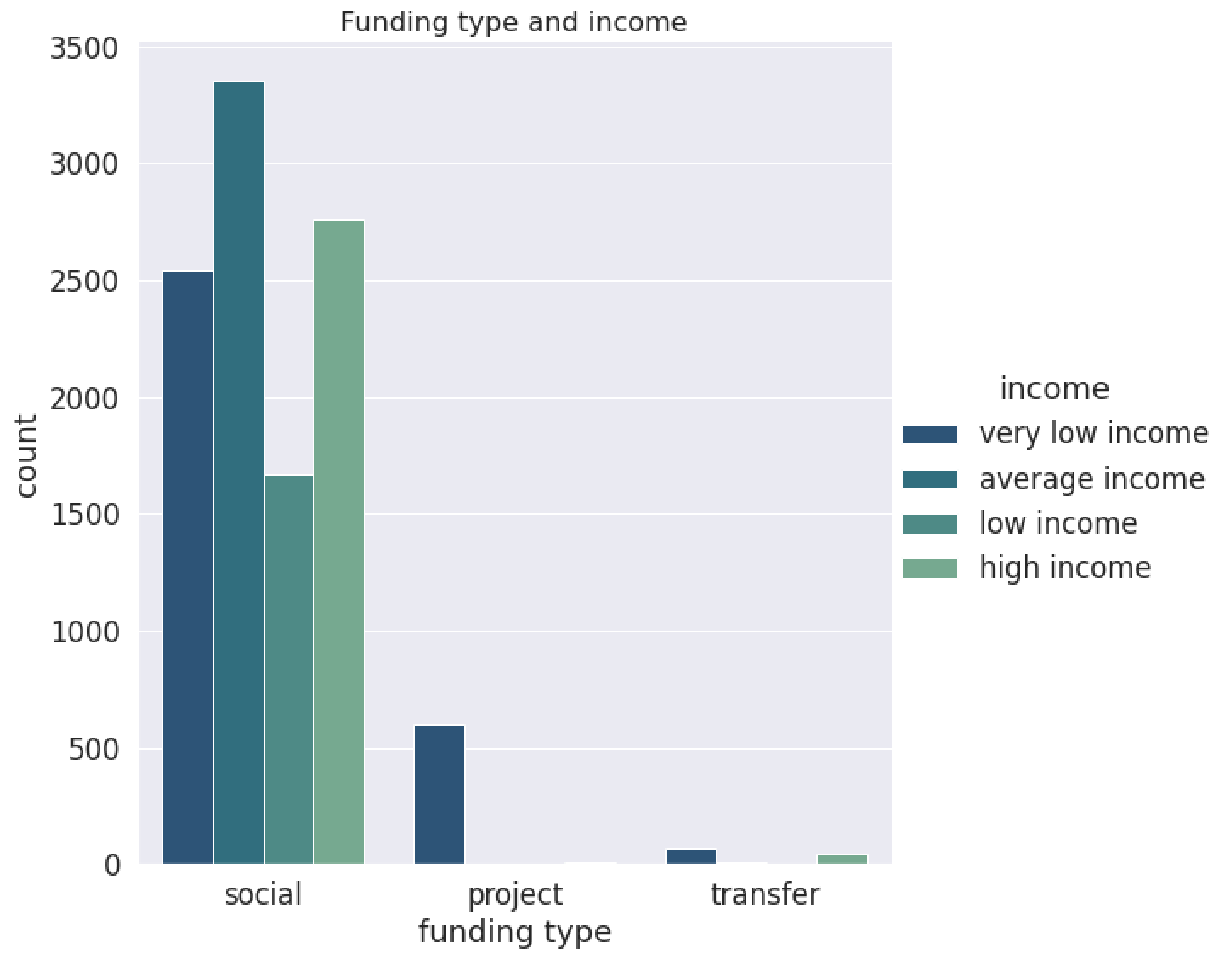


Region and number of loans
Count
3500
3000
2500
2000
1500
1000
500
0
Western Region
Southern Region
Central Region
Northern Region
Eastern Region

