Machine Learning Engineer - Challenge

The following document informs all the development done to overcome the challenges sent in the previous document: Machine_Learning_Engineer_Challenge. The following sections report 5 stages implementation of machine learning project development classic pipeline. These phases are: exploratory data analysis, feature engineer, model implementation, API design and model deployment process.

1. Data Exploratory Analysis

This stage analyzes the dataset_credit_risk.csv dataset. Taking into account the number of users, number of loans, different sources of income, gender, occupation, amount of loan, birth date, job start date, marital status, among others.

For more information check the notebook: **Exploratory Data Analysis.ipynb

These are some of the figures, plot or/and tables obtained when analyzing the dataset:

- This tables show the number of missing values for each column

```
missing_df.rename(index=str,columns={0:'number of missing values'})
```

ication_type	name_family_status	...	flag_work_phone	flag_phone	flag_email	occupation_type	cnt_fam_members	status	birthday	job_start_date	loan_date
0	0	...	0	0	0	240048	0	0	0	0	0

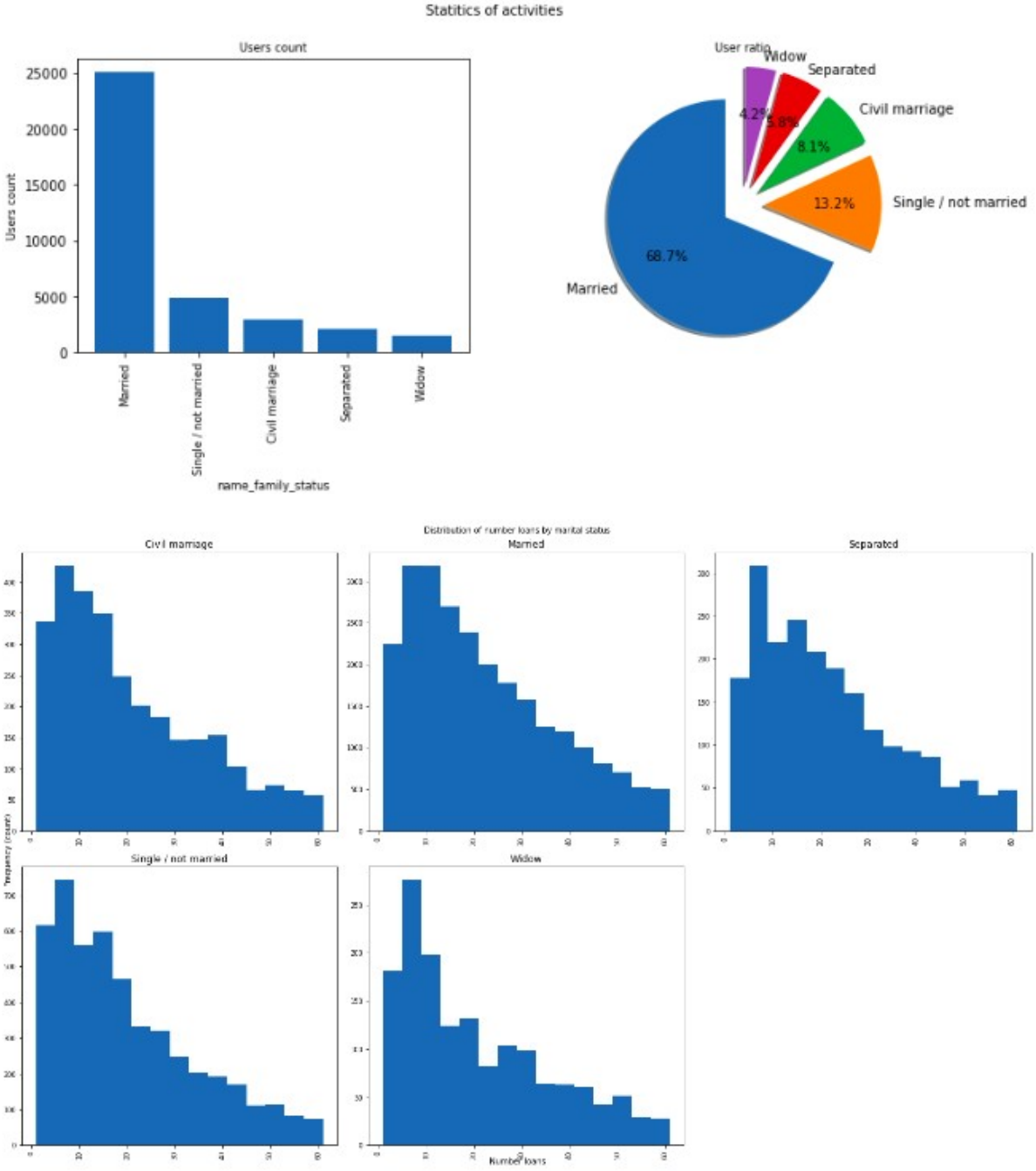
- Distribution gender vs user vs total lon amount

	Gender	# of id	Count	Total loans
0	F	24430	518851	67,178,718.752191
1	M	12027	258864	33,555,688.054158

