MASTERARBEIT | MASTER'S THESIS

Titel | Title

AI-Enabled Marketing Optimization

verfasst von | submitted by

Rebeka Gosztom BSc

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien | Vienna, 2025

Studienkennzahl lt. Studienblatt | Degree programme code as it appears on the student record sheet:	UA 066 915
Studienrichtung lt. Studienblatt | Degree programme as it appears on the student record sheet:	Masterstudium Betriebswirtschaft
Betreut von | Supervisor:	Mag. Dr. Raimund Kovacevic

ABSTRACT (DE)

E-Mail-Marketing bleibt aufgrund seiner Effizienz, Skalierbarkeit und unmittelbaren Reichweite ein Grundpfeiler von Strategien zur Kundenbindung. Da den Verbraucher:innen immer häufiger mobile Technologien und digitale Plattformen zur Verfügung stehen – insbesondere in technikaffinen Regionen wie Tschechien – hat die Relevanz einer Optimierung der E-Mail-Marketing-Praxis zugenommen. Ein entscheidender Erfolgsfaktor ist dabei die Send Time Optimization (STO), also die Wahl des optimalen Versandzeitpunkts, um die Öffnungsrate zu maximieren.

Diese Arbeit untersucht, wie die künstliche Intelligenz Einstein die STO verbessert, anhand einer vergleichenden Fallstudie zweier E-Mail-Marketing-Kampagnen der Česká Spořitelna, einer großen tschechischen Bank und Mitglied der Erste Group, die im Dezember 2024 durchgeführt wurden. Kampagne A nutzte eine herkömmliche, von Menschen kuratierte STO, während Kampagne B die automatisierte Versandzeitoptimierung von Einstein AI einsetzte. Während Kampagne A eine eher bescheidene Öffnungsrate von 48,85 % erzielte, erreichte Kampagne B mit Einsteins optimiertem Versandzeitpunkt einen deutlich höheren Wert von 71,20 %. Der wesentliche Unterschied zwischen den Kampagnen bestand darin, dass Einsteins STO die E-Mails ausschließlich um 07:00 Uhr und 09:00 Uhr versandte.

Um die zugrunde liegenden Verhaltensfaktoren zu ermitteln, wurden verschiedene Machine-Learning-Modelle – darunter Logistische Regression, Random Forest, XGBoost, LightGBM, ein Neural-Network-Ensemble sowie ein weiteres Mischmodell – auf die Datensätze der beiden Kampagnen trainiert und getestet. Die Analyse konzentrierte sich auf den Einfluss von Kundendemografie (Geschlecht, Alter, Lebensphase – Verpflichtungen gegenüber Familie oder nicht), Kontokennzahlen (Saldo) und zeitlichen Merkmalen (Versandstunde, Wochentag). Bemerkenswert ist, dass sich die Modellgenauigkeit in Kampagne B deutlich verbesserte: Das Neural-Network erreichte 90,84 % und identifizierte das Geschlecht als wichtigsten Prädiktor bei der gegebenen STO. Ensemble- und Interaktions-Modelle zeigten zudem, dass Lebensphase, Kontosaldo und deren Wechselwirkungen mit dem Versandzeitpunkt maßgebliche Determinanten des E-Mail-Engagements sind.

Die KI-gestützte STO übertraf die manuellen Methoden nicht nur in dem reinen Engagement-Kennzahlen, sondern ermöglichte durch fortgeschrittene prädiktive Modellierung auch die Entdeckung nuancierter Verhaltensmuster. Die Ergebnisse unterstreichen das Potenzial, KI-Lösungen wie Einstein mit maßgeschneiderten statistischen Modellen zu kombinieren, um E-Mail-Marketingstrategien im Bankensektor und darüber hinaus weiter zu optimieren. Für die Kampagnen der Česká Spořitelna wird die fortgesetzte Nutzung von Einstein AI empfohlen; das auf

Neural-Networks basierende Modell zeigt jedoch zugleich, dass die Optimierung noch weiter vorangetrieben werden kann.

ABSTRACT (EN)

Email marketing remains a cornerstone of customer engagement strategies due to its efficiency, scalability, and immediate reach. As consumers increasingly have the availability of mobile technology and digital platforms – especially in tech-adept regions like Czechia – the relevance of optimizing email marketing practices has intensified. One critical aspect of success is Send Time Optimization (STO) – selecting the ideal time to dispatch marketing emails to maximize open rates.

This thesis explores how the Einstein artificial intelligence (AI) enhances STO through a comparative case study of two email marketing campaigns run by Česká Spořitelna, a major Czech bank and member of the Erste Group, that was conducted during December of 2024. Campaign A used a conventional, human-curated STO, while Campaign B employed Einstein AI for automated send-time optimization. While Campaign A had a modest open rate of 48.85%, Campaign B, with Einstein's optimized send time, achieved a significantly higher 71.20% open rate. The key difference between campaign A and B was that Einstein's STO had emails sent out only at 07:00 AM and 09:00 AM.

To uncover the underlying behavioural factors driving these outcomes, various machine learning models – including Logistic Regression, Random Forest, XGBoost, LightGBM, a NeuralNetwork ensemble and a further mixed model – were trained and tested on the two campaigns' datasets. The analysis focused on the influence of client demographics (gender, age, life phase (has familial obligations or not)), account metrics (balance), and temporal features (send hour, day of the week). Notably, model accuracy significantly improved in Campaign B, with the NeuralNetwork reaching 90.84%, identifying gender as the most influential predictor with the given STO. Ensemble and interaction-feature models further revealed that life phase, client balance, and their interactions with send timing were crucial determinants of email engagement.

The AI-driven STO did not only outperform manual methods in raw engagement metrics but also enabled the discovery of nuanced behavioural patterns through advanced predictive modelling. The findings highlight the potential of combining AI like Einstein with tailored statistical models to further optimize email marketing strategies in the banking sector or beyond. As for the campaigns done by Česká Spořitelna, the continued usage of the Einstein AI is recommended, however the NeuralNetwork trained model also reveals that the optimization can be furthered.

LIST OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

AI – Artificial Intelligence

DL – Deep Learning algorithm

ML – Machine Learning algorithm

STO - Send Time Optimization

1. INTRODUCTION

Email marketing is a cornerstone of digital advertisement, due to its cost-effectiveness, scalability, and direct engagement with existing and potential clients. However, by 2025 the ubiquity and sheer volume of marketing emails received daily by the general public poses a critical challenge – how to ensure that a given email is opened and the content is acted upon (Sabbagh, 2025). Among the strategies developed to address this issue, in the late 2010s the idea of Send Time Optimization (STO) has gained prominence. STO refers to the process of selecting the optimal time to deliver an email so that the recipient is most likely to engage with it (Patil 2024).

Initially STO has been curated by human touch (marketing teams), through manual input of probability by pattern, A/B testing, rule-based segmentation, etc., but with the rise of Artificial Intelligence (AI) and Machine Learning (ML) in the 2020s, the approach to STO is changing rapidly. AI-powered STO systems promise dynamic adaptation to user behavior patterns and make individualized predictions at scale, so that appropriate adjustions. This shift raises important questions about the effectiveness, efficiency, and reliability of AI-driven tools in comparison to human-curated methods, especially within high-stakes industries like banking (Patil 2024).

The present research investigates this intersection of AI and email marketing with a focus on Česká Spořitelna, a leading financial institution in the Czech Republic. By analyzing real-world campaign data and evaluating different STO strategies, this study seeks to explore the empirical benefits and limitations of AI in this domain.

1.1. Research Questions, Aims and Relevance

The overarching aim of this thesis is to evaluate the performance and potential of AI-driven Send Time Optimization in the context of email marketing, through practical testing of a real-world scenario – through the email campaigns run by the Czech bank Česká Spořitelna, of which one used human-curated STO, the other used SalesForce’s Einstein AI for STO.

The study is guided by the following research questions:

- ***Can Einstein AI’s STO performance be further improved using ML or DL to adapt to more nuanced feature-interaction to predict open rate?*** – to answer this, assessment must be made whether custom-built ML and DL models, based on client behavior data, can outperform, or enhance the decision-making process of existing AI tools.

- ***Would the human-curated STO reach the level of success the Einstein STO had with specific ML or DL models?*** – this question can be answered by an A/B testing, investigating the internal consistency and reliability of current either STO methods, and whether tool-based experimentation reveals systemic biases or advantages that can further enhance it.

