FP_Group9_HCDR_Part_2

April 8, 2024

1 Final Project Phase 2 - Home Credit Default Risk

Spring 2024

Team Members: - Glen Colletti - Alex Bordanca - Paul Miller

1.1 Abstract

> "The project is based on the Home Credit Default Risk (HCDR) Kaggle Competition. The goal of the competition is to predict whether or not a client will repay a loan. In order to make sure that people who struggle to get loans due to insufficient or non-existent credit histories have a positive loan experience, Home Credit makes use of a variety of alternative data–including telco and transactional information–to predict their clients' repayment abilities."

During this phase of the project the team will accomplish the following tasks: - Download and load the data from Kaggle. - Perform exploratory data analysis (EDA). - Define several baseline classification models using KNN, XG Boost and Logistic Regression. - Create pipelines to standardize any numerical feature and one hot encode categorial features. - Calculate several metrics to evaluate each model and pipeline. - Record 2 minute video discussing our progress.

[35]:
```python
from pandas import concat, DataFrame,read_csv, set_option
import matplotlib.pyplot as plt
%matplotlib inline

from __future__ import print_function

import numpy as np

from sklearn.compose import ColumnTransformer
from sklearn.impute import SimpleImputer
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import accuracy_score, f1_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler, OneHotEncoder

from xgboost import XGBClassifier
```

```
np.random.seed(0)
```

All data downloaded from Kaggle:

https://www.kaggle.com/competitions/home-credit-default-risk/data

There are several other data files which contain information regarding a customer's financial profile, payment history etc obtained from Kaggle: - `bureau.csv` - `bureau_balance.csv` - `credit_card_balance.csv` - `installments_payments.csv` - `previous_application.csv` - `POS_CASH_balance.csv`

The baseline pipelines will only use data loaded from `application_train.csv`

[7]:
```
train_data = read_csv('data/application_train.csv')

print(f'Loaded {train_data.shape[0]:,} records.')
print()
train_data.head()
```

```
Loaded 307,511 records.
```

[7]:
```
   SK_ID_CURR  TARGET NAME_CONTRACT_TYPE CODE_GENDER FLAG_OWN_CAR  \
0      100002       1         Cash loans           M            N
1      100003       0         Cash loans           F            N
2      100004       0    Revolving loans           M            Y
3      100006       0         Cash loans           F            N
4      100007       0         Cash loans           M            N

  FLAG_OWN_REALTY  CNT_CHILDREN  AMT_INCOME_TOTAL  AMT_CREDIT  AMT_ANNUITY  \
0               Y             0          202500.0    406597.5      24700.5
1               N             0          270000.0   1293502.5      35698.5
2               Y             0           67500.0    135000.0       6750.0
3               Y             0          135000.0    312682.5      29686.5
4               Y             0          121500.0    513000.0      21865.5

   …  FLAG_DOCUMENT_18 FLAG_DOCUMENT_19 FLAG_DOCUMENT_20 FLAG_DOCUMENT_21  \
0  …                 0                0                0                0
1  …                 0                0                0                0
2  …                 0                0                0                0
3  …                 0                0                0                0
4  …                 0                0                0                0

  AMT_REQ_CREDIT_BUREAU_HOUR AMT_REQ_CREDIT_BUREAU_DAY  \
0                        0.0                       0.0
1                        0.0                       0.0
2                        0.0                       0.0
3                        NaN                       NaN
4                        0.0                       0.0
```

```
   AMT_REQ_CREDIT_BUREAU_WEEK  AMT_REQ_CREDIT_BUREAU_MON  \
0                         0.0                        0.0
1                         0.0                        0.0
2                         0.0                        0.0
3                         NaN                        NaN
4                         0.0                        0.0

   AMT_REQ_CREDIT_BUREAU_QRT  AMT_REQ_CREDIT_BUREAU_YEAR
0                        0.0                         1.0
1                        0.0                         0.0
2                        0.0                         0.0
3                        NaN                         NaN
4                        0.0                         0.0

[5 rows x 122 columns]
```

1.2 Data Dictionary

As provided in the Kaggle data.

[13]:
```python
data_dictionary = read_csv('data/HomeCredit_columns_description.csv')
data_dictionary.head(3)
```

[13]:
```
   Unnamed: 0                         Table                 Row  \
0           1  application_{train|test}.csv          SK_ID_CURR
1           2  application_{train|test}.csv              TARGET
2           5  application_{train|test}.csv  NAME_CONTRACT_TYPE

                Description  \
0
ID of loan in our sample
1  Target variable (1 - client with payment difficulties: he/she had late
payment more than X days on at least one of the first Y installments of the loan
in our sample, 0 - all other cases)
2
Identification if loan is cash or revolving

  Special
0     NaN
1     NaN
2     NaN
```

Display all columns and their descriptions in the application_train and test data.

[15]:
```python
# Drop first column (an unneeded index)
data_dictionary.drop('Unnamed: 0', axis=1, inplace=True)
```

```python
set_option('display.max_columns', None)
set_option('display.max_rows', None)
set_option('display.max_colwidth', None)
data_dictionary[data_dictionary['Table'] == 'application_{train|test}.csv']
```

[15]:

	Table	Row \
0	application_{train|test}.csv	SK_ID_CURR
1	application_{train|test}.csv	TARGET
2	application_{train|test}.csv	NAME_CONTRACT_TYPE
3	application_{train|test}.csv	CODE_GENDER
4	application_{train|test}.csv	FLAG_OWN_CAR
5	application_{train|test}.csv	FLAG_OWN_REALTY
6	application_{train|test}.csv	CNT_CHILDREN
7	application_{train|test}.csv	AMT_INCOME_TOTAL
8	application_{train|test}.csv	AMT_CREDIT
9	application_{train|test}.csv	AMT_ANNUITY
10	application_{train|test}.csv	AMT_GOODS_PRICE
11	application_{train|test}.csv	NAME_TYPE_SUITE
12	application_{train|test}.csv	NAME_INCOME_TYPE
13	application_{train|test}.csv	NAME_EDUCATION_TYPE
14	application_{train|test}.csv	NAME_FAMILY_STATUS
15	application_{train|test}.csv	NAME_HOUSING_TYPE
16	application_{train|test}.csv	REGION_POPULATION_RELATIVE
17	application_{train|test}.csv	DAYS_BIRTH
18	application_{train|test}.csv	DAYS_EMPLOYED
19	application_{train|test}.csv	DAYS_REGISTRATION
20	application_{train|test}.csv	DAYS_ID_PUBLISH
21	application_{train|test}.csv	OWN_CAR_AGE
22	application_{train|test}.csv	FLAG_MOBIL
23	application_{train|test}.csv	FLAG_EMP_PHONE
24	application_{train|test}.csv	FLAG_WORK_PHONE
25	application_{train|test}.csv	FLAG_CONT_MOBILE
26	application_{train|test}.csv	FLAG_PHONE
27	application_{train|test}.csv	FLAG_EMAIL
28	application_{train|test}.csv	OCCUPATION_TYPE
29	application_{train|test}.csv	CNT_FAM_MEMBERS
30	application_{train|test}.csv	REGION_RATING_CLIENT
31	application_{train|test}.csv	REGION_RATING_CLIENT_W_CITY
32	application_{train|test}.csv	WEEKDAY_APPR_PROCESS_START
33	application_{train|test}.csv	HOUR_APPR_PROCESS_START
34	application_{train|test}.csv	REG_REGION_NOT_LIVE_REGION
35	application_{train|test}.csv	REG_REGION_NOT_WORK_REGION
36	application_{train|test}.csv	LIVE_REGION_NOT_WORK_REGION
37	application_{train|test}.csv	REG_CITY_NOT_LIVE_CITY
38	application_{train|test}.csv	REG_CITY_NOT_WORK_CITY
39	application_{train|test}.csv	LIVE_CITY_NOT_WORK_CITY

40	application_{train|test}.csv	ORGANIZATION_TYPE
41	application_{train|test}.csv	EXT_SOURCE_1
42	application_{train|test}.csv	EXT_SOURCE_2
43	application_{train|test}.csv	EXT_SOURCE_3
44	application_{train|test}.csv	APARTMENTS_AVG
45	application_{train|test}.csv	BASEMENTAREA_AVG
46	application_{train|test}.csv	YEARS_BEGINEXPLUATATION_AVG
47	application_{train|test}.csv	YEARS_BUILD_AVG
48	application_{train|test}.csv	COMMONAREA_AVG
49	application_{train|test}.csv	ELEVATORS_AVG
50	application_{train|test}.csv	ENTRANCES_AVG
51	application_{train|test}.csv	FLOORSMAX_AVG
52	application_{train|test}.csv	FLOORSMIN_AVG
53	application_{train|test}.csv	LANDAREA_AVG
54	application_{train|test}.csv	LIVINGAPARTMENTS_AVG
55	application_{train|test}.csv	LIVINGAREA_AVG
56	application_{train|test}.csv	NONLIVINGAPARTMENTS_AVG
57	application_{train|test}.csv	NONLIVINGAREA_AVG
58	application_{train|test}.csv	APARTMENTS_MODE
59	application_{train|test}.csv	BASEMENTAREA_MODE
60	application_{train|test}.csv	YEARS_BEGINEXPLUATATION_MODE
61	application_{train|test}.csv	YEARS_BUILD_MODE
62	application_{train|test}.csv	COMMONAREA_MODE
63	application_{train|test}.csv	ELEVATORS_MODE
64	application_{train|test}.csv	ENTRANCES_MODE
65	application_{train|test}.csv	FLOORSMAX_MODE
66	application_{train|test}.csv	FLOORSMIN_MODE
67	application_{train|test}.csv	LANDAREA_MODE
68	application_{train|test}.csv	LIVINGAPARTMENTS_MODE
69	application_{train|test}.csv	LIVINGAREA_MODE
70	application_{train|test}.csv	NONLIVINGAPARTMENTS_MODE
71	application_{train|test}.csv	NONLIVINGAREA_MODE
72	application_{train|test}.csv	APARTMENTS_MEDI
73	application_{train|test}.csv	BASEMENTAREA_MEDI
74	application_{train|test}.csv	YEARS_BEGINEXPLUATATION_MEDI
75	application_{train|test}.csv	YEARS_BUILD_MEDI
76	application_{train|test}.csv	COMMONAREA_MEDI
77	application_{train|test}.csv	ELEVATORS_MEDI
78	application_{train|test}.csv	ENTRANCES_MEDI
79	application_{train|test}.csv	FLOORSMAX_MEDI
80	application_{train|test}.csv	FLOORSMIN_MEDI
81	application_{train|test}.csv	LANDAREA_MEDI
82	application_{train|test}.csv	LIVINGAPARTMENTS_MEDI
83	application_{train|test}.csv	LIVINGAREA_MEDI
84	application_{train|test}.csv	NONLIVINGAPARTMENTS_MEDI
85	application_{train|test}.csv	NONLIVINGAREA_MEDI
86	application_{train|test}.csv	FONDKAPREMONT_MODE

```
87   application_{train|test}.csv             HOUSETYPE_MODE
88   application_{train|test}.csv             TOTALAREA_MODE
89   application_{train|test}.csv         WALLSMATERIAL_MODE
90   application_{train|test}.csv        EMERGENCYSTATE_MODE
91   application_{train|test}.csv   OBS_30_CNT_SOCIAL_CIRCLE
92   application_{train|test}.csv   DEF_30_CNT_SOCIAL_CIRCLE
93   application_{train|test}.csv   OBS_60_CNT_SOCIAL_CIRCLE
94   application_{train|test}.csv   DEF_60_CNT_SOCIAL_CIRCLE
95   application_{train|test}.csv     DAYS_LAST_PHONE_CHANGE
96   application_{train|test}.csv            FLAG_DOCUMENT_2
97   application_{train|test}.csv            FLAG_DOCUMENT_3
98   application_{train|test}.csv            FLAG_DOCUMENT_4
99   application_{train|test}.csv            FLAG_DOCUMENT_5
100  application_{train|test}.csv            FLAG_DOCUMENT_6
101  application_{train|test}.csv            FLAG_DOCUMENT_7
102  application_{train|test}.csv            FLAG_DOCUMENT_8
103  application_{train|test}.csv            FLAG_DOCUMENT_9
104  application_{train|test}.csv           FLAG_DOCUMENT_10
105  application_{train|test}.csv           FLAG_DOCUMENT_11
106  application_{train|test}.csv           FLAG_DOCUMENT_12
107  application_{train|test}.csv           FLAG_DOCUMENT_13
108  application_{train|test}.csv           FLAG_DOCUMENT_14
109  application_{train|test}.csv           FLAG_DOCUMENT_15
110  application_{train|test}.csv           FLAG_DOCUMENT_16
111  application_{train|test}.csv           FLAG_DOCUMENT_17
112  application_{train|test}.csv           FLAG_DOCUMENT_18
113  application_{train|test}.csv           FLAG_DOCUMENT_19
114  application_{train|test}.csv           FLAG_DOCUMENT_20
115  application_{train|test}.csv           FLAG_DOCUMENT_21
116  application_{train|test}.csv AMT_REQ_CREDIT_BUREAU_HOUR
117  application_{train|test}.csv  AMT_REQ_CREDIT_BUREAU_DAY
118  application_{train|test}.csv AMT_REQ_CREDIT_BUREAU_WEEK
119  application_{train|test}.csv  AMT_REQ_CREDIT_BUREAU_MON
120  application_{train|test}.csv  AMT_REQ_CREDIT_BUREAU_QRT
121  application_{train|test}.csv AMT_REQ_CREDIT_BUREAU_YEAR

                              Description  \
0
ID of loan in our sample
1
Target variable (1 - client with payment difficulties: he/she had late payment
more than X days on at least one of the first Y installments of the loan in our
sample, 0 - all other cases)
2
Identification if loan is cash or revolving
3
Gender of the client
```

4
Flag if the client owns a car
5
Flag if client owns a house or flat
6
Number of children the client has
7
Income of the client
8
Credit amount of the loan
9
Loan annuity
10
For consumer loans it is the price of the goods for which the loan is given
11
Who was accompanying client when he was applying for the loan
12
Clients income type (businessman, working, maternity leave,…)
13
Level of highest education the client achieved
14
Family status of the client
15
What is the housing situation of the client (renting, living with parents, …)
16
Normalized population of region where client lives (higher number means the client lives in more populated region)
17
Client's age in days at the time of application
18
How many days before the application the person started current employment
19
How many days before the application did client change his registration
20
How many days before the application did client change the identity document with which he applied for the loan
21
Age of client's car
22
Did client provide mobile phone (1=YES, 0=NO)
23
Did client provide work phone (1=YES, 0=NO)
24
Did client provide home phone (1=YES, 0=NO)
25
Was mobile phone reachable (1=YES, 0=NO)
26

Did client provide home phone (1=YES, 0=NO)
27
Did client provide email (1=YES, 0=NO)
28
What kind of occupation does the client have
29
How many family members does client have
30
Our rating of the region where client lives (1,2,3)
31
Our rating of the region where client lives with taking city into account (1,2,3)
32
On which day of the week did the client apply for the loan
33
Approximately at what hour did the client apply for the loan
34
Flag if client's permanent address does not match contact address (1=different, 0=same, at region level)
35
Flag if client's permanent address does not match work address (1=different, 0=same, at region level)
36
Flag if client's contact address does not match work address (1=different, 0=same, at region level)
37
Flag if client's permanent address does not match contact address (1=different, 0=same, at city level)
38
Flag if client's permanent address does not match work address (1=different, 0=same, at city level)
39
Flag if client's contact address does not match work address (1=different, 0=same, at city level)
40
Type of organization where client works
41
Normalized score from external data source
42
Normalized score from external data source
43
Normalized score from external data source
44 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
45 Normalized information about building where the client lives, What is

average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
46 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
47 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
48 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
49 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
50 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
51 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
52 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
53 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
54 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
55 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
56 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor

57 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
58 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
59 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
60 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
61 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
62 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
63 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
64 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
65 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
66 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
67 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
68 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of

entrances, state of the building, number of floor
69 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
70 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
71 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
72 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
73 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
74 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
75 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
76 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
77 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
78 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
79 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
80 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment

size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
81 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
82 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
83 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
84 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
85 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
86 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
87 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
88 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
89 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
90 Normalized information about building where the client lives, What is average (_AVG suffix), modus (_MODE suffix), median (_MEDI suffix) apartment size, common area, living area, age of building, number of elevators, number of entrances, state of the building, number of floor
91
How many observation of client's social surroundings with observable 30 DPD (days past due) default
92
How many observation of client's social surroundings defaulted on 30 DPD (days

past due)
93
How many observation of client's social surroundings with observable 60 DPD (days past due) default
94
How many observation of client's social surroundings defaulted on 60 (days past due) DPD
95
How many days before application did client change phone
96
Did client provide document 2
97
Did client provide document 3
98
Did client provide document 4
99
Did client provide document 5
100
Did client provide document 6
101
Did client provide document 7
102
Did client provide document 8
103
Did client provide document 9
104
Did client provide document 10
105
Did client provide document 11
106
Did client provide document 12
107
Did client provide document 13
108
Did client provide document 14
109
Did client provide document 15
110
Did client provide document 16
111
Did client provide document 17
112
Did client provide document 18
113
Did client provide document 19
114
Did client provide document 20

```
115
Did client provide document 21
116
Number of enquiries to Credit Bureau about the client one hour before
application
117
Number of enquiries to Credit Bureau about the client one day before application
(excluding one hour before application)
118
Number of enquiries to Credit Bureau about the client one week before
application (excluding one day before application)
119
Number of enquiries to Credit Bureau about the client one month before
application (excluding one week before application)
120
Number of enquiries to Credit Bureau about the client 3 month before application
(excluding one month before application)
121
Number of enquiries to Credit Bureau about the client one day year (excluding
last 3 months before application)

                                   Special
0                                      NaN
1                                      NaN
2                                      NaN
3                                      NaN
4                                      NaN
5                                      NaN
6                                      NaN
7                                      NaN
8                                      NaN
9                                      NaN
10                                     NaN
11                                     NaN
12                                     NaN
13                                     NaN
14                                     NaN
15                                     NaN
16                             normalized 
17   time only relative to the application
18   time only relative to the application
19   time only relative to the application
20   time only relative to the application
21                                     NaN
22                                     NaN
23                                     NaN
24                                     NaN
```

25	NaN
26	NaN
27	NaN
28	NaN
29	NaN
30	NaN
31	NaN
32	NaN
33	rounded
34	NaN
35	NaN
36	NaN
37	NaN
38	NaN
39	NaN
40	NaN
41	normalized
42	normalized
43	normalized
44	normalized
45	normalized
46	normalized
47	normalized
48	normalized
49	normalized
50	normalized
51	normalized
52	normalized
53	normalized
54	normalized
55	normalized
56	normalized
57	normalized
58	normalized
59	normalized
60	normalized
61	normalized
62	normalized
63	normalized
64	normalized
65	normalized
66	normalized
67	normalized
68	normalized
69	normalized
70	normalized
71	normalized

72	normalized
73	normalized
74	normalized
75	normalized
76	normalized
77	normalized
78	normalized
79	normalized
80	normalized
81	normalized
82	normalized
83	normalized
84	normalized
85	normalized
86	normalized
87	normalized
88	normalized
89	normalized
90	normalized
91	NaN
92	NaN
93	NaN
94	NaN
95	NaN
96	NaN
97	NaN
98	NaN
99	NaN
100	NaN
101	NaN
102	NaN
103	NaN
104	NaN
105	NaN
106	NaN
107	NaN
108	NaN
109	NaN
110	NaN
111	NaN
112	NaN
113	NaN
114	NaN
115	NaN
116	NaN
117	NaN
118	NaN

```
119                                      NaN
120                                      NaN
121                                      NaN
```

1.3 Exploratory Data Analysis (EDA)

[10]:
```
train_data.info(verbose=True)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Data columns (total 122 columns):
 #    Column                         Dtype
---   ------                         -----
 0    SK_ID_CURR                     int64
 1    TARGET                         int64
 2    NAME_CONTRACT_TYPE             object
 3    CODE_GENDER                    object
 4    FLAG_OWN_CAR                   object
 5    FLAG_OWN_REALTY                object
 6    CNT_CHILDREN                   int64
 7    AMT_INCOME_TOTAL               float64
 8    AMT_CREDIT                     float64
 9    AMT_ANNUITY                    float64
 10   AMT_GOODS_PRICE                float64
 11   NAME_TYPE_SUITE                object
 12   NAME_INCOME_TYPE               object
 13   NAME_EDUCATION_TYPE            object
 14   NAME_FAMILY_STATUS             object
 15   NAME_HOUSING_TYPE              object
 16   REGION_POPULATION_RELATIVE     float64
 17   DAYS_BIRTH                     int64
 18   DAYS_EMPLOYED                  int64
 19   DAYS_REGISTRATION              float64
 20   DAYS_ID_PUBLISH                int64
 21   OWN_CAR_AGE                    float64
 22   FLAG_MOBIL                     int64
 23   FLAG_EMP_PHONE                 int64
 24   FLAG_WORK_PHONE                int64
 25   FLAG_CONT_MOBILE               int64
 26   FLAG_PHONE                     int64
 27   FLAG_EMAIL                     int64
 28   OCCUPATION_TYPE                object
 29   CNT_FAM_MEMBERS                float64
 30   REGION_RATING_CLIENT           int64
 31   REGION_RATING_CLIENT_W_CITY    int64
 32   WEEKDAY_APPR_PROCESS_START     object
 33   HOUR_APPR_PROCESS_START        int64
 34   REG_REGION_NOT_LIVE_REGION     int64
```

35	REG_REGION_NOT_WORK_REGION	int64
36	LIVE_REGION_NOT_WORK_REGION	int64
37	REG_CITY_NOT_LIVE_CITY	int64
38	REG_CITY_NOT_WORK_CITY	int64
39	LIVE_CITY_NOT_WORK_CITY	int64
40	ORGANIZATION_TYPE	object
41	EXT_SOURCE_1	float64
42	EXT_SOURCE_2	float64
43	EXT_SOURCE_3	float64
44	APARTMENTS_AVG	float64
45	BASEMENTAREA_AVG	float64
46	YEARS_BEGINEXPLUATATION_AVG	float64
47	YEARS_BUILD_AVG	float64
48	COMMONAREA_AVG	float64
49	ELEVATORS_AVG	float64
50	ENTRANCES_AVG	float64
51	FLOORSMAX_AVG	float64
52	FLOORSMIN_AVG	float64
53	LANDAREA_AVG	float64
54	LIVINGAPARTMENTS_AVG	float64
55	LIVINGAREA_AVG	float64
56	NONLIVINGAPARTMENTS_AVG	float64
57	NONLIVINGAREA_AVG	float64
58	APARTMENTS_MODE	float64
59	BASEMENTAREA_MODE	float64
60	YEARS_BEGINEXPLUATATION_MODE	float64
61	YEARS_BUILD_MODE	float64
62	COMMONAREA_MODE	float64
63	ELEVATORS_MODE	float64
64	ENTRANCES_MODE	float64
65	FLOORSMAX_MODE	float64
66	FLOORSMIN_MODE	float64
67	LANDAREA_MODE	float64
68	LIVINGAPARTMENTS_MODE	float64
69	LIVINGAREA_MODE	float64
70	NONLIVINGAPARTMENTS_MODE	float64
71	NONLIVINGAREA_MODE	float64
72	APARTMENTS_MEDI	float64
73	BASEMENTAREA_MEDI	float64
74	YEARS_BEGINEXPLUATATION_MEDI	float64
75	YEARS_BUILD_MEDI	float64
76	COMMONAREA_MEDI	float64
77	ELEVATORS_MEDI	float64
78	ENTRANCES_MEDI	float64
79	FLOORSMAX_MEDI	float64
80	FLOORSMIN_MEDI	float64
81	LANDAREA_MEDI	float64
82	LIVINGAPARTMENTS_MEDI	float64

```
 83   LIVINGAREA_MEDI               float64
 84   NONLIVINGAPARTMENTS_MEDI      float64
 85   NONLIVINGAREA_MEDI            float64
 86   FONDKAPREMONT_MODE            object
 87   HOUSETYPE_MODE                object
 88   TOTALAREA_MODE                float64
 89   WALLSMATERIAL_MODE            object
 90   EMERGENCYSTATE_MODE           object
 91   OBS_30_CNT_SOCIAL_CIRCLE      float64
 92   DEF_30_CNT_SOCIAL_CIRCLE      float64
 93   OBS_60_CNT_SOCIAL_CIRCLE      float64
 94   DEF_60_CNT_SOCIAL_CIRCLE      float64
 95   DAYS_LAST_PHONE_CHANGE        float64
 96   FLAG_DOCUMENT_2               int64
 97   FLAG_DOCUMENT_3               int64
 98   FLAG_DOCUMENT_4               int64
 99   FLAG_DOCUMENT_5               int64
 100  FLAG_DOCUMENT_6               int64
 101  FLAG_DOCUMENT_7               int64
 102  FLAG_DOCUMENT_8               int64
 103  FLAG_DOCUMENT_9               int64
 104  FLAG_DOCUMENT_10              int64
 105  FLAG_DOCUMENT_11              int64
 106  FLAG_DOCUMENT_12              int64
 107  FLAG_DOCUMENT_13              int64
 108  FLAG_DOCUMENT_14              int64
 109  FLAG_DOCUMENT_15              int64
 110  FLAG_DOCUMENT_16              int64
 111  FLAG_DOCUMENT_17              int64
 112  FLAG_DOCUMENT_18              int64
 113  FLAG_DOCUMENT_19              int64
 114  FLAG_DOCUMENT_20              int64
 115  FLAG_DOCUMENT_21              int64
 116  AMT_REQ_CREDIT_BUREAU_HOUR    float64
 117  AMT_REQ_CREDIT_BUREAU_DAY     float64
 118  AMT_REQ_CREDIT_BUREAU_WEEK    float64
 119  AMT_REQ_CREDIT_BUREAU_MON     float64
 120  AMT_REQ_CREDIT_BUREAU_QRT     float64
 121  AMT_REQ_CREDIT_BUREAU_YEAR    float64
dtypes: float64(65), int64(41), object(16)
memory usage: 286.2+ MB
```

[ ]:
```
# NUMERICAL FEATURES
train_data.describe()
```

[ ]:
```
# ALL FEATURES
train_data.describe(include='all')
```

Explore missing data in training dataset

[18]:
```python
#MISSING DATA FOR APPLICATION TRAIN

percent = (train_data.isnull().sum() / train_data.isnull().count()*100).
↪sort_values(ascending = False).round(2)
sum_missing = train_data.isna().sum().sort_values(ascending = False)
missing_application_train_data  = concat([percent, sum_missing], axis=1,␣
↪keys=['Percent', "Train Missing Count"])
missing_application_train_data.head(20)
```

[18]:

	Percent	Train Missing Count
COMMONAREA_MEDI	69.87	214865
COMMONAREA_AVG	69.87	214865
COMMONAREA_MODE	69.87	214865
NONLIVINGAPARTMENTS_MODE	69.43	213514
NONLIVINGAPARTMENTS_AVG	69.43	213514
NONLIVINGAPARTMENTS_MEDI	69.43	213514
FONDKAPREMONT_MODE	68.39	210295
LIVINGAPARTMENTS_MODE	68.35	210199
LIVINGAPARTMENTS_AVG	68.35	210199
LIVINGAPARTMENTS_MEDI	68.35	210199
FLOORSMIN_AVG	67.85	208642
FLOORSMIN_MODE	67.85	208642
FLOORSMIN_MEDI	67.85	208642
YEARS_BUILD_MEDI	66.50	204488
YEARS_BUILD_MODE	66.50	204488
YEARS_BUILD_AVG	66.50	204488
OWN_CAR_AGE	65.99	202929
LANDAREA_MEDI	59.38	182590
LANDAREA_MODE	59.38	182590
LANDAREA_AVG	59.38	182590

[ ]:

[ ]:

1.4 Correlations

Find the features most correlated to the target. List the 10 most positive and 10 most negative

[22]:
```python
correlations = train_data.corr()['TARGET'].sort_values()
print('Most Positive Correlations:\n', correlations.tail(10))
print('\nMost Negative Correlations:\n', correlations.head(10))
```

```
Most Positive Correlations:
 FLAG_DOCUMENT_3               0.044346
REG_CITY_NOT_LIVE_CITY        0.044395
FLAG_EMP_PHONE                0.045982
```

```
REG_CITY_NOT_WORK_CITY         0.050994
DAYS_ID_PUBLISH                0.051457
DAYS_LAST_PHONE_CHANGE         0.055218
REGION_RATING_CLIENT           0.058899
REGION_RATING_CLIENT_W_CITY    0.060893
DAYS_BIRTH                     0.078239
TARGET                         1.000000
Name: TARGET, dtype: float64

Most Negative Correlations:
 EXT_SOURCE_3                   -0.178919
EXT_SOURCE_2                   -0.160472
EXT_SOURCE_1                   -0.155317
DAYS_EMPLOYED                  -0.044932
FLOORSMAX_AVG                  -0.044003
FLOORSMAX_MEDI                 -0.043768
FLOORSMAX_MODE                 -0.043226
AMT_GOODS_PRICE                -0.039645
REGION_POPULATION_RELATIVE     -0.037227
ELEVATORS_AVG                  -0.034199
Name: TARGET, dtype: float64
```

Print out most correlated features in list form to use when modeling.

[26]:
```
list(correlations[0:10].index)
```

[26]:
```
['EXT_SOURCE_3',
 'EXT_SOURCE_2',
 'EXT_SOURCE_1',
 'DAYS_EMPLOYED',
 'FLOORSMAX_AVG',
 'FLOORSMAX_MEDI',
 'FLOORSMAX_MODE',
 'AMT_GOODS_PRICE',
 'REGION_POPULATION_RELATIVE',
 'ELEVATORS_AVG']
```

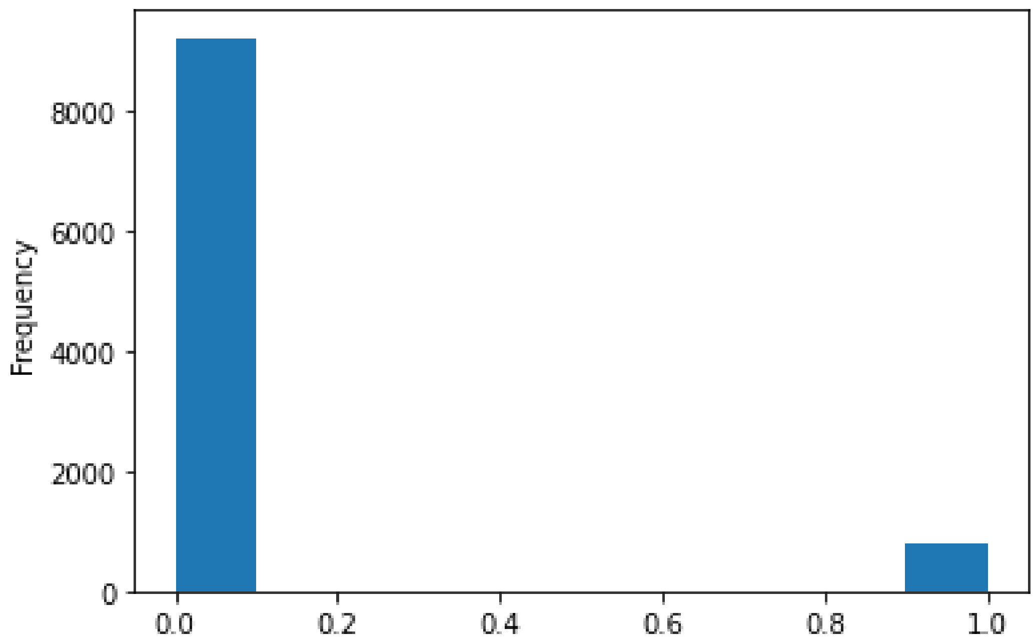
[27]:
```
list(correlations.tail(10).index)
```

[27]:
```
['FLAG_DOCUMENT_3',
 'REG_CITY_NOT_LIVE_CITY',
 'FLAG_EMP_PHONE',
 'REG_CITY_NOT_WORK_CITY',
 'DAYS_ID_PUBLISH',
 'DAYS_LAST_PHONE_CHANGE',
 'REGION_RATING_CLIENT',
 'REGION_RATING_CLIENT_W_CITY',
 'DAYS_BIRTH',
```

'TARGET']

1.5 Visual EDA

[60]:
```
#EXPLORE DISTRIBUTION OF TARGET VARIABLE

train_data["TARGET"].plot.hist();
```



[68]:
```
train_data['TARGET'].value_counts(normalize=True)
```

[68]:
```
0    0.9227
1    0.0773
Name: TARGET, dtype: float64
```

Inspect features most correlated to the target variable.

