UNIVERSITÄT ZU KÖLN

MACHINE LEARNING

GABRIEL AUSTIN KESSLER - 7352773

KATERYNA TONIA - 7404636

LUCAS SALAMUNI - 7429674

Sparkasse Project: **Final Report**

Köln,
Summer 2025

Contents

Introduction

The project involves developing models for classifying banking e-mails into 12 categories with a high degree of class imbalance. Extensive feature engineering, including structural indicators, banking-specific keyword counts, and n-gram TF-IDF representations, is applied to address potential overfitting and reduce dimensionality. Three classification algorithms are implemented: Support Vector Machines, Random Forest, and Ensemble. Model performance is evaluated using a broad set of metrics, including Macro-F1, Weighted-F1, accuracy, precision, recall, ROC-AUC, and Log-Loss, alongside confusion matrices and per- class error rates.

All script's versions are available on the following public repository:
https://github.com/lsalamuni/Sparkasse_ML

1 Text Processing (R)

The `v5_rscript.R` constitutes a sophisticated and methodologically rigorous text processing and feature engineering pipeline specifically architected for the automated classification of German banking customer communications within the Sparkasse financial services ecosystem. This comprehensive computational framework transforms unstructured customer inquiry texts into a meticulously engineered feature matrix that simultaneously captures both fundamental linguistic patterns and highly specialized domain-specific banking characteristics, thereby establishing a robust foundation for multi-class classification models capable of autonomously routing customer communications to appropriate service categories while addressing the inherent challenges posed by severely imbalanced class distributions characteristic of real-world customer service applications. The implementation represents a synthesis of traditional natural language processing methodologies with contemporary dimensionality reduction techniques and domain-specific feature engineering approaches, ensuring that the resulting feature representation captures not only general linguistic patterns but also the nuanced semantic content specific to banking communications.

The architectural design of this text processing pipeline addresses several critical computational and linguistic challenges that are inherent in enterprise-level text classification systems for customer service applications, particularly within the highly regulated and terminology-rich banking sector. The script systematically confronts the class imbalance problem through comprehensive statistical analysis, implementing detailed metrics that quantify the distributional inequality across the twelve distinct inquiry categories, revealing that certain specialized categories represent less than ten percent of the total dataset while others dominate the distribution, thereby necessitating sophisticated handling mechanisms in both the feature engineering and subsequent model training phases. The methodology embraces established natural language processing best practices while incorporating domain-specific adaptations that recognize the unique linguistic characteristics, specialized terminology, and structural elements inherent in banking customer communications, which generic text processing approaches typically fail to capture adequately.

The initial data preparation and preprocessing phases establish the computational foundation through the systematic loading of essential R packages and the methodical preparation of the raw Excel-based dataset, converting it into a structured tibble format with standardized column nomenclature to ensure consistency and facilitate efficient downstream processing operations. The structural feature engineering component implements a comprehensive extraction methodology for quantitative features derived from email structural characteristics, including character and word counts, punctuation pattern analysis, sentence structure metrics, and capitalization pattern recognition, all of which capture the formal communicative characteristics of customer interactions such as urgency indicators, formality levels, and communication styles. Additionally, the script incorporates banking-specific structural pattern recognition capabilities, including International Bank Account Number (IBAN) detection algorithms and monetary amount recognition systems, which are crucial for accurate financial service classification and represent domain-specific enhancements that significantly improve classification performance.

The text tokenization and Term Frequency-Inverse Document Frequency (TF-IDF) processing components implement a sophisticated approach that employs both standardized German stopwords and custom banking-specific stopword lists, ensuring that common but informationally irrelevant terms are systematically filtered while preserving semantically meaningful content that contributes to classification accuracy. The implementation incorporates stemming algorithms specifically calibrated for the German language model, normalizing morphological word variations to their root forms and thereby reducing feature space dimensionality while preserving semantic content. The TF-IDF methodology captures the relative importance of terms within individual documents as compared to their frequency distribution across the entire corpus, effectively identifying discriminative terms that are most indicative of specific categories while simultaneously reducing the influence of commonly occurring but non-informative terms. The n-gram extraction process extends beyond traditional unigram analysis to incorporate bigram and trigram extraction, thereby preserving important multi-word expressions and contextual relationships that single-word analysis might overlook, which is particularly valuable in banking contexts where compound phrases carry significantly more semantic weight than their individual constituent terms.

The banking domain-specific feature engineering component recognizes the highly specialized nature of financial communications and implements comprehensive keyword detection algorithms across eight distinct semantic categories: account-related terminology, credit and loan-related expressions, payment and transaction-related terms, banking card-related vocabulary, investment and securities-related language, insurance and protection-related terms, online banking and digital services terminology, and customer service and support-related expressions. This systematic categorization ensures that banking-specific semantic content is explicitly captured and quantified, complementing the statistical text features with domain expertise that significantly enhances the discriminative power of the resulting feature representation. The dimensionality reduction and optimization component addresses the computational challenges associated with high-dimensional feature spaces through the implementation of Singular Value Decomposition (SVD) using the computationally efficient irlba algorithm, which reduces the potentially thousands of dimensions in the document-term matrices while preserving approximately 95% of the total variance, effectively capturing the most significant patterns in the textual data while eliminating noise and redundant information.

The final integration and machine learning optimization phase synthesizes all feature types into a unified dataset that is specifically optimized for machine learning applications, incorporating 19 structural email features that capture communication patterns and stylistic characteristics, 8 banking domain-specific keyword features that quantify specialized terminology usage, SVD-reduced text features typically comprising 30-50 components that capture the most significant linguistic patterns, and 30-dimensional text embeddings that encode semantic relationships and contextual meaning. The comprehensive sparsity analysis ensures computational efficiency while maintaining representational completeness, and the detailed feature analysis comparing minority and majority classes identifies the most discriminative features for classification tasks, providing valuable insights that guide feature selection and help understand which textual patterns are most indicative of different inquiry categories. The output dataset is systematically saved in multiple formats, including a dimensionality-reduced version optimized for computational efficiency and a full-feature version for comparative analysis, both maintaining the original categorical structure while providing rich numerical representations suitable for various machine learning algorithms, ultimately producing a robust, interpretable, and highly effective feature representation that addresses the key challenges of text classification in customer service environments while maintaining the semantic fidelity and domain-specific knowledge essential for accurate banking communication classification.

2 Model I - Support Vector Machine (R/Py)

The Support Vector Machine implementation employs a systematic approach to multi-class classification of German banking customer communications, addressing the critical challenges posed by severe class imbalance through strategic preprocessing, hyperparameter optimization, and comprehensive evaluation. The methodology leverages SVM's inherent strengths in handling high-dimensional feature spaces while implementing sophisticated scaling techniques that preserve binary feature integrity and class weighting mechanisms to compensate for distributional inequalities across the twelve customer inquiry categories.

The dataset undergoes careful 80/20 stratified train-test partitioning to ensure proportional class representation. Feature scaling applies z-score standardization exclusively to continuous variables while preserving binary features (containing only 0 and 1 values) to prevent categorical distortion. Class imbalance is addressed through inverse frequency weighting, where underrepresented categories receive proportionally higher weights during training: $w_i = \frac{\max(n_j)}{n_i}$, effectively amplifying minority class influence during margin optimization.

Hyperparameter optimization employs stratified 5-fold cross-validation across two kernel functions. Linear kernels are evaluated with regularization parameter $C \in \{0.1, 1, 10, 100\}$, while RBF kernels explore the same C range combined with gamma values $\{0.001, 0.01, 0.1, 1\}$. Evaluation uses macro-F1 score to ensure equal weighting across all classes regardless of frequency, providing meaningful performance assessment for imbalanced classification scenarios.

Cross-validation results demonstrate that RBF kernels with moderate regularization ($C = 10$) and low gamma values ($\gamma = 0.01$) achieve optimal performance, attaining mean macro-F1 scores of 0.876. This configuration indicates that moderately complex decision boundaries with smooth class transitions effectively capture the underlying data structure without overfitting.

Table 1: Top-performing SVM hyperparameter combinations from 5-fold cross-validation.