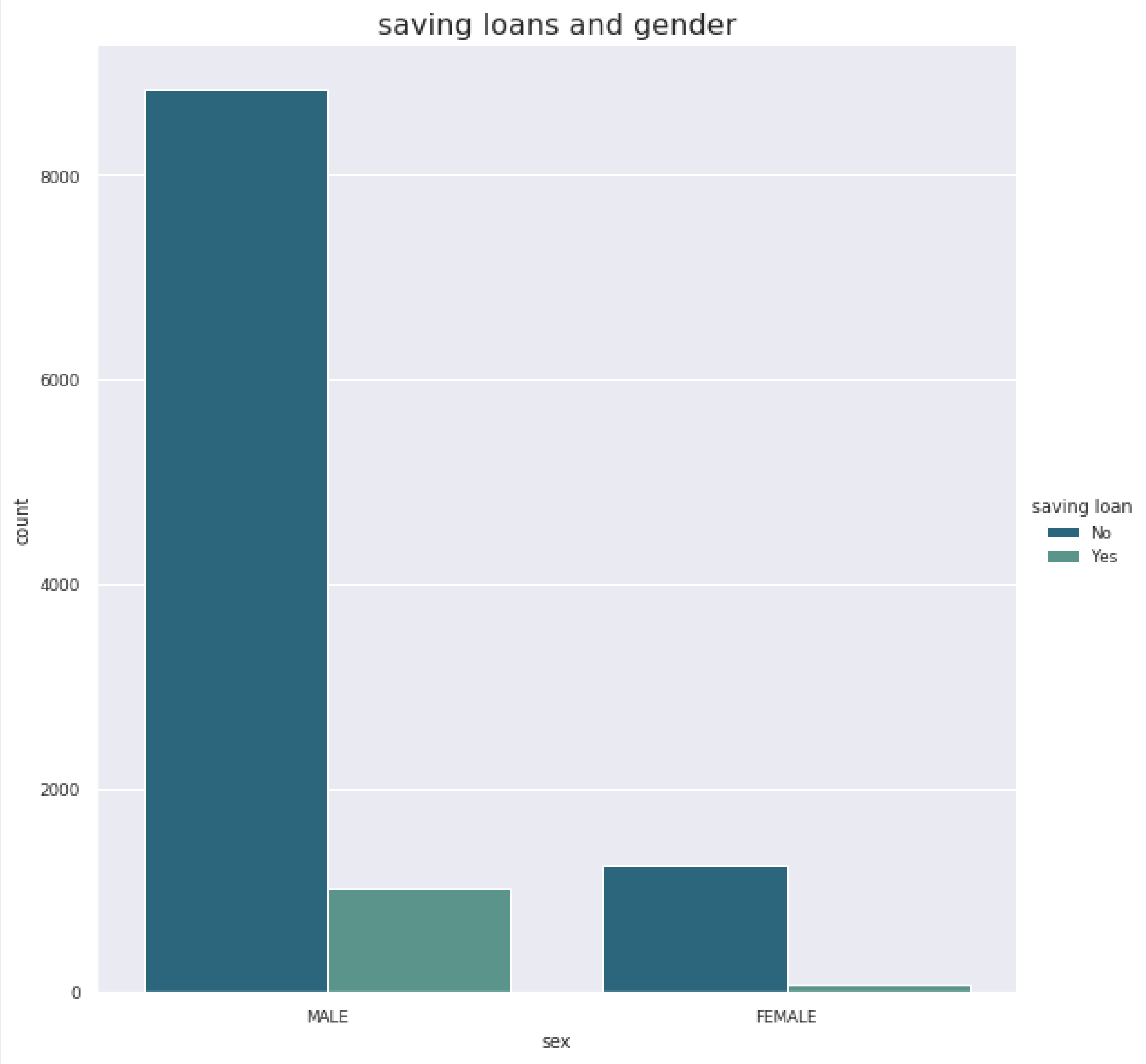
EDA


Citizens sectors distribution
customer sector
government employee
employee of a government company
private sector employee
Government retired
Retired insurances
Affiliate of the Association