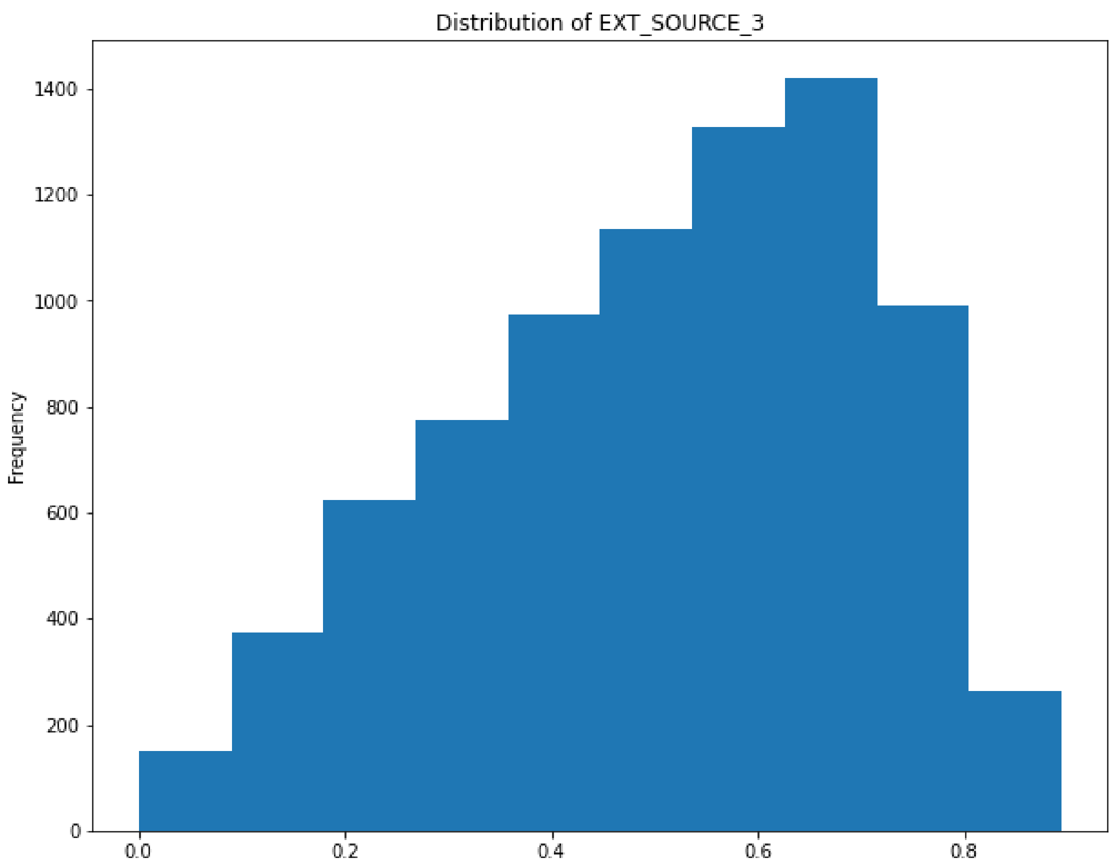
[61]:
```
most_pos_corr = correlations.tail(11)
most_neg_corr = correlations.head(10)

neg_cols = most_neg_corr.index.to_list()
pos_cols = most_pos_corr.index.to_list()
```

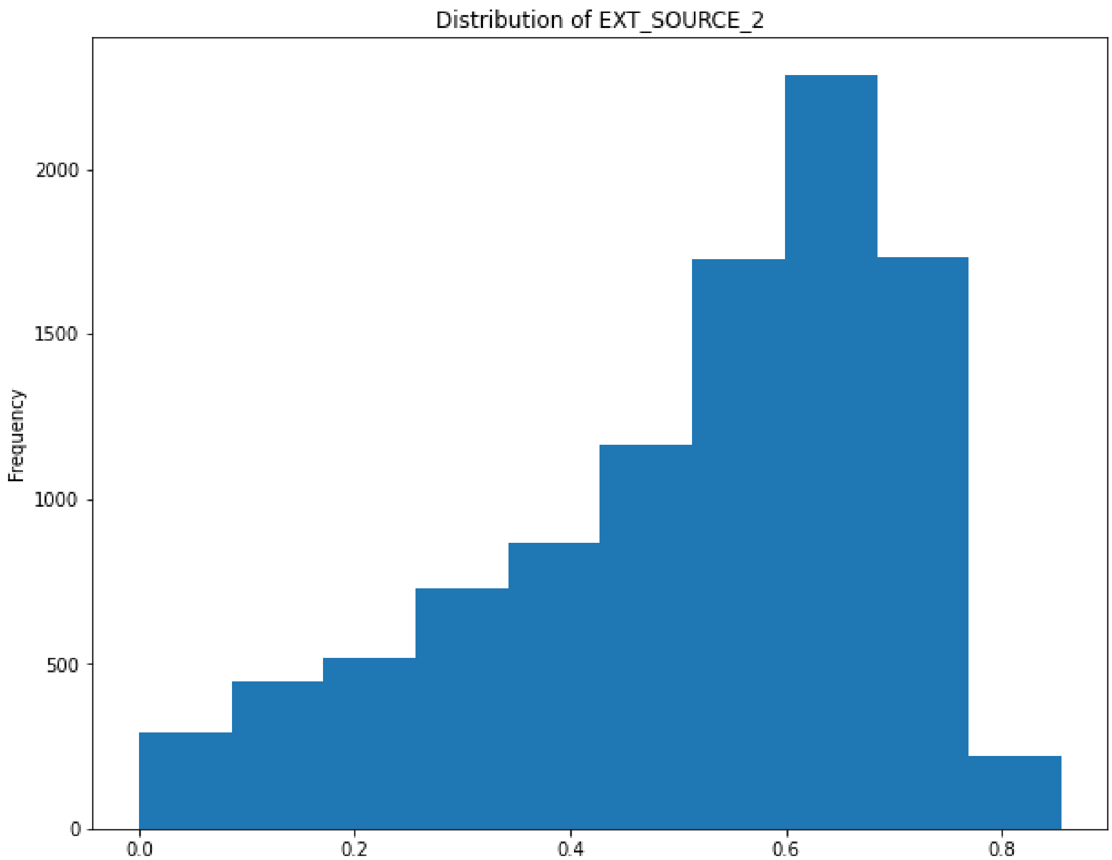
[62]:
```
neg_cols
```

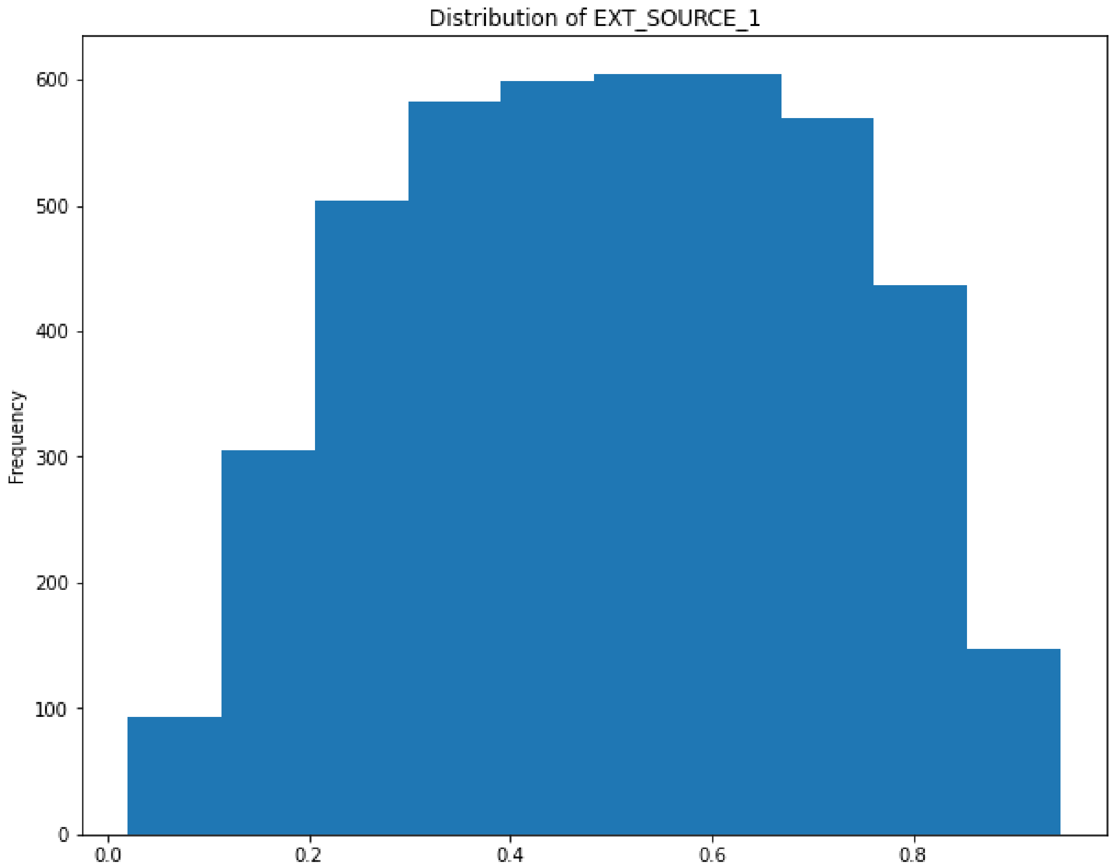
[62]:
```
['EXT_SOURCE_3',
 'EXT_SOURCE_2',
```

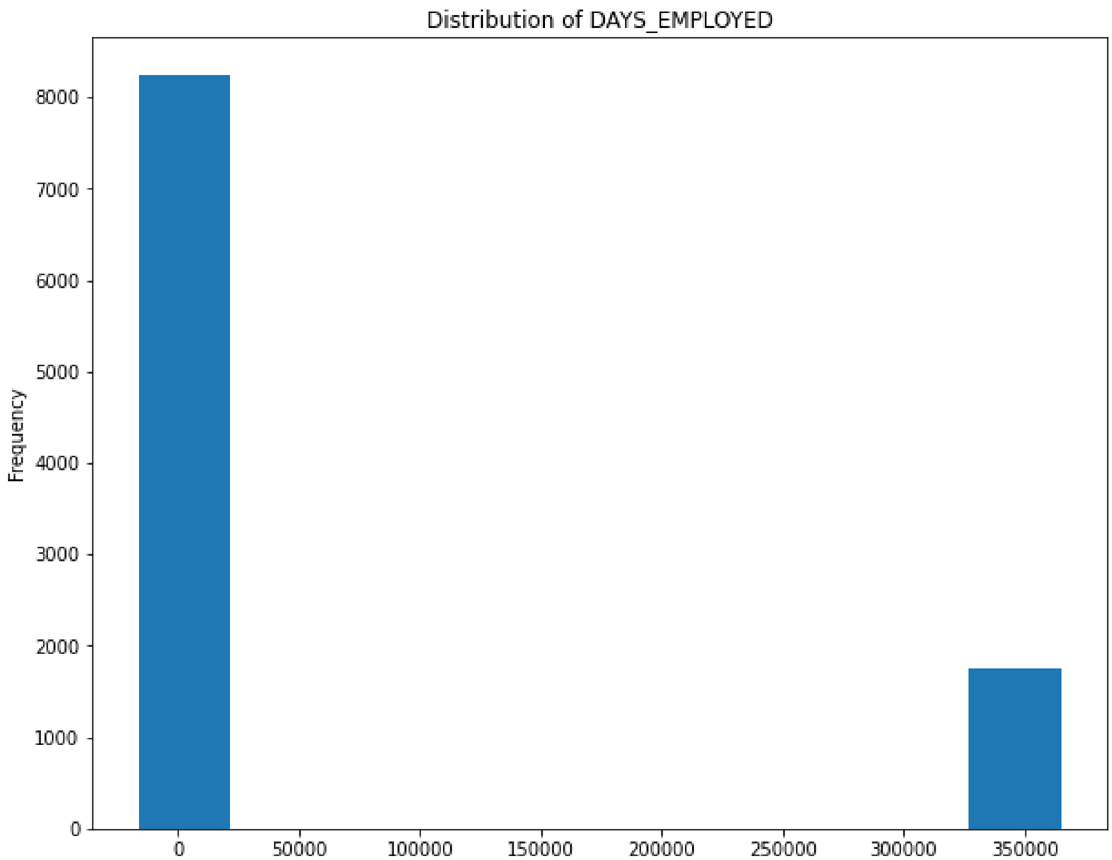
```
'EXT_SOURCE_1',
'DAYS_EMPLOYED',
'FLOORSMAX_AVG',
'FLOORSMAX_MEDI',
'FLOORSMAX_MODE',
'AMT_GOODS_PRICE',
'REGION_POPULATION_RELATIVE',
'ELEVATORS_AVG']
```

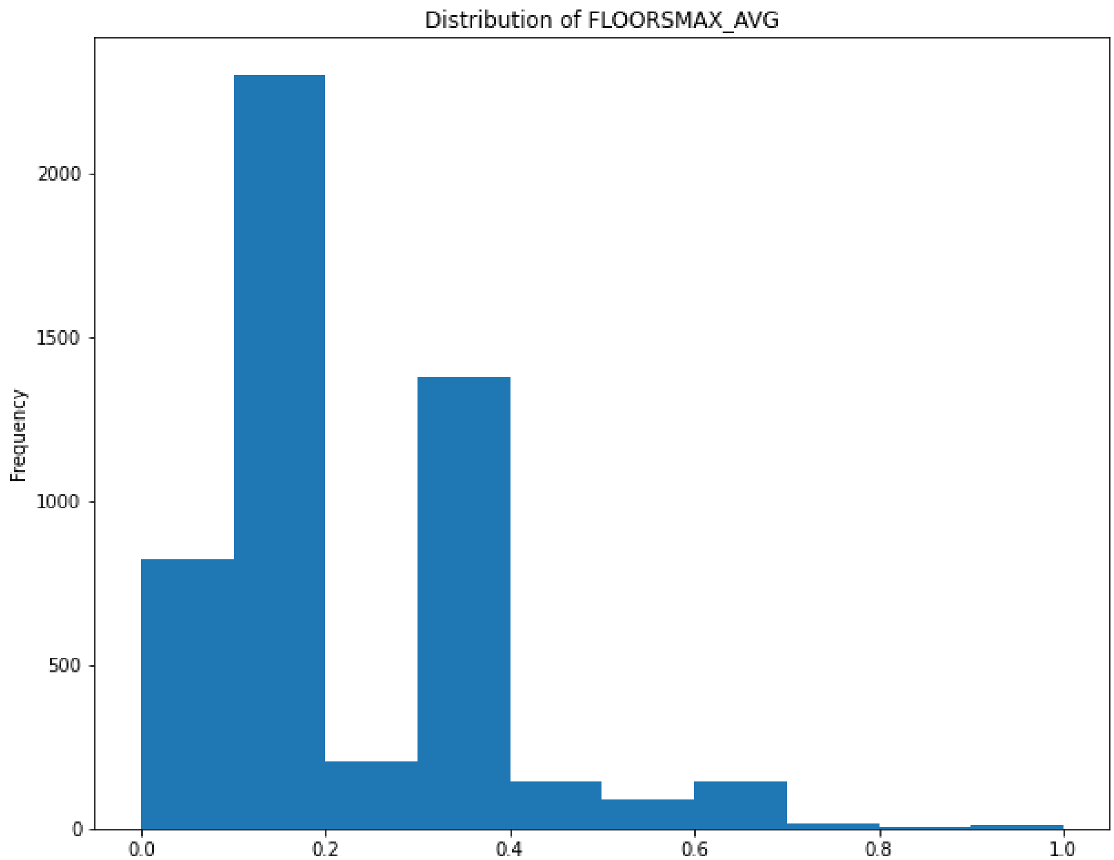
[64]:
```
#EXPLORE DISTRIBUTION FOR MOST NEGATIVELY CORRELATED FEATURES

for col in neg_cols:
    plt.figure(figsize=(10,8))
    plt.title(f"Distribution of {col}")
    train_data[col].dropna().plot.hist()
```