The practical relevance of these questions is not only interesting to marketing specialists, but also to data scientists, automation specialists and AI developers designing next-generation communication tools. Should using ML or DL models to further improve upon Eintein's STO could lead to higher engagement rates, better ROI, and more personalized client experiences – especially in regulated sectors where trust and timing are crucial – improving the performance of the subject bank Česká Spořitelna.

1.2. Research Method, Scope and Literature

The study will employ a quantitative, data-driven approach using logistic regression and Machine Learning (ML) classification models to predict the likelihood of email opens. The case study dataset used originates from the Czech bank Česká Spořitelna and their real marketing campaigns, which provides detailed information that includes both recipient demographic variables and campaign parameters such as send time, sent day, open state, etc.

The scope of the study is limited to "warm" email campaigns, aka the emails are sent to existing clients who have previously opted in with the bank. The analysis compares STO outcomes from the Einstein AI-generated timing decisions versus an original human-curated strategy, and subsequently custom ML models on these empiric data will be trained to assess possible improvements.

The literature grounding this research spans several core areas:

- Academic and industry literature on email marketing strategies and its important metrics to give foundation for the marketing side-approach.
- Research on Send Time Optimization and temporal engagement behavior to see contemporary views on the topic.
- Studies on AI and machine learning applications in email marketing and STO, to seek out possibly useable ML or DL models for improvement.
- Research methodological frameworks in logistic regression and ensemble models relevant to predictive modeling.

2. LITERATURE

2.1. Email Marketing

Email marketing emerged with the development and spread of personal digital computing units (personal computers, PCs) and the internet itself in the 1980s USA. However, the very first email marketing campaign was done through the predecessor of the internet, through the Advanced Research Projects Agency Network (ARPAnet) of the United States Defense Department in the USA, which was the equivalent (and basis) of the modern internet, just limited to corporate use at the time. In 1978 Gary Thuerk, a marketing manager of the now defunct computer hardware developer company Digital Equipment Corporation (DEC), in order to boost the sales of his company, sent out invitations to a product demonstration of the then relatively cutting-edge DECSYSTEM-2020, 2020T, 2060 and 2060T computers to 397 corporate email accounts within the ARPAnet. This idea was radical and even dangerous, as the ARPAnet was intended to be used only for official (legal) reasons mostly, however the reception of the idea was good, and resulted in 13 million USD in sales for the DEC. Since then, Gary Thuerk is regarded as the "Father of Email Marketing" and the "Father of Spam", and this campaign has entered the Guinness Book of Records as the "World's Oldest Spam" (Pavlidou, 2025).

After the stunt of Thuerk, email marketing was rapidly adopted and had become a popular and reliable marketing tool in all types of commerce. By the later decades of the 20th century, as the now ubiquitous public internet got developed and PCs became more and more relevant on household user level, email marketing enabled companies to reach clients of all levels. Now, in the 21st century, with the invention and widespread daily use of internet-supporting mobile phone devices and eventually the smartphones, email marketing is at the height of its relevance, however by 2025 it is a bit played out, and its perception drastically changed over time, not in small thanks to disingenuous marketing spams, impersonations, and fake emails with malicious intent (Sabbagh, 2025).

2.1.1. Types of email marketing

The term "email marketing" refers to marketing engagement via electronic mail systems with potential or engaged beforehand customers. Multitude of email content types can be considered "marketing email", with different aims and results, but ultimately with the end goal of inspiring the recipient to engage with the sender company's goods or services (Rabab'ah, 2024).

Marketing emails can be categorized into two major groups:

1. Cold emails – sent to unengaged recipients, who have not previously interacted with the sender company. The email addresses for these types of engagement are often obtained via third parties or purchased private / public databases. Their main goal is outreach, lead generation, and brand exposure. The following marketing email types belong to this category:

- **Prospecting emails** – introduces the business and recommends their services, usually accompanied by first-time engagement benefits.
- **Lead generation emails** – sent to purchased or scraped lists, typically offering value upfront (e.g., free trial, free eBook, discount on goods/services, etc.). Can be referred to as a more desperate prospecting email.
- **Partnership outreach** – B2B-oriented, aims communication about potential partnerships and collaborations.
- **Event invitation to non-customers** – usually randomizes targets, inviting potential customers to some kind of event held by the sender company.

(Lutfil 2021; Kaminyar 2023; Le Plaisir 2024a; Le Plaisir 2024b)

2. Warm emails – emails sent to known, engaged users (typically continuous customers or users who have willingly shared their information with the company). These emails focus on retention, upselling, loyalty programs and relationship-building. The following can be categorized as warm-type:

- **Promotional email** – email that promotes the company, a specific product, service, or event, aimed at generating sales or driving specific actions (like visiting a website or redeeming an offer).
- **Transactional email** – triggered by user actions and typically contains important information rather than marketing content, but usually these also include subtle upselling or cross-selling opportunities.
- **Lifecycle (or "drip" / re-engagement) emails** – a sequence of emails based on user behavior over time, designed to guide customers, like onboarding guides, cart abandonment follow-ups (in case of online marketplaces) and re-engagement campaigns.
- **Newsletter** – many companies that have some kind of user email-subscription utilize a newsletter system, periodically sending automated emails which contain highlights of discounts, product launches and positive events and programs done by the company (like charity).

- **Survey/feedback emails** – post-purchase or service use feedback, which usually also contains either benefit for completion or recommendation on similar or "*Others also bought/used…*" type of further recommendations of products or services.

(Sabbagh 2021; Rabab'ah et al. 2024; Gul et al. 2024; Ravichandran and Kumar 2024)

2.1.2. Pros and Cons of Email Marketing

The advantages of email marketing are:

- **Ease of use and low cost –** through emails, clients are easy to reach directly, and even if the engagement is not always successful, usually at least puts the company on the radar of the client. By 2025, there are many ways for a company to make emails with mass appeal or personalized content, through developed softwares specifically for email marketing or broader PR/sales systems.
- **Customer targeting the correct way –** while traditional methods of advertisement (posters, TV advertisement, board signs, newspaper/magazine) or even other modern digital methods like social media or streamer/content creator advertisement requires generalized approach for mass appeal, while marketing emails allow tailored content through the known metrics of the target recipients, like age, gender, etc., to have the most possible engagement.
- **The ability to monitor and analyze success rate** – modern marketing programs track emails and provide immediate feedback of success by open rates and visits of provided links, which makes the reason for successful engagement trackable.

(Sabbagh 2021; Ravichandran and Kumar, 2024).

The disadvantages of email marketing are:

- **Cold emails rarely work** – due to ubiquity of scams, random marketing emails are rarely engaged.
- **Excessive use of non-legal databases –** third parties often sell databases illegally, resulting in the probability of potential legal consequences of using them.
- **Deception** – sometimes companies require to use underhanded deceptive tactics to generate engagement, which results in dissatisfaction afterwards.

(Tambe and Solanki 2022; Kaminyar 2023; Torossian, 2025)

2.1.3. Key Recipient properties in Email Marketing

Email marketing is not instant success – it has to be curated and tailored to the target demographic. This concept hangs on the understanding of key properties, aka "metrics" or "features" of the engaged customers. The only four universally basic metrics are the key demographic data: age, gender, geographic location and language preference – because, roughly putting it, a 50-year-old woman in Germany is interested in different things than a 20 years old man in Spain, and neither of them would understand a Russian language marketing email. The consideration for these four metrics is vital, however there are other beneficial information, that if available, can contribute to the success of an email marketing campaign:

- Client age grouping:
    - Life phase – unemployed, student, working, retiree
    - Life phase category – has familial obligations or not
- Socio-economic profile:
    - Income bracket
    - Occupation/employment status
    - Education level
- Device and technology usage:
    - Preferred device (mobile or desktop)
    - Email client (Gmail, Yahoo, etc.)
- Email engagement behaviour:
    - Open rate history
    - Click-through rate (CTR, how willing is the client to interact with the link in the sent email)
    - General time spent on reading emails
    - Preferred interaction time (more likely to open in the morning, around noon, afternoon, evening, night, etc.)
- Client-life-cycle stage:
    - New customer
    - Returning customer
    - Inactive/lapsed customer
    - Loyal customer
    - VIP customer

(Sabbagh 2021; Deshmukh 2024)

But aside from this, there are also industry-specific metrics that should be considered when developing marketing emails. In the case of banking:

- Financial profile:
    - Account balance
    - Transaction activity
    - Card ownership
    - Credit status (loan/mortgage)
    - Client tenure (regular or VIP)
- Service interaction frequency
    - Banking app usage
    - Physical location visits
    - Past responses to marketing emails

(Kalmár et al. 2015; Gupta and Rana 2017; Al-Ababneh 2024)

Tailoring personalized marketing emails in accordance with these metrics per available data makes for the most efficient marketing campaigns. Of course, it is often unrealistic to have all of this data, and also, there is yet to be an efficiently holistic marketing email system that could handle all of these variables to find all possible pattern of preference and behavior. However, one key metric stands above all else – preferred interaction time.