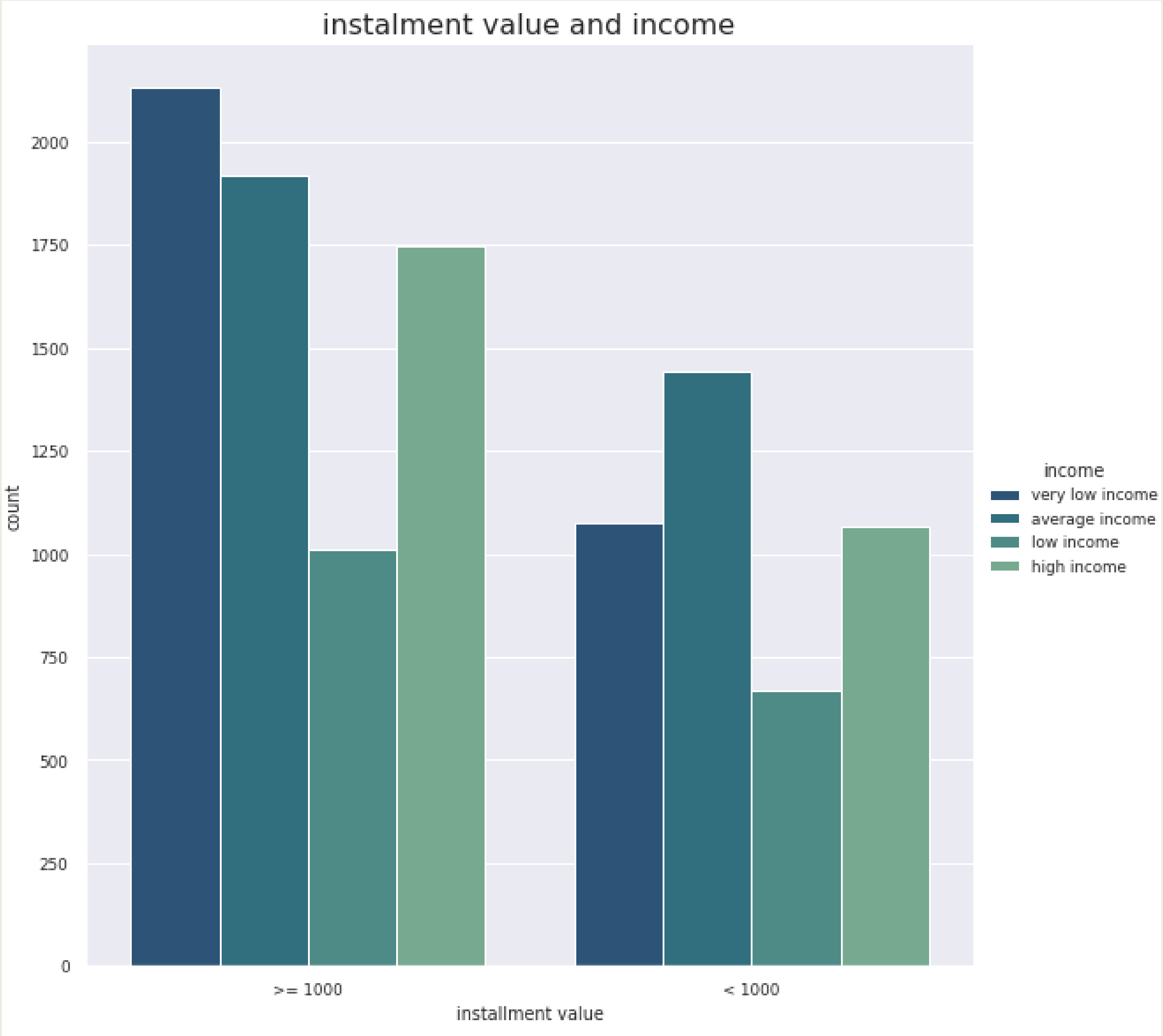
EDA


Funding type and income
3500
3000
2500
2000
1500
1000
500
0
count
social
project
transfer
funding type
income
very low income
average income
low income
high income

EDA

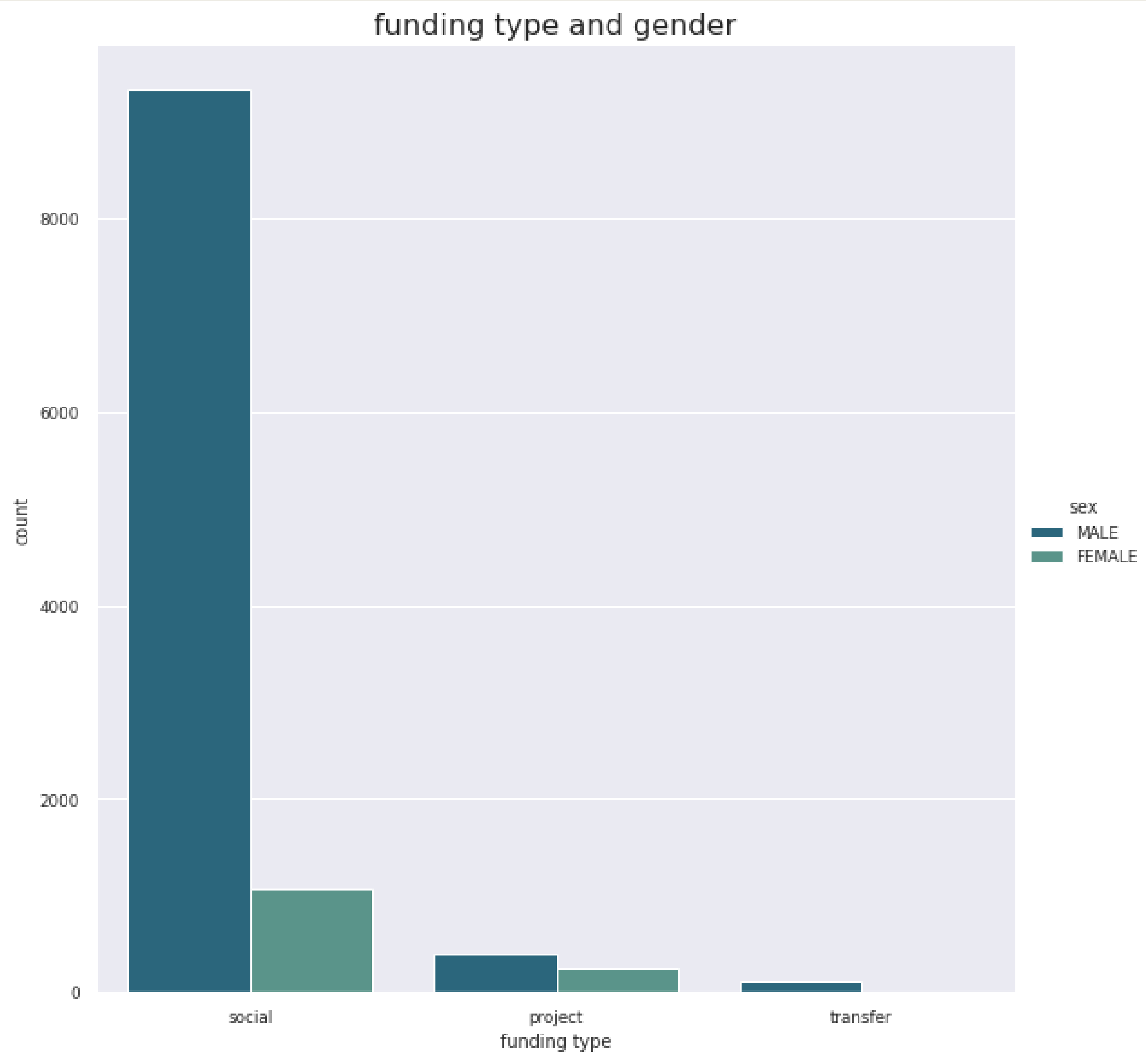

saving loans and gender
8000
6000
4000
2000
0
count
MALE
FEMALE
sex
saving loan
No
Yes

