Advanced Programming 2025

Bankruptcy Prediction Using Machine Learning

Final Project Report

Tschumi Richard
richard.tschumi@unil.ch
Student ID: 2095165530

December 20, 2025

Abstract

This project develops a machine-learning pipeline to forecast U.S. firm bankruptcies one year ahead using accounting information only, while explicitly addressing extreme class imbalance and temporal constraints. Using a panel dataset of publicly listed U.S. firms from 1995 to 2023, several models are evaluated, including Logistic Regression, Random Forest, and XGBoost, in both Baseline and Tuned configurations.

The methodology emphasizes time-consistent train–test separation, feature selection through recursive feature elimination, and imbalance handling via class weighting and controlled oversampling. Model performance is assessed using metrics suited to rare-event prediction, including AUC, precision–recall curves, and yearly backtesting. Results show that tree-Based models substantially outperform Logistic Regression in discriminatory power, with XGBoost (Base) achieving an AUC of 0.92 and providing the best balance between predictive performance and temporal stability. Hyperparameter tuning and aggressive rebalancing do not systematically improve out-of-sample results and may increase false positives.

Interpretability is ensured through the combined use of Feature Importance and SHAP values, enabling an economically coherent analysis of the main drivers of bankruptcy risk. Overall, the project delivers a decision-oriented pipeline suitable for early-warning and screening applications.

Keywords: data science, Python, machine learning, Logistic Regression, Random Forest, XGBoost, bankruptcy prediction, financial distress, imbalanced classification, SHAP

Contents

1 Introduction **3**

2 Literature Review / Related Work **3**
2.1 Previous Approaches to Bankruptcy Prediction 3
2.2 Machine Learning Approaches and Relevant Methodologies 4
2.3 Datasets Used in Related Studies 4

3 Methodology **4**
3.1 Data Description 4
3.1.1 Company distribution 5
3.1.2 Features selection 5
3.2 Approach 5
3.2.1 System architecture 5
3.2.2 Preprocessing 6
3.2.3 Class Imbalance 6
3.2.4 Algorithms Base and Tuned 7
3.2.5 Evaluation metrics 7
3.3 Implementation and Key code components 7

4 CodeBase and Reproducibility **8**

5 Results **8**
5.1 Experimental Setup 8
5.2 Performance Evaluation 9
5.3 Visualizations 9

6 Discussion **13**
6.1 Challenges 13
6.2 Comparison with Initial Expectations 13
6.3 Limitations 13

7 Conclusion and Future Work **14**
7.1 Summary 14
7.2 Future Directions 14

References **15**

A Additional Figures **16**

B Additional Tables **18**

C Code Repository **19**

D AI Utilisation **19**

1 Introduction

Predicting corporate bankruptcy has long been a critical topic in finance, risk management, and regulatory supervision. Accurate early-warning systems allow investors, creditors, and policy-makers to identify financially distressed firms before failure materializes, thus reducing losses and improving capital allocation. With the growing availability of large-scale financial datasets and the progress of machine learning techniques, the development of robust predictive models has become increasingly feasible and valuable.

My motivation for this project builds on my earlier academic work. During my Bachelor thesis, I investigated financial risks in private, non-listed firms, focusing primarily on the qualitative drivers that lead companies toward failure. While this research highlighted important behavioral, managerial, and governance-related mechanisms behind distress, it also made clear how difficult it is to access reliable quantitative data for private firms. This project, therefore, represents a natural continuation of my previous research: by shifting to publicly available U.S. listed-firm data, I aim to explore a more quantitative and data-driven approach to bankruptcy prediction.

Forecasting U.S. Firm Bankruptcy at Horizon T+1 : The core problem addressed in this study is the development of a machine-learning pipeline capable of detecting firms that are likely to fail within the next year, using only financial statement information.

More specifically, the goals of this project are:

- To construct a clean, consistent dataset of U.S. firms spanning 1995–2023, including raw accounting variables and derived financial ratios.
- To design a principled preprocessing pipeline addressing missing values, outliers, and temporal train–test separation.
- To develop and compare Baseline and Tuned machine-learning models (Logistic Regression, Random Forest, XGBoost), incorporating feature selection, class weighting, and controlled SMOTE oversampling.
- To evaluate the models using metrics suited to rare-event prediction, including AUC, precision-recall curves, and yearly backtesting.
- To analyze model behavior through SHAP values and Feature Importance to identify which financial indicators contribute the most to bankruptcy risk.

2 Literature Review / Related Work

2.1 Previous Approaches to Bankruptcy Prediction

Early research on corporate bankruptcy prediction relied primarily on statistical models Based on financial ratios. Altman's Z-Score [1] represents the seminal contribution, using discriminant analysis to classify firms through a linear combination of accounting variables. While historically influential, this approach relies on strong distributional assumptions that are often violated in financial data.

Ohlson [2] later introduced Logistic Regression, which relaxes many of these assumptions and became a standard tool for modeling bankruptcy probabilities. Subsequent work by Zmijewski [3] highlighted remaining issues such as sample selection bias and sensitivity to model specification. A major methodological advance was proposed by Shumway [4], who demonstrated that static cross-sectional models underestimate bankruptcy risk by ignoring its dynamic nature.

By framing bankruptcy as a time-dependent process using panel data, Shumway established temporal modeling as a key requirement for credible prediction.

Overall, early studies emphasize that bankruptcy is a rare and evolving event, requiring both appropriate statistical treatment and a temporal perspective.

2.2 Machine Learning Approaches and Relevant Methodologies

With increasing computational power, machine learning methods have gained prominence in bankruptcy prediction. Early applications of tree-Based models [5] relaxed linearity assumptions, while ensemble methods such as Random Forest [6] and gradient boosting models like XGBoost [7] further improved predictive performance by capturing complex nonlinear relationships.

A central challenge in this literature is extreme class imbalance, as bankruptcies represent only a small fraction of firm-year observations. To address this, researchers have proposed class-weighted learning [8], synthetic oversampling techniques such as SMOTE [9], and evaluation metrics better suited to rare events, including ROC and precision–recall analysis.

Model validation has also evolved. Earlier studies often relied on random cross-validation, which can introduce temporal leakage. Bergmeir and Benítez [10] emphasized the need for time-consistent validation schemes, reinforcing earlier findings that respecting temporal structure is essential for reliable bankruptcy prediction.

2.3 Datasets Used in Related Studies

A central reference point for empirical bankruptcy research is the long historical panel assembled by Shumway, which covers U.S. publicly listed firms from 1962 to 1992 using data from Compustat and CRSP [4]. This dataset provides harmonized annual accounting information together with market-Based variables such as excess returns and return volatility, and it documents firm exits with precise identifiers for bankruptcy and delisting events.

