4IZ451 - Knowledge Discovery in Databases

Time deposit propensity data analysis based on the CRISP-DM methodology

Prague University of Economics and Business

Bc. Martin Fanta, Bc. Sofiia Shtol, Bc. Artem Vitkov

LS 2024/2025

Table of contents

Introduction

The aim of this project is to apply data mining methods and tools to solve a binary classification problem using customer banking data. Specifically, we focus on predicting whether a customer is likely to open a new term deposit account. This type of analysis is highly relevant for financial institutions aiming to improve targeted marketing and increase product adoption.

The task is based on a real-world dataset containing information on 120,000 bank clients with low funds position. The CRISP-DM (Cross-Industry Standard Process for Data Mining) methodology is applied to guide the project workflow. This process supports a structured approach to defining objectives, preparing data, selecting modeling techniques, and evaluating results, and it consists of six main phases: business understanding, data understanding, data preparation, modeling, evaluation, and deployment.

The goal is to develop a predictive model that enables more effective targeting of customers for time deposit promotions based on their profile and behavior patterns.

Business Understanding

Problem Description

In the highly competitive banking industry, attracting deposits is crucial for maintaining liquidity, funding loans, and driving profitability. Time deposits (also known as term deposits) are particularly valuable to banks because they represent stable and predictable sources of funding.

However, marketing these products effectively requires precise targeting to identify customers most likely to open a new term deposit. Without targeted campaigns, banks face:

- **Increased marketing costs:** Broad, untargeted marketing approaches lead to higher costs and lower efficiency.
- **Lower conversion rates:** Poor targeting results in lower customer response rates and wasted resources.

Therefore, the bank seeks to build an accurate predictive classification model that identifies customers with a high propensity to open a new time deposit account.

Business Objectives

The primary business goal of this analysis is to:

- **Increase campaign effectiveness:** Optimize marketing resources by targeting only those customers with the highest likelihood of opening a new time deposit.
- **Boost deposit inflow:** Improve overall bank liquidity by attracting a higher volume of stable, long-term deposits.

Data Mining Goals

To achieve these business objectives, the data mining tasks will involve:

- Building a robust classification model to predict the binary target attribute (Time_Deposits_Flag), indicating whether a customer will purchase a new time deposit.

- Identifying and evaluating key factors (demographic, transactional, and product-related attributes) that most strongly influence the customer's decision to purchase a new time deposit.

Success Criteria

Success will be measured by the model's performance in terms of:

- **Precision:** Maximizing the percentage of correctly targeted customers among those predicted positive, thus improving resource allocation.
- **Recall:** Ensuring a high proportion of potential buyers are identified, increasing campaign coverage and effectiveness.
- **Balanced accuracy and AUC:** Ensuring model robustness and stability, addressing class imbalance, and achieving generalizable results.

Ultimately, a successful predictive model will enable the bank to conduct highly targeted marketing campaigns, enhancing customer acquisition rates and reducing overall marketing costs.

Data Understanding

This section provides a detailed description of the dataset, including all the available attributes, their data types, and explanations relevant to the classification task at hand.

Our data represent a sample of customers from a banking institution, specifically focusing on predicting the propensity of customers to open new time deposit accounts (term deposits).

The dataset includes demographic variables, product ownership indicators, transactional behaviors, and customer interaction metrics that may influence the propensity to purchase new time deposits

#	Attribute Name	Type	Description
1	Customer_ID	Identifier	Unique customer identification number
2	Gender	Nominal	Customer gender (Male/Female)
3	Birth_Date	Date	Customer's birth date
4	Ref_Date	Date	Reference date for age calculation
5	Marital_Status	Nominal	Marital status (Married, Single, Divorced, etc.)
6	Children_Num	Numeric	Number of children
7	Occupation_Category	Nominal	Occupation category (Employee, Retiree, Business Owner)
8	Total_Income	Numeric	Annual income of the customer
9	Payroll_Flag	Binary	Payroll relationship with the bank
10	Business_Flag	Binary	Business relationship with the bank
11	Saving_Current_Accounts_Flag	Binary	Ownership of saving or current accounts

12	Investment_Products_Flag	Binary	Ownership of investment products
13	Insurance_Products_Flag	Binary	Ownership of insurance products
14	Business_Loans_Flag	Binary	Ownership of business loans
15	Housing_Loans_Flag	Binary	Ownership of housing loans
16	Consumer_Loans_Flag	Binary	Ownership of consumer loans
17	Credit_Cards_Flag	Binary	Ownership of credit cards
18	Saving_Current_Balance	Numeric	Mean balance of saving/current accounts (last 6 months)
19	Investment_Products_Balance	Numeric	Mean balance of investment products (last 6 months)
20	Insurance_Balances	Numeric	Mean balance of insurance products (last 6 months)
21	Business_Loans_Balance	Numeric	Mean balance of business loans (last 6 months)
22	Housing_Loans_Balance	Numeric	Mean balance of housing loans (last 6 months)
23	Consumer_Loans_Balance	Numeric	Mean balance of consumer loans (last 6 months)
24	Credit_Cards_Balance	Numeric	Mean balance of credit cards (last 6 months)
25	Branch_Trans_Num	Numeric	Avg. monthly branch transactions (last 6 months)
26	ATM_Trans_Num	Numeric	Avg. monthly ATM transactions (last 6 months)
27	APS_Trans_Num	Numeric	Avg. monthly APS transactions (last 6 months)
28	Phone_Trans_Num	Numeric	Avg. monthly phone transactions (last 6 months)
29	Internet_Trans_Num	Numeric	Avg. monthly internet transactions (last 6 months)
30	Deposit_Trans_Num	Numeric	Avg. monthly deposit transactions (last 6 months)
31	Withdrawl_Trans_Num	Numeric	Avg. monthly withdrawal transactions (last 6 months)
32	Payment_Trans_Num	Numeric	Avg. monthly payment transactions (last 6 months)
33	Transfer_Trans_Num	Numeric	Avg. monthly transfer transactions (last 6 months)
34	Deposit_Trans_Amount	Numeric	Avg. monthly deposit transaction amount (last 6 months)
35	Withdrawl_Trans_Amount	Numeric	Avg. monthly withdrawal transaction amount (last 6 months)
36	Payment_Trans_Amount	Numeric	Avg. monthly payment transaction amount (last 6 months)

37	Transfer_Trans_Amount	Numeric	Avg. monthly transfer transaction amount (last 6 months)
38	Credit_Cards_Installments	Numeric	Total credit card installments (last 6 months)
39	Credit_Cards_Payments_Num	Numeric	Avg. monthly credit card payments number (last 6 months)
40	Credit_Cards_Purchases_Num	Numeric	Avg. monthly credit card purchases number (last 6 months)
41	Credit_Cards_Withdrawals_Num	Numeric	Avg. monthly credit card withdrawals number (last 6 months)
42	Credit_Cards_Payments_Amount	Numeric	Avg. monthly credit card payments amount (last 6 months)
43	Credit_Cards_Purchases_Amount	Numeric	Avg. monthly credit card purchases amount (last 6 months)
44	Credit_Cards_Withdrawals_Amount	Numeric	Avg. monthly credit card withdrawals amount (last 6 months)
45	Arrears_Months_Max	Numeric	Max. months in delinquency (last year)
46	**Time_Deposits_Flag (Target)**	Binary	New Time Deposit Purchase (True/False)

Propensity Modeling Context

The objective of this analysis is to classify customers based on their propensity to purchase a new time deposit. The provided attributes offer detailed insights into customers' financial behavior, demographic profiles, product preferences, and interaction channels, all of which are potential predictors of their likelihood to engage in new deposit activities.