Kernel	**C**	**Gamma**	**CV Macro-F1**	**Std Dev**
RBF	10.0	0.01	0.876	0.023
RBF	100.0	0.01	0.876	0.021
Linear	0.1	—	0.857	0.028
RBF	10.0	0.001	0.839	0.031
Linear	1.0	—	0.839	0.025

The final optimized SVM model demonstrates substantial performance improvements over the baseline linear configuration. Test set evaluation reveals accuracy improvement from 80.0% to 82.0% and macro-F1 enhancement from 0.774 to 0.794, representing meaningful gains for automated customer service applications where misclassification costs are significant.

Table 2: Train–test comparison of SVM model performance.

Model Configuration	**Test Accuracy**	**Macro-F1**	**Improvement**
Baseline Linear SVM	0.800	0.774	—
Optimized RBF SVM	0.820	0.794	+0.020

Confusion matrix analysis reveals perfect classification for well-represented categories including “Auslandsbuchung”, “Gerätewechsel”, “Kreditkarte”, and “Phishing”, achieving 100% accuracy for these classes. However, persistent challenges remain for conceptually overlapping categories, particularly “Gebühren” and “Girokonto”, where 2 out of 3 “Gebühren” samples are misclassified as “Girokonto”, suggesting semantic similarity that challenges current feature representations.

Permutation importance analysis identifies the most influential features for classification decisions. Structural features like “all_caps_words” (importance: 0.038) capture urgency patterns, while banking-specific keywords including “investment_keywords” (0.030), “credit_keywords” (0.029), and “card_keywords” (0.026) provide domain-specific discrimination. Text embeddings, particularly “Embed_14” (0.034), demonstrate the value of semantic representation learning beyond explicit keyword matching.

The dominance of banking-specific keyword features in importance rankings validates the domain-specific feature engineering approach, demonstrating that explicit banking terminology detection provides more reliable classification signals than generic text features alone. The combination of structural features (capturing communication patterns), domain-specific keywords (banking terminology), and semantic embeddings (contextual relationships) creates a comprehensive feature representation that effectively distinguishes customer inquiry categories.

Cross-validation robustness analysis shows consistent performance across folds with macro-F1 scores ranging from 0.851 to 0.903 and low standard deviation (0.023), indicating stable generalization capabilities. The consistent superiority of RBF kernels over linear alternatives across all validation folds strengthens confidence in the non-linear decision boundary

Table 3: Top 10 most important features in the optimized SVM model.

Feature Name	**Importance Score**	**Feature Type**
all_caps_words	0.0381	Structural
Embed_14	0.0339	Text Embedding
investment_keywords	0.0297	Domain-Specific
credit_keywords	0.0288	Domain-Specific
card_keywords	0.0261	Domain-Specific
customer_service_keywords	0.0243	Domain-Specific
Embed_22	0.0234	Text Embedding
is_urgent	0.0221	Structural
online_banking_keywords	0.0219	Domain-Specific
payment_keywords	0.0198	Domain-Specific

approach for this classification task.

Despite encouraging performance improvements, limitations remain due to the constrained dataset size (249 samples across 12 categories) and persistent confusion between semantically related categories. The 82.0% accuracy and 0.794 macro-F1 performance represents solid achievement for academic purposes, though production deployment would benefit from expanded training data and enhanced feature engineering targeting problematic category pairs.

3 Model II - Random Forest (R)

The basic strategy for tuning the random forest is to divide the dataset into training and testing sets using a 60/40 split. On the training set, we perform cross-validation for the number of trees (`ntree`), node size (`nodesize`), and the number of features considered at each split (`mtry`).

We first cross-validate the number of trees (`ntree`) while holding `mtry` fixed at $\sqrt{m}$ and `nodesize` at 5. Once a stable value for `ntree` is selected, we jointly tune `mtry` and `nodesize`. This approach balances computational efficiency and effectiveness in identifying the parameter combination that maximizes performance.

To ensure robustness against random variation in data splits and model fitting, the entire cross-validation procedure is repeated across multiple random seeds. Performance metrics (Macro-F1 and Log Loss) are averaged over these seeds to obtain stable estimates and reduce the influence of any single random split. For reproducibility, we explicitly fix the random seed values used for both the data partitioning and the internal bootstrap sampling in the random forest training, ensuring that results can be exactly replicated.

Support for the number of trees is `ntree` $\in \{50, 100, 200, 300, 500, 1000\}$

After averaging over multiple seeds, `ntree = 1000` appears to yield the highest Macro-F1. However, the differences in results across 100–1000 trees are small relative to the standard deviations and are likely not statistically meaningful. Therefore, using 500 trees offers a good balance between performance and computational efficiency.

Next, we fix `ntree` at 500 and proceed to jointly cross-validate `nodesize` and the number

Table 4: Random Forest performance across `ntree` values (mean ± SD over seeds).

ntree	Macro-F1_Mean	Macro-F1_SD	LogLoss_Mean	LogLoss_SD
1000	0.814	0.0150	0.984	0.0101
500	0.809	0.0148	0.985	0.00964
300	0.808	0.0130	0.989	0.0142
200	0.801	0.0193	0.987	0.0141
100	0.800	0.0246	0.997	0.0183
50	0.793	0.0173	1.020	0.0669

of features with the following support: `mtry` $\in \{9, 16, 24, 40\}$ and `nodesize` $\in \{5, 7, 10\}$. From the results (Table 5), it is apparent that Log Loss is primarily influenced by the `mtry` parameter. For example, models with higher `mtry` values (e.g., `mtry` = 40) tend to achieve lower Log Loss, which aligns with the intuition that using more features allows the model to produce more confident estimates.

Among the top-performing combinations, the differences in Macro-F1 scores are negligible relative to their standard deviations. Therefore, we select the combination `mtry` = 40 and `nodesize` = 7, which yields one of the highest Macro-F1 scores and significantly lower Log Loss compared to the first two alternatives.

Table 5: Random Forest performance across `mtry` and `nodesize` combinations (mean ± SD over seeds).

mtry	nodesize	Macro-F1 Mean	Macro-F1 SD	LogLoss Mean	LogLoss SD
9	5	0.811	0.0131	0.941	0.00943
16	5	0.807	0.0131	0.861	0.0113
40	7	0.806	0.0225	0.806	0.0192
40	5	0.805	0.0149	0.798	0.0175
24	5	0.804	0.00938	0.827	0.0156
9	7	0.803	0.0188	0.953	0.0126
16	7	0.802	0.0113	0.871	0.0124
24	7	0.801	0.0171	0.832	0.0153
16	10	0.781	0.0218	0.897	0.0131
24	10	0.778	0.0160	0.859	0.0154
40	10	0.777	0.0209	0.829	0.0190
9	10	0.770	0.0178	0.985	0.0103

While high `mtry` values could typically increase the risk of overfitting, we believe this concern is mitigated in our case. The feature space is already regularized and consists of engineered, meaningful features. In addition, a `nodesize` of 7 produces moderately shallow trees, which helps to further reduce the risk of overfitting.

To evaluate the final performance of the model, we fit a random forest with the chosen parameters on the training set (60%) using 5-fold cross-validation, and then on the held-out test set (40%). We report Macro-F1, Weighted-F1, and Log-Loss, along with the train–test gap to assess overfitting. The confusion matrix from the held-out test set is used to generate a heat map for visual evaluation.

The final random forest model performed well on the training data, achieving reasonably strong Macro-F1 (0.774 ± 0.022) and Weighted-F1 (0.793 ± 0.021) scores, along with a low Log Loss (0.698 ± 0.062), indicating confident probability estimates. On the held-out test set, performance declined modestly in terms of Macro-F1 (gap: –0.024) and Weighted-F1 (–0.030), while Log Loss increased by 0.064, a change roughly equal to its training standard deviation. Overall, these results suggest only minor generalization loss, with no major overfitting concerns.

Table 6: Train–test comparison of final model performance.

Metric	**Train (mean ± SD)**	**Test**	**Δ (Test − Train)**
Macro-F1	0.774 ± 0.022	0.750	–0.024
Weighted-F1	0.793 ± 0.021	0.763	–0.030
Log-Loss	0.698 ± 0.062	0.762	+0.064

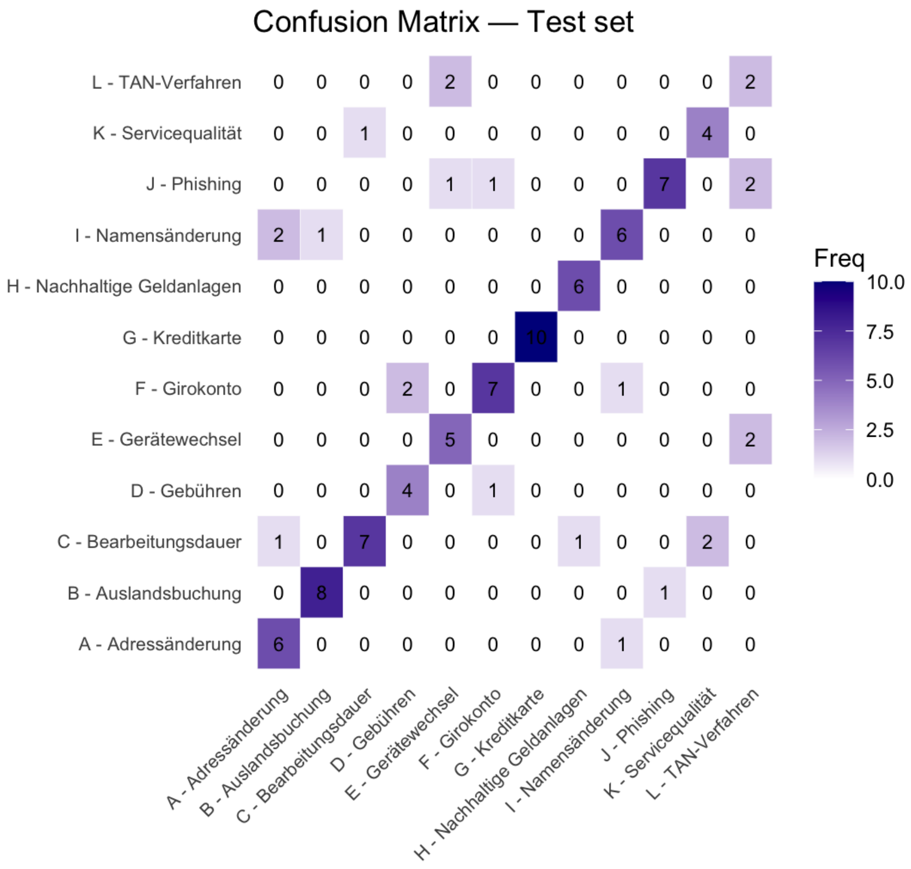


Figure 1: Heat map of confusion matrix for the tuned random forest model.

From the heat map, performance tends to be weaker for categories with small sample sizes, such as “L TAN-Verfahren” and “K Servicequalität,” which show frequent misclassifications. Stronger results are observed for “G Kreditkarte” and “B Auslandsbuchung,” where the number of observations is larger. However, the pattern is not entirely consistent: “D Gebühren” and “K Servicequalität” perform well despite having few samples, whereas “C Bearbeitungsdauer” and “F Girokonto” have a notable number of misclassifications. Overall, a larger dataset would likely improve performance, particularly for underrepresented and more error-prone classes.

4 Model III - Ensemble (Py)

The ensemble approach for the email classification task employs a sophisticated stacking architecture designed to address class imbalance and maximize performance across all categories. The model divides the dataset into training, validation, and test sets using a 65/15/20 split respectively. All results presented are based on a fixed random seed of 42 to ensure reproducibility.

The core strategy centers on a weighted stacking ensemble that combines nine diverse base learners, each optimized based on their contribution to the overall predictive performance. The architecture prioritizes models that demonstrated superior performance in preliminary analysis: K-Nearest Neighbors (KNN), Naive Bayes, and Logistic Regression form the foundation, while gradient boosting methods (XGBoost, LightGBM, CatBoost) and tree-based ensembles (Random Forest, Extra Trees) provide complementary strengths. A novel SVM ensemble utilizing both linear and RBF kernels adds additional diversity to capture different decision boundaries.

To address the severe class imbalance (with several categories containing fewer than 20 samples) the model implements a multi-pronged approach. First, advanced feature engineering creates class-specific signals targeting problematic categories. For instance, “processing_signal” features are engineered for the “Bearbeitungsdauer” class (baseline F1: 0.333), while “quality_signal” features target “Servicequalität” (baseline F1: 0.500). These domain-informed features help the model better distinguish underrepresented classes.

The feature selection process employs intelligent filtering that preserves critical domain features while using mutual information and chi-squared statistics to select the most informative remaining features. This hybrid approach ensures that engineered class-specific signals are retained while maintaining a compact feature space of 20 dimensions to prevent overfitting.

Sample weighting plays a crucial role in the training process. The model applies balanced class weights as a baseline, then further boosts weights for problematic classes based on their target improvement thresholds. Additionally, targeted SMOTE (Synthetic Minority Oversampling Technique) using BorderlineSMOTE is applied selectively to the most challenging classes, increasing their representation to the median class size while avoiding excessive synthetic sample generation.

The stacking architecture uses 3-fold stratified cross-validation to generate meta-features, with a regularized logistic regression serving as the meta-learner (C=1.5). This configuration balances model complexity with generalization capability. The model also attempts dynamic pseudo-labeling with confidence thresholds adjusted per class, lower for problematic classes

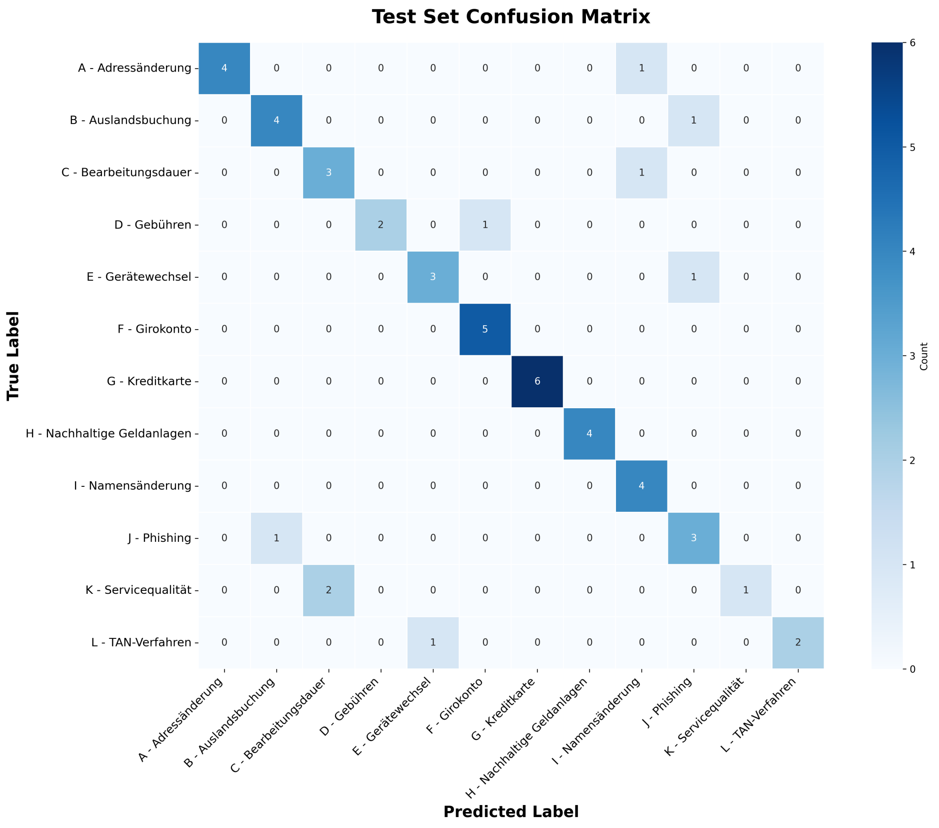
(0.83) and higher for well-represented classes (0.90), though in practice, the conservative thresholds often result in no additional pseudo-labels being generated.

Table 7: Final ensemble model performance across data splits (seed = 42).

Dataset	**Accuracy**	**Bal. Accuracy**	**Macro-F1**	**Weighted-F1**	**Cohen's κ**
Train	0.770	0.758	0.754	0.764	0.749
Validation	0.711	0.694	0.685	0.696	0.683
Test	0.820	0.793	0.798	0.816	0.802

The final model achieves strong performance on the test set with 82.0% accuracy and a macro-F1 score of 0.798, demonstrating effective handling of the multi-class imbalance. Notably, the model shows a *negative* train-test gap in macro-F1 (–0.045), indicating excellent generalization without overfitting, a critical achievement given the small dataset size.

Analysis of problematic classes reveals significant improvements: "Bearbeitungsdauer" improved from F1 0.333 to 0.667 (Δ+0.334), "Namensänderung" from 0.500 to 0.800 (Δ+0.300), and "Gerätewechsel" from 0.571 to 0.750 (Δ+0.179). Only "Servicequalität" remained unchanged at F1 0.500, suggesting this category requires additional domain-specific features or more training samples.



True \ Predicted	A	B	C	D	E	F	G	H	I	J	K	L
A - Adressänderung	4	0	0	0	0	0	0	0	1	0	0	0
B - Auslandsbuchung	0	4	0	0	0	0	0	0	0	1	0	0
C - Bearbeitungsdauer	0	0	3	0	0	0	0	0	1	0	0	0
D - Gebühren	0	0	0	2	0	1	0	0	0	0	0	0
E - Gerätewechsel	0	0	0	0	3	0	0	0	0	1	0	0
F - Girokonto	0	0	0	0	0	5	0	0	0	0	0	0
G - Kreditkarte	0	0	0	0	0	0	6	0	0	0	0	0
H - Nachhaltige Geldanlagen	0	0	0	0	0	0	0	4	0	0	0	0
I - Namensänderung	0	0	0	0	0	0	0	0	4	0	0	0
J - Phishing	0	1	0	0	0	0	0	0	0	3	0	0
K - Servicequalität	0	0	2	0	0	0	0	0	0	0	1	0
L - TAN-Verfahren	0	0	0	0	1	0	0	0	0	0	0	2

Figure 2: Heat map of confusion matrix for the tuned ensemble model.

The confusion matrix reveals that perfect classification (F1 = 1.000) is achieved for well-represented classes like “Kreditkarte” and “Nachhaltige Geldanlagen,” while smaller classes show more variability. The average prediction confidence of 0.623 indicates the model maintains appropriate uncertainty levels rather than overconfident predictions.

This ensemble approach successfully balances multiple objectives: maintaining high overall accuracy, improving performance on minority classes, and avoiding overfitting through careful regularization and feature selection. The negative train-test gap and strong Cohen’s kappa (0.802) validate the robustness of this multi-faceted strategy for imbalanced email classification.

Final Remarks

This project successfully demonstrates automated classification of German banking customer communications across 12 categories through comprehensive feature engineering and three distinct machine learning approaches. The text processing pipeline combined structural email features, TF-IDF representations, banking-specific keywords, and semantic embeddings to create a robust feature space addressing severe class imbalance challenges.

The Support Vector Machine achieved 82.0% accuracy and 0.794 macro-F1 using RBF kernels ($C = 10$, $\gamma = 0.01$), with perfect classification for well-represented categories but persistent confusion between semantically similar classes. The Random Forest attained 75.0% accuracy and 0.750 macro-F1 with minimal overfitting, though performance remained limited for minority classes. The stacking ensemble delivered the strongest results with 82.0% accuracy and 0.798 macro-F1, demonstrating excellent generalization through negative train-test gaps and significant improvements for problematic categories, including “Bearbeitungsdauer” (F1: 0.333$\rightarrow$0.667) and “Namensänderung” (F1: 0.500$\rightarrow$0.800).

Feature importance analysis consistently highlighted banking-specific keywords as dominant predictors across all models, validating the domain-informed engineering approach. The consistent 82% performance ceiling suggests current limitations stem primarily from insufficient training data, particularly for categories with fewer than 15 samples.

For production deployment, the ensemble approach offers the most robust solution through superior minority class handling and generalization capabilities. However, the 18% error rate necessitates human-in-the-loop mechanisms for critical categories like phishing detection. Future work should prioritize expanding the dataset and developing advanced feature engineering for semantic disambiguation between conceptually related categories.