Team: "" (pronounced "empty string")

Members: Noah Stäuble, Michal Mikuta, Elias Mbarek, Mikael Makonnen

Challenge: Julius Bär – Onboarding Quest

Setting

Private banks face significant delays and errors in manual client verification processes that could be avoided using emerging technologies. For this purpose we developed an explainable machine learning model to automate KYC checks and predict client acceptance based on their filed documents. We worked with a dataset of 10,000 client profiles containing passport, demographic, account, and KYC information. The model flags inconsistencies and provides clear justifications for each decision, enabling faster and more reliable onboarding while ensuring regulatory compliance.
Our primary goal was to develop not only an accurate acceptance-rejection algorithm but also one that was **explainable** and that could handle textual data. To achieve this, we combined a rule-based approach with a machine learning pipeline. From the outset, we implemented algorithmic rules that client data had to satisfy in order to be accepted (see examples below). In parallel, we designed a flexible ML framework by aggregating data, conducting exploratory analysis, and engineering features. Ultimately, we arrived at a hybrid system in which both approaches complemented each other. The rule-based logic was translated into features for the ML model, while discrepancies detected by the ML pipeline informed new rules that could be applied directly. This dual structure resulted in a robust and interpretable system. In fact we achieve high accuracies of 0.72 while being able to completely explain every single rejection.
Our approach consisted of three stages:

1. **Tabular Data Modeling:** We defined acceptance and rejection rules, selected relevant features, and trained interpretable ML models such as decision trees and gradient boosting classifiers.
2. **Natural Language Processing:** We handled unstructured data, such as free-text notes and passport strings, using embedding models and LLM-based retrieval techniques.
3. **Explainability & Integration:** We aligned ML outputs with rule-based justifications to ensure transparency and maintain regulatory compliance.

Divide and Conquer: A Rule-Based Approach

We found multiple discrepancies in the data, contained in the four documents at our disposal for each client. We systematically evaluated them through the metric of accuracy and especially by using confusion matrices. As our base-line we implemented rules that look at differences amongst entries, like date entries and name entries with regex, expired passports,... We then turned to more sophisticated rules, like checking the timelines in the client description files and at the later stages using LLM parsing methods. Here are a few interesting rules we want to highlight:

Child prodigies:

We have identified multiple clients who have entered their graduation date (be it secondary or tertiary studies) to have happened during their early childhood (age <= 17 years old). These entries are flagged.

Hard-working or hardly working…

An interesting rule that we wanted to introduce was the rule of rejecting clients if they had started their employment before graduating, as this is a pattern we recognized in rejected cases. However it turns out that implementing this rule resulted in large true negatives. We couldn't explain this result and therefore decided to not use this rule. It might be that some other factors were at play when rejecting the client.

LLM parsing

We have found the client_description.json freeform text to be difficult to parse using simple regex/rule-based approaches. As such we have opted to query GPT4o-mini to identify discrepancies in property prices, inheritance details, and AUM. We settled for gemini-1.5-flash-8B for a good trade-off between speed and quality. Mistral would have been 15% faster but with a significant loss in accuracy.

Transform and Triumph: Harnessing Feature Engineering

In parallel to the approach described above we implemented a basic feature engineering pipeline and performed exploratory data analysis. We noticed that in the data itself we couldn't find any clear indicator of acceptance or rejection. The data was very unbiased, all the acceptance and rejection labels having the same ratio on the different values of the categorical features. Furthermore we didn't find any significant correlation between features and acceptance or rejection. The feature with the highest correlation to the acceptance label was the inherited amount of wealth (though this does not imply causation). However when we introduced filters that we added as features and would just perform an exact match check on the same features that would appear in the four different files, we noticed a huge predictive power. We therefore created a clean dataset specifically curated for the ML approach, getting rid of the different files and unifying the same features, but adding the discrepancy mask as a feature in our clean dataset. A penalized logistic regression with L1 loss, which provides sparse features for a big penalization parameter (being the best convex approximation of the l_0 norm and therefore easy to compute) and a K-best subset feature selection we performed on our transformed and clean data with the discrepancy features showed that the most significant features were the discrepancy features by a large margin and then features related to the wealth of the client.
To take advantage of the natural language data in the files, we embedded the files into embeddings, therefore vectorizing them and creating a new set of features, consisting of the dimension of the embedding space (size of the vector). We used PCA to choose the best k-dimensional linear projection of the embedding space, and found ourselves with k(= 5) embedding features. As tree methods like XGBoost and Random Forest Classifiers have empirically proven to be the gold standard for tabular data and beating Deep Learning and Neural Network approaches by a large margin, we decided to focus on those. They are as well an aggregation of weak learners that perform a linear decomposition of the feature space, and therefore one could expect to get explainable results by overfitting a simple Tree on the model and therefore providing an explainable model of the ML approach. Eventually we opted for an ensemble meta-method (majority vote model), taking advantage of the rules we found before and our NLP and LLM approaches and having the advantage of outputting a confidence score, based on the outcome of the vote of the different models.

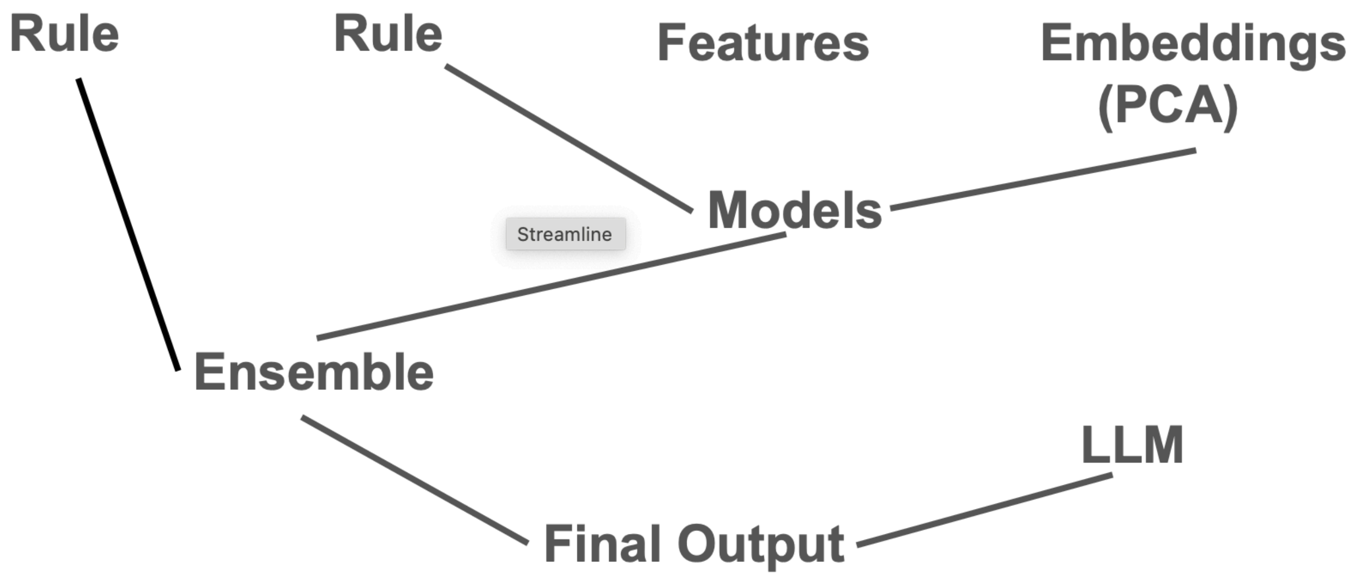
A Needle in the Haystack: Retrieving Relevant Information

As discussed above, we implemented an embedding model to augment our features with natural language data. After further detailed data analysis, when individually examining each rejected case, we noticed a significant pattern: cases were frequently dismissed due to mismatches between the summary description and the data provided in the three other files. These mismatches included instances of missing information in the files that were present in the summary or conflicting values between the summary and the other files. To address this, we implemented a retrieval framework prompting an LLM with relevant sections from client_profile.json and client_description.json, which should exactly match in their values. The model was prompted to return a JSON-formatted response indicating different keys found in equivalent sections, assigning 'false' as the value if there was no mismatch and 'true' if a mismatch was present. Although sensitive to the specific prompt formulation, this method produced very robust results following prompt engineering, validated through evaluation on the rejected cases available to us. Consequently, we obtained a powerful new feature capable of acting as a clear rule-based indicator and effectively leveraging textual data. Additionally, this feature provided explainability by precisely identifying the mismatch locations within the JSON files. We also dedicated significant effort toward accelerating LLM evaluation by parallelizing API calls and using slightly less accurate but lower-latency models (e.g., Gemini-1.5-flash instead of GPT-4o). This optimization allowed us to process around 1,000 clients in approximately 10 minutes, compared to the previously required hour or longer.

Conclusion and Challenges:

Our main challenge was identifying rules and methods capable of efficiently discarding true negatives that were undetected by our initial model. We encountered diminishing returns due to an abundance of soft filters (rules or binary features) that, although not introducing false negatives, contributed minimally to accuracy improvements and risked diluting the model's simplicity and explainability. The promising LLM retrieval approach that seemed to leverage a completely unused set of data ultimately had less impact than anticipated.
This realization led us to adopt a simple majority-voting as our final model, aggregating predictions from various machine learning and rule-based models which has the advantage of aggregating models that look at completely different data aspects and therefore covers as many possibilities as possible. We prioritized reasonable accuracy combined with strong explainability over higher accuracy with limited transparency. This decision is valuable because understanding the reasons behind rejections enables clear guidance for corrections. Our final model delivers high precision (accepted cases are reliably correct), albeit with modest recall. Despite imperfect accuracy, significant manual review workload is reduced, as only rejected cases require periodic checking. Furthermore, the model clearly communicates rejection reasons, streamlining the review process.

Appendix



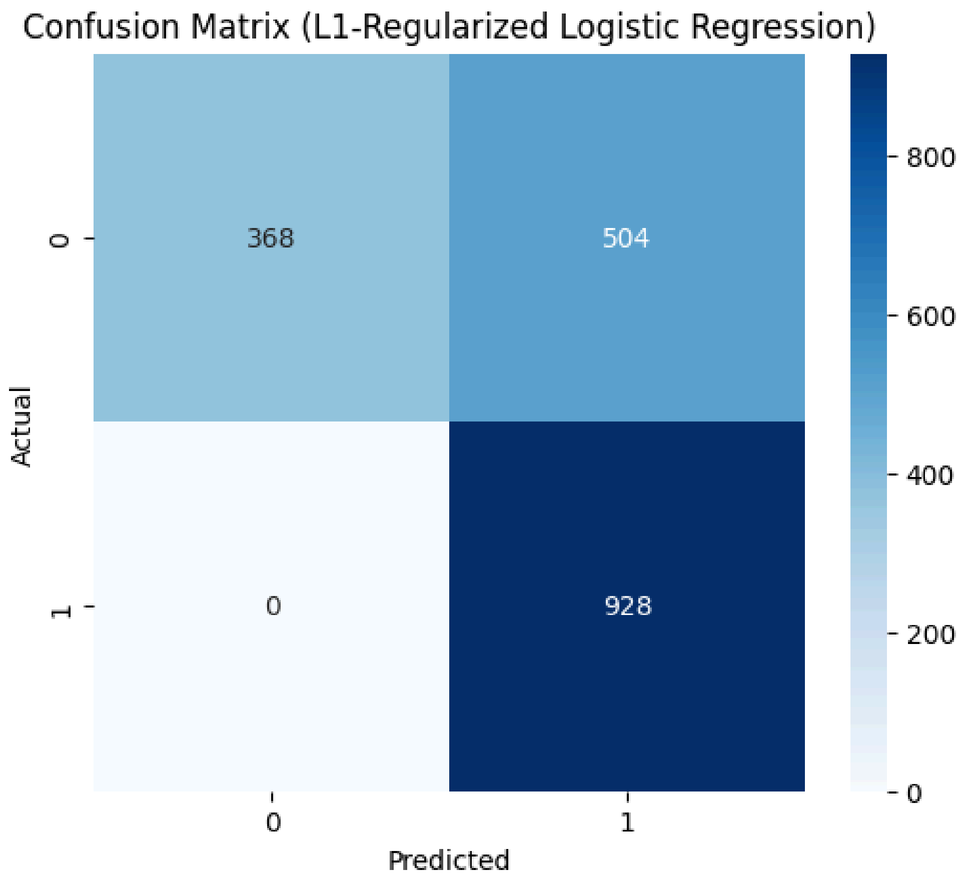
Complete model architecture employed, implementing majority voting and adding LLM improvements on top

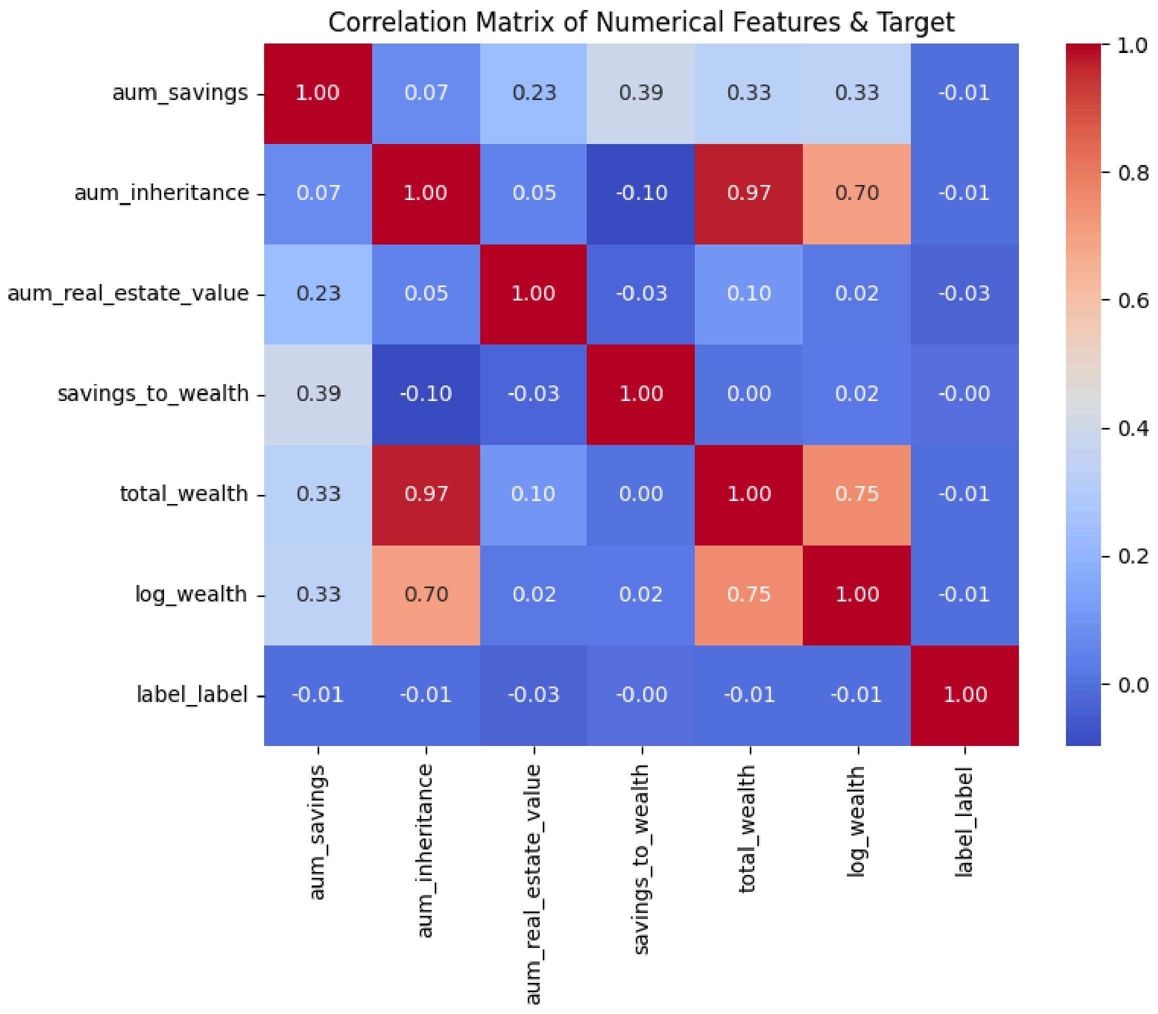
Selected Features based on L1 regularization for logistic regression	
Features	**Coefficients**
inconsistencies_phone_number	-5.399848
inconsistencies_last_name	-4.601399
inconsistencies_birth_date	-4.354373
inconsistencies_passport_expiry_date	-4.346649
inconsistencies_first_name	-4.183949
inconsistencies_nationality	-4.074496
inconsistencies_email_address	-4.038810
inconsistencies_country_of_domicile	-3.735925
inconsistencies_currency	-3.134006
inconsistencies_passport_number	-3.016217

Selected Features based on L1 regularization for logistic regression	
aum_real_estate_value	-0.0000000194
aum_savings	0.00000000645
aum_inheritance	-0.00000000343

Classification	**Precision**	**Recall**	**F1-score**	**Support**
0	1.00	0.42	0.59	872
1	0.65	1.00	0.79	928
accuracy			0.72	1800

This is the problem we face during most of the challenge, a perfect precision but a recall that needs to be optimized. It is very well represented by this simple naive baseline model that we used during our exploratory data analysis, using features, transformed features and inconsistencies in data. Below are the confusion matrix of the penalized logistic regression model and a correlation matrix of our features without the inconsistencies. Reject = 1, Accept = 0





	aum_savings	aum_inheritance	aum_real_estate_value	savings_to_wealth	total_wealth	log_wealth	label_label
aum_savings	1.00	0.07	0.23	0.39	0.33	0.33	-0.01
aum_inheritance	0.07	1.00	0.05	-0.10	0.97	0.70	-0.01
aum_real_estate_value	0.23	0.05	1.00	-0.03	0.10	0.02	-0.03
savings_to_wealth	0.39	-0.10	-0.03	1.00	0.00	0.02	-0.00
total_wealth	0.33	0.97	0.10	0.00	1.00	0.75	-0.01
log_wealth	0.33	0.70	0.02	0.02	0.75	1.00	-0.01
label_label	-0.01	-0.01	-0.03	-0.00	-0.01	-0.01	1.00