UNIVERSITY OF BELGRADE
Faculty of Economics and Business

UNIVERSITY OF BELGRADE

FACULTY OF ECONOMICS AND BUSINESS

CREDIT RISK ASSESSMENT USING FINANCIAL STATEMENT DATA: COMPARATIVE MODELING IN SAS AND PYTHON

Natalija Pajtić, 2805/18

Dragana Radojičić, PhD, Assistant Professor

UNIVERSITY OF BELGRADE
Faculty of Economics and Business

UNIVERSITY OF BELGRADE

FACULTY OF ECONOMICS AND BUSINESS

CREDIT RISK ASSESSMENT USING FINANCIAL STATEMENT DATA: COMPARATIVE MODELING IN SAS AND PYTHON

Program: International Master in Quantitative Finance

Author: Natalija Pajtić, 2805/18

Supervisor: Dragana Radojičić, PhD, Assisstent Profesor

Committee Members: Dragana Draganac, PhD, Assistant Profesor

Željko Jović, PhD, Assistant Profesor

Date: October 2025

UNIVERSITY OF BELGRADE
Faculty of Economics and Business

Statement of Academic Integrity

Student: Natalija Pajtić

Student ID number: 2805/18

The author of the work entitled:

Credit Risk Assessment Using Financial Statement Data: Comparative Modeling in SAS and Python

By signing, I declare:

- that the work is solely the result of my own research work;
- that I indicated or cited the work and opinions of other authors that I used in this paper in accordance with the Instructions;
- that all works and opinions of other authors are listed in the list of literature/references that are an integral part of this work and written in accordance with the Instructions; that I have obtained all permissions for the use of the author's work that are fully included in the submitted work and that I have clearly stated this;
- that I am aware that plagiarism is the use of other people's works in any form (such as quotations, paraphrases, images, tables, diagrams, designs, plans, photographs, films, music, formulas, websites, computer programs, etc.) without stating the author or presenting other people's works as mine, punishable by law (Act on Copyright and Related Rights, Official Gazette of the Republic of Serbia, No. 104/2009, 99/2011, 119/2012), as well as other laws and relevant acts of the University of Belgrade;
- that I am aware that plagiarism includes presenting, using, and distributing the work of lecturers or other students as one's own;
- that I am aware of the consequences that proven plagiarism can have on the submitted master's thesis and my status;
- that the electronic version of the master's thesis is identical to the printed copy and I agree to its publication under the conditions prescribed by the University's acts.

Belgrade, 4.11.2025. Signature Natalija Pajtić

UNIVERSITY OF BELGRADE
Faculty of Economics and Business

Statement of Personal Data Use

I authorize the publication of my personal data related to obtaining the academic title of master, such as my first name and surname, year and place of birth, and date of the thesis defense. This personal information may be published on the digital library's webpages, in the electronic catalog, and publications of the University of Belgrade – Faculty of Economics and Business.

I authorize the University of Belgrade – Faculty of Economics and Business library to enter my final (master's) thesis titled:

Credit Risk Assessment Using Financial Statement Data: Comparative Modeling in SAS and Python

into its digital repository, which is my original work.

I have submitted the final (master's) thesis along with all appendices in an electronic format suitable for permanent archiving.

My final (master's) thesis, stored in the Digital Repository of the University of Belgrade – Faculty of Economics and Business and available in open access, may be used by anyone who adheres to the provisions contained in the Creative Commons CC BY license, which allows for reproduction, distribution, and public disclosure of the work, as well as adaptations, with appropriate attribution to the author's name, even for commercial purposes.

In Belgrade, 4.11.2025.

Author's Signature

Natalija Pajtić

ABSTRACT

The subject of this thesis is the development of credit scoring models based on financial statements data as of December 31st, 2018, to December 31st, 2022, implemented in two statistical environments: SAS and Python. The objective is to develop and evaluate predictive models that assess the creditworthiness of Small Business (SB) companies in Serbia. That is legal entities with annual turnover below one million euros, operating in the Small Business segment under double-entry bookkeeping, meaning they must provide both balance sheets and income statements.

The thesis pursues two main goals: (1) comparing the performance of models developed in SAS, a commercial statistical software with integrated credit scoring capabilities, and Python, an open-source platform requiring complete customization of the modeling process, and (2) evaluation the predictive values of financial ratios in assessing credit risk for SB companies in Serbia.

The research questions address differences in performance between SAS's standardized algorithms and Python's custom-built procedures in Weight of Evidence transformation, variable grouping, and brute-force model selection, as well as the overall predictive capacity of financial ratios in credit risk modeling.

Both the SAS and Python final models demonstrate broad consistent predictive accuracy, with AUC values of ~0.725 and Gini coefficients of ~0.45 on training sample. On validation sample, the SAS model shows a modest advantage (AUC = 0.7228 vs. 0.7141). The Python model favors simplicity, achieving comparable accuracy with six variables, with maximal residual correlations of 51.21%, and slightly tighter control of collinearity. The SAS model incorporates seven variables, producing slightly higher validation AUC but also introducing a near-threshold correlation of 64.4%. Thus, SAS offers marginally stronger discrimination, while Python offers a leaner, more interpretable structure.

The findings provide contributions to both academic literature and professional practice, offering insights into the feasibility of applying open-source tools for advanced financial analytics within regulated banking environments.

TERMINOLOGY

ABT - Analytical Base Table

BRUTE-FORCE – The brute-force approach systematically develops all possible combinations of models (with predefined number of variables in the model) with variables from the shortlist of variables to identify the logistic regression with the best discriminatory power.

DE – Double-Entry bookkeeping is the basic principle of bookkeeping in which each transaction is recorded in at least two accounts - one on the debit side and one on the credit side.

DEFAULT STATUS - Default status means that the client is past due more than 90 days on any material credit obligation, or the client is unlikely to fully meet its credit obligations, regardless of collateral or guarantees.

DEFAULTER - Client in default status.

DEFAULT RATE - One-year rate of default status (DR=D/N, where D is the number of defaults during the period and N is the number of obligors in the portfolio not in default status at the beginning of the period).

DISCRIMINATORY POWER – Ability to distinguish differences between defaulted (bad) and non-defaulted (good) clients.

PD - Probability of default measures the probability of default of a counterparty over a period of one year from the observation date.

SAS - SAS Enterprise Miner is commercial statistical tool used for modeling.

SB – Small Business are legal entities with annual turnover below one million euros, operating in the Small Business segment under double-entry bookkeeping.

WoE - Weight of Evidence is statistical transformation which converts categorical variables or binned continuous variables into numerical values that reflects their predictive power.

Contents

INTRO

Credit scoring models are a core instrument of credit risk management because they enable banks to make consistent, automated decisions for clients whose observed risk is low. These models help screen clients efficiently and align lending practices with regulatory expectations for creditworthiness assessment and prudent risk-taking.

The mission of credit scoring models is to predict the probability of defaults of the clients based on their historical data. Default represents the clients' inability to repay their debt to the bank, basically default status means that client is more than 90 days past due on any material credit obligation, or that client is unlikely to fully meet its credit obligations, regardless of collateral or guarantees.

The development of credit scoring models has evolved from early expert-based assessments toward advanced, data-driven methodologies. The first empirical approaches to credit risk evaluation, introduced by Durand[1] (1941), applied statistical techniques such as discriminant analysis to quantify default probability. The adoption of logistic regression in the 1960s and the introduction of standardized scorecards by Fair, Isaac and Company (FICO) revolutionized credit evaluation, bringing consistency and transparency to lending decisions (Hand & Henley, 1997; Thomas, 2009)[2]. Subsequent technological progress and the implementation of international regulatory frameworks, notably Basel II and Basel III, reinforced the role of quantitative models in financial risk management (Basel Committee on Banking Supervision, 2006; 2011)[3]. In recent years, traditional statistical models have been increasingly complemented by machine learning and big data techniques, improving predictive power and adaptability to evolving market conditions (Lessmann et al., 2015[4]). This historical progression illustrates the ongoing shift from judgmental to fully analytical credit risk assessment, forming the foundation for modern credit scoring practices.

---

[1] Durand, D. (1941). Risk Elements in Consumer Instalment Financing.

[2] Hand, D. J., & Henley, W. E. (1997). Statistical classification methods in consumer credit scoring: A review.
Thomas, L. C. (2009). Consumer Credit Models: Pricing, Profit, and Portfolios.

[3] Basel Committee on Banking Supervision. (2006). International Convergence of Capital Measurement and Capital Standards: A Revised Framework (Comprehensive Version).
Basel Committee on Banking Supervision. (2011). Basel III: A Global Regulatory Framework for More Resilient Banks and Banking Systems.

[4] Lessmann, S., Baesens, B., Seow, H.-V., & Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research.

This thesis investigates whether a credit scoring model developed exclusively from financial statement data can deliver robust discrimination for Small Business companies. Rather than selecting only a few well-known metrics, a comprehensive set of financial ratios is calculated from the financial statements and then narrowed to a business-plausible long list. The objective of modeling is to quantify default risk with interpretable predictors and to obtain a model with stable discriminatory power.

Given the rapid evolution of analytical tools, a secondary objective is to compare a high-cost commercial tool (SAS) with an open-source tool with substantially lower licensing costs (Python). SAS has been long prevalent in banking sector due to its integrated modeling procedures and capacity for working with large data sets, while Python offers quick processing of large amounts of data, but with necessary good knowledge of programming, also there is a risk of human mistake in algorithms, and it is more difficult to validate the models through control functions. A parallel development of the models allows an evidence-based view of how the two environments differ across preprocessing, binning, variable reduction, and model selection, and whether an open-source pipeline can meet practical standards in banking sector.

The empirical strategy is to develop two models in parallel, one in SAS and one in Python, using the same data set, feature definitions, and selection criteria. The comparison spans data partitioning, Weight of Evidence transformation with automatic binning, univariate analysis, grouping by clustering and correlation thresholds, and brute-force logistic regression for model development.

These ideas lead to two central objectives of this thesis:

- Evaluate the predictive capacity of financial statement data for modeling default risk of firms within the Small Business companies in Serbia.
- Compare the predictive performance and practical usability of parallel developed models in SAS and Python.

These objectives lead to the following research questions:

- To what extent do financial ratios derived from annual financial statements predict one-year default events in the target population?

- How do SAS and Python differ in Weight of Evidence transformation, variable grouping, and brute-force logistic model construction when they use identical data and selection criteria?
- Can a fully open-source pipeline (Python) produce outcomes comparable to a commercial analytical tool (SAS) for use in regulated banking environments?

The data set contains financial ratios derived from five consecutive annual reports of financial statements for 13,920 Small Business companies in Serbia. A broad set of ratios is engineered to represent liquidity, leverage, profitability, growth, and activity. After quality checks, variables are transformed using Weight of Evidence with automatic binning, univariate analysis using Information Value and Population Stability Index for primary variables selection, and variable grouping combines variable clustering in SAS and correlation-based grouping in Python. In multivariate analysis, logistic regression is used as model development technique following a brute-force approach under statistical and business constraints. Final models are selected for their interpretability and discriminatory power.

Thesis is structured in five chapters and conclusion at the end. Chapter 1 details the modeling methodology. Chapter 2 describes the data set, default definition, performance window, and financial ratios. Chapter 3 covers data preparation and sample partitioning. Chapter 4 presents univariate analysis and variable selection. Chapter 5 presents multivariate analysis, development of the models and comparative analysis for models developed in SAS and Python. The thesis is finished with conclusion and main findings.

1 Model development methodology

Credit scoring models are statistical frameworks designed to assist financial institutions in assessing borrower creditworthiness and supporting lending decisions. By analyzing the relationship between borrower characteristics and repayment outcomes, these models help distinguish between defaulted[5] (bad) and non-defaulted (good) clients. A robust credit scoring system enables banks to minimize loan losses, allocate capital more efficiently, and comply with regulatory requirements (Thomas, 2009; Anderson, 2007) [6].

The foundation of credit scoring lies in the use of representative historical data sets. These data sets must capture sufficient information on borrowers' profiles and repayment behavior while clearly defining the target variable, distinguishing between defaulted and non-defaulted clients (Hand & Henley, 1997[7]). With reliable data, credit scoring models can reflect realistic credit risk patterns and ensure stability over time.

In this thesis, logistic regression is used as the principal modeling technique. Logistic regression is often used in credit risk development due to its interpretability, robustness, and regulatory acceptance. It estimates the probability of default (PD) based on borrower characteristics and financial indicators. The resulting probabilities can be converted into a scorecard that allocates points to specific attributes, producing a single numerical score for each borrower (Hosmer, Lemeshow, & Sturdivant, 2013; Siddiqi, 2017)[8]. This score provides an objective basis for credit decision-making and portfolio risk management.

To operationalize this approach, a carefully structured data set was prepared, containing financial ratios as independent variables and a binary target variable indicating default status. The definition of default follows a 12-month performance window, meaning that a counterparty is labeled as defaulted if it fails to meet its obligations within one year after the observation date.

Following standard practice, the development data set is partitioned into:

---

[5] Default definition is explained in section 2.2. Default Definition

[6] Anderson, R. (2007). The credit scoring toolkit: Theory and practice for retail credit risk management and decision automation.
Thomas, L. C. (2009). Consumer Credit Models: Pricing, Profit, and Portfolios.

[7] Hand, D. J., & Henley, W. E. (1997). Statistical classification methods in consumer credit scoring: A review.

[8] Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression
Siddiqi, N. (2017). Intelligent credit scoring: Developing and implementing better credit risk scorecards

- Training sample – used to estimate model parameters and train the model.
- Validation sample – used to evaluate the model's generalizability and predictive performance on unseen data (Larose & Larose, 2015[9]).

Once the data set is prepared, the modeling process follows several key steps:

- Transformation of variables through Weight of Evidence (WoE) binning,
- Univariate analysis based on Information Value (IV) and Population Stability Index (PSI),
- Variable reduction using clustering or correlation-based grouping,
- Model estimation using brute-force logistic regression,
- Model selection based on model performance metrics.

These procedures are implemented in both SAS and Python, allowing for parallel comparison of results across different platforms. The subsequent sections provide detailed descriptions of each stage in the modeling process.

1.1 Data Preparation

The quality of any credit scoring model depends on the accuracy, completeness, and representativeness of the underlying data. Data preparation therefore represents a fundamental step in the modeling process, directly influencing both model robustness and predictive power (Hand & Henley, 1997; Siddiqi, 2017)[10]. This phase encompasses data collection, data cleaning and preprocessing, and data partitioning of development sample into training and validation samples.

1.1.1. Data Collection

The data set consists primarily of financial ratios derived from annual financial statements, supplemented with client identifiers, segmentation data, and risk drivers such as year, region, exposure, and turnover. Each observation is associated with a binary target

[9] Larose, D. T., & Larose, C. D. (2015). Data mining and predictive analytics

[10] Hand, D. J., & Henley, W. E. (1997). Statistical classification methods in consumer credit scoring: A review.
Siddiqi, N. (2017). Intelligent credit scoring: Developing and implementing better credit risk scorecards

variable, which indicates whether the client defaulted within a 12-month observation window following the observation date.

Financial ratios were selected because they provide insight into firms' liquidity, profitability, leverage, and growth potential - key dimensions of creditworthiness frequently used in the assessment of small and medium-sized enterprises (Altman, Iwanicz-Drozdowska, Laitinen, & Suvas, 2017[11]). The availability of multiple years of historical data further allows for the evaluation of temporal stability and performance across economic cycles.

Ensuring data quality is essential in credit risk modeling, as unreliable or inconsistent input data can bias parameter estimates and reduce the model's discriminatory power. For this reason, the data set underwent strict validation procedures before being used for empirical analysis.

1.1.2. Data Cleaning and Preprocessing

The raw data set was subjected to several cleaning and preprocessing steps to improve accuracy and ensure compatibility with statistical modeling techniques:

- Handling of missing values: Missing observations, often due to incomplete financial statements, were coded as special values to distinguish them from valid data.
- Treatment of outliers and erroneous entries: Extreme or illogical values were coded as special values which were treated in WoE approach afterwards, to prevent distortions in the modeling process.
- Weight of Evidence (WoE): All variables were transformed using the WoE approach, which re-codes continuous predictors into values that are linearly related to the log-odds of default. This transformation facilitates compliance with the assumptions of logistic regression and improves interpretability (Hosmer, Lemeshow, & Sturdivant, 2013; Siddiqi, 2017)[12].