2.2. Send Time Optimization (STO)

In the 2010s a pattern had been discovered by the study of the then-counted as novelty advanced email-tracking systems, which showed that the most important factor of the success of email marketing is the proper send time (Patil 2024). This developed the idea of Send Time Optimization (STO), which aims to exploit the preferred interaction time of the recipients by timing the sending (and estimated arrival) of the marketing email and categorizing clients per their interaction pattern. Later, in the 2020s this approach was enhanced with Machine Learning (ML) and Deep Learning (DL) technology that enabled the creation of statistical algorithms based on limited inputs to predict client interactions with marketing emails (Patil 2024).

To establish relationships between variables for predictions, generally regression statistical models are used for STO. This statistically estimates the relationships between a dependent variable (often called the "label" in ML/DL) and one or more error-free independent variables (often called "regressor" or "predictor") (Bagui et al. 2021).

As seen on Figure 1, Araujo et al. in 2022 proposed an ensemble ML model based on the Random Forest statistical model with adding further Multiple Linear and K-neighbor model features, and also a Recurrent Artificial Neural Network ML+DL model. The model based on 9 other attempts to create a holistic STO ML or ML+DL model, with Araujo et al identifying key features that they believed that if viewed together would create the perfect STO model. The database used to train the model contained the following features:

- Open rate (total open rate of sent emails)
- Click-through rate (aside from opening, how many interacted with the link provided)
- Time interval (time between receiving and opening the email)
- Email content (this was analyzed through Natural Language Learner Artificial Intelligence (NLLAI) system, which can analyze and interpret textual input)
- Recipient profile information (unspecified)

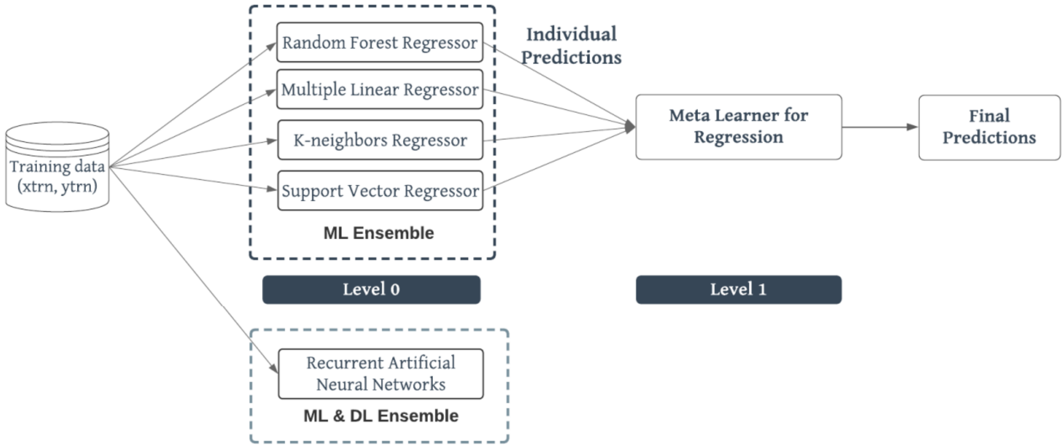

Figure 1: Diagram of the stacking proposed approach for the ML ensemble and in addition to the DL algorithm by Araujo et al (Araujo et al. 2022).

The ML+DL ensemble resulted in a measly 27.1% accuracy rate predicting, while the ML ensemble's individual results of the trained models in the Level 0 varied vastly, which resulted in poor performance in the Level 1, but still vastly outperformed the ML+DL approach, which supports the idea of a stacking ML model is more likely to be the ideal STO model than a ML+DL one.

Similarly, Potrimba (2022) also came to the same conclusion in their own research – using the "A/B testing" method, where two group are examined with the same tools, one group is controlled and another one is varied, Potrimba also concluded that DL models distort data due to the inability to inability to treat individual cases of not following the pattern as unique cases and not influencer on the overall data. Using Linear Regression, Random Forest, Bayesian Linear Regression, Least Angle

Regression and Neural Network models, once again a stacked ML model deemed to be the most optimal tool.

In a study on attempting to improve STO with ML models, researchers utilized A/B testing with an email campaign done by the Swedish furniture and home-design company IKEA in Sweden, where campaign A didn’t utilize any STO, while campaign B utilized human-curated STO. For their research, they utilized a Logistic regression, Random Forest, and Extreme Gradient Boost (XGBoost) enhanced Neural Network ML model, to create predictive learners. The XGBoost enables weak learner models to have a better understanding of irrationally patterned databases (which within email marketing campaigns happen due to clients having lives which alter their behaviour), which resulted in the highest accuracy rate within their research, at 50% accuracy rate (Abbas and Al-Jailawi 2024).

Searching for higher accuracy rate models, the Light Gradient Boost Machine (LightGBM) model has yielded an accuracy rate of 86% in the research of Hafeez et al in October of 2024. The researchers analyzed the possibility of recognizing effectively behavioural patterns with interaction metrics and creating personalized marketing emails for the successful Pakistani e-commerce business Daraz, by utilizing decision-tree models that can effectively recognize patterns in a highly varied dataset with erratic patterns – Naive Bayes, Latent Dirichlet Allocation (LAD) and Light Gradient Boosting Machine (LightGBM). While both the Naive Bayes and LAD models didn’t reach any significant level of accuracy, Hafeez et al reported a staggering 86% accuracy rating of the LightGBM, suggesting the model may be worthwhile in the study and research of optimization (Hafeez et al 2024).

3. ČESKÁ SPOŘITELNA

3.1. History of Česká Spořitelna

The Česká Spořitelna (translates to: Czech Savings) was founded as "Böhmische Sparkasse" (translates to: Bohemian Savings/Czech Savings) in 1825 by the entry of Austrian financial institutions into the Czech market, as with the dissolution of the Holy Roman Empire in 1806, the Kingdom of Bohemia (the predecessor state of the modern Czech Republic) was incorporated into the Austrian Empire (Irozhlas 2020).

The idea of savings in a bank was a still a relative novelty concept for the region at the time, and the type of service offered by the Böhmische Sparkasse itself originated from Germany in the late 18th century. Initially the Böhmische Sparkasse was only accessible to the nobility as the set minimum fund to open an account was the GDP of multiple villages combined, but by the mid-19th century, as the bank grew, it gradually allowed commoners to create accounts, though still only available at lowest for high-middle class citizens. Real inclusivity began in the beginning of the 20th century, as there was an investor panic from the early 1910s as the possibility of war in the surrounding area increased, and by the First World War the bank almost went under would've it not had its rural branches see improvement by the nobility attempting to save their fortune (CAS, 2025).

After the dissolution of the Austro-Hungarian Empire, the newly formed Czechoslovak Republic (1918-1938) in 1918 had introduced changes to the banking system – only saving banks insured by territorial governing units (municipalities or districts) were allowed to exist, and federal savings banks, such as the Böhmische Sparkasse, were banned. However, due to their historical popularity and public outcry, the Böhmische Sparkasse and the První Moravská Spořitelna banks were allowed to remain in operation (Hruška and Dvořáková 2015).

After the hyperinflation in Central Europe during the first half of the 1920s, the stabilized economy in the second half of 1920s brough rapid increase in foreign interest. This resulted in a banking bubble, which bursted in 1931, now known as the "European banking crisis of 1931", which brought down with it the complete banking sector of Germany, Austria, Hungary, Poland, Romania and to a lesser extent Czechoslovakia. During this time the non-insured banks like the Böhmische Sparkasse lost quite a bit of its clientele, but once again after the stabilization in the following years it the bank's popularity of services increased beyond its previous state (Macher 2018).

In 1938, the Sudeten German minority in western Czechoslovakia (with the support of the neighboring Nazi Germany (1933-1945)) pressured the parliament to cede the German-native

Sudetenland to Germany. Following this the remaining Czechoslovakia was named the Second Czechoslovakian Republic (1938-1939), but in 1939 further geopolitical plays by Nazi Germany dissolved Czechoslovakia, leaving the Czech side as a partially annexed drone country for Nazi Germany called the "Protectorate of Bohemia and Moravia" (1939-1945). During this time, economically the Protectorate remained in control, however the Czech banking system was demanded to adapt the new German banking model (illiberal, state controlled). This meant a connected network between financial institutions, which led to multiple banks merging under a single brand, and the Böhmische Sparkasse together with the three other largest banks in Prague, together forming the Prager Sparkasse ("Prague Savings") (Tvrdá 2017).

After the end of World War II and the fall of Nazi Germany, the Czechs and Slovaks once again united under the same flag, forming the Third Czechoslovak Republic (1945-1948). The original plans for the economy were a liberal market system, however the financial leadership of the country was infiltrated by socialists, which led to even further strengthening the state control over the institutions, culminating with the coup d'état of 1948 by the Czechoslovakian Communist Party, the success of which resulted in the Fourth Czechoslovak Republic (1948-1989), a satellite state of the Soviet Union. In the communist era, financial institutions were centralized, merging all of the institutions under a singular brand, differentiated by purpose (Soukup and Žídek, 2021). During 1948-1967 these were:

- District capital cities: Okresní Spořitelny a Záložny (District Savings and Credit Unions, OSAZ);
- Other cities: Spořitelny a Záložny (Savings and Credit Unions, SAZ);
- Villages: Záložny-Kampeličky (Credit Unions, ZK).

(Soukup and Žídek, 2021)

In accordance with this the Prager Sparkasse transformed and its branches became part of the financial scheme of the communist Czechoslovakia. Nothing major happened until 1967, where all saving institutions were merged into a single institution called Státná Spořitelna (State Savings). In 1969 the Státná Spořitelna split intwo two cells – the Česká Státní Spořitelna (Czech State Savings, ČSTSP) and the Slovenská Státna Sporiteľňa (Slovak State Savings). This formula remained until in 1989 when the State Bank of Czechoslovakia transformed the previously single-tier banking system of the soviet formula into a two-tier banking system where investment and commercial banks were formed (Soukup and Žídek, 2021).

Less the two weeks after this change came the Velvet Revolution, which marked the end of the communist rule over Czechoslovakia. The country changed its name to the Czech and Slovak

Federative Republic (1989-1992), however after two years the country dissolved into the Czech Republic and the Slovak Republic respectively (Soukup and Žídek, 2021).

On February 1, 1992, the Česká Státní Spořitelna had changed its legal form and became a commercial bank with its old-new name of Česká Spořitelna. Initially the state still owned a majority of shares of the company, however in 2000 the Erste group bought 52% of the Česká Spořitelna, then by the 2010s the Erste Group held 98% of the company (Česká Spořitelna, 2018).

3.2. Česká Spořitelna's economic overview

Financial Position Indicators

CZK million	2024	2023	2022	2021	2020
Total assets	2,030,076	1,797,820	1,639,938	1,641,741	1,537,780
Loans and advances to banks, net[2]	443,389	304,007	266,675	364,994	424,838
Loans and advances to customers, net	1,090,958	1,010,592	913,847	836,949	757,324
Securities[3]	426,968	408,678	378,216	358,466	277,646
Deposits from banks	275,456	122,287	113,541	49,695	91,335
Deposits from customers[4]	1,483,174	1,366,038	1,256,795	1,184,543	1,100,450
Equity attributable to owners of the parent	154,144	143,955	137,232	142,744	149,125

Income Statement Indicators

CZK million	2024	2023	2022	2021	2020
Net interest income	39,811	34,583	36,719	31,083	29,099
Net fee and commission income	12,778	10,894	9,504	9,186	8,243
Operating income	56,486	48,365	49,875	42,354	40,147
Operating expenses	(24,304)	(23,144)	(21,335)	(20,398)	(19,110)
Operating result	32,182	25,221	28,540	21,956	21,037
Net profit attributable to owners of the parent	26,162	18,614	20,161	14,181	10,006

Basic Ratios

	2024	2023	2022	2021	2020
ROE	18.4%	13.9%	15.2%	9.8%	7.0%
ROA	1.3%	1.0%	1.1%	0.8%	0.6%
ROA (Česká spořitelna, a.s. only)	1.3%	1.0%	1.2%	0.8%	0.6%
Cost/Income ratio	43.0%	47.9%	42.8%	48.2%	47.6%
Non-interest operating income/ operating income	29.5%	28.5%	26.4%	26.6%	27.5%
Net interest margin on interest-bearing assets	2.0%	1.9%	2.1%	2.0%	1.9%
Loan/Deposit ratio	73.6%	74.0%	72.7%	70.7%	68.8%
Capital adequacy (unconsolidated)	18.8%	18.8%	19.3%	23.0%	24.7%

Figure 2: Outtake from the "Annual Report 2024" by Česká Spořitelna (Česká Spořitelna 2025).

Even today Česká Spořitelna is the largest bank and the de facto leader of commercial banking of the Czech scene, by the end of 2024 counting 4,63 million customers of the 10 million total population of the Czech Republic, with more than 1200 ATMs operated and 9,600 personnel employed in the country. It is also worth mentioning, that as of 2025, Česká Spořitelna is 200 years old, despite its various changes (Česká Spořitelna, 2025).

Looking at the current financial performance of the bank at Figure 2, 2024 marks the highest yielding year of its recent memory, and it's easy to see why – one of the lowest operating costs with all-around highest income numbers across the board. As seen on Figure 3, unlike other institutes who still hasn't recovered from the Covid-19 pandemic and the Russo-Ukrainian War's effect on the EU's economy, Česká Spořitelna stood strong during the trying times, and now still dominates the Czech banking market.

Financial Position Indicators

CZK million	2020	2019	2018	2017	2016
Total assets	1,537,780	1,458,650	1,426,465	1,329,223	1,066,526
Loans and advances to banks, net [3]	424,838	411,650	389,844	157,525	22,328
Loans and advances to customers, net	757,324	729,206	694,065	638,694	577,453
Securities [4]	277,646	242,301	246,781	219,501	257,183
Deposits from banks	91,335	292,111	318,861	295,232	114,282
Deposits from customers [5]	1,096,666	993,257	952,506	881,997	786,876
Equity attributable to owners of the parent	149,125	137,128	122,473	120,810	121,564

Income Statement Indicators

CZK million	2020	2019	2018	2017	2016
Net interest income	29,099	30,261	27,821	25,350	25,512
Net fee and commission income	8,243	8,591	8,540	8,803	9,308
Operating income	40,147	41,899	39,088	37,227	38,227
Operating expenses	(19,110)	(19,352)	(18,327)	(18,240)	(18,146)
Operating result	21,037	22,547	20,761	18,987	20,081
Net profit attributable to owners of the parent	10,006	17,743	15,362	14,610	15,457

Basic Ratios

	2020	2019	2018	2017	2016
ROE	7.0%	14.5%	13.2%	12.4%	13.2%
ROA	0.6%	1.2%	1.1%	1.1%	1.5%
Cost/income ratio	47.6%	46.2%	46.9%	49.0%	47.5%
Non-interest income/operating income	27.5%	27.8%	28.8%	31.9%	33.3%
Net interest margin on interest-bearing assets	1.9 %	2.1%	2.2%	2.6%	3.1%
Loan/deposit ratio	68.6%	72.9%	72.7%	72.2%	73.2%
Capital adequacy (unconsolidated)	24.7%	21.8%	19.0%	18.6%	20.1%

Figure 3: Outtake from the "Annual Report 2020" by Česká Spořitelna (Česká Spořitelna 2021).

3.3. Email Marketing by Česká Spořitelna (Internal Source)

Information for the following section was obtained by firsthand interaction with the marketing team of Česká Spořitelna as its employee.

As part of the modern economical ecosystem, Česká Spořitelna also does email marketing. Recent review of the January of 2024 email campaign by Česká Spořitelna found that the key reasons users do not interact with the bank's emails is due to its being considered spam, but also due to lack of active subscription to the bank's newsletter, perhaps irrelevant content, frequent emails from the same sender, relative inability to unsubscribe and dissatisfaction with services.

Table 1 shows the general email performance metrics for Česká Spořitelna, highlighting the effectiveness of their previous email marketing campaigns.

Email metric	Data Česká Spořitelna
Delivery/Bounce Rate	0,36%
Spam Rate	< 0,02%
Open Rate	37%
Click Through Rate (CTR)	1,9%
Click to Open Rate	5,14%
Unsubscribe Rate	0,03%
Conversion Rate	N/A

Table 1: Performance of the January of 2024 email campaign by Česká Spořitelna, without the Einstein AI (Source: Information provided directly by Česká Spořitelna).

The low bounce rate of 0.36% reflects a well-maintained recipient list, indicating that the majority of emails are successfully delivered to recipients. This is further supported by a spam rate of less than 0.02%.

A high open rate of 37% suggests that Česká Spořitelna's "Subject" lines and preheaders are effective in capturing the attention of their audience. However, the click-through rate (CTR) of 1.9% reveals a low level of interaction, suggesting that there is room for improvement in driving deeper engagement, and that the 94,86% of those who opened the email did not click the provided content-link..

To optimize email delivery, Česká Spořitelna's domain, info.csas.cz, is integrated with SalesForce's Feedback Loop list, enabling monitoring of messages classified as "Delivery" or "Bulk" and tracking campaigns flagged as spam by recipients. The graph shows the analysis of delivery rates during the first quarter of 2024 highlighted key trends, including the total delivery of 8 million messages, of which only 2 million reached recipients' primary inboxes, while 7 million were categorized as bulk mail and 402,000 flagged as spam.

Figure 4 illustrates important patterns in recipient responses to emails during the observed period. A total of 3 million recipients, represented by the blue line, responded to the emails. Approximately 88,000 recipients, shown by the green line, clicked on links within the emails. Meanwhile, 8,000 messages were automatically flagged as spam, and 1,000 emails, represented by the yellow line, were manually marked as spam. These insights provide valuable guidance for refining email strategies to enhance engagement and minimize negative interactions.

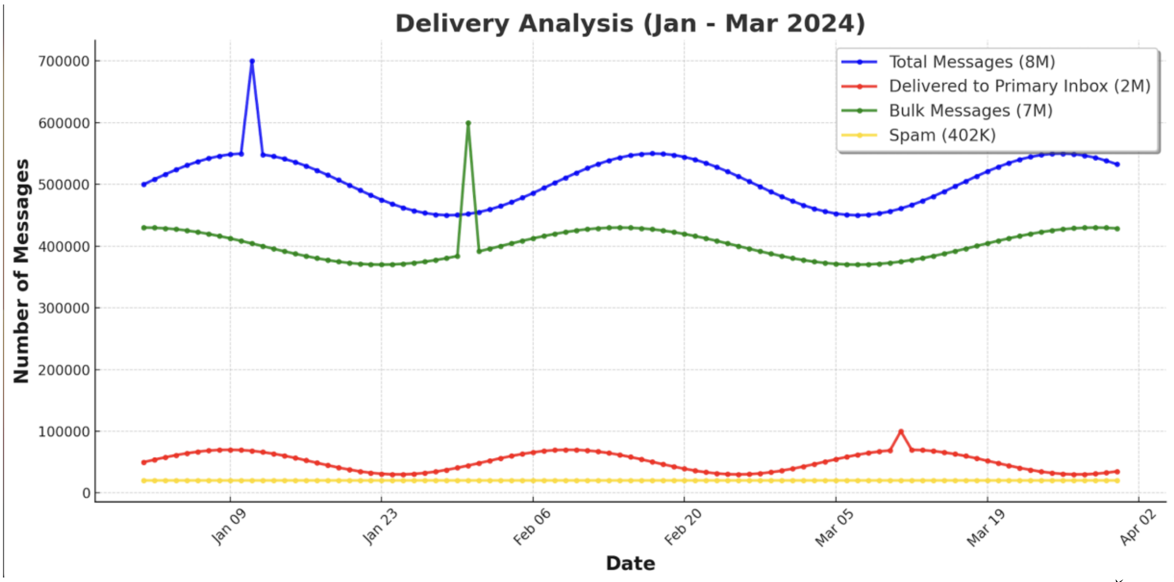

Figure 4: Received emails' categorizations f the January of 2024 email campaign by Česká Spořitelna (Source: Information provided directly by Česká Spořitelna).

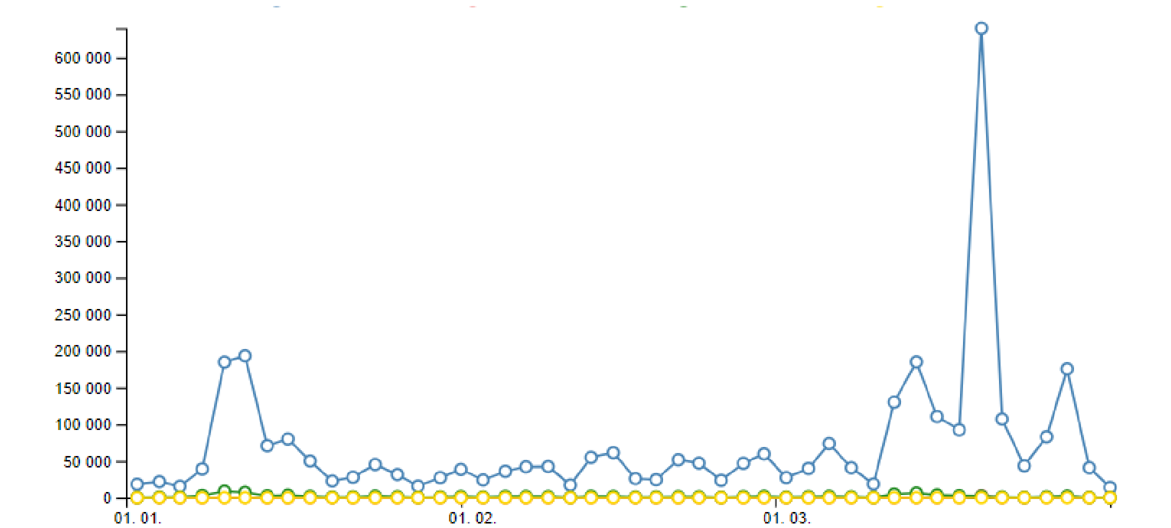

Figure 5: Open time after delivery of the January of 2024 email campaign by Česká Spořitelna (Source: Information provided directly by Česká Spořitelna).

As shown on Figure 5, within the January of 2024 campaign, most recipients opened their email 3 days after it has been sent. When recipients open their inbox, the first elements they notice are the sender's name and logo, which play a critical role in establishing trust and encouraging interaction. Subject lines, typically ranging between 41 and 70 characters, are crafted to quickly capture attention, while preheaders, usually between 85 and 100 characters, provide contextual information to influence open rates.

The study categorized recipient reading behaviors into four types:

- **skimming readers** glance through the emails without detailed reading;
- **minimal readers** focus only on the first sentence.
- **engaged readers** selectively read headlines, bold CTAs, images, and bullet lists,
- **full readers** scroll through and engage with the entire email.

These varying behaviors highlight the importance of structuring content to appeal to all audience segments.

3.4. Einstein's STO implementation by Česká Spořitelna

Salesforce Marketing Cloud's Einstein is a market leading example of how an ML-based AI transforms marketing strategies. Adopted in the second half of 2024, by integrating ML, predictive analytics, and automation, Einstein optimizes email marketing campaigns and other customer engagement efforts. Its features are designed to maximize engagement, improve campaign performance, and provide actionable insights into customer behavior, making it perfect for the research.

One of Einstein's standout features is its STO automation. Using historical engagement data and advanced ML algorithms to determine the best time to send emails to individual recipients, Einstein identified intricate patterns in customer behavior, such as the likelihood of an email being opened at specific times. By integrating Einstein's STO into the email campaign, Česká Spořitelna may ensure that messages are delivered at the most optimal moments, significantly increasing the likelihood of recipient interaction.

Česká Spořitelna tested Einstein's abilities in an email campaign conducted in December of 2024. One campaign, titled "Campaign A", used human-curated STO based on the marketing team's on-hand analysis and predictions, while a second campaign, titled "Campaign B", was done using Einstein's STO. Einstein's STO had the major distinction that it only sent emails at 07:00 AM and 09:00 AM respectively.

4. METHODOLOGY

The thesis centers on evaluating the two email marketing campaigns by Česká Spořitelna using key metrics (features). By analyzing open rates in accordance with these features (sent day, sent time, gender, age, account balance), the aim is to uncover relevant patterns that inform about what metrics to keep in mind with the general target audience to determine how to further improve upon Einstein's STO.

4.1. Research Design

In the case of Česká Spořitelna's campaigns, A/B testing is an appropriate way to explore which are the key features which may lead to higher email engagement rate, staying purely at an open rate research, due to email content's inclusion would require a much larger scale investigation that may involve the necessity of tools that beyond the scope of this humble study. This approach is used to compare traditional methods with a more advanced version that employed Einstein AI for STO. Both campaigns were designed to be as similar as possible – targeting the same goals, audience, and messages – to ensure that any differences in outcomes could be attributed to the use of AI. It should be mentioned, however, that Einstein's STO allowed for reaching more clients, therefore Campaign B works with a larger dataset.

