Credit underwriting ML project

Fred Serfati - 07/23/2024

Plan

1. Introduction: Credit underwriting
2. EDA
3. Data Preprocessing & Feature Engineering
4. Data Visualization
5. Modeling
6. Results and Feature importances
7. Conclusions
8. Q&A

Introduction: Credit underwriting

- Determine the creditworthiness of an applicant, by identifying whether they should be given a loan in the future
- Helps financial institutions manage risks and ensure profitability
- Accurate credit underwriting decisions:
  - Protect lenders from financial losses
  - Support borrowers in accessing fair credit opportunities

Modelisation using Machine Learning

- **Supervised problem**: access to outcome/label of loan
- **Binary classification:** categorical target with 2 categories: good (non-default) & bad (default)

Exploratory Data Analysis

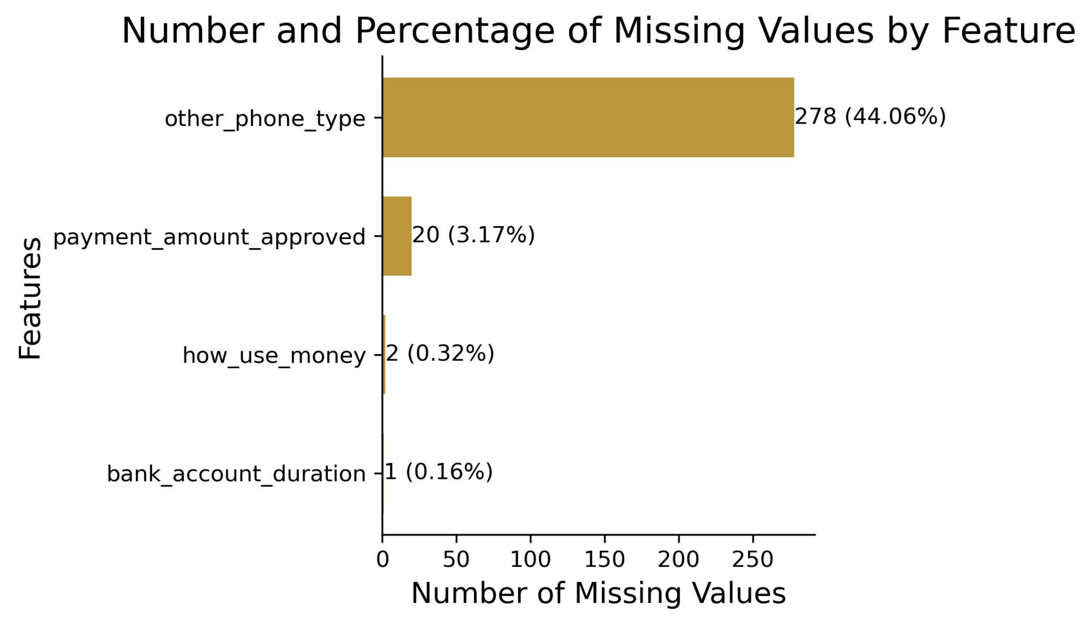
Data preparation

- 2 datasets:
    - **Application** (644 records) for every customer that has been given a loan in a 6 month period
    - **Loan** (1266 records) for the outcome of those loans: good (i.e., non default) or bad (default)
- **Merged** Loan data (target) to Application data (variables) using a **left join**
- Final dataset: only 631 records: **tiny dataset!**

Dataset presentation

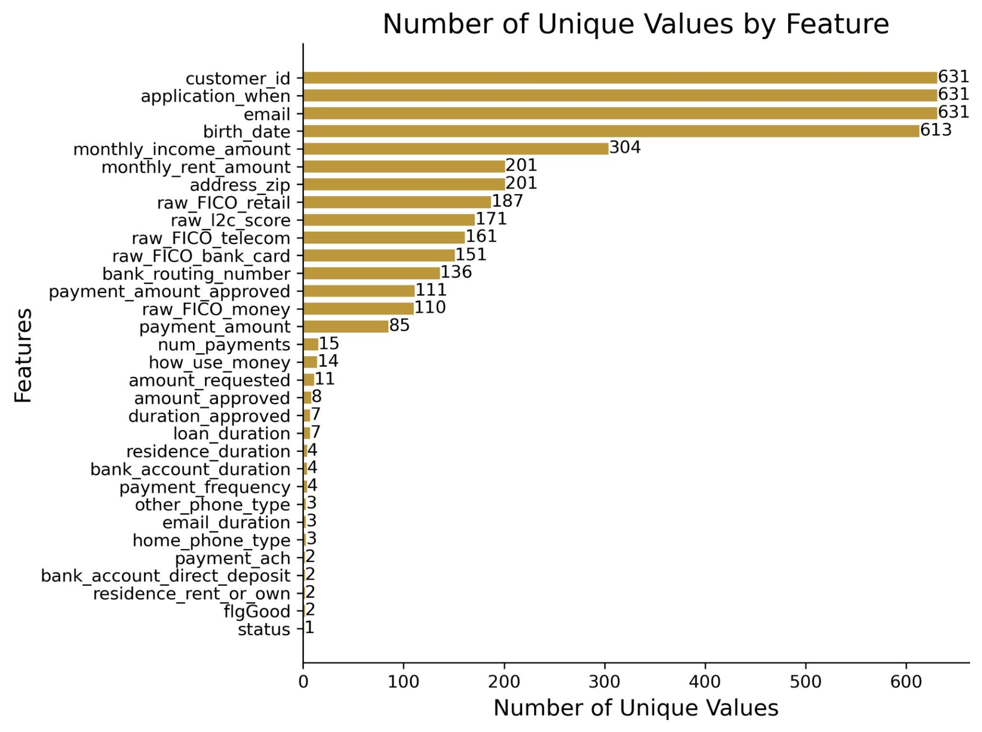
- 631 records, 32 columns
- **Target**: flgGood, i.e., whether a loan defaulted (‘Bad’, 0) or not (‘Good’, 1)
- Different types of variables:
  - **Numerical**: e.g., amount_requested, monthly_rent_amount
  - **Categorical**:
    - **Ordinal**: e.g., email_duration, residence_duration
    - **Nominal**: e.g., bank_account_direct_deposit, how_use_money
- (Stratified) train test split:
  - **80% (488) for train/validation**, used for model training, fine-tuning and model selection
  - **20% (122) for test**, used for predictions (**nothing else!**)