EDA



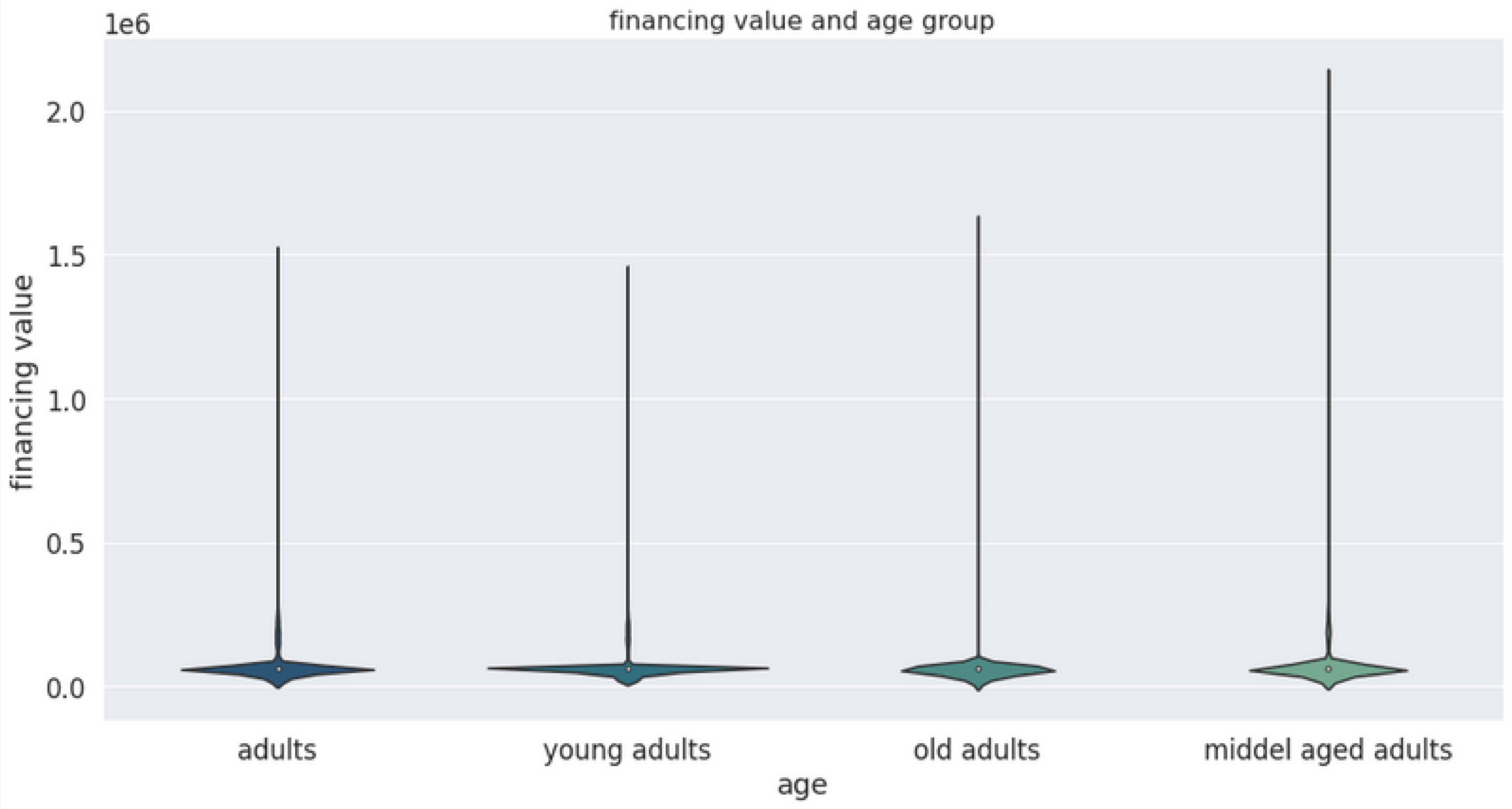
instalment value and income
count
2000
1750
1500
1250
1000
750
500
250
0
>= 1000
< 1000
installment value
income
very low income
average income
low income
high income

EDA



funding type and gender
count
8000
6000
4000
2000
0
social
project
transfer
funding type
sex
MALE
FEMALE

EDA


financing value and age group
1e6
2.0
1.5
1.0
0.5
0.0
financing value
adults
young adults
old adults
middel aged adults
age

05.