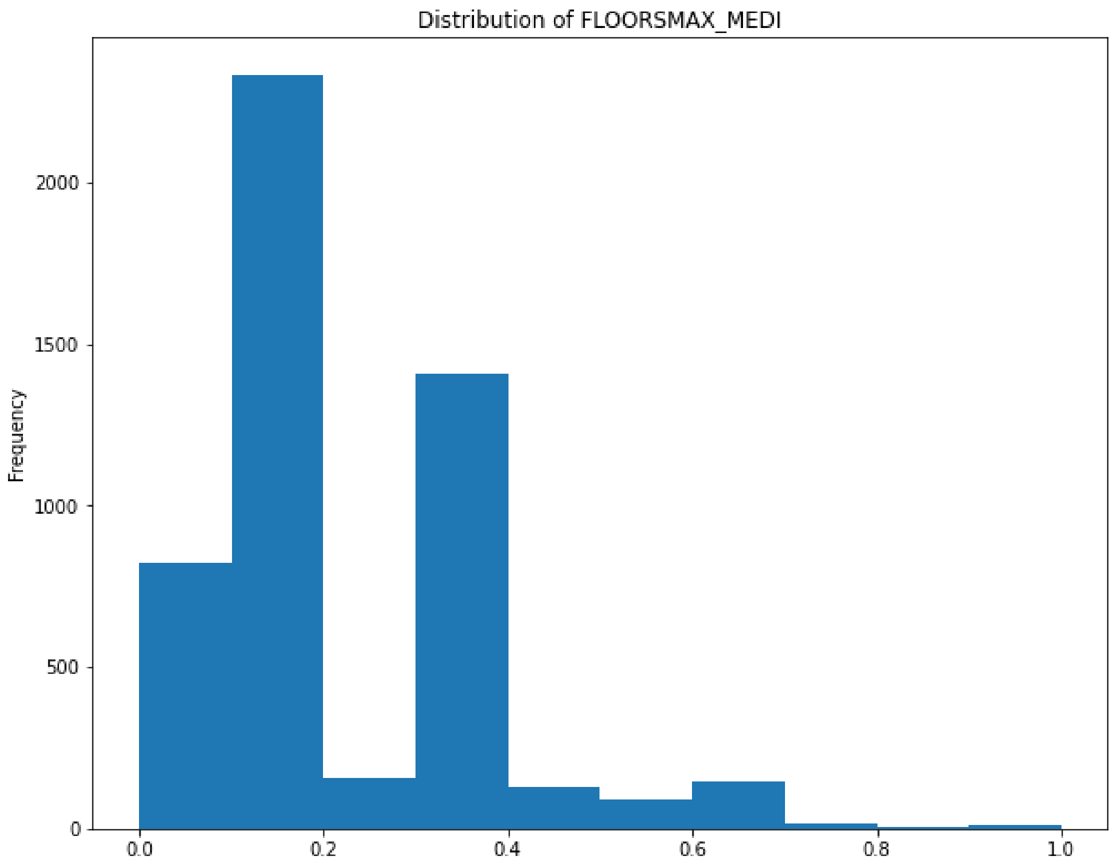


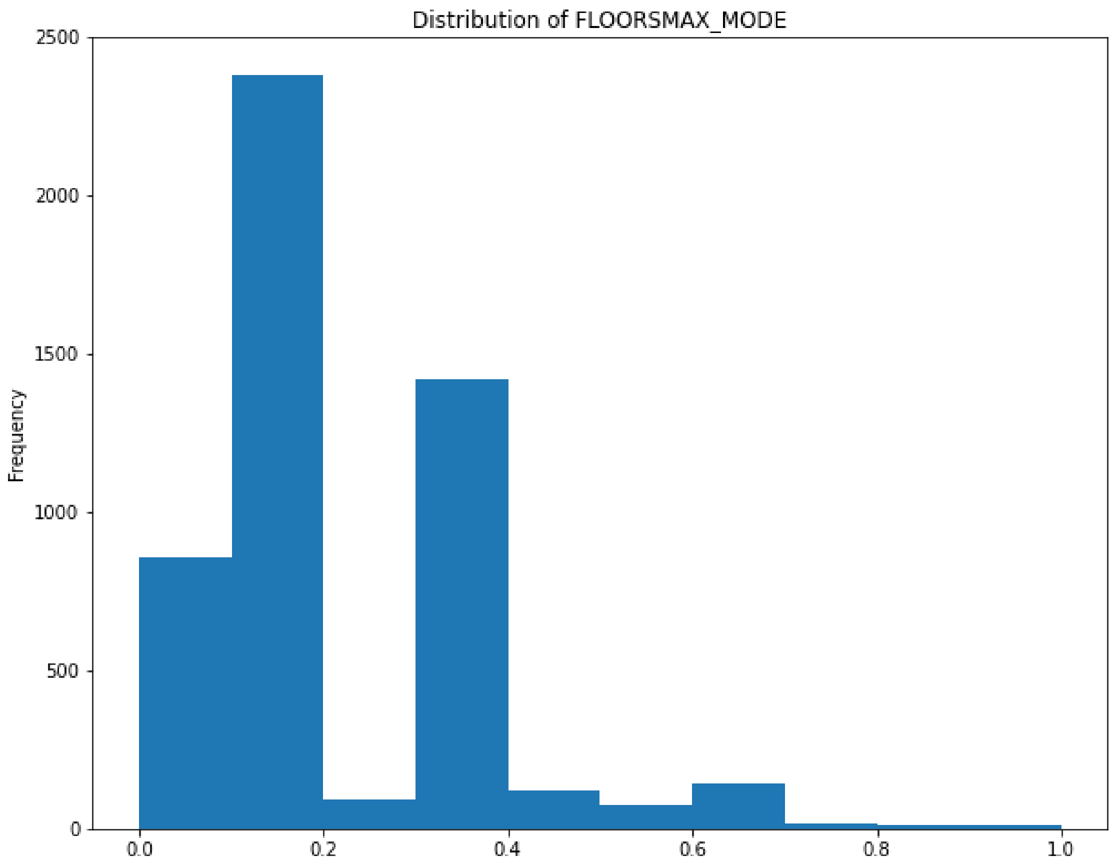

Distribution of EXT_SOURCE_2
Frequency
0
500
1000
1500
2000
0.0
0.2
0.4
0.6
0.8

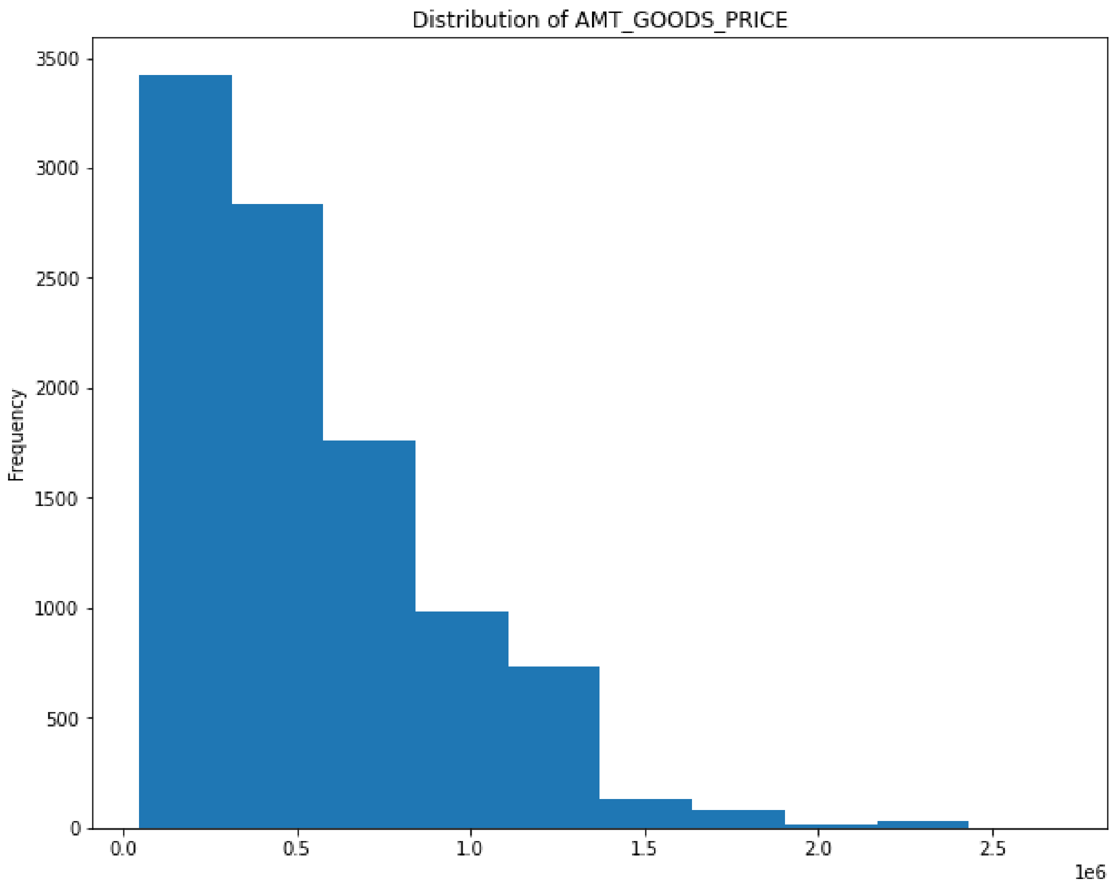

Distribution of EXT_SOURCE_1
Frequency
0
100
200
300
400
500
600
0.0
0.2
0.4
0.6
0.8

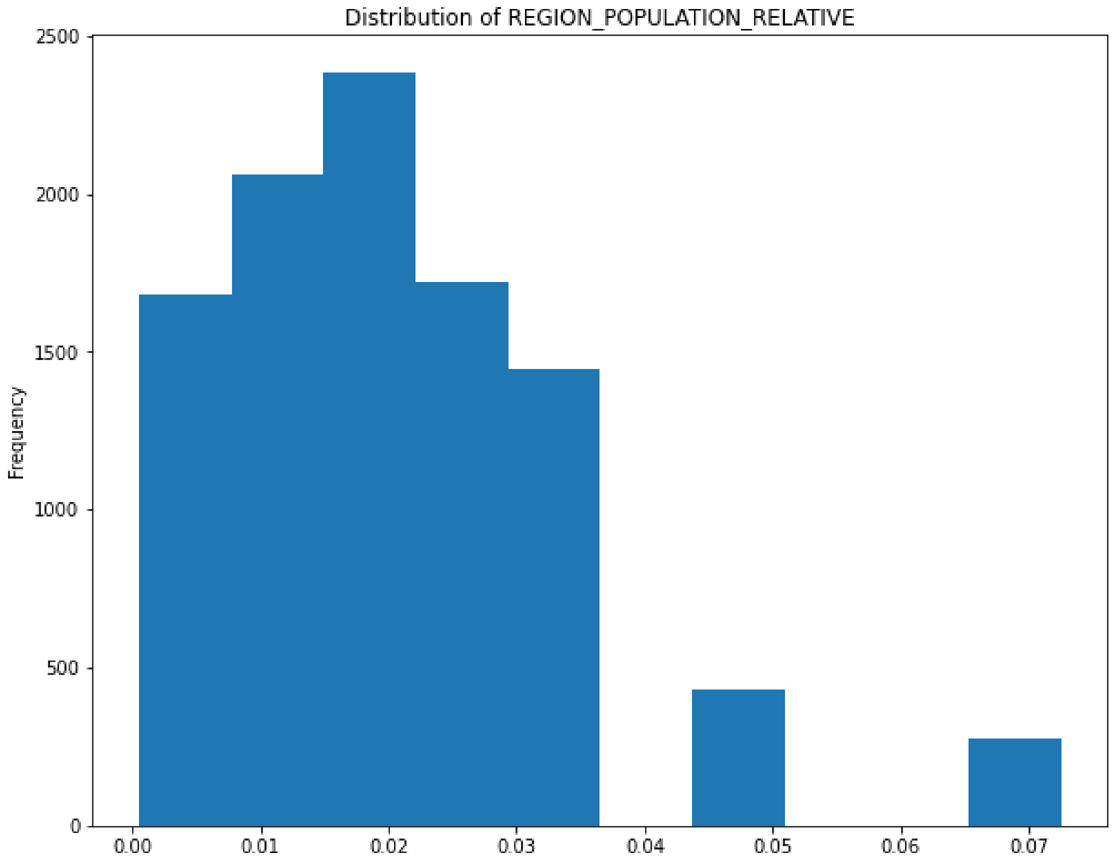

Distribution of DAYS_EMPLOYED
Frequency
0
1000
2000
3000
4000
5000
6000
7000
8000
0
50000
100000
150000
200000
250000
300000
350000

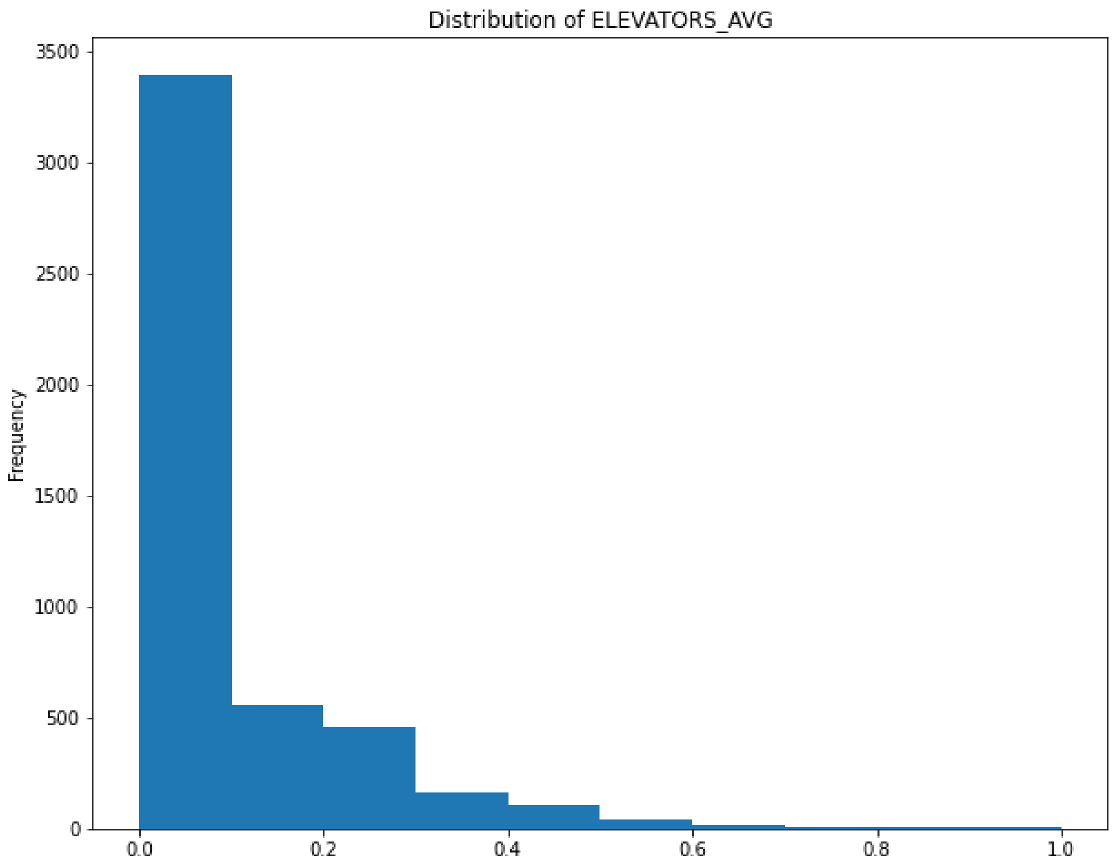

Distribution of FLOORSMAX_AVG
Frequency
0
500
1000
1500
2000
0.0
0.2
0.4
0.6
0.8
1.0


