ML-Driven Loan Prediction

July 22, 2022

Overview

Revamping and de-risking the process of loan-lending using Machine Learning Models

Machine Learning to Improve Credit Models

Business Problem

How to improve the process of lending credit to borrowers?

Conventional Approach

- Credit scores
  - FICO
  - L2C
- Income

Present/Future

Machine Learning based models using conventional credit scores and more!

Objective:
Create a Machine Learning Model to Approve a Loan Application

Understanding the Data

Data

- Application data for every customer that has been given a loan in a 6 months period.
  - 648 Applications
  - 31 Loan Features
  - No target variable assigned
- Loan Performance data over the same period
  - 1286 Applications
  - Target variable : Good Loan or Bad Loan

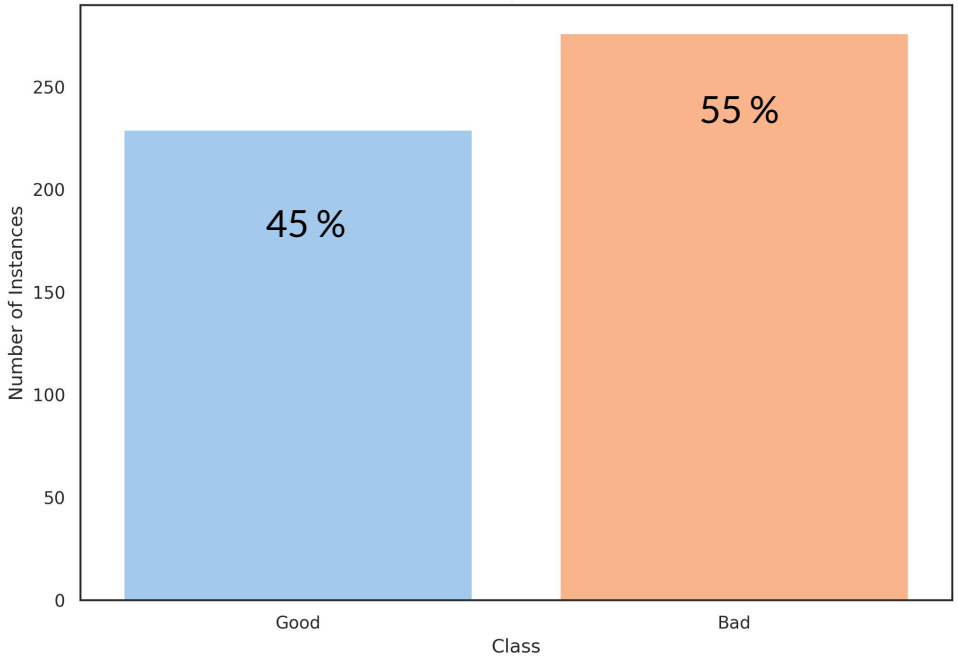
Assumptions

- Data Sources:
  - Credit scores are reliable and from agencies
  - Features come from applicant-filled questionnaires
- Features:
  - Feature durations are in months
  - Feature and Approved-Feature : requested by the applicant and approved by the bank
- Only one loan per applicant

Data Wrangling

- Primary Key (idLoan) is cleaned
  - Duplicate keys found, removed keys with mixed data (Good & Bad)
- Merged the data on Primary Key
- Data shape - (631 x 31)
- Data-types: object, int, float, bool, datetime

Class Balance



Feature Engineering

- Split the data to training and testing data to avoid data leakage when engineering features.
- Missing values
  - Bank account duration, money usage have a few missing values → Mode of the training data used to fill values. (Same for test data)
  - Other Phone Type has many missing values → 'NA' added as a new category.
  - Approved Payment Type is numerical, so median value is used.

Feature Engineering

New features!

- routing number → Bank name (Python API)
- address zip → County name (Python API)
- birth date and application date → Age of borrower
- application date → Year, month, hour, and day of week
- email address → Email provider (string manipulation)
- rent & income → Monthly income balance

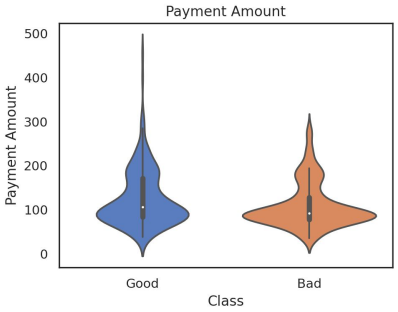
Dropped features

- status
- customer_id
- payment_ach
- routing number
- birthdate
- zip code
- email
- application date
- num_payments

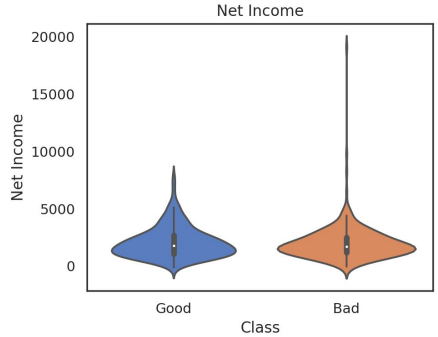
Feature Engineering

Categorical Encoding & Feature Scaling

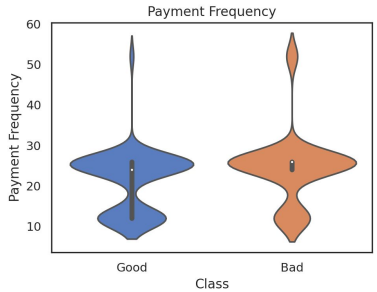
- Features with ordinal valued categories were mapped manually, such as bank account duration & loan duration
- Boolean features were encoded as 0 and 1
- Scikit-learn's Ordinal Encoder used to encode the rest
- Feature Scaling: Robust Scaler
- Outlier treatment for numerical features

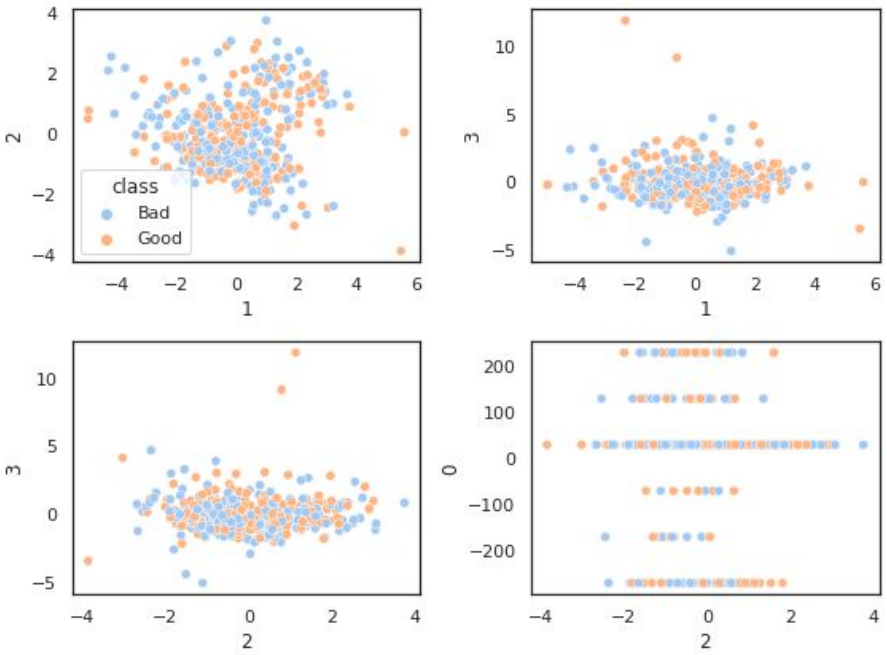


Feature Engineering: Visualization



- Significant feature overlap for both classes
- No clear separation
- High dimensional visualization using dimensionality reduction?



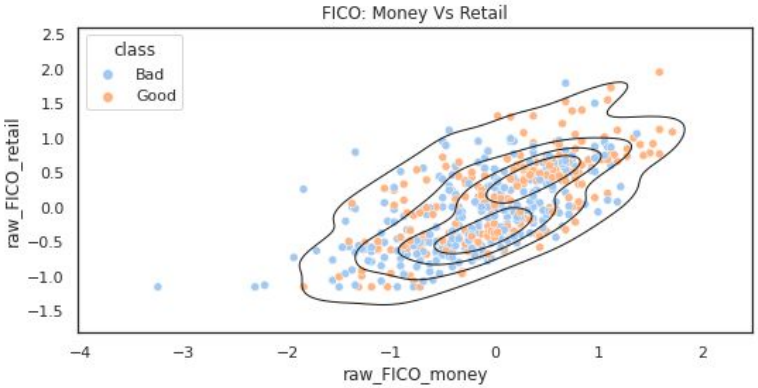


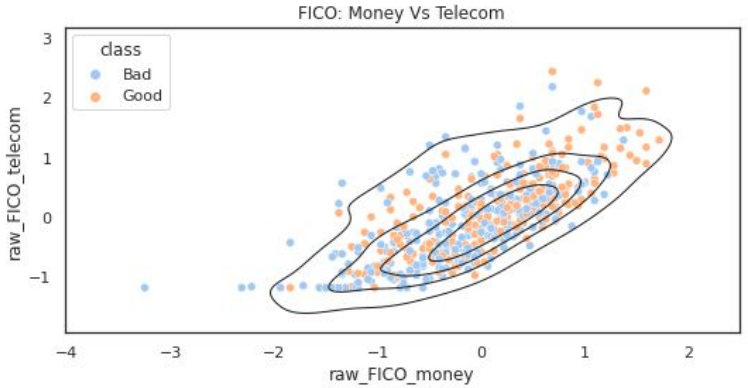
Visualizing the Low Dimensional Data

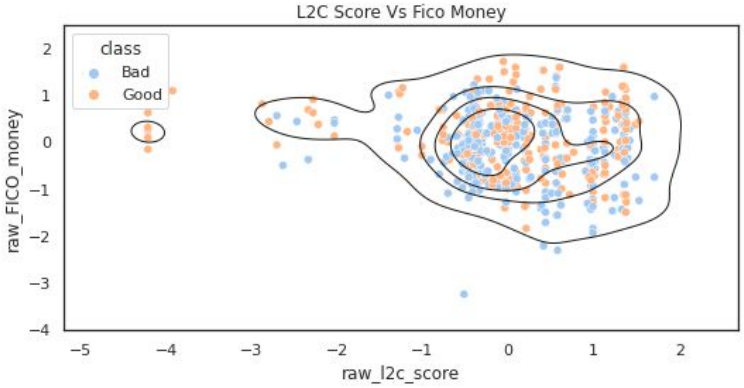
PCA Dimensionality Reduction

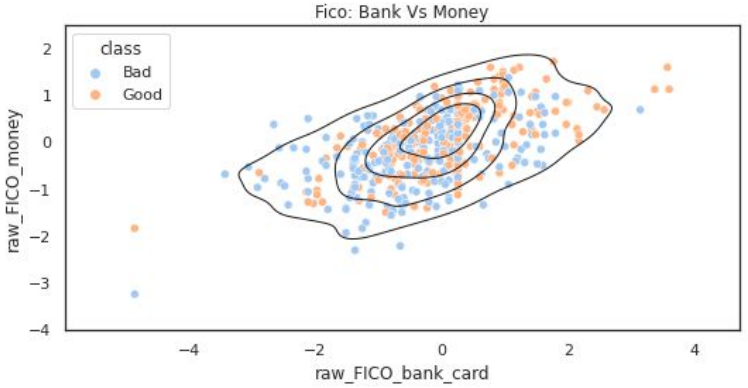
No clear separation of classes

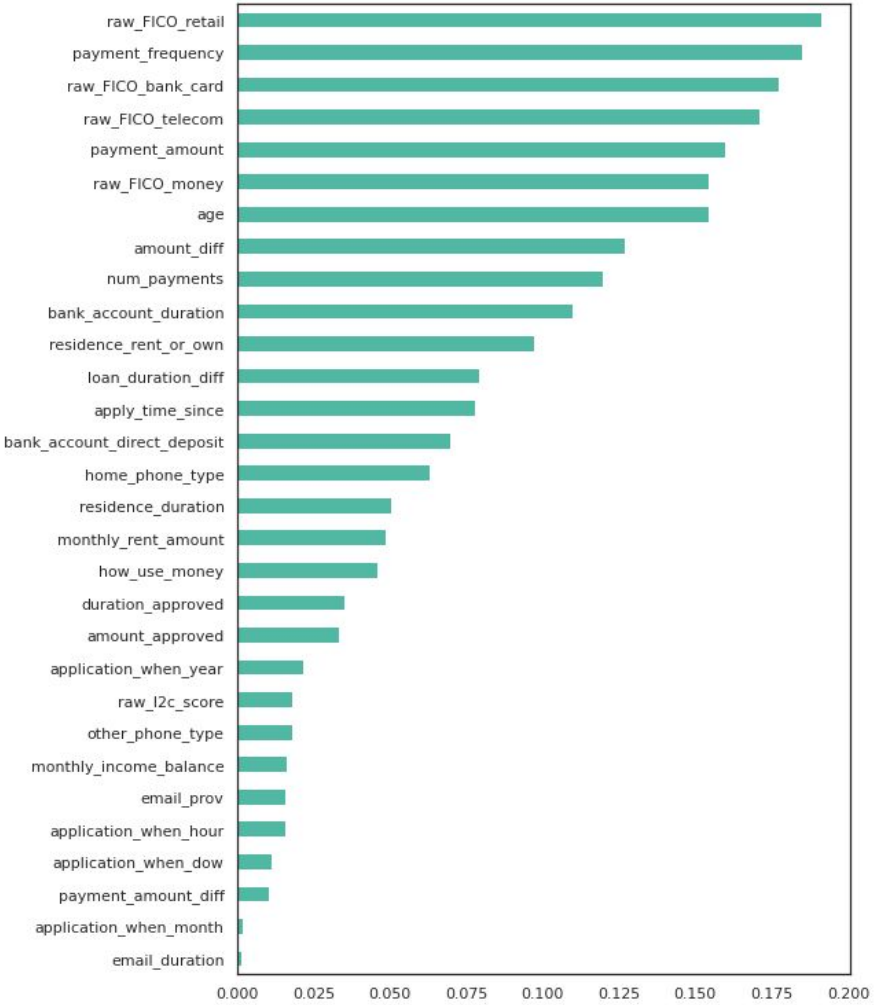
Bivariate Distributions of Credit Scores





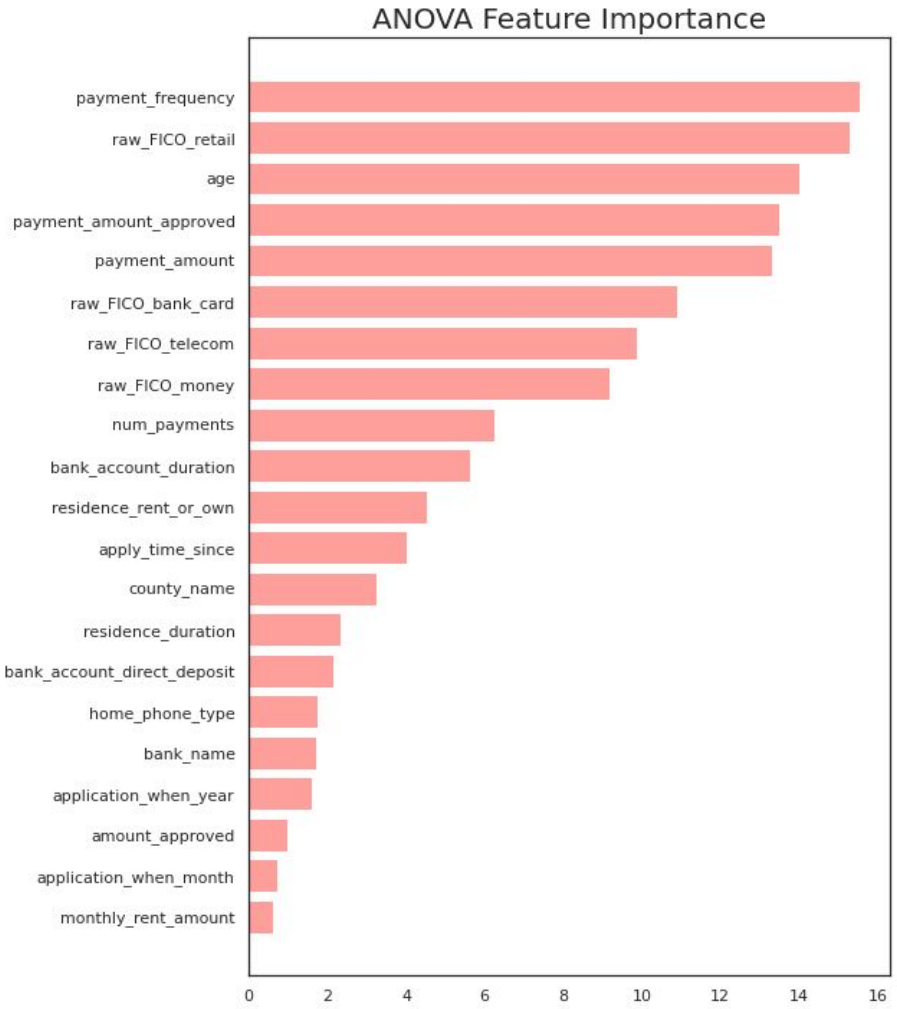






Feature Correlations to Target

Pearson Correlation



Feature Importance

ANOVA F-value

ML Model Building

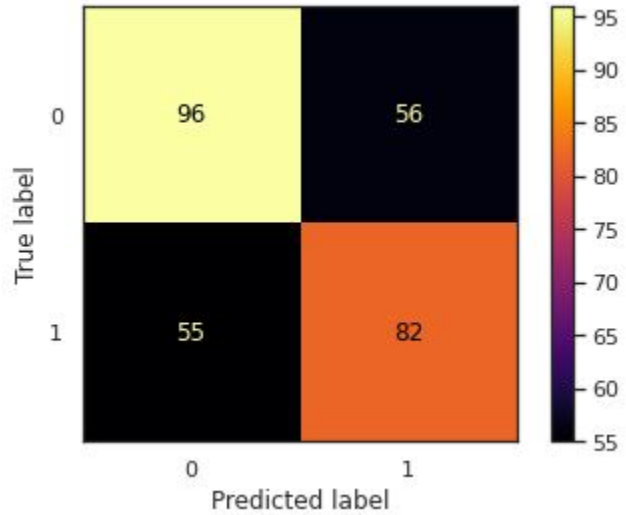
Tabular data bodes well for a tree-based algorithm

- Training 80%, Testing 20%
- Stratified Cross-validation (10-folds)
- Random Forest for a preliminary check of performance
- EXtreme Gradient Boosted Trees - a.k.a. XGBoost - for hyper-parameter tuning → better performance!
- Parameter grid (GridSearchCV)

Model Performance on Test data

ML Model	ROC-AUC	F1
Random Forest	**0.65**	0.57
XGBoost	0.62	0.58
XGBoost Tuned	0.62	**0.60**

XGBoost Confusion Matrix



Parameters: *n_estimators*, *eta*, *max_depth*, *min_child_weight*, *subsample*, *colsample_bytree*, *gamma*

A Counterfactual Case Study

- What if we only use the credit scores for the model ? (Simulation)
- Using unconventional features increases the performance.

	Counterfactual (F1)	ML Model (F1)	Lift (F1)
Random Forest	0.50	0.57	14%
XGBoost	0.51	0.58	14%
Tuned Model	0.51	0.60	18%

Conclusion & Introspection

- Tree-based ML Model outperforms the conventional model by 18%.
- Extensive feature engineering did not improve the model
- More data and addition of novel features will be key!
- Choose a different F-measure: based on the client's need!
- Build an MVP model and set a benchmark before augmenting and feature engineering significantly.

Future Direction

Ensemble Model	• Random Forest and XGBoost performed comparatively for different metrics (F1 & AUC) • An stacked model could take the best out of both
Productionizing	• End-point creation using Cloud (Sagemaker/AWS) • Monitoring performance using dashboards, setting alerts, and continuous improvements

Thank you!