DATA PREPROCESSING

Prepare and clean the data

Data Cleaning

- Missing data handling

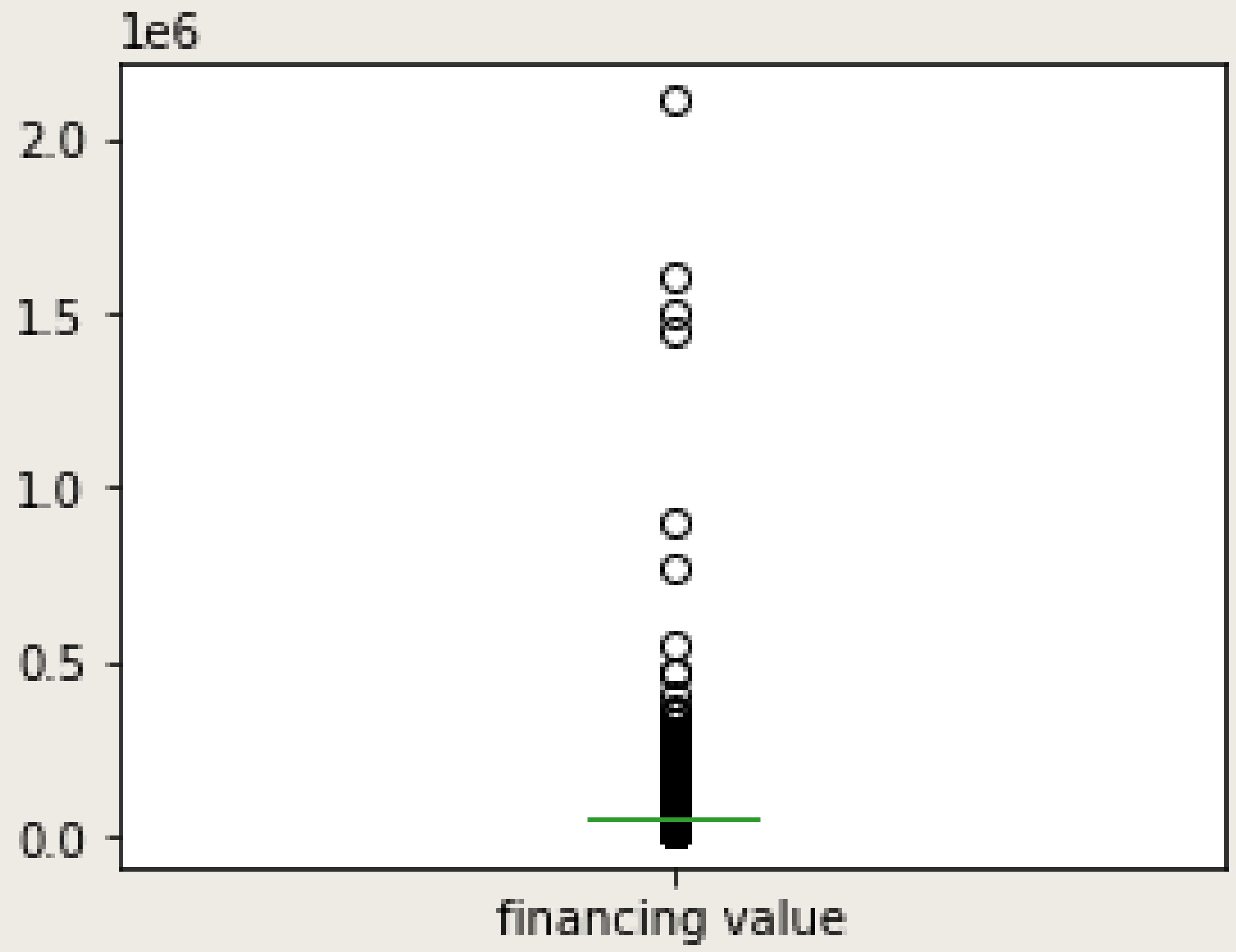
```
#Checking for the null values in the dataset
data.isnull().sum()
```

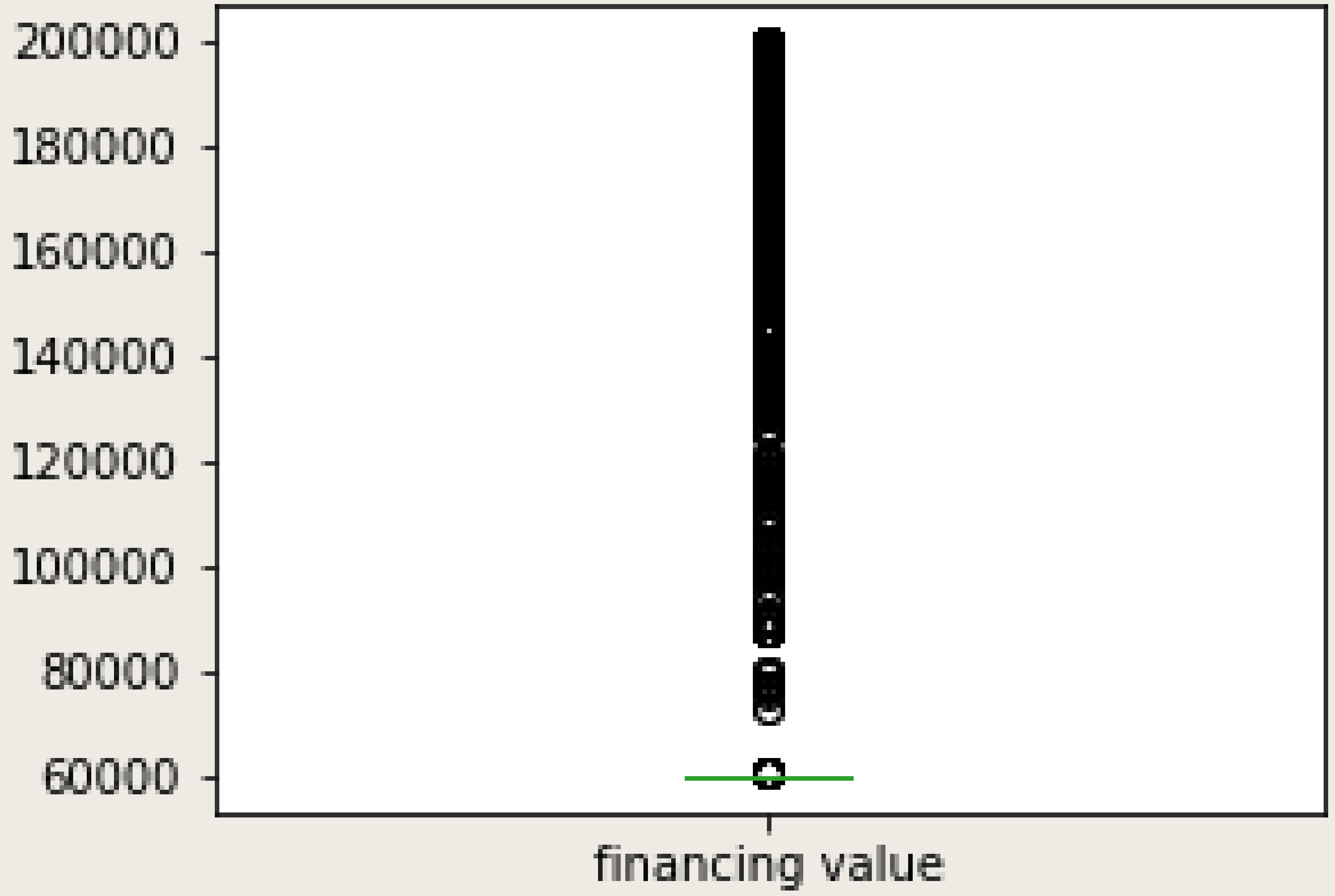
```
ID                             0
bank branch                    0
funding type                   0
funding classification         0
customer sector             3950
financing value                0
installment value              0
cashing date                   0
sex                            0
age                            6
social status                  0
special needs                  0
number of family members      43
saving loan                    0
income                       114
dtype: int64
```

```
# Fill in the Missing Values using the Simple Imputer with the Most Frequent strategy
imputer = SimpleImputer(strategy='most_frequent', missing_values=np.nan)
imputer = imputer.fit(data[['customer sector', 'income', 'number of family members', 'age']])
data[['customer sector', 'income','number of family members', 'age']] = imputer.transform(
    data[['customer sector', 'income', 'number of family members', 'age']])
```