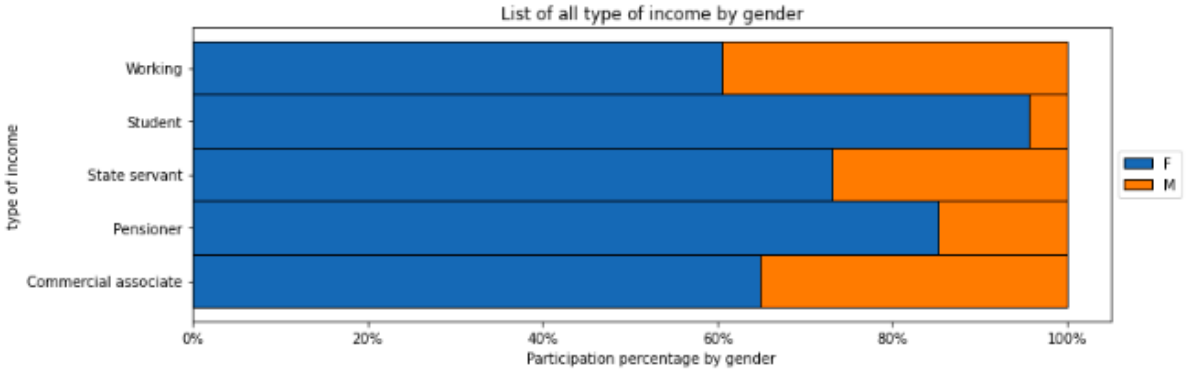
- Number of user, loans and total loan amount for each income’s type

	name_income_type	Users_count	Loans_count	percentage
0	Working	18819	400164	51.619716
1	Commercial associate	8490	183385	23.287709
2	Pensioner	6152	128392	16.874674
3	State servant	2985	65437	8.187728
4	Student	11	337	0.030173

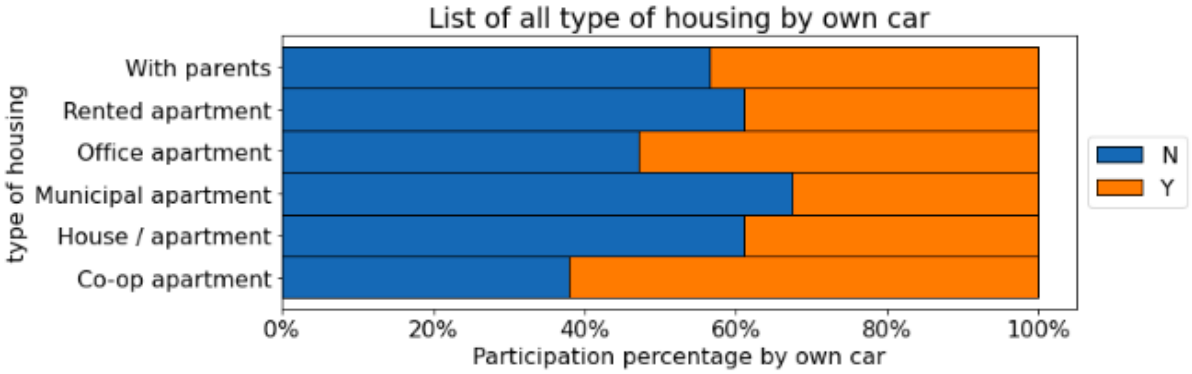
- Histogram, pie plot, bar plot for each marital status



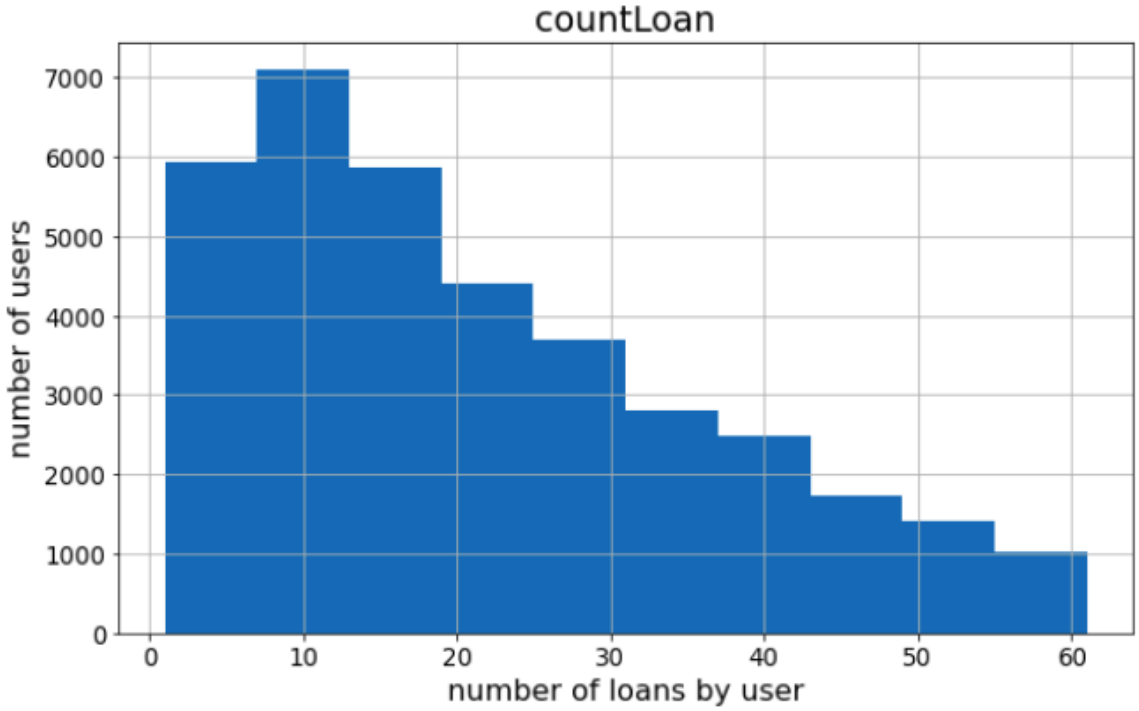
- Plot comparison between feature with more of 2 values (name_income_type, name_education_type, etc) vs features with 2 values (gender, flag_own_car, flag_email, etc). For example: income vs gender