Building on this foundation, later studies such as Campbell, Hilscher and Szilagyi expanded the Compustat and CRSP panels to include additional financial ratios and market indicators, including measures of size, momentum and short-term volatility. These extended datasets typically span several decades, from 1963 to 2003 [11], and provide a sufficiently broad cross-section of firms and economic conditions to study variations in financial distress across industries and time periods.

3 Methodology

3.1 Data Description

The dataset used in this project originates from Compustat North America [12] and covers U.S. publicly listed firms between 1995 and 2023. After loading the raw file, the pipeline applies strict validation and filtering rules to enforce consistency in firm-year reporting, including consolidation filters and removal of duplicated observations at the firm-year level. These steps reduce the initial 138,975 rows to 115,194 unique (firm, year) pairs, as documented in the loading module.

Although the period spans nearly three decades, not all firms report every year. These temporal irregularities are handled directly in the data loading module, which sorts observations by firm and by year and applies a structured imputation procedure. Missing financial items are first forward filled and then backward filled within each firm. Remaining gaps are treated in a status dependent manner: for firms that eventually fail, unresolved values are replaced with the cross sectional median, while for firms that remain active, observations that still contain missing values are removed. This approach preserves information near distress while ensuring internal consistency of the panel. After imputation, the dataset contains 74 934 observations and no remaining missing values.

3.1.1 Company distribution

The target variable is constructed so that a firm receives a value of 1 in the last year it appears before failure, and 0 otherwise. This results in 74,175 non-failed observations (0.9899) and 759 failures (0.0101), confirming extreme class imbalance consistent with corporate default data. The distribution is illustrated in the following Figure 1.

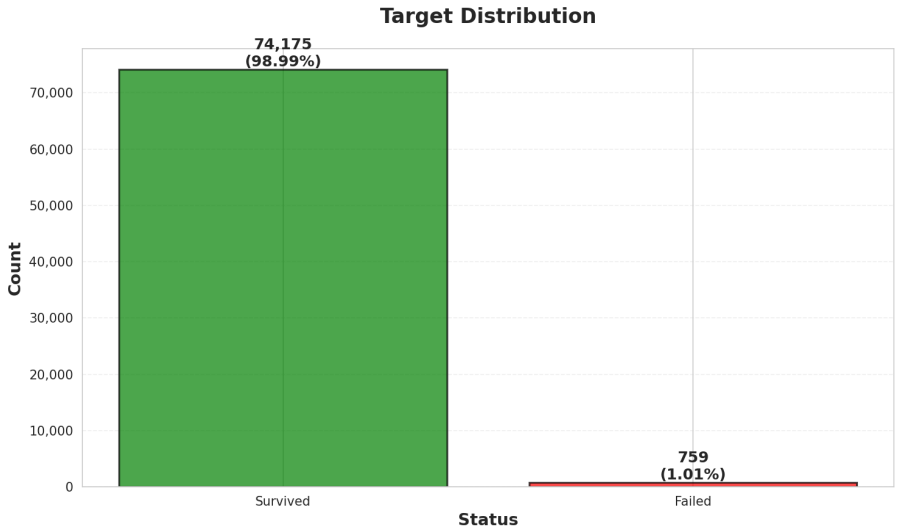


Figure 1: Target distribution

3.1.2 Features selection

The dataset contains 35 financial predictors, including 20 raw financial statement variables (e.g., assets, liabilities, cash flow items, leverage components) and 15 accounting ratios that are directly computed during dataset preparation. No market-Based variables are included.

To reduce redundancy and limit the high dimensionality typical of financial ratio datasets, the modeling pipeline applies Recursive Feature Elimination (RFE) with an XGBoost estimator, selecting the 20 most informative features, Table 5. These include most important features identified automatically from the initial 35-feature set.

In addition to this selection mechanism, a correlation heatmap, Figure 9, is examined to characterise associations among the retained predictors. The purpose of this analysis is interpretative rather than restrictive. Models such as Random Forest and gradient boosting are naturally robust to collinearity, and no further exclusion of variables is imposed on this basis.

3.2 Approach

3.2.1 System architecture

The pipeline follows a modular architecture organised around five sequential stages, each encapsulated in a dedicated Python module. Figure 2 illustrates the data flow and module dependencies.

```
ml_bankruptcy_prediction/
├── main.py                    # Pipeline orchestrator
├── requirements.txt           # Python dependencies
├── environment.yml            # Conda environment specification
├── data/                      # Input data and intermediate outputs
│   └── us-bkty-final.csv      # Raw Compustat dataset
├── outputs/
│   ├── models/                # Serialised trained models (.pkl)
│   ├── plots/                 # Diagnostic figures and SHAP plots
│   ├── logs/                  # Execution logs
│   └── reports/               # Hyperparameter reports
└── src/                       # Source modules
    ├── config.py              # Centralised configuration
    ├── data_features.py       # Data loading and feature engineering
    ├── preprocessing.py       # Cleaning, splitting, scaling, SMOTE
    ├── models.py              # Model training (base + tuned)
    ├── evaluation.py          # Metrics, curves, backtesting
    └── eda_interpretation.py  # EDA and SHAP analysis
```

Figure 2: Project architecture

3.2.2 Preprocessing

After the construction of the analytical dataset described in Section 3.1, the panel is prepared for estimation through a sequence of transformations designed to preserve temporal integrity and ensure comparability of predictors across firms and years. The sample is divided chronologically, with all observations from 1995 to 2014 forming the training period and all observations from 2015 to 2023 forming the testing period. This split prevents any information from later years affecting the estimation of the models.

All numerical variables are then standardized using scaling parameters estimated exclusively on the training period and subsequently applied to the test period. This procedure is required for Logistic Regression and improves comparability across variables with different magnitudes, while remaining neutral for tree Based models that are invariant to monotonic transformations.

3.2.3 Class Imbalance

Given the extremely low incidence of bankruptcy, class imbalance is addressed through two complementary approaches. First, each model incorporates class weights that penalize misclassification of failed firms according to their relative scarcity. Second, for the Tuned model specifications, the minority class in the training set is augmented through synthetic oversampling with SMOTE, increasing its share to approximately two and a half percent while maintaining the chronological separation between training and testing samples.

Step	n_samples	n_failures	failure_rate (%)
Data	74 934	759	1.01
Train Set	46 473	535	1.15
After SMOTE (Train)	47 086	1 148	2.44
Test Set	28 461	224	0.79

Table 1: Failures before and after preprocessing/SMOTE

3.2.4 Algorithms Base and Tuned

Three classes of models are estimated in two versions each, yielding a total of six specifications. The Base models consist of Logistic Regression, Random Forest and XGBoost with conservative hyperparameters (Table 6) and without synthetic oversampling. These models represent benchmark configurations intended to illustrate the predictive capacity of the algorithms under minimal tuning.

The Tuned models employ the same three algorithms but incorporate hyperparameter optimisation (Table 7), carried out with the RandomizedSearchCV functionality from Scikit-learn combined with a TimeSeriesSplit cross-validation strategy to preserve temporal ordering, class weighting and SMOTE. This design allows a direct comparison between unoptimised and optimised versions of each method and enables an assessment of the incremental value of model complexity. Logistic Regression provides a linear reference point, whereas Random Forest and XGBoost capture a broad range of nonlinear effects that are relevant in financial distress prediction.

Hyperparameter optimisation is conducted using a randomized search procedure combined with time-series cross validation, Table 8. Each validation fold consists of a training window that chronologically precedes its corresponding validation window, thereby preventing any future data leakage. Once the optimal parameters are identified, the Tuned models are re-estimated on the full training set, including SMOTE oversampling and class weighting, and are subsequently evaluated on the untouched test period.

3.2.5 Evaluation metrics

Model performance is assessed through a set of evaluation metrics that are appropriate for rare event prediction and consistent with the structure of the dataset. The principal metric is the area under the receiver operating characteristic curve. This measure evaluates the ability of a model to discriminate between failed and non failed firms across all possible classification thresholds and is used as the objective function during cross validation and hyperparameter optimisation.

Complementary metrics are reported to provide additional insight into classification performance in an imbalanced setting. Precision, recall and the F1 score are computed on the test period to quantify the trade off between false positives and false negatives. Confusion matrices are generated for each model to display directly the distribution of correct and incorrect classifications.

An overfitting analysis is also conducted by comparing training and testing performance. A model is considered overfitted when the gap between the training and testing accuracy or AUC exceeds a threshold of five percent.

Precision–recall curves and ROC-curves are produced for all Base and Tuned models. These visualizations summarise the balance between sensitivity and specificity and allow a direct comparison between algorithms and tuning strategies.

To examine performance over time, yearly values of the AUC are computed for each model in the test period. This measure of temporal stability assesses whether predictive accuracy is concentrated in specific market conditions or sustained across years.

3.3 Implementation and Key code components

The empirical pipeline is implemented entirely in Python using a modular architecture that separates data construction, preprocessing, model estimation, evaluation and interpretation into distinct components. This structure allows the workflow to remain transparent, reproducible and easy to maintain while ensuring that each stage of the analysis can be run independently.

Data Loading and Feature Engineering (data_features.py) The raw Compustat extract is filtered to retain only standardized annual (datafmt='STD') and consolidated statements

(consol='C'). Duplicate firm-year entries are removed. The module then constructs all financial ratios used in the analysis and applies the missing-data strategy described in Section 3.1. The one-year-ahead bankruptcy indicator (fail_next_year) is generated by shifting the failure flag forward by one year, ensuring that prediction always uses information strictly prior to the failure event.

Preprocessing (preprocessing.py) The cleaned dataset is split chronologically: 1995–2014 for training, 2015–2023 for out-of-sample testing. Numerical predictors are standardized (z-score) for Logistic Regression using training data statistics, while tree-Based models use raw values. Class weights are set inversely to the failure class frequency to address class imbalance. For Tuned models, SMOTE is applied to the training set to augment the minority class, while the test set remains strictly out-of-sample and temporally ordered.

Model Training (models.py) Three classifiers are used: Logistic Regression (L2), Random Forest and XGBoost. Base models use default hyperparameters and class weighting on the original data. Tuned models are optimized via RandomizedSearchCV with TimeSeriesSplit cross-validation to preserve temporal order. Before modeling, Recursive Feature Elimination (RFE) with XGBoost selects the top 20 predictors to reduce dimensionality and improve stability.

Evaluation (evaluation.py) Model performance is evaluated on the test set using AUC-ROC, precision, recall, and F1-score (threshold 0.5). Diagnostics include ROC and PR curves, confusion matrices, cross-model comparison plots and a metrics table. Yearly AUCs (2015–2023), allowing detection of performance sensitivity to market conditions. Overfitting is checked by comparing training and test accuracy and AUC, highlighting models with large generalization gaps.

Interpretation (eda_interpretation.py) The module generates exploratory plots (target distribution, failure incidence over time, predictor correlations). Model interpretation uses Feature Importance for Random Forest and XGBoost, and coefficients for Logistic Regression. SHAP values are also computed for ensemble models. Both measures provide global feature rankings and local attributions that clarify the drivers of predicted bankruptcy risk.

Pipeline Orchestration (main.py) The main execution script coordinates all five stages of the pipeline in a fully automated sequence. It manages configuration parameters, directs data flow between modules, records detailed logs to both console and file, and stores all intermediate objects and final outputs in a structured directory.

4 CodeBase and Reproducibility

The complete CodeBase for this project is publicly available at `https://github.com/RichT09/project-tschumi`, ensuring reproducibility of all results. The repository includes the full modular pipeline together with configuration settings defined in config.py and the raw dataset, which must be placed in the data/ directory. All dependencies can be installed via pip install -r requirements.txt, and the entire experiment can be executed from the command line 1 using python main.py, with optional flags 2 to skip SHAP, EDA modules or to override the default split year.

5 Results

5.1 Experimental Setup

- **Hardware:** All computations were executed in a virtualized environment provided by the Nuvolos cloud platform. Development was carried out using Visual Studio Code 1.102.1.

- **Software:** The project was implemented in Python 3.13.5 using the following core libraries: NumPy 1.24+, pandas 2.0+, SciPy 1.10+, scikit-learn 1.3+, XGBoost 2.0+, imbalanced-learn 0.11+, matplotlib 3.7+, seaborn 0.12+, SHAP 0.42+, tqdm 4.65+ and

joblib 1.3+. Dependency management and exact versioning are documented in the project's `requirements.txt` file.

- **Hyperparameters:** Full hyperparameter values, Baseline, Tuned and Search Grids are reported in Tables 6, 7 and 8.

5.2 Performance Evaluation

Table 2 reports the main performance indicators for each model. XGBoost (Base) achieves the highest out-of-sample AUC (0.9248), followed by Random Forest (Base) (0.9137). Logistic Regression models perform worse (AUC $\simeq 0.8300$). Logistic Regression yields the highest recall (0.8616–0.8973) but very low precision (0.0189–0.0204), indicating many false signals. Tree-Based models offer a better balance: Random Forest (Base) has precision 0.0720 and recall 0.6652, while XGBoost (Base) has precision 0.0636 and recall 0.7589. These differences are consistent with the flexibility of tree-Based methods in capturing nonlinearities and interactions among financial variables

Tuned models do not systematically outperform their Base counterparts. XGBoost (Tuned) exhibits both lower AUC and lower recall than XGBoost (Base). Random Forest (Tuned) achieves higher recall but with lower precision and AUC. These results indicate that aggressive class rebalancing via SMOTE, combined with time-series cross-validation, does not guarantee improved out-of-sample performance for bankruptcy prediction.

Model	**AUC**	**F1** ($y = 1$)	**Precision** ($y = 1$)	**Recall** ($y = 1$)
Logistic Regression (Base)	0.8294	0.0369	0.0189	0.8616
Logistic Regression (Tuned)	0.8359	0.0399	0.0204	0.8973
Random Forest (Base)	0.9137	0.1300	0.0720	0.6652
Random Forest (Tuned)	0.9052	0.0819	0.0433	0.7723
XGBoost (Base)	0.9248	0.1174	0.0636	0.7589
XGBoost (Tuned)	0.9085	0.1002	0.0538	0.7232

Table 2: Performance metrics on the 2015–2023 test period (positive class = bankruptcy, $y = 1$)

As anticipated, Table 3 shows that the Tuned Tree-Based models present clearer signs of overfitting. Because accuracy is unreliable in highly imbalanced settings, the AUC gap offers a better indication of generalisation. Both Random Forest (Tuned) and XGBoost (Tuned) exceed the 0.05 AUC gap threshold, signalling reduced out-of-sample robustness. This outcome is consistent with the combined effect of hyperparameter optimisation, SMOTE oversampling and class-weighting, which tends to amplify overfitting.

Model	**Acc Gap**	**AUC Gap**	**Overfit (Gap $>$ 0.05)**
Logistic Regression (Base)	-0.0161	0.0151	NO
Logistic Regression (Tuned)	-0.0211	0.0100	NO
Random Forest (Base)	0.0122	0.0049	NO
Random Forest (Tuned)	0.0183	0.0530	YES
XGBoost (Base)	0.0158	0.0380	NO
XGBoost (Tuned)	0.0168	0.0755	YES

Table 3: Train–test performance gaps used for overfitting assessment

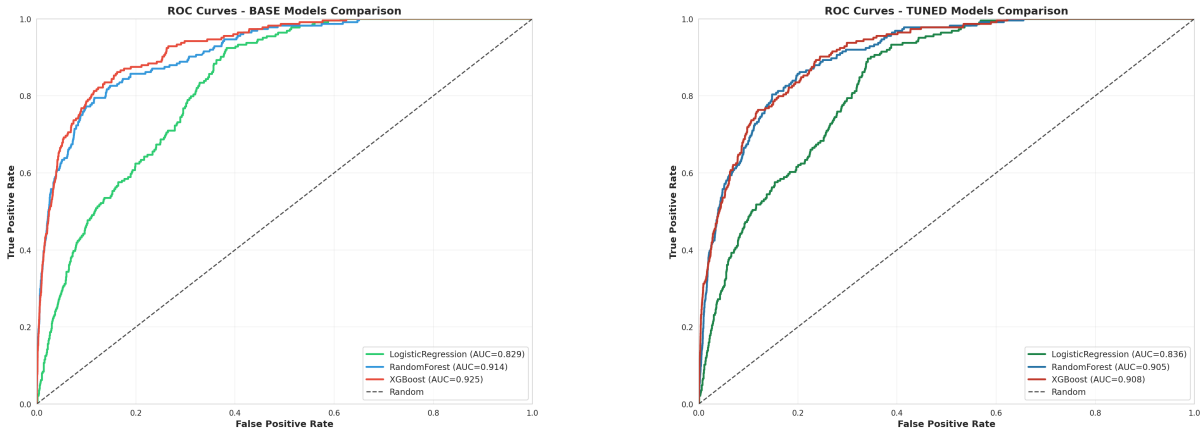
5.3 Visualizations

The confusion matrices (Figure 3) report that Logistic Regression, both Base and Tuned, are highly aggressive: they assign higher probabilities to the minority class (bankruptcies) to maxi-

mize recall, which can increase both true positives and false positives. Random Forest (Base) is more conservative, missing a significant portion of bankruptcies (75) but producing fewer false positives (1,920). Random Forest (Tuned) and XGBoost (Tuned) detect well failures (173 and 162) but also show inflated false-positive counts (3,827 and 2,847), consistent with their overfitting patterns. XGBoost (Base) offers the strongest balance, identifying (170) true positives while limiting false positives to (2,501).

Figure 3: Confusion Matrices for Baseline and Tuned models

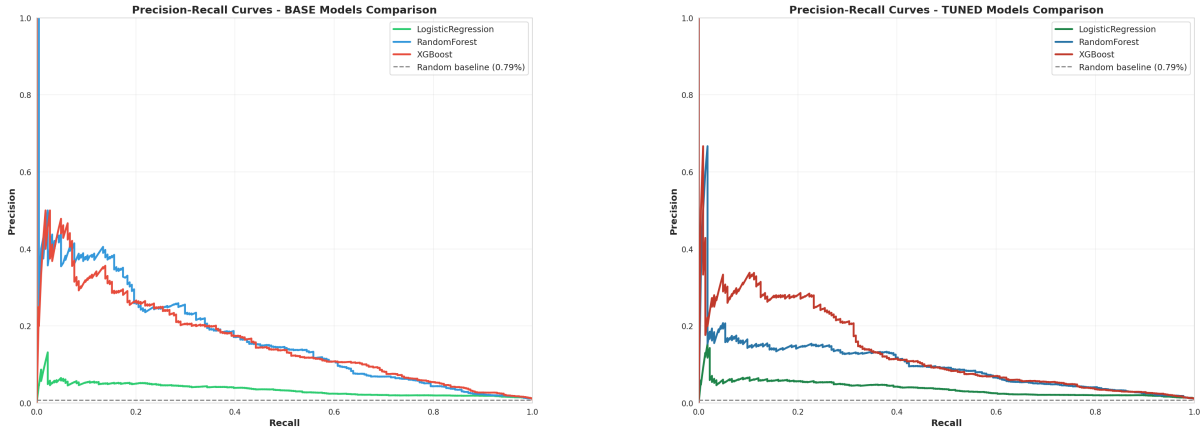
Figure 4 shows that XGBoost (Base) provides the strongest discrimination between bankrupt and non-bankrupt firms, achieving the highest AUC (0.925), followed closely by Random Forest (Base) (0.914), while Logistic Regression (Base) displays substantially weaker separation (0.829). After tuning, both XGBoost and Random Forest exhibit slightly lower AUC values (0.908 and 0.905), indicating a modest trade-off in overall ranking performance, whereas Logistic Regression (Tuned) improves marginally (0.836) but remains clearly dominated by the tree-Based models.



(a) Baseline Models (b) Tuned Models

Figure 4: ROC curves for Baseline and Tuned models

Figure 5 highlights the impact of severe class imbalance on model performance. XGBoost achieves the highest precision–recall trade-off in both its Base and Tuned versions, particularly at low to intermediate recall levels, where precision remains clearly above the random Baseline. Random Forest performs slightly below XGBoost, with precision declining more rapidly as recall increases, while Logistic Regression remains close to the Baseline across most of the recall range. As recall approaches one, precision converges toward zero for all models, reflecting the intrinsic difficulty of bankruptcy detection in highly imbalanced data.



(a) Baseline Models (b) Tuned Models

Figure 5: Precision–Recall curves for Baseline and Tuned models

Overall, the ROC and PR visualisations corroborate the quantitative metrics: XGBoost (Base) provides the best balance between recall and manageable false-positive rates, while Tuned tree-Based models show improved recall but at the cost of noisier precision and reduced generalisation.

Since XGBoost (Base) is the best model, it is natural to rely on its outputs to identify the variables that contribute most strongly to bankruptcy prediction. Two complementary importance measures are used: Feature Importance, which reflects how the model structurally relies on each variable for its decision splits, and SHAP values, which quantify each feature's marginal effect on predicted bankruptcy risk. Together, they provide a comprehensive view of feature relevance from both model-level and prediction-level perspectives, and the final ranking in Table 4 is derived accordingly.

Feature	**Feature Importance**	**SHAP**	**Combined**	**Effect**
1. int_coverage	0.1614	0.0174	0.0894	Negative
2. roa_approx	0.1357	0.0253	0.0805	Positive
3. ocf	0.0804	0.0384	0.0594	Negative
4. quick_ratio	0.0613	0.0437	0.0525	Negative
5. ebit	0.0615	0.0361	0.0488	Negative
6. inventories	0.0649	0.0287	0.0468	Negative
7. shareholder_equity	0.0567	0.0355	0.0461	Negative
8. current_assets	0.0392	0.0386	0.0389	Negative
9. retained_earnings	0.0630	0.0098	0.0364	Negative
10. interest_exp	0.0382	0.0273	0.0328	Positive

Table 4: Top features affecting bankruptcy risk using XGBoost (Base)

The interpretation of feature effects follows a simple convention: a negative effect indicates that higher values of the variable are associated with a lower predicted probability of bankruptcy, while a positive effect indicates the opposite. Under this convention, interest coverage exhibits a clear negative effect, indicating that firms with a stronger ability to service interest payments face a substantially lower bankruptcy risk. Operating cash flow (OCF) also shows a negative effect, highlighting the protective role of internally generated liquidity in mitigating financial distress.

ROA, by contrast, presents a more ambiguous relationship. Although high ROA values are frequent and associated with risk-reducing effects, the overall relationship is dominated by a smaller number of very low ROA observations that generate large positive SHAP contributions (Figure 6). These extreme distress cases drive the aggregated positive effect. Among the top features, ROA is the only variable whose interpretation is not monotonic.

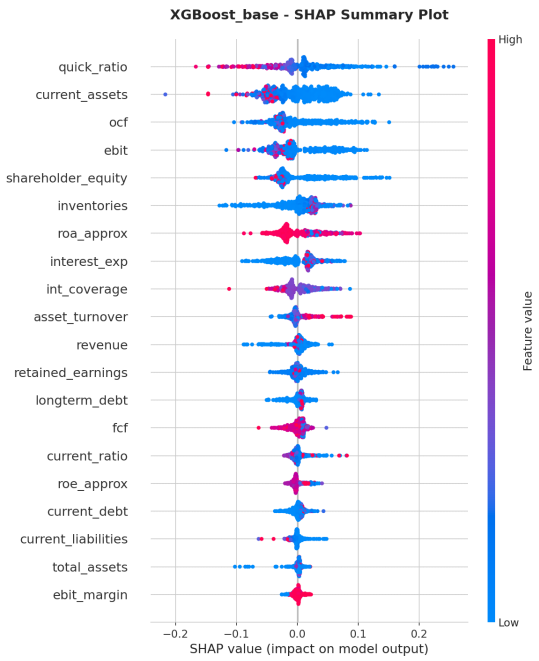


Figure 6: SHAP summary for the XGBoost (Base)

Because financial distress risk is sensitive to macroeconomic cycles and market conditions, a single aggregate AUC may conceal important variation over time. Figure 7 summarizes the yearly AUC values for each model across the nine test years.

Overall, tree-Based models exhibit the highest temporal stability. Random Forest (Base) achieves a mean yearly AUC of 0.9103 with a standard deviation of 0.0286, while XGBoost (Base) attains a slightly higher mean AUC of 0.9176 with a comparable level of variability (0.0286). Both models consistently outperform Logistic Regression across all years.

Tuned models exhibit a modest reduction in temporal stability relative to their Base counterparts, particularly for XGBoost, consistent with the fact that oversampling and hyperparameter optimization accentuate sensitivity to minority patterns, which may fluctuate across business cycles.

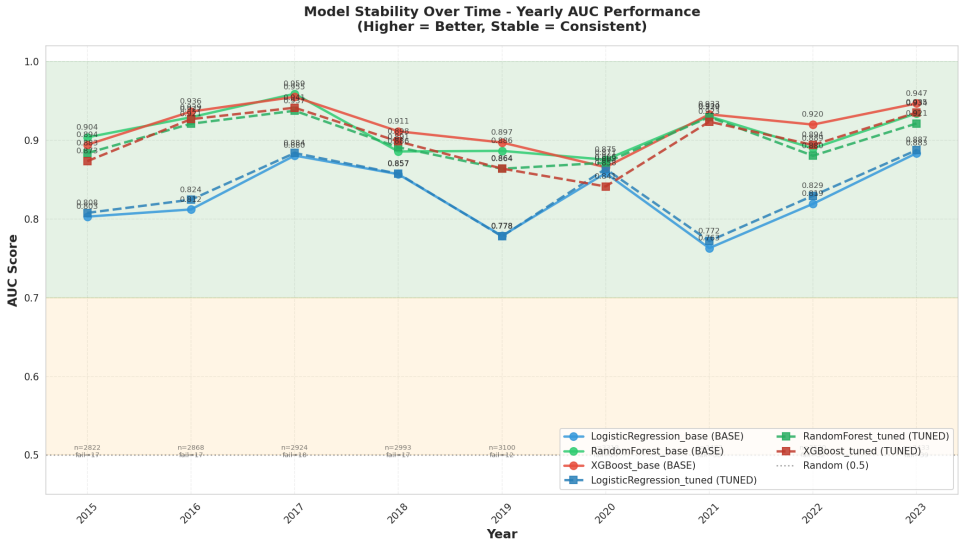


Figure 7: Yearly AUC Performance

6 Discussion

XGBoost and Random Forest, achieve the strongest discriminatory performance in terms of AUC and clearly outperform Logistic Regression, confirming their ability to capture complex relationships among financial ratios. These tree-Based models also exhibit satisfactory temporal stability, with relatively consistent performance across test years despite changing economic conditions. In addition, the strict respect of the temporal structure of the data ensures a realistic evaluation and prevents information leakage. Finally, model interpretability constitutes a key strength of the approach: the combined analysis of Feature Importance and SHAP values provides an economically coherent interpretation of the drivers of bankruptcy risk.

6.1 Challenges

During training, class imbalance is addressed using class_weight or scale_pos_weight. In highly imbalanced settings, these mechanisms can overly emphasize the minority class, leading to aggressive decision boundaries with many false positives. To mitigate this effect, an fp_weight_multiplier $= 0.45$ is applied to reduce cost asymmetry in the loss function, yielding a more balanced trade-off between bankruptcy detection and false alarms while retaining cost sensitivity.

SMOTE should be applied with caution, particularly when class-weighting mechanisms are already used during model training. By artificially altering the distribution of the training data, excessive oversampling of bankrupt firms can lead models to learn a distorted representation of reality. In settings where bankruptcies are structurally rare in the underlying population, overly aggressive SMOTE configurations may cause models to overestimate default risk and generate an excessive number of false alarms.

In the test phase, several decision-threshold optimization strategies were explored (including F1-score maximization, recall- or precision-constrained thresholds and Youden's J), but they did not yield meaningful improvements in the overall trade-off. These approaches generally led to a substantial increase in false positives for only marginal gains in bankruptcy detection. Consequently, the default threshold of 0.5 is retained in order to limit false alarms while maintaining an acceptable level of true positive detection.

6.2 Comparison with Initial Expectations

Overall, the observed results are consistent with initial expectations. The superiority of nonlinear models is empirically confirmed, with XGBoost and Random Forest outperforming Logistic Regression in terms of discriminatory power. The effects of hyperparameter tuning vary across models, leading to either performance gains or losses in stability, which reflects the trade-off between optimization and generalization. The behavior of the Precision–Recall curves is consistent with the rarity of bankruptcy events, as precision declines rapidly as recall increases. Finally, cost-sensitive learning improves bankruptcy detection at the expense of a higher risk of false alarms, requiring careful calibration.

6.3 Limitations

Despite the achieved performance, the proposed approach presents several limitations. It relies primarily on historical accounting data, which may respond with delay to rapid financial deterioration, and the absence of dynamic or lagged variables limits the detection of early distress signals. The use of a single decision threshold may also be suboptimal across different operational contexts. In addition, SMOTE-Based oversampling introduces artificial training observations that may deviate from the true bankruptcy distribution. Finally, the lack of external validation and the exclusion of macroeconomic factors restrict the generalizability of the results to other periods or economic environments.

7 Conclusion and Future Work

7.1 Summary

This study analyzes bankruptcy prediction in a context of severe class imbalance and temporal constraints using several machine learning models. The results confirm the superiority of nonlinear approaches: XGBoost and Random Forest outperform Logistic Regression in terms of discriminatory power, with XGBoost (Base) offering the best trade-off between predictive performance and temporal stability. The analysis also highlights that class imbalance handling must be carefully calibrated, as overly aggressive corrections can degrade precision without proportional gains in bankruptcy detection, while the choice of the probability threshold remains a key lever for balancing bankruptcy detection and false alarms. The yearly evaluation underscores the importance of temporal validation, with tree-Based models exhibiting relative stability in the face of economic fluctuations, and the combined analysis of Feature Importances and SHAP values provides an economically coherent interpretation of the main drivers of bankruptcy risk.

Overall, the study results in a reproducible, decision-oriented pipeline. Although the number of threads was fixed to limit sources of randomness, XGBoost may still exhibit minor variations across operating systems and hardware configurations.

7.2 Future Directions

Several directions for future research can be considered. The introduction of dynamic or lagged variables, the adoption of adaptive probability thresholds defined according to decision contexts and error costs, and improvements in probabilistic calibration could enhance the operational relevance of the models. Access to more precise and comprehensive data, including a larger number of bankruptcy observations while preserving realistic population representativeness, as well as validation on external datasets, sectoral analyses, and comparisons with survival-type models, constitute natural extensions. Finally, the developed models are intended to be used as screening or early warning tools, in complement to human analysis.

References

[1] Edward I. Altman. "Financial Ratios, Discriminant Analysis and the Prediction of Corporate Bankruptcy". In: *The Journal of Finance* 23.4 (1968), pp. 589–609.

[2] James A. Ohlson. "Financial Ratios and the Probabilistic Prediction of Bankruptcy". In: *Journal of Accounting Research* 18.1 (1980), pp. 109–131.

[3] Mark E. Zmijewski. "Methodological Issues Related to the Estimation of Financial Distress Prediction Models". In: *Journal of Accounting Research* 22 (1984), pp. 59–82.

[4] Tyler Shumway. "Forecasting Bankruptcy More Accurately: A Simple Hazard Model". In: *The Journal of Business* 74.1 (2001), pp. 101–124.

[5] Halina Frydman, Edward I. Altman, and Duen Li Kao. "Introducing Recursive Partitioning for Financial Classification". In: *The Journal of Finance* 40.1 (1985), pp. 269–291.

[6] Leo Breiman. "Random Forests". In: *Machine Learning* 45.1 (2001), pp. 5–32.

[7] Tianqi Chen and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2016, pp. 785–794.

[8] Charles Elkan. "The Foundations of Cost-Sensitive Learning". In: *International Joint Conference on Artificial Intelligence (IJCAI)* (2001), pp. 973–978.

[9] Nitesh V. Chawla et al. "SMOTE: Synthetic Minority Over-sampling Technique". In: *Journal of Artificial Intelligence Research* 16 (2002), pp. 321–357.

[10] Christoph Bergmeir and Jose M. Benitez. "On the Use of Cross-Validation for Time Series Predictor Evaluation". In: *Information Sciences* 191 (2012), pp. 192–213.

[11] John Y. Campbell, Jens Hilscher, and Jan Szilagyi. "In Search of Distress Risk". In: *The Journal of Finance* 63.6 (2008), pp. 2899–2939.

[12] Wharton Research Data Services. *CRSP/Compustat Merged (CCM) Database*. Accessed: 2025-10-01. 2025. URL: `https://wrds-www.wharton.upenn.edu/pages/get-data/compustat-capital-iq/ccm/`.

A Additional Figures

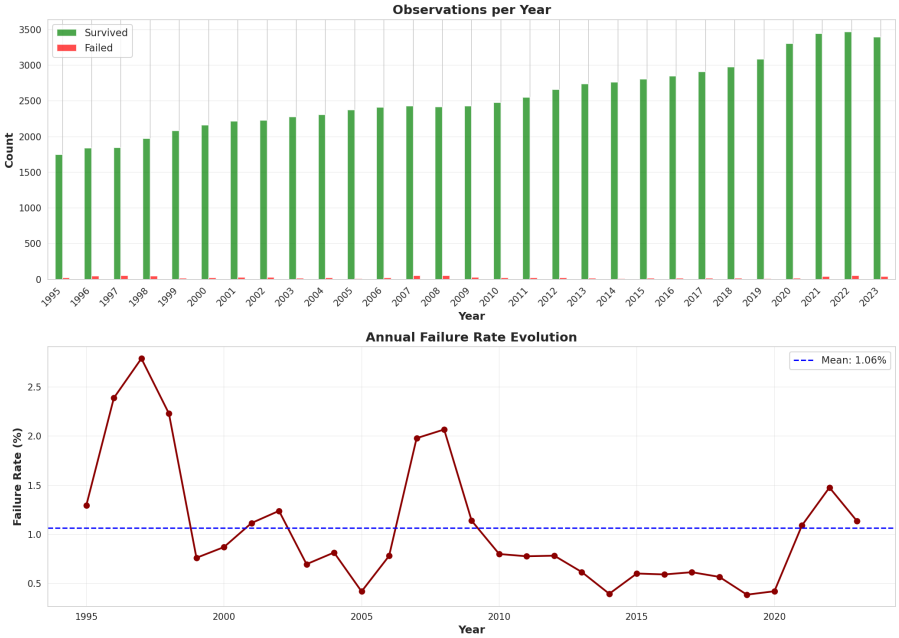


Figure 8: Temporal distribution

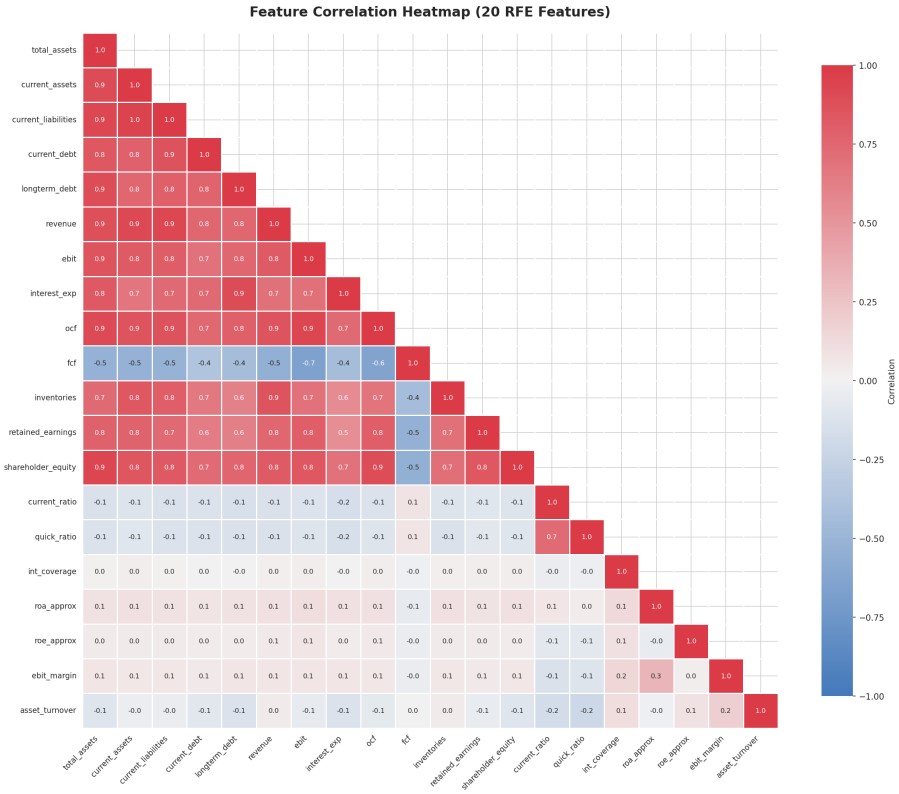


Figure 9: Correlation heatmap

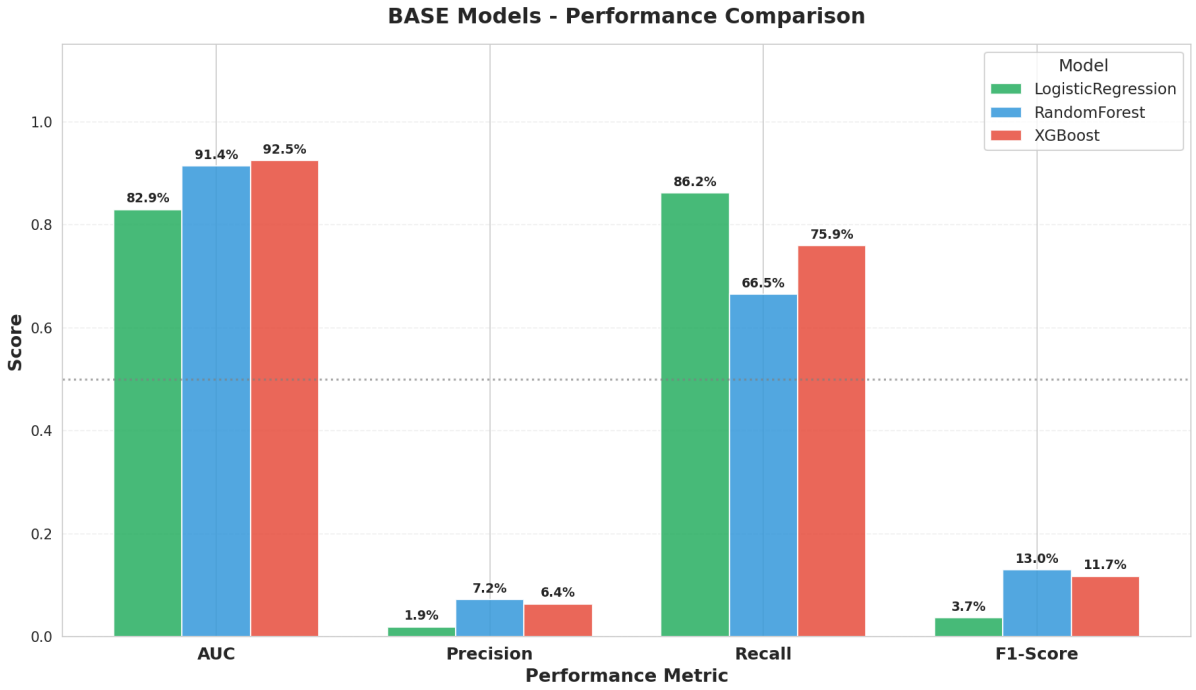


Figure 10: Model comparison (Base)

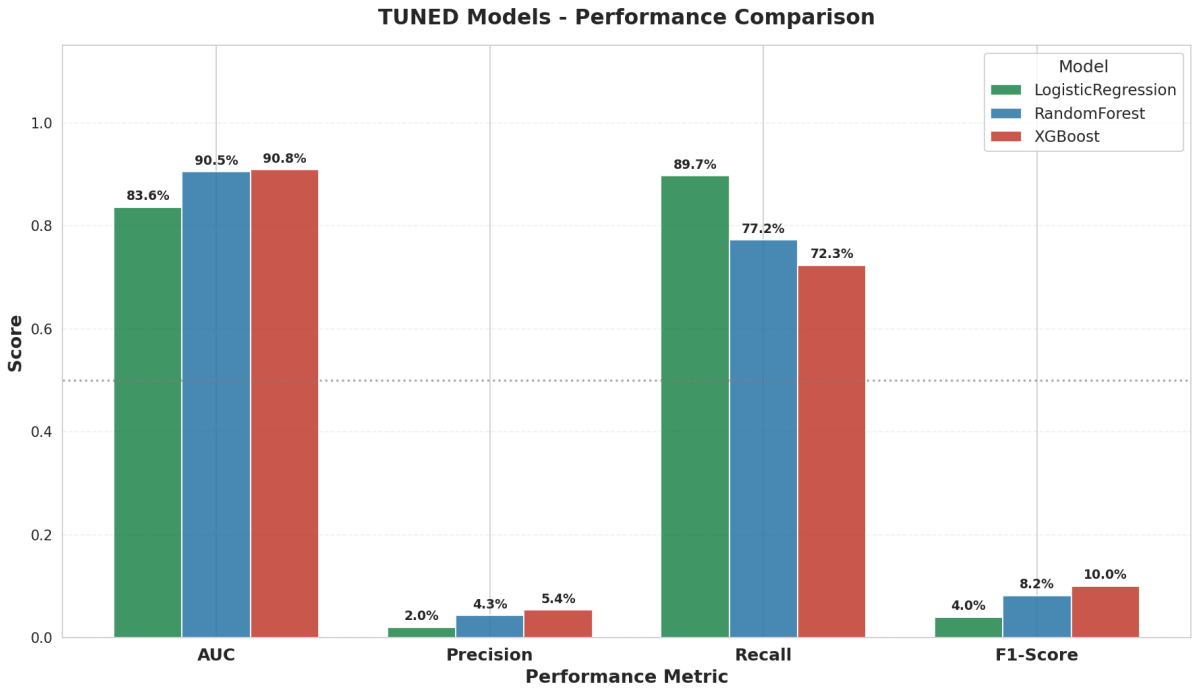


Figure 11: Model comparison (Tuned)

B Additional Tables

Category	Feature	Economic Interpretation (Short)
Financial Variables		
	total_assets	Larger asset bases generally provide greater financial stability.
	current_assets	Higher short-term assets improve immediate liquidity.
	current_liabilities	High short-term liabilities increase repayment pressure.
	current_debt	Short-term debt raises refinancing and liquidity risk.
	longterm_debt	High long-term debt reflects structural leverage risk.
	revenue	Declining revenues signal weakening business activity.
	ebit	Lower operating earnings reduce debt repayment capacity.
	interest_exp	High interest costs indicate heavier debt burden.
	ocf	Weak operating cash flow signals liquidity stress.
	fcf	Negative free cash flow implies reliance on external funding.
	inventories	Excess inventories may indicate weak sales or demand.
	retained_earnings	Low retained earnings reflect accumulated poor performance.
	shareholder_equity	Low equity increases vulnerability to insolvency.
Financial Ratios		
Liquidity	current_ratio	Measures ability to meet short-term obligations.
	quick_ratio	Liquidity measure excluding inventories.
Coverage	int_coverage	Indicates ability to cover interest payments.
Profitability	roa_approx	Measures profitability relative to total assets.
	roe_approx	Indicates return to shareholders; low values signal distress.
Margins	ebit_margin	Lower margins reduce operational resilience.
Efficiency	asset_turnover	Low turnover reflects inefficient asset utilization.

Table 5: Economic interpretation of the 20 features selected by RFE.

Logistic Regression (Base)		**Random Forest (Base)**		**XGBoost (Base)**	
C	1.0	n_estimators	100	n_estimators	100
penalty	l2	max_depth	3	max_depth	3
solver	lbfgs	min_samples_split	20	learning_rate	0.1
class_weight	balanced	min_samples_leaf	10	subsample	0.8
max_iter	1000	max_features	sqrt	colsample_bytree	0.8
fit_intercept	True	class_weight	{0: 0.506, 1: 19.545}	scale_pos_weight	38.64
random_state	42	bootstrap	True	eval_metric	logloss
		criterion	gini	objective	binary:logistic
		n_jobs	-1	random_state	42

Table 6: Hyperparameters for the Baseline models.

Logistic Regression (Tuned)		**Random Forest (Tuned)**		**XGBoost (Tuned)**	
C	2.0	n_estimators	150	n_estimators	200
penalty	l2	max_depth	8	max_depth	6
solver	lbfgs	min_samples_split	20	learning_rate	0.03
max_iter	2000	min_samples_leaf	10	min_child_weight	3
class_weight	balanced	min_impurity_decrease	0.0005	gamma	0.05
fit_intercept	True	max_features	sqrt	subsample	0.8
tol	0.0001	class_weight	{0: 0.506, 1: 19.545}	colsample_bytree	1.0
random_state	42	bootstrap	True	scale_pos_weight	38.64
		criterion	gini	eval_metric	logloss
		n_jobs	-1	objective	binary:logistic

Table 7: Optimized hyperparameters for the Tuned models using time-series and cross-validation.

Logistic Regression Grid		Random Forest Grid		XGBoost Grid	
C	{0.5, 1.0, 2.0}	n_estimators	{150, 250}	n_estimators	{150, 200}
penalty	{l2}	max_depth	{3, 5, 8}	max_depth	{3, 4, 6}
solver	{lbfgs}	min_samples_split	{10, 20, 30}	learning_rate	{0.03, 0.05, 0.1}
		min_samples_leaf	{5, 10, 15}	min_child_weight	{3, 5, 7}
				subsample	{0.7, 0.8, 1.0}
				colsample_bytree	{0.6, 0.8, 1.0}

Table 8: Search grids used in the RandomizedSearchCV for Tuned models.

C Code Repository

```
# Modules 01 to 05:
1. python src/data_features.py
2. python src/preprocessing.py
3. python src/models.py

# Cannot be launched on their own without first registering the models
4. python src/evaluation.py
5. python src/eda_interpretation.py

# Run the full pipeline (recommended)
python main.py
```

Listing 1: Execution Commands

```
# Skip SHAP analysis (faster)
python main.py --skip-shap

# Skip EDA & interpretation
python main.py --skip-eda

# Override split year
python main.py --split-year 2012
```

Listing 2: Optional Commands

GitHub Repository: `https://github.com/RichT09/project-tschumi.git`

D AI Utilisation

AI tools were used in a limited and supportive role throughout this project, such as:

Claude AI assisted with code-related tasks such as debugging, clarifying implementation details and improving code readability.

ChatGPT was also used to help improve the clarity and organization of the written report, as well as to support the planning and structuring of the Python modules.

All methodological choices, experiments, analyses, interpretations, and final conclusions were entirely developed by myself.