Dealing with missing values



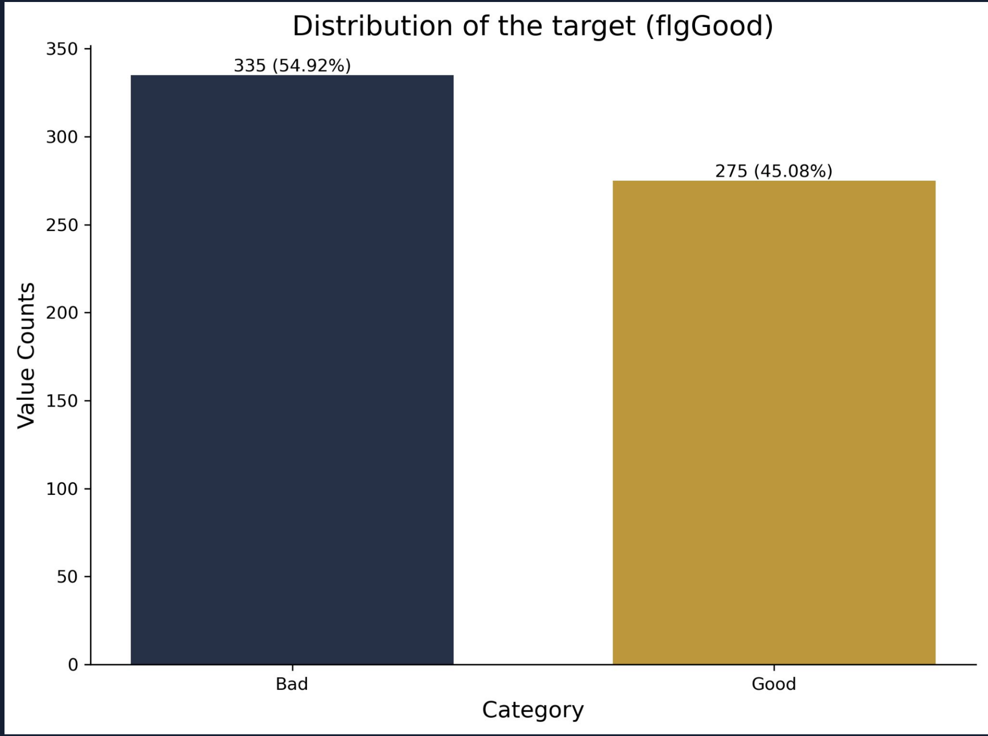
- Other_phone_type: dropped column
- 3 other features: dropped rows with null values (very few)
- Tried imputation with mean for payment_amount_approved but let to poorer performances

Unique values



- Customer_id, application_when and email: dropped or transformed (1 unique value per application)
- Status: dropped (1 unique value, because all loans in this dataset were approved; i.e., 0 variance)

A slightly imbalanced dataset



- **45%** of **good** loans vs **55%** of **bad** loans: **slight imbalance**
- Still very limited, so no need for advanced sampling methods (such as SMOTE)
- However, **choice of metrics** still **very important** (can’t use accuracy)

Data Preprocessing & Feature Engineering

Data preprocessing

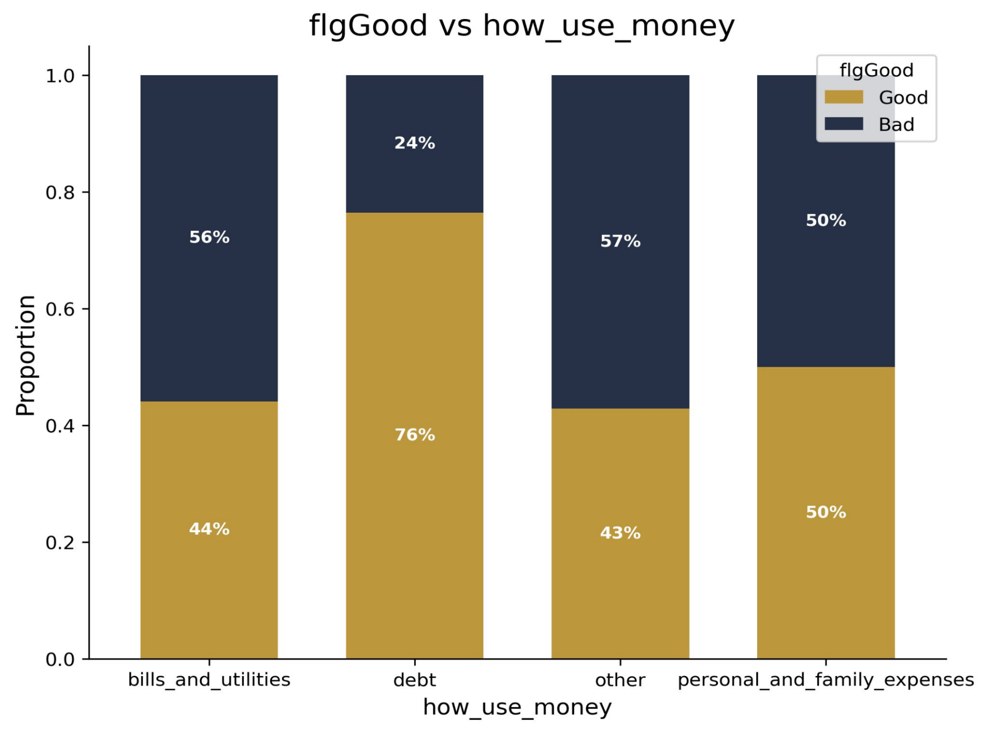
- **Numerical features**:
  - Standard scaling
  - No need for missing values imputation: already removed all missing values
- **Nominal variables**: one-hot-encoding
- **Ordinal variables**: ordinal encoding

Feature engineering

- Age
- Year, month and day of week of application
- How_use_money: categories regroupment
- Zip code: only kept 3 first digits to reduce dimensionality
- Average FICO score (money, retail, telecom, bank card)
- Potentially useful ratios, e.g.:
  - Debt-to-income (DTI) and approved DTI
  - Approved-to-requested-amount: proportion of the requested amount that was granted
  - Approved-to-requested-loan-duration: proportion of the requested duration that was granted

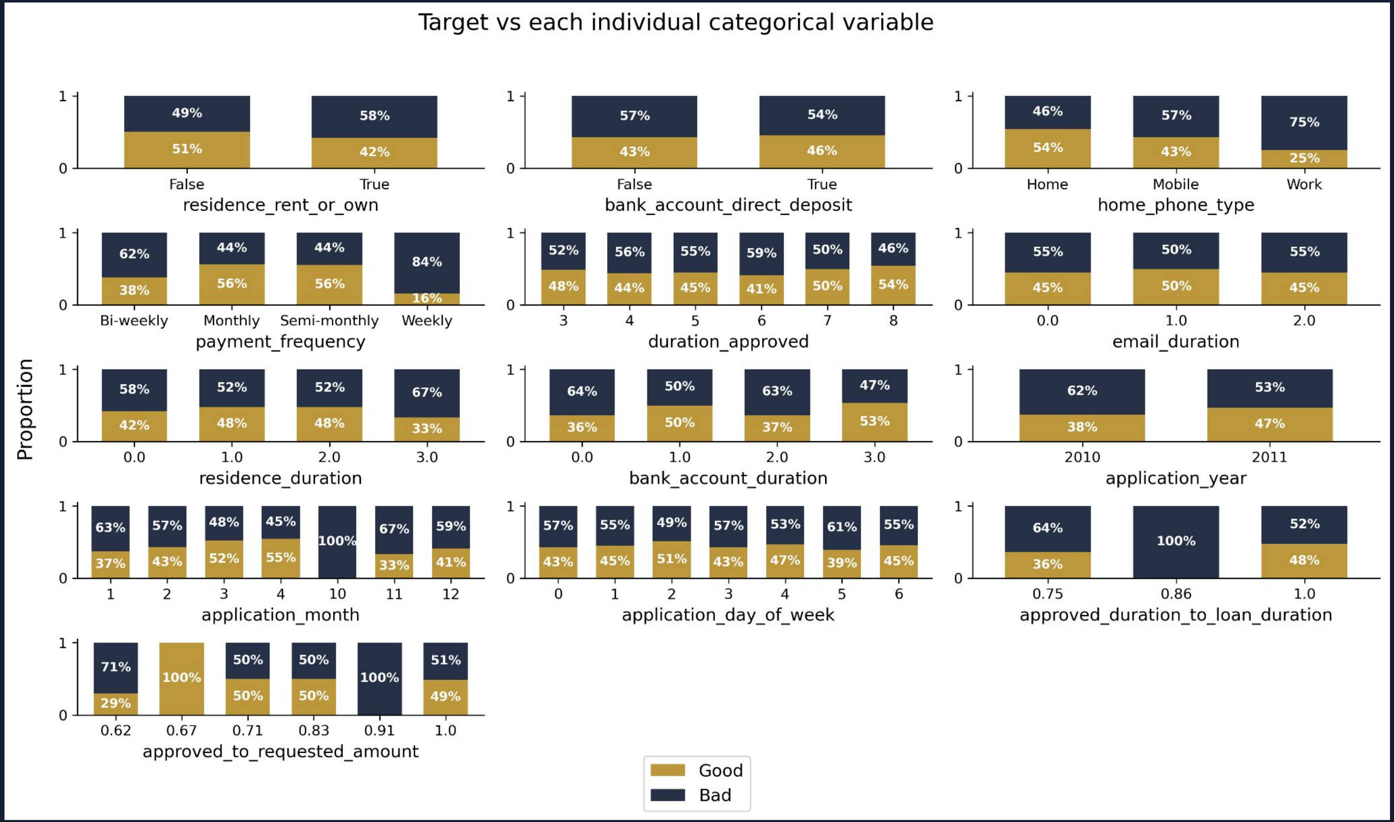
Data Visualization

Relationship between target and categorical features (1/2)

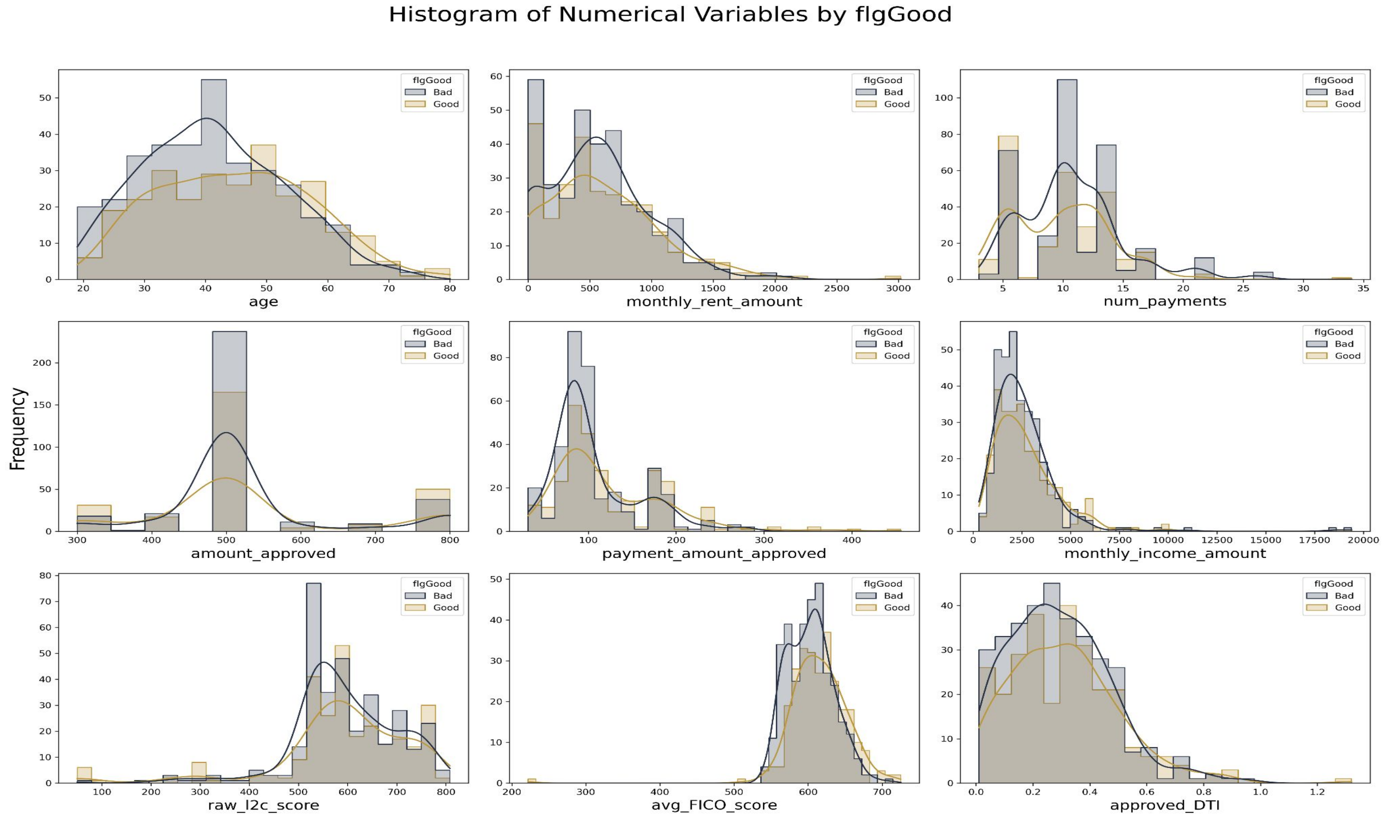


- 76% of loans used for debt reimbursement were good
- Only 50% for personal and family expenses

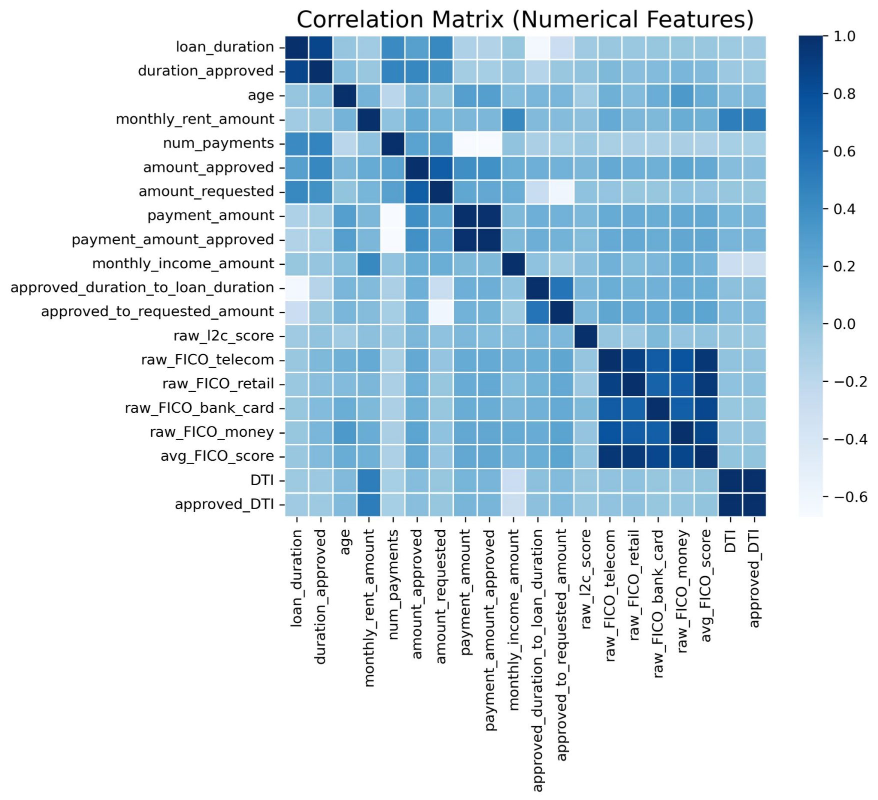
Relationship between target and categorical features (2/2)



Relationship between target and numerical features

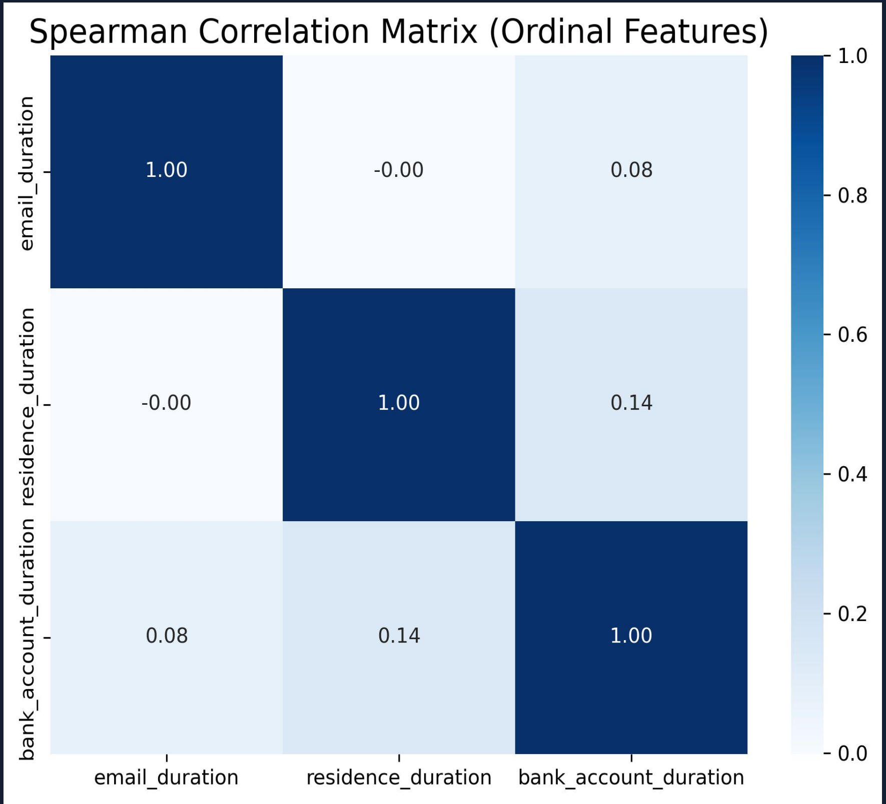


Pairwise relationships between **numerical** features



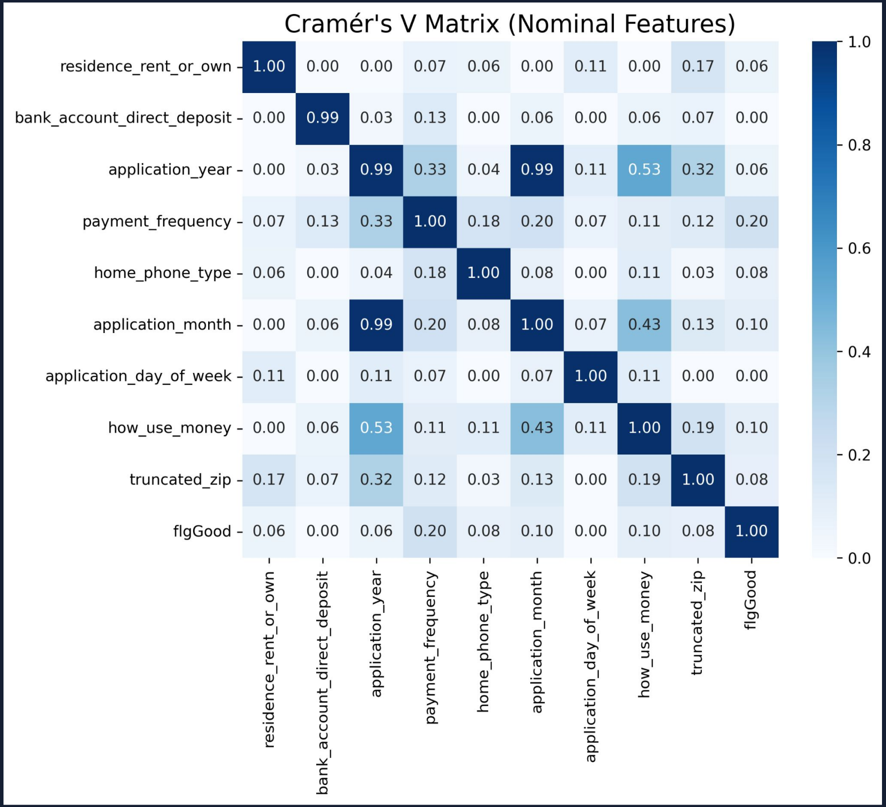
- Strong correlation between:
  - Requested & approved duration
  - Requested & approved amount
  - Requested & approved payment amount
  - Loan duration & number of payments
  - All FICO scores
  - …
- Moderate correlation between:
  - Num_payments & approved_duration
  - Num_payment & amount_approved
  - …

Pairwise relationships between **ordinal** features



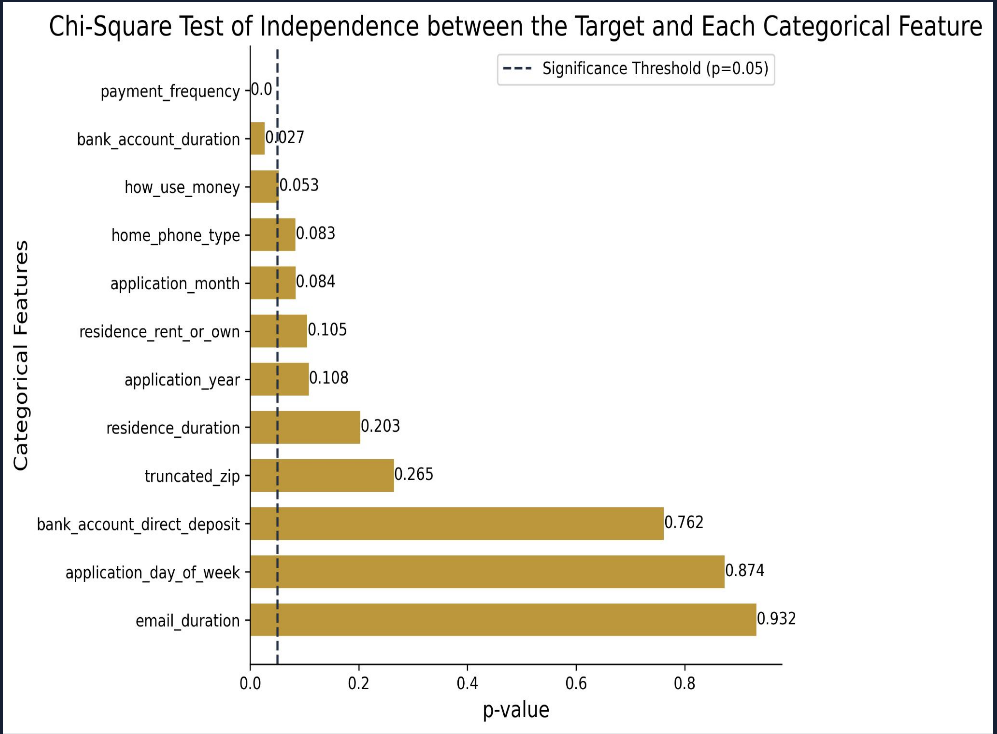
Nothing remarkable here: no "correlation" between any pair of ordinal features

Pairwise relationships between **nominal** features



- Strong relationship between application year & month:
  - Probably because data from 2010 only cover October to December while data from 2011 only cover January April
- Moderate relationship between application month (and year) & how_ use_money

Chi-square tests of independence



- H0: ind. vs H1: non-ind.
- Payment_frequency, bank_account_duration and how_use_money:
  - **Reject H0**: target not independent of those variables
- All other categorical variables:
  - **Fail to reject H0**: not enough evidence to conclude variable not independent from target
- Careful with interpretation because of multiple testing problem

Modeling

Why we can’t use accuracy

- (Slight) imbalance problem, so biased towards majority class (‘Bad’)
- More importantly, in credit underwriting, we don’t only care about overall correctness, but also about:
  - **False positives** (granting a credit that will default: **money loss**)
  - **False negatives** (missing a good loan: **missed opportunity**)

Metrics used

- **Balanced accuracy**: weighted version of accuracy, taking into account class imbalance
- **Precision**: proportion of true positive (good loans) out of all loans predicted as good
  - The higher, the more we can trust that if a loan is predicted as good, it will really be good
- **Recall (TPR)**: proportion of true positives (good loans) out of all actual good loans
  - The higher, the more good loans are identified by the model
- **F1-score**: balance between precision and recall
  - Useful here, as we care both about precision and recall
- **AUC-ROC**:
  - The higher, the better the model is able to distinguish between good and bad loans
- **AU-PRC**:
  - The higher, the better the model performs with the positive (good) class

Model evaluation: stratified 5-fold cross-validation

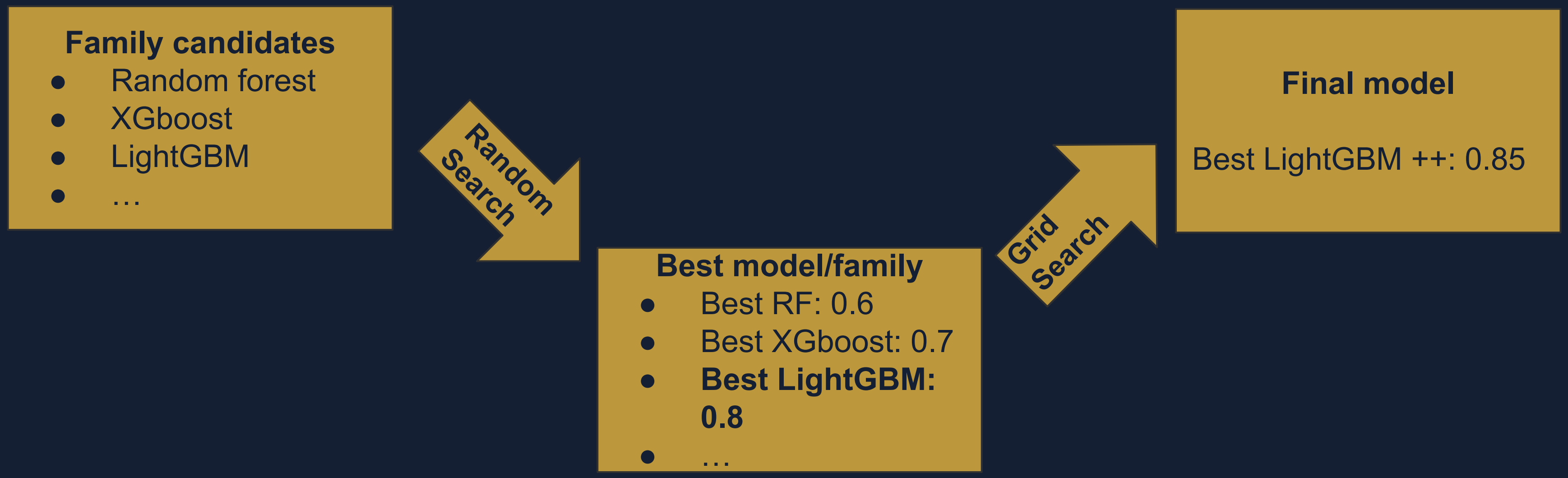
- Why **cross-validation** instead of a **simple train/valid split**?
    - **Very small training dataset** (less than 500 records):
        - High variance in metrics from one split to another: not trustworthy
    - To avoid bad surprises once model in production, need to robustly evaluate generalization error: **cross-validation**
    - The higher the number of folds, the more robust (but computationally more expensive)
- Why **stratified**?
    - Same proportion of Good vs Bad loans in each fold (compare apples to apples)

Model selection: random search then grid search

- For each model:
  - Narrow down the best hyperparameter region using a 100-iteration randomized search
- Select the best model according to the metrics *
- Define a refined and with lower cardinality hyperparameter grid around the parameters of the best model
- Run a grid search on the model with the previous grid to fine-tune it even further
- Best of both worlds:
  - efficiency of randomized search
  - accuracy of grid search

* What if different metrics lead to different best models? Fortunately not the case here, as there was always a consensus between metrics.

Model selection: random search then grid search

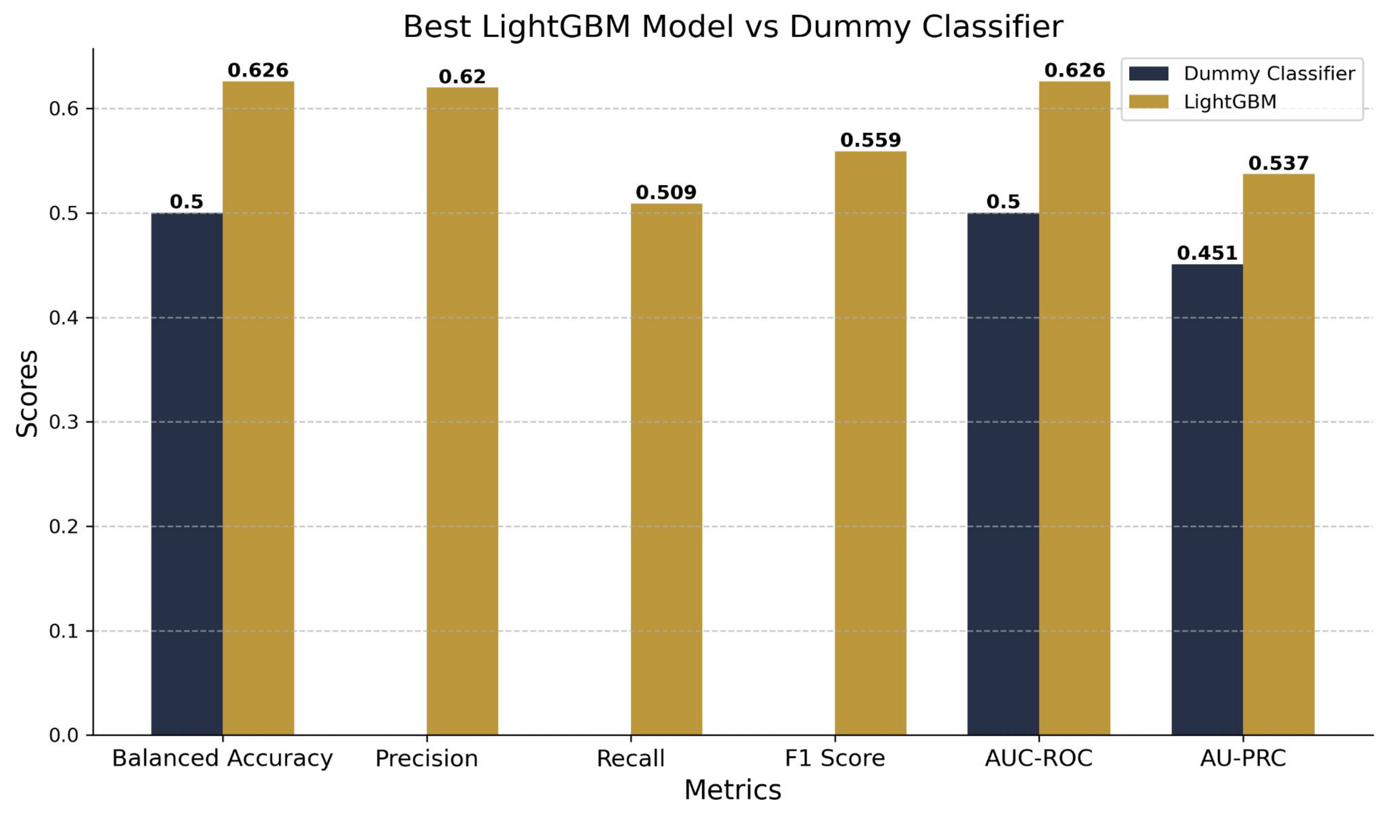


Machine learning models explored

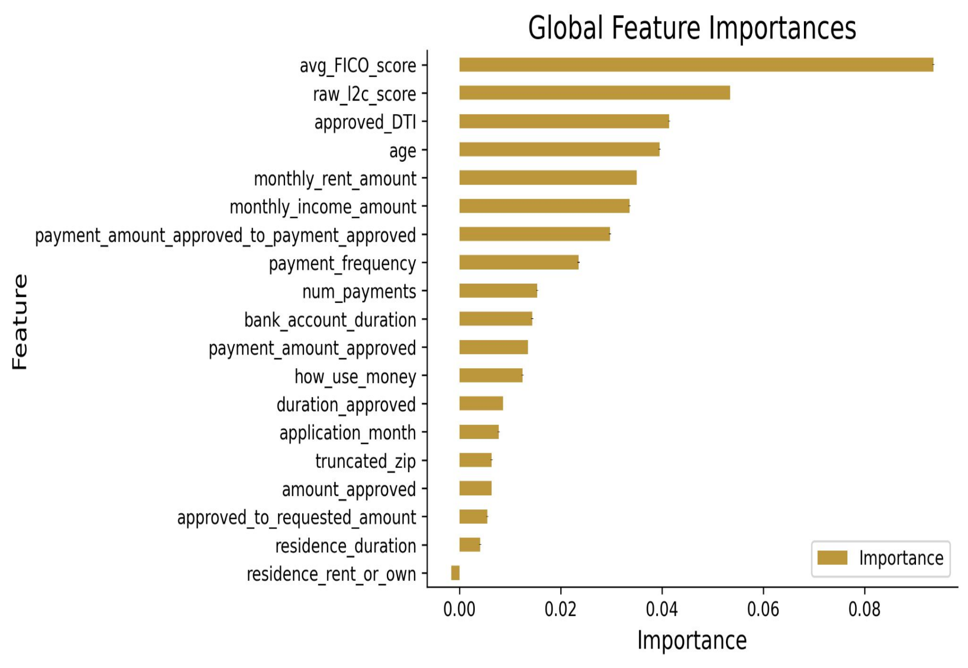
- Dummy classifier (always predicts the majority class): serves as a **baseline**
- Logistic regression
- Tree based models:
  - Decision Tree
  - Random Forest
  - Gradient boosting:
    - XGBoost
    - LightGBM
    - CatBoost

Results & Feature Importances

Results


Best LightGBM Model vs Dummy Classifier
Dummy Classifier
LightGBM
Scores
Metrics
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Balanced Accuracy
Precision
Recall
F1 Score
AUC-ROC
AU-PRC
0.5
0.626
0.62
0.509
0.559
0.5
0.626
0.451
0.537

Global Feature importances



- Method: **Permutation importance**
- **Relative** importances (not absolute)
- **Global** importances: most important features **overall**, not locally
- Most important features:
  - FICO score, L2C score, DTI, age, monthly rent amount, monthly income amount
- Least important features:
  - Residence rent or own, residence duration, approved to requested amount
- Careful with interpretation because of **correlated features** and **poor performances**

Final predictions (extract)

	predicted_probability	predicted_label	true_label
50	0.611983	1	1
51	0.595950	0	0
52	0.821817	0	0
53	0.694141	1	1
54	0.702674	0	0
55	0.659986	0	1
56	0.731026	1	1
57	0.672661	1	1
58	0.653270	1	1
59	0.720309	1	0

Conclusions

Challenges & Limitations

- Biggest challenge: **making the most out of such a small dataset**
- Lack of feature description:
  - Some variables were not self-explanatory, and thus I had to assume several things (e.g., residence_rent_or_own, or difference between payment amount and payment amount approved)

Recommendations & Next steps

- **Collect more data** and re-train the models
  - If no additional data are available, use data augmentation techniques to generate synthetic data
- **Discuss with domain experts** to gain knowledge about feature importances in credit underwriting
- Try other feature engineering techniques (e.g., cyclical encoding of time-related features)
- Use other feature importances methods, such as:
  - L2 logistic regression coefficients
  - Local methods for explaining individual predictions: **SHAP, LIME**

Thank you for your attention!
Questions?