In propensity modeling, understanding the relationship between customer characteristics and their future actions allows banks to execute highly targeted marketing campaigns, significantly improving their acquisition efficiency and reducing resource wastage.

Data Preprocessing in RapidMiner

Step 1: Data Import (Import Data Operator)

We have loaded the file TimeDeposit_10K.csv into RapidMiner and defined correct attribute types, which is essential for subsequent preprocessing tasks.

For example, all the flags were encoded as binary ‘dummy’ variables, all amounts, balances and numbers were coded as ‘real’, with the respective integer attributes, like ‘Children_num’ and ‘Arrears_Months_Max’ given the ‘integer’ data type. The ‘Birth_date’ and ‘Ref_date were encoded as ‘datetime’, which will be used for the subsequent preprocessing steps.

Step 2: Generate Attributes (Feature Engineering)

The `Age` attribute was calculated using RapidMiner by first computing the time difference between the reference date (`Ref_Date`) and the customer's birth date (`Birth_Date`) in milliseconds,

converting it into years (considering average year length including leap years), and then rounding it to the nearest integer. This attribute provides a concise numeric representation of the customer's age, essential for demographic profiling, and predictive modeling.

`Total_Products`, `Total_Balance`, `Total_Transactions` summarize related financial behaviors of customers. For instance, `Total_Balance` represents the cumulative balance from various financial accounts, thus capture the overall financial health or stability of the customer.

`Total_Transactions` indicates the aggregated number of customer transactions across different channels, which helps to identify transaction activity levels and engagement patterns.

Attribute Name	Expression for RapidMiner
Age	`round (-1*date_diff(Ref_Date, Birth_Date))/ (1000 * 60 * 60 * 24 * 365.25))`
Total_Balance	`Saving_Current_Balance + Investment_Products_Balance + Insurance_Balances + Business_Loans_Balance + Housing_Loans_Balance + Consumer_Loans_Balance + Credit_Cards_Balance`
Total_Transactions	`Branch_Trans_Num + ATM_Trans_Num + APS_Trans_Num + Phone_Trans_Num + Internet_Trans_Num + Deposit_Trans_Num + Withdrawl_Trans_Num + Payment_Trans_Num + Transfer_Trans_Num + Credit_Cards_Payments_Num + Credit_Cards_Purchases_Num + Credit_Cards_Withdrawals_Num`

Step 3: Replace Infinite Values

In this preprocessing step, the **Replace Infinite Values** operator in RapidMiner was applied to handle infinite numeric values within the dataset. Infinite values can emerge as a result of calculations involving divisions (e.g., ratios where denominators may be zero), causing potential problems in modeling steps. We replace positive infinite values (+∞) with NAs to ensure dataset integrity and consistency and computational stability of the next methods.

Step 4: Generate Additional Attributes (Percentage-based Features)

In this step, additional percentage-based attributes were created using RapidMiner's **Generate Attributes** operator. These attributes represent ratios of individual balances relative to the customer's total financial balance. Such ratios effectively capture the relative importance of each financial product within the customer's overall portfolio.

Ratio attributes (`Deposits_to_Loans_Ratio`, `Deposit_vs_Withdrawal_Ratio`) reveal critical financial behavior patterns by comparing key transaction types.

Attribute Name	Expression for RapidMiner

Deposits_to_Loans_Ratio	Saving_Current_Balance / (Housing_Loans_Balance + Consumer_Loans_Balance + Business_Loans_Balance)
Deposit_vs_Withdrawal_Ratio	Deposit_Trans_Amount / Withdrawl_Trans_Amount

Deposits_to_Loans_Ratio represents the balance between savings and outstanding loans, providing insight into customer liquidity and financial stability. Deposit_vs_Withdrawal_Ratio reflects the ratio between deposit and withdrawal transaction amounts, highlighting saving versus spending behaviors.

Percentage attributes indicate the relative importance of each product category within a customer's financial portfolio, providing deeper insights into customer financial strategies.

Attribute Name	Expression for RapidMiner
Saving_Balance_Perc	Saving_Current_Balance / Total_Balance
Investment_Balance_Perc	Investment_Products_Balance / Total_Balance
Insurance_Balance_Perc	Insurance_Balances / Total_Balance
Business_Loan_Balance_Perc	Business_Loans_Balance / Total_Balance
Housing_Loan_Balance_Perc	Housing_Loans_Balance / Total_Balance
Consumer_Loan_Balance_Perc	Consumer_Loans_Balance / Total_Balance
Credit_Card_Balance_Perc	Credit_Cards_Balance / Total_Balance

Saving_Balance_Perc shows the proportion of total assets held in savings, highlighting customers' saving habits.

Investment_Balance_Perc represents the share of financial assets invested, indicating investment preferences.

Insurance_Balance_Perc represents the percentage of total balances allocated to insurance products, providing insights into the customer's inclination towards financial protection and long-term security.

Business_Loan_Balance_Perc represents the percentage of total balances allocated to insurance products, providing insights into the customer's inclination towards financial protection and long-term security.

Housing_Loan_Balance_Perc shows the fraction of the customer's total debt associated with housing loans, reflecting long-term financial responsibilities related to home ownership.

`Consumer_Loan_Balance_Perc` provides insight into the percentage of a customer's total financial obligations from consumer loans, illustrating personal borrowing and consumption behavior.

`Credit_Card_Balance_Perc` demonstrates the proportion of total balances stemming from credit card debt, offering valuable insights into consumer spending patterns and debt management tendencies.

Step 5: Replace Infinite Values

See Step 3. We reiterate this step to make sure that there is no division by zero in the case of the percentage attributes with `Total_Balance` in the denominator. We posit, that in those cases the factual percentage in the attribute is zero, as `Total_Balance` accounts for the sum of all relevant account balances.

Step 6: Select Attributes (Removal of Redundant and Irrelevant Variables)

In this preprocessing step, the **Select Attributes** operator was used to remove redundant and irrelevant attributes from the dataset, based on aggregation and ratio generation performed in previous steps.

Removing irrelevant attributes prevents bias in the predictive model and reduces complexity, speeding up modeling processes. This careful selection of attributes improves model interpretability, reducing the risk of multicollinearity, and enhancing the overall efficiency and accuracy of the subsequent predictive modelling process.

Attributes Selected for Removal:

- **Original date and identifier attributes**:
  - Customer_ID
  - Birth_Date
  - Ref_Date

- **Balances (replaced by aggregated and ratio attributes)**:
  - Business_Loans_Balance
  - Consumer_Loans_Balance
  - Credit_Cards_Balance
  - Housing_Loans_Balance
  - Insurance_Balances
  - Investment_Products_Balance
  - Saving_Current_Balance

- **Transaction amounts (used to create ratio attributes)**:
    - Deposit_Trans_Amount
    - Payment_Trans_Amount
    - Transfer_Trans_Amount
    - Withdrawl_Trans_Amount
- **Transaction counts (aggregated into Total_Transactions)**:
    - APS_Trans_Num
    - ATM_Trans_Num
    - Branch_Trans_Num
    - Deposit_Trans_Num
    - Internet_Trans_Num
    - Payment_Trans_Num
    - Phone_Trans_Num
    - Transfer_Trans_Num
    - Withdrawl_Trans_Num
- **Credit card detailed attributes (redundant due to aggregated balances and transactions)**:
    - Credit_Cards_Installments
    - Credit_Cards_Payments_Amount
    - Credit_Cards_Payments_Num
    - Credit_Cards_Purchases_Amount
    - Credit_Cards_Purchases_Num
    - Credit_Cards_Withdrawals_Amount
    - Credit_Cards_Withdrawals_Num

Rationale for Attribute Removal:

- **Aggregated Redundancy**:
    - Individual balances (`Saving_Current_Balance`, `Housing_Loans_Balance`, etc.) were removed because they are effectively captured in aggregate attributes (`Total_Balance`) and percentage-based attributes (`Saving_Balance_Perc`, etc.).

- **Ratio-Based Redundancy**:
    - Transaction amounts and counts were used in ratio attributes (`Deposit_vs_Withdrawal_Ratio`) or aggregated into total attributes (`Total_Transactions`), rendering their original values redundant.
- **Irrelevant Attributes**:
    - `Ref_Date` contains only a single date (Dec 31, 2008), providing no predictive information.
    - `Birth_Date` was removed because the relevant information is already encoded in the derived Age attribute.
    - `Customer_ID` is an identifier with no predictive value.

Steps 6 and 7:

The row-wise deletion of the missing values would have at this stage of preprocessing left us with a mere 886 observations. This approach can enforce the loss of valuable and potentially introduce bias if the missing values are not missing completely at random.

So, we have instead decided to imputing the NAs with column means in the case of quantitative variables and the modes in the case of categorical attributes. This approach can potentially be harmful, as it can concentrate the data around the mean, thus lowering the dispersion, but artificially increasing the bias towards the mean value.

Using the imputation models, such as kNN or linear regression was, in our case, redundant, since we can assume the data are MNAR (Missing Not at Random): wealthier clients tend not to share their income, type of occupation, investment and other sensitive personal information. We also have a big inherent bias considering the regression to the mean, considering the imbalanced nature of our target variable, which we will address later.

As for replacing infinite values, the same logic follows as in the previous steps.

Step 8: Normalization (Z-Transformation)

In this preprocessing step, the **Normalize** operator with the **Z-transformation method** was applied to the dataset in RapidMiner. This procedure adjusts numeric attribute values by transforming them to have a mean of zero and a standard deviation of one.

Rationale and Benefits for Modelling:

- **Consistent Scale**:
    - Z-transformation ensures all numerical attributes are on the same scale. This prevents attributes with larger numerical ranges from disproportionately influencing the predictive model.
- **Improved Algorithm Performance**:
    - Many predictive algorithms (e.g., logistic regression, XGBoost) rely on or significantly benefit from normalized data, as uneven attribute scales can negatively impact model accuracy and convergence speed.

- **Reduces Risk of Model Bias**:
    - Normalization reduces the potential bias toward attributes with inherently larger values or variance, enabling the model to treat each feature equally during learning.
- **Better Interpretability of Results**:
    - Standardized attributes facilitate easier interpretation of model coefficients, making it clearer to understand each predictor's relative importance.

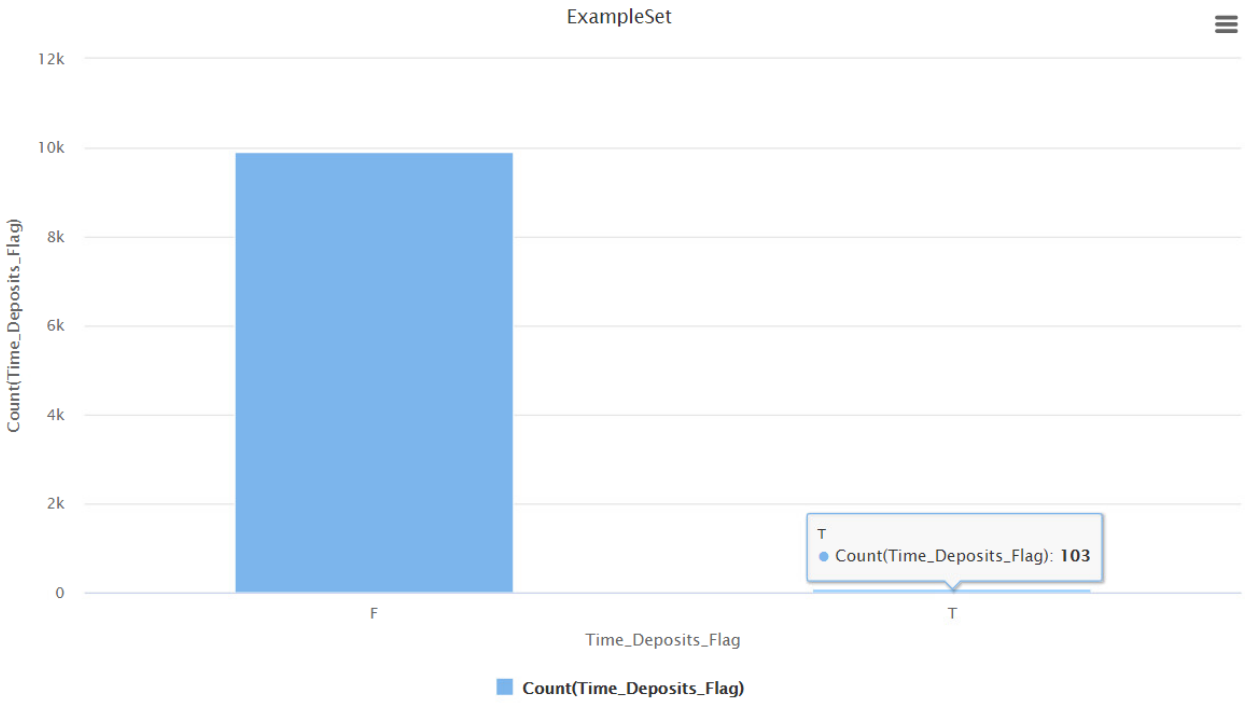
Step 9: Setting Target Variable Role (Set Role)

In this preprocessing step, the **Set Role** operator was utilized in RapidMiner to explicitly define the attribute **`Time_Deposits_Flag`** as the target variable (label) for the predictive modeling task. We will need this for our classification task and the filtering process.

Step 10: Addressing Class Imbalance via Stratified Sampling (1:6 Ratio)

In this preprocessing step, to tackle the significant class imbalance in the dataset—**103 instances labeled TRUE** (customers who opened a Time Deposit) versus **9,897 instances labeled FALSE**—a stratified sampling approach was employed. This method involved retaining all 103 TRUE instances and selecting 618 FALSE instances, achieving a **1:6 ratio** between the minority and majority classes.

The original dataset exhibited an imbalance ratio of approximately 1:96, which can severely hinder a model's ability to learn patterns associated with the minority class. While a 1:1 ratio might seem ideal, it often leads to the loss of valuable information from the majority class. We thus assume, that the **population distribution on the given parameter is 1:6**. By adjusting this ratio to 1:6, the model is better positioned to recognize and learn from the minority class instances, enhancing its predictive performance.



The SMOTE resampling technique was, unfortunately, not available in this version of RapidMiner

So, we have reserved to using stratified sampling to undersample the observations of the majority class. This is, of course, not a very big problem, since this study (7) demonstrated that techniques like oversampling, undersampling, and threshold adjustment could generate similar results by producing classifiers that fall along the same ROC curve.

(Google, 2025)**Degree of imbalance**	**Proportion of Minority Class**
Mild	20-40% of the data set
Moderate	1-20% of the data set
Extreme	<1% of the data set

Further research indicates that moderately imbalanced ratios, such as 1:6, often yield better model performance compared to extreme imbalances or perfectly balanced datasets. This approach helps in maintaining the natural distribution of data while still addressing the imbalance issue. (6, 8)

Step 11: Feature Selection Using Information Gain

To enhance the predictive accuracy and interpretability of our classification model, the **Weight by Information Gain** operator was utilized to calculate the importance of each predictor attribute based on their information gain concerning the target variable (`Time_Deposits_Flag`). Subsequently, the **Select by Weights** operator was applied to retain only those attributes exceeding the specified threshold (≥ 0.2), ensuring selection of highly informative predictors.

The threshold of **0.2** effectively filters out less informative predictors, retaining only those attributes strongly associated with the target class, thereby increasing predictive reliability and model interpretability.

The ranked Information Gain between the two variables can be roughly interpreted as the best 'splitter' of the dataset between the two target classes in the Decision Tree model. It minimizes the entropy inside the separated groups.

The provided table lists attributes sorted by their **Information Gain**, a measure used to determine how effectively each attribute helps in distinguishing between classes in a classification task—in this case, predicting customers' propensity to open new time deposits (`Time_Deposits_Flag`).

Attribute Name	**Information Gain**
Marital_Status	1.0000
Deposit_Trans_Amount	0.9556
Total_Balance	0.8332
Payment_Trans_Amount	0.7852
Total_Income	0.7680
Business_Flag	0.6768
Occupation_Category	0.6477

Attribute Name	Information Gain
Business_Loan_Balance_Perc	0.6005
Deposit_vs_Withdrawal_Ratio	0.5995
Consumer_Loans_Flag	0.5900
Consumer_Loan_Balance_Perc	0.5276
Total_Transactions	0.5128
Credit_Card_Balance_Perc	0.4984
Saving_Balance_Perc	0.4750
Credit_Cards_Payments_Amount	0.4458
Transfer_Trans_Amount	0.4310
Credit_Cards_Installments	0.3970
Housing_Loan_Balance_Perc	0.3883
Deposits_to_Loans_Ratio	0.3791
Withdrawl_Trans_Amount	0.3274
Insurance_Products_Flag	0.3253
Insurance_Balance_Perc	0.3253
Age	0.3006
Children_Num	0.2648
Business_Loans_Flag	0.2388

We can see that the **`Marital_Status`** attribute has perfect information gain, indicating marital status is an extremely strong predictor of customers' likelihood to open new time deposits. Distinct segments (single, married, etc.) clearly differ in their purchasing behavior.

`Deposit_Trans_Amount` has an Information Gain value of 0.9556, suggesting that the amount of deposit transactions strongly correlates with opening new deposits. Customers depositing larger sums are likely candidates for new time deposit products.

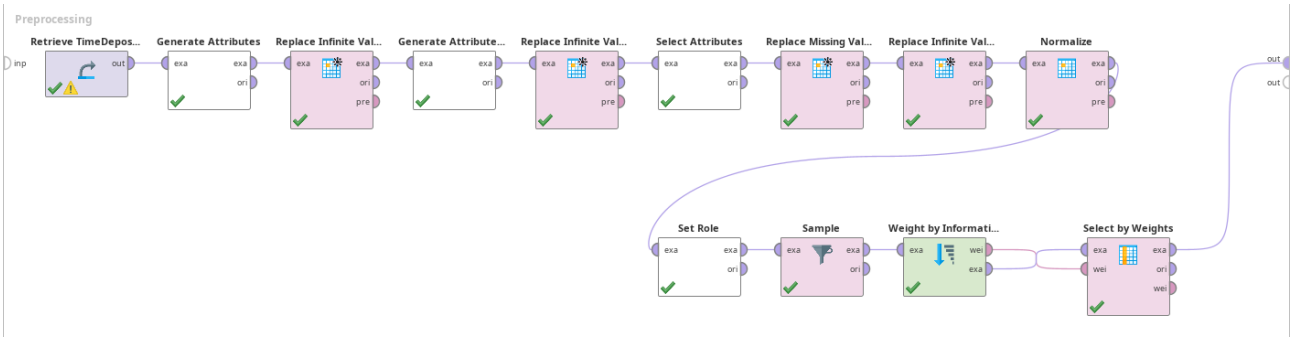
`Total_Balance`, being ranked the third by Information Gain (0.8332) is showing that customers with higher total balances likely have available funds and a higher propensity to secure their money via time deposits.

Overall, this feature ranking clearly highlights the importance of demographic attributes (marital status, income), transaction patterns (deposit, payment amounts), and aggregate financial behaviors

(total balance, transactions) as primary factors influencing customers' likelihood to open a new Time Deposit.

The resulting scheme

Here is the final data pipeline scheme for the preprocessing phase:



Common Sense

Information gain values from the previous chapter highlight which variables might deserve more attention. For example, marital status ranks highest, which could make sense if married individuals tend to plan ahead financially or prefer more stable savings options like time deposits. Likewise, customers with higher deposit amounts or larger balances might simply have more money available, making them more likely to consider locking some of it in a term deposit.

On the other hand, some lower-ranked variables might also tell us something. Having several children could mean higher day-to-day expenses, leaving less room for savings products. A business loan might indicate financial responsibility, but it could just as easily signal existing debt or tighter cash flow. These are things a model won't know unless we think them through.

Modeling

For our data analysis, we decided to choose the following algorithms: Naive Bayes, Logistic Regression, Random Forest, and Gradient Boosted Tree.

Validation approach

It should also be noted that instead of using a standard train-test split for model evaluation, we implemented **10-fold cross-validation** for each model. This strategy was selected because it provides a more robust and reliable assessment of model performance, minimizing variance in evaluation metrics. Cross-validation ensures that every instance in the dataset has the opportunity to be part of both the training and test sets, thereby giving a comprehensive evaluation of how each model generalizes to unseen data.

Chosen models

Naive Bayes

The Naive Bayes algorithm is a probabilistic classifier that simplifies learning by assuming that all features are conditionally independent given the class label. This assumption significantly reduces computational complexity and allows the model to scale well to high-dimensional data (Rish, 2001). This algorithm performs well even when its assumption of feature independence is violated. In the chosen dataset, many features are likely only weakly related (for example, credit card ownership and internet transactions), so Naive Bayes can still give competitive results.

In our modeling, the Naive Bayes algorithm was configured with Laplace correction enabled. This decision was made to handle zero-frequency problems, improving prediction stability when dealing with categorical variables or small subsets of data, which often occur in imbalanced datasets.

Logistic regression

Logistic regression is a commonly used algorithm for binary classification problems. It estimates the probability that a given input belongs to a particular category by applying the logistic function to a linear combination of the input features. This results in outputs between 0 and 1, which can be interpreted as probabilities (Berka, 2003). Logistic regression is well-suited for analyzing the effect of multiple explanatory variables simultaneously, helping to control confounding effects in complex datasets like the chosen one.

For logistic regression, we used a regularized version of the algorithm with automatic lambda search enabled. Regularization helps prevent overfitting by penalizing excessively complex models. Automatic lambda search ensures optimal regularization strength, enhancing model generalization.

Random Forest

Random Forest is an ensemble learning method that builds multiple decision trees and combines their predictions to improve classification or regression performance. Each tree is trained on a bootstrap sample of the data, and at each node split, a random subset of features is selected. This strategy reduces correlation between trees and enhances generalization. The final model aggregates the outputs of all individual trees (Breiman, 2001). Because of its robustness to noise and irrelevant variables, it could be particularly useful in this dataset where many features may have weak or indirect relationships to the target. Because Random Forest builds multiple decision trees on random subsets of features and data, it can capture non-linear patterns and complex interactions that simpler models might miss.

We configured the Random Forest model with 100 trees, a maximal tree depth of 10, and the criterion set to gain ratio. The choice of 100 trees ensures a robust ensemble that captures diverse patterns within the data, while limiting maximal depth to 10 avoids overly complex models prone to overfitting. Using gain ratio as a splitting criterion effectively handles attributes of varying scales and types, making it particularly suitable for our heterogeneous dataset.

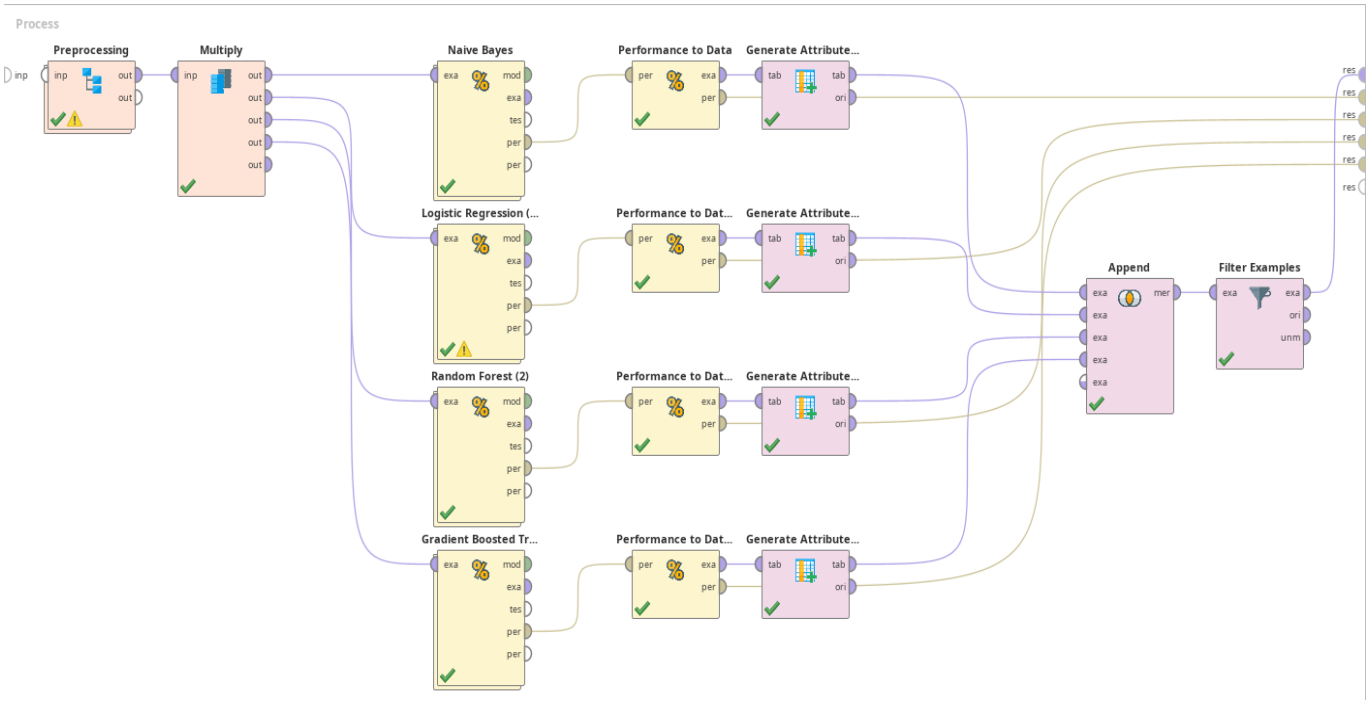
Gradient Boosted Tree

Gradient Boosted Trees are a class of ensemble machine learning methods that build predictive models through sequentially adding decision trees. Each new tree is trained to correct the residual errors made by the combined ensemble of previous trees, using the gradient of a specified loss function. This boosting strategy enables this algorithm to achieve high accuracy and capture complex, non-linear relationships in data (Friedman, 2001). For our analysis, we used the built-in implementation of Gradient Boosted Trees provided by RapidMiner.

For the Gradient Boosted Trees, we set the model to train 50 trees with a maximal depth of 5, a learning rate of 0.01, and a minimum split improvement threshold of 1.0E-5. These parameters balance predictive power and model complexity by sequentially correcting residuals, thus effectively capturing intricate non-linear relationships without substantial overfitting. A lower learning rate ensures stable convergence and improved generalization.

Evaluation

Here is the complete data pipeline schema of our project, including the Modeling phase:

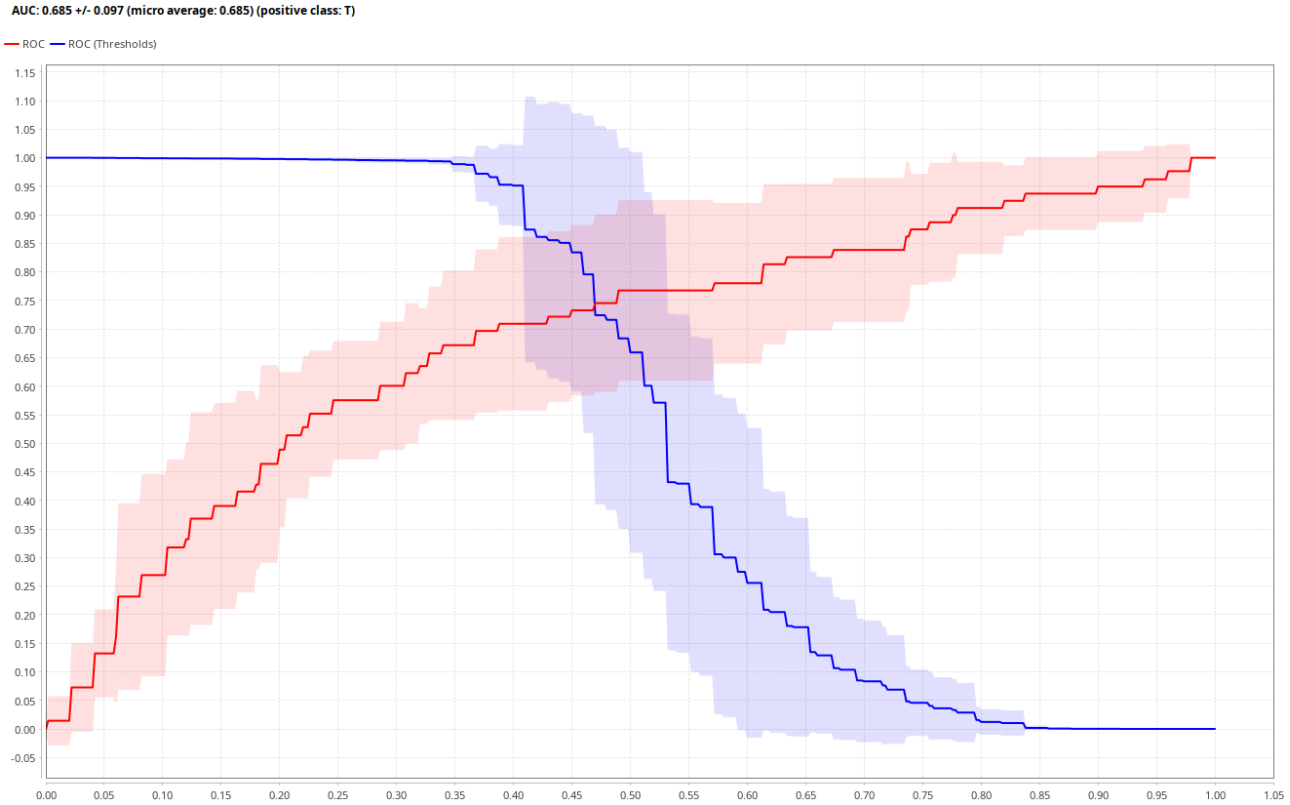


ROC (Receiver Operator Characteristic)

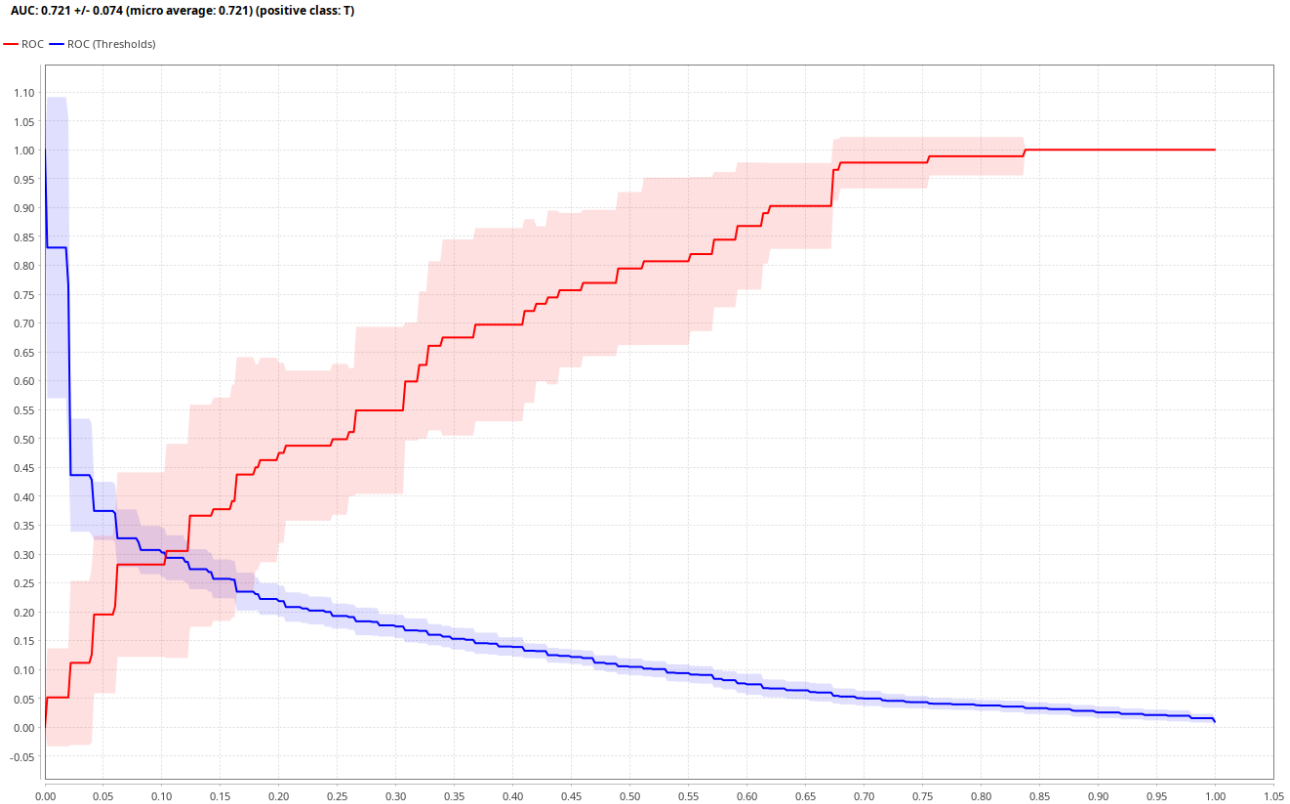
This section presents the ROC curves for the four developed models. The ROC curve is a graphical representation used to evaluate the performance of classification models by plotting the True Positive Rate (sensitivity) against the False Positive Rate (1 - specificity) across different threshold values. It illustrates the trade-off between correctly identifying positive cases and incorrectly classifying negative ones. A curve closer to the top-left corner indicates better model performance. Additionally, the Area Under the Curve (AUC) provides a single metric to compare models, with higher values reflecting stronger discriminatory power. We could roughly interpret each models' discrimination ability with this metric.

We can tell from the visual output, that none of our models exhibit overfitting issues, in many thanks to the cross-validation approach we have used. The maximum AUC value out of all the models does not automatically signal the optimality of a given model. It signals, that the model can identify the classes correctly on aggregate, but does not discern between the ability to correctly predict a given class.

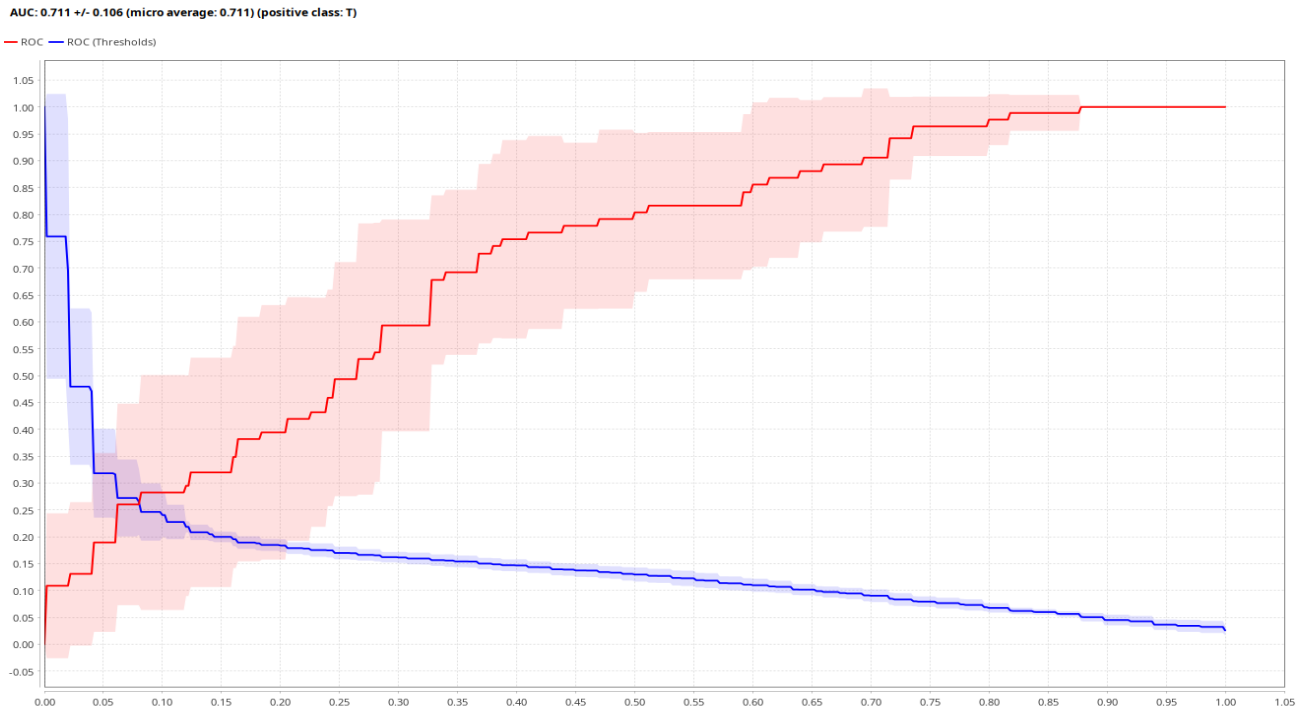
Naive Bayes



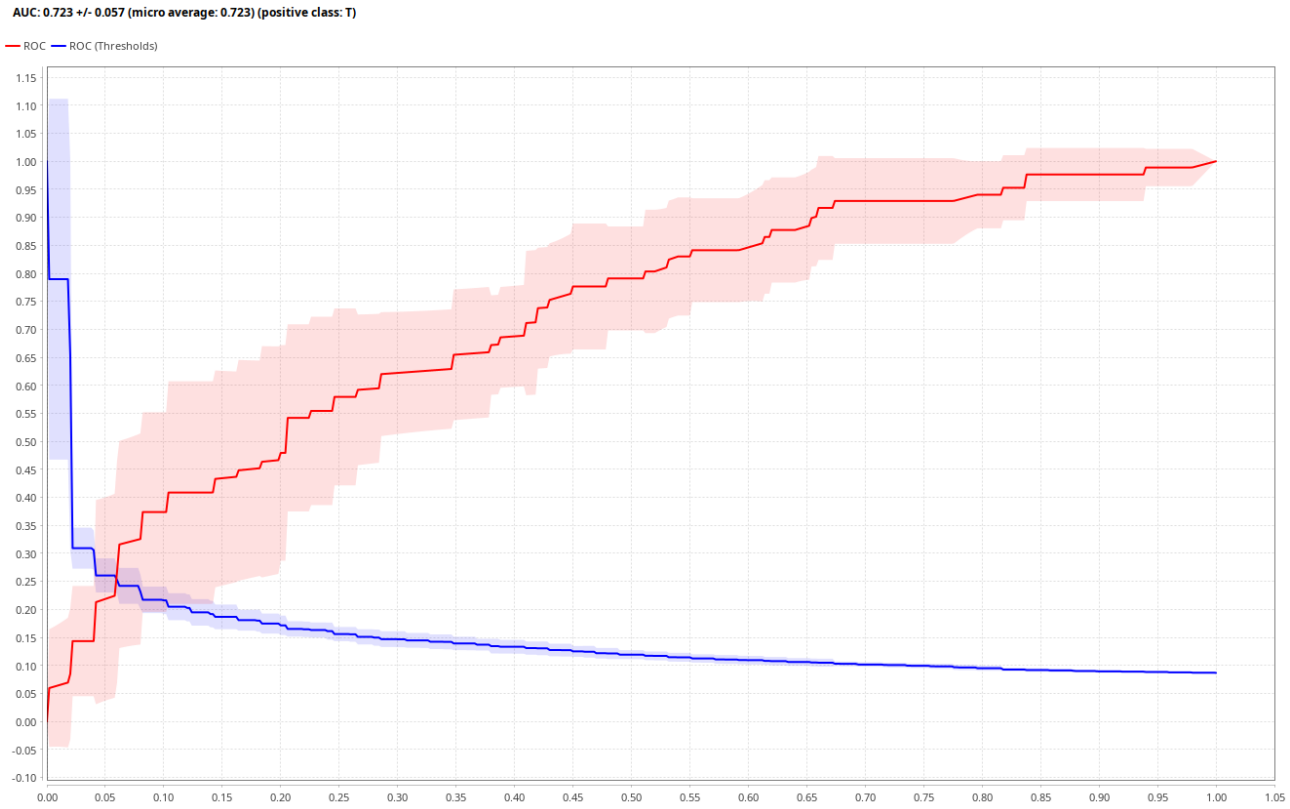
Logistic regression



Random Forest



Gradient Boosted Tree



Business interpretation

In our case, the banking institution would be more interested on accurately predicting **the positive class**, meaning the clients, that have opened of a new term deposit account, while being in a low funds

position. This gives the financial institution meaningful predictors of retaining the customer, while maintaining adequate capital reserves due to BASEL IV regulations.

That being said, the banks would realistically need to use easily interpretable probabilistic models, such as Logistic Regression or Naïve Bayes to report on the credit risk and to measure a given customers chance of opening a deposit account. We thus have to find an optimal tradeoff between model accuracy in predicting the positive target class and model interpretability.

We would need to use a meaningful metric to address this in discrepancy, since a boosting and an ensemble model do not provide nice credit risk measures, such as odds ratio or posterior conditional probabilities for a given predictor.

We have decided to use the following metrics computed on the positive class:

1. Precision: the ratio of true positives (TP) to the total number of predicted positives (TP + FP)

2. Recall: the ratio of true positives (TP) to the total number of actual positives (TP + FN)

3. F1-score: the harmonic mean of Precision and Recall

In our case we are ought to maximize the F1-score of the model, while retaining adequate overall discrimination power given by the AUC. We also prioritize recall over precision, since the missing of a potential depositor can be detrimental to the financial state of the bank, which then cannot adequately predict the amount of given out loans for the next quarter.

The banking institution would interpret the classifiers precision as the ratio of clients, who out of all the model's positive predictions turned out to be true. The number of 'false alarms' (marking a person, that withdrew liquidity, as a stable customer) is not that problematic, since most of the time the amount of money on average would be low, given the scope of the data.

High recall signalizes the model's ability not to miss out on potential depositors, which extremely important in the case of low-fund clients. Not finding positive instances, when they are there can be problematic for the banks, given the present value of investments, the opportunity cost of capital and the need for financial planning, as a depositor keeps his funds for relatively long period of time.

We can clearly see from the summary below, that recall is maximal in the Naïve Bayes classifier, which suggests the model to be optimal for customer retention. On the other hand, it marked a lot of non-depositors to be such, which is not optimal for potential targeting of these clients.

The less interpretable Random Forest had higher precision, but catastrophic recall. It makes it unusable for modeling positive cases. The confusion matrix indicated, that the model did not handle slightly imbalanced data well, just marking almost every observation as positive. That is precisely why AUC can be misleading.

Logistic regression can be seen as the baseline model for this use case, as it has a great probabilistic interpretation, while retaining a close to optimal AUC and F1-score. The standard deviation on it is lower, than in XGBoost, making the interval of the parameter even better. This is why **we have chosen logit as a preferred model** for our task.

Gradient Boosted Trees had comparable, but slightly better results, than logit, maxing out on AUC and F1-score, and having a higher recall, than logit. We, unfortunately, cannot extract meaningful predictor interpretations from it, making it too bad for interpretation purposes and still not good enough for classification. It can be used as an auxiliary validation model to Logistic Regression.

Row No.	Criterion	Value	Standard D...	Variance	Model
1	AUC	0.685	0.097	0.009	Naive Bayes
5	AUC	0.721	0.074	0.005	Logistic Regr...
9	AUC	0.711	0.106	0.011	Random Forest
13	AUC	0.723	0.057	0.003	Gradient Boo...
4	f_measure	0.321	0.053	0.003	Naive Bayes
8	f_measure	0.328	0.136	0.018	Logistic Regr...
12	f_measure	0.108	?	?	Random Forest
16	f_measure	0.349	0.155	0.024	Gradient Boo...
2	precision	0.203	0.035	0.001	Naive Bayes
6	precision	0.335	0.117	0.014	Logistic Regr...
10	precision	0.455	?	?	Random Forest
14	precision	0.306	0.119	0.014	Gradient Boo...
3	recall	0.780	0.136	0.018	Naive Bayes
7	recall	0.366	0.201	0.040	Logistic Regr...
11	recall	0.058	0.081	0.007	Random Forest
15	recall	0.420	0.221	0.049	Gradient Boo...

Deployment

This project was conducted solely for educational purposes; therefore, no actual deployment phase was carried out. However, developed models could be integrated into a real-world banking environment as part of a decision-support system. It could be used to assist marketing or CRM teams in identifying customers with a high propensity to open a time deposit account. This would allow the bank to tailor its outreach and promotional strategies more effectively, optimizing resource allocation and increasing conversion rates.

Conclusion

In this project, we applied the CRISP-DM methodology to develop a model identifying bank clients with a high propensity to open new term deposit accounts. Through extensive preprocessing, including feature engineering, normalization, and addressing class imbalance, we ensured data quality and model readiness. Then we trained several models (Naive Bayes, Logistic Regression, Random Forest, and Gradient Boosted Trees) using 10-fold cross-validation to ensure performance evaluation.

Our findings showed that while ensemble models such as Random Forest and Gradient Boosted Trees offered strong overall predictive performance, simpler probabilistic models like Logistic Regression and Naive Bayes provided better interpretability and more stable recall. Ultimately, Logistic Regression emerged as a balanced choice for deployment due to its solid tradeoff between interpretability and accuracy.

References

1. *Balancing the Scales: A comprehensive study on tackling class imbalance in binary classification*. https://arxiv.org/html/2409.19751v1#bib.bib15
2. Berka, P. (2003). *Dobývání znalostí z databází.* Praha: Academia.
3. Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32. https://link.springer.com/article/10.1023/A:1010933404324

4. Chapman P., Clinton J., Kerber R. (2012). *CRISP-DM 1.0, Step-by-step data mining guide.* http://www.kde.cs.uni-kassel.de/lehre/ws2012-13/kdd/files/CRISPWP-0800.pdf
5. Friedman, J. H. (2001). *Greedy function approximation: A gradient boosting machine. The Annals of Statistics*, *29*(5). https://doi.org/10.1214/aos/1013203451
6. Google. *Imbalanced datasets*. Google Machine Learning Crash Course. https://developers.google.com/machine-learning/crash-course/overfitting/imbalanced-datasets. Searched 01.06.2025.
7. Kumar, V., Lalotra, G. S., Sasikala, P., Rajput, D. S., Kaluri, R., Lakshmanna, K., Shorfuzzaman, M., Alsufyani, A., & Uddin, M. (2022). *Addressing Binary Classification over Class Imbalanced Clinical Datasets Using Computationally Intelligent Techniques. Healthcare*, *10*(7), 1293. https://doi.org/10.3390/healthcare10071293
8. Khan, A. A., Chaudhari, O., & Chandra, R. (2024). A review of ensemble learning and data augmentation models for class imbalanced problems: Combination, implementation and evaluation. *Expert Systems with Applications*, *244*, 122778. https://doi.org/10.1016/j.eswa.2023.122778
9. Rish, I. (2001). *An empirical study of the naive Bayes classifier*. Dostupné z: https://www.researchgate.net/publication/228845263_An_Empirical_Study_of_the_Naive_Bayes_Classifier/
10. *Home—Altair RapidMiner Documentation*. (n.d.). Retrieved June 1, 2025, from https://docs.rapidminer.com/