Data Cleaning

- Outliers data handling





Feature Engineering

Dropping Certain Columns

```
# Delete unneeded coulmns, ID, and bank branch.
data.drop(['ID'], axis=1, inplace=True)
data.drop(['cashing date'], axis=1, inplace=True)
data.drop(['social status'], axis=1, inplace=True)
data.drop(['special needs'], axis=1, inplace=True)
```

Mapping Certain Columns

```
# Map the saving loan coulmn from Yes/No to 0/1
data['saving loan'] = data['saving loan'].map({'Yes': 0, 'No': 1})

# Map the sex column with 0 -> male , 1 -> female
data['sex'] = data['sex'].map({'MALE': 0, 'FEMALE': 1})
```

Apply label Encoding

```
# Apply Label Encoding to convert categorical type columns into numerical ones.
# Create a list of the columns to be converted into numerical values.
cols = ['bank branch', 'funding type', 'funding classification',  'customer sector', 'installment value', 'age', 'number of family members', 'income']

# Encode labels of multiple columns at once
data[cols] = data[cols].apply(LabelEncoder().fit_transform)
```

06.

THE MODEL

Building Regression Models

Split data

```
# Split data into X and y.
# X for the train set , y for the test set
X = data.drop(columns='financing value')
y = pd.DataFrame(data['financing value']) #target class


print('X shape :', {X.shape})
print('y shape :', {y.shape})
```

```
X shape : {(8942, 10)}
y shape : {(8942, 1)}
```

```
# Split Dataset into Train and Test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

X_train.shape, X_test.shape
```

```
((6259, 10), (2683, 10))
```

Machine Learning Models

```python
# 1. Linear Regression

lin_reg=LinearRegression() # Initialize the model
lin_reg.fit(X_train,y_train) # Fit the model

preds_lin = lin_reg.predict(X_test) # Predict X_test
```

```python
# 2. Random Forest Regression

rf_reg = RandomForestRegressor(n_estimators=10, max_depth=6, random_state=42) # Initialize the model
rf_reg.fit(X_train,y_train) # Fit the model

preds_rfr = rf_reg.predict(X_test) # Predict X_test
```

```python
# 3. Decision Tree Regression

reg_tree = DecisionTreeRegressor(random_state = 42, max_depth= 4, criterion= 'mse') # Initialize the model
reg_tree.fit(X_train, y_train) # Fit the model

preds_tree = reg_tree.predict(X_test) # Predict X_test
```

```python
# 4. Support Vector Regression

svr_reg = SVR(kernel = 'rbf') # Initialize the model
svr_reg.fit(X_train, y_train) # Fit the model

preds_svr = svr_reg.predict(X_test) # Predict X_test
```

Models' Results

Model	R2 Score	MAE
Linear regression	0.64	6972.72
Random Forest regression	0.92	1190.1
Decision Tree regression	0.92	1197.66
Support Vector regression	-0.05	4697.16

Model Optimization

Grid Search for RF Model

```
param_grid = {

    "n_estimators": [5,7,10, 15], # how many trees in our forest
    "max_depth": [2,4,6] # how deep each decision tree can be
}

grid = GridSearchCV(
    rf_reg,
    param_grid,
    cv = 5,
    n_jobs=-1,
    verbose=1,
    scoring="neg_mean_absolute_error"
)

grid.fit(X_train, y_train)
```

```
# Re-create the model using the best parameters
Rf = RandomForestRegressor(max_depth = 6, n_estimators = 5)
Rf.fit(X_train,y_train)
preds_rf = Rf.predict(X_test)

# Calculate the accuracy score for Decision Tree regression
r2_score(y_test,preds_rf)
```

```
0.9187159701066172
```

```
# Calculate the MSE for Random Forest Regression
mean_absolute_error(y_true=y_test, y_pred=preds_rf)
```

```
1186.6477430321252
```

