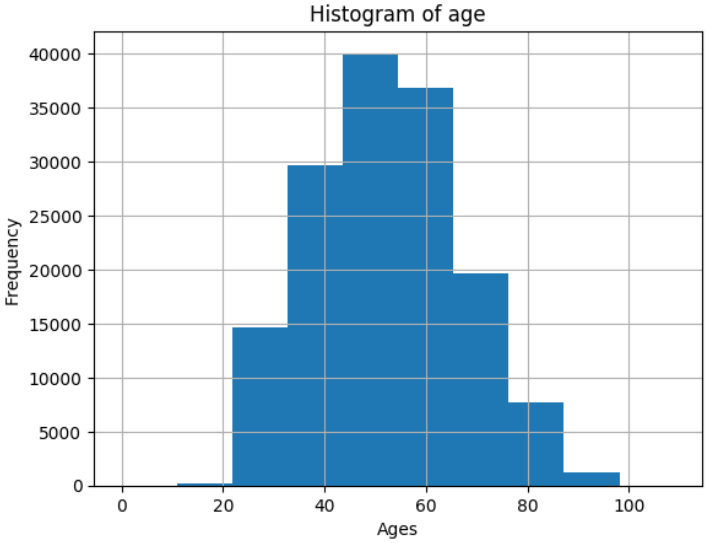
Rodrigo Valdes Ortiz
Machine Learning for Public Policy

Assignment #2

1. Read data. See python file. Defined functions:
    a. load_data(NAME_FILE) → Returns a dataframe with small names of the columns, in case the name of the columns are large.

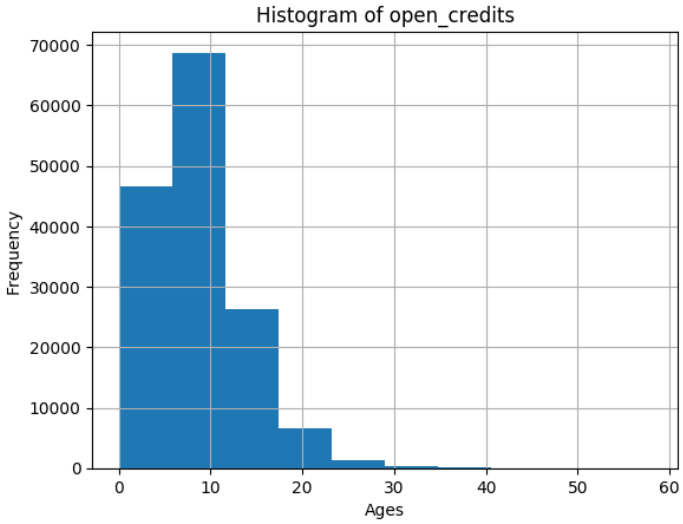
2. Explore data

Graph 1. Ages



Most of the population in this dataset is about 50 years old.

Graph 2. Open Credits



The proportion of those with more than eight credits is quite high.

Table 1. Summary statistics by 90 past due delinquency or worse

```
In [16]: d_serious
Out[16]:
   serious_dlq      PersonID  revolving        age       zipcode  30_59_days  \
0            0  74968.042429   6.168855  52.751375  60648.732507    0.280109
1            1  75453.643427   4.367282  45.926591  60649.892081    2.388490

    DebtRatio  MonthlyIncome  open_credits   90_late  real_state  60_89_days  \
0  357.151168    6732.277204      8.493620  0.135225    1.020368    0.126666
1  295.121066    5803.851610      7.882306  2.091362    0.988530    1.828047

   dependents
0    0.743787
1    0.944798
```

It is observed that the mean age for people that experienced 90 days past due delinquency is lower that the complement. In the same vein, their income is lower for about 900 USD monthly. Furthermore, the average number of dependents is higher.

3. Pre-Process Data. Please see the python file. Defined functions.
    a. describe_data(df) → Auxiliary function to undertand the general characteristics of the data.
    b. fill_na(df) → fill NaNs with the mean.

4. Generate Features/Predictors. Please see python file. Relevant defined functions.
    a. make_discrete(df, name_column, name_new_column, number_buckets = 5) → Dicretize a continuous variable, returns a dataframe.
    b. categorical_to_dummy(df, name_column, name_new_column, threshold) → transforms a categorical variable to a dummy one, given a threshold.

5. Build Classifier. I built a Logistic Regression Model, with this equation,

serious_dlq ~ age + MonthlyIncome + dependents

6. Evaluate Classifier
    a. The accuracy is quite high, 0.93 against 0.07 of the simple mean. This is in the case where I do not split in test and training set. However, the model predicts zeros all time. Then, this is not the best model.
    b. In the cross-validation approach, where I use 30% of testing set, the results are similar. The accuracy is 0.932, and the ROC is 0.6451.
    c. Classification report

```
             precision    recall  f1-score   support

        0.0       0.93      1.00      0.97     41965
        1.0       0.00      0.00      0.00      3035

avg / total       0.87      0.93      0.90     45000
```

As showed here, the model has a precision and recall of zero to predict our relevant variable. Then, despite an apparently high accuracy, this model is quite bad. It is easier predict that every observation will be zero. The same conclusion is supported by the approach with test in sample, and cross validation.