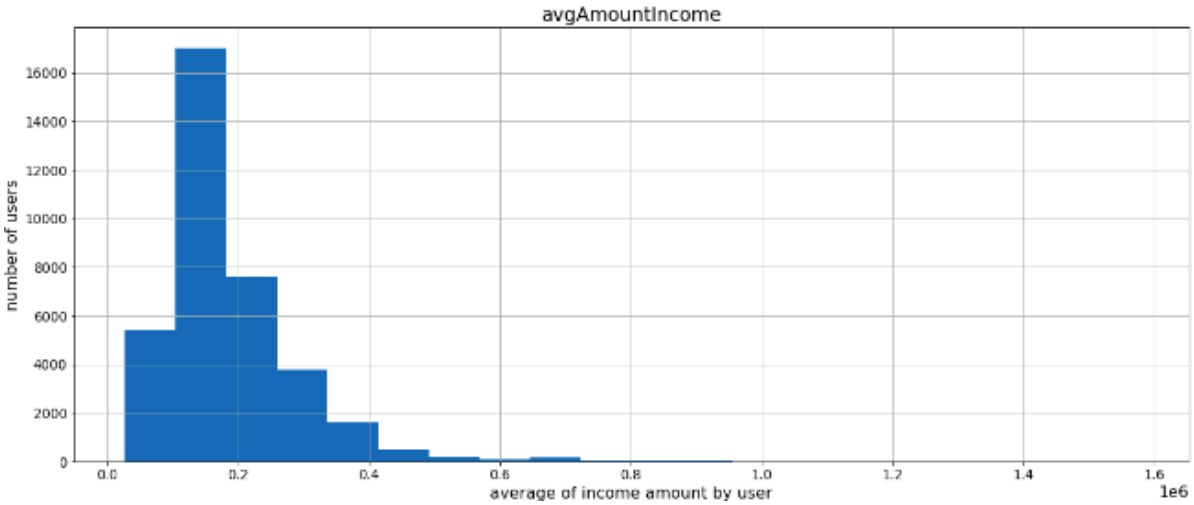
- Plot comparison housing type vs own car