Distribution of FLOORSMAX_MEDI
Frequency
0
500
1000
1500
2000
0.0
0.2
0.4
0.6
0.8
1.0


Distribution of FLOORSMAX_MODE
Frequency
2500
2000
1500
1000
500
0
0.0
0.2
0.4
0.6
0.8
1.0


Distribution of AMT_GOODS_PRICE
Frequency
3500
3000
2500
2000
1500
1000
500
0
0.0
0.5
1.0
1.5
2.0
2.5
1e6

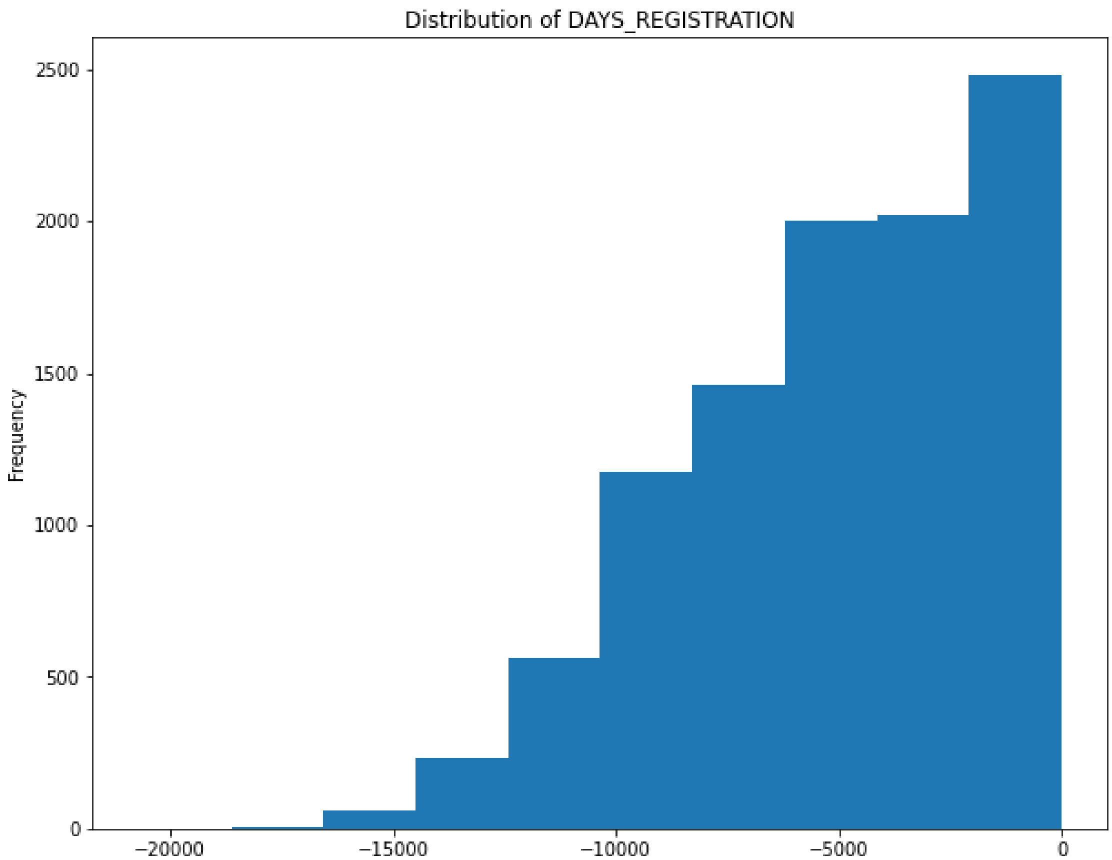

Distribution of REGION_POPULATION_RELATIVE
Frequency
2500
2000
1500
1000
500
0
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07

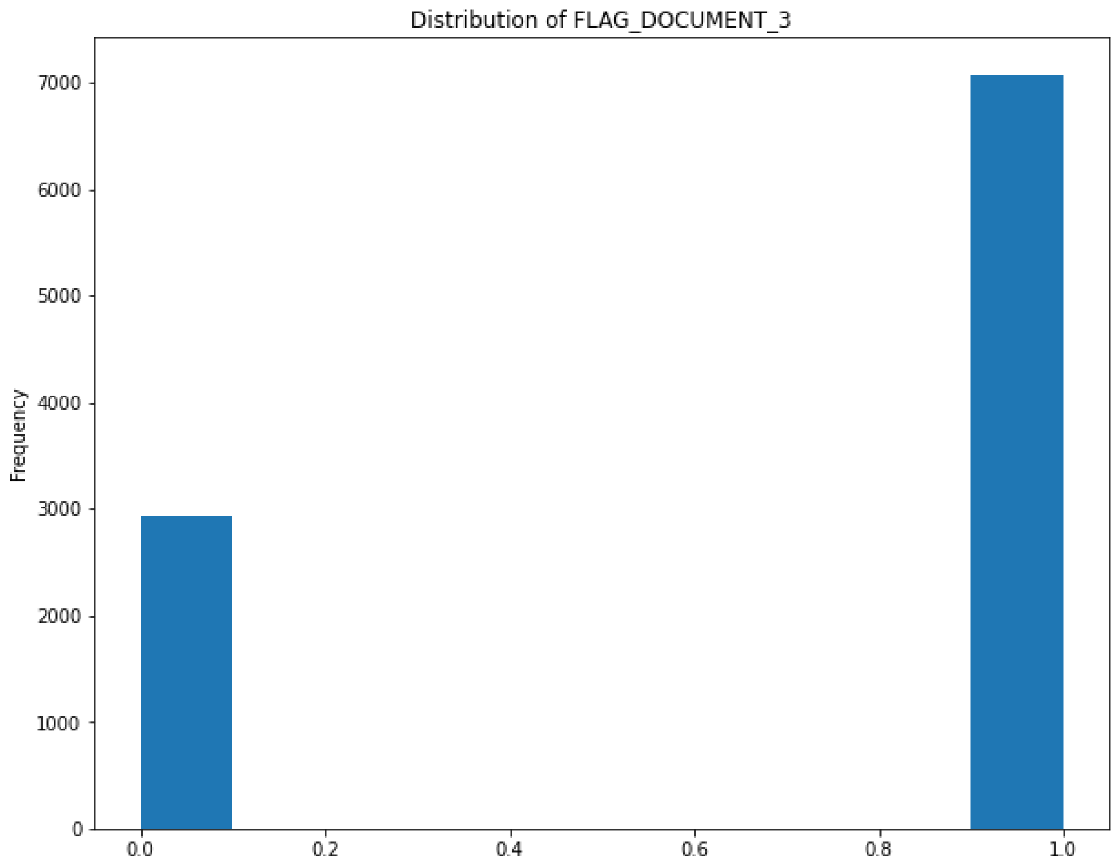


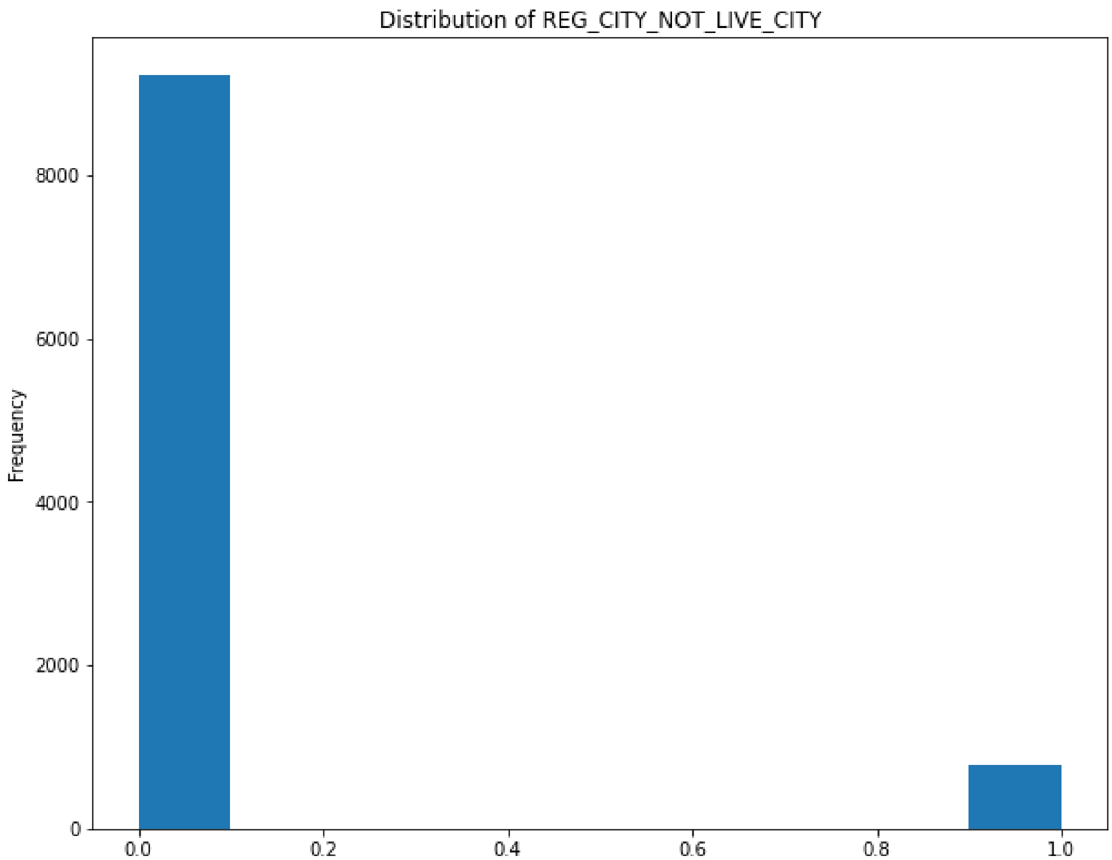
[65]:
```
pos_cols
```

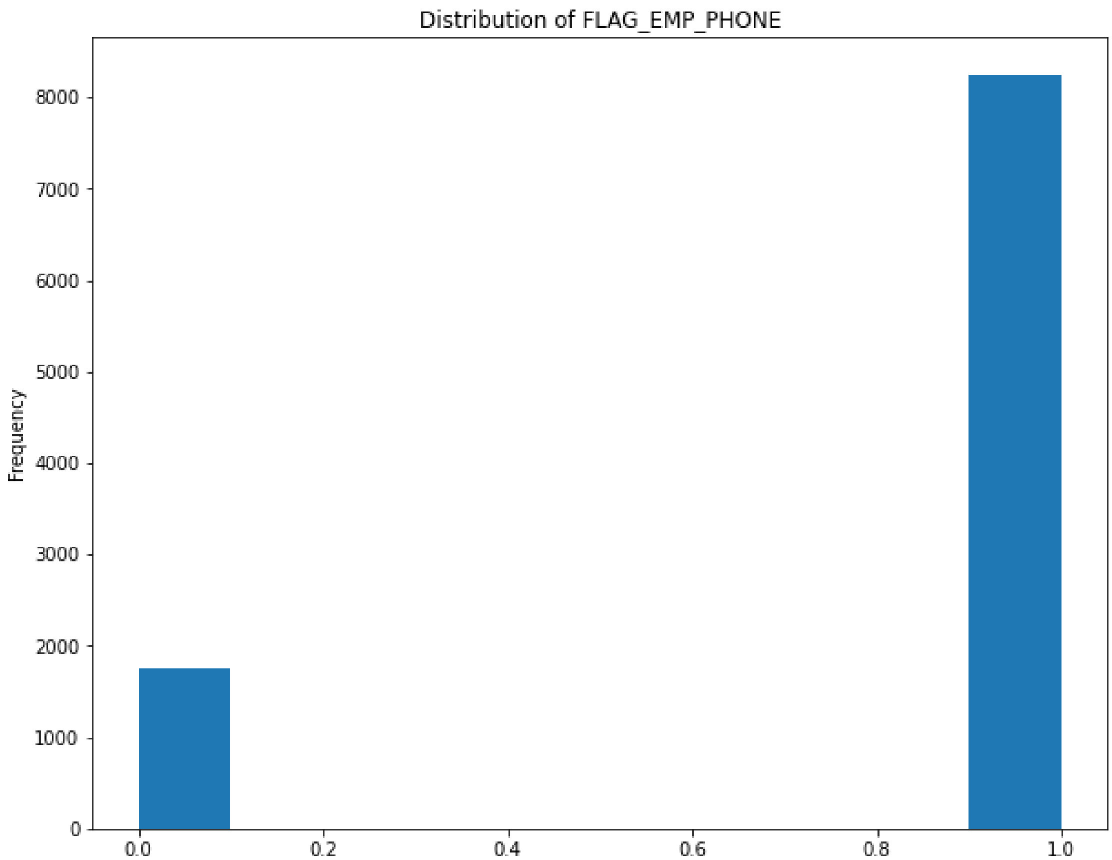
[65]:
```
['DAYS_REGISTRATION',
 'FLAG_DOCUMENT_3',
 'REG_CITY_NOT_LIVE_CITY',
 'FLAG_EMP_PHONE',
 'REG_CITY_NOT_WORK_CITY',
 'DAYS_ID_PUBLISH',
 'DAYS_LAST_PHONE_CHANGE',
 'REGION_RATING_CLIENT',
 'REGION_RATING_CLIENT_W_CITY',
 'DAYS_BIRTH',
 'TARGET']
```

[66]:
```
# EXPLORE DISTRIBUTION FOR MOST NEGATIVELY CORRELATED FEATURES

for col in pos_cols[:-1]:
    plt.figure(figsize=(10,8))
    plt.title(f"Distribution of {col}")
    train_data[col].dropna().plot.hist()
```

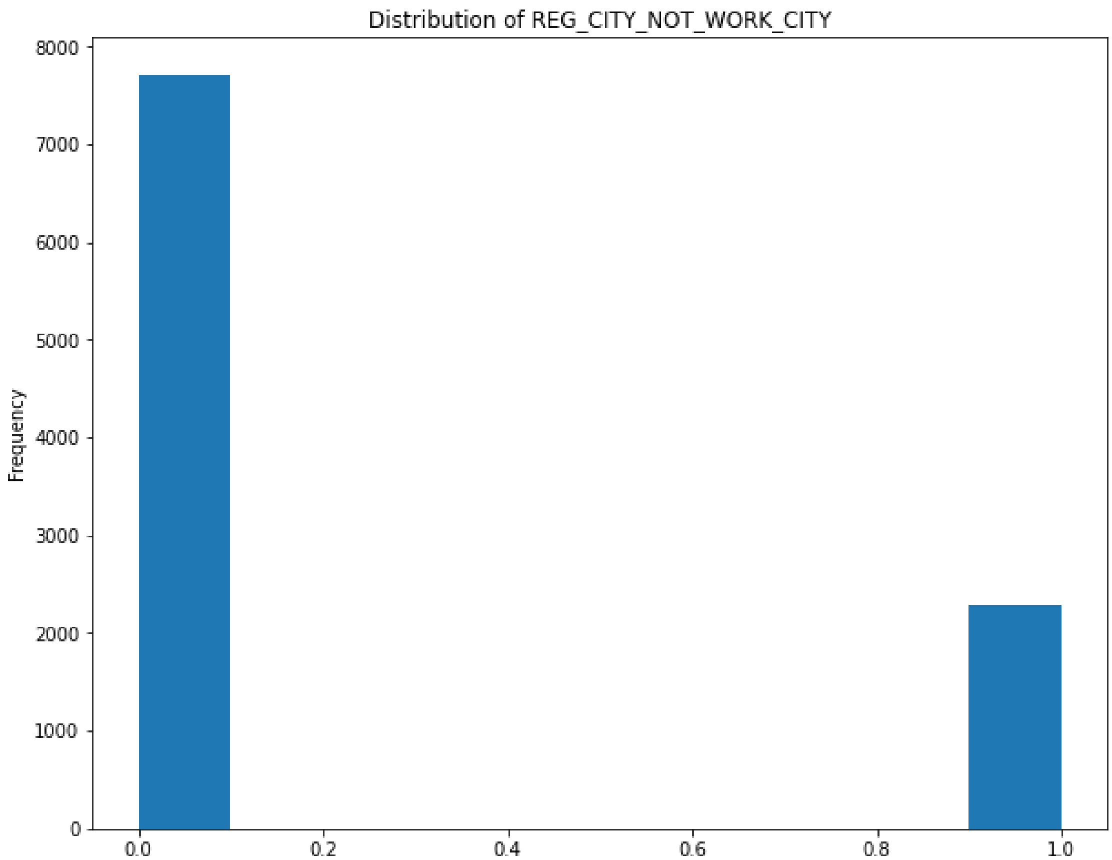

Distribution of DAYS_REGISTRATION
Frequency
2500
2000
1500
1000
500
0
−20000
−15000
−10000
−5000
0

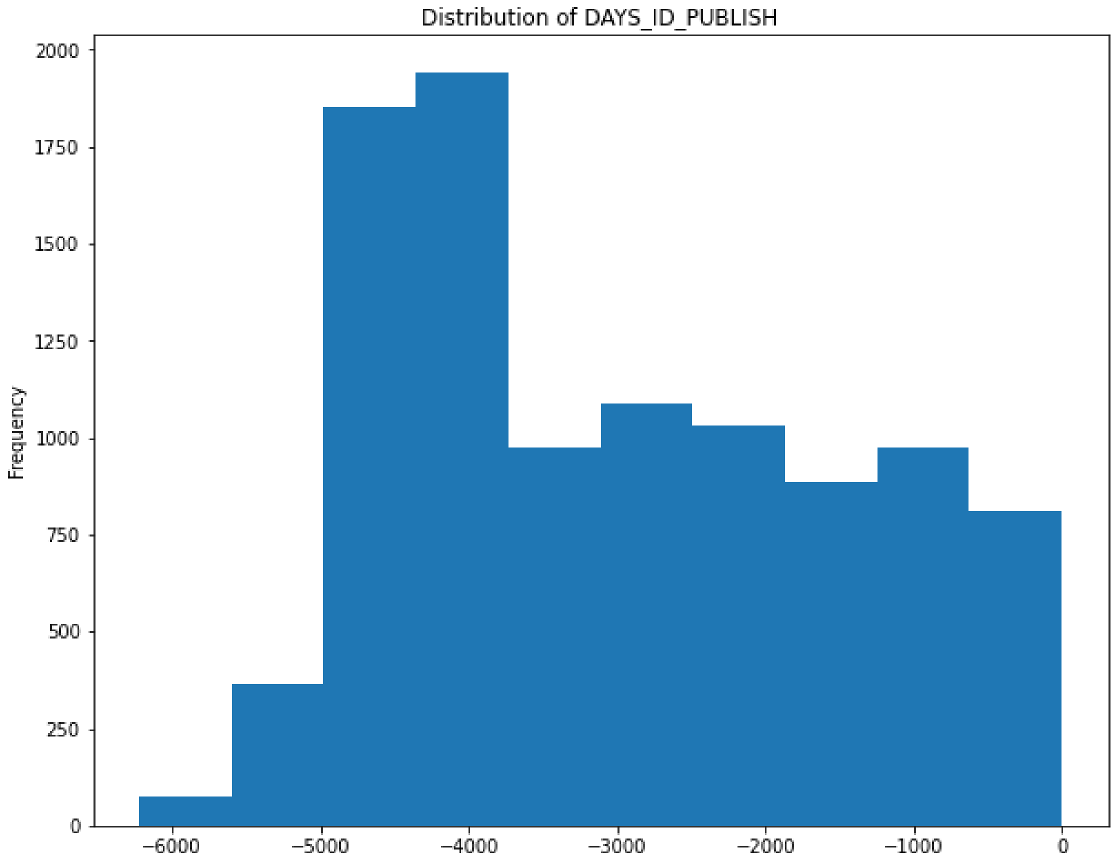

Distribution of FLAG_DOCUMENT_3
Frequency
7000
6000
5000
4000
3000
2000
1000
0
0.0
0.2
0.4
0.6
0.8
1.0

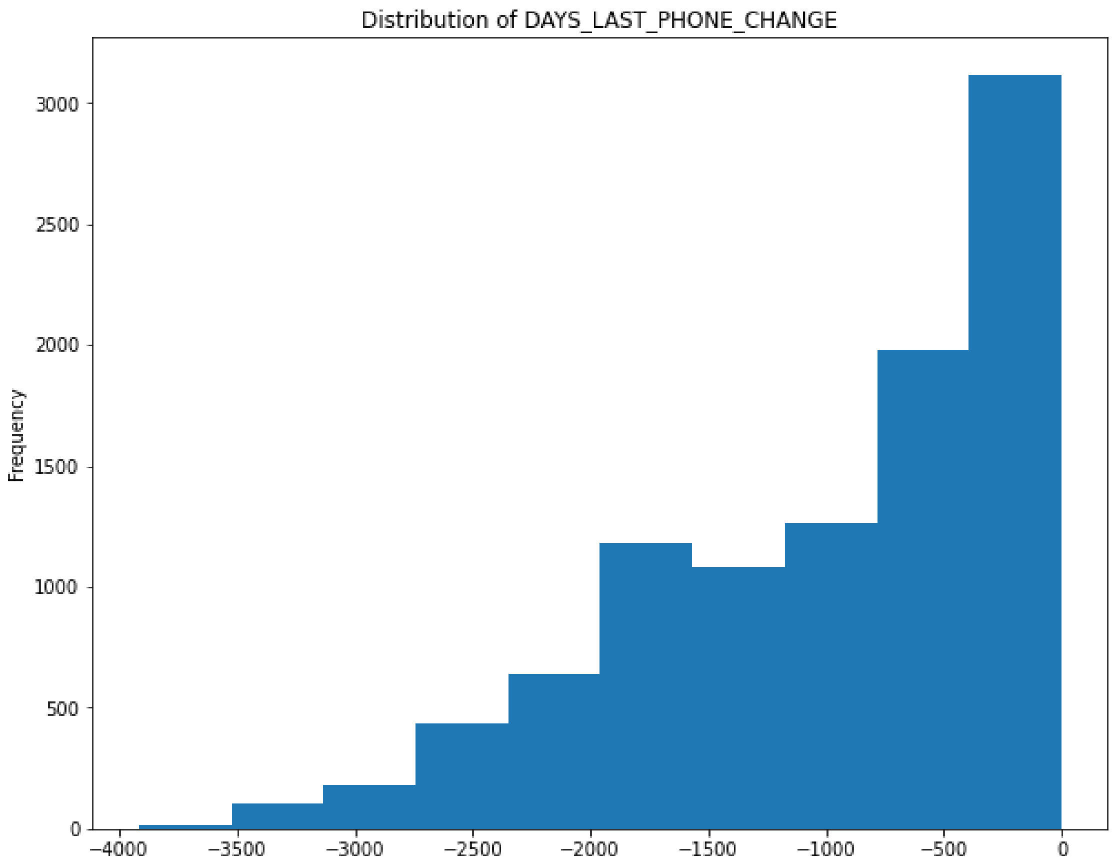

Distribution of REG_CITY_NOT_LIVE_CITY
Frequency
0
2000
4000
6000
8000
0.0
0.2
0.4
0.6
0.8
1.0



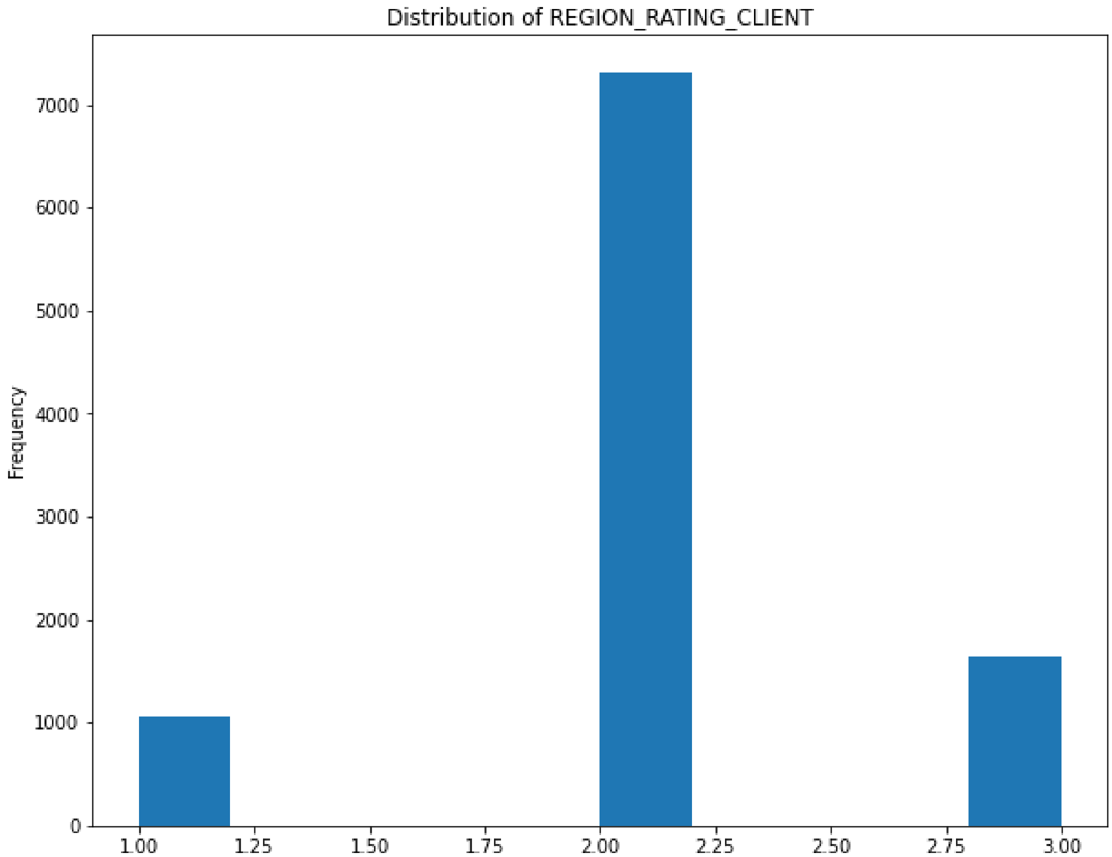
Distribution of FLAG_EMP_PHONE
Frequency
0
1000
2000
3000
4000
5000
6000
7000
8000
0.0
0.2
0.4
0.6
0.8
1.0

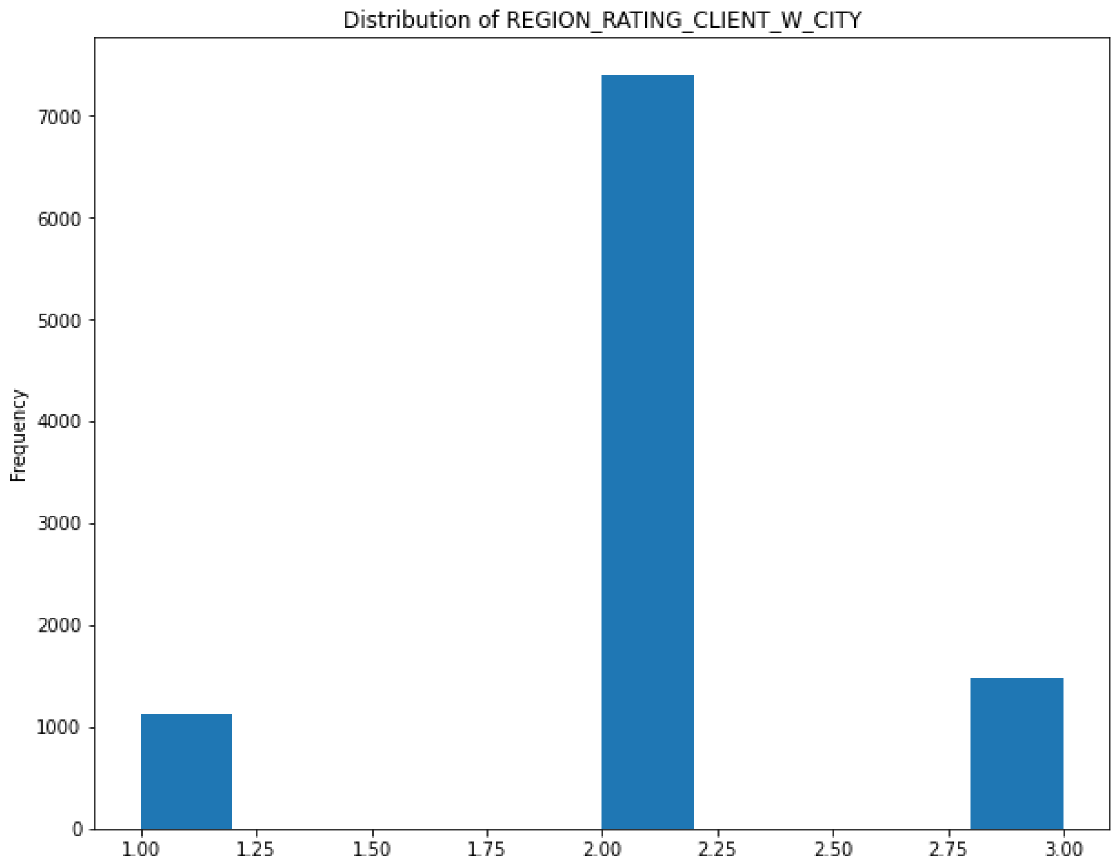

Distribution of REG_CITY_NOT_WORK_CITY
Frequency
8000
7000
6000
5000
4000
3000
2000
1000
0
0.0
0.2
0.4
0.6
0.8
1.0

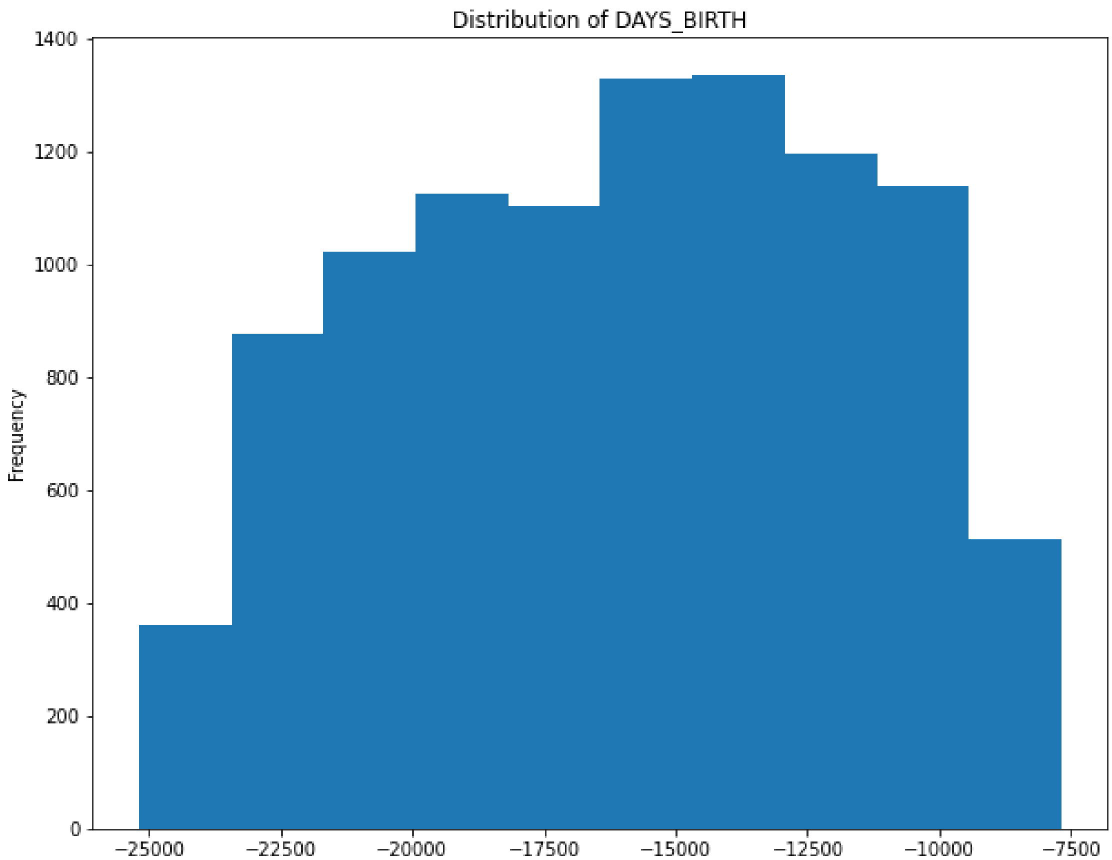

Distribution of DAYS_ID_PUBLISH
Frequency
2000
1750
1500
1250
1000
750
500
250
0
−6000
−5000
−4000
−3000
−2000
−1000
0



Distribution of DAYS_LAST_PHONE_CHANGE
Frequency
3000
2500
2000
1500
1000
500
0
−4000
−3500
−3000
−2500
−2000
−1500
−1000
−500
0


Distribution of REGION_RATING_CLIENT
Frequency
0
1000
2000
3000
4000
5000
6000
7000
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00


Distribution of REGION_RATING_CLIENT_W_CITY
Frequency
0
1000
2000
3000
4000
5000
6000
7000
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00



[ ]:

```

```

[ ]:

```

```

[ ]:

```

```

2 Train, Test, Split data from `application_train.csv`

Initial attempts were crashing kernels because the dataset was too large (over 300,000 records)

We are taking a random sample of 10,000 rows to allow our models to function. Future models may expand the size of the model training set to improve model performance.

[30]:

```
# Take random sample of overall training data
train_data = train_data.sample(10_000)
```

Create training, validation and testing data.

[31]:

```
X = train_data.drop('TARGET', axis=1)
y = train_data['TARGET']
```

```
# Split the provided training data into training and validationa and test
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.15,␣
↪random_state=42)
X_train, X_valid, y_train, y_valid = train_test_split(X_train, y_train,␣
↪test_size=0.2, random_state=42)
print(f"X train           shape: {X_train.shape}")
print(f"X validation      shape: {X_valid.shape}")
print(f"X test            shape: {X_test.shape}")


X_train.head()
```

```
X train           shape: (6800, 121)
X validation      shape: (1700, 121)
X test            shape: (1500, 121)
```

```
[31]:         SK_ID_CURR NAME_CONTRACT_TYPE CODE_GENDER FLAG_OWN_CAR  \
215244      349412         Cash loans           F            Y
137066      258971         Cash loans           M            N
126411      246600    Revolving loans           M            Y
123541      243271         Cash loans           F            N
165451      291801         Cash loans           F            N

       FLAG_OWN_REALTY  CNT_CHILDREN  AMT_INCOME_TOTAL  AMT_CREDIT  \
215244               Y             0          117000.0    592560.0
137066               Y             0          112500.0    254700.0
126411               Y             0          153000.0    180000.0
123541               N             0           54000.0    261153.0
165451               Y             1          112500.0   1255680.0

        AMT_ANNUITY  AMT_GOODS_PRICE NAME_TYPE_SUITE      NAME_INCOME_TYPE  \
215244      35806.5         450000.0   Unaccompanied               Working
137066      16276.5         225000.0   Unaccompanied               Working
126411       9000.0         180000.0   Unaccompanied  Commercial associate
123541      22414.5         225000.0   Unaccompanied               Working
165451      41629.5        1125000.0   Unaccompanied               Working

                  NAME_EDUCATION_TYPE NAME_FAMILY_STATUS  NAME_HOUSING_TYPE  \
215244  Secondary / secondary special            Married  House / apartment
137066  Secondary / secondary special            Married  House / apartment
126411               Higher education            Married  House / apartment
123541               Higher education            Married  House / apartment
165451  Secondary / secondary special            Married  House / apartment

        REGION_POPULATION_RELATIVE  DAYS_BIRTH  DAYS_EMPLOYED  \
215244                    0.018029      -13483           -547
```

```
137066                   0.018634      -16345          -4175
126411                   0.022800      -13813           -254
123541                   0.019101      -10330           -483
165451                   0.008019      -18336           -261

        DAYS_REGISTRATION  DAYS_ID_PUBLISH  OWN_CAR_AGE  FLAG_MOBIL  \
215244            -7537.0            -4677          0.0           1
137066            -6490.0            -4172          NaN           1
126411            -4745.0            -4250          7.0           1
123541            -3008.0             -153          NaN           1
165451            -1533.0            -1883          NaN           1

        FLAG_EMP_PHONE  FLAG_WORK_PHONE  FLAG_CONT_MOBILE  FLAG_PHONE  \
215244               1                1                 1           1
137066               1                1                 1           1
126411               1                0                 1           0
123541               1                0                 1           1
165451               1                0                 1           0

        FLAG_EMAIL OCCUPATION_TYPE  CNT_FAM_MEMBERS  REGION_RATING_CLIENT  \
215244           0     Sales staff              2.0                     3
137066           0        Laborers              2.0                     2
126411           0         Drivers              2.0                     2
123541           1      Core staff              2.0                     2
165451           0     Sales staff              3.0                     2

        REGION_RATING_CLIENT_W_CITY WEEKDAY_APPR_PROCESS_START  \
215244                            3                    TUESDAY
137066                            2                   SATURDAY
126411                            2                   THURSDAY
123541                            2                  WEDNESDAY
165451                            2                     MONDAY

        HOUR_APPR_PROCESS_START  REG_REGION_NOT_LIVE_REGION  \
215244                       12                           0
137066                       14                           0
126411                       13                           0
123541                       12                           0
165451                       15                           0

        REG_REGION_NOT_WORK_REGION  LIVE_REGION_NOT_WORK_REGION  \
215244                           0                            0
137066                           0                            0
126411                           0                            0
123541                           0                            0
165451                           0                            0
```

```
        REG_CITY_NOT_LIVE_CITY  REG_CITY_NOT_WORK_CITY  \
215244                       0                       0
137066                       0                       0
126411                       1                       1
123541                       0                       0
165451                       0                       0

        LIVE_CITY_NOT_WORK_CITY       ORGANIZATION_TYPE  EXT_SOURCE_1  \
215244                        0  Business Entity Type 3           NaN
137066                        0  Business Entity Type 3      0.177842
126411                        0  Business Entity Type 3      0.297692
123541                        0              Government      0.314317
165451                        0            Trade: type 3      0.781515

        EXT_SOURCE_2  EXT_SOURCE_3  APARTMENTS_AVG  BASEMENTAREA_AVG  \
215244      0.530192      0.456110             NaN               NaN
137066      0.559699      0.644679          0.2330            0.0952
126411      0.312431      0.190706             NaN               NaN
123541      0.765318      0.593718             NaN               NaN
165451      0.588985      0.848244          0.1536            0.1689

        YEARS_BEGINEXPLUATATION_AVG  YEARS_BUILD_AVG  COMMONAREA_AVG  \
215244                          NaN              NaN             NaN
137066                       0.9801              NaN             NaN
126411                          NaN              NaN             NaN
123541                          NaN              NaN             NaN
165451                       0.9786           0.7076          0.2896

        ELEVATORS_AVG  ENTRANCES_AVG  FLOORSMAX_AVG  FLOORSMIN_AVG  \
215244            NaN            NaN            NaN            NaN
137066           0.08         0.0690         0.3333            NaN
126411            NaN            NaN            NaN            NaN
123541            NaN            NaN            NaN            NaN
165451           0.00         0.3448         0.1667         0.0417

        LANDAREA_AVG  LIVINGAPARTMENTS_AVG  LIVINGAREA_AVG  \
215244           NaN                   NaN             NaN
137066        0.0790                   NaN          0.1046
126411           NaN                   NaN             NaN
123541           NaN                   NaN             NaN
165451        0.0821                0.1252          0.1484

        NONLIVINGAPARTMENTS_AVG  NONLIVINGAREA_AVG  APARTMENTS_MODE  \
215244                      NaN                NaN              NaN
137066                      NaN              0.009           0.2374
126411                      NaN                NaN              NaN
123541                      NaN                NaN              NaN
```

```
165451                        0.0              0.000            0.1565

        BASEMENTAREA_MODE  YEARS_BEGINEXPLUATATION_MODE  YEARS_BUILD_MODE  \
215244                NaN                           NaN               NaN
137066             0.0988                        0.9801               NaN
126411                NaN                           NaN               NaN
123541                NaN                           NaN               NaN
165451             0.1753                        0.9786             0.719

        COMMONAREA_MODE  ELEVATORS_MODE  ENTRANCES_MODE  FLOORSMAX_MODE  \
215244              NaN             NaN             NaN             NaN
137066              NaN          0.0806          0.0690          0.3333
126411              NaN             NaN             NaN             NaN
123541              NaN             NaN             NaN             NaN
165451           0.2923          0.0000          0.3448          0.1667

        FLOORSMIN_MODE  LANDAREA_MODE  LIVINGAPARTMENTS_MODE  LIVINGAREA_MODE  \
215244             NaN            NaN                    NaN              NaN
137066             NaN         0.0808                    NaN           0.1090
126411             NaN            NaN                    NaN              NaN
123541             NaN            NaN                    NaN              NaN
165451          0.0417         0.0839                 0.1368           0.1546

        NONLIVINGAPARTMENTS_MODE  NONLIVINGAREA_MODE  APARTMENTS_MEDI  \
215244                       NaN                 NaN              NaN
137066                       NaN              0.0095           0.2352
126411                       NaN                 NaN              NaN
123541                       NaN                 NaN              NaN
165451                       0.0              0.0000           0.1551

        BASEMENTAREA_MEDI  YEARS_BEGINEXPLUATATION_MEDI  YEARS_BUILD_MEDI  \
215244                NaN                           NaN               NaN
137066             0.0952                        0.9801               NaN
126411                NaN                           NaN               NaN
123541                NaN                           NaN               NaN
165451             0.1689                        0.9786            0.7115

        COMMONAREA_MEDI  ELEVATORS_MEDI  ENTRANCES_MEDI  FLOORSMAX_MEDI  \
215244              NaN             NaN             NaN             NaN
137066              NaN            0.08          0.0690          0.3333
126411              NaN             NaN             NaN             NaN
123541              NaN             NaN             NaN             NaN
165451           0.2915            0.00          0.3448          0.1667

        FLOORSMIN_MEDI  LANDAREA_MEDI  LIVINGAPARTMENTS_MEDI  LIVINGAREA_MEDI  \
215244             NaN            NaN                    NaN              NaN
137066             NaN         0.0804                    NaN           0.1065
```

```
126411           NaN           NaN                      NaN             NaN
123541           NaN           NaN                      NaN             NaN
165451        0.0417        0.0835                   0.1274          0.1510

        NONLIVINGAPARTMENTS_MEDI  NONLIVINGAREA_MEDI FONDKAPREMONT_MODE  \
215244                       NaN                 NaN                NaN
137066                       NaN              0.0092                NaN
126411                       NaN                 NaN                NaN
123541                       NaN                 NaN                NaN
165451                       0.0              0.0000   reg oper account

        HOUSETYPE_MODE  TOTALAREA_MODE WALLSMATERIAL_MODE EMERGENCYSTATE_MODE  \
215244             NaN             NaN                NaN                 NaN
137066  block of flats          0.1322       Stone, brick                  No
126411             NaN             NaN                NaN                 NaN
123541             NaN             NaN                NaN                 NaN
165451  block of flats          0.1583              Panel                  No

        OBS_30_CNT_SOCIAL_CIRCLE  DEF_30_CNT_SOCIAL_CIRCLE  \
215244                       0.0                       0.0
137066                       0.0                       0.0
126411                       0.0                       0.0
123541                       1.0                       0.0
165451                       1.0                       0.0

        OBS_60_CNT_SOCIAL_CIRCLE  DEF_60_CNT_SOCIAL_CIRCLE  \
215244                       0.0                       0.0
137066                       0.0                       0.0
126411                       0.0                       0.0
123541                       1.0                       0.0
165451                       1.0                       0.0

        DAYS_LAST_PHONE_CHANGE  FLAG_DOCUMENT_2  FLAG_DOCUMENT_3  \
215244                  -405.0                0                1
137066                  -422.0                0                1
126411                  -630.0                0                0
123541                 -1030.0                0                1
165451                 -2244.0                0                1

        FLAG_DOCUMENT_4  FLAG_DOCUMENT_5  FLAG_DOCUMENT_6  FLAG_DOCUMENT_7  \
215244                0                0                0                0
137066                0                0                0                0
126411                0                0                0                0
123541                0                0                0                0
165451                0                0                0                0

        FLAG_DOCUMENT_8  FLAG_DOCUMENT_9  FLAG_DOCUMENT_10  FLAG_DOCUMENT_11  \
```

215244	0	0	0	0
137066	0	0	0	0
126411	0	0	0	0
123541	0	0	0	0
165451	0	0	0	0

	FLAG_DOCUMENT_12	FLAG_DOCUMENT_13	FLAG_DOCUMENT_14	\
215244	0	0	0	
137066	0	0	0	
126411	0	0	0	
123541	0	0	0	
165451	0	0	0	

	FLAG_DOCUMENT_15	FLAG_DOCUMENT_16	FLAG_DOCUMENT_17	\
215244	0	0	0	
137066	0	0	0	
126411	0	0	0	
123541	0	0	0	
165451	0	0	0	

	FLAG_DOCUMENT_18	FLAG_DOCUMENT_19	FLAG_DOCUMENT_20	\
215244	0	0	0	
137066	0	0	0	
126411	0	0	0	
123541	0	0	0	
165451	0	0	0	

	FLAG_DOCUMENT_21	AMT_REQ_CREDIT_BUREAU_HOUR	\
215244	0	0.0	
137066	0	0.0	
126411	0	0.0	
123541	0	0.0	
165451	0	0.0	

	AMT_REQ_CREDIT_BUREAU_DAY	AMT_REQ_CREDIT_BUREAU_WEEK	\
215244	0.0	0.0	
137066	0.0	0.0	
126411	0.0	0.0	
123541	0.0	0.0	
165451	0.0	0.0	

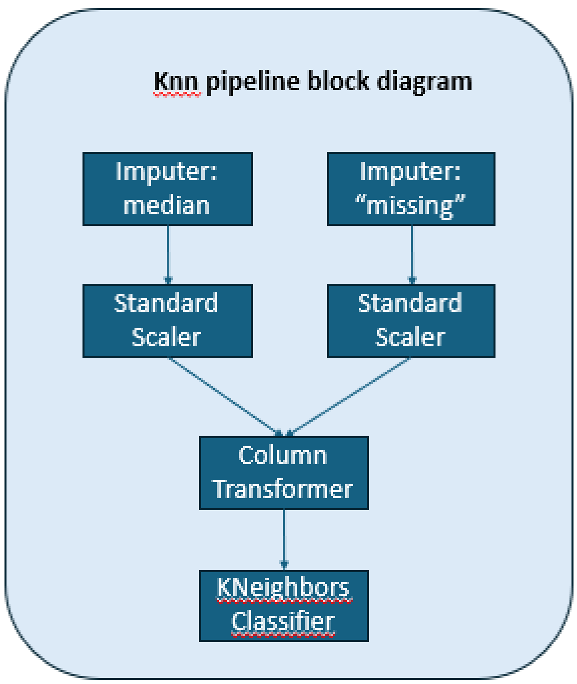
	AMT_REQ_CREDIT_BUREAU_MON	AMT_REQ_CREDIT_BUREAU_QRT	\
215244	0.0	0.0	
137066	0.0	0.0	
126411	0.0	0.0	
123541	0.0	1.0	
165451	0.0	0.0	

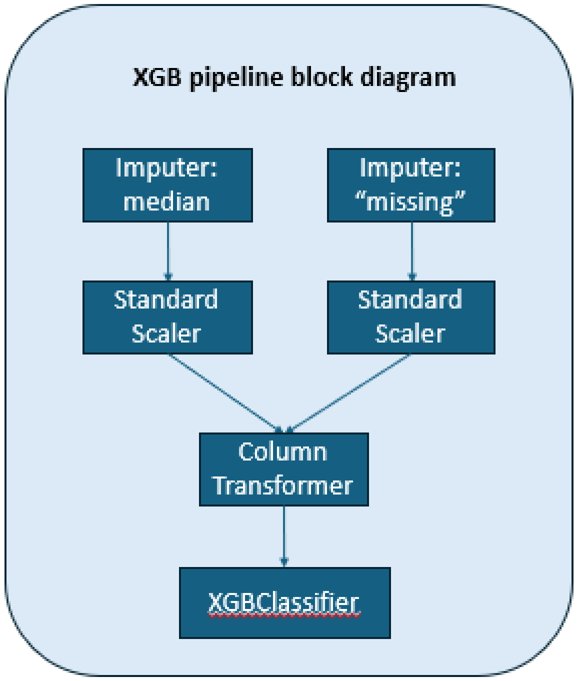
```
        AMT_REQ_CREDIT_BUREAU_YEAR
215244                         0.0
137066                         5.0
126411                         2.0
123541                         1.0
165451                         1.0
```

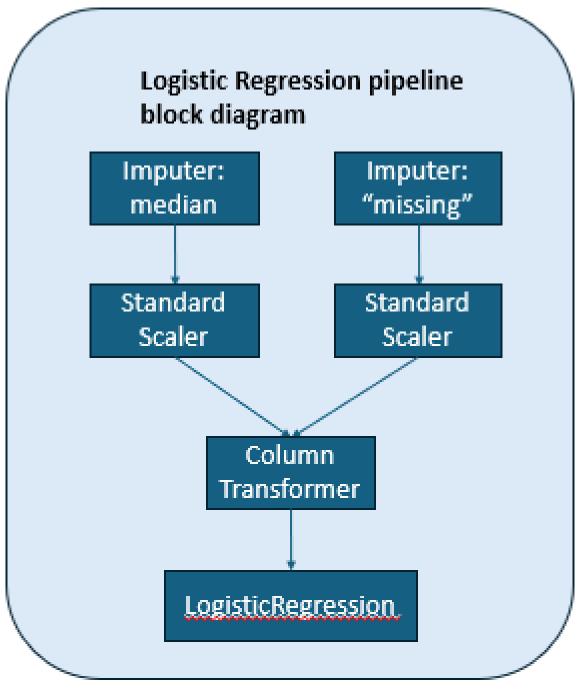
3 Pipelines

3.1 KNN Classification

We have included block diagrams detailing each pipeline.







[32]:
```
numeric_features = ['CNT_CHILDREN','AMT_INCOME_TOTAL', 'AMT_CREDIT',␣
↪'AMT_ANNUITY', 'AMT_GOODS_PRICE', 'DAYS_BIRTH', 'DAYS_EMPLOYED']
numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())])
```

```
categorical_features = ['NAME_CONTRACT_TYPE', 'CODE_GENDER', 'FLAG_OWN_CAR',␣
↪'FLAG_OWN_REALTY','OCCUPATION_TYPE']
categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='constant', fill_value='missing')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))])

preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_features),
        ('cat', categorical_transformer, categorical_features)])

knn_model = Pipeline(steps=[('preprocessor', preprocessor),
                      ('classifier', KNeighborsClassifier())])


knn_model.fit(X_train, y_train)
print(knn_model.score(X_test, y_test))
```

```
0.9133333333333333
```

Create a log to compare the different models' perfomance.

Code adapted from `HW04-LinRegrBoston_Bike-Demand.ipynb`

[41]:
```
try:
    del expLog
except:
    pass

exp_name = "knn_baseline"
try:
    expLog
except NameError:
    expLog = DataFrame(columns=["exp_name",
                                  "Train Acc",
                                  "Valid Acc",
                                  "Test  Acc",
                                  "Train F1",
                                  "Valid F1",
                                  "Test  F1"
                                 ])

expLog.loc[len(expLog)] = [f"{exp_name}"] + list(np.round(
               [accuracy_score(y_train, knn_model.predict(X_train)),
                accuracy_score(y_valid, knn_model.predict(X_valid)),
                accuracy_score(y_test, knn_model.predict(X_test)),
                f1_score(y_train, knn_model.predict(X_train)),
                f1_score(y_valid, knn_model.predict(X_valid)),
```

```
                f1_score(y_test, knn_model.predict(X_test))],
    4))
expLog
```

[41]:

	exp_name	Train Acc	Valid Acc	Test Acc	Train F1	Valid F1	Test F1
0	knn_baseline	0.9234	0.9282	0.9133	0.0876	0.0	0.058

3.2 XG Boost Pipeline

[42]:

```
numeric_features = ['CNT_CHILDREN','AMT_INCOME_TOTAL', 'AMT_CREDIT',␣
 ↪'AMT_ANNUITY', 'AMT_GOODS_PRICE', 'DAYS_BIRTH', 'DAYS_EMPLOYED']
numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())])

categorical_features = ['NAME_CONTRACT_TYPE', 'CODE_GENDER', 'FLAG_OWN_CAR',␣
 ↪'FLAG_OWN_REALTY','OCCUPATION_TYPE']
categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='constant', fill_value='missing')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))])

preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_features),
        ('cat', categorical_transformer, categorical_features)])

XGB_model = Pipeline(steps=[('preprocessor', preprocessor),
                      ('classifier', XGBClassifier())])


XGB_model.fit(X_train, y_train)
```

```
/usr/local/lib/python3.9/site-packages/xgboost/sklearn.py:1224: UserWarning: The
use of label encoder in XGBClassifier is deprecated and will be removed in a
future release. To remove this warning, do the following: 1) Pass option
use_label_encoder=False when constructing XGBClassifier object; and 2) Encode
your labels (y) as integers starting with 0, i.e. 0, 1, 2, …, [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
```

```
[02:18:51] WARNING: ../src/learner.cc:1115: Starting in XGBoost 1.3.0, the
default evaluation metric used with the objective 'binary:logistic' was changed
from 'error' to 'logloss'. Explicitly set eval_metric if you'd like to restore
the old behavior.
```

[42]:

```
Pipeline(steps=[('preprocessor',
                 ColumnTransformer(transformers=[('num',
                                                  Pipeline(steps=[('imputer',
SimpleImputer(strategy='median')),
```

```
                                                               ('scaler',
StandardScaler())]),
                                                 ['CNT_CHILDREN',
                                                  'AMT_INCOME_TOTAL',
                                                  'AMT_CREDIT', 'AMT_ANNUITY',
                                                  'AMT_GOODS_PRICE',
                                                  'DAYS_BIRTH',
                                                  'DAYS_EMPLOYED']),
                                                ('cat',
                                                 Pipeline(steps=[('imputer',
SimpleImputer(fill_value='missing',
 strategy…
                              gamma=0, gpu_id=-1, importance_type=None,
                              interaction_constraints='',
                              learning_rate=0.300000012, max_delta_step=0,
                              max_depth=6, min_child_weight=1, missing=nan,
                              monotone_constraints='()', n_estimators=100,
                              n_jobs=12, num_parallel_tree=1, predictor='auto',
                              random_state=0, reg_alpha=0, reg_lambda=1,
                              scale_pos_weight=1, subsample=1,
                              tree_method='exact', validate_parameters=1,
                              verbosity=None))])
```

[43]:

```python
exp_name = "XGB_baseline"

expLog.loc[len(expLog)] = [f"{exp_name}"] + list(np.round(
               [accuracy_score(y_train, XGB_model.predict(X_train)),
                accuracy_score(y_valid, XGB_model.predict(X_valid)),
                accuracy_score(y_test, XGB_model.predict(X_test)),
                f1_score(y_train, XGB_model.predict(X_train)),
                f1_score(y_valid, XGB_model.predict(X_valid)),
                f1_score(y_test, XGB_model.predict(X_test))],
    4))

expLog
```

[43]:

	exp_name	Train Acc	Valid Acc	Test Acc	Train F1	Valid F1	Test F1
0	knn_baseline	0.9234	0.9282	0.9133	0.0876	0.0000	0.058
1	XGB_baseline	0.9638	0.9300	0.9127	0.7007	0.0165	0.015

3.3 Logistic Regression Pipeline

[44]:

```python
numeric_features = ['CNT_CHILDREN','AMT_INCOME_TOTAL', 'AMT_CREDIT',␣
↪'AMT_ANNUITY', 'AMT_GOODS_PRICE', 'DAYS_BIRTH', 'DAYS_EMPLOYED']
numeric_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='median')),
    ('scaler', StandardScaler())])
```

```
categorical_features = ['NAME_CONTRACT_TYPE', 'CODE_GENDER', 'FLAG_OWN_CAR',␣
↪'FLAG_OWN_REALTY','OCCUPATION_TYPE']
categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='constant', fill_value='missing')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))])

preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_features),
        ('cat', categorical_transformer, categorical_features)])

logreg_model = Pipeline(steps=[('preprocessor', preprocessor),
                      ('classifier', LogisticRegression(max_iter=10000))])


logreg_model.fit(X_train, y_train)
```

[44]:

```
Pipeline(steps=[('preprocessor',
                 ColumnTransformer(transformers=[('num',
                                                  Pipeline(steps=[('imputer',
SimpleImputer(strategy='median')),
                                                                  ('scaler',
StandardScaler())]),
                                                  ['CNT_CHILDREN',
                                                   'AMT_INCOME_TOTAL',
                                                   'AMT_CREDIT', 'AMT_ANNUITY',
                                                   'AMT_GOODS_PRICE',
                                                   'DAYS_BIRTH',
                                                   'DAYS_EMPLOYED']),
                                                 ('cat',
                                                  Pipeline(steps=[('imputer',
SimpleImputer(fill_value='missing',
 strategy='constant')),
                                                                  ('onehot',
OneHotEncoder(handle_unknown='ignore'))]),
                                                  ['NAME_CONTRACT_TYPE',
                                                   'CODE_GENDER',
                                                   'FLAG_OWN_CAR',
                                                   'FLAG_OWN_REALTY',
                                                   'OCCUPATION_TYPE'])])),
                ('classifier', LogisticRegression(max_iter=10000))])
```

[45]:

```
exp_name = "logreg_baseline"

expLog.loc[len(expLog)] = [f"{exp_name}"] + list(np.round(
               [accuracy_score(y_train, logreg_model.predict(X_train)),
```

```
                 accuracy_score(y_valid, logreg_model.predict(X_valid)),
                 accuracy_score(y_test, logreg_model.predict(X_test)),
                 f1_score(y_train, logreg_model.predict(X_train)),
                 f1_score(y_valid, logreg_model.predict(X_valid)),
                 f1_score(y_test, logreg_model.predict(X_test))],
    4))
expLog
```

[45]:

```
          exp_name  Train Acc  Valid Acc  Test  Acc  Train F1  Valid F1  \
0     knn_baseline     0.9234     0.9282     0.9133    0.0876    0.0000
1     XGB_baseline     0.9638     0.9300     0.9127    0.7007    0.0165
2  logreg_baseline     0.9216     0.9329     0.9160    0.0000    0.0000

   Test  F1
0     0.058
1     0.015
2     0.000
```

4 Results

All models were trained on the 20 most correlated features of the 120 available in the `application_train.csv`. Future work will add additional features, as well as implemented feature engineering to create new features.

The baseline models all performed relatively well in predicting an outcome of a customer's default risk in both the validation and test sets in terms of accuracy. However the F1 scores are extremely low for all models. Since the F1 score is a combination of precision and recall, this implies one of those two metrics may be zero.

Precision is defined as

$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Recall is defined as

$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

The target is imbalanced, since 92.2% of the customers did not default. It is possible there are 0 True Positives in the validation and test set.

[50]:

```
train_data['TARGET'].value_counts(normalize=True)
```

[50]:

```
0    0.9227
1    0.0773
Name: TARGET, dtype: float64
```

[56]:
```
print (f'The target class is 1 in the validation data: {round(y_valid.sum() /␣
 ↪len(y_valid), 4)}')

print (f'The target class is 1 in the test data: {round(y_test.sum() /␣
 ↪len(y_test), 4)}')
```

```
The target class is 1 in the validation data: 0.0671
The target class is 1 in the test data: 0.084
```

Of the three trained and fitted models, the XGBoost Classifier performed the best. It was the only model to have an F1 score greater than zero, implying that it did classify correctly at least test record as a true positive.

5 Conclusions

The goal of this project is to use the data as provided by Kaggle and predict whether or not a client will repay a loan. The target column contains a 1 if the client does not pay their loan and is a credit risk.

This phase of the project dealt mainly with creating baseline pipelines and models. These model metrics will help guide our future efforts as we perform feature engineering, add and remove features, and use GridSearch to tune model hyperparameters.

Our hypothesis for now is that by using the top twenty features we are able to build an accurate model. However since the data is imbalanced, we may need to use revisit which features to use in the model, or dive deeper into the other data files Kaggle provides.

Our key steps: - Download and load the data from Kaggle. - Perform exploratory data analysis (EDA). - Define several baseline classification models using KNN, XG Boost and Logistic Regression. - Create pipelines to standardize any numerical feature and one hot encode categorial features. - Calculate several metrics to evaluate each model and pipeline.

Results are slightly misleading because a high accuracy score does not necessarily imply a quality model. If the model predicts 0 for every record in the test set, it would be correct around 92% of the time.

Future steps will include further investigation into the various features in the data sets. Checking for multicollinearity may identify issues with feature selection as well. We intend to build a neural network model and compare it's performance to the baseline models above.

6 Credit assignment plan

Home Credit Default Risk
Credit assignment

Applied Machine Learning

	Phase 2: EDA and baseline pipeline. Team Lead: Paul Miller	Team member
	Abstract, organize notebook	Paul Miller
	Load data	Glen Colletti
	EDA	Alex Bordanca
	Visual EDA	Alex Bordanca
	Baseline models and pipelines. XGBoost, KNN, Logistic Regression	Glen Colletti
	Create presentation slides	Glen Colletti
	Credit Assignment	Paul Miller
	Record video	All members

[ ]: