OeNB Industry Lab - Documentation

Tristan Leiter

November 14, 2025

Contents

Chapter 1

Exploratory Data Analysis

1.1 Dependent Variable Default

In this credit risk dataset, the target variable, y, is inherently imbalanced. The number of non-defaults (0) significantly outnumbers the number of defaults (1). This is a common and expected characteristic of credit risk data.

This imbalance poses a significant challenge for model development. If we were to use a simple random split to create our training, validation, and test sets, we would face a high risk of creating unrepresentative samples. For example, a small test set could, purely by chance, end up with a much higher or lower percentage of defaults than the original dataset or, in the worst case, zero defaults.

1.1.1 Data imbalance

Below, we can observe the magnitude of the imbalance in the dataset. Table 1 shows that less then 1% of all observations in the dataset include defaults.

Table 1.1: Proportion of Defaulting vs. Non-Defaulting Firms

Default Status	**Number of Firms**	**Proportion**
1 (Default)	2,096	0.008576
0 (No Default)	242,305	0.991424
Total	**244,401**	**1.000000**

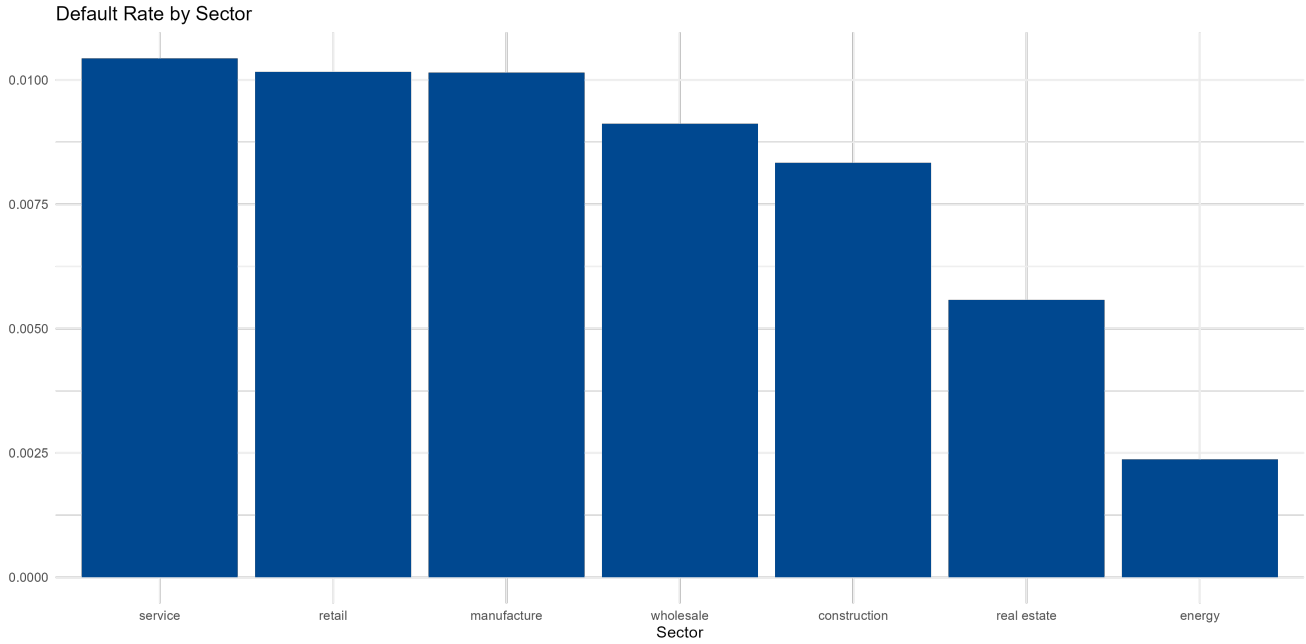

Figure 1.1: Defaulting Firms

1.1.2 Implication

To address potential sampling bias from the severe data imbalance, we employ **stratified sampling**. This technique ensures that the original class distribution of the target variable is preserved in both the training and test sets.

The primary advantages of this approach are:

- **Guaranteed Representation:** Stratification forces the data splits to maintain the original ratio of defaults to non-defaults. Given that 0.0865% of the original dataset consists of defaults, both the training and test sets will contain approximately this same percentage.
- **Reliable Evaluation:** When the test set is truly representative, the performance metrics we calculate (such as accuracy, precision, recall, and F1-score) are meaningful. Evaluating a model on a test set with a skewed default rate would give a misleading and overly optimistic (or pessimistic) performance score.
- **Improved Model Generalization:** Training a model on a set that accurately reflects the real-world data distribution helps it learn the distinct patterns of both the majority (non-default) and minority (default) classes. This leads to a more robust and generalizable model.