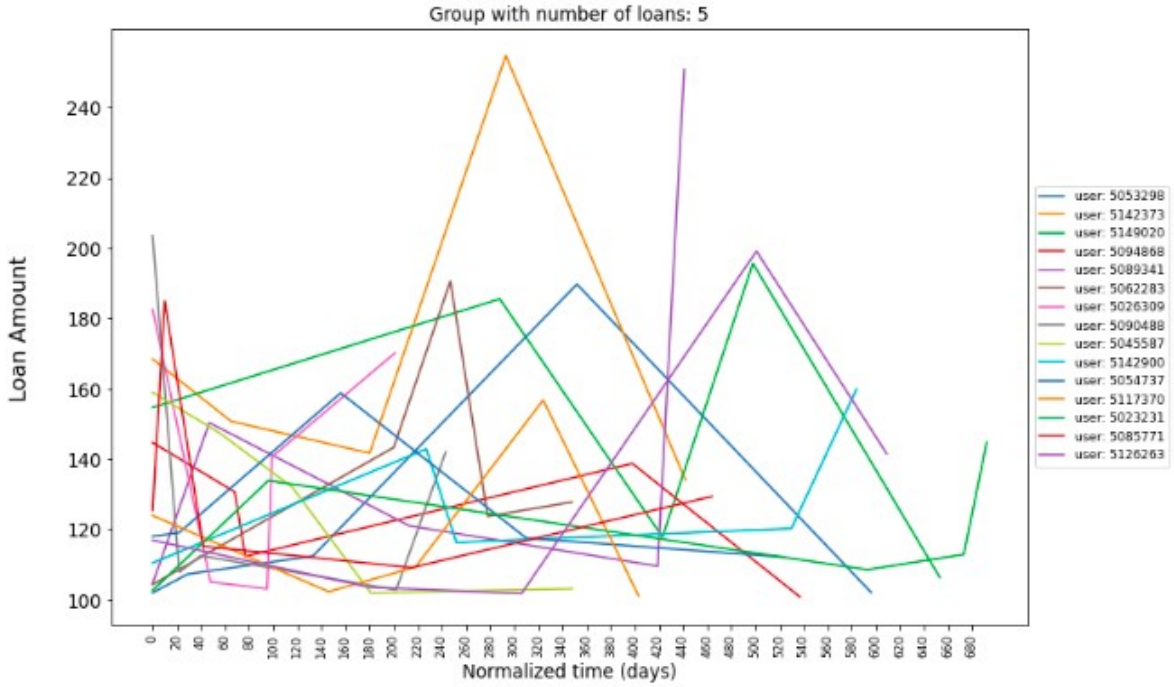
- Histogram of number loans by user



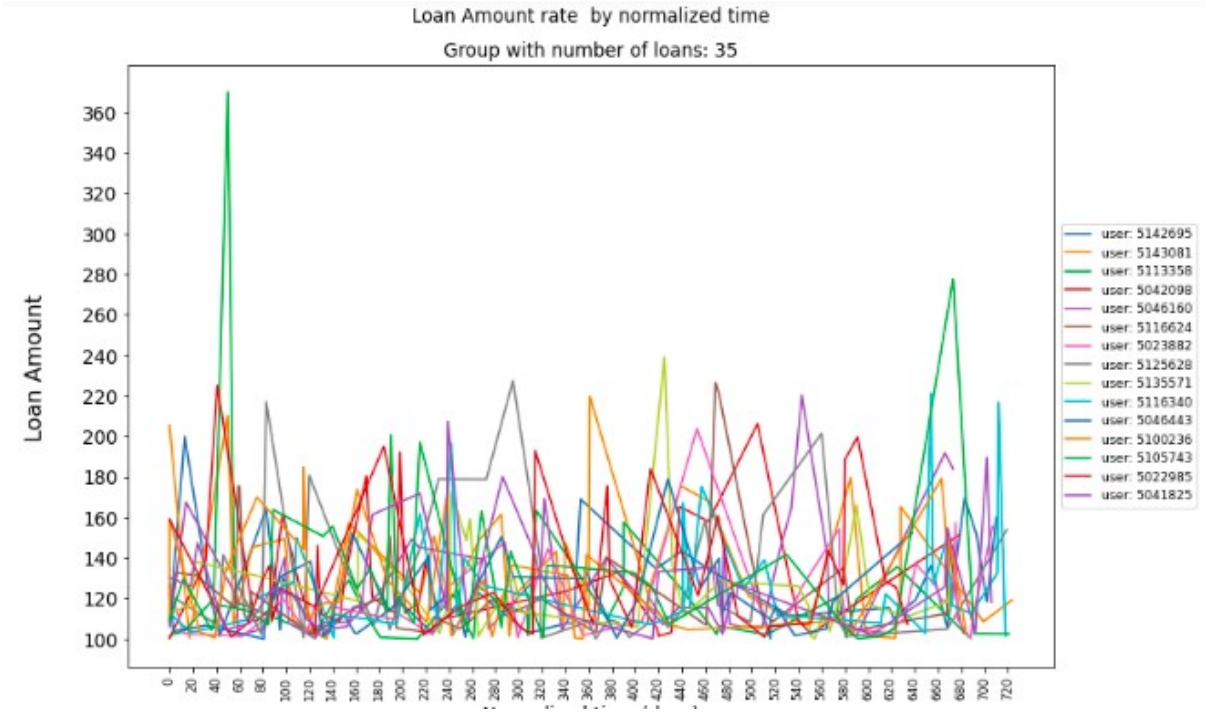
- Histograma of average of income amount by user



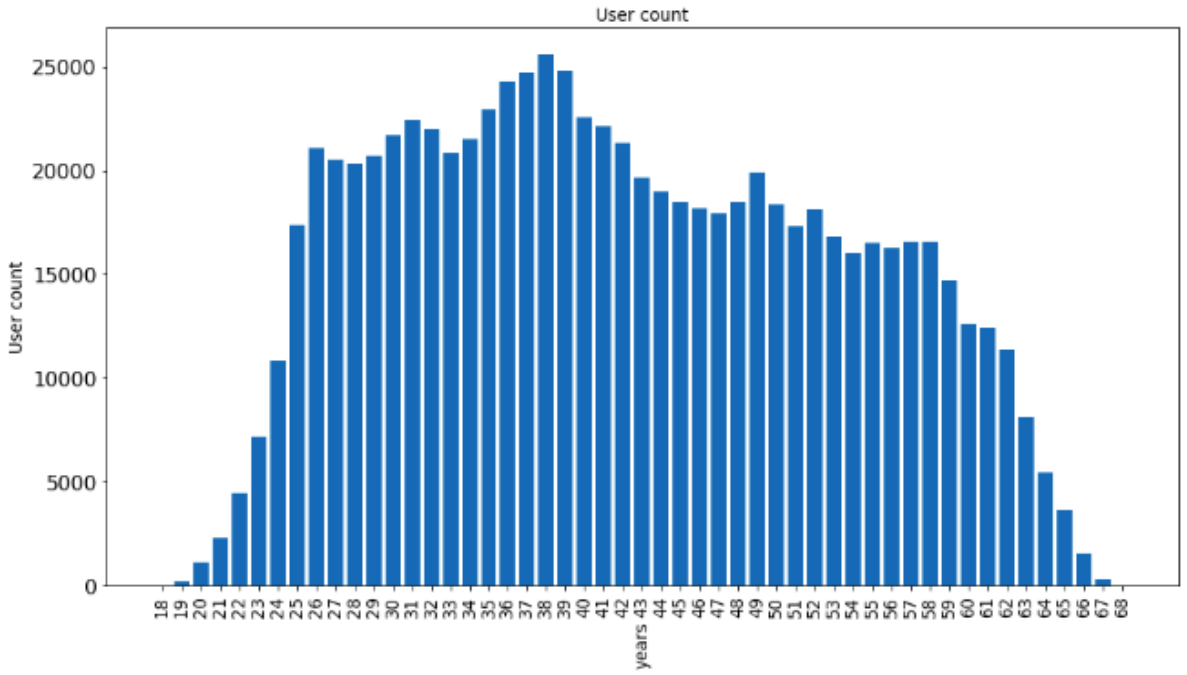
- Plot distribution over time of loan amount of some 15-random users who have requested 5 loans