[11] Altman, E. I., Iwanicz-Drozdowska, M., Laitinen, E. K., & Suvas, A. (2017). Financial distress prediction in an international context: A review and empirical analysis of Altman's Z-score model.
[12] Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression
Siddiqi, N. (2017). Intelligent Credit Scoring: Building and Implementing Better Credit Risk Scorecards

1.1.3. Data Partition

The partitioning of the development data set into separate training and validation samples is a very important step. Proper partitioning ensures that model performance is evaluated on observations not used in estimation, thereby providing an unbiased measure of predictive accuracy and reducing the risk of overfitting (Hosmer, Lemeshow, & Sturdivant, 2013[13]).

In this thesis, the development data set was randomly stratified into a 70% training sample and a 30% validation sample. Stratified sampling was applied to maintain the same default rate across both partitions, ensuring that the class distribution of defaults and non-defaults is preserved. This is especially important in credit risk data, where the proportion of defaults is typically much smaller than the proportion of non-defaults. Stratification prevents distortions that could arise if the split were performed without accounting for class imbalance (Larose & Larose, 2015[14]).

- Training sample – used to estimate parameters, fit logistic regression models, and perform selection process.
- Validation sample – used exclusively for out-of-sample evaluation to confirm predictive performance and model robustness.

The objective of this partitioning process is to replicate real-world applications. In practice, a model is expected to classify new applicants whose repayment outcomes are unknown at the time of lending. Validating performance on a held-out sample provides an empirical test of how well the model generalizes. It allows the detection of overfitting or underfitting, facilitates the comparison of competing specifications, and yields a more reliable estimate of the model's long-term discriminatory power.

This practice aligns with recommendations found in both academic literature (Anderson, 2007; Thomas, 2009)[15] and regulatory guidance for internal rating systems, which emphasize the importance of out-of-sample validation in ensuring model credibility and stability.

[13] Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression
[14] Larose, D. T., & Larose, C. D. (2015). Data Mining and Predictive Analytics
[15] Anderson, R. (2007). The credit scoring toolkit: Theory and practice for retail credit risk management and decision automation.
Thomas, L. C. (2009). Consumer Credit Models: Pricing, Profit, and Portfolios.

1.2. Univariate Analysis

The purpose of the univariate analysis is to screen the initial pool of financial ratios and retain only those variables that demonstrate meaningful discriminatory power with respect to the target variable (default status). This step ensures that the subsequent modeling process is based on predictors that are both statistically reliable and interpretable from a business perspective (Siddiqi, 2017; Anderson, 2007) [16].

Weight of Evidence transformation was done to all variables in SAS and Python in two different ways:

- In SAS, WoE was done using integrated function by applying Interactive grouping node.
- In Python, WoE was implemented thought custom-built automatic binning which is explain in detail in following sections.

Two key indicators were employed:

- Information Value (IV)**:** measures the predictive strength of each variable in distinguishing between defaulted and non-defaulted clients.
- Population Stability Index (PSI): evaluates the stability of the variable's distribution over time, ensuring that its predictive capacity remains consistent across different periods.

Variables that exhibited low IV scores or unstable PSI values, are systematically excluded. This univariate analysis reduces the long list of variables into a concise short list of robust, and stable variables.

1.2.1. Weight of Evidence (WoE)

An essential step in preparing variables for logistic regression is transforming them into a form that establishes a linear relationship with the log-odds of default. This is achieved through the Weight of Evidence (WoE) transformation, a technique widely applied in the development of credit scoring models (Siddiqi, 2017[17]).

[16] Anderson, R. (2007). The credit scoring toolkit: Theory and practice for retail credit risk management and decision automation.
Siddiqi, N. (2017). Intelligent Credit Scoring: Building and Implementing Better Credit Risk Scorecards
[17] Siddiqi, N. (2017). Intelligent Credit Scoring: Building and Implementing Better Credit Risk Scorecards

Each continuous predictor is initially divided into a fixed number of categories (quantile-based bins). For each bin *i*, the WoE value is defined as:

$$WoE_i = \ln\left(\frac{\frac{N_i^{\text{non-default}}}{\sum_{i=1}^{n} N_i^{\text{non-default}}}}{\frac{N_i^{\text{default}}}{\sum_{i=1}^{n} N_i^{\text{default}}}}\right)$$

where:

- $N_i^{\text{non-default}}$: number of non-defaults in group i,
- N_i^{default}: number of defaults in group i,
- n: total number of groups.

This transformation ensures that the predictor variable is directly related to the log-odds of default, aligning with the assumptions of logistic regression (Hosmer, Lemeshow, & Sturdivant, 2013[18]).

1.2.2. Information Value (IV)

The Information Value (IV) complements the WoE transformation by quantifying the predictive strength of each variable. It is calculated as:

$$IV = \sum_{i=1}^{n} \left(\frac{N_i^{\text{non-default}}}{\sum_{i=1}^{n} N_i^{\text{non-default}}} - \frac{N_i^{\text{default}}}{\sum_{i=1}^{n} N_i^{\text{default}}}\right) \times WoE_i$$

Where:

- $N_i^{\text{non-default}}$: number of non-defaults in group i,
- N_i^{default}: number of defaults in group i,
- n: total number of groups.

Precondition: $N_i^{\text{non-default}}>0$ and $N_i^{\text{default}}>0$ for all groups i.

Interpretation of IV values follows industry guidelines (Siddiqi, 2017[19]):

[18] Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression
[19] Siddiqi, N. (2017). Intelligent Credit Scoring: Building and Implementing Better Credit Risk Scorecards

- IV < 0.02 → Not predictive,
- 0.02 ≤ IV < 0.1 → Weak predictive power,
- 0.1 ≤ IV < 0.3 → Medium predictive power,
- IV ≥ 0.3 → Strong predictive power.

The IV metric thus serves as a filter for excluding weak variables and retaining those with meaningful discriminatory capacity.

1.2.3. Automatic Binning in Python

In this thesis, WoE transformation was done in SAS, using integrated function for WoE in SAS EM – Interactive Binning node. Parallel, in Python WoE transformation was done using custom-built function for automatic binning.

Automatic binning function in Python was done in the following manner - each financial ratio was initially divided into 20 quantile bins. Each bin should contain enough data, including both default and non-default values, to ensure a stable WoE value. In this thesis, limitations for each bin were at least 30 observations and at least 10 defaulters. The WoE transformation was performed separately for each bin, under two monotonic assumptions:

- Increasing trend – higher values imply lower credit risk.
- Decreasing trend – higher values imply higher credit risk.

For each variable, the trend yielding the higher IV value was retained. Additionally, practical constraints were applied to ensure stability:

- A minimum default proportion of 1% in each class,
- A minimum WoE difference of 0.1 between adjacent bins.

These safeguards reduce the risk of overfitting and help ensure that the final set of variables is both predictive and robust across time horizons.

1.2.3.1. Automatic Binning – Growing trend

Following the initial WoE transformation, an automatic binning procedure was applied to improve stability and ensure monotonicity. The objective of this step is to merge adjacent categories in such a way that the relationship between the predictor variable and default risk follows a consistent trend. In this thesis, the growing trend assumption was adopted,

meaning that higher values of the variable are expected to be associated with lower default risk.

At each iteration, the following rules were applied:

- Recalculation of WoE values
    - After each merger, WoE values were recomputed for all classes to reflect the updated group structure.
- Merging based on similarity
    - Adjacent classes with an absolute WoE gap below 0.1 were merged.
    - If multiple candidate pairs existed, the first eligible pair was merged.
- Treatment of zero-default classes
    - Classes with no defaults were assigned a placeholder WoE (e.g., a large number such as 999999999) to flag instability.
    - These classes were then merged with the nearest class to ensure that both defaults and non-defaults were represented.
- Minimum default threshold
    - Classes with a default proportion below 1% were merged with the following class. This ensures that each class contributes meaningfully to the estimation of log-odds.
- Monotonicity enforcement
    - If the monotonicity assumption (increasing WoE with lower risk) was violated, the pair of classes with the largest deviation was merged.
    - This iterative process was repeated until the monotonic pattern was restored.

1.2.3.1.1. Automatic Binning – Example

To clarify the procedure, an example of automatic binning is provided in the tables below. In Table 1, the starting data set is divided into deciles (Step 0), with each class containing information on the number of defaults and non-defaults, and the corresponding Weight of Evidence (WoE) values.

Table 1 Example of Automatic Binning: Step 0

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	865	135	1,000	9.36%	17.88%	10.00%	13.50%	-0.6857
2	835	165	1,000	9.03%	21.85%	10.00%	16.50%	-0.9285
3	910	90	1,000	9.84%	11.92%	10.00%	9.00%	-0.2104
4	915	85	1,000	9.90%	11.26%	10.00%	8.50%	-0.1443
5	900	100	1,000	9.73%	13.25%	10.00%	10.00%	-0.3326
6	905	95	1,000	9.79%	12.58%	10.00%	9.50%	-0.2731
7	950	50	1,000	10.28%	6.62%	10.00%	5.00%	0.4684
8	1,000	-	1,000	10.82%	0.00%	10.00%	0.00%	999999999
9	985	15	1,000	10.65%	1.99%	10.00%	1.50%	2.7066
10	980	20	1,000	10.60%	2.65%	10.00%	2.00%	2.0084
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

Note: Class 8 contains no defaults, and therefore a placeholder WoE (999999999) is assigned to indicate instability.

In step 1, classes 5 and 6 from Table 1 are aggregated in class 5, as their WoE gap is below 0.1 and it is smaller than the gap between Classes 3 and 4. The results of the aggregation are presented in Table 2.

Table 2 Example of Automatic Binning: Growing Trend - Step 1

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	865	135	1,000	9.36%	17.88%	10.00%	13.50%	-0.6857
2	835	165	1,000	9.03%	21.85%	10.00%	16.50%	-0.9285
3	910	90	1,000	9.84%	11.92%	10.00%	9.00%	-0.2104
4	915	85	1,000	9.90%	11.26%	10.00%	8.50%	-0.1443
5	1,805	195	2,000	19.52%	25.83%	20.00%	9.75%	-0.3032
6	950	50	1,000	10.28%	6.62%	10.00%	5.00%	0.4684
7	1,000	-	1,000	10.82%	0.00%	10.00%	0.00%	999999999
8	985	15	1,000	10.65%	1.99%	10.00%	1.50%	2.7066
9	980	20	1,000	10.60%	2.65%	10.00%	2.00%	2.0084
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

In step 2, classes 3 and 4 from Table 2 are aggregated in class 3, due to their WoE gap being less than 0.1. The results of aggregation are presented in Table 3.

Table 3 Example of Automatic Binning - Growing Trend - Step 2

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	865	135	1,000	9.36%	17.88%	10.00%	13.50%	-0.6857
2	835	165	1,000	9.03%	21.85%	10.00%	16.50%	-0.9285
3	1,825	175	2,000	19.74%	23.18%	20.00%	8.75%	-0.1778
4	1,805	195	2,000	19.52%	25.83%	20.00%	9.75%	-0.3032
5	950	50	1,000	10.28%	6.62%	10.00%	5.00%	0.4684

6	1,000	-	1,000	10.82%	0.00%	10.00%	0.00%	999999999
7	985	15	1,000	10.65%	1.99%	10.00%	1.50%	2.7066
8	980	20	1,000	10.60%	2.65%	10.00%	2.00%	2.0084
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

It is shown that class 6 from Table 3 has no defaults, so in step 3, class 6 is6 merged with its neighbor class to ensure stability. This merger is presented in Table 4.

Table 4 Example of Automatic Binning – Growing Trend - Step 3

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	865	135	1,000	9.36%	17.88%	10.00%	13.50%	-0.6857
2	835	165	1,000	9.03%	21.85%	10.00%	16.50%	-0.9285
3	1,825	175	2,000	19.74%	23.18%	20.00%	8.75%	-0.1778
4	1,805	195	2,000	19.52%	25.83%	20.00%	9.75%	-0.3032
5	950	50	1,000	10.28%	6.62%	10.00%	5.00%	0.4684
6	1,985	15	2,000	21.47%	1.99%	20.00%	0.75%	3.4023
7	980	20	1,000	10.60%	2.65%	10.00%	2.00%	2.0084
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

Since the default rate in class 6 from Table 4 is below the 1% threshold, in step 4, class 6 is merged again with its neighbor. The results of the merger are presented in Table 5.

Table 5 Example of Automatic Binning – Growing trend - Step 4

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	865	135	1,000	9.36%	17.88%	10.00%	13.50%	-0.6857
2	835	165	1,000	9.03%	21.85%	10.00%	16.50%	-0.9285
3	1,825	175	2,000	19.74%	23.18%	20.00%	8.75%	-0.1778
4	1,805	195	2,000	19.52%	25.83%	20.00%	9.75%	-0.3032
5	950	50	1,000	10.28%	6.62%	10.00%	5.00%	0.4684
6	2,965	35	3,000	32.07%	4.64%	30.00%	1.17%	2.7066
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

A reversal in trend is detected between classes 1 and 2 from Table 5. Therefore, in step 5, they are merged into one class, and the results are presented in Table 6.

Table 6 Example of Automatic Binning – Growing trend - Step 5

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	1,700	300	2,000	18.39%	39.74%	20.00%	15.00%	-0.8123
2	1,825	175	2,000	19.74%	23.18%	20.00%	8.75%	-0.1778
3	1,805	195	2,000	19.52%	25.83%	20.00%	9.75%	-0.3032
4	950	50	1,000	10.28%	6.62%	10.00%	5.00%	0.4684
5	2,965	35	3,000	32.07%	4.64%	30.00%	1.17%	2.7066
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

Finally, another trend violation is found between classes 2 and 3 from Table 6, prompting a merger. The results of final merger, in step 6, are presented in Table 7.

Table 7 Example of Automatic Binning - Growth Trend - Step 6

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	1,700	300	2,000	18.39%	39.74%	20.00%	15.00%	-0.8123
2	3,630	370	4,000	39.26%	49.01%	40.00%	9.25%	-0.2421
3	950	50	1,000	10.28%	6.62%	10.00%	5.00%	0.4684
4	2,965	35	3,000	32.07%	4.64%	30.00%	1.17%	2.7066
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

After six iterations, no further aggregations are necessary. The resulting bins respect all the rules imposed:

- Minimum WoE gap (≥ 0.1),
- Minimum default share (≥ 1%),
- Monotonic trend assumption.

This final structure provides stable WoE values that reflect a consistent growing trend and ensure reliable predictive use of the variable in logistic regression.

1.2.3.2. Automatic Binning – Declining trend

The same iterative binning procedure was also applied under the assumption of a declining relationship between predictor values and credit risk. In this case, higher variable values are assumed to correspond to higher risk of default. At each step, WoE values were recalculated and compared to assess monotonicity. The following rules govern the aggregation:

- Adjacent classes with an absolute WoE gap of less than 0.1 were merged (ties resolved in sequential order).
- Classes containing zero defaults were merged with the previous class.
- Classes with a default share below 1% also merged with the previous class.
- If the monotonic declining trend was violated, the two classes with the largest deviation were merged.

This process was repeated until either all criteria were satisfied or the variable collapsed into a single class. Tables 8–16 illustrate the successive steps of the declining trend aggregation.

After both (growing and declining) procedures were performed, the version with the higher Information Value (IV) was retained as the final binning structure.

1.2.3.2.1. Automatic Binning – Example

To illustrate the procedure, the same variable used in the growing trend example is now re-evaluated under the declining trend assumption. As before, classes 5 and 6 show the smallest WoE gap (< 0.1) and therefore in Table 8, they are aggregated first.

Table 8 Example of Automatic Binning – Declining Trend - Step 1

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	865	135	1,000	9.35%	18.12%	10.00%	13.50%	-0.6857
2	835	165	1,000	9.02%	22.15%	10.00%	16.50%	-0.9285
3	910	90	1,000	9.83%	12.08%	10.00%	9.00%	-0.2104
4	915	85	1,000	9.89%	11.41%	10.00%	8.50%	-0.1443
5	1,805	195	2,000	19.50%	26.17%	20.00%	9.75%	-0.3032
6	950	50	1,000	10.26%	6.71%	10.00%	5.00%	0.4684
7	1,000	-	1,000	10.80%	0.00%	10.00%	0.00%	999999999
8	985	15	1,000	10.65%	1.99%	10.00%	1.50%	2.7066
9	980	20	1,000	10.60%	2.65%	10.00%	2.00%	2.0084
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

In step 2, classes 3 and 4 from Table 8 are merged into one class, because they had WoE gap less than 0.1. The results of the merger are presented in Table 9.

Table 9 Example of Automatic Binning – Declining Trend - Step 2

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	865	135	1,000	9.35%	18.12%	10.00%	13.50%	-0.6857
2	835	165	1,000	9.02%	22.15%	10.00%	16.50%	-0.9285
3	1,825	175	2,000	19.72%	23.49%	20.00%	8.75%	-0.1778
4	1,805	195	2,000	19.50%	26.17%	20.00%	9.75%	-0.3032
5	950	50	1,000	10.26%	6.71%	10.00%	5.00%	0.4684
6	1,000	-	1,000	10.80%	0.00%	10.00%	0.00%	999999999
7	985	15	1,000	10.65%	1.99%	10.00%	1.50%	2.7066
8	980	20	1,000	10.60%	2.65%	10.00%	2.00%	2.0084
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

In step 3, classes 5 and 6 from Table 9 are merged, because class 6 has no defaulters. The results of the merger are presented in Table 10.

Table 10 Example of Automatic Binning – Declining Trend - Step 3

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	865	135	1,000	9.35%	18.12%	10.00%	13.50%	-0.6857
2	835	165	1,000	9.02%	22.15%	10.00%	16.50%	-0.9285
3	1,825	175	2,000	19.72%	23.49%	20.00%	8.75%	-0.1778
4	1,805	195	2,000	19.50%	26.17%	20.00%	9.75%	-0.3032
5	1,950	50	2,000	21.07%	6.71%	20.00%	2.50%	1.1848
6	985	15	1,000	10.65%	1.99%	10.00%	1.50%	2.7066
7	980	20	1,000	10.60%	2.65%	10.00%	2.00%	2.0084
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

In step 4, classes 5 and 6 from Table 10 are aggregated into one class, due to trend violation. The results of the aggregation are presented in Table 11.

Table 11 Example of Automatic Binning – Declining Trend - Step 4

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	865	135	1,000	9.35%	18.12%	10.00%	13.50%	-0.6857
2	835	165	1,000	9.02%	22.15%	10.00%	16.50%	-0.9285
3	1,825	175	2,000	19.72%	23.49%	20.00%	8.75%	-0.1778
4	1,805	195	2,000	19.50%	26.17%	20.00%	9.75%	-0.3032
5	2,935	65	3,000	31.75%	8.61%	30.00%	2.17%	1.4908
6	980	20	1,000	10.60%	2.65%	10.00%	2.00%	2.0084
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

In step 5, classes 4 and 5 from Table 11 are merged into one class due to trend violation. The results of the merger are presented in Table 12.

Table 12 Example of Automatic Binning – Declining Trend - Step 5

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	865	135	1,000	9.35%	18.12%	10.00%	13.50%	-0.6857
2	835	165	1,000	9.02%	22.15%	10.00%	16.50%	-0.9285
3	1,825	175	2,000	19.72%	23.49%	20.00%	8.75%	-0.1778
4	4,740	260	5,000	51.27%	34.44%	50.00%	5.20%	0.4228
5	980	20	1,000	10.60%	2.65%	10.00%	2.00%	2.0084
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

In step 6, classes 4 and 5 from Table 12 are merged again into one class due to trend violation. The results of the merger are presented in Table 13.

Table 13 Example of Automatic Binning – Declining Trend - Step 6

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	865	135	1,000	9.35%	18.12%	10.00%	13.50%	-0.6857
2	835	165	1,000	9.02%	22.15%	10.00%	16.50%	-0.9285
3	1,825	175	2,000	19.72%	23.49%	20.00%	8.75%	-0.1778

4	5,720	280	6,000	61.87%	37.09%	60.00%	4.67%	0.5699
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

In step 7, classes 2 and 3 from Table 13 are merged into one class due to trend violation. The results of the merger are presented in Table 14.

Table 14 Example of Automatic Binning – Declining Trend - Step 7

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	865	135	1,000	9.35%	18.12%	10.00%	13.50%	-0.6857
2	2,660	340	3,000	28.74%	45.64%	30.00%	11.33%	-0.4788
3	5,720	280	6,000	61.87%	37.09%	60.00%	4.67%	0.5699
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

In step 7, classes 2 and 3 from Table 14 are merged again into one class due to trend violation. The results of the merger are presented in Table 15.

Table 15 Example of Automatic Binning – Declining Trend - Step 8

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	865	135	1,000	9.35%	18.12%	10.00%	13.50%	-0.6857
2	8,380	620	9,000	90.64%	82.12%	90.00%	6.89%	0.1072
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

In step 9, only two classes remained, violating the declining trend assumption. As a result, the final aggregation collapsed into a single class with a neutral WoE of zero. The results of the final step are presented in Table 16.

Table 16 Example of Automatic Binning – Declining Trend - Step 9

Class	Non-default	Default	Total	% Non-default	% Default	% Total	% Default rate	WOE
1	9,245	755	10,000	100.00%	100.00%	100.00%	7.55%	0
Total	**9,245**	**755**	**10,000**	**100.00%**	**100.00%**	**100.00%**	**7.55%**	

1.2.3.3. Outcome of Automatic Binning

The declining trend procedure ultimately led to an excessive number of aggregations, ending with a collapsed single-class structure and a WoE equal to zero. In contrast, the growing trend procedure preserved more granular predictive structure and yielded a higher Information Value (IV).

Accordingly, the growing trend binning was selected as the final transformation for this variable, since it maintained a meaningful inverse relationship between variable values

and observed default risk (higher values corresponded to lower risk, reflected in higher WoE scores).

1.2.3.4. Treatment of Missing Data and Outliers

The Weight of Evidence (WoE) framework provides a structured, model-consistent approach to handling both missing values and outliers. Decisions were guided by statistical evidence (e.g., bin default rates, WoE monotonicity, and Information Value) and by business logic.

- Missing values
    - When the share of missing observations for a variable was non-trivial, a dedicated “Missing” bin was created and carried through the WoE transformation and evaluation (IV, PSI, and monotonicity checks).
    - When the missing share was small, missing observations were merged with the bin that produced the most coherent risk ordering - typically the highest-risk bin when missingness was plausibly adverse (e.g., absent or stale financials), or the “Other”/nearest neighboring bin when missingness appeared random.
    - If a temporary bin containing only missing values produced zero defaults or zero non-defaults, it was merged at the next iteration in line with the automatic binning rules to avoid infinite WoE.
- Outliers
    - Genuine but extreme observations were retained and assigned to boundary bins so that their contribution was captured without destabilizing intermediate bins: negative outliers were grouped with the lowest-value class; positive outliers with the highest-value class.
    - If a boundary bin contained zero events (defaults or non-defaults), it triggered an immediate merge with the adjacent bin as per the automatic binning procedure to restore finite WoE and preserve monotonicity.

This treatment preserves the logistic link’s approximate linearity in WoE space, supports monotone risk ordering, and prevents inflation of IV due to data artifacts.

All resulting bins (including “Missing”) were re-evaluated for stability (PSI) and predictive strength (IV) and were subject to the same merging rules used in the growing/declining trend procedures.

1.2.4. Population Stability Index (PSI)

The Population Stability Index (PSI) was applied to evaluate whether the distribution of each predictor remained stable across time. This measure is important in credit risk modeling, since variables that exhibit substantial changes in their distribution may not provide reliable long-term predictive power (Anderson, 2007[20]).

The PSI is computed as:

$$PSI = \sum_{i=1}^{n} \left({S_{1i}}/{\sum_{i=1}^{n} S_{1i}} - {S_{2i}}/{\sum_{i=1}^{n} S_{2i}} \right) \times ln \left(\frac{{S_{1i}}/{\sum_{i=1}^{n} S_{1i}}}{{S_{2i}}/{\sum_{i=1}^{n} S_{2i}}} \right)$$

Where:

- S_{1i}: baseline sample.
- S_{2i}: comparison sample.
- i: attributes within sample.

Interpretation thresholds commonly used in practice are:

- PSI < 0.1: Stable distribution (no meaningful change).
- 0.1 ≤ PSI < 0.25: Moderate shift (requires monitoring).
- PSI ≥ 0.25: Significant shift (variable likely unsuitable for modeling).

The stability analysis is composed of two different tests:

- PSI Single Year: This test has been performed on all the indicators. In this step, the distribution of the last year included in the risk differentiation phase 2023 has been compared with the single year. To make an example, the distribution concerning 2023 has been compared with the one related to 2019 in the first run.

[20] Anderson, R. (2007). The credit scoring toolkit: Theory and practice for retail credit risk management and decision automation.

- PSI Time Window: This test has been performed on all the indicators. In this step, the distribution of the last year included in the risk differentiation phase 2023 has been compared with the time windows. To make an example, the distribution concerning 2023 has been compared with the one related to 2019-2023 in the first run.

The variables evaluated in this step have been considered stable when PSIs, calculated on both single year and time windows, are below the threshold of 0.1. So, the variables with PSI greater than 0.1 were excluded.

1.3. Multivariate Analysis

Variable clustering is the first step in multivariate analysis. The aim of clustering is to reduce the variables that express multicollinearity. Variables that exhibited excessive correlation with other predictors are systematically excluded. This clustering process reduced the list of variables into a concise short list suitable for model development.

One important aspect of multivariate analysis is understanding the relationships between the selected variables. After the clustering process, it is important to examine the correlations between the short-listed variables to ensure that multicollinearity (high correlations between predictor variables) does not adversely affect the model's performance.

Logistic regression is the dominant technique in credit risk modeling, primarily due to its interpretability, statistical robustness, and widespread regulatory acceptance (Hosmer, Lemeshow, & Sturdivant, 2013; Siddiqi, 2017)[21]. Basic premise for logistic regression was fulfilled by using short list of WoE transformed variables.

In this thesis, a brute-force approach was implemented in both SAS and Python, enabling a systematic model development based on all combinations of short listed variables and identification of combination of financial ratios that provides the strongest predictive capacity.

[21] Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression

1.3.1. Variable Clustering

To minimize redundancy and preserve interpretability, a variable clustering phase was performed after the univariate analysis. This step ensures that the final short list of predictors is not only statistically robust but also parsimonious, reducing the risk of overfitting while retaining sufficient discriminatory power (Anderson, 2007; Thomas, Crook, & Edelman, 2017) [22].

The clustering technique ensures that highly correlated variables do not dominate the model and that each chosen variable contributes distinct predictive information.

Two complementary approaches were applied, reflecting the different functionalities of SAS and Python:

- Clustering was performed using SAS Enterprise Miner, utilizing the Variable Clustering node.
- Correlation-based clustering was performed using custom-build algorithm in Python.

1.3.1.1 Clustering Algorithm in SAS

The clustering process was performed using the variable clustering node in SAS Enterprise Miner. The inputs for this clustering process were the WoE-transformed variables, which had already undergone the Weight of Evidence (WoE) transformation during the univariate analysis phase. The WoE transformation is designed to improve the relationship between predictor variables and the target variable (default status) by converting continuous variables into categorical ones with meaningful trends.

The procedure followed these steps:

- A correlation matrix of all variables was computed.
- Variables were assigned to global clusters based on eigenvalue decomposition of the correlation matrix.

[22] Anderson, R. (2007). The credit scoring toolkit: Theory and practice for retail credit risk management and decision automation
Thomas, L. C., Crook, J. N., & Edelman, D. B. (2017). Credit scoring and its applications

- Within each global cluster, additional subdivisions were created to refine the grouping structure.
- From each cluster, the most representative variable was selected according to:
    - squared correlation with the cluster centroid, and
    - predictive strength measured by Information Value (IV).

This clustering process ensured that each retained variable added unique information to the model while avoiding excessive multicollinearity.

1.3.1.2 Correlation-Based Clustering in Python

In Python, clustering approach was also applied directly to the Weight of Evidence (WoE) transformed variables. In Python, correlation-based clustering was through custom built algorithm. Pearson correlation coefficients were calculated for all variable pairs, and a threshold of 65% was imposed.

The procedure worked as follows:

- Variables were ranked by their univariate discriminatory power (measured by Information Value).
- Starting from the top-ranked variable, each subsequent variable was compared against the already selected “OK” set.
- If its correlation with all selected variables was below the threshold, it was retained.
- If the correlation exceeded the threshold with any retained variable, it was excluded.

This method systematically retained the strongest predictors while removing highly correlated alternatives. Compared with clustering, correlation-based grouping provides a more direct mechanism to enforce independence across variables.

1.3.1.3. Pearson Correlation Coefficient

The Pearson correlation coefficient (r) was employed to measure the degree of linear association between pairs of independent variables. High correlations indicate

redundancy, which can result in multicollinearity problems and unstable coefficient estimates in logistic regression (Hand & Henley, 1997[23]).

The coefficient is calculated as:

$$r = \frac{Cov(X_1, X_2)}{\sigma_{X_1} - \sigma_{X_2}}$$

Where:

- $Cov(X_1, X_2)$: covariance between variables X_1 and X_2,
- $\sigma_{X_1}, \sigma_{X_2}$: standard deviations of X_1 and X_2.

Interpretation guidelines:

- |r| < 0.3: Weak correlation.
- 0.3 ≤ |r| < 0.7: Moderate correlation.
- |r| ≥ 0.7: Strong correlation, indicating redundancy.

For this thesis, a threshold of 65% ($|\, r \,| = 0.65$) was applied. When two variables exceeded this threshold, the one with the lower Information Value (IV) was excluded. This procedure ensured that only the most informative and independent predictors were retained for further analysis, reducing the risk of multicollinearity in the final models.

1.3.2. Logistic regression

Logistic regression is a binary classification method designed to estimate the probability of a discrete event, such as borrower default. The model expresses the log-odds of default as a linear function of the predictors:

$$\text{logit}(p) = \ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n$$

where:

- p: probability of default,
- β_0: model intercept,

[23] Hand, D. J., & Henley, W. E. (1997). Statistical classification methods in consumer credit scoring: A review.

- $\beta_1, \ldots, \beta_n$: regression coefficients,
- $X_1, \ldots, X_n$: explanatory variables.

Parameters are estimated using maximum likelihood estimation (MLE), which selects coefficient values that maximize the probability of observing sample data (Menard, 2010[24]). This method is particularly suitable for credit scoring since it accommodates both continuous and categorical predictors. Moreover, the use of the Weight of Evidence (WoE) transformation ensures that predictors display a monotonic and approximately linear relationship with the log-odds of default, thereby improving both model stability and interpretability (Siddiqi, 2017[25]).

1.3.3. Brute force approach

The brute-force approach systematically evaluates different subsets of explanatory variables to identify the logistic regression model with the best predictive performance. Unlike heuristic methods such as stepwise regression, brute-force search ensures that all feasible combinations are explored, reducing the risk of overlooking relevant interactions or variable synergies (Siddiqi, 2017; Thomas, Crook, & Edelman, 2017) [26].

Brute-force in SAS made all possible combinations of models with four, five, six and seven variables in model from the short list of variables. In Python, brute-force was implemented slightly differently, it made combinations of valid models with two, three, four, five, six, seven, eight, nine and ten variables in model from the short list of variables by following predefined rules, which are explained in the sections bellow.

1.3.3.1. SAS Procedure

In SAS, brute-force modeling was implemented in an exhaustive manner. All possible combinations of models with four, five, six and seven variables in the model, from the short list of variables, were generated, resulting in more than 25,000 candidate potential models, of which approximately 13,000 satisfied validity checks. Validation criteria were:

- All variables in the model were statistically significant ($p < 0.05$),

[24] Menard, S. (2010). Logistic Regression: From Introductory to Advanced Concepts and Applications.
[25] Siddiqi, N. (2017). Intelligent credit scoring: Developing and implementing better credit risk scorecards
[26] Siddiqi, N. (2017). Intelligent credit scoring: Developing and implementing better credit risk scorecards
Thomas, L. C., Crook, J. N., & Edelman, D. B. (2017). Credit scoring and its applications

- All coefficients were negative,
- MLE converge.

This approach ensured full coverage of potential specifications, but it was computationally intensive.

1.3.3.2. Python Procedure

In contrast, Python employed a custom-built automated brute-force pipeline designed to balance completeness with efficiency. Brute-force in Python made all combinations of valid models with two, three, four, five, six, seven, eight, nine and ten variables in the model, from the short list of variables, in accordance with the following rules. A 65% correlation threshold was enforced at every stage to control multicollinearity. The modeling proceeded iteratively:

- Pairwise models – All two-variable combinations were estimated. Only pairs that met the following criteria were retained:
  - coefficients negative and statistically significant ($p < 0.05$),
  - each variable's marginal contribution to Gini ≥ 5%,
  - correlation between variables below the 65% threshold.
- Triplets – The top 40 performing pairs from the previous step were expanded with additional, third variable. Model combinations with failing variable correlation or variable significance checks were discarded.
- Higher-order models – This process continued iteratively up to ten-variable models. At each stage, retained models had to satisfy statistical validity and economic interpretability.

This approach resulted in 283 valid models. While the total number of models is much lower than in SAS, the difference stems from the pre-filtering rules embedded in the Python pipeline. By discarding invalid or redundant combinations early and always picking top 40 models for each number of variables in model, Python's approach avoided generating thousands of weak or collinear models, leading to a more efficient but still comprehensive exploration of the modeling space.

1.3.4. Final model selection

From the pool of valid candidate models, the final specification was chosen using both statistical performance measures and practical considerations. The following criteria guided the decision:

- Consistent performance across both training and validation samples, as measured by the Gini coefficient;
- Business interpretability, ensuring that selected predictors align with established financial and economic intuition;
- Coverage of key financial dimensions, so that the model reflects liquidity, leverage, profitability, and growth;
- Absence of multicollinearity, thereby improving stability and reliability of parameter estimates.

This balanced approach ensured that the chosen model was not only statistically robust but also applicable in real-world decision-making contexts (Anderson, 2007; Siddiqi, 2017)[27].

In the following sections, the metrics used for model selection are described along with interpretation of their results.

1.3.4.1 Receiver Operating Characteristic (ROC) Curve

The Receiver Operating Characteristic (ROC) curve illustrates the trade-off between sensitivity (true positive rate) and 1-specificity (false positive rate) across all classification thresholds. A well-performing model produces a curve that bends sharply toward the upper-left corner, indicating a strong ability to detect defaults while minimizing false alarms (Hosmer, Lemeshow, & Sturdivant, 2013[28]).

[27] Anderson, R. (2007). The credit scoring toolkit: Theory and practice for retail credit risk management and decision automation.
Siddiqi, N. (2017). Intelligent credit scoring: Developing and implementing better credit risk scorecards
[28] Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). Applied logistic regression

1.3.4.2 Area Under the Curve (AUC)

The AUC condenses the ROC curve into a single summary metric, representing the probability that a randomly chosen defaulted borrower receives a higher predicted score than a randomly chosen non-defaulted borrower.

Interpretation:

- AUC = 0.5 → No discriminatory power.
- AUC > 0.7 → Acceptable.
- AUC > 0.8 → Good.
- AUC > 0.9 → Excellent.

1.3.4.3 Gini Coefficient

The Gini coefficient, directly derived from AUC, is a widely used performance indicator in credit risk modeling:

$$\text{Gini} = 2 \times AUC - 1$$

Interpretation:

- Gini = 0 → No predictive power.
- Gini > 0.3 → Acceptable in credit risk models.
- Gini > 0.5 → Good discrimination.
- Gini = 1.0 → Perfect model.

Its interpretability and linear relationship with AUC make Gini a preferred comparison measure in the banking industry (Anderson, 2007[29]).

1.3.4.5 Kolmogorov–Smirnov (KS) Statistic

The Kolmogorov–Smirnov (KS) statistic quantifies the maximum difference between the cumulative distributions of predicted probabilities for defaulters and non-defaulters.

Interpretation:

[29] Anderson, R. (2007). The credit scoring toolkit: Theory and practice for retail credit risk management and decision automation.

- KS ≈ 0 → Poor discriminatory power.
- KS > 20% → Acceptable discrimination.
- KS > 40% → Strong discrimination.

The KS statistic is particularly valuable because it highlights the point of maximum separation between defaulted and non-defaulted clients, making it a practical tool for validation in credit scoring applications (Thomas, Crook, & Edelman, 2017[30]).

1.3.4.6 Akaike Information Criterion (AIC)

The Akaike Information Criterion (AIC) assesses model fit while penalizing excessive complexity:

$$AIC = 2k - 2\ln(L)$$

Where:

- k: number of estimated parameters,
- L: maximized likelihood value.

A lower AIC indicates a more parsimonious model, striking a balance between explanatory power and simplicity (Burnham & Anderson, 2002[31]).

1.3.4.7 Bayesian Information Criterion (BIC)

The Bayesian Information Criterion (BIC) functions similarly to AIC but applies a stronger penalty on complexity:

$$BIC = k\ln(n) - 2\ln(L)$$

Where:

- n: sample size.

[30] Thomas, L. C., Crook, J. N., & Edelman, D. B. (2017). Credit scoring and its applications
[31] Burnham, K. P., & Anderson, D. R. (2002). Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach.

BIC is especially useful for large data sets, as it systematically favors simpler, more stable models. In credit risk modeling, BIC is often used alongside AIC to confirm robustness and avoid overfitting (Greene, 2012[32]).

[32] Greene, W. H. (2012). Econometric Analysis

2. Description of the sample

The credit scoring models in this thesis are developed using financial statement data at the counterparty level, enabling a detailed assessment of each firm's financial condition and probability of default (PD). The data set consists financial statements data as of December 31st, 2018, to December 31st, 2022, of 13,920 Small Business companies in Serbia.

From the available financial statements, a total of 123 financial ratios were derived, representing multiple dimensions of business performance:

- Financial position – solvency and capital structure,
- Profitability – returns and margins,
- Liquidity – short-term repayment capacity,
- Operational efficiency – asset utilization and turnover.

Together, these indicators provide a comprehensive framework for analyzing the borrower's repayment capacity and assessing default risk.

2.1. Criteria for Initial Sample Selection

The construction of the initial data set followed a rigorous set of eligibility criteria to ensure consistency with regulatory standards and alignment with the research objectives. The criteria applied were:

- Non-default status: Firms already in default at the observation date were excluded. Default was defined as the failure to meet financial obligations.
- Counterparty type: Only legal entities and entrepreneurs were considered eligible.
- Segment classification: Small Business segment includes the following rules:
    - Revenue limits – annual revenues not exceeding 1,000,000 EUR.
    - SB-DE risk segment classification – only firms assigned to the Small Business with Double-Entry Bookkeeping (SB-DE) risk category were included.

This requirement ensures the availability of structured balance sheet and income statement data, allowing reliable verification.

- Availability of financial statements: To qualify, each counterparty had to provide a valid t-1 financial statement, available at the observation date (June 30th of year t).

2.2. Default Definition

Default status is determined at the client's level and classified under two conditions:

- Subjective criterion: The bank judges that the obligor is unlikely to fully meet its credit obligations, regardless of collateral or guarantees. Indicators may include bankruptcy, liquidation, severe financial distress, or classification under IFRS non-performing loan categories such as "Unlikely to Pay" or "Doubtful."
- Objective criterion: The obligor is more than 90 days past due on any material credit obligation. Materiality is defined if both of following conditions are breached:
    - Relative threshold: at least 1% of total on-balance exposure, and
    - Absolute threshold: more than 1,000 RSD for retail obligors and 10,000 RSD for others.

A counterparty is recorded as defaulted if subjective or objective criterion is met. Reclassification to non-default status requires that these conditions are no longer present for at least three consecutive months, consistent with regulatory guidelines[33].

2.3. Performance Windows

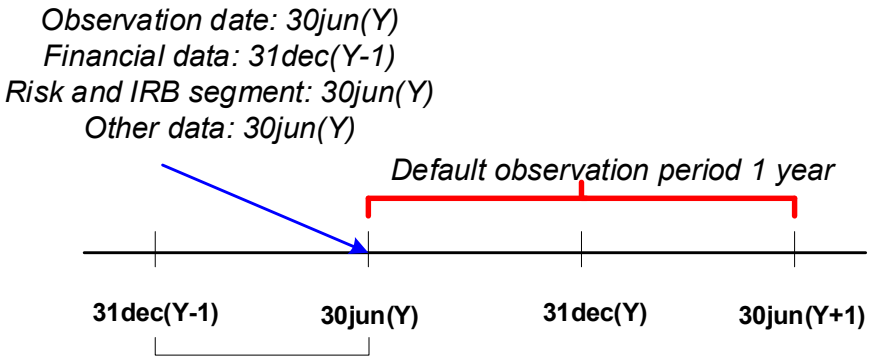
The performance window defines the horizon over which the probability of default (PD) is assessed, based on financial and risk data available at the observation date. In this thesis, the default observation period is fixed at 12 months after the observation date, consistent with regulatory practice.

The observation date is set at 30th June for each year from 2019 to 2023, when counterparties are assigned to their risk segments. Financial ratios are derived from the

[33] National Bank of Serbia (2024) Decision on the Classification of Bank Balance Sheet Assets and Off-Balance Sheet Items

previous fiscal year's financial statements (31st December of year *Y–1*), ensuring that only information available at the observation date is used for risk assessment.

Figure 1 Diagram of default observation period



Key components of the performance window shown in Figure 1 include:

- Observation date (30th June Y): Reference point for risk classification and segmentation.
- Financial data (31st December Y–1): Source for calculating financial ratios used as explanatory variables.
- Segmentation (30th June Y): Assignment of clients to segments ensures comparability and regulatory alignment.
- Default observation period (30th June Y → 30th June Y+1): Twelve-month horizon during which repayment behavior is monitored. Defaults within this window are attributed to the financial and risk profile at the observation date.

By repeating this procedure for five consecutive years (2019–2023), the data set captures both cross-sectional and temporal dynamics of SB default risk, providing a robust foundation for model development.

2.4. Financial Ratios

The credit scoring model incorporates a comprehensive set of financial ratios to evaluate counterparties' financial health and default risk. In total, 123 financial ratios were derived

from financial statements, grouped into six categories reflecting distinct dimensions of financial performance.

- Activity Ratios (9 ratios): Measure how effectively a counterparty uses its resources to generate sales and manage working capital (Brigham & Ehrhardt, 2019[34]).

  Example:

$$\text{Sales to Average Liabilities} = \frac{\text{Net Sales}}{\text{Average Total Liabilities}}$$

  Where:

  - Net Sales refers to total revenue from operations after deducting returns, allowances, and discounts.
  - Average Total Liabilities is the average of beginning and ending total liabilities for the period, smoothing out fluctuations and providing a more accurate denominator.

  Interpretation:

  - Higher values indicate efficient liability utilization to generate revenue; lower values may suggest inefficiency or reliance on debt without adequate revenue generation.

- Cash Flow Ratios (13 ratios): Assess liquidity and financial flexibility by linking operating cash flows to other metrics (Penman, 2012[35]).

  Example:

$$\text{CFOBIT2EBITDA} = \frac{\text{CFOBIT}}{\text{EBITDA}}$$

  Where:

[34] Brigham, E. F., & Ehrhardt, M. C. (2019). *Financial management: Theory & practice* (16th ed.). Cengage Learning.
[35] Penman, S. H. (2012). Financial statement analysis and security valuation.

- CFOBIT (Cash Flow from Operations Before Interest and Tax) refers to the operating cash flow adjusted to exclude interest and tax payments.

- EBITDA (Earnings Before Interest, Taxes, Depreciation, and Amortization) is a commonly used proxy for operating profitability, as it excludes non-operating expenses and non-cash items.

Interpretation:

- Values close to or above 1 imply strong earnings supported by cash flows; values below 1 may indicate weak cash conversion.

- Growth Ratios (16 ratios): Capture performance trends over time by comparing year-over-year changes (Ciampi & Gordini, 2015[36]).

Example:

$$\text{Gross Profit Growth} = \frac{\text{Gross Profit}_{t1} - \text{Gross Profit}_{t0}}{\text{Gross Profit}_{t1}}$$

Where:

- Gross Profit_T0 = Gross profit reported in the most recent financial year.

- Gross Profit_T1 = Gross profit reported in the prior financial year.

Interpretation:

- Growth suggests operational improvement; negative growth may reflect declining sales, rising costs, or inefficiency.

- Leverage Ratios (22 ratios): Assess capital structure and reliance on debt financing (Altman et al., 2017[37]).

Example:

[36] Ciampi, F., & Gordini, N. (2015). Small enterprise default prediction modeling through artificial neural networks: An empirical analysis of Italian SMEs.
[37] Altman, E. I., Iwanicz-Drozdowska, M., Laitinen, E. K., & Suvas, A. (2017). Financial distress prediction in an international context: A review and empirical analysis of Altman's Z-score model.

$$\text{Average Liabilities to Assets} = \frac{\text{Average Total Liabilities}}{\text{Total Assets}}$$

Where:

- Average Total Liabilities is calculated as the average of total liabilities at the beginning and end of the period.
- Total Assets refers to the value of everything the company owns as reported at the end of the financial period.

Interpretation:

- Higher ratios signal greater dependence on debt, increasing financial risk; lower ratios suggest more conservative equity-based funding.

- Liquidity Ratios (16 ratios): Evaluate the ability to meet short-term obligations (Brigham & Ehrhardt, 2019[38]).

  Example:

$$\text{Cash-to-Assets} = \frac{\text{Cash and Cash Equivalents}}{\text{Total Assets}}$$

Where:

- Cash and Cash Equivalents include highly liquid assets such as bank balances, marketable securities, and short-term investments that are readily convertible to known amounts of cash.
- Total Assets represent the total book value of all resources owned by the company at the end of the financial year.

Interpretation:

- Higher values indicate stronger liquidity buffers; lower values suggest potential cash shortfalls.

[38] Brigham, E. F., & Ehrhardt, M. C. (2019). Financial management: Theory & practice.

- Profitability Ratios (47 ratios): Measure the ability to generate returns relative to revenue, assets, or liabilities (Penman, 2012; Altman et al., 2017[39]).

  Example:

$$\text{Gross Profit Growth to Liabilities} = \frac{\text{Growth in Gross Profit}}{\text{Total Liabilities}}$$

  Where:

  - Growth in Gross Profit is calculated as the percentage or absolute increase in gross profit between the current period (T0) and the previous period (T1).
  - Total Liabilities represent the company's total financial obligations reported at the end of the current period (T0).

  Interpretation:

  - High ratios indicate that profit growth supports debt servicing; low values may indicate that liabilities are growing faster than profitability.

Together, these six categories provide a multidimensional assessment of counterparties' liquidity, solvency, efficiency, leverage, growth, and profitability. They form the analytical foundation of the scoring model, ensuring that risk assessment captures both current financial standing and performance dynamics (Altman et al., 2017; Ciampi & Gordini, 2015[40]).

[39] Penman, S. H. (2012). Financial statement analysis and security valuation.
Altman, E. I., Iwanicz-Drozdowska, M., Laitinen, E. K., & Suvas, A. (2017). Financial distress prediction in an international context: A review and empirical analysis of Altman's Z-score model.
[40] Altman, E. I., Iwanicz-Drozdowska, M., Laitinen, E. K., & Suvas, A. (2017). Financial distress prediction in an international context: A review and empirical analysis of Altman's Z-score model.
Ciampi, F., & Gordini, N. (2015). Small enterprise default prediction modeling through artificial neural networks: An empirical analysis of Italian SMEs.

3. Data preparation

The quality and robustness of a credit scoring model depends strongly on the way data are prepared before model development. Data collection, cleaning, preprocessing, and partitioning are crucial steps, as they directly affect the model's predictive power and stability.

3.1. Data Collection

The development data set used in this thesis covers 13,920 counterparties belonging to the Small Business segment, observed over a five-year period from June 30th, 2019 to June 30th, 2023.

The data set integrates four key elements:

- Segmentation data – Counterparty classifications at each observation date.
- Risk drivers – Non-financial attributes such as year, region, exposure level, and turnover, consistently captured across all observation dates.
- Financial ratios – A total of 123 indicators, derived from year-end financial statements (December 31st of the year prior to each observation date). These ratios span profitability, liquidity, leverage, growth, and efficiency. For example, ratios calculated from financial statements as of December 31st, 2022, correspond to the observation date of June 30th, 2023.
- Default status – Defined at the counterparty level, tracking whether an entity defaulted within the 12-month performance window following each observation date. For instance, June 30th, 2023, observation captures defaults recorded until June 30th, 2024.

3.2. Data Cleaning and Preprocessing

The data set included multiple categories of variables, each serving a distinct role in the analysis (Table 17).

Table 17 Explanation of variables in data table

Variable type	Variable type description	Number of variables
ID	Unique Identifier	1
Auxilary	Used only for analysis	12

TG	Target variable (default status)	1
Core	Present a summary of the main items from financial statement e.g. ASSET, EBIT, EBITDA, NET SALES	4
Ratio	Formed by dividing the corresponding items from financial statement	107
Core Growth	Compare certain item from the last available (T0) compared to the previous (T1) end year financial statement	9
Ratio Growth	Compare certain ratio from the last available (T0) compared to the previous (T1) end year financial statement	7

Auxiliary and core variables were excluded from modeling because they are not suitable as predictors. The data set contained 123 financial ratios, which formed the pool of candidate predictors for the SB Financial Model.

3.2.1. Development Sample

Development sample was created by excluding from the initial data set the clients who were in default at observation date. So the exclusion rule that was applied is:

- Default exclusion: Clients in default at the observation date were removed.

The effect of these exclusions is summarized in Table 18.

Table 18 Exclusions on development sample (2019-2023)

Data set	Number of observations	Number of observations %	Exposure (EUR)	Exposure share %
STARTING DATA SET	**62,248**	**100.00%**	**603,762,700**	**100.00%**
DEFAULT EXCLUSION	9,201	14.78%	49,623,292	8.22%
FINAL DATA SET	**53,047**	**85.22%**	**554,139,408**	**91.78%**

Out of the initial 62,248 observations, 9,201 were excluded due to default on observation date, leaving 53,047 valid records. Within the final data set, 1,697 clients defaulted, corresponding to a default rate of 3.20%. The yearly breakdown is shown in Table 19.

Table 19 The Annual Default Rate in the Data Set

Year	Number of observations in the final sample	Number of defaulted observations	% Default rate
2019	9,679	307	3.17%
2020	10,070	272	2.70%
2021	9,868	337	3.42%
2022	9,931	348	3.50%
2023	13,499	433	3.21%
Total	**53,047**	**1,697**	**3.20%**

The annual results show declining default rate in first two years, which felt from 3.17% in 2019 (DR from June 30th 2018 to June 30th 2019) to 2.70% in 2020 (DR from June 30th 2019 to June 30th 2020). Afterwards there were increase in default rate in 2021 (DR from

June 30th 2020 to June 30th 2021) and 2022 (DR from June 30th 2021 to June 30th 2022), which grow to 3.42% and 3.50% respectively. The reason for increase in default rate was due to Covid 19 pandemic which happend during 2020. In 2023 (DR from June 30th 2022 to June 30th 2023) there was decline in default rate, which felt to 3.21% which is in line with average default rate.

3.3. Data Partition

Development data set was split into two subsets to enable robust model estimation and unbiased performance assessment:

- Training sample (70%) – used to estimate model parameters and identify statistical relationships between financial ratios and default status.
- Validation sample (30%) – reserved exclusively for performance testing on out-of-sample data.

To maintain comparability, stratified sampling was applied, preserving the default rate across both partitions. This approach minimizes bias and ensures that the risk structure of the data is consistently reflected in both samples. Partitioning was implemented in SAS and Python, with nearly identical outcomes (Table 20).

Table 20 Development Sample Randomly Stratified on Training and Validation Samples in SAS and Python

Sample	SAS			PYTHON		
	Number Of Observations	Number Of Defaulted Observations	Default Rate	Number Of Observations	Number Of Defaulted Observations	Default Rate
TRAINING	37,133	1,188	3.20%	37,135	1,189	3.20%
VALIDATION	15,914	509	3.20%	15,912	508	3.19%
TOTAL	**53,047**	**1,697**	**3.20%**	**53,047**	**1,697**	**3.20%**

The near-identical distributions of defaults between SAS and Python confirm that both procedures generated balanced data sets. This alignment provides a sound foundation for subsequent model development and ensures that performance comparisons across platforms are not distorted by uneven class representation.

3.3.1. Default Rate Analysis

To ensure that stratified sampling preserved the underlying risk characteristics, default rate distributions were examined across key risk drivers: year, organization form, region,

exposure, and turnover. The analysis compared the development[41] (DVL), training (TRN), and validation (VLD) samples in both SAS and Python, and the results are presented in Table 21.

Table 21 Default Rate per Year on Development, Training and Validation Samples in SAS and Python

YEAR	SAS			PYTHON		
	DR DVL	DR TRN	DR VLD	DR DVL	DR TRN	DR VLD
2019	3.17%	3.11%	3.30%	3.17%	3.34%	3.29%
2020	2.70%	2.75%	2.60%	2.70%	2.58%	2.59%
2021	3.42%	3.46%	3.33%	3.42%	3.16%	3.32%
2022	3.50%	3.48%	3.55%	3.50%	3.73%	3.54%
2023	3.21%	3.21%	3.21%	3.21%	3.17%	3.20%
TOTAL	**3.20%**	**3.20%**	**3.20%**	**3.20%**	**3.20%**	**3.19%**

Default rates steadily declined from around 3.23% in 2019 to 2.65% in 2020 across both SAS and Python samples. This trend indicates improving portfolio quality. Default rates increase to 3.35% in 2021, and 3.55% in 2022. This trend indicates problem that Small Business clients had due to Covid 19 pandemics. Default rate slightly decreases to 3.20% in 2023. Trends of default rates between development, training and validation samples are the same in SAS and Python across the years, indicating accurate partitioning of development sample between SAS and Python.

Default rate per organization form (entrepreneurs and others) on development, training and validation samples in SAS and Python, are presented in Table 22.

Table 22 Default Rate per Organization Form on Development, Training and Validation Samples in SAS and Python

ORGANIZATION FORM	SAS			PYTHON		
	DR DVL	DR TRN	DR VLD	DR DVL	DR TRN	DR VLD
ENTREPRENEURS	2.95%	2.91%	3.05%	2.95%	2.96%	2.92%
OTHERS	3.26%	3.28%	3.24%	3.26%	3.25%	3.27%
TOTAL	**3.20%**	**3.20%**	**3.20%**	**3.20%**	**3.20%**	**3.20%**

Entrepreneurs displayed slightly lower default rates (2.95 – 3.05%) than other legal entities (3.24 – 3.28%). This suggests marginally stronger repayment behavior among entrepreneurs, though the difference is modest. The distribution of default rate between entrepreneurs and other clients is stable between development, training and validation

[41] Development sample is training and validation samples together.

samples in SAS and Python, suggesting good partition of development sample in both environments.

Default rate per region (Belgrade, Kragujevac, Niš and Novi Sad) on development, training and validation samples in SAS and Python, are presented in Table 23.

Table 23 Default Rate per Region on Development, Training and Validation Samples in SAS and Python

REGION	SAS			PYTHON		
	DR DVL	DR TRN	DR VLD	DR DVL	DR TRN	DR VLD
BELGRADE	3.64%	3.73%	3.45%	3.64%	3.66%	3.58%
KRAGUJEVAC	2.98%	2.92%	3.12%	2.98%	2.90%	3.14%
NIŠ	3.21%	3.11%	3.45%	3.21%	3.41%	3.10%
NOVI SAD	2.84%	2.89%	2.75%	2.84%	2.87%	2.77%
TOTAL	**3.20%**	**3.20%**	**3.20%**	**3.20%**	**3.20%**	**3.20%**

Regional variation is moderate. Belgrade had the highest default rate (3.6% in average), while Novi Sad recorded the lowest (2.83% in average). Both SAS and Python reflected this pattern consistently across development, training and validation samples, meaning development samples are partitioned similar in both tools.

Default rate per exposure groups on development, training and validation samples in SAS and Python, are presented in Table 24.

Table 24 Default Rate per Exposure on Development, Training and Validation Samples in SAS and Python

EXPOSURE	SAS			PYTHON		
	DR DVL	DR TRN	DR VLD	DR DVL	DR TRN	DR VLD
EXP <= 1,000 EUR	5.33%	5.24%	5.52%	5.33%	5.42%	5.15%
1,000 EUR < EXP <= 2,500 EUR	3.53%	3.61%	3.36%	3.53%	3.52%	3.56%
2,500 EUR < EXP <= 6,000 EUR	3.14%	3.07%	3.31%	3.14%	3.28%	3.01%
6,000 EUR < EXP <= 12,500 EUR	2.81%	2.83%	2.76%	2.81%	2.71%	2.99%
12,500 EUR < EXP <= 25,000 EUR	2.49%	2.56%	2.31%	2.49%	2.47%	2.55%
25,000 EUR < EXP <= 50,000 EUR	2.60%	2.69%	2.38%	2.60%	2.66%	2.55%
EXP > 50,000 EUR	3.49%	3.36%	3.99%	3.49%	3.38%	3.69%
TOTAL	**3.20%**	**3.20%**	**3.20%**	**3.20%**	**3.20%**	**3.20%**

Clients with very small exposures (≤ 1,000 EUR) showed substantially higher default rates (5.15 – 5.52%). Mid-exposure ranges (6,000 - 25,000 EUR) were associated with lower risk (2.71 – 2.99%). High exposures (> 50,000 EUR) again show elevated default rates (3.36 - 3.99%), suggesting a U-shaped risk profile. Default rate per exposure has the

same distribution in SAS and Python, indicating that development samples are accurately partitioned in both environments.

Default rate per turnover groups on development, training and validation samples in SAS and Python, are presented in Table 25.

Table 25 Default Rate per Turnover on Development, Training and Validation Samples in SAS and Python

TURNOVER	SAS			PYTHON		
	DR DVL	DR TRN	DR VLD	DR DVL	DR TRN	DR VLD
TURNOVER <= 35,000 EUR	4.19%	4.25%	4.03%	4.19%	4.10%	4.30%
35,000 EUR < TURNOVER <= 75,000 EUR	3.52%	3.56%	3.41%	3.52%	3.69%	3.31%
75,000 EUR < TURNOVER <= 150,000 EUR	2.83%	2.75%	3.02%	2.83%	2.71%	3.03%
150,000 EUR < TURNOVER <= 300,000 EUR	3.20%	3.28%	3.03%	3.20%	3.35%	3.01%
300,000 EUR < TURNOVER <= 600,000 EUR	2.98%	2.98%	2.97%	2.98%	2.88%	3.03%
600,000 EUR < TURNOVER <= 1,000,000 EUR	2.83%	2.70%	3.13%	2.83%	2.85%	2.73%
TOTAL	**3.20%**	**3.20%**	**3.20%**	**3.20%**	**3.20%**	**3.20%**

Smaller firms (≤ 35,000 EUR turnover) were riskier (4.03 – 4.30%), while larger firms generally had lower default rates (2.7 – 2.99%). This highlights the stabilizing effect of size and diversification. Default rate per turnover is stable between development, training and validation samples in both SAS and Python, indicating the partition of development samples was done correctly.

Although SAS and Python produced small discrepancies in partitioning due to randomization, overall patterns were consistent across platforms. The analysis of default rate per risk drivers through development, training and validation samples confirms that stratification preserved the risk structure of the data, ensuring that training and validation samples remain representative of the development sample.

3.3.2. Population Stability Index (PSI)

The Population Stability Index (PSI) is widely applied to assess the stability of distributions across samples, ensuring that training and validation data remain representative of the development sample. For this thesis, PSI was calculated on the main risk drivers (year, organization form, region, exposure, turnover) using both SAS and Python. Three comparisons were evaluated: development (DVL) vs. training (TRN), development (DVL) vs. validation (VLD), and training (TRN) vs. validation (VLD). The results of PSI analysis are presented in Table 26.

Table 26 PSI by Count per Risk Drivers in SAS and in Python

RISK DRIVERS	SAS	PYTHON	SAS	PYTHON	SAS	PYTHON
	PSI BY COUNT DVL vs TRN		PSI BY COUNT DVL vs VLD		PSI BY COUNT TRN vs VLD	
YEAR	0.004000%	0.002492%	0.021500%	0.013680%	0.043900%	0.027848%
ORGANISATION FORM	0.001600%	0.002438%	0.008700%	0.013119%	0.017600%	0.026869%
REGION	0.001300%	0.003990%	0.006900%	0.021599%	0.014100%	0.044157%
EXPOSURE	0.004700%	0.009658%	0.025100%	0.052412%	0.051300%	0.107067%
TURNOVER	0.003600%	0.002915%	0.019600%	0.015848%	0.040100%	0.032356%

Interpretation of the results:

- SAS results: All PSI values are extremely low (well below 1%), demonstrating excellent sample stability.
- Python results: All PSI values are extremely low (well below 1%), demonstrating excellent sample stability.

Across both environments, PSI analysis confirms that partitioned data sets preserve the structure of the development sample. Minor deviations between SAS and Python reflect only implementation nuances, without introducing material bias. The results support the robustness of the data set for subsequent model development.

4. Univariate Analysis

Univariate analysis was performed on the training samples in both SAS and Python to identify variables with sufficient predictive strength and long-term stability. After the initial cleaning stage, a set of 123 WoE-transformed financial ratios formed the long list of candidates. This set was reduced using two standard filters applied consistently across both platforms:

- Predictive Power (Information Value, IV): variables with IV < 0.10 were excluded.
- Stability (Population Stability Index, PSI): variables with PSI > 0.10 in any of the tests (PSI single year or PSI time window) were excluded.

These filters are standard in credit risk modeling practice and ensure that variables entering multivariate modeling are predictive and stable.

The results of univariate analysis of the long list of variables are presented in Table 27. Information values in SAS and Python are presented for each variable on the long list, in descending order in terms of IV in Python. Illogical trends of variables are marked with not ok if violation of variable trend is presented. PSI in SAS and Python are marked with not ok if PSI > 0.1 in any of the tests. And finally short list in SAS and Python are presented in the last two columns with value 1 if variable in on the short list of variables.

Table 27 Long list of Variables with Results of Univariate Anaysis

VARIABLE	VARIABLE TYPE	IV SAS	IV in Python	Ilogical trend	PSI SAS	PSI Python	Short list SAS	Short list Python
VAR_19	LEVERAGE	0.37700	0.40112				1	1
VAR_21	LEVERAGE	0.32600	0.37765				1	1
VAR_29	PROFITABILITY	0.35800	0.36699				1	1
VAR_15	LEVERAGE	0.32200	0.33461				1	1
VAR_79	PROFITABILITY	0.29400	0.31926				1	1
VAR_24	LEVERAGE	0.30500	0.31010				1	1
VAR_34	LEVERAGE	0.33400	0.27730				1	1
VAR_82	PROFITABILITY	0.25800	0.27442				1	1
VAR_83	PROFITABILITY	0.25300	0.26716				1	1
VAR_25	LIQUIDITY	0.25800	0.25545				1	1
VAR_3	LIQUIDITY	0.22900	0.24604				1	1
VAR_1	LIQUIDITY	0.23500	0.24353				1	1
VAR_80	PROFITABILITY	0.20200	0.23983				1	1
VAR_78	ACTIVITY	0.21700	0.23921				1	1
VAR_61	PROFITABILITY	0.25500	0.23590				1	1
VAR_44	PROFITABILITY	0.23500	0.22760				1	1

VAR_22	LEVERAGE	0.21600	0.22098				1	1
VAR_18	LEVERAGE	0.21400	0.22088				1	1
VAR_59	PROFITABILITY	0.25600	0.22041				1	1
VAR_111	GROWTH	0.18500	0.21645	Not ok				
VAR_85	PROFITABILITY	0.11200	0.20917				1	1
VAR_26	LIQUIDITY	0.20300	0.20847				1	1
VAR_81	PROFITABILITY	0.19500	0.20707				1	1
VAR_60	ACTIVITY	0.16700	0.20598				1	1
VAR_43	PROFITABILITY	0.20600	0.20515				1	1
VAR_58	PROFITABILITY	0.20200	0.19978				1	1
VAR_98	CASH FLOW	0.18700	0.19184	Not ok				
VAR_8	LEVERAGE	0.17900	0.18467				1	1
VAR_110	GROWTH	0.21000	0.18435	Not ok				
VAR_16	PROFITABILITY	0.17500	0.18414				1	1
VAR_84	ACTIVITY	0.24200	0.18187				1	1
VAR_17	LIQUIDITY	0.16800	0.17434				1	1
VAR_35	LEVERAGE	0.18500	0.17153				1	1
VAR_113	GROWTH	0.18800	0.16838				1	1
VAR_42	PROFITABILITY	0.16200	0.16455				1	1
VAR_120	GROWTH	0.16300	0.15875	Not ok				
VAR_112	GROWTH	0.24000	0.15387				1	1
VAR_48	PROFITABILITY	0.14400	0.15336				1	1
VAR_105	CASH FLOW	0.23300	0.14868	Not ok				
VAR_49	PROFITABILITY	0.14100	0.14784				1	1
VAR_96	CASH FLOW	0.09600	0.14095					1
VAR_13	LIQUIDITY	0.14000	0.13879				1	1
VAR_77	PROFITABILITY	0.13400	0.13697				1	1
VAR_53	PROFITABILITY	0.14200	0.13688				1	1
VAR_97	CASH FLOW	0.09400	0.13667					1
VAR_54	PROFITABILITY	0.14100	0.13502				1	1
VAR_101	CASH FLOW	0.20300	0.13433	Not ok				
VAR_52	PROFITABILITY	0.13900	0.13158				1	1
VAR_50	PROFITABILITY	0.13500	0.12808	Not ok				
VAR_51	ACTIVITY	0.13400	0.12775				1	1
VAR_12	LEVERAGE	0.14000	0.12399	Not ok				
VAR_27	LEVERAGE	0.11100	0.12200				1	1
VAR_91	PROFITABILITY	0.16300	0.12053	Not ok				
VAR_14	LIQUIDITY	0.11900	0.11986				1	1
VAR_31	LEVERAGE	0.14400	0.11861				1	1
VAR_36	LEVERAGE	0.14300	0.11799				1	1
VAR_32	LEVERAGE	0.14900	0.11633				1	1
VAR_121	GROWTH	0.12600	0.11510				1	1
VAR_116	GROWTH	0.10900	0.11304	Not ok				
VAR_6	LIQUIDITY	0.07900	0.10770					1
VAR_73	PROFITABILITY	0.07500	0.10756					1
VAR_109	GROWTH	0.10700	0.10527	Not ok				
VAR_107	CASH FLOW	0.10100	0.10003	Not ok				

VAR_90	PROFITABILITY	0.17000	0.09997	Not ok				
VAR_69	PROFITABILITY	0.07800	0.09861					
VAR_72	LIQUIDITY	0.08300	0.09822					
VAR_7	LIQUIDITY	0.06800	0.09800					
VAR_102	CASH FLOW	0.03600	0.09785					
VAR_100	CASH FLOW	0.05700	0.09752					
VAR_47	PROFITABILITY	0.09400	0.09615					
VAR_86	PROFITABILITY	0.07500	0.09403					
VAR_115	GROWTH	0.09700	0.09387					
VAR_28	LEVERAGE	0.11600	0.09189				1	
VAR_87	PROFITABILITY	0.10100	0.09178	Not ok				
VAR_62	PROFITABILITY	0.23900	0.09146				1	
VAR_122	GROWTH	0.07500	0.09029		Not ok			
VAR_30	LIQUIDITY	0.14000	0.08771	Not ok				
VAR_93	PROFITABILITY	0.14000	0.08754	Not ok				
VAR_117	GROWTH	0.10300	0.08725	Not ok				
VAR_92	PROFITABILITY	0.13200	0.08708	Not ok				
VAR_106	CASH FLOW	0.10000	0.08624				1	
VAR_55	LIQUIDITY	0.07100	0.08190					
VAR_46	PROFITABILITY	0.07400	0.08113					
VAR_56	ACTIVITY	0.09200	0.08046					
VAR_89	PROFITABILITY	0.13200	0.08002	Not ok				
VAR_88	PROFITABILITY	0.16000	0.07946				1	
VAR_103	CASH FLOW	0.09600	0.07811		Not ok			
VAR_45	PROFITABILITY	0.10700	0.07777				1	
VAR_57	ACTIVITY	0.07000	0.07692					
VAR_95	CASH FLOW	0.05100	0.07083					
VAR_5	LEVERAGE	0.04900	0.07019					
VAR_63	PROFITABILITY	0.05900	0.06854					
VAR_67	PROFITABILITY	0.03800	0.06753					
VAR_74	PROFITABILITY	0.11400	0.06666	Not ok				
VAR_38	PROFITABILITY	0.10700	0.06598	Not ok				
VAR_39	PROFITABILITY	0.06000	0.06598					
VAR_37	LEVERAGE	0.03300	0.06414					
VAR_75	ACTIVITY	0.06000	0.06359					
VAR_68	PROFITABILITY	0.05100	0.06315					
VAR_114	GROWTH	0.08100	0.06239					
VAR_76	ACTIVITY	0.06100	0.06165					
VAR_41	PROFITABILITY	0.12900	0.06038				1	
VAR_94	ACTIVITY	0.05300	0.06022					
VAR_71	LEVERAGE	0.03900	0.05857					
VAR_99	CASH FLOW	0.05800	0.05847		Not ok			
VAR_33	LIQUIDITY	-	0.05452					
VAR_66	PROFITABILITY	0.04500	0.05406					
VAR_4	LIQUIDITY	0.06700	0.05307					
VAR_20	LEVERAGE	0.05500	0.04742					
VAR_123	GROWTH	0.08700	0.04445					

VAR_119	GROWTH	0.04900	0.04199					
VAR_118	GROWTH	0.04300	0.03802					
VAR_40	PROFITABILITY	0.03400	0.03633					
VAR_70	PROFITABILITY	0.02500	0.03483					
VAR_104	CASH FLOW	0.04000	0.03387					
VAR_23	LEVERAGE	0.02000	0.02758					
VAR_11	LIQUIDITY	0.02400	0.02694					
VAR_65	PROFITABILITY	0.11500	0.02491	Not ok				
VAR_2	LEVERAGE	0.01600	0.02319					
VAR_9	LIQUIDITY	0.00700	0.02131					
VAR_64	PROFITABILITY	0.09600	0.01475					
VAR_10	LEVERAGE	0.01400	0.00648					
VAR_108	GROWTH	0.03000	0.00156					

4.1. Information Value

The Weight of Evidence (WoE) transformation was performed using integrated WoE approach in SAS (Interactive grouping node in SAS EM) and custom-built WoE automatic binning algorithm in Python. WoE grouping aimed to maximize the Information Value (IV) of each variable while maintaining monotonicity.

- In SAS, 48 variables were excluded due to IV < 0.10. Out of these, 44 were also excluded in Python, while 4 had borderline IV values (0.10 – 0.15).
- In Python, 60 variables were excluded due to IV < 0.10. Out of these, 44 matched SAS results, 10 were additionally flagged for illogical patterns in SAS, and 6 had IV values between 0.10 and 0.25 in SAS.
- SAS identified 22 variables with illogical patterns, out of which 10 were also excluded in Python due to weak IV.

Despite different training samples and independent WoE grouping algorithms, using integrated WoE function in SAS and custom-built WoE automatic binning algorithm in Python, information values (IV) rankings were broadly consistent across both tools, supporting the reliability of the exclusion process.

4.2. Population Stability Index

Two stability tests were conducted:

- PSI Single Year: comparing the distribution in 2023 against each previous single year.

- PSI Time Window: comparing 2023 against aggregated distributions of 2019–2022.

Variables were considered stable when PSI values were below 10% in both tests.

- In SAS, all remaining variables passed the PSI criteria, leaving 53 variables.
- In Python, all remaining variables were also stable, leaving 51 variables.

In accordance with integrated PSI functions in SAS and custom-built PSI tests in Python, the results confirmed that WoE transformed variables exhibited stable distributions over time, reducing the risk of model deterioration.

4.3. Variable Selection in SAS and Python

The final short list emerged after applying all exclusion criteria among the different platforms and on different training samples, in SAS and Python are summarized in Table 28.

Table 28 Exclusion Criteria for Long list of Variables both in SAS and Python

	SAS			PYTHON		
Exclusion Criteria	**Start number of variables**	**Number of excluded variables**	**Number of remaining variables**	**Start number of variables**	**Number of excluded variables**	**Number of remaining variables**
IV < 0.1	123	48	75	123	60	63
Illogical trend	75	22	53	63	12	51
PSI > 10%	53	0	53	51	0	51
TOTAL	**123**	**70**	**53**	**123**	**72**	**51**

Based on the results of information values on WoE grouping in SAS and Python, despite differences in training samples and different codes used for WoE grouping (completely integrated codes in SAS and custom-built codes in Python), the predictive power of WoE variables, as measured by IV, is largely consistent across both platforms.

Considering the results of illogical trend of variables, the conclusion is that Python had better automatic recognition of variables with illogical trend. In Python, 10 variables out of 22 with illogical trend, were regonized automaticaly though insuficient information values.

The PSI analysis in both SAS and Python showed that all remaining variables were stable (according to integrated PSI functions in SAS and custom-built PSI tests in Python)

leaving 53 variables in SAS and 51 variables in Python (apart from ID, datetime variable and target variable) ready for the clustering phase and remained as potential final model candidates.

Variable selection methods both in SAS and Python converged toward compact sets of predictors, there are 53 variables on short list in SAS and 51 variables on short list in Python.

5. Multivariate Analysis

Following the definition of the short list of variables in SAS and Python (in chapter 4), multivariate analysis was next and variable clustering was first methodological step within:

- In SAS, variable clustering was performed using integrated function for variable clustering (variable clustering node in SAS EM).
- In Python, variable grouping was performed using custom-built correlation-based algorithm for grouping.

Logistic regression was employed to develop the models. The development process applied a brute force approach, testing numerous combinations of candidate models and retaining only those models that satisfied predefined validity criteria.

From these candidate models, the final models were selected separately in SAS and Python and then compared to assess performance, stability, and parsimony.

5.1. Variables Clustering

The clustering procedure was executed using the training samples, which contained the 53 WoE transformed variables on short list in SAS and 51 WoE transformed variables on short list in Python, that survived the univariate analysis phase. The goal of the clustering process was to group similar variables together based on their correlations and predictive power, effectively reducing redundancy while preserving the most informative features for the model. By grouping variables with similar behavior, the clustering process helped in identifying variables that provide unique information, which is essential for enhancing the model's ability to predict default accurately.

Following IV and PSI filtering in univariate analysis, variables were grouped to reduce redundancy in multivariate analysis.

- In SAS, clustering excluded 41 variables. Of these, 27 were also excluded in Python due to high correlation, 6 were already dropped in Python for low IV, and 8 remained in Python despite being excluded in SAS.

- In Python, correlation filtering excluded 35 variables with |r| > 0.65. Of these, 27 overlapped with SAS exclusions, 1 was excluded in SAS for IV < 0.1, while 7 were retained in SAS.

After the clustering phase in SAS, using variable clustering node in SAS EM, 12 variables remained on the final short list for modeling phase.

After the variable grouping phase in Python, using correlation-based algorithm, custom-built in Python, 16 variables remained on the final short list for modeling phase.

This comparison shows that Python's correlation-based grouping was slightly stricter and better in automatically recognition of correlated variables, while SAS relied on clustering, which sometimes retained correlated variables.

5.2. Correlation Analysis

Correlation matrices were examined to detect potential multicollinearity among the short-listed variables.

The correlations matrix of the 12 short-listed variables in SAS is shown in Figure 2.

Pearson Correlation Coefficients, Prob > |r| under H0: Rho=0	WOE_VAR_49	WOE_VAR_83	WOE_VAR_44	WOE_VAR_19	WOE_VAR_27	WOE_VAR_79	WOE_VAR_113	WOE_VAR_80	WOE_VAR_43	WOE_VAR_26	WOE_VAR_17	WOE_VAR_112
WOE_VAR_49	100.00%	80.41%	68.96%	31.11%	10.93%	24.17%	15.42%	46.06%	42.27%	22.80%	8.94%	7.32%
WOE_VAR_83	80.41%	100.00%	70.28%	45.83%	15.74%	46.19%	17.73%	62.33%	47.87%	37.93%	17.84%	11.71%
WOE_VAR_44	68.96%	70.28%	100.00%	27.03%	7.33%	23.70%	15.46%	33.34%	65.42%	20.15%	14.35%	11.54%
WOE_VAR_19	31.11%	45.83%	27.03%	100.00%	40.52%	42.47%	10.20%	39.92%	25.58%	64.41%	26.74%	7.95%
WOE_VAR_27	10.93%	15.74%	7.33%	40.52%	100.00%	19.45%	2.31%	25.09%	16.42%	9.84%	-0.88%	3.70%
WOE_VAR_79	24.17%	46.19%	23.70%	42.47%	19.45%	100.00%	30.90%	53.77%	31.29%	32.03%	23.34%	13.97%
WOE_VAR_113	15.42%	17.73%	15.46%	10.20%	2.31%	30.90%	100.00%	23.47%	18.59%	8.60%	8.88%	47.43%
WOE_VAR_80	46.06%	62.33%	33.34%	39.92%	25.09%	53.77%	23.47%	100.00%	54.49%	29.95%	13.00%	11.95%
WOE_VAR_43	42.27%	47.87%	65.42%	25.58%	16.42%	31.29%	18.59%	54.49%	100.00%	16.61%	13.38%	10.86%
WOE_VAR_26	22.80%	37.93%	20.15%	64.41%	9.84%	32.03%	8.60%	29.95%	16.61%	100.00%	14.85%	5.70%
WOE_VAR_17	8.94%	17.84%	14.35%	26.74%	-0.88%	23.34%	8.88%	13.00%	13.38%	14.85%	100.00%	7.91%
WOE_VAR_112	7.32%	11.71%	11.54%	7.95%	3.70%	13.97%	47.43%	11.95%	10.86%	5.70%	7.91%	100.00%

Figure 2 Pearson Correlation Matrix of Short-listed Variables in SAS

The correlations matrix of the 16 short-listed variables in Python is shown in Figure 3.

Pearson Correlation Coefficients, Prob > |r| under H0: Rho=0	WOE_VAR_19	WOE_VAR_79	WOE_VAR_82	WOE_VAR_3	WOE_VAR_78	WOE_VAR_43	WOE_VAR_16	WOE_VAR_113	WOE_VAR_96	WOE_VAR_53	WOE_VAR_27	WOE_VAR_14	WOE_VAR_31	WOE_VAR_121	WOE_VAR_6	WOE_VAR_73
WOE_VAR_19	100.00%	46.64%	46.31%	51.21%	40.18%	24.99%	59.95%	7.78%	22.46%	25.55%	41.12%	53.53%	32.27%	17.07%	51.15%	41.90%
WOE_VAR_79	46.64%	100.00%	51.36%	36.04%	60.83%	30.48%	35.38%	32.53%	32.02%	21.91%	20.91%	25.78%	23.87%	13.68%	35.18%	24.74%
WOE_VAR_82	46.31%	51.36%	100.00%	32.18%	48.93%	45.48%	39.70%	21.65%	40.65%	24.68%	13.87%	30.64%	18.87%	46.00%	35.69%	38.48%
WOE_VAR_3	51.21%	36.04%	32.18%	100.00%	33.24%	17.73%	30.80%	8.81%	12.28%	59.23%	12.20%	30.30%	56.06%	5.60%	42.79%	19.31%
WOE_VAR_78	40.18%	60.83%	48.93%	33.24%	100.00%	19.72%	26.48%	22.78%	22.51%	18.40%	4.93%	26.91%	24.78%	17.66%	29.74%	28.69%
WOE_VAR_43	24.99%	30.48%	45.48%	17.73%	19.72%	100.00%	21.72%	21.40%	42.24%	29.59%	17.50%	12.78%	11.89%	23.02%	16.45%	29.74%
WOE_VAR_16	59.95%	35.38%	39.70%	30.80%	26.48%	21.72%	100.00%	4.76%	21.07%	15.83%	24.00%	64.02%	11.28%	13.45%	40.65%	35.25%
WOE_VAR_113	7.78%	32.53%	21.65%	8.81%	22.78%	21.40%	4.76%	100.00%	37.83%	19.88%	1.82%	4.94%	5.77%	35.90%	3.68%	21.39%
WOE_VAR_96	22.46%	32.02%	40.65%	12.28%	22.51%	42.24%	21.07%	37.83%	100.00%	38.04%	18.19%	13.49%	2.91%	37.88%	13.26%	42.87%
WOE_VAR_53	25.55%	21.91%	24.68%	59.23%	18.40%	29.59%	15.83%	19.88%	38.04%	100.00%	3.60%	15.79%	38.03%	16.21%	16.10%	47.91%
WOE_VAR_27	41.12%	20.91%	13.87%	12.20%	4.93%	17.50%	24.00%	1.82%	18.19%	3.60%	100.00%	0.61%	8.36%	4.07%	22.81%	6.89%
WOE_VAR_14	53.53%	25.78%	30.64%	30.30%	26.91%	12.78%	64.02%	4.94%	13.49%	15.79%	0.61%	100.00%	10.23%	11.05%	43.38%	51.93%
WOE_VAR_31	32.27%	23.87%	18.87%	56.06%	24.78%	11.89%	11.28%	5.77%	2.91%	38.03%	8.36%	10.23%	100.00%	2.53%	8.07%	7.67%
WOE_VAR_121	17.07%	13.68%	46.00%	5.60%	17.66%	23.02%	13.45%	35.90%	37.88%	16.21%	4.07%	11.05%	2.53%	100.00%	6.64%	29.34%
WOE_VAR_6	51.15%	35.18%	35.69%	42.79%	29.74%	16.45%	40.65%	3.68%	13.26%	16.10%	22.81%	43.38%	8.07%	6.64%	100.00%	27.40%
WOE_VAR_73	41.90%	24.74%	38.48%	19.31%	28.69%	29.74%	35.25%	21.39%	42.87%	47.91%	6.89%	51.93%	7.67%	29.34%	27.40%	100.00%

Figure 3 Pearson Correlation Matrix of Short-listed Variables in Python

- In SAS, three variable pairs exceeded the 65% correlation threshold. These required further consideration to avoid instability in logistic regression.
- In Python, no short-listed variables exceeded the 65% threshold after correlation-based grouping, demonstrating more effective elimination of redundant predictors.

This highlights that Python's self-automated correlation-based grouping algorithm provided slightly stronger safeguards against multicollinearity compared to SAS's integrated clustering procedure (using variable clustering node in SAS EM).

5.3. Brute Force Approach

The brute force procedure systematically estimated logistic regression models on the training sample using all feasible combinations of short-listed predictors, in both SAS and Python. Brute-force approach is implemented in SAS with aim to make all possible combinations of potential models with four to seven variables from the short list. In Python, brute-force approach is integrated differently, with aim of making combinations of only valid models from two to ten variables from the short list of variables. Only models meeting the following validity criteria were retained:

- Statistical significance: all coefficients with *p*-values < 0.05.
- Economic consistency: all coefficients negative (due to WoE transformation).
- Technical validity: maximum likelihood estimation (MLE) converged successfully.

Models satisfying these conditions were ranked by discriminatory power (Gini) and assessed for parsimony and collinearity. Validation-sample performance of the models was then used to confirm stability and prevent overfitting.

In SAS, models with 4 to 7 variables were estimated, producing more than 25,000 potential models, of which around 13,000 were valid (Table 29).

In Python, a custom-built procedure generated 283 valid models with 2 to 10 variables, of which all satisfied the validity checks (Table 30).

Table 29 Number of Models Developed with Brute Force Approach in SAS

Number of variables in the model	Number of trial models	Number of valid models
4	3,131	2,470
5	6,490	4,181
6	8,643	4,148
7	7,049	2,293
TOTAL	**25,313**	**13,092**

Table 30 Number of Models Developed with Brute Force Approach in Python

Number of variables in the model	Number of valid models
2	40
3	40
4	40
5	40
6	40
7	40
8	28
9	13
10	2
TOTAL	**283**

Although SAS generated a larger pool of candidate models, both approaches applied identical validity filters, ensuring comparability of the final selections.

5.3.1. Comparison of Brute Force Approach in SAS and Python

The differences in model counts reflect methodological choices: SAS performed an exhaustive brute-force search without early filtering, while Python employed a structured, stepwise expansion process with integrated correlation and significance checks. Both approaches ultimately converged on robust models, but Python's method illustrates how open-source pipelines can be customized to enforce stricter quality controls throughout

the modeling process. The comparison of brute force approach in SAS and Python is shown in Table 31.

Table 31 Comparison of Brute Force Approach in SAS and Python

Aspect	SAS Implementation	Python Implementation
Search strategy	Exhaustive brute-force: made all possible combinations of models with 4–7 variables from the short list	Iterative pipeline: brute-force started with combinations of models with 2 variables from the short list, expanded step by step with combinations of models with up to 10 variables from the short list in accordance with predefined rules
Multicollinearity control	No early filtering; correlations checked only after model estimation	65% correlation threshold applied at every stage to discard collinear combinations
Statistical criteria	Variables and models' significance checked post-estimation (p-value < 0.05)	Variables and models' significance checked during pipeline (p-value < 0.05)
Economic interpretability	Evaluated after brute-force search	Built-in rule: only negative coefficients retained (consistent with risk logic)
Candidate models generated	> 25,000 potential model combinations with 4–7 variables	283 valid model combinations with 2–10 variables
Valid models retained	~ 13,000	283
Computational cost	High (long runtime, heavy resources)	Moderate (efficient pruning of weak/collinear models)
Strengths	Comprehensive coverage of specifications	Efficient, customizable, avoids redundant/invalid models
Limitations	Computationally expensive, including many redundant models	Smaller model set, risk of missing some marginal specifications

While SAS produced a significantly larger number of candidate models due to its exhaustive brute-force search, this approach also resulted in high computational costs and many redundant specifications. By contrast, the Python pipeline applied stricter criteria during the model-building process - including correlation thresholds, statistical significance checks, and economic interpretability rules (negative betas) - leading to a smaller but more targeted set of candidate models. This difference reflects a trade-off between comprehensiveness (SAS) and efficiency (Python): SAS explores a broader search space, whereas Python eliminates weak or collinear specifications earlier, thereby improving interpretability and reducing computational burden.

5.4. Final SB Financial Model in SAS

The SAS brute force procedure identified a seven-variable model as the optimal specification. The final model achieved a GINI of 0.4499 and AUC of 0.725, consistent with accepted benchmarks for credit scoring models.

Variables in the final model in SAS are presented in Table 32, along with the group of financial ratios where the chosen financial ratio belongs:

Table 32 List of Variables in the Final Model in SAS

Variable name	Description
VAR_113	Var_113 belongs to Growth ratios
VAR_19	Var_19 belongs to Leverage ratios
VAR_17	Var_17 belongs to Liquidity ratios
VAR_43	Var_43 belongs to Profitability ratios
VAR_27	Var_27 belongs to Leverage ratios
VAR_26	Var_26 belongs to Liquidity ratios
VAR_79	Var_79 belongs to Profitability ratios

The SAS brute-force search selected a seven-variable specification spanning growth (VAR_113), leverage (VAR_19, VAR_27), liquidity (VAR_17, VAR_26) and profitability (VAR_43, VAR_79) ratios. This composition captures the principal channels through which Small Business companies transition to distress - capital structure pressure, short-term solvency, earnings capacity, and momentum - thereby supporting both statistical performance and business interpretability.

The correlation matrix between variables in the final model in SAS is presented in Table 33.

Table 33 Pearson Correlation Matrix of the Variables in the Final Model in SAS

Pearson Correlation Coefficients, Prob > |r| under H0: Rho=0	WOE_VAR_113	WOE_VAR_19	WOE_VAR_17	WOE_VAR_43	WOE_VAR_27	WOE_VAR_26	WOE_VAR_79
WOE_VAR_113	100.00%	10.20%	8.88%	18.59%	2.31%	8.60%	30.90%
WOE_VAR_19	10.20%	100.00%	26.74%	25.58%	40.52%	64.41%	42.47%
WOE_VAR_17	8.88%	26.74%	100.00%	13.38%	-0.88%	14.85%	23.34%
WOE_VAR_43	18.59%	25.58%	13.38%	100.00%	16.42%	16.61%	31.29%
WOE_VAR_27	2.31%	40.52%	-0.88%	16.42%	100.00%	9.84%	19.45%
WOE_VAR_26	8.60%	64.41%	14.85%	16.61%	9.84%	100.00%	32.03%
WOE_VAR_79	30.90%	42.47%	23.34%	31.29%	19.45%	32.03%	100.00%

The correlation matrix shows a maximum pairwise r = 64.41% (WOE_VAR_19 vs. WOE_VAR_26), marginally below the 65% screening threshold, all other pairwise correlations are lower. Consequently, the final set meets the pre-specified collinearity constraint without sacrificing coverage of distinct financial mechanisms. Given the near-

threshold pair, routine post-implementation monitoring of variables stability is advisable, but no remedial action is indicated at this stage.

The results of fit statistics of Final SB Financial model in SAS on the training and validation sample are presented in Table 34.

Table 34 Fit Statistics of the Final Model in SAS

Target	Fit Statistics	Statistics Label	Training	Validation
DEF_FLG	AIC	Akaike's Information Criterion	9,754.56	
DEF_FLG	BIC	Bayesian Information Criterion	9,819.81	
DEF_FLG	MISC	Misclassification Rate	0.0525	0.0526
DEF_FLG	KS	Kolmogorov-Smirnov Statistic	0.3356	0.3453
DEF_FLG	AUC	Area Under Curve	0.7250	0.7228
DEF_FLG	Gini	Gini Coefficient	0.4499	0.4457
DEF_FLG	AR	Accuracy Ratio	0.4499	0.4457

The model achieves AUC = 0.7250 (training) / 0.7228 (validation) and Gini = 0.4499 / 0.4457, with KS = 0.3356 / 0.3453 and nearly identical misclassification rates (0.0525 / 0.0526). All metrics are above thresholds for building acceptable models. The extremely small training – validation drift indicates limited optimism and sound generalization. Reported AIC (9,754.56) and BIC (9,819.81) on training sample are consistent with a parsimonious well-converged solution and were used for internal ranking during the SAS model search. The ROC curves of the final model in SAS on training and validation samples are shown in Figure 4.

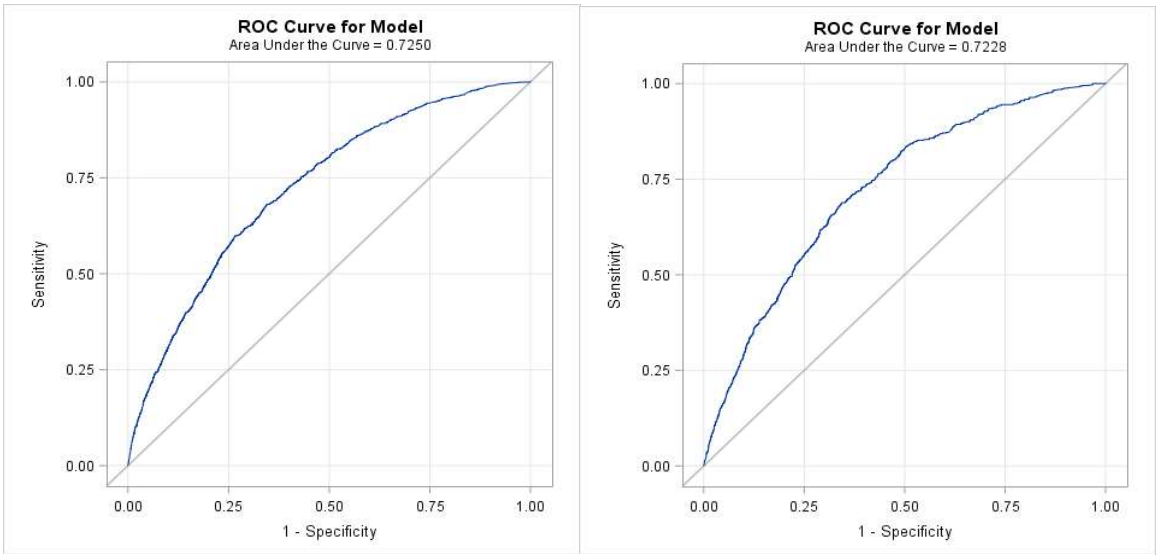


Figure 4 ROC Curves of the Final Model in SAS on Training and Validation Samples

The results of SAS logistic regression significance on training and validation samples are presented in Table 35.

Table 35 Testing Null Hypothesis of the Final Model in SAS on the Training and Validation Samples

Training sample				Validation sample			
Testing Global Null Hypothesis: BETA=0				Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq	Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	863.4025	7	<.0001	Likelihood Ratio	346.4945	7	<.0001
Score	882.0422	7	<.0001	Score	362.2739	7	<.0001
Wald	792.7515	7	<.0001	Wald	329.7712	7	<.0001

Likelihood-ratio, Score, and Wald tests decisively reject the null hypothesis of no explanatory power on both training and validation samples ($p < 0.0001$). Meaning that final model in SAS is significant on both training and validation data sets.

The results of parameters significant of final model in SAS on training sample, are presented in Table 36.

Table 36 Analysis of Maximum Likelihood Estimates of Variables on Training Sample in SAS

Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq	Variable Weights
Intercept	1	- 2.8968	0.0304	9,080.8797	<.0001	
WOE_VAR_113	1	- 0.6211	0.0675	84.7496	<.0001	17.40%
WOE_VAR_19	1	- 0.4090	0.0689	35.2425	<.0001	16.73%
WOE_VAR_17	1	- 0.6078	0.0723	70.6076	<.0001	16.01%
WOE_VAR_43	1	- 0.4824	0.0679	50.4819	<.0001	14.86%
WOE_VAR_27	1	- 0.5652	0.1080	27.3875	<.0001	13.91%
WOE_VAR_26	1	- 0.3361	0.0910	13.6365	0.0002	11.08%
WOE_VAR_79	1	- 0.2935	0.0596	24.2512	<.0001	10.02%
					Total	**100.00%**

The results of parameters significant of final model in SAS on validation sample, are presented in Table 37.

Table 37 Analysis of Maximum Likelihood Estimates of Variables on Validation Sample in SAS

Parameter	DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq	Weight of variables
Intercept	1	- 2.8968	0.0464	3,904.6902	<.0001	
WOE_VAR_113	1	- 0.6211	0.1031	36.2809	<.0001	17.51%
WOE_VAR_19	1	- 0.4090	0.1052	15.1286	0.0001	16.61%
WOE_VAR_17	1	- 0.6078	0.1098	30.6515	<.0001	16.10%
WOE_VAR_43	1	- 0.4824	0.1033	21.8231	<.0001	14.88%

WOE_VAR_27	1	- 0.5652	0.1644	11.8266	0.0006	13.83%
WOE_VAR_26	1	- 0.3361	0.1395	5.8045	0.016	11.02%
WOE_VAR_79	1	- 0.2935	0.0909	10.4175	0.0012	10.05%
					Total	100.00%

All variables are significant and have the expected negative sign under WoE transformation on training sample. Also all variables remain significant and have negative sings on validation, out-of-sample, indicating that the direction of risk implied by each predictor is robust. Variable contribution shares are stable: on the training sample, the largest shares are VAR_113 (≈17.4%), VAR_19 (≈16.7%), and VAR_17 (≈16.0%); on validation, the ordering and magnitudes are essentially preserved. This cross-sample alignment reduces concerns about overfitting and supports deployment.

- Growth (VAR_113): The largest individual contribution aligns with the notion that decelerating growth (or contraction) precedes liquidity stress and rising default risk.
- Leverage (VAR_19, VAR_27): Both proxies capture balance-sheet fragility; their combined presence suggests independent information beyond a single leverage gauge.
- Liquidity (VAR_17, VAR_26): Short-term repayment capacity remains a distinct and material dimension even after controlling profitability and leverage.
- Profitability (VAR_43, VAR_79): Earnings capacity provides additional separation, consistent with the empirical link between sustained margins and survivability for Small Businesses.

Coefficients remain statistically significant on validation sample, retain the expected negative sign under WoE transformation, and exhibit consistent relative weights. This stability suggests that model is generalizable and not over-fit with its respective training sample.

5.5. Final SB Financial Model in Python

The final SB Financial Model in Python was developed on the training sample using a custom-built brute force logistic regression approach. From 283 valid models with 2 to 10 variables, the optimal specification was a six-variable model, balancing predictive performance, parsimony, and statistical validity. The model achieved a Gini coefficient of

0.4506 and AUC of 0.7253, fully consistent with accepted benchmarks for credit scoring models.

Variables in the final model in Python are presented in Table 38, along with the group of financial ratios where the chosen financial ratio belongs.

Table 38 List of Variables in the Final Model in Python

Variable name	Description
VAR_19	Var_19 belongs to Leverage ratios
VAR_43	Var_43 belongs to Profitability ratios
VAR_3	Var_3 belongs to Liquidity ratios
VAR_113	Var_113 belongs to Growth ratios
VAR_27	Var_27 belongs to Leverage ratios
VAR_78	Var_78 belongs to Activity ratios

The final model in Python incorporates variables across five ratio categories, ensuring broad coverage of financial dimensions. This structure highlights a well-diversified set of financial drivers, with profitability and leverage as core anchors, complemented by liquidity, growth, and operating efficiency factors.

The correlation matrix between variables in the final model in Python is presented in Table 39.

Table 39 Pearson Correlation Matrix of the Variables in the Final Model in Python

Pearson Correlation Coefficients, Prob > |r| under H0: Rho=0	WOE_VAR_19	WOE_VAR_43	WOE_VAR_3	WOE_VAR_113	WOE_VAR_27	WOE_VAR_78
WOE_VAR_19	100.00%	24.99%	51.21%	7.78%	41.12%	40.18%
WOE_VAR_43	24.99%	100.00%	17.73%	21.40%	17.50%	19.72%
WOE_VAR_3	51.21%	17.73%	100.00%	8.81%	12.20%	33.24%
WOE_VAR_113	7.78%	21.40%	8.81%	100.00%	1.82%	22.78%
WOE_VAR_27	41.12%	17.50%	12.20%	1.82%	100.00%	4.93%
WOE_VAR_78	40.18%	19.72%	33.24%	22.78%	4.93%	100.00%

The correlation matrix (Table 39) confirms that all pairwise coefficients remain below the 0.65 threshold after correlation-based grouping. This demonstrates that multicollinearity is effectively managed, with the tightest association being 51.2% between VAR_19 (leverage) and VAR_3 (liquidity).

The results of fit statistics of final SB financial model in Python on the training and validation sample are presented in Table 40.

Table 40 Fit Statistics of the Final Model in Python

Target	Fit Statistics	Statistics Label	Training	Validation
DEF_FLG	AIC	Akaike's Information Criterion	9,765.90	
DEF_FLG	BIC	Bayesian Information Criterion	9,823.00	
DEF_FLG	MISC	Misclassification Rate	0.05250	0.05237
DEF_FLG	KS	Kolmogorov-Smirnov Statistic	0.34216	0.33960
DEF_FLG	AUC	Area Under ROC	0.72532	0.71414
DEF_FLG	Gini	Gini Coefficient	0.45065	0.42828
DEF_FLG	AR	Accuracy Ratio	0.45065	0.42828

The ROC curves of the final model in Python are shown in Figure 5.

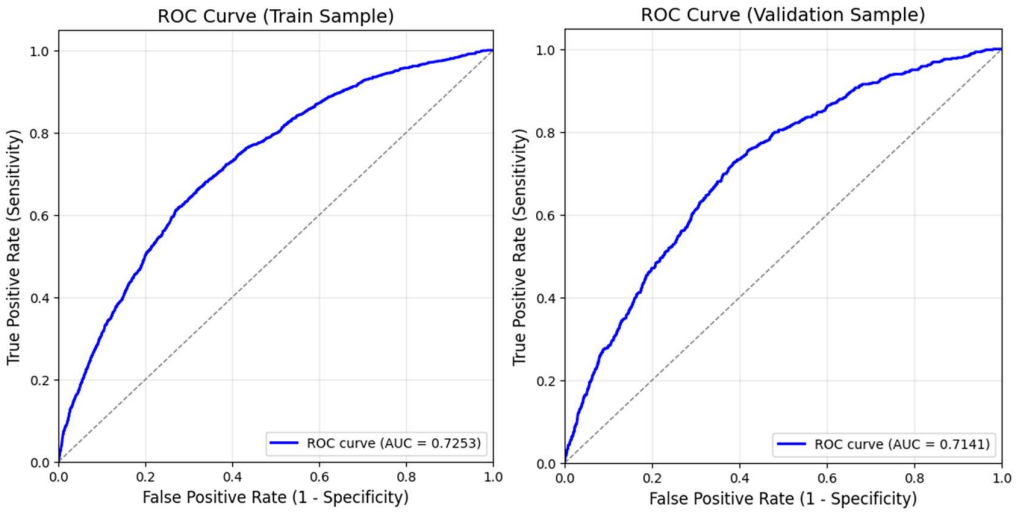


Figure 7 ROC Curve on Traning and Validation Samples in SAS and Python

Fit statistics on the training and validation samples are summarized in Table 41.

- Discrimination: On training, AUC = 0.7253 (Gini = 0.4506); on validation, AUC = 0.7141 (Gini = 0.4283). While the validation AUC and Gini are slightly lower, the drop is modest and within expected ranges, indicating good generalization.
- KS statistic: KS values are highly consistent (0.3422 training vs. 0.3396 validation), suggesting stable discriminatory power across samples.
- Misclassification rate: Nearly identical on training (5.25%) and validation (5.24%) samples, confirming balanced predictive error.

- Information criteria: AIC (9,765.90) and BIC (9,823.00) support model compactness relative to candidate alternatives.

Table 41 Testing Null Hypothesis of the Final Model in Python on the Training and Validation Samples

Training sample				Validation sample			
Testing Global Null Hypothesis: BETA=0				Testing Global Null Hypothesis: BETA=0			
Test	**z**	**DF**	**Pr > | z |**	**Test**	**z**	**DF**	**Pr > | z |**
Likelihood Ratio	-5304	6	0.0000	Likelihood Ratio	-2267.4	6	0.000

Global null hypothesis tests (Table 42) strongly reject the hypothesis that all coefficients are zero ($p < 0.001$ on both training and validation). This validates the joint explanatory power of the selected predictors.

The results of parameters significance of the final model in Python on training sample, are presented in Table 42.

Table 42 Analysis of Maximum Likelihood Estimates of Variables on Training Sample in Python

Parameter	DF	Estimate	Standard Error	z	P > |z|	Variable Weight
Intercept	1	-2.8968	0.0300	-95.4210	0.0000	
WOE_VAR_19	1	-0.4968	0.0580	-8.6230	0.0000	22.56%
WOE_VAR_43	1	-0.5038	0.0700	-7.2490	0.0000	16.93%
WOE_VAR_3	1	-0.4552	0.0700	-6.5350	0.0000	16.56%
WOE_VAR_113	1	-0.6283	0.0670	-9.3660	0.0000	16.44%
WOE_VAR_27	1	-0.5202	0.0990	-5.2440	0.0000	13.96%
WOE_VAR_78	1	-0.4134	0.0630	-6.5470	0.0000	13.55%
					Total	100.00%

The results of parameters significance of the final model in SAS on validation sample, are presented in Table 43.

Table 43 Analysis of Maximum Likelihood Estimates of Variables on Validaion Sample in Python

Parameter	DF	Estimate	Standard Error	Z	P > |z|	Variable Weight
Intercept	1	-2.8885	0.0460	-62.9950	0.0000	
WOE_VAR_19	1	-0.4741	0.0870	-5.4250	0.0000	23.33%
WOE_VAR_43	1	-0.4553	0.1050	-4.3250	0.0000	16.66%
WOE_VAR_3	1	-0.5357	0.1060	-5.0580	0.0000	21.14%
WOE_VAR_113	1	-0.6238	0.1030	-6.0400	0.0000	17.61%

WOE_VAR_27	1	-0.3536	0.1480	-2.3930	0.0170	10.38%
WOE_VAR_78	1	-0.3009	0.0960	-3.1420	0.0020	10.88%
					Total	100.00%

Across training (Table 42) and validation (Table 43) samples, all coefficients are:

- Negative, consistent with WoE transformation expectations.
- Statistically significant at conventional levels.
- Stable in magnitude and variable weighting, indicating resilience to sample shifts.

Relative contribution analysis shows:

- Leverage (VAR_19): The single strongest predictor, contributing ~22–23% of explanatory power.
- Liquidity (VAR_3): Increases weight slightly on validation (~21%), underscoring its relevance in out-of-sample performance.
- Profitability (VAR_43) and Growth (VAR_113): Both stable, ~16–18%.
- Activity (VAR_78): Adds operational efficiency perspective, with ~11%.
- Leverage (VAR_27): Lowest contribution (~10%), but still significant and complementary.

The Python final model provides a parsimonious yet diversified structure, balancing traditional financial stability indicators (leverage, liquidity) with forward-looking (growth) and efficiency-oriented (activity) metrics. Its strong performance, coefficient stability, and controlled correlations make it a viable candidate for deployment.

5.6. Cross-Model Assessment

Both the SAS and Python final models demonstrate broad consistent predictive accuracy, with train-time AUC values of ~0.725 and Gini coefficients of ~0.45. On validation, the SAS model shows a modest advantage (AUC = 0.7228 vs. 0.7141), while the Python model delivers nearly identical KS statistics. Misclassification rates are nearly the same across both platforms (~5.2%). These differences are minor, falling within expected sampling variability given two independently stratified partitions, and do not materially alter risk ranking quality.

The Python model favors simplicity, achieving comparable accuracy with six variables, no residual correlations above 0.65, and slightly tighter control of collinearity. This parsimony may reduce model maintenance costs and facilitate clearer communication of risk drivers. The SAS model incorporates seven variables, producing slightly higher validation AUC but also introducing a near-threshold correlation (64.4%). Thus, SAS offers marginally stronger discrimination, while Python offers a leaner, more interpretable structure.

In both models, coefficient signs are economically coherent (all negative under WoE transformation), statistically significant across samples, and stable in relative contributions. Together with earlier PSI results, this confirms robustness and low risk of overfitting. Either model can be confidently considered for deployment.

At the achieved levels of AUC and KS, both models are well-suited for rank-ordering borrowers, supporting cut-off thresholds, exposure limits, and pricing overlays. If a single champion must be selected, the SAS model offers marginally better discriminatory power, while the Python model offers greater parsimony and slightly cleaner statistical structure. The choice can therefore be guided by organizational priorities:

- Discrimination-focused governance → SAS may be preferred.
- Simplicity and operational efficiency → Python may be favored.

Both final models meet the standards of regulatory-acceptable, stable, and interpretable credit scoring models. The SAS model excels in marginal separation, while the Python model excels in simplicity and robustness. Their proximity in performance suggests that either could serve as a production-ready solution, with the final choice reflecting the bank’s governance framework and strategic preferences.

Conclusion

The purpose of this thesis was to develop and evaluate credit scoring models based on financial statement data, using two statistical environments, SAS and Python, in order to assess their comparative performance and examine the predictive power of financial ratios for Small Business companies in Serbia. The analysis was motivated by the practical importance of reliable credit risk assessment in the banking sector and by the growing academic and professional interest in the use of open-source tools (Python) as alternatives to commercial statistical software (SAS).

The thesis addressed three central research questions. First, it examined the extent to which financial ratios can serve as predictors of default risk in Small Businesses. The results confirmed that carefully selected financial ratios provide significant discriminatory power, capturing essential aspects of liquidity, leverage, profitability, and operational efficiency. These findings reinforce the role of financial statement data as a reliable basis for risk assessment in the Small Business sector, where other types of borrower information are often limited.

Second, the thesis compared SAS and Python in their handling of the main modeling steps: Weight of Evidence (WoE) transformation, variable grouping, and brute-force logistic regression. While SAS offered a standardized and streamlined process with integrated functionalities, Python required the development of custom algorithms for each step. Despite these differences, the results demonstrated that Python-based models can achieve performance levels comparable to those obtained in SAS. This outcome suggests that open-source solutions can serve as cost-effective and transparent alternatives to commercial platforms, provided that the necessary programming expertise is available.

Third, the thesis explored whether Python could meet the standards required for regulated banking environments. Although SAS remains the industry standard due to its reliability, documentation, and regulatory acceptance, the findings indicate that Python, when carefully implemented, can deliver models that are both statistically sound and practically applicable. This supports the feasibility of incorporating open-source approaches into risk management frameworks.

In conclusion, the thesis demonstrates that financial ratios provide strong predictive power in assessing Small Business credit risk and that Python, despite requiring greater

customization and advanced programming knowledge, can yield results on par with SAS. These findings highlight both the continuing relevance of financial statement data in risk modeling and the growing potential of open-source platforms in advancing financial analytics within regulated environments.

However, the study also faced several constraints. The modeling process relied on a single data source - financial statement data, which may limit the generalizability of results to broader market conditions. Moreover, the absence of an out-of-time validation sample restricted the assessment of long-term model stability and performance across different economic cycles.

Future research could expand upon these findings by integrating additional data sources, such as macroeconomic indicators, behavioral data, and credit bureau data, to capture a wider range of risk drivers. Furthermore, the application of advanced machine learning techniques could provide deeper insights into non-linear relationships and improve predictive performance. Combining financial ratios with alternative data sources and more sophisticated modeling methods would represent a valuable next step in credit risk assessment for Small Business companies.

References

- Durand, D. (1941). Risk Elements in Consumer Instalment Financing. National Bureau of Economic Research.
- Burnham, K. P., & Anderson, D. R. (2002). *Model selection and multimodel inference: A practical information-theoretic approach* (2nd ed.). Springer.
- Anderson, R. (2007). *The credit scoring toolkit: Theory and practice for retail credit risk management and decision automation*. Oxford University Press.
- Thomas, L. C. (2009). *Consumer credit models: Pricing, profit, and portfolios*. Oxford University Press.
- Menard, S. (2010). *Logistic regression: From introductory to advanced concepts and applications*. Sage Publications.
- Greene, W. H. (2012). *Econometric analysis* (7th ed.). Pearson Education.
- Penman, S. H. (2012). *Financial statement analysis and security valuation* (5th ed.). McGraw-Hill.
- Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied logistic regression* (3rd ed.). Wiley.
- Larose, D. T., & Larose, C. D. (2015). *Data Mining and Predictive Analytics* (2nd ed.). Wiley.
- Siddiqi, N. (2017). *Intelligent Credit Scoring: Building and Implementing Better Credit Risk Scorecards* (2nd ed.). Wiley.
- Thomas, L. C., Crook, J. N., & Edelman, D. B. (2017). *Credit scoring and its applications* (2nd ed.). SIAM Publications.
- Brigham, E. F., & Ehrhardt, M. C. (2019). *Financial management: Theory & practice* (16th ed.). Cengage.
- SAS Institute Inc. (2025). *Credit scoring for SAS® Enterprise Miner™*. https://support.sas.com

- Scikit-learn developers (2024). *Scikit-learn: Machine Learning in Python*. https://scikit-learn.org
- Hand, D. J., & Henley, W. E. (1997). Statistical classification methods in consumer credit scoring: A review. *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, *160*(3), 523–541. https://academic.oup.com/jrsssa/article/160/3/523/7102381
- Lessmann, S., Baesens, B., Seow, H.-V., & Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research*, 247(1), 124–136. https://cer.business-school.ed.ac.uk/wp-content/uploads/sites/55/2017/02/Benchmarking-State-of-the-Art-Classification-Algorithms-for-Credit-Scoring-Lessmann-Seow-Baesens-and-Thomas.pdf
- Ciampi, F., & Gordini, N. (2015). Small enterprise default prediction modeling through artificial neural networks: An empirical analysis of Italian SMEs. *Journal of Small Business Management*. https://www.researchgate.net/publication/256041626_Small_Enterprise_Default_Prediction_Modeling_Through_Artificial_Neural_Networks_An_Empirical_Analysis_of_Italian_Small_Enterprises
- Altman, E. I., Iwanicz-Drozdowska, M., Laitinen, E. K., & Suvas, A. (2017). Financial distress prediction in an international context: A review and empirical analysis of Altman's Z-score model. *Journal of International Financial Management & Accounting.* *https://onlinelibrary.wiley.com/doi/abs/10.1111/jifm.12053*
- Basel Committee on Banking Supervision. (2006). International Convergence of Capital Measurement and Capital Standards: A Revised Framework (Comprehensive Version). Bank for International Settlements.
- Basel Committee on Banking Supervision. (2011). Basel III: A Global Regulatory Framework for More Resilient Banks and Banking Systems. Bank for International Settlements.

- National Bank of Serbia (2024) *Decision on the Classification of Bank Balance Sheet Assets and Off-Balance Sheet Items* (Decision Nos. 10/2024, 52/2024 & 21/2025). RS Official Gazette.