Grid Search for DT Model

```
param_grid2 = {
    "max_depth": [4, 6, 10] # how deep decision tree can be
}

grid2 = GridSearchCV(
    reg_tree,
    param_grid2,
    cv = 5,
    n_jobs=-1,
    verbose=1,
    scoring="neg_mean_absolute_error"
)

grid2.fit(X_train, y_train)
```

```
# Re-create the model using the best parameters
DT = DecisionTreeRegressor(max_depth= 4)
DT.fit(X_train, y_train)
preds_dt = DT.predict(X_test)

# Calculate the accuracy score for Decision Tree regression
r2_score(y_test,preds_dt)
```

```
0.9179531356056327
```

```
# Calculate the MSE for Decision Tree regression
mean_absolute_error(y_true=y_test, y_pred=preds_dt)
```

```
1197.6638302295048
```

Pipeline

```
pipe = make_pipeline(
    # Step-1 Scale parameters
    StandardScaler(),
    # Step-2 fit the principles to the ML model
    RandomForestRegressor(max_depth = 6, n_estimators = 5)
)

pipe.fit(X_train, y_train)
pipe.score(X_train, y_train)
```

```
0.9278863735651952
```

07.

THE DASHBOARD

KPI Dashboard

Social Development Bank Loans Analysis

cashing date: All