- Plot distribution over time of loan amount of some 15-random users who have requested 35 loans



- Histogram of age by user



2. Feature Engineer

This stage obtains the feature vector of the dataset_credit_risk.csv. This feature vector will use for training the model and making prediction. These features are:

- **nb_previous_loans**: number of loans granted to a given user, before the current loan.
- **avg_amount_loans_previous**: average amount of loans granted to a user, before the current loan.

- **age**: user age in years.
- **years_on_the_job**: years the user has been in employment.
- **flag_own_car**: flag that indicates if the user has his own car.

These features were obtained using **pyspark** (spark for python). For example:

- Those lines of code for converting the flag_own_car (string format) to [0-1] format.

```
dfDataSetFeatureEngineer = dfDataSetFeatureEngineer \
                    .withColumn('flag_own_car_n', map_func(col('flag_own_car'))) \
                    .drop('flag_own_car') \
                    .withColumnRenamed('flag_own_car_n', 'flag_own_car')
```

- Those lines of code for converting the years on the job, among others

```
dfDataSetFeatureEngineer = dfDataSetFeatureEngineer \
                .withColumn('years_on_the_job', functions.months_between(functions.current_date(), col('job_sta
                .withColumn('years_on_the_job', col('years_on_the_job').cast(IntegerType())) \

# drop birthday column
dfDataSetFeatureEngineer = dfDataSetFeatureEngineer.drop('job_start_date')

dfDataSetFeatureEngineer.limit(7).toPandas().head()
```

For more information read the **MLE_challenge - Features engineering - Notebook 1-withSolution.ipynb** notebook**.** This notebook contain code cells of exploratory analysis notebook for a better understanding of the features.

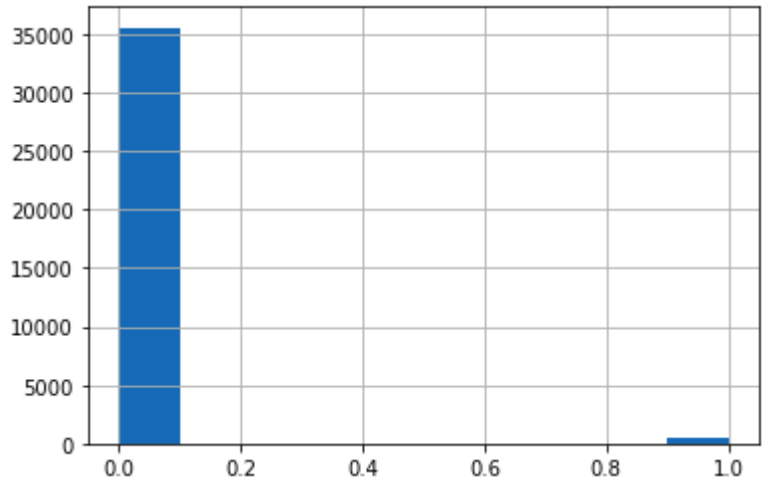
3. Model Implementation/Model Training

This stage uses the dataset with obtained features of section 2. For the y-label (prediction column) uses the status column. The status value for the most recent loan for each user.

Line codes for obtaining the y-value for each user:

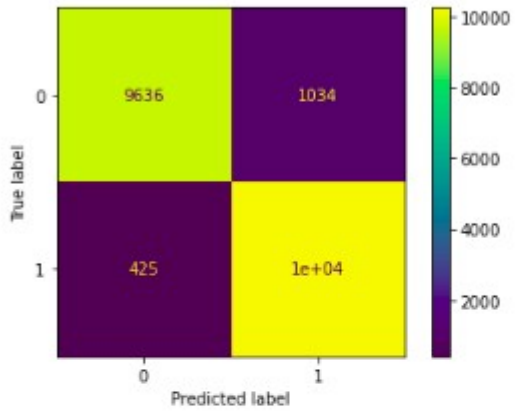
```
# drop records with nb_previous_loans == 0
# because this record can't have a valid status

numberRecordsWith0PreviousLoans = dfDataSetFeatureEngineer.where(co
totalUsers = dfDataSetFeatureEngineer.count()
print(f'number of records with 0 previous loans: {numberRecordsWith

dfDataSetFeatureEngineer = dfDataSetFeatureEngineer.where(col('nb_p
```

