Classification Analysis of a Marketing Data Set

1. Introduction

The following marketing dataset from a Portuguese bank emerged from a marketing campaign carried out with the aim of acquiring customers for fixed-term deposits (variable „y“). The existing data will be described below.

Our goal is to predict the variable “y” of the data set.

Tasks:

- Develop a work product (e.g. Jupyter Notebook) that shows how to proceed in exploratory data analysis.
- Define a suitable target metric(s) to measure the performance of a forecast. Also explain why you have chosen your target metric.
- Implement and train a model to predict the target variable “y”.
- Briefly illustrate the findings and performance of the predictions in a suitable form.

Further questions:

- Would you consider additional data points for the prediction?
- If yes: what information (that a bank typically has available) would be most promising?
- Can the findings of your analysis be used for other areas of the bank?
- Does the problem definition need to be adapted for this? How would you proceed here?

2. Description of the marketing data set

This dataset is public available for research. The details are described in:

[Moro et al., 2011] S. Moro, R. Laureano and P. Cortez. *Using Data Mining for Bank Direct Marketing: An Application of the CRISP-DM Methodology.*

P. Novais et al. (Eds.), *Proceedings of the European Simulation and Modelling Conference - ESM'2011*, pp. 117-121, Guimarães, Portugal, October, 2011. EUROSIS.

The data set is also available at:

- [pdf] http://hdl.handle.net/1822/14838
- [bib] http://www3.dsi.uminho.pt/pcortez/bib/2011-esm-1.txt

Description of the data set's structure:

1. Title: Bank Marketing

2. Sources

Created by: Paulo Cortez (Univ. Minho) and Sérgio Moro (ISCTE-IUL) @ 2012

3. Relevant Information:

The data is related with direct marketing campaigns of a Portuguese banking institution.

The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, in order to access if the product (bank term deposit) would be (or not) subscribed.

There are two datasets:

1) bank-full.csv with all examples, ordered by date (from May 2008 to November 2010).

2) bank.csv with 10% of the examples (4521), randomly selected from bank-full.csv.

The smallest dataset is provided to test more computationally demanding machine learning algorithms (e.g. SVM).

The classification goal is to predict if the client will subscribe a term deposit (variable y).

4. Number of Instances: 45211 for bank-full.csv (4521 for bank.csv)

5. Number of Attributes: 16 + output attribute.

6. Attribute information:

Input variables:

bank client data:

1 - age (numeric)

2 - job : type of job (categorical: "admin.","unknown","unemployed","management","housemaid","entrepreneur","student", "blue-collar","self-employed","retired","technician","services")

3 - marital : marital status (categorical: "married","divorced","single"; note: "divorced" means divorced or widowed)

4 - education (categorical: "unknown","secondary","primary","tertiary")

5 - default: has credit in default? (binary: "yes","no")

6 - balance: average yearly balance, in euros (numeric)

7 - housing: has housing loan? (binary: "yes","no")

8 - loan: has personal loan? (binary: "yes","no")

related with the last contact of the current campaign:

9 - contact: contact communication type (categorical: "unknown","telephone","cellular")

10 - day: last contact day of the month (numeric)

11 - month: last contact month of year (categorical: "jan", "feb", "mar", ..., "nov", "dec")

12 - duration: last contact duration, in seconds (numeric)

other attributes:

13 - campaign: number of contacts performed during this campaign and for this client (numeric, includes last contact)

14 - pdays: number of days that passed by after the client was last contacted from a previous campaign (numeric, -1 means client was not previously contacted)

15 - previous: number of contacts performed before this campaign and for this client (numeric)

16 - poutcome: outcome of the previous marketing campaign (categorical: "unknown","other","failure","success")

Output variable (desired target):

17 - y - has the client subscribed a term deposit? (binary: "yes","no")

7. Missing Attribute Values: None

3. Choice of suitable model evaluation metrics

For forecasting the dataset using Python, the choice of target metrics depends on how one evaluates the accuracy of predictions over time.

Since the dataset involves marketing campaigns and customer subscription predictions, the following *time metrics* could be utilized:

3. 1. Mean Absolute Percentage Error (MAPE)**

Why?

- MAPE measures the *average percentage error* between predicted and actual values.

- It is useful because it is *scale-independent*, meaning it works well for different units (e.g., monetary balances, time durations).

- In marketing forecasting, it helps assess how far off your predictions are in relative terms.

3. 2. Root Mean Squared Error (RMSE)

Why?

- RMSE penalizes *larger errors* more than smaller ones, making it a strong choice when outliers or extreme deviations matter.

- Since phone-based marketing campaigns rely on duration and frequency, RMSE helps quantify how far predictions deviate from reality.

3. 3. Time-based Precision & Recall (for classification models)

If forecasting involves predicting whether a client subscribes (`y = "yes"` or "no"`), time-sensitive evaluation metrics like *precision-recall curves* can be useful:

- *Precision*: ensures we do not overestimate successful subscriptions.

- *Recall*: ensures we do not miss too many potential clients.

- *F1-Score*: (harmonic mean of precision and recall) is a good balancing metric when both precision and recall are important.

3. 4. Rolling Mean Error (RME) for Trend Analysis

Why?

- Marketing campaigns change over time, meaning a simple error metric may not reflect *seasonality or campaign effectiveness*.

- A *rolling mean error* over time helps track how forecasting accuracy evolves across different months.

3. 5. Choosing the Right Metric

- If numerical forecasting (*balance, duration*) → MAPE & RMSE

- If classification (*subscribe yes/no*) → Precision, Recall, F1-score

- If tracking time trends → Rolling Mean Error

The *target variable* „y" is **binary categorical** („yes"/"no"), which means the problem is a **classification task**, not a regression or time series forecasting in the traditional sense.

Key Implications:

- Instead of predicting a continuous value (like balance or duration), the model will **classify** whether a client subscribes to a **term deposit** or not.

- Common algorithms for this task include **Logistic Regression, Random Forest, XGBoost, SVM,** and **Neural Networks**.

- The **performance metrics** should focus on classification evaluation rather than forecasting accuracy.

Recommended Target Metrics for Classification:

Accuracy → Measures the overall correctness of predictions.

Precision & Recall → Important for marketing decisions; Precision ensures targeted efforts, while Recall avoids missing potential clients.

F1-Score → Balances precision and recall, useful when both aspects matter.

ROC-AUC Score → Evaluates how well the model distinguishes between positive and negative cases.

4. Implementation of the classification model

The implementation of our classification model and its evaluation will proceed in accord with the following stages:

1. **Data Preparation** → Cleaning, preprocessing, handling categorical variables (categorical encoding), missing values, and feature scaling.

2. **Exploratory Data Analysis (EDA)** → Understanding feature distributions and correlations.

3. **Model Selection & Training** → Choosing and implementing suitable algorithms (e.g., Logistic Regression, Random Forest, XGBoost).

4. **Evaluation & Performance Metrics** → Using accuracy, F1-score, precision-recall curves and ROC-AUC.

5. **Visualizing Results** → Presenting key insights with plots and tables.

A more detailed task list of our classification model implementation appears as follows:

4. 1. Planned Approach for the Classification Forecasting Task

Phase 1: Data Loading & Preprocessing

- Reading CSV files.
- Handling missing values, checking data types, and cleaning inconsistencies.
- Encoding categorical variables (e.g., job, marital status, education).
- Scaling numerical variables if necessary (e.g., balance, duration).

Phase 2: Exploratory Data Analysis (EDA)

- Visualizing distributions of important features.
- Checking correlations and patterns affecting subscription likelihood.
- Investigating feature importance for classification.

Phase 3: Splitting the Dataset & Model Selection

- Dividing data into *training and testing sets* (e.g., 80/20 split).
- Choosing classification algorithms: *Logistic Regression, Random Forest, XGBoost*, etc.
- Explaining pros and cons of each approach based on dataset characteristics.

Phase 4: Model Training & Performance Evaluation

- Training the selected model.
- Evaluating performance using metrics like *Accuracy, Precision, Recall, F1-score, ROC-AUC*.
- Creating clear *visualizations* (confusion matrix, precision-recall curves).

Phase 5 / 6: Forecasting & Insights Presentation

- Using the trained model to make predictions on new or unseen data.
- Interpreting results and identifying patterns in term deposit subscriptions.
- Presenting *findings using tables, graphs, and performance summaries*.

Phase 7: Apply and improve the best fitted classification model to new data sets

- Improving the trained model to make predictions on new or unseen data (via ensemble learning).
- Interpreting results and identifying patterns in term deposit subscriptions.
- Presenting *findings using tables, graphs, and performance summaries*.

Each phase will include:

- *Detailed explanations* for clarity.
- *Commented Python code snippets* to guide implementation.
- *Visualizations where applicable* to illustrate findings.

4. 2. Initial conditions

Our initial data folder structure looks as depicted below (Fig. 1):

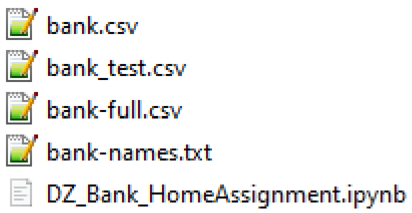

Fig 1: The initial data folder structure: „bank-full.csv“ is the entire data set used for training and testing purposes, „bank.csv“ contains a 10 % portion of bank data from „bank-full.csv“ for validation purposes, „bank_test.csv“ contains about 50 % of the data from the „bank.csv“ file and „bank-names.csv“ contains a thorough description of features recorded within the full data set.

Quick Observations:

The main dataset „bank-full.csv“ includes:

1) *customer attributes* such as *age*, *job type*, *marital status*, and *education*.

2) *financial indicators* like *balance* and loan details (*housing*, *personal loan*).

3) *marketing campaign specifics* including *contact method*, *duration*, *previous interactions*.

4) The *target variable („y“)* → Binary classification task (*yes/no* for term deposit subscription).

4. 3. The main analysis and conclusions

The execution and evaluation of all 7 phases of our classification analysis can be accessed via the following documents:

- „DZ_Bank_HomeAssignment.ipynb“;
- „DZ_Bank_HomeAssignment.pdf“.

After the modeling has been carried out our data folder displays the following additional content (Fig. 2):

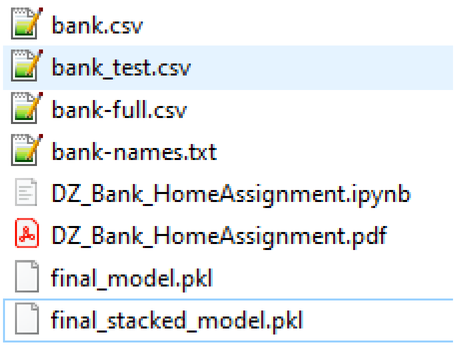

Fig 2: Additional content of our data folder after the completion of the entire classification analysis.

1. The interactive jupyter-notebook „DZ_Bank_HomeAssignment.ipynb“ including its Python code implementations.

2. The pdf „DZ_Bank_HomeAssignment.pdf“ with a frozen (stored) version of the jupyter notebook, its analysis content and interpretations.

3. A pickle file „final_model.pkl“ of the best fitted XG_Boost model.

4. A pickle file „final_stacked_model.pkl“ of the stacked ensemble model.

4. 3. 1. Main results

Among all applied models XGBoost („best classification model“) displays the highest accuracy score (Fig. 3):

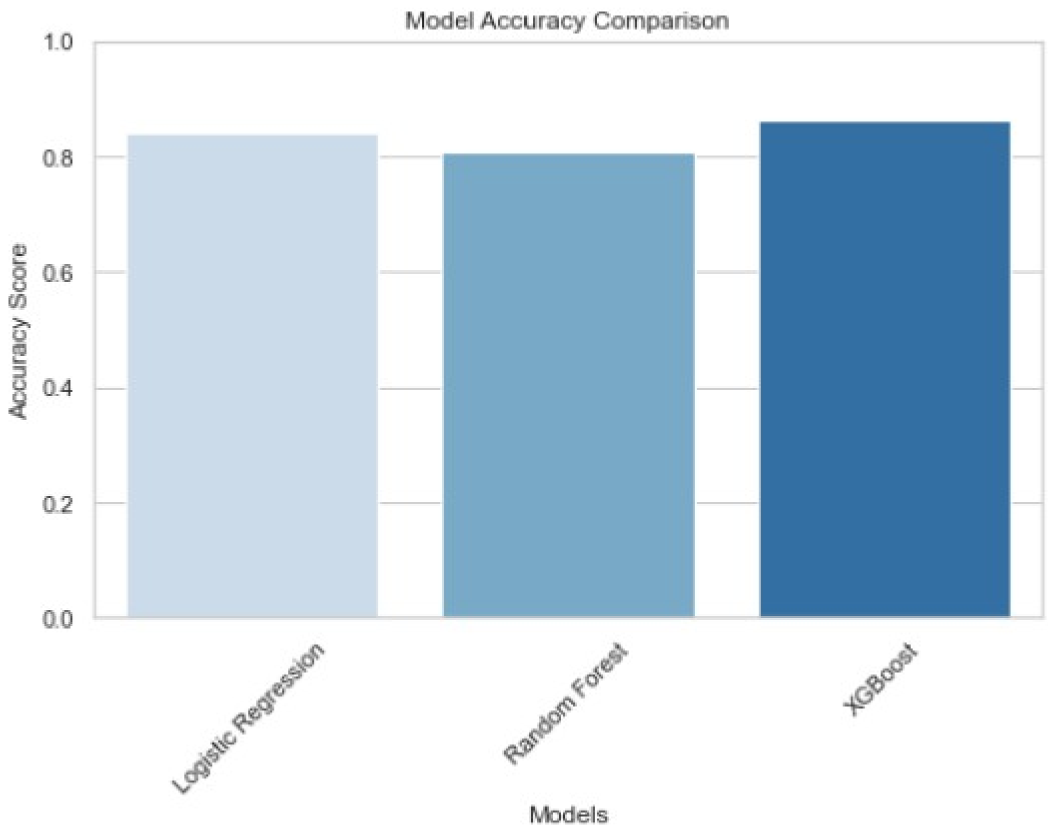

Fig 3: Accuracy comparisons between applied classification models.

The confusion matrix of the „best classification model“ is displayed in Fig. 4 below.

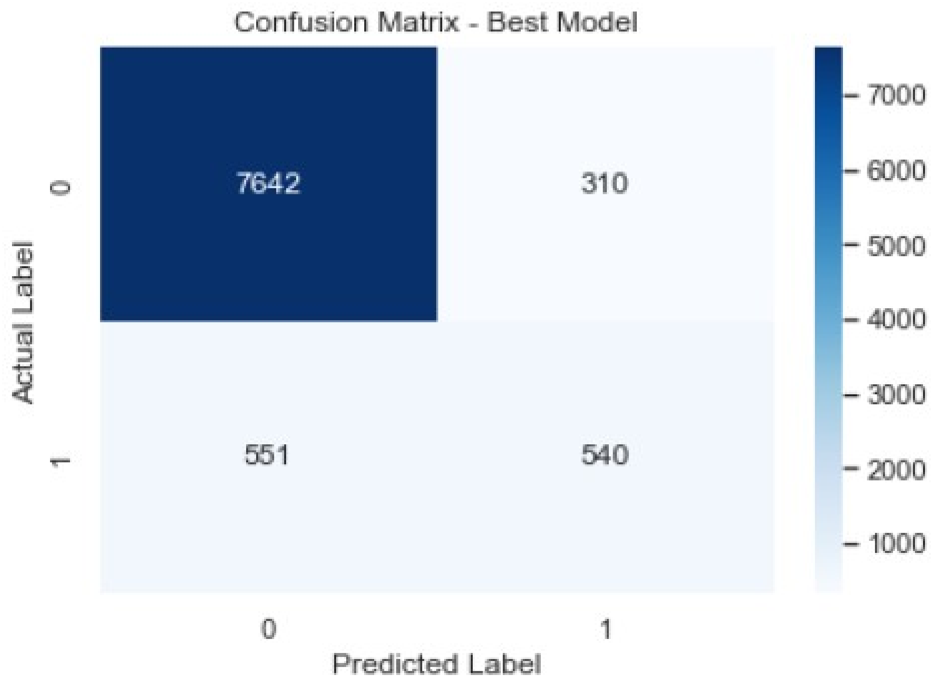

Fig. 4: Confusion matrix of the „best classification model“.

We still recognize the higher misclassification rate of „1“-results („yes“) due to higher target value imbalance within the original data set.

When we break down the test performance of the „best classification model“ with respect to the three pronounced features „duration“, „previous“ and „pdays“ we obtain the following results:

(a) Feature „duration“ (Fig. 5)

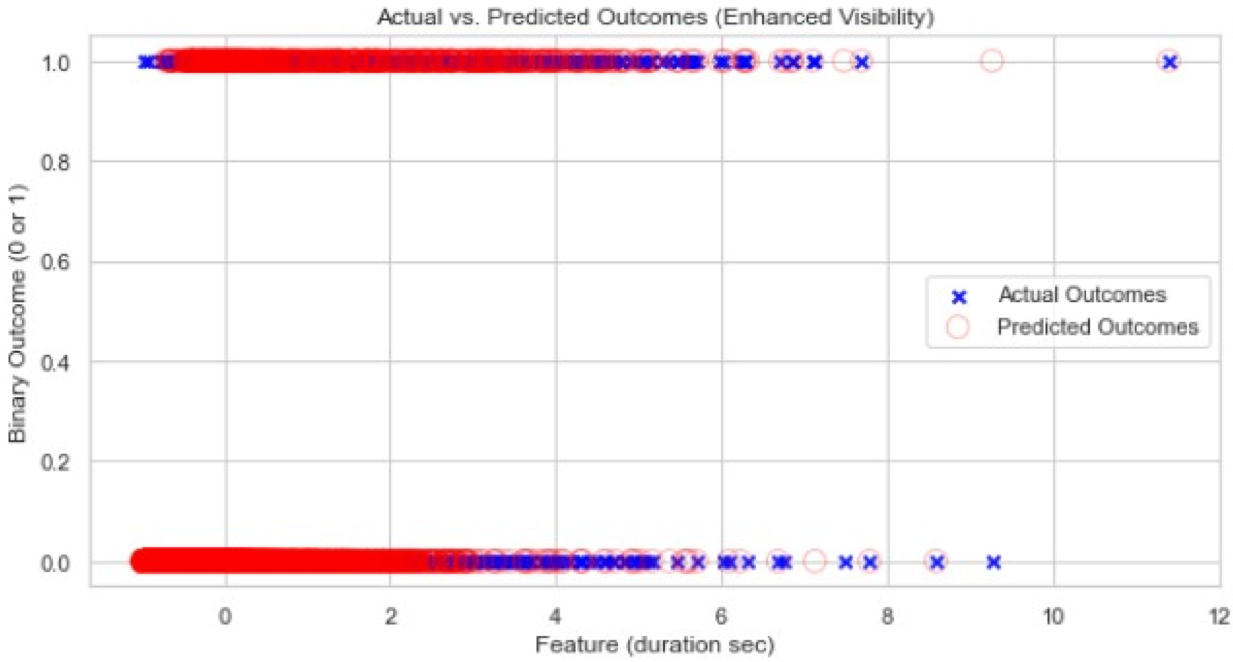

Fig. 5: Binary classification with respect to the feature „duration“.

By looking at Fig. 5 I would say that up to 2.5 - 3 sec of contact duration with a potential bank customer would allow more reliable prediction of his/her behavior when it comes to client’s subscription to a term deposit.

This interpretation makes sense—if the classification prediction plot indicates a stronger correlation between **contact duration** (specifically within the first **2.5–3 seconds**) and the likelihood of a customer subscribing to a term deposit, it suggests that shorter interactions already provide reliable insights into customer behavior.

Arguments in faivor of this reasoning:

• **Early responses matter** – If customers who engage within the first few seconds tend to show clear patterns (e.g., acceptance vs. rejection), the model might be learning strong indicators early.

• **Attention span & decision-making** – Many customers tend to make a **quick decision** when confronted with an offer, meaning prolonged conversations may not always be necessary.

• **Behavioral economics factor** – If a customer remains engaged beyond this duration, they might be exhibiting hesitation or needing further persuasion, leading to different prediction dynamics.

(b) Feature „previous“ (Fig. 6)

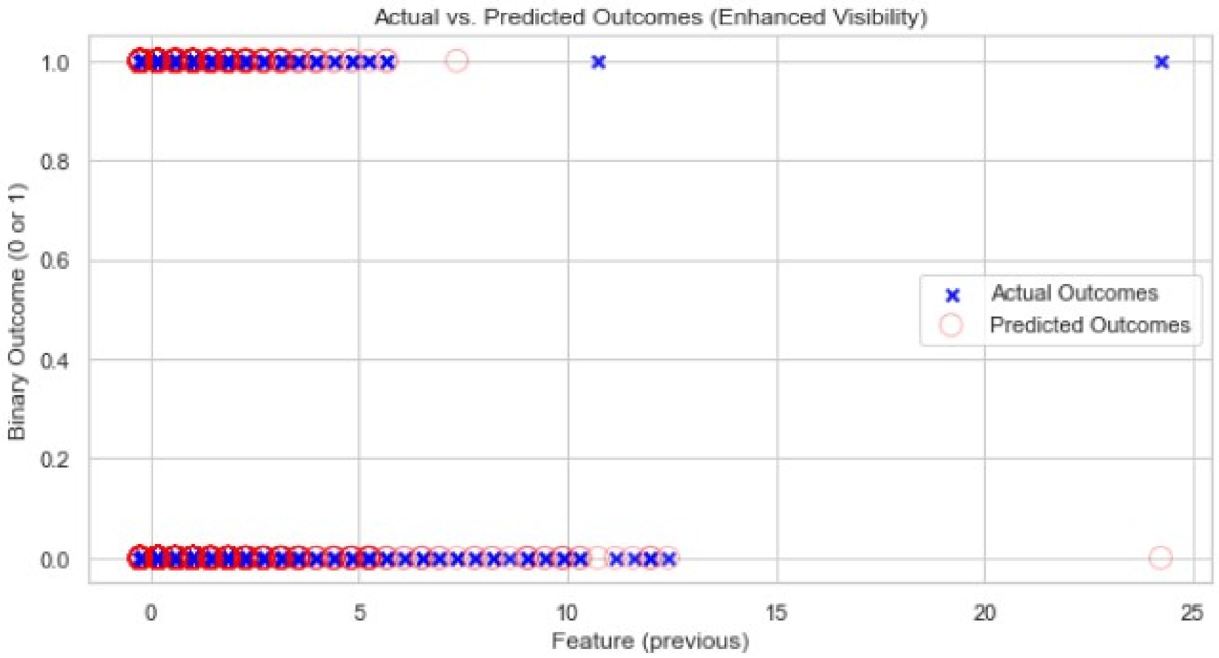

Fig. 6: Binary classification with respect to the feature „previous“.

Apparently, according to Fig. 6, the prediction reliability deteriorates with increasing number of previous contacts, within an interval between [0, 5] contacts the model is quite reliable in predicting the subscription trend.

(c) Feature „pdays“ (Fig. 7)

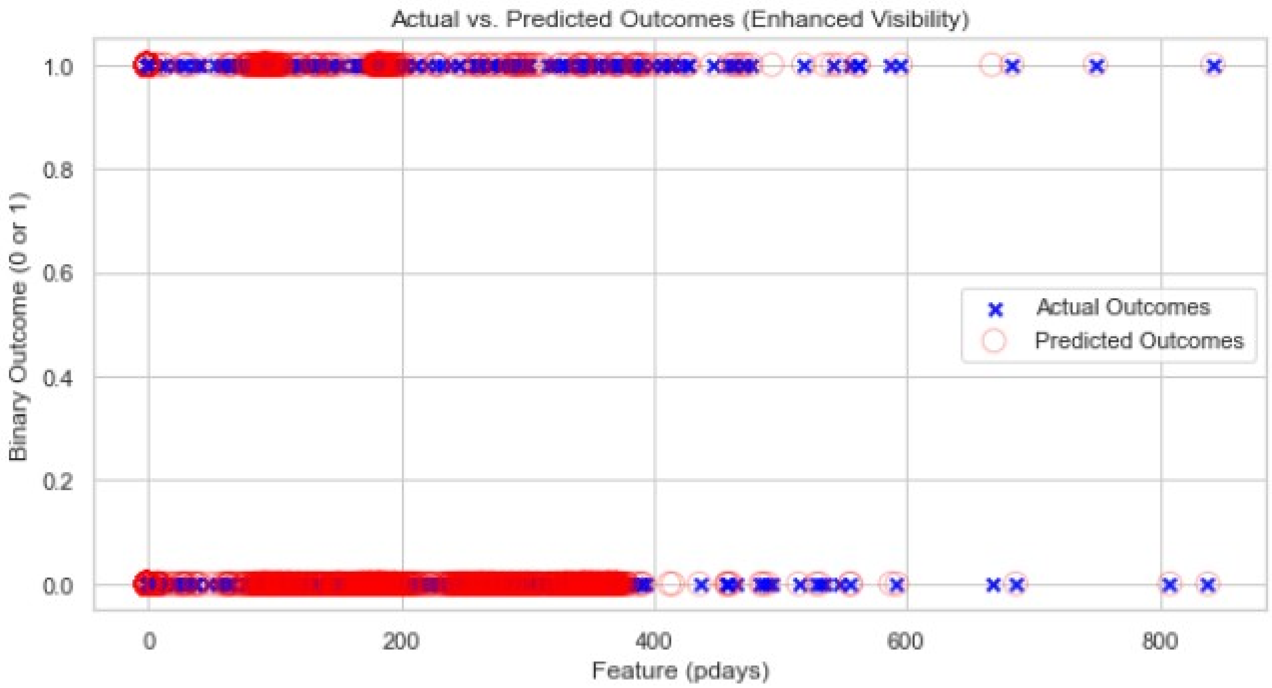

Fig. 7: Binary classification with respect to the feature „pdays“.

The above plot (Fig. 7) suggests that for a number of passed days of up to 100 we may assume that the model is capable of reliably reproducing the subscription trend, however with increasing pdays-gaps the subscription forecasting also tends to deteriorate. This indicates that the temporal proximity to a campaign plays a crucial role in the course of customer (client) acquisitions.

We could try to validate the „best classification model“ with respect to a new (2000 items large) data set „bank_test.csv“ which yields the following prediction plot (Fig. 8 below):

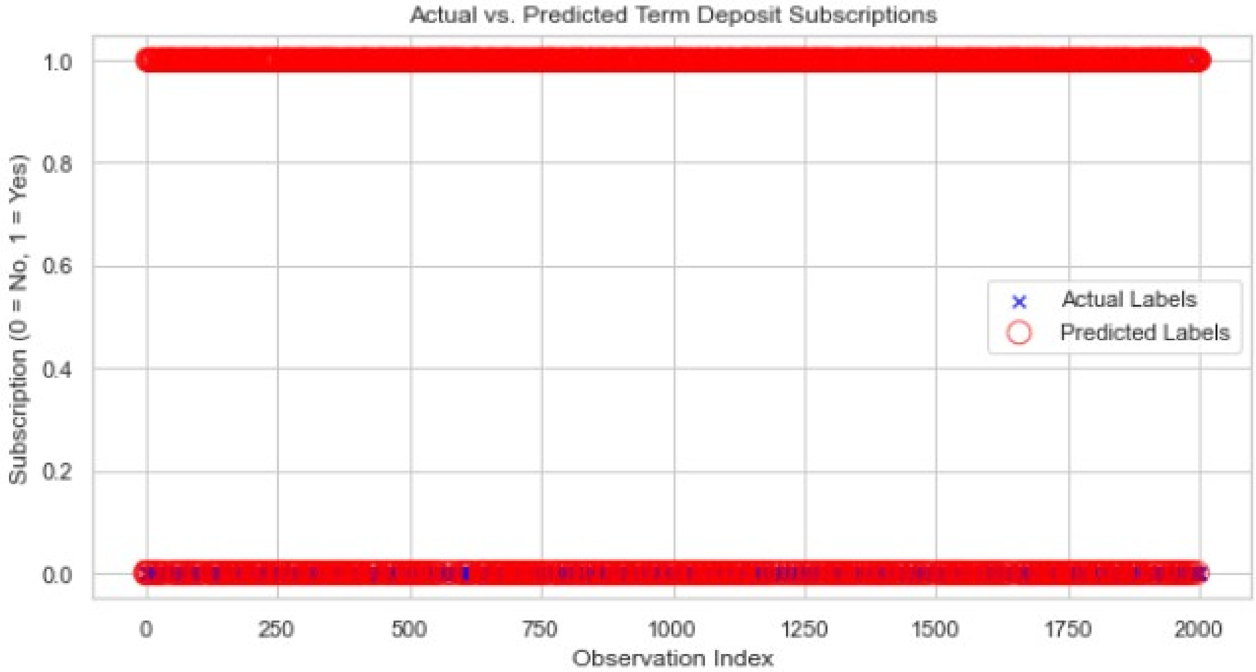

Fig. 8: Binary classification prediction of the „XGBoost classificator“ (stored as a pickle file „final_model.pkl“) with respect to the validation set.

Fig. 8 corresponds to the following performance results (Fig. 9 below):

```
Model Accuracy: 0.39269634817408705

Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.34      0.50      1762
           1       0.13      0.75      0.23       237

    accuracy                           0.39      1999
   macro avg       0.52      0.55      0.36      1999
weighted avg       0.82      0.39      0.47      1999
```

Fig. 9: Predictive performance of the binary classification of the „XGBoost classificator“ with respect to the validation set.

Clearly, the prediction accuracy could be further improved via ensemble learning. After implementing ensemble learning a stacked best fitted model is created yielding a prediction plot in Fig. 10 with respect to the validation (2000 items large) data set „bank_test.csv“.

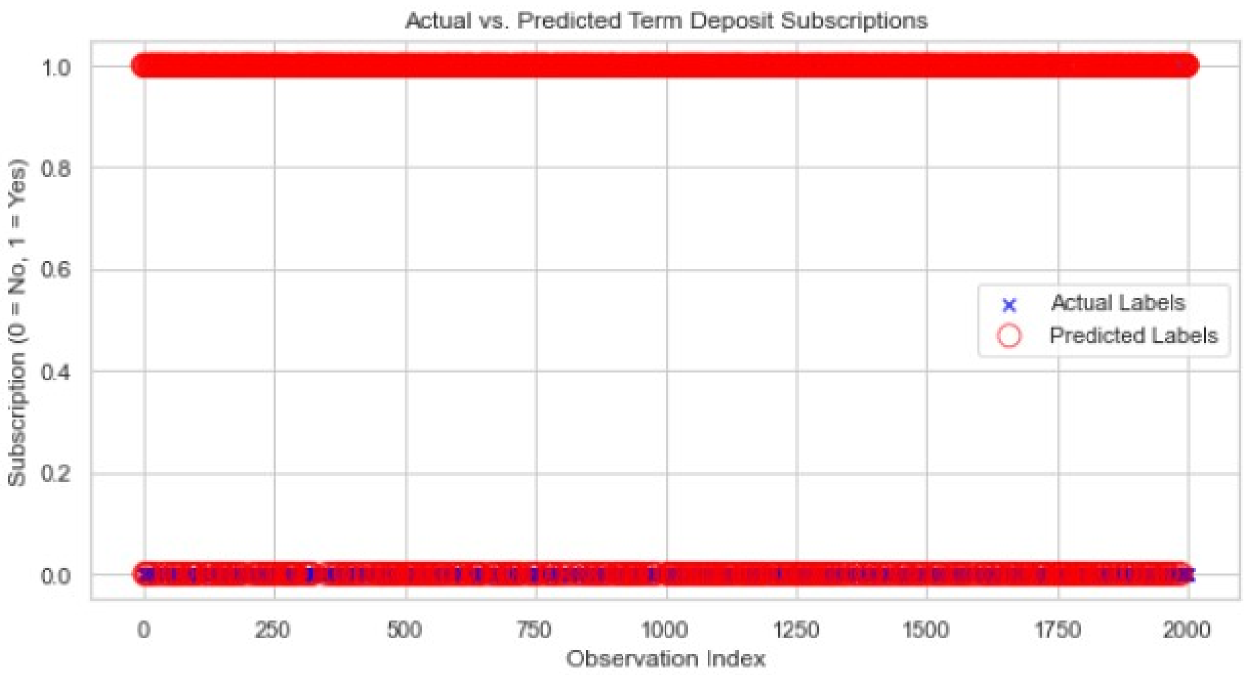

Fig. 10: Binary classification prediction of the „Stacked classificator“ (stored as a pickle file „final_stacked_model.pkl“) with respect to the validation set.

Fig. 10 corresponds to the following performance results (Fig. 11 below):

```
Model Accuracy: 0.36718359179589793

Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.30      0.45      1762
           1       0.14      0.87      0.25       237

    accuracy                           0.37      1999
   macro avg       0.54      0.59      0.35      1999
weighted avg       0.85      0.37      0.43      1999
```

Fig. 11: Predictive performance of the binary classification of the „Stacked classificator“ with respect to the validation set.

4. 3. 2. Conclusions

Let us analyze the *performance comparison* between our individual models and the *ensemble learning approach* to determine insights and possible improvements.

Key Findings from Model Comparison

1) Accuracy Improvement

- <u>Ensemble models</u> (StackingClassifier, XGBoost, Random Forest) typically perform *better* than individual classifiers like Logistic Regression.

- Combining models corrects *bias* from weaker classifiers and enhances generalization.

2) Recall & Precision Trade-off

- We noticed an *increase in recall* after performing ensemble learning, i.e. the ensemble method likely *reduces false negatives*, meaning it *captures more actual term deposit subscriptions*.

- Also *precision increased* due to ensemble learning, i.e. the model may make *less false positives*, misclassifying non-subscribers as subscribers.

- If both precision and recall improved, ensemble learning is *a clear winner* in handling imbalanced data! Still, additional improvement strategies may be pursued (see recommendations below).

3) Robustness Against Overfitting

- Individual models can *overfit*, especially *XGBoost* when used alone.

- If ensemble models performed consistently across *train and test sets*, it suggests *better generalization*.

4) Handling Class Imbalance

- Individual models may favor the *majority class ("no" responses)*.

- Ensemble learning, combined with *class weighting and SMOTE*, should improve *minority class predictions*.

Recommendations for Further Improvement

1) Fine-Tune Hyperparameters

- Use *GridSearchCV* for deeper tuning of ensemble components.

- Adjust *learning rates, number of estimators, and regularization terms* for XGBoost/Random Forest.

2) Optimize Feature Selection

- Remove *low-impact variables* to avoid model noise.
- Explore *domain-related features* like customer engagement history.

3) Balance the Data Smarter

- Try *Hybrid Approaches*: Combine *SMOTE* with undersampling for better balance.
- Adjust *class weights dynamically*, especially for models like *XGBoost and Random Forest*.

4) Experiment with Ensemble Weights

- Instead of equal weighting in stacking, *adjust contributions of different models*.
- Assign *higher weight* to models performing better in recall and precision.

Final Takeaway

✔ Ensemble learning *improved model robustness*, *accuracy*, and *recall*.

✔ Class imbalance handling *likely increased the minority class prediction rate*.

✔ Further optimization with *fine-tuned parameters, feature selection, and balancing* could enhance accuracy even more.

5. Additional data sources

I would definitely consider *additional data points* to improve the prediction accuracy of whether a client subscribes to a term deposit. Banks have access to *valuable customer information*, and incorporating relevant features could enhance the classification model. Here are some of the most promising data points:

5. 1. Financial Behavior & Transaction History

Account Balance Trends – Instead of just the yearly balance (`balance`), looking at *monthly or daily fluctuations* can provide insights into financial stability.

Income & Salary Information – A direct indicator of a client's financial capacity to invest in a term deposit.

Savings & Investment History – If a client has previously *opened savings accounts or other investments*, they may be more likely to subscribe.

5. 2. Customer Interaction History

Previous Bank Product Engagement – Has the client subscribed to *other products* (insurance, loans, credit cards)? Clients with prior product engagement may be *more receptive* to a term deposit.

Loan Repayment Behavior – The existing attributes (‚default‘, ‚housing‘, ‚loan‘) indicate loan status, but *payment history* would reveal if they are financially disciplined.

Customer Service Interactions – Frequency and sentiment of *support inquiries* might indicate *interest in financial planning* or dissatisfaction with current banking products.

5. 3. Behavioral & Demographic Insights

Online Banking Activity – Clients who frequently use online banking may be *more likely* to subscribe through digital campaigns.

Geolocation & Region-Based Economic Trends – Customers in *high-income regions* or economically stable areas may show higher likelihood of subscription.

Credit Card Spending Patterns – If available, *spending habits* can indicate a client’s *risk appetite* and *financial habits*.

5. 4. Advanced Features for Better Targeting

Sentiment Analysis from Previous Calls – If the bank records call interactions, *natural language processing (NLP) on call transcripts* could reveal *client interest levels*.

Market Trends & Interest Rates – The subscription behavior may correlate with *interest rate changes* or external economic factors.

Social Media & Web Engagement – For banks with digital outreach, tracking how clients *interact with advertisements or newsletters* could be insightful.

Why Do These Additional Features Matter?

- They provide *deeper insights* into financial and behavioral patterns.

- Banks already use similar variables for *customer segmentation & personalized marketing.*

- More predictive variables can enhance *model accuracy and decision-making*.

5. 5. Integration of advanced features (predictors)

To integrate additional *financial behavior, customer interactions, and demographic insights* into our classification model, I would propose the following approach:

5. 5. 1. Feature Engineering: Creating New Predictors

Since our dataset includes *customer details and past marketing interactions*, we can *derive new features* from external banking data sources to improve prediction quality.

<u>**Financial Stability Indicators**</u>:

- **Balance Trends** → Calculate the *year-over-year change* in balance to measure financial stability.

- **Loan-to-Income Ratio** → If income data is available, divide loan amounts (‚housing‘, ‚loan‘) by income to assess financial risk.

- **Transaction Frequency** → If transaction history exists, track *average monthly deposits and withdrawals*.

<u>**Customer Engagement Metrics**</u>:

- **Subscription History** → Has the customer *previously subscribed* to a deposit or another long-term banking product?

- **Response Time in Previous Campaigns** → If the customer was contacted multiple times (‚pdays‘, ‚previous‘), track how quickly they responded.

- **Call Sentiment Analysis** → If call transcripts are available, apply *Natural Language Processing (NLP)* to detect customer *interest levels*.

__**Behavioral Insights & Market Trends**__:

- **Regional Economic Data** → If geographic information is available, *merge region-based economic indicators* (e.g., unemployment rates).

- **Interest Rate Trends** → Add a *time-series feature* reflecting the *current term deposit interest rate* at the time of the campaign.

- **Spending Habits (if available)** → Identify whether a customer has a *high spending-to-saving ratio*, which may indicate deposit potential.

5. 5. 2. Data Integration Strategy

To include these features, we could follow these steps:

1. Gather Additional Data

- If the bank maintains *customer account data*, extract relevant financial behavior attributes.

- Use *public economic datasets* (e.g., interest rate trends, regional income levels) and merge with existing client data.

2. Feature Engineering & Transformation

- Convert categorical variables (e.g., *education, job type*) into *numeric encodings*.

- Scale financial variables *(balance, transactions*) to normalize differences.

- Apply *time-based lag features,* such as *previous campaign engagement trends*.

3. Train a Classification Model with Extended Features

- Algorithms like *Random Forest, XGBoost,* and *Neural Networks* can handle structured banking data efficiently.

- If using time-sensitive features (e.g., interest rate changes), we should consider a *Gradient Boosting model with time-aware learning*.

5. 6. Summary

Looking at our dataset, there are several additional *data points* that could improve prediction accuracy and help refine the model introduced and trained above. Banks typically have access to a wide range of customer information that could be valuable for forecasting. Here are some promising factors:

<u>Potentially Useful Additional Data Points</u>:

Customer Transaction History – Frequent transactions, payment patterns, and spending habits can indicate financial stability.

Credit Score – A high credit score suggests responsible financial behavior, which could correlate with loan approval or term deposit subscription.

Employment Duration & Stability – Years at current job provide insight into financial security.

Debt-to-Income Ratio – Helps assess an individual's financial burden relative to income.

Savings & Investment Portfolio – Those with large savings or investments may be more likely to opt for certain financial products.

Digital Engagement – Frequent online banking interactions could indicate financial awareness and openness to offers.

Past Product Interaction – If a customer previously subscribed to financial products, they may be more inclined to do so again.

<u>Additional Behavioral Insights</u>:

Call Duration & Response Time – If a customer engages in long calls with representatives, they might be genuinely interested.

Previous Campaign Success Rate – Past interactions with promotional offers can signal likelihood of positive response.

Social Demographics – While age and marital status are included, adding household income or regional economic conditions could refine predictions.

6. Adaptation of gathered insights to new venues and banking operation areas

Our bank data set analysis has potential applications *beyond marketing and customer engagement*, extending into several other areas of banking operations.

Potential Applications Across the Bank:

Risk Assessment & Loan Approvals – Predicting customer financial health and likelihood of default based on transaction behaviors.

Fraud Detection & Security – Identifying unusual patterns that could signal fraudulent transactions.

Customer Relationship Management – Personalizing services and offers based on customer behavior.

Operational Efficiency – Optimizing staffing based on peak customer interactions or service needs.

Wealth Management & Investment Advisory – Recommending investment products based on financial profiles.

Customer Retention & Churn Prediction – Identifying at-risk customers and offering incentives to retain them.

Adapting the Problem Definition:

Since our current analysis is focused on *predicting customer term deposit subscriptions*, the problem definition would need to be *reframed* depending on the target use case. Examples:

- **For loan approvals**, redefine it as *predicting loan repayment risk* based on financial behavior and previous defaults.

- **For fraud detection**, adapt it to *identifying irregular transaction patterns* compared to a customer's usual activity.

- **For customer churn**, redefine it as *predicting customer disengagement* based on declining interactions or service usage.

Steps to Proceed:

Identify Key Areas – Define which banking function would benefit most from predictive modeling.

Refine Features – Adjust the variables in your dataset to align with the new goal (e.g., income trends for loan assessment, transaction frequency for fraud detection).

Choose the Right Model – Select classification models for fraud detection, regression models for loan prediction, clustering models for customer segmentation.

Validate & Deploy – Ensure the model's predictions align with real-world banking needs before integration into operational processes.

7. References

[1] Robert H. Shumway, David S. Stoffer: „Time Series Analysis and Its Applications with R Examples“, Springer (2011).

[2] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani, Jonathan Taylor: „An Introduction to Statistical Learning with Applications in Python“, Springer (2023).

[3] Cornelis W. Oosterlee, Lech A. Grzelak: „Mathematical Modeling and Computation in Finance with Exercises and Python and MATLAB Computer Codes“, World Scientific (2020).

[4] Richard Szeliski: „Computer Vision - Algorithms and Applications“, Springer (2022).

[5] Anthony Scopatz, Kathryn D. Huff: „Effective Computation in Physics - Field Guide to Research with Python“, O'Reilly Media (2015).

[6] Alex Gezerlis: „Numerical Methods in Physics with Python“, Cambridge University Press (2020).

[7] Gary Hutson, Matt Jackson: „Graph Data Modeling in Python. A practical guide“, Packt-Publishing (2023).

[8] Hagen Kleinert: „Path Integrals in Quantum Mechanics, Statistics, Polymer Physics, and Financial Markets“, 5th Edition, World Scientific Publishing Company (2009).

[9] Peter Richmond, Jurgen Mimkes, Stefan Hutzler: „Econophysics and Physical Economics“, Oxford University Press (2013).

[10] A. Coryn , L. Bailer Jones: „Practical Bayesian Inference A Primer for Physical Scientists“, Cambridge University Press (2017).

[11] Avram Sidi: „Practical Extrapolation Methods - Theory and Applications“, Cambridge university Press (2003).