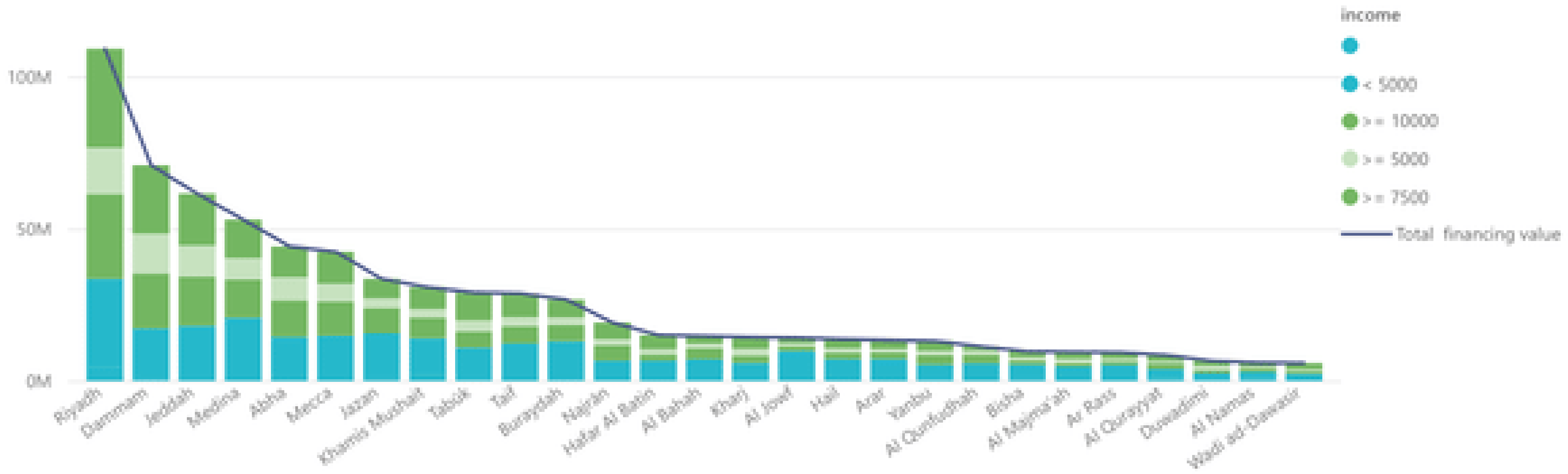
customer sector: All

bank branch: All

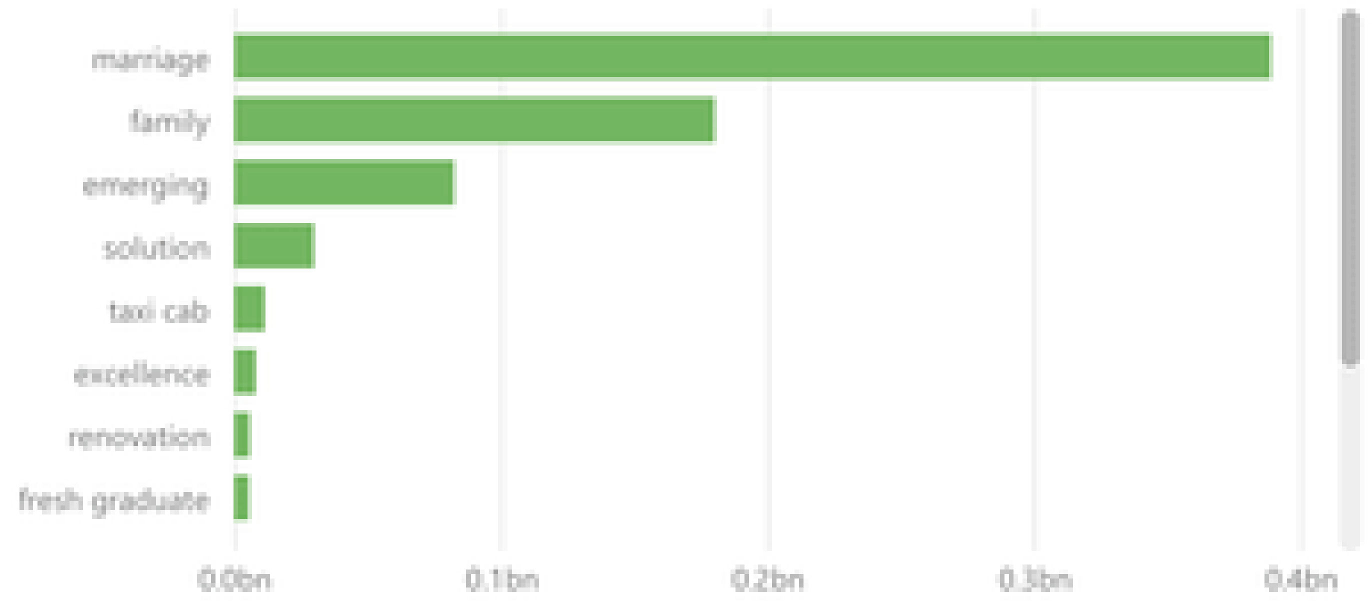
715M
Total financing value

11.18K
Total beneficiaries

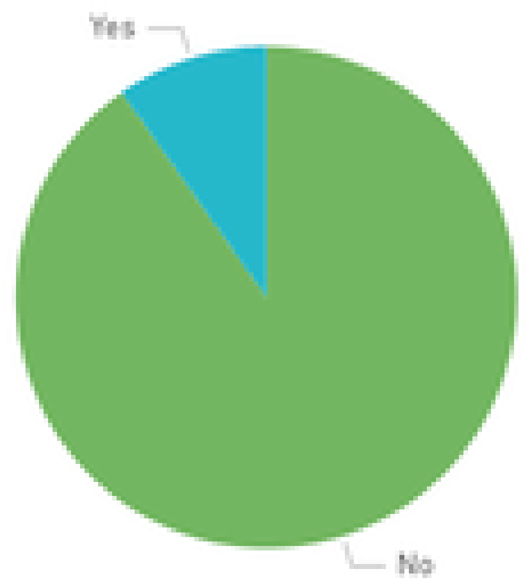
Distribution of citizens' income per branch



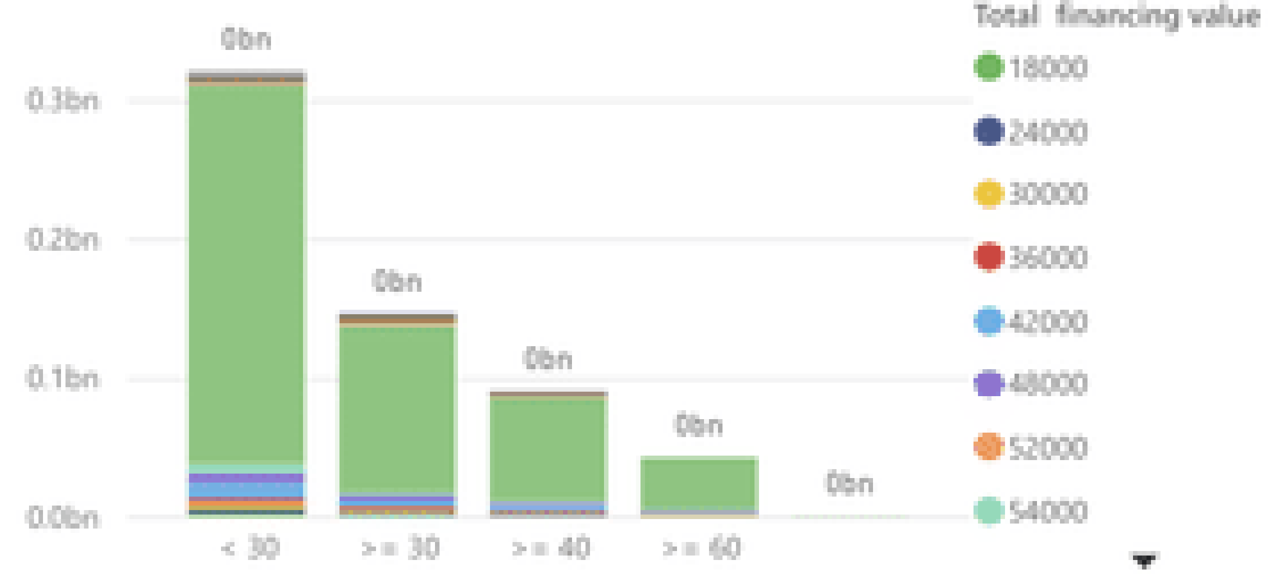
The financing value per funding types



The percentage of saving loans



Total financing value per age group



Conclusion

For further improvements in the future, we aim to enhance our model by getting more data over the next years. Such improvements would help predict the future value of financing loans granted to the individual and predict how the individual will become financially independent, enhance financial sufficiency and raise economic productivity.

We can also extend the model’s capabilities by deploying it to make analytical predictions or feeding it with new types of data. Moreover, creating a model that can classify requests as being approved or rejected by training the model on the complete set of data where some citizen requests were denied.

THANK YOU!