A simple statistical overview, based on simple comparisons, would not provide sufficient data on the matter. Following a quick comparison of the two campaigns, aka what did Einstein do differently, the main research is done by employing trained ML statistical models and algorithms to determine whether the STOs employed can be improved further.

4.2. Statistical models

Using the features of the provided campaigns (introduced later), based on the research of the previously discussed Araujo et al. 2022, Potrimba 2022, Abbas and Al-Jailawi 2024 and Hafeez et al 2024 a variety of ML models will be used to try and improve upon the STOs of both the human-curated and Einstein's STOs.

4.2.1. Simple trained Logistic Regression model

First, a simple trained Logistic Regression model is a useful starting point because it provides a clear and interpretable way to assess how different variables influence the likelihood of an email being opened. Furthermore, the Logistic Regression model serves as a strong baseline due to its simplicity and efficiency. It allows for quick experimentation and validation without requiring extensive computational resources. Since it assumes a linear relationship between predictor variables and log-odds of the outcome, it also helps detect whether certain features show a strong positive or negative correlation with email opens. Even if it does not provide the highest accuracy, its outputs lay the groundwork for deeper investigation, offering a fundamental understanding of how key factors contribute to open rates.

The model outputs a table of coefficients and associated statistics for each feature:

- **Coefficient** – indicates the direction and strength of the relationship with the target (positive means it increases the likelihood of opening).
- **Std. Error** – measures uncertainty in the coefficient estimates.
- **Z value** and **Pr(>|z|)** – used to test whether the coefficient is significantly different from zero (i.e., whether the variable matters).
- **Odds** – an easier-to-interpret version of the coefficient correlation-interpretation, that shows how the odds of opening the email change with each feature.

To assess model performance:

- The dataset was split into **training** (70%) and **testing** (30%) portions.
- Predictions were made on the test set using the trained model.
- Key evaluation metrics included:
    - **Accuracy** – the percentage of correctly predicted email openings.
    - **Confusion Matrix** – shows how many emails were correctly and incorrectly predicted as opened or not opened.
    - **Classification Report** – includes *precision*, *recall* (true positive rate), and *F1 score* for more nuanced performance insights.

This model estimates the probability that a given customer opens the email based on their profile. An intercept term was added to the model to account for the baseline likelihood of opening an email when all other variables are at their average level. Using the *statsmodels* library in Python, the model was trained to find the best-fitting relationship between the input features and the likelihood of an email being opened.

4.2.2. Logistic Regression Pipeline model

Building on the initial insights from the basic Logistic Regression model, a more structured approach with a Logistic Regression Pipeline model is a reasonable follow-up because it allows for better feature engineering and preprocessing, ensuring that the data is optimized for prediction. The pipeline-based Logistic Regression model uses pre-processing the data for easier pattern-recognition.

This model uses *scikit-learn*'s ***Pipeline*** and ***ColumnTransformer***, allowing simultaneous treatment of numerical and categorical features. Numerical inputs (such as age, balance, and day of the week) were standardized using ***StandardScaler***, which adjusts values to a common scale. Categorical inputs (such as gender, life obligations, and time of day) were transformed using ***OneHotEncoder***, which converts categories into binary columns for model consumption. The model was then trained within this pipeline using the transformed data. This model remains interpretable and efficient, making it suitable for initial deployment or as a comparative baseline. It estimates the probability of a customer opening an email based on their profile and campaign timing.

To evaluate performance:

- The data was randomly split into 80% **training** and 20% **testing**, with class distribution preserved.
- Predictions were made on the test set after model training.
- Metrics included accuracy, a confusion matrix, and a classification report (*precision*, *recall*, *F1-score*), which together offer a well-rounded view of predictive power.
- The pipeline design also facilitates easy tuning or replacement of components (like testing different classifiers or encoders), making it scalable for future development.

4.2.3. Random Forest (Hyperparameter) model

A Random Forest model with hyperparameter optimization was implemented to attempt to enhance prediction accuracy and capture complex interactions in the data. Unlike simpler linear models, Random Forests are ensemble-based (multiple learners incorporated), non-parametric classifiers (used in case of unknown density function and used to estimate the probability density function) that build multiple decision trees and aggregate their results to improve robustness and reduce overfitting.

This model is particularly well-suited for handling data that exhibits non-linear relationships and mixed data types. It does not require assumptions about feature distribution or linearity, making it more flexible than logistic regression. Additionally, feature engineering was applied prior to modeling by introducing interaction terms to capture cross-variable effects that may influence email opening behavior.

Key steps include:

- **Feature Scaling** – features are normalized using ***StandardScaler*** to stabilize variance and assist in interaction term formation.
- **Hyperparameter Tuning** – a randomized grid search (***RandomizedSearchCV***) explores various combinations of model parameters (number of trees, tree depth, minimum split size, leaf size, and class weights). This allows optimization of the model's generalization performance on unseen data.
- **Training and Evaluation –** after splitting the data (80% **training** / 20% **testing**, stratified), the best model is selected based on 5-fold cross-validated accuracy.

The RF model's outputs include:

- **Accuracy –** a direct measure of correct predictions on the test set.
- **Confusion Matrix** and **Classification Report** – provides detailed insights into true positives, false negatives, precision, recall, and F1 score.
- **Feature Importance** table – offers a ranked list of variables based on how much they contributed to prediction decisions, making the model interpretable despite its complexity.

4.2.4. Extreme Gradient Boost model (XGBoost)

XGBoost is a high-performance, gradient-boosted decision tree algorithm renowned for its speed, scalability, and predictive accuracy, particularly in structured/tabular data environments. According to literature, it improves upon traditional ensemble methods (like Random Forests) by applying boosting (adding fitting complementer data to increase the visibility of patterns) strategies that sequentially correct predecessor errors, thereby refining predictions.

Key characteristics include:

- **Interaction features** – the dataset was enhanced with cross-variable terms to capture synergistic effects, such as: [Account balance × Client's age]; [Client's gender × Sent day]; [Account balance × Sent day]. These variables allow the model to explore how combinations of user attributes influence email engagement beyond isolated effects.
- **Scaling** – numeric features were normalized using StandardScaler prior to model training. While tree-based models like XGBoost aren't sensitive to scaling, this step improves model generalizability and harmonizes with prior models for comparison consistency.
- Training Procedure:
  - data was split into **training** (70%) and **testing** (30%) subsets using stratification to preserve class balance.

    - the model was trained using **logistic loss** for binary classification (***binary:logistic objective***).
    - ***eval_metric="logloss"*** was used to monitor training convergence and performance.
    - ***use_label_encoder=False*** was specified to maintain compatibility with the current XGBoost package version.
- Model Evaluation:
    - predictions on the test set were evaluated using accuracy, confusion matrix, and a full classification report (*precision*, *recall*, *F1-score*).
    - feature importance scores were extracted to assess which predictors contributed most to model decisions.

Distinct (potential) advantages of the XGBoost:

- **Boosting power** – XGBoost's gradient boosting mechanism sequentially learns from its errors, typically resulting in higher accuracy than standalone models.
- **Regularization** – implicit L1/L2 regularization helps prevent overfitting, making it suitable for complex datasets with potentially redundant features.
- **Speed and efficiency** – optimized training routines, parallelization, and cache awareness make XGBoost significantly faster and more memory-efficient than traditional boosting libraries.
- **Robustness to multicollinearity** – unlike linear models, XGBoost is tolerant of correlated features, benefiting from the inclusion of multiple engineered interaction terms.

This model is a strong candidate for the campaigns as predictive precision is critical and variable interactions are expected to play a key role. Its *feature importance* insights further support strategic decision-making, particularly in identifying which customer traits or send conditions influence open rates most strongly.

4.2.5. LightGBM model

A LightGBM (Light Gradient Boosting Machine) classifier was implemented to model the likelihood of email openings. LightGBM is a high-performance gradient boosting framework developed by Microsoft that is particularly efficient with large datasets and high-dimensional features. It uses histogram-based algorithms for fast training and lower memory usage while maintaining strong predictive accuracy.

Just like with the XGBoost model, the LightGBM model also has predefined interactions to help uncover nonlinear and compounded influences on email open behavior. Also, similarly, has categorical variables that were encoded as integers to be compatible with LightGBM's native handling of numeric inputs, and the included features were standardized using **StandardScaler** to maintain consistency across models and ensure convergence stability, even though LightGBM is not sensitive to scaling.

The LighGBM model also features **hyperparameter tuning**, a grid search with 5-fold cross-validation was used to identify optimal model hyperparameters, tuning the best estimator was selected based on cross-validated accuracy performance:

- ***n_estimators*** (number of boosting rounds),
- ***max_depth*** (maximum depth of trees),
- ***learning_rate*** (step size shrinkage).

Model's predictions are evaluated using accuracy, confusion matrix, and a full classification report. A heatmap visualization of the confusion matrix is generated to clearly show misclassification patterns.

In LightGBM a SHAP (*Shapley Additive Explanations*) system is computed to interpret model predictions at both global and individual levels. SHAP explains the prediction of a data sample by calculating the contribution of each feature to the prediction of the algorithm.

A SHAP summary plot was produced, providing insight into which features drive the model's decisions most frequently and strongly. This enhances transparency and supports explainable AI practices, which are crucial in data-driven marketing decisions. Standard LightGBM feature importances were also calculated and saved, offering a straightforward ranking of predictive variables by their average gain contribution.

Distinct (potential) advantages of LightGBM:

- **Speed and efficiency** – LightGBM's histogram-based learning and leaf-wise tree growth strategy significantly accelerate training and reduce memory consumption compared to other gradient boosting implementations.
- **Accurate for sparse and large datasets** – it handles missing values and sparse data efficiently, making it ideal for real-world marketing datasets.
- **Interpretability via SHAP** – the integration of SHAP adds a layer of model transparency, enabling data analysts and marketers to understand why the model predicted what it did.
- **Fine-Tuned performance** – hyperparameter optimization via GridSearchCV ensures the model is tailored to this dataset's structure and class distribution.

4.2.6. Enhanced ensemble NeuralNetwork model

Should the machine learning (ML) models fail, the NeuralNetwork DL model is incorporated into a stacking ensemble (multiple learners ensembled) framework to predict the likelihood of email opens. The neural network utilizes a **Multi-Layer Perceptron** (**MLPClassifier**), which is a supervised learning algorithm that learns a function by training on a dataset and optimizing through back-propagation – input and output mapping. This architecture is particularly useful in capturing complex nonlinear relationships in high-dimensional feature spaces. Just like in previous models, interaction terms were engineered – such as combinations of balance with client age and send day – to expose compound effects on the probability of email engagement. Furthermore, polynomial interactions were introduced explicitly (for example, the data squared) using ***PolynomialFeatures***, ensuring the neural network could learn from structured, high-order interactions.

To maintain a consistent preprocessing approach, categorical variables like gender, day segment, and life phase were encoded as integers. Even though neural networks do not require feature scaling to the same extent as other models, standardization using StandardScaler was applied to enhance training stability and ensure convergence across all base learners in the ensemble. Rather than relying solely on a neural model, a stacking ensemble was designed, combining predictions from three powerful base learners:

- Random Forest classifier;
- An XGBoost classifier;
- The Neural Network (MLPClassifier);

The final predictions were made by a secondary Random Forest model, trained on the outputs of the base learners. This ensemble strategy should allow each model to compensate for the weaknesses of others, improving overall robustness and performance.

Hyperparameters and structure:

- The neural network was configured with:
- Hidden layers: (128, 64, 32) for depth and flexibility,
- Activation function: ReLU (Rectified Linear Unit) to introduce non-linearity,
- Solver: Adam optimizer for efficient gradient-based learning,
- Max iterations: 1000, ensuring sufficient learning epochs.

To address class imbalance and improve generalizability, SMOTE (Synthetic Minority Oversampling Technique) was applied to the training data, synthesizing new examples of the minority class (opened emails) to balance the dataset.

Predictions were evaluated using standard metrics: accuracy, confusion matrix, and classification report (precision, recall, F1-score).

To interpret feature influence across the full – ensemble including the neural network – permutation importance was computed, which technique shuffles each feature to observe its effect on prediction accuracy, offering a model-agnostic method to understand variable contributions.

This hybrid design should balance predictive strength, resilience, and interpretability – permutation importance key for marketing analytics where actionable insights are as crucial as model performance:

- **Deep learning capacity** – the MLP can capture complex, nonlinear relationships that simpler models might miss, especially when enhanced with engineered interaction terms.
- **Resilient ensemble structure** – The stacking approach aggregates strengths from diverse model types (trees, boosting, and neural nets), enhancing prediction accuracy and reliability.
- **Balanced learning** – SMOTE ensures the model does not bias toward the majority class, allowing better sensitivity to rare but important events like email opens.
- **Interpretable outputs** – Although neural networks are traditionally opaque, permutation importance adds transparency by ranking the influence of input variables on the outcome.

4.2.7. Mixed model

And lastly, due to the enhanced ensemble NeuralNetwork showing promise, a more nuanced ML+DL model is made enhance predictive performance and resilience in classifying email open behavior, further elements of Random Forest and XGBoost classifiers were incorporated into a stacking ensemble. The dataset was enriched with several interaction terms to better express latent relationships among variables. These included cross-interactions of client balance and age, balance and send day, and categorical-numeric mixes like gender by send day. These interactions were designed to surface complex behavioral trends that simpler linear models would likely miss.

In addition to manual interactions, polynomial feature interactions were generated via PolynomialFeatures(interaction_only=True) between balance_mean and Client_Age. This strategy allowed tree-based models to leverage high-order combinatorial relationships without assuming any specific functional form. A stacking ensemble approach was adopted, integrating:

- A Random Forest Classifier with 200 estimators, emphasizing robustness and interpretability through ensemble bagging.
- An XGBoost Classifier, tuned with a learning rate of 0.2 and maximum tree depth of 8, capable of handling subtle patterns in sparse and imbalanced data.

- A Neural Network (MLPClassifier) with deep architecture to capture highly nonlinear signal patterns.

The final predictions were made using a secondary Random Forest trained on the probabilistic outputs of the three base models. This architecture ensures that predictions benefit from the combined strengths of all model types.

To unify data across the ensemble, SMOTE (Synthetic Minority Over-sampling Technique) was applied to balance the class distribution in the training set. This was crucial for improving recall on the minority class (email opens). All numerical features were standardized using StandardScaler to prevent scale variance from biasing models, especially the neural network.

Hyperparameters of the mixed models are:

- Random Forest:
  - n_estimators: 200
  - random_state: 42
  - Default Gini criterion and no max depth specified, allowing flexible tree growth.
- XGBoost:
  - learning_rate: 0.2
  - max_depth: 8
  - eval_metric: 'logloss'
  - Random seed for reproducibility.
- Meta-learner (Final Estimator):
  - Another Random Forest model, selected for its ability to aggregate outputs while preserving generalization.

Model performance measured using:

- Accuracy score on test data.
- Confusion matrix heatmaps, providing insights into false positives and negatives.
- Classification report, summarizing precision, recall, and F1-score per class.

To interpret the model's predictions, permutation feature importance was employed. This technique estimates the contribution of each feature by measuring the decrease in model performance when that feature's values are randomly shuffled. Despite tree-based models being naturally interpretable, permutation importance adds a unified, model-agnostic lens across the ensemble.

Distinct Advantages:

- **Robust learning** – random Forest's averaging and XGBoost's boosting strategies offer complementary strengths in bias-variance tradeoff.
- **Interaction awareness** – manual and polynomial interaction terms reveal behavior patterns across demographics, time, and financial status.
- **Balanced classification** – through SMOTE, minority class performance is improved, reducing model bias and capturing rarer engagement behaviors.
- **Unified interpretability** – even in a complex ensemble, feature importance remains accessible through permutation-based ranking, enhancing business trust in model outputs.

This dual-tree ensemble, backed by deep learning and a robust preprocessing pipeline, delivers a high-performing and interpretable system ideal for data-rich environments like campaign analytics, where both accuracy and actionability are key.

5. CASE STUDY OF ČESKÁ SPOŘITELNA'S EMAIL MARKETING CAMPAIGN

As explained before, Česká Spořitelna did two distinct campaigns – campaign "A", that was used without Einstein's STO, and Campaign B that used Einstein's STO. For the purpose of this research, the content of the emails is not analyzed and does not present a relevant feature, only the STO counts which is what prompts opening emails, not the content itself.

5.1. Campaign A: STO application by Česká Spořitelna without Einstein AI

Campaign A used human-curated STO by the Česká Spořitelna's marketing team that was based upon advanced pre-AI marketing analytical tools. The campaign lasted from 9th of December 2024 to the 22nd of December 2024, reaching 1,005 recipients.

5.1.1. Campaign A metrics and statistics

The performance of the human-curated STO shows better performance than the campaign of January of 2024 did, with the open rate being 48.85% across 1,003 accountable recipients (Table 2).

Email metrics	Note	Calculation	Result
Send total	Number of emails sent	-	1,005
Sends	Number of emails with accountable feedback	-	1,003
Opens	Number of emails opened	-	490
Open Rate	Open rate	$\frac{Opens}{Sends}$	48.85%

Table 2: Campaign A's performance metrics (Source: own calculations based on information provided by Česká Spořitelna).

Gender	Sends	% of total sends	Opens	Open Rate in %
Male	489	48.75%	245	50,10%
Female	514	51.25%	245	47,67%

Table 3: Campaign A's gender-based performance metrics (Source: own calculations based on information provided by Česká Spořitelna).

As gender-based statistics (Table 3) show that while more female clients were engaged, an equal number of recipients opened the email, which means male recipients were more likely to open the email.

Age group-based statistics reveal that the bulk of recipients were above the age of 45 (727 out of 1.003). Meanwhile, what is also interesting to see, is that the 60+ years old age group was the second most inclined to open amongst all the age groups, with 51.08% open rate within the age group. Less surprisingly the first place is that of the 18-25 year old category, but the engagement rate in the 36-45 years old group is low, with only 44.23% (Table 4).

Age Group	Sends	Opens	Open Rate in %
18-25	41	26	63.41%
26-35	79	36	45.57%
36-45	156	69	44.23%
46-60	310	146	47.10%
60+	417	213	51.08%

Table 4: Campaign A's age group-based performance metrics (Source: own calculations based on information provided by Česká Spořitelna).

Česká Spořitelna keeps track of its clients life conduct situation (Table 5). While the approach may not be holistic enough, as some of these categories are not mutually exclusive, it definitely reveals interesting data – within Campaign A, while students are the most likely to open the emails, retired and middle aged clients are the second and third most likely life phased individuals to open emails sent by Česká Spořitelna, respectively.

Life phase	Sends	Opens	Open Rate in %
1 - Student	11	7	63.64%
2 - Youth, living with parents	37	17	45.95%
3 - Single	79	38	48.10%
4 - Parent	160	67	41.88%
5 - Middle Aged	394	195	49.49%
6 - Retired	322	166	51.55%

Table 5: Campaign A's life phase-based performance metrics (Source: own calculations based on information provided by Česká Spořitelna).

Lastly, the feature "Account Balance Category" has also been calculated from the raw numbers provided by Česká Spořitelna of its clients. This metric may also serve as a key interest point in the determination of good STO strategies. As seen on Table 6, the results are highly varied within the engagement rate as sorted by account balance category, but it is highly hilarious to see that in general, at least with the human-curated STO, people with debt in Campaign A are the least likely to open emails sent by the bank (36.36%).

Account balance category	Sends	Opens	Open Rate in %
Negative account balance (debt)	11	4	36.36%
0 - 1,000	2	1	50.00%
1,000 - 2,500	5	3	60.00%
2,500 - 5,000	27	14	51.85%
5,000 - 7,500	34	17	50.00%
7,500 - 10,000	50	24	48.00%
10,000 - 15,000	91	46	50.55%
15,000 - 20,000	100	44	44.00%
20,000 - 30,000	136	69	50.74%
30,000 - 40,000	81	36	44.44%
40,000 - 50,000	75	43	57.33%
50,000 - 75,000	101	46	45.54%
75,000 - 100,000	69	34	49.28%
100,000+	221	109	49.32%

Table 6: Campaign A's account balance categorization-based performance metrics (Source: own calculations based on information provided by Česká Spořitelna).

5.1.2. Simple Logistic Regression model used on Campaign A

As seen on Table 7 the simple Logistic Regression model suggests:

- "Sent weekday" feature has a positive coefficient of 0.114, but this effect was not statistically significant (p = 0.202).
- Similarly, emails sent before noon showed a small potential improvement in open rates, with a coefficient of 0.102, yet this effect was also not significant (p = 0.501).
- Gender had a negative coefficient of -0.129, suggesting that the model considered male recipients to be slightly less likely to open the email, with an estimated 12% reduction in

odds, however, once again the p-value of 0.318 indicates a lack of statistical support for this conclusion.

- Age also showed to be a more notable effect, with a coefficient of 0.130, meaning older clients may be more likely to open emails, though this was still not significant (p = 0.124).
- The “Life phase” variable, comparing clients without a family to those with one, had a negative coefficient of -0.095 and an odds ratio of 0.909. This would imply a slight decrease in open rate among those with family responsibilities, but the p-value (0.701) shows the result is far from meaningful.
- And lastly, “Account balance category” had an almost negligible effect (coefficient = 0.006, odds ratio = 1.006) with an extremely high p-value of 0.926.

LOGISTIC REGRESSION MODEL					
Feature	**Coefficient**	**Std. Error**	**Z value**	**Pr (>|z|)**	**Odds**
Sent weekday	0.114	0.089	1.277	0.202	1.120
Send day time (before noon / afternoon)	0.102	0.151	0.673	0.501	1.107
Gender	-0.129	0.129	-0.998	0.318	0.879
Client’s age	0.130	0.085	1.536	0.124	1.139
Life phase (categorized as without (1-3) or with family (4-6))	-0.095	0.248	-0.384	0.701	0.909
Account balance category	0.006	0.068	0.093	0.926	1.006

Table 7: Campaign A’s trained simple Logistic Regression model’s feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

Model accuracy	49.83%

Table 8: Campaign A’s trained simple Logistic Regression model’s predictive accuracy (Source: own calculations based on information provided by Česká Spořitelna).

Confusion matrix prediction in a 301 unit sample (Table 9 and Table 10):

	Predicted not OPEN	Predicted OPEN
Actual not OPEN	70	79
Actual OPEN	72	80

Table 9: Campaign A's trained simple Logistic Regression model's predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna).

The simple Logistic Regression model trained on campaign A had the accuracy rate of predicting OPEN state using its training and projecting onto the actual campaign A database, has resulted in an accuracy rate of only 49.83% (Table 8). This number shows that the linear logistics-interpreting regression model is incapable of accurately grasping the behavioral pattern of the clients, therefore it cannot contribute to enhancing campaign A's STO.

	Precision	Recall	F1-score	Support
Not OPEN	0.49296	0.46980	0.48110	149
OPEN	0.50314	0.52632	0.51447	152
Macro average	0.49805	0.49806	0.49778	-
Weighted average	0.49810	0.49834	0.49795	-

Table 10: Campaign A's trained simple Logistic Regression model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

Overall, it is obvious the model does not find compelling evidence that any single feature used is a strong predictor of whether an email will be opened or not. However, the simple Logistic Regression model can be improved upon by introducing the Pipeline processing feature, which uses a different interpretation logic during processing, changing certain assumptions and biases used by the simple Logistic Regression without a dedicated processing module.

5.1.3. Pipeline Logistic Regression model used on Campaign A

As seen on Table 11, the Pipeline Logistic Regression model has determined the following:

- Gender is the most significant predictor, with a coefficient of 1.628 and a p-value of approximately 0.00000006, meaning the odds of an email being opened are about 5.09 more likely for men compared to women.
- Account balance category also has a significant impact, the coefficient of 1.483 and a p-value near 0.00007 indicate that higher account balances are associated with increased email

open rates. This supports the idea that financially better-off clients may be more receptive to communication with the bank.

- The pre/post 12:00 send time, while having a relatively high coefficient (1.035), did not reach statistical significance (p = 0.092).
- Other factors such as client age, life phase category (with or without family), and sent weekdays displayed no statistically significant relationship with open rates (p > 0.05).

LOGISTIC REGRESSION PIPELINE MODEL					
Factor	**Coefficient**	**Std. Error**	**Z value**	**Pr (>|z|)**	**Odds**
Pre 12:00 or post 12:00 sent time	1.035	0.614	1.691	0.092	2.820
Sent weekday	1.084	0.518	2.095	0.036	2.961
Life phase (categorized as without (1-3) or with family (4-6))	-0.405	-0.909	-0.452	0.656	0.679
Client's age	-0.952	1.829	-0.526	0.604	0.393
Account balance category	1.483	-0.374	3.978	0.000007	4.411
Gender	1.628	0.300	5.430	0.00000006	5.09

Table 11: Campaign A's trained Pipeline Logistic Regression model's feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

Model accuracy	54.23%

Table 12: Campaign A's trained Pipeline Logistic Regression model's predictive accuracