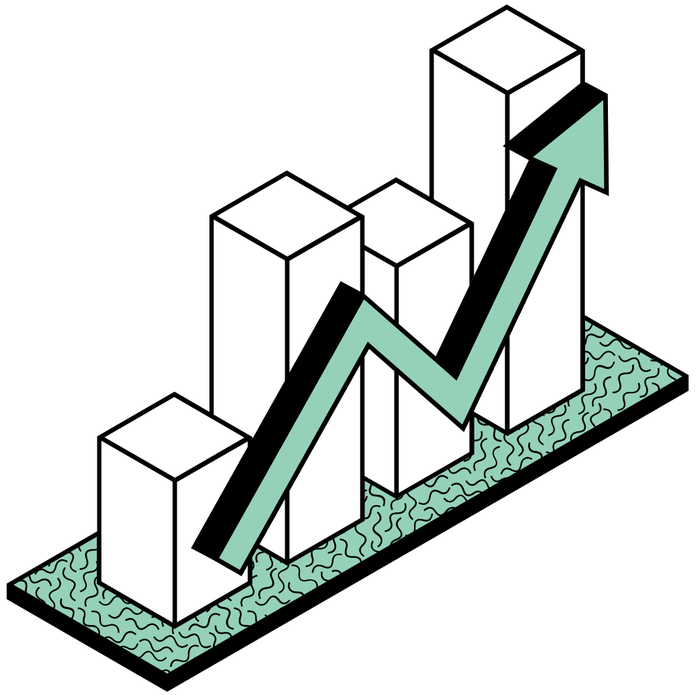
CREDIT RISK
PRESENTATION

Group 13 – Tres Gato
Nguyen Cam Ly
Vo Thi Yen Nhi
Nguyen Thanh Long



CONTENT

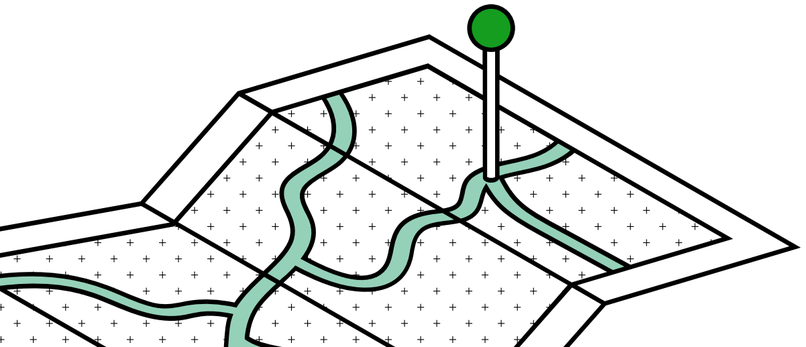
Step 1 — Overview

Step 2 — Exploratory Data Analysis

Step 3 — Data Preparation

Step 4 — Hyperparameter Tuning

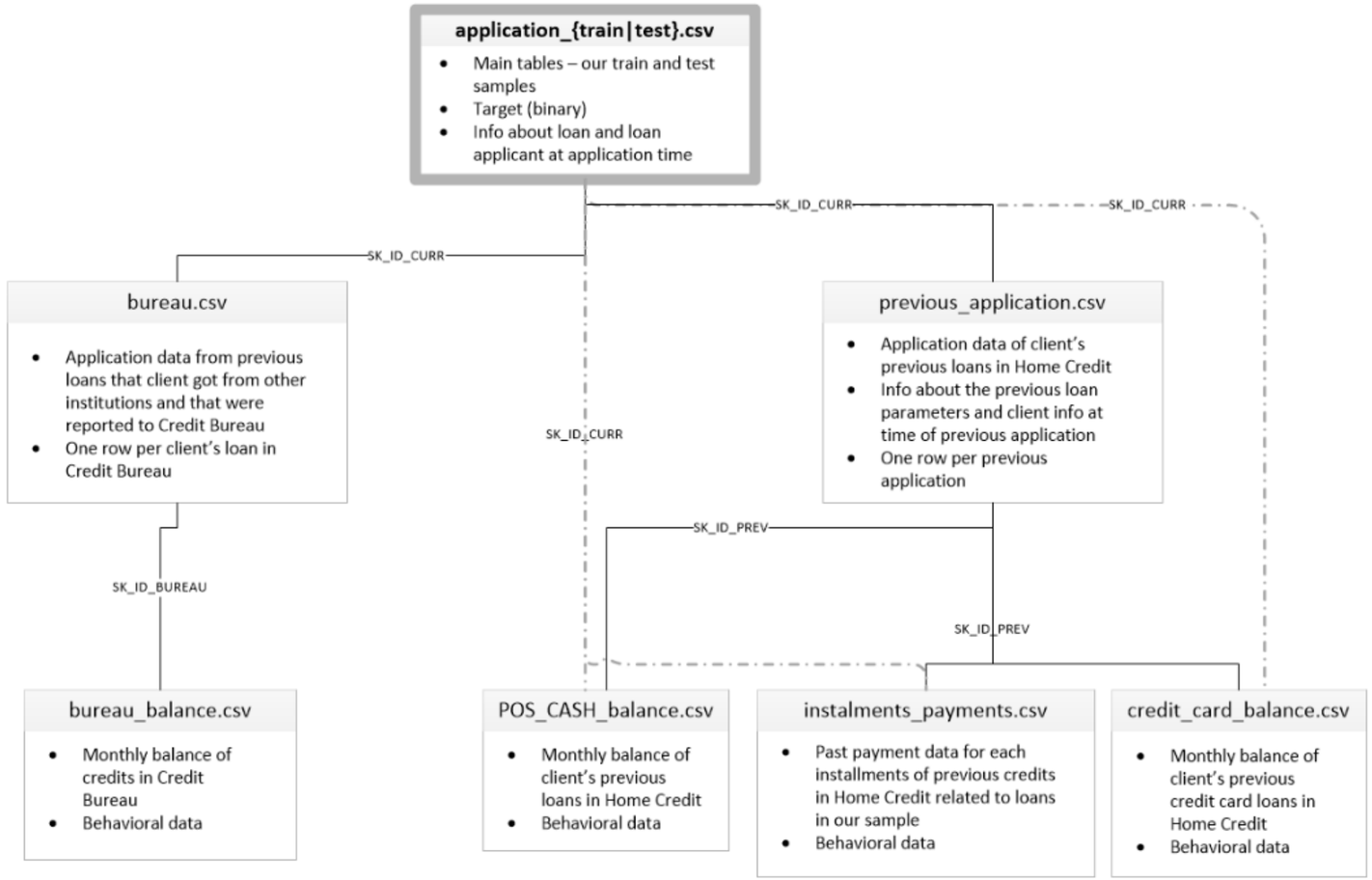
Step 5 — Prediction



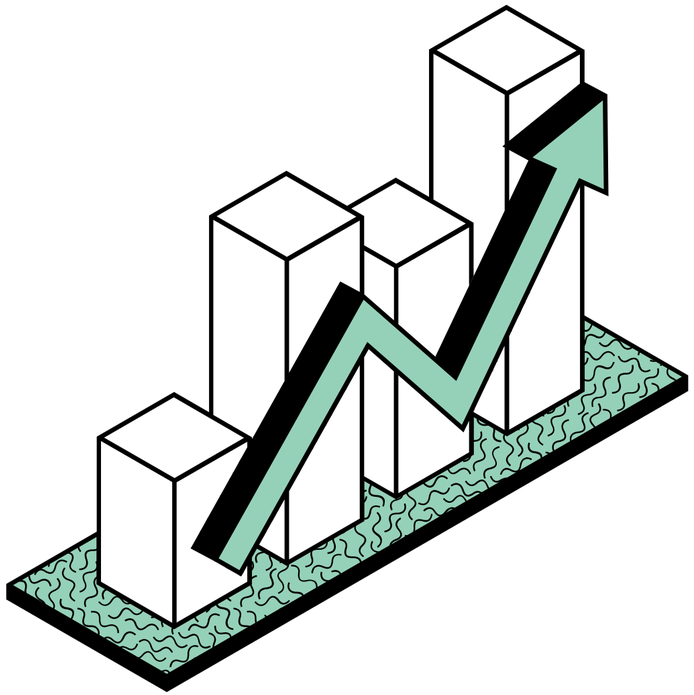
I. Overview

In this project, we aim to develop a predictive model that evaluates loan applicants' creditworthiness and minimizes the risk of default. This project is crucial because it can help financial institutions make informed decisions about loan approvals, which can ultimately reduce the risk of losses due to unpaid loans.

The dataset consists of 7 tables, with the main one being application. They contain information related to current and previous applications, credit, as well as repayment history.



II. Exploratory Data Analysis



1. Import Libraries
2. Load data & Basic information
3. Target column
4. Missing values
5. Define numeric and category features
6. Imbalanced categorical features
7. Outliers
8. Anomalies
9. Correlation
10. Categorical analysis
11. Numerical analysis

Basic information

1. **Shape**
2. **Number of categorical variables and numerical variables**
3. **Number of duplicate values**
    - There is no duplicated in data
4. **Missing value**
5. **Dtype of features**

TARGET column

Class imbalanced problem

- The proportion between class 0 and class 1 is about 92:8
- Using SMOTE to deal with this problem

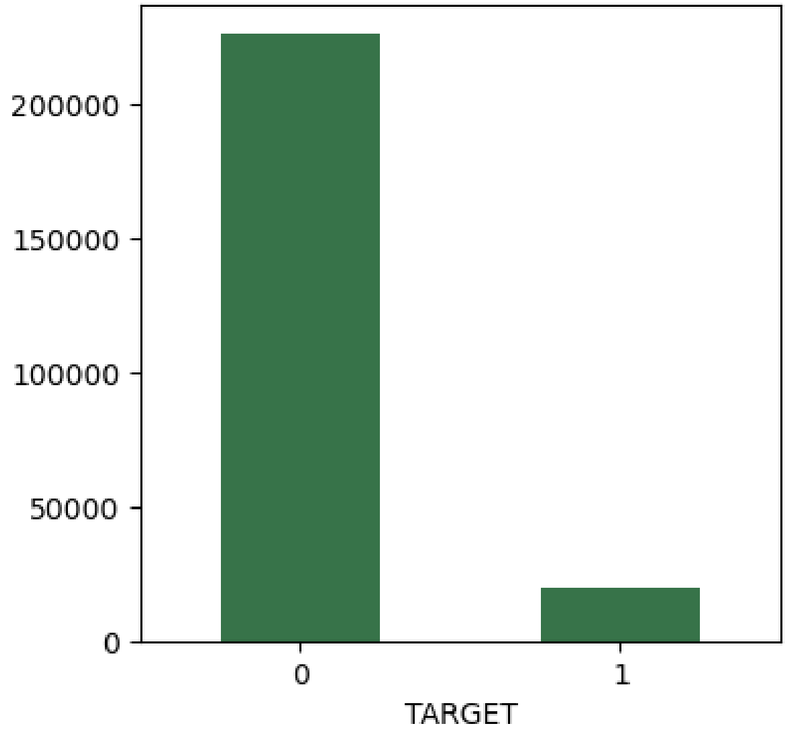


table: application_train

Missing values

- All tables have missing values
- Some features have up to 70% missing data
- Consider:
  - For features with fewer missing values, we can predict or fill missing values using regression or mean values, depending on the feature.
  - For features with high missing values, it's better to drop them as they provide little insight.

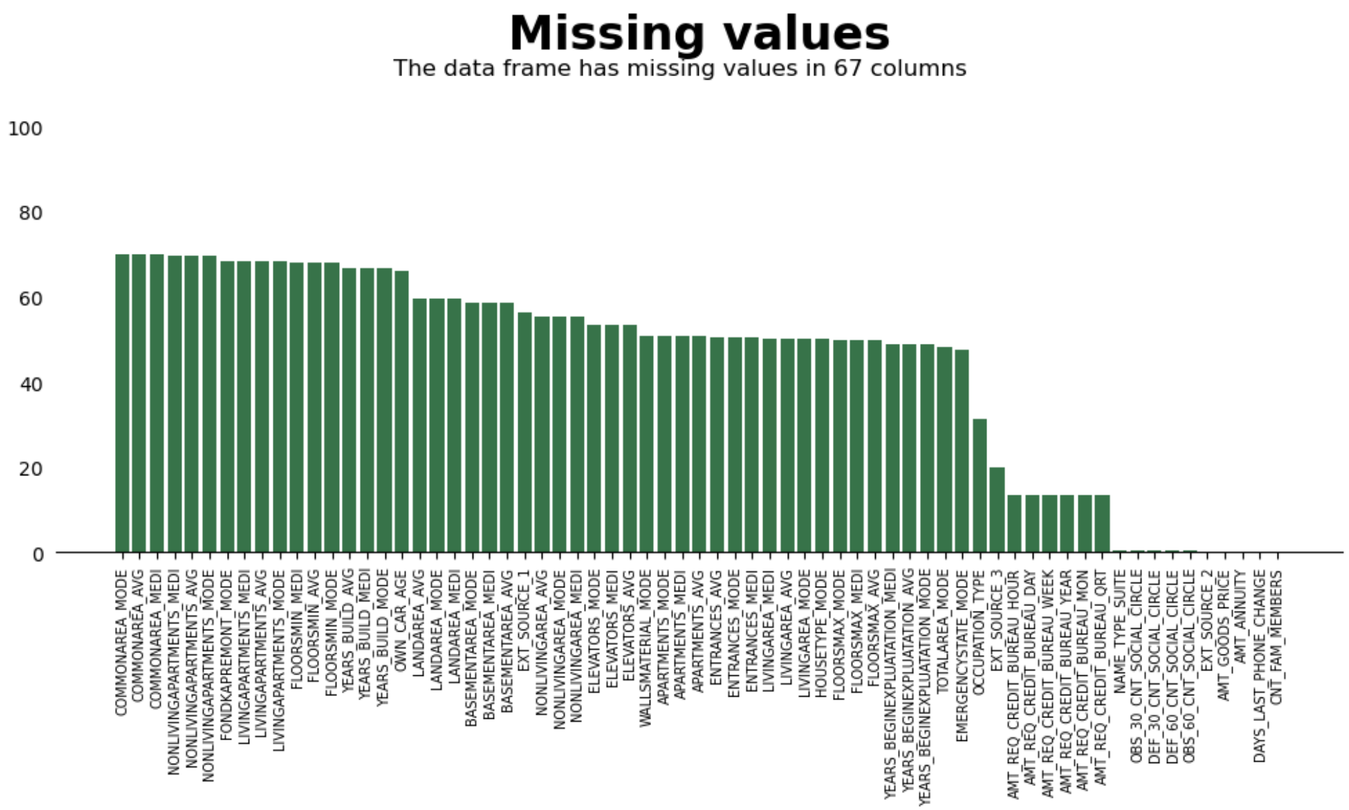


table: application_train

Define numeric and category features

- Except for the application table, numeric features have the numeric dtype and categorical features have object dtype
- In the application table: all categorical features whose dtype are int8 have less than 4 nunique

Imbalanced categorical features

Many columns have high imbalanced classes that the top class can account for nearly 100% frequency

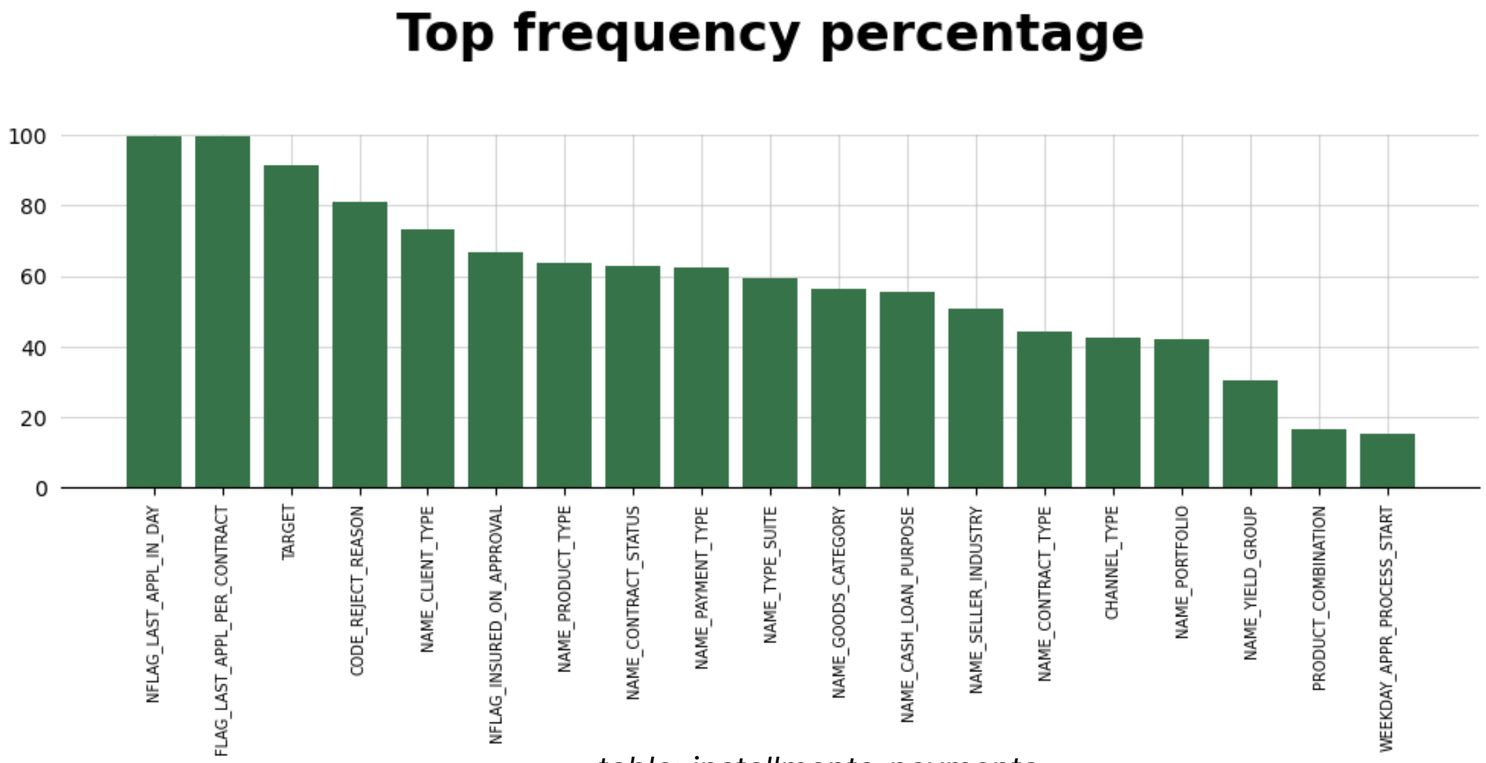


table: installments_payments

Anomalies

- In categorical features: value ‘XNA
- In day features: value ‘365243’
- Both 2 values can defined as Null value

Outliers

Most features in the dataset have many outliers and skewed distributions

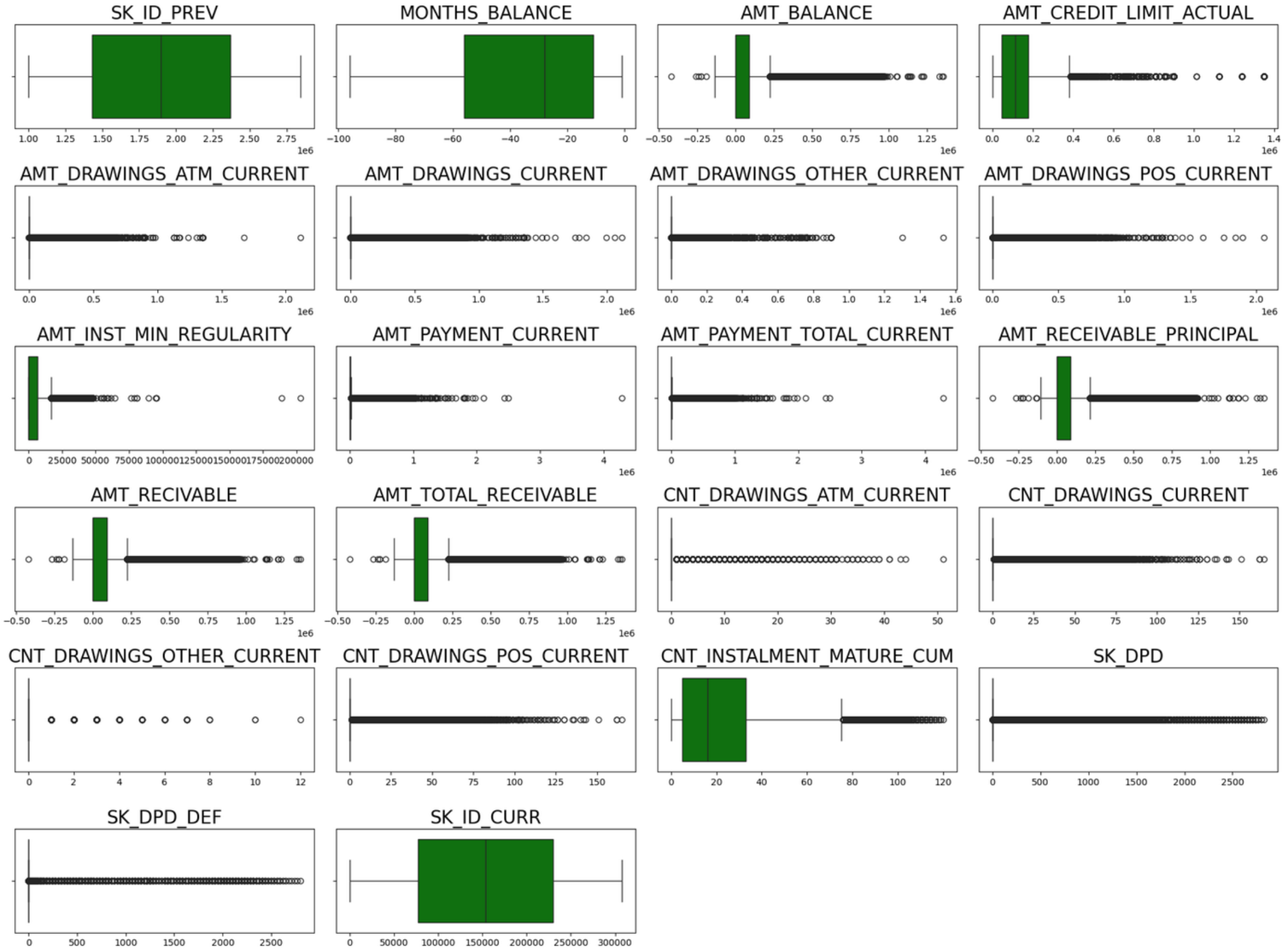


table: credit_card_balance

Correlation

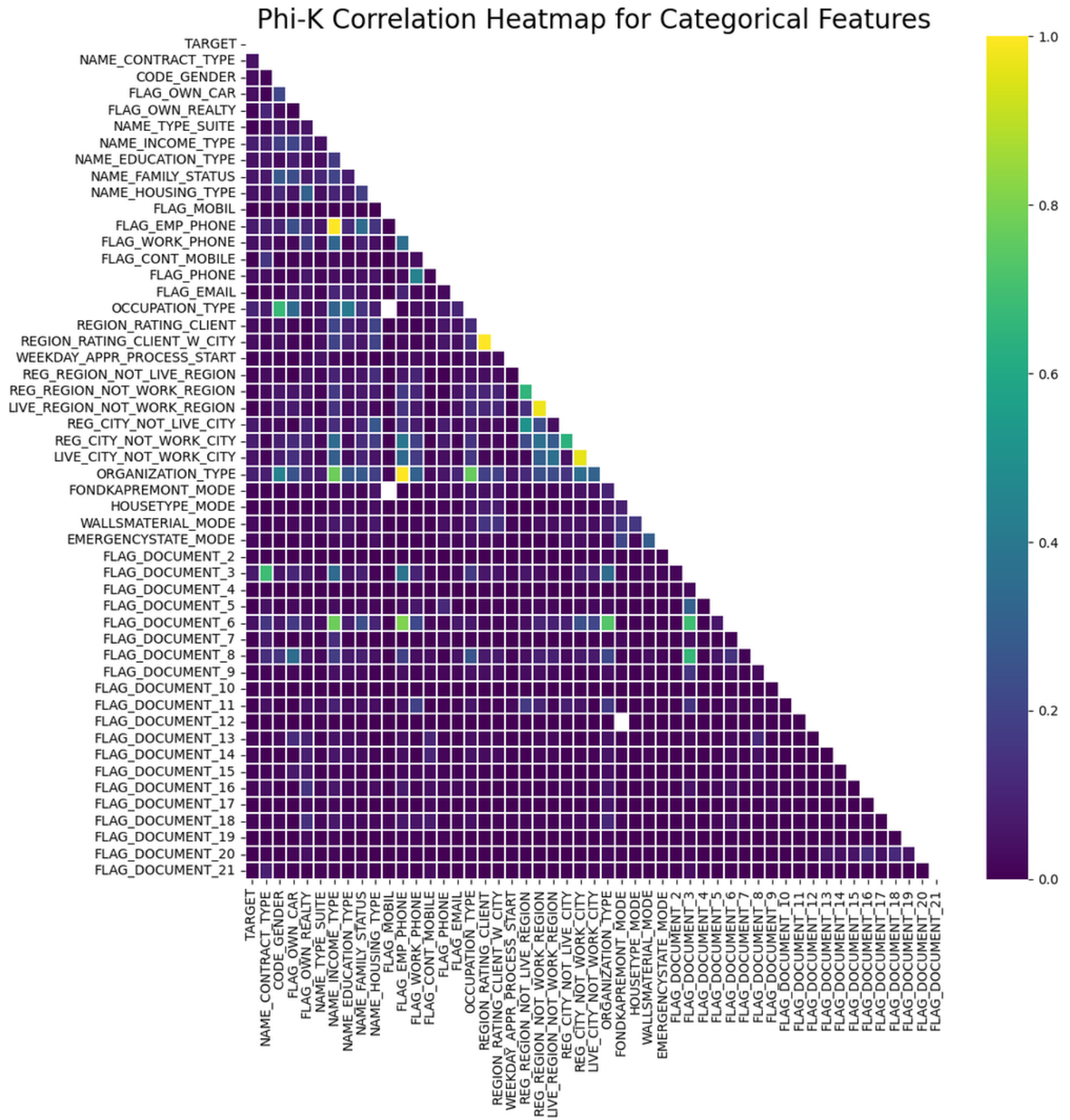
Categorical features

Use Phi-K to calculate the correlation

Features highly correlated with each other often come from the same field

E.g.
REGION_RATING_CLIENT_W_CITY
REGION_RATING_CLIENT

table: application_train



Correlation

Numerical features

Use a heatmap to visualize the correlation

.

- Features highly correlated with each other often come from the same field

E.g.
AMT_RECEIVABLE_PRINCIPAL,
AMT_RECIVABLE,
AMT_TOTAL_RECEIVABLE,
AMT_BALANCE, and
AMT_INST_MIN_REGULARITY

- EXT_SOURCE_3, EXT_SOURCE_2, EXT_SOURCE_1, and DAYS_BIRTH are features with the most highly correlated with the TARGET column. These might be important for our classification task.

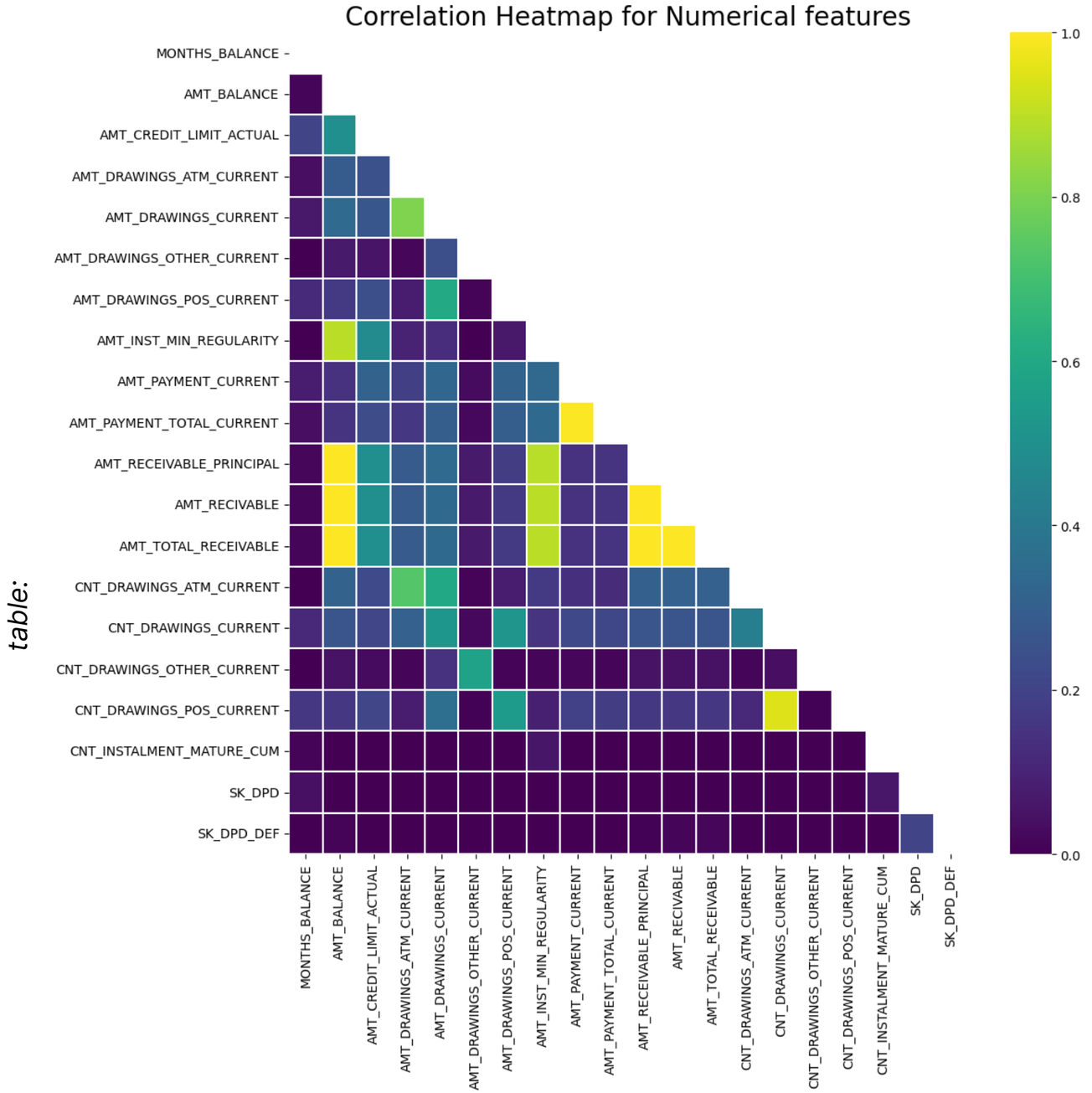
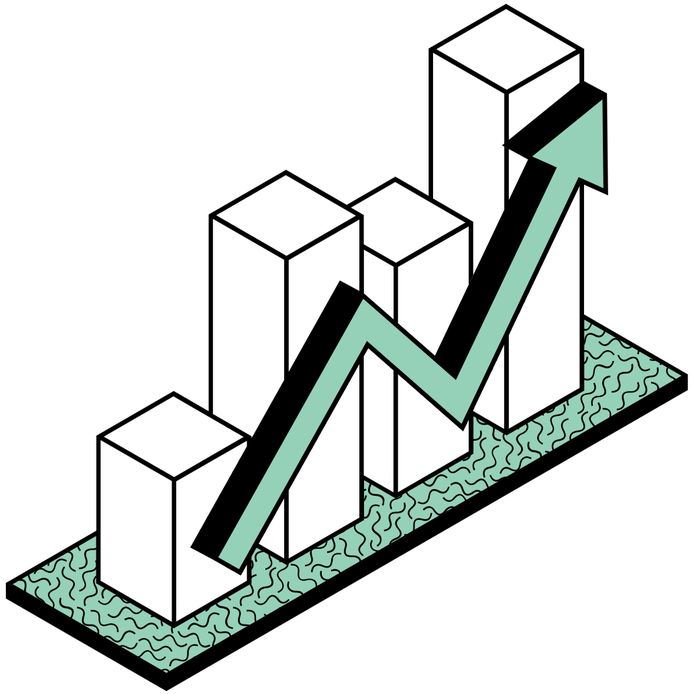
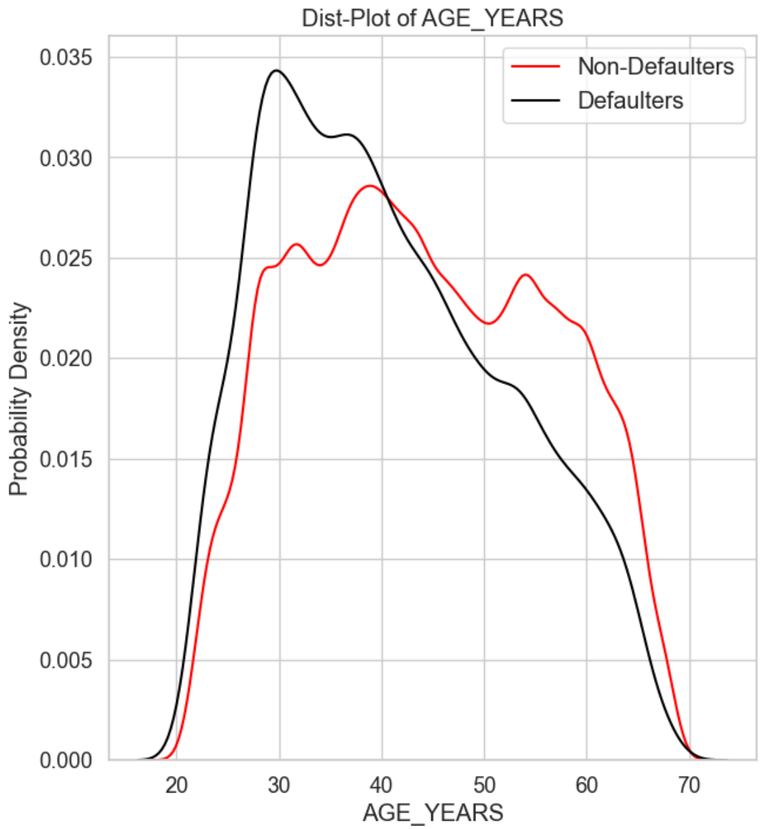


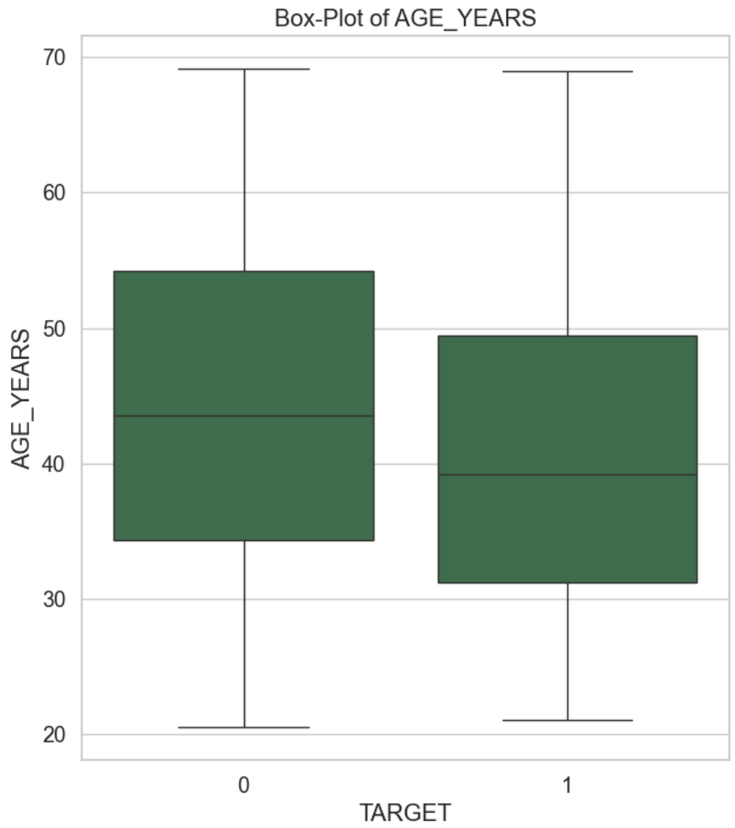
table:

Categorical and Numeric analysis



DAYS_BIRTH

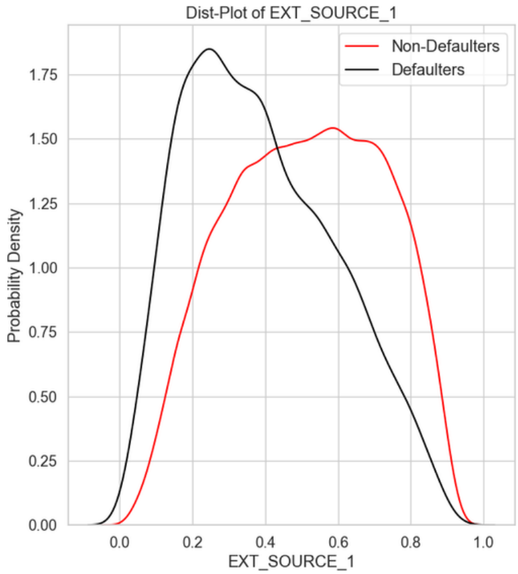


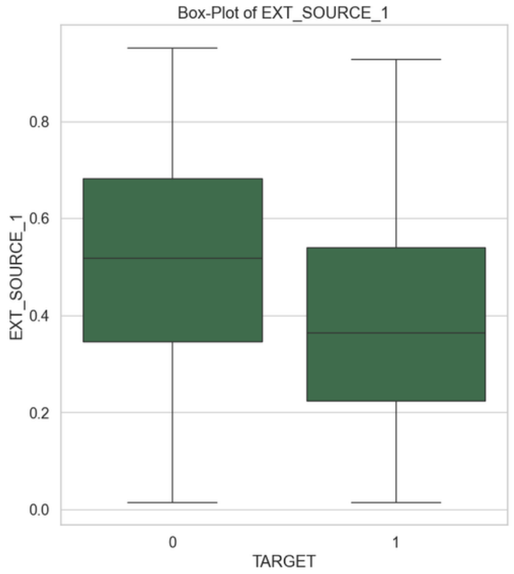


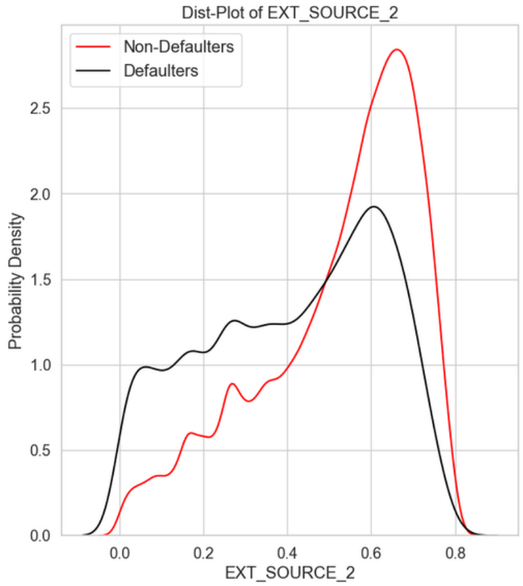
Defaulters are often younger than non-defaulters

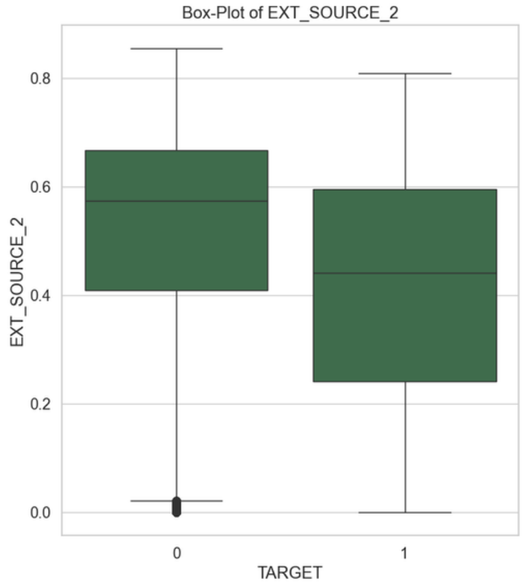
EXT_SOURCES

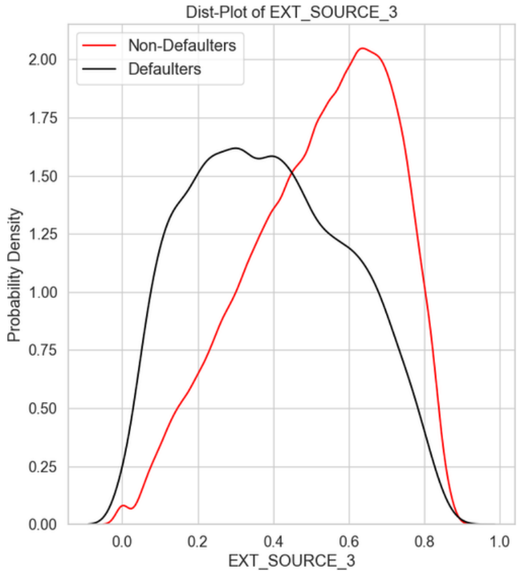
Most effective in linearly separating Defaulters and Non-Defaulters among all the features explored so far.

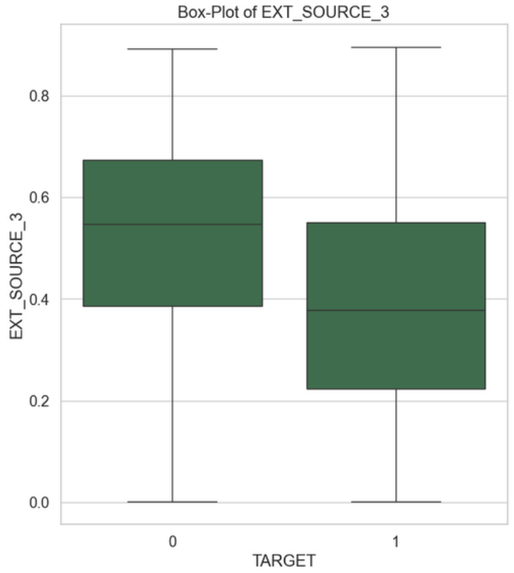






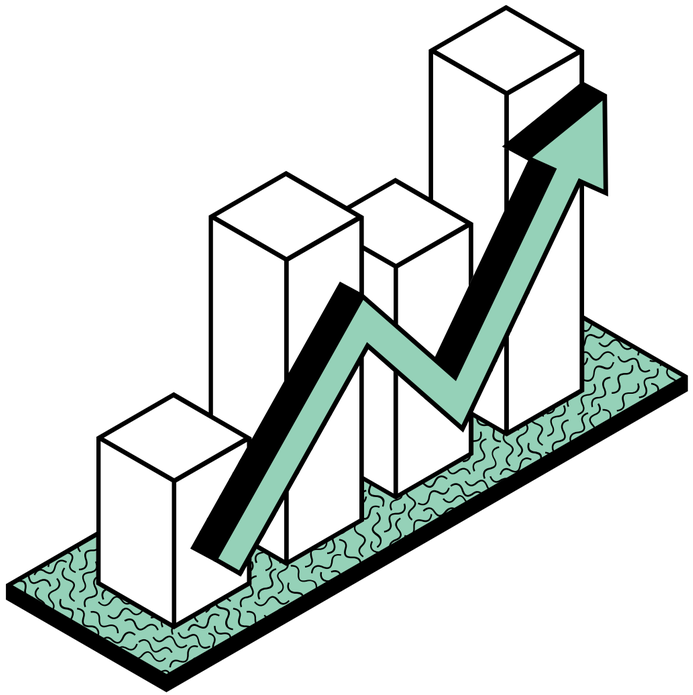




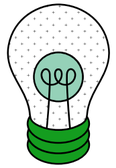


III. Data Preparation

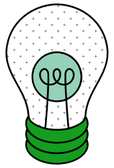
- Data cleaning
- Feature engineering



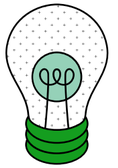
Data cleaning



Replace infinity values with NaN due to division by 0

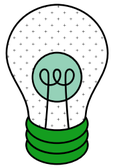


Remove high imbalance columns



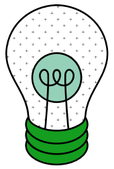
Remove anomalies

- Replace XNA value with np.nan
- Replace 365243 with np.nan in day column



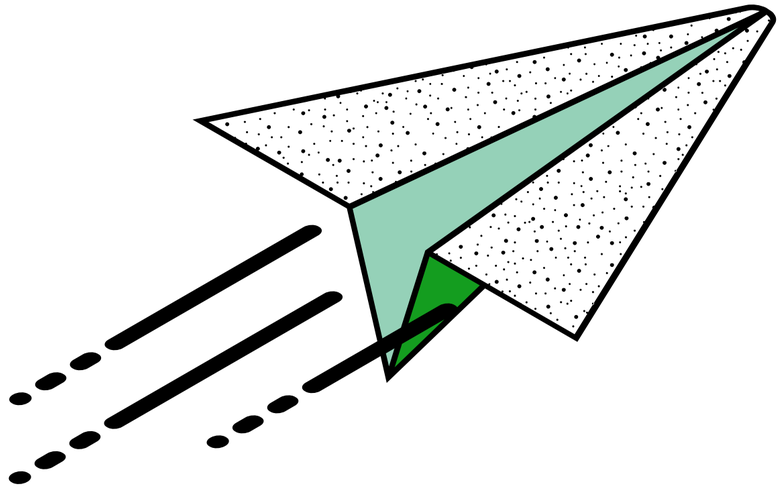
Missing values

- Numerical columns: mean
- Categorical columns: 'Missing'
- Remove columns with high missing percentage



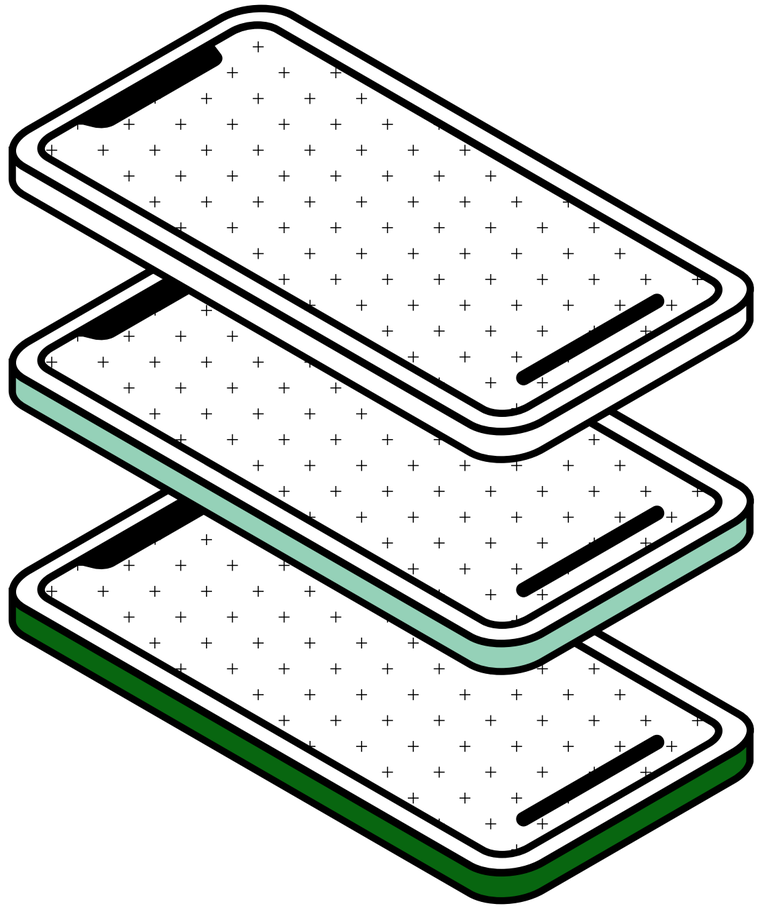
Remove outliers

with 3 standard deviations



Feature engineering

- Create new features based on original features
- Aggregations
- Create polynomial features
- Data binning and feature selection with WOE, IV



Feature engineering

- **Create new features based on original features**

  E.g. Employment duration as a percentage of life predicts loan payment ability.

  app['DAYS_EMPLOYED_PCT'] = app['DAYS_EMPLOYED'] / app['DAYS_BIRTH']

- **Aggregations**

  E.g. Counts of a client's previous loans

  cnt_previous_loan = self.bureau.groupby('SK_ID_CURR')[['SK_ID_BUREAU']].count()

- **Create polynomial features on EXT_SOURCE_1, EXT_SOURCE_2, EXT_SOURCE_3, DAYS_BIRTH**

Feature engineering - Data Binning

Binning numerical features into categorical features and merging some classes having small nunique categorical features

E.g. feature AMT_INCOME_TOTAL of application table:

```
X.APP_AMT_INCOME_TOTAL
```

```
0         270000.0
1         135000.0
2         121500.0
3          99000.0
4         360000.0
            ...
246004    157500.0
246005     72000.0
246006    153000.0
246007    171000.0
246008    157500.0
Name: APP_AMT_INCOME_TOTAL, Length: 246009, dtype: float64
```

```
bins['APP_AMT_INCOME_TOTAL']
```

```
[-inf,
 76477.5,
 98977.5,
 139484.25,
 160814.25,
 185517.0,
 223456.5,
 229601.25,
 313875.0,
 inf]
```

WEIGHT OF EVIDENCE (WOE)

- **WOE (weight of evidence)** is a widely used feature engineering and feature selection technique in scorecard modeling.
- This method ranks variables into strong, medium, weak, useless, etc. based on their ability to predict bad debt.

$$WOE = \ln\left(\frac{\%Good}{\%Bad}\right)$$

Where:

- No observation: Number of observations in each bin. It will usually be divided equally
- between bins.
- No Good: Number of bad debt records in each bin (label 1)
- No Bad: Number of non-bad debt records in each bin (label 0)
- Good/Bad: Ratio of Good/Bad records in each bin.
- %Good: Distribution of good records across all bins
- %Bad: Distribution of bad records across all bins

Information Value (IV)

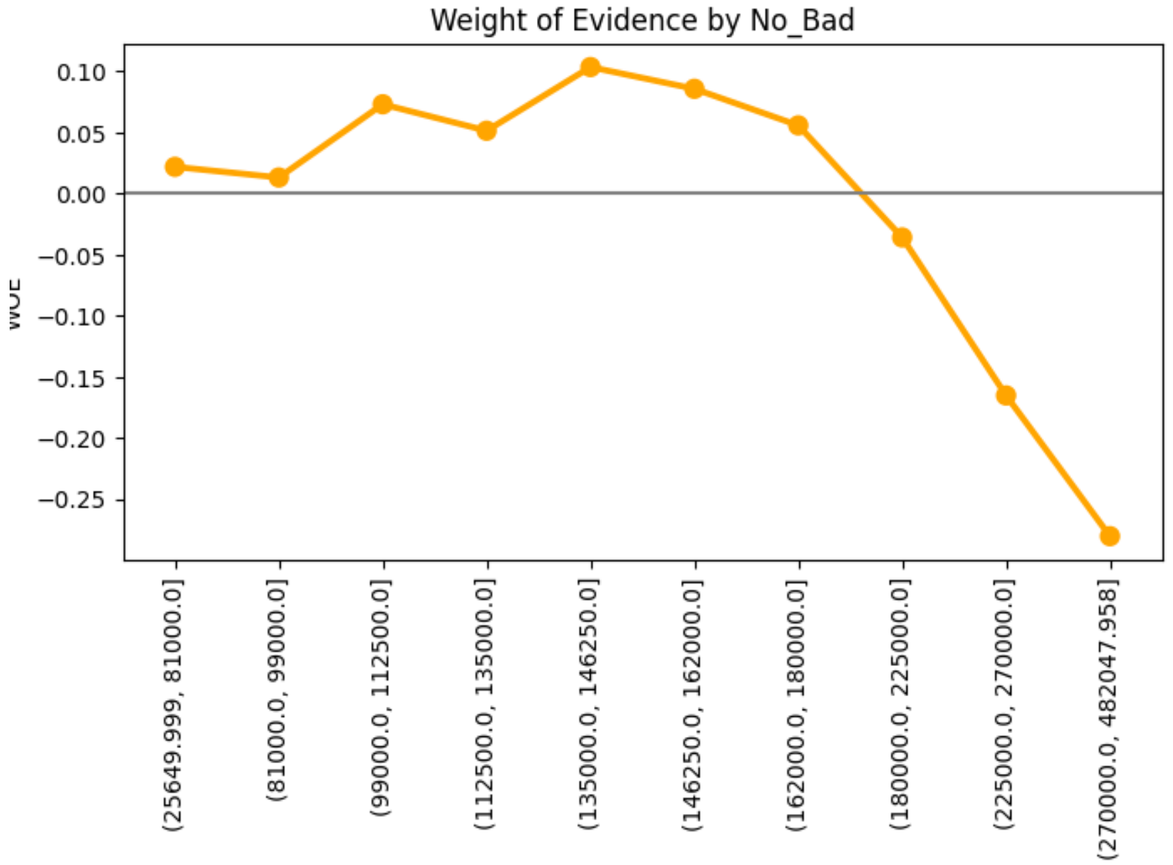
- Information value evaluates whether a variable has power in classifying bad debt or not

$$IV = \sum_{i=1}^{n} (\%Good_i - \%Bad_i).WOE_i$$

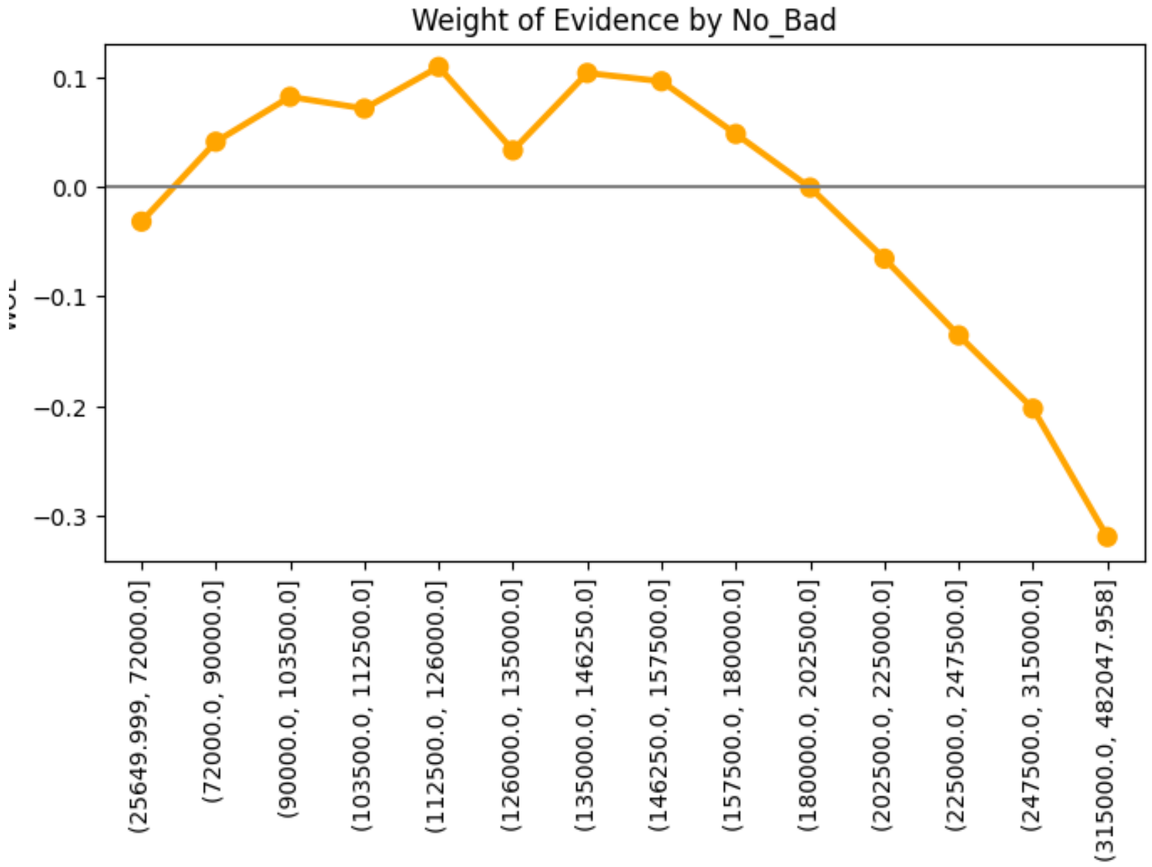
- Some documents provide standards for classifying the power of variables according to the IV value as below:
  - ≤ 0.02: useless (feature has no effect in classifying good/bad records)
  - 0.02 – 0.1: weak
  - 0.1 – 0.3: medium
  - 0.3 – 0.5: strong
  - ≥ 0.5: suspicious

WEIGHT OF EVIDENCE (WOE)

```
plot_woe(calculate_WOE(X, y, 'APP_AMT_INCOME_TOTAL', nbins=10)[0], rot=90)
```




```
plot_woe(calculate_WOE(X, y, 'APP_AMT_INCOME_TOTAL', nbins=15)[0], rot=90)
```



	Features	IV	Rank
0	APP_EXT_SOURCE_1 EXT_SOURCE_2 EXT_SOURCE_3	0.622848	suspicious
1	APP_EXT_SOURCE_2 EXT_SOURCE_3 DAYS_BIRTH	0.562629	suspicious
2	APP_EXT_SOURCE_2 EXT_SOURCE_3	0.555144	suspicious
3	APP_EXT_SOURCE_2 EXT_SOURCE_3^2	0.544560	suspicious
4	APP_EXT_SOURCE_2^2 EXT_SOURCE_3	0.518506	suspicious
17	APP_EXT_SOURCE_3	0.332887	Strong
20	APP_EXT_SOURCE_2	0.315026	Strong
33	CURRENT_DEBT_TO_CREDIT_RATIO_MEAN	0.124582	Medium
34	DAYS_CREDIT_MEAN	0.115925	Medium
35	APP_CREDIT_TERM	0.115534	Medium
36	APP_DAYS_EMPLOYED	0.106796	Medium

Polynomial features based on EXT_SOURCE_1, EXT_SOURCE_2, EXT_SOURCE_3, DAYS_BIRTH

Feature with highly correlation with 'TARGET'

Feature created by aggregations

New features created by original features

Model using binning

- **About 160 features selected**
- **Score: 0.54 - 0.59**

Model without binning

- **About 800 features selected**
- **Score: 0.48 - 0.51**

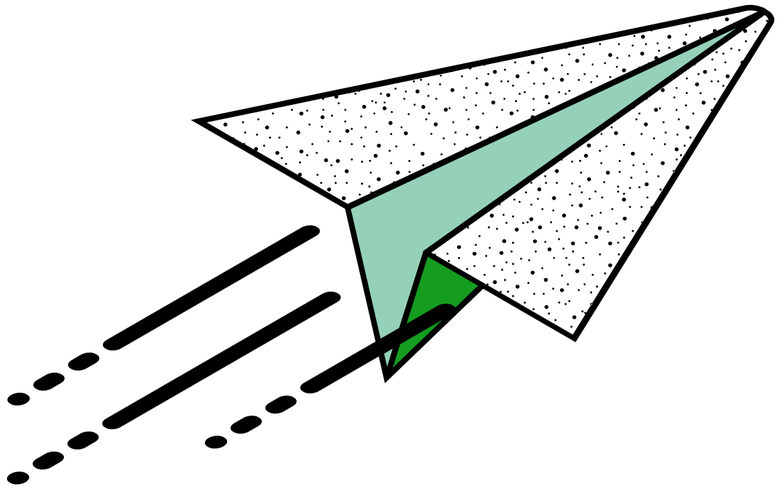
IV. Hyperparameter tuning

- GridSearchCV
- RandomizedSearchCV

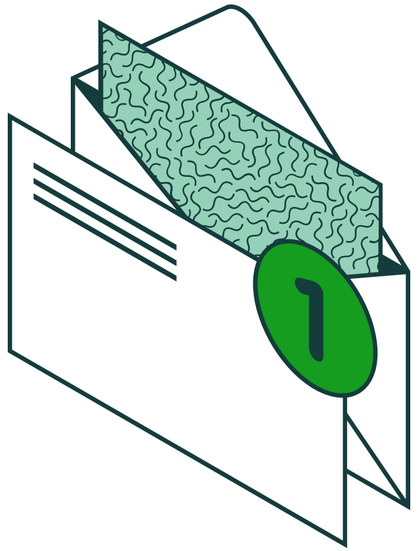
V. Prediction

- Split data and resample train set (using SMOTE)
- Model: Logistic regression
- Score: Gini

  Formula: gini = AUC x 2 - 1



Thanks for listening



Do you have any questions?