In summary, using stratified sampling is a critical step to ensure our model is trained and evaluated on a reliable, representative foundation.

1.1.3 Theory

The dataset is partitioned into a training set and a testing set using the `rsample` package, which is part of the tidymodels framework. This process ensures that the resulting datasets are representative of the original data's class distribution, which is a critical step in handling data imbalance.

The used `initial_split` function does not simply take a random 80% of the data. When the `strata = y` argument is used, it follows a more intelligent **stratified sampling** algorithm:

1. **Group Data by Stratum:** The function first identifies all unique values in the specified `strata` column (`y`). In this case, it creates two distinct groups: one for all the defaulting firms (`y = 1`) and another for all the non-defaulting firms (`y = 0`).
2. **Sample Proportionally within Each Group:** The function then takes an 80% sample from the defaulting firms group and a separate 80% sample from the non-defaulting firms group.
3. **Create the Training Set:** These two separately sampled subsets (80% of defaults and 80% of non-defaults) are combined to form the final training set (`Train`).
4. **Create the Test Set:** The remaining 20% of the defaulting firms and the remaining 20% of the non-defaulting firms are combined to form the final testing set (`Test`).

1.2 Distributions and bivariate Data Analysis

1.2.1 Distributions

1.2.2 Data dependency

1.3 Feature selection

Before building the predictive model, it is essential to assess the predictive power of each independent variable. This helps in eliminating irrelevant features, understanding variable relationships, and focusing on the most promising candidates. We use the Information Value (IV), significance tests and evaluation of multicollinearity for this purpose.

1.3.1 Informational Value

The function follows a systematic statistical procedure to assign a single score—the Information Value—to each variable, quantifying its ability to separate the "good" outcomes (non-defaults, y=0) from the "bad" ones (defaults, y=1). Here is the step-by-step process:

1. **Binning (Grouping):** The function first discretizes each continuous independent variable into a set of bins or groups (e.g., by quantiles). For categorical variables, each category is treated as a separate group.
2. **Counting Goods and Bads:** For each bin of a variable (e.g., for the "Age: 20-30" bin), the algorithm counts:
   - The number of non-defaults.
   - The number of defaults.
3. **Calculating Proportions:** It then calculates the proportion of the total goods and total bads that fall into that specific bin:
   - **% Goods** = (# Goods in the bin) / (Total # Goods in the entire dataset)
   - **% Bads** = (# Bads in the bin) / (Total # Bads in the entire dataset)
4. **Calculating Weight of Evidence (WoE):** The WoE for each bin measures how much the evidence in that bin supports one outcome over the other. The formula is:

   $$\text{WoE} = \ln\left(\frac{\%\text{Goods}}{\%\text{Bads}}\right)$$

   A large positive WoE means the bin is strongly associated with non-defaults, while a large negative WoE means it is strongly associated with defaults.
5. **Calculating Information Value (IV):** Finally, the total IV for the entire variable is calculated by summing a weighted value across all its bins. The formula is:

   $$\text{IV} = \sum(\%\text{Goods} - \%\text{Bads}) \times \text{WoE}$$

   This single number represents the total predictive power of the variable.

Why This is Useful Before Modeling

Exploring the IV summary is not just a formality; it is a critical step in the exploratory data analysis (EDA) phase for several reasons:

- **Efficient Feature Selection:** IV provides a simple, powerful metric to rank all variables by their predictive strength.
  - **IV $<$ 0.02:** Useless predictor.
  - **0.02 to 0.1:** Weak predictor.
  - **0.1 to 0.3:** Medium predictor.
  - **0.3 to 0.5:** Strong predictor.
  - **IV $>$ 0.5:** Suspiciously high; may indicate data leakage or a variable that is too good to be true.

- **Understanding Variable Relationships:** While the total IV gives a summary, the detailed WoE for each bin of a variable reveals the nature of its relationship with the outcome. For example, by looking at the WoE for different age groups, you can see if the default risk increases, decreases, or follows a non-linear pattern as age changes.
- **Data Quality and Sanity Checks:** The IV calculation can expose data issues. A variable with an unexpectedly high IV might be a "leaky" variable (e.g., a variable that is only known *after* the default has occurred, pointing to a problematic quasi-seperation issue). Similarly, strange WoE patterns can highlight potential data entry errors or anomalies that need investigation.

1.3.2 Signifiance Tests

1.3.3 Multicollinearity

After evaluating the individual predictive power of each variable using Information Value, the next critical step is to examine the relationships between the predictor variables themselves. **Multicollinearity** occurs when two or more independent variables are highly correlated with each other.

While multicollinearity does not necessarily reduce the overall predictive accuracy of a model, it poses a significant problem for interpretation. When variables are highly correlated, it becomes difficult for the model to distinguish their individual impacts on the target variable. This can lead to:

- Unstable coefficient estimates that can change dramatically with small changes in the data.
- High standard errors for the coefficients, making them seem statistically insignificant even when they are not.
- Difficulty in interpreting the model's logic, as the effect of one variable is confounded by the effect of another.