Histogram of y-label

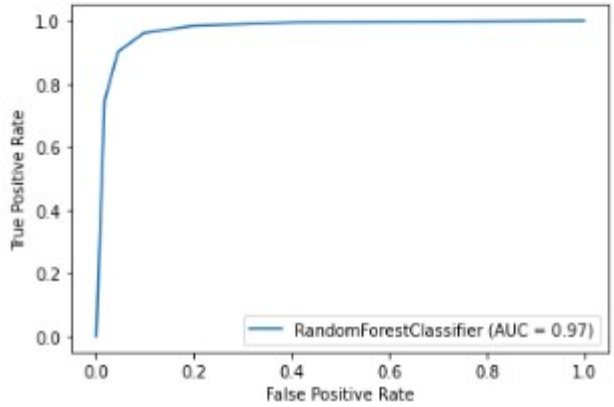


Some plots for training model, accuracy, predicting model

```
Accuracy Score is 0.93163
Precision Score is 0.90832
Recall Score is 0.90832
      0      1
0  9636   1034
1   425  10244
```




```
]:  1 plot_roc_curve(model, X_test, y_test)
    2 plt.show()
```



For more information read the **MLE_challenge - Features engineering - Notebook 2-withSolution.ipynb** notebook

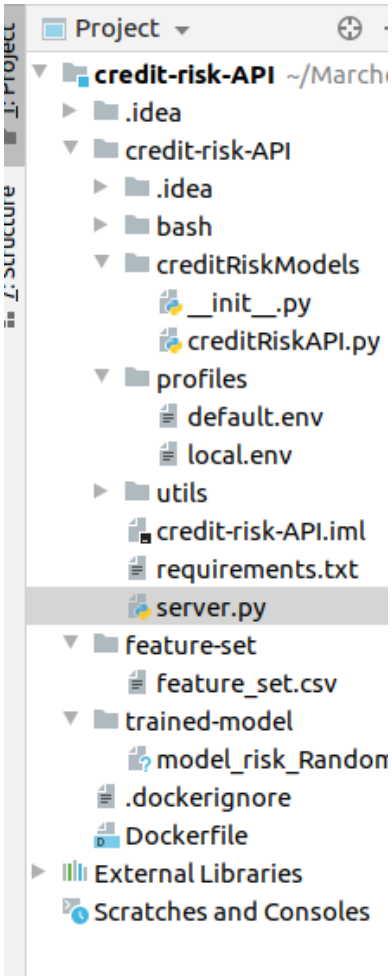
4. API design, Implementation

This section was implemented in the **credit-risk-API** IntellIj project. This project presents 4 services:

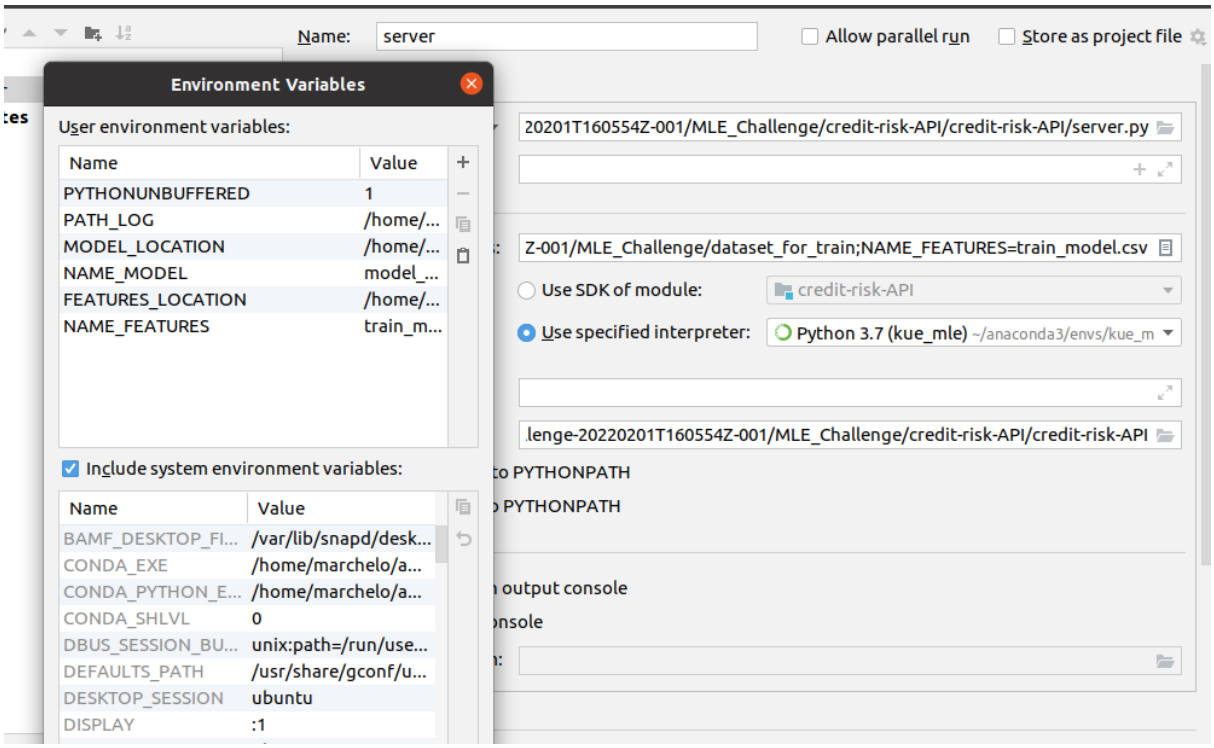
- **version-check** /GET method for getting version of component (1.0.0)
- **health-check** /GET method for getting status component
- **getFeaturesByUserId** GET/POST for getting the feature vector by user via web explorer or postman tool
- **prediction** /POST method for predicting the status base of the five features request.