From the correlation matrix, several pairs of variables exhibit strong linear relationships (correlation > 0.70), which is a clear indicator of multicollinearity.

The analysis reveals two primary groups of highly inter-correlated variables:

1. **Group 1: `f1`, `f2`, `f6`, and `f11`**
   - The correlation between `f2` and `f1` is extremely high at **0.92**.
   - `f6` is also highly correlated with both `f1` (0.79) and `f2` (0.74).
   - `f11` is highly correlated with `f1` (0.74) and `f2` (0.66).

   These variables likely measure a similar underlying economic or financial concept. Including all of them in a model would be redundant and problematic.

2. **Group 2: `f8` and `f9`**
   - The correlation between `f8` and `f9` is exceptionally high at **0.91**.

   These two variables are nearly interchangeable from a linear perspective.

Our strategy will be to select only one representative variable from each of these groups to proceed with modeling. The best candidate for selection is typically the variable with the highest Information Value (IV), as it holds the most individual predictive power.

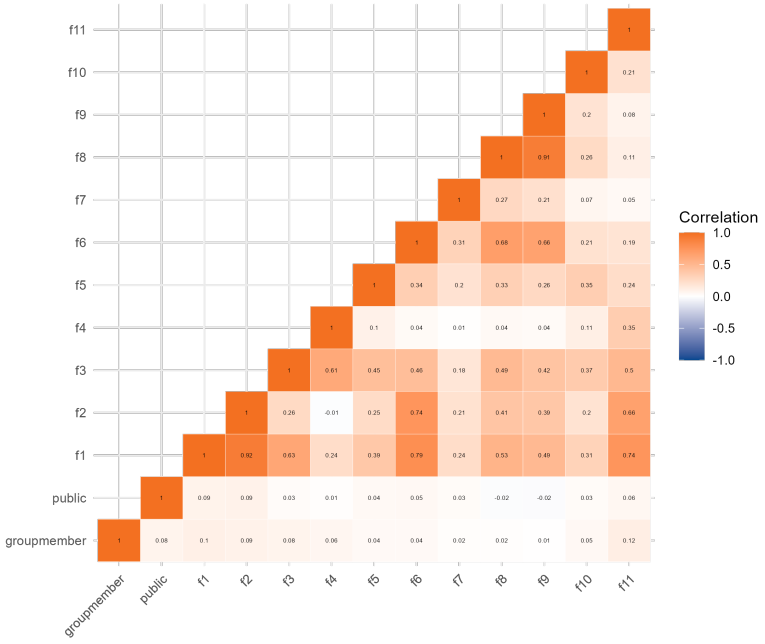

Figure 1.2: Multicollinearity

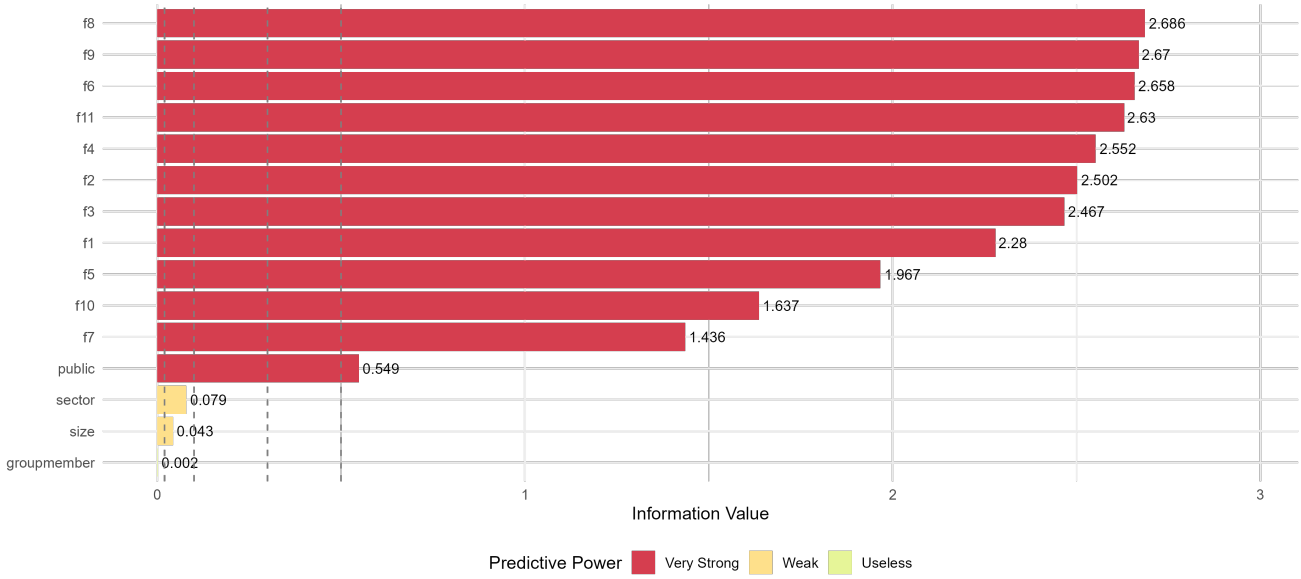

Figure 1.3: IV per feature

1.3.4 Implication

IV per feature:

IV and multicollinearity:

Table 1.2: Variable Selection Summary: IV and Multicollinearity

Variable	Information Value (IV)	Multicollinearity Concern	Action
Highly Correlated Group 1			
f6	2.658	High (Group 1)	Keep (Highest IV in Group)
f11	2.630	High (Group 1)	Remove
f2	2.502	High (Group 1)	Remove
f1	2.280	High (Group 1)	Remove
Highly Correlated Group 2			
f8	2.686	High (Group 2)	Keep (Highest IV in Group)
f9	2.670	High (Group 2)	Remove
Variables with Low Multicollinearity			
f4	2.552	Low	Keep
f3	2.467	Low	Keep
f5	1.967	Low	Keep
f10	1.637	Low	Keep
f7	1.436	Low	Keep
public	0.549	Low	Keep
sector	0.079	Low	Keep
size	0.043	Low	Keep
Variable with Low Predictive Power			
groupmember	0.002	Low	Remove (Useless IV)

“‘

1.4 Implementation

1.4.1 Data splitting

In our methodology, we partition the data into an 80% training set and a 20% final hold-out test set. All model development, including hyperparameter tuning and feature selection, is performed on the 80% training set using a technique called **k-fold cross-validation**. This approach is methodologically superior to a simpler three-way split (e.g., 50% train, 25% validation, 25% test) for several critical reasons.

Superiority Over a 50/25/25 Split

1. **Maximizes Data for Model Training:** The most significant drawback of a 50/25/25 split is that the model is only ever trained on 50% of the available data. In contrast, with an 80/20 split and k-fold cross-validation, the model is repeatedly trained on a much larger portion of the data (e.g., in 5-fold CV, it trains on 80% of the 80% training set, which is 64% of the total data, and this is done 5 times). More training data allows the model to learn more complex patterns and generalize better, reducing the risk of underfitting.

2. **Robust and Stable Performance Estimation:** In a 50/25/25 split, model performance is evaluated and tuned based on a single, static 25% validation set. The results on this single set can be highly variable depending on the "luck of the draw" of that particular split. A different random split could lead to different hyperparameter choices and a misleading performance estimate.

   Cross-validation solves this problem. By creating multiple (k) validation sets from the training data and averaging the performance metrics across all k folds, it provides a much more stable and reliable estimate of the model's true performance. It mitigates the risk of overfitting to a single validation set, leading to a more robust final model.

3. **More Efficient Use of Data:** With cross-validation, every single observation in the 80% training set is used as part of a validation set exactly once. This ensures that the entire training dataset is leveraged for both training and validation, making the process highly data-efficient. In a 50/25/25 split, the 25% validation set is never used for training, and the 50% training set is never used for validation, effectively "wasting" data from the perspective of each task.

Summary of Comparison

The following table provides a direct comparison of the two approaches.

Table 1.3: Comparison of Data Splitting Strategies

Aspect	**80/20 Split with Cross-Validation**	**50/25/25 Train/Validation/Test Split**
Training Data Size	Large (e.g., 64%-72% of total data used in each CV fold).	Small (fixed at 50% of total data).
Performance Estimate Reliability	**High.** Performance is averaged over multiple validation sets, leading to a stable and robust estimate.	**Low to Medium.** Performance depends on a single, potentially biased validation set.
Risk of Overfitting to Validation Set	**Low.** It is difficult to overfit to multiple, distinct validation sets simultaneously.	**High.** Hyperparameters can be easily tuned to perform well on this specific 25% set by chance.
Data Efficiency	**Very High.** Every observation in the training set is used for both training and validation across the folds.	**Medium.** Data is strictly segregated; the validation set is never learned from.
Computational Cost	Higher, as the model must be trained k times.	Lower, as the model is only trained once per hyperparameter set.

In conclusion, while a simple three-way split is faster, the 80/20 split combined with cross-validation is the modern gold standard. It produces more robust, reliable, and better-performing models by using the available data more effectively, which is a trade-off well worth the additional computational cost.

1.5 Outliers

1.5.1 Overview

1.6 Feature engineering

1.6.1 Overview

Appendix A

Overview