* There are a process validation for each request type

Project structure

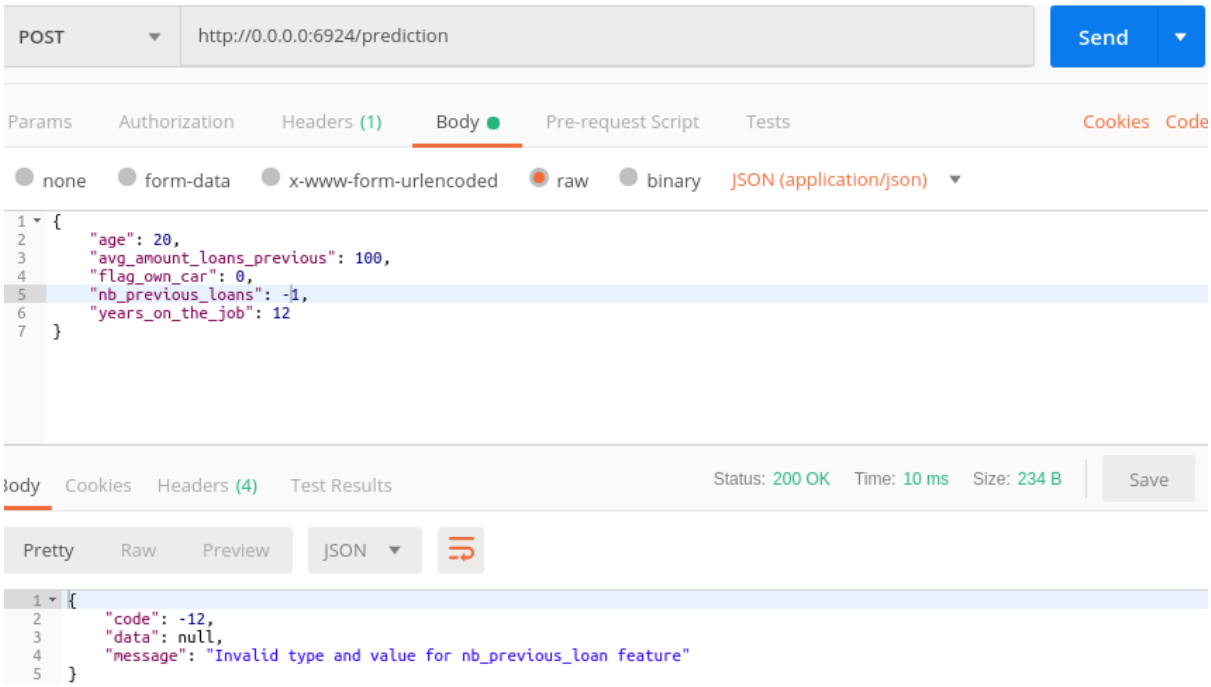


For running the project on local mode, this configuration is needed:



This some examples of execution of API

- Using postman tool:

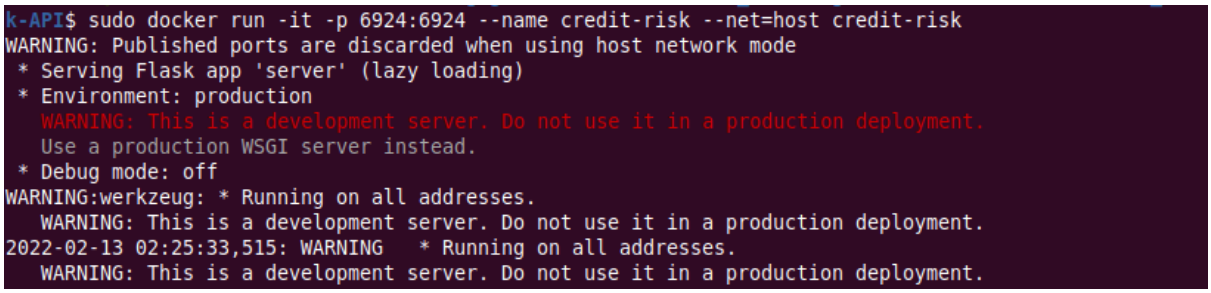


- Using web explorer url

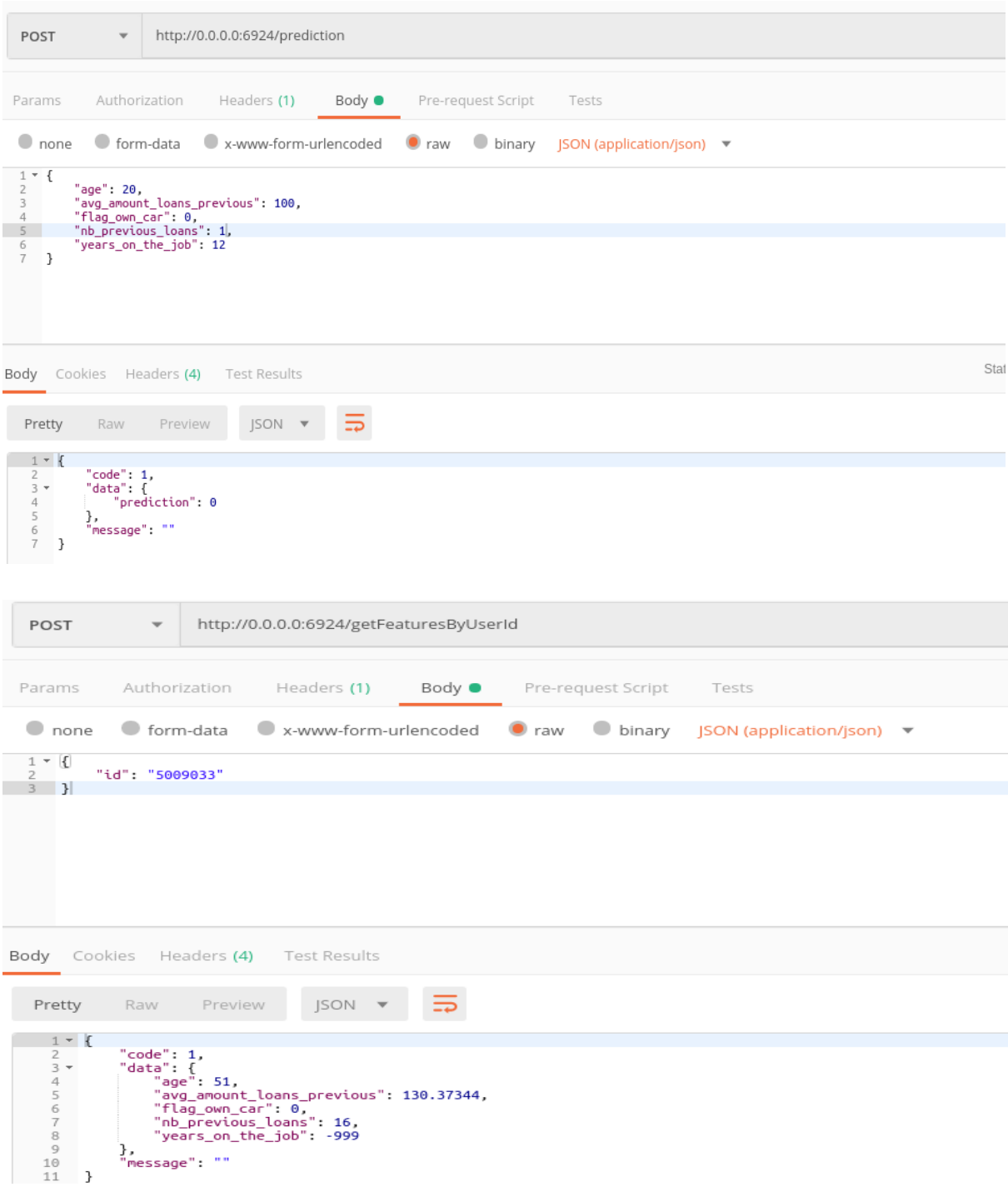
5. **Deploy API**

The following commands are needed for deploying the component in docker environment

```
$ cd {dir-project}
$ sudo docker build -t credit-risk .
$ sudo docker run -it -p 6924:6924 --name credit-risk --net=host credit-risk
```



These are examples of using the component:


POST
http://0.0.0.0:6924/prediction
Params
Authorization
Headers (1)
Body
Pre-request Script
Tests
none
form-data
x-www-form-urlencoded
raw
binary
JSON (application/json)
{
"age": 20,
"avg_amount_loans_previous": 100,
"flag_own_car": 0,
"nb_previous_loans": 1,
"years_on_the_job": 12
}
Body
Cookies
Headers (4)
Test Results
Pretty
Raw
Preview
JSON
{
"code": 1,
"data": {
"prediction": 0
},
"message": ""
}

POST
http://0.0.0.0:6924/getFeaturesByUserId
Params
Authorization
Headers (1)
Body
Pre-request Script
Tests
none
form-data
x-www-form-urlencoded
raw
binary
JSON (application/json)
{
"id": "5009033"
}
Body
Cookies
Headers (4)
Test Results
Pretty
Raw
Preview
JSON
{
"code": 1,
"data": {
"age": 51,
"avg_amount_loans_previous": 130.37344,
"flag_own_car": 0,
"nb_previous_loans": 16,
"years_on_the_job": -999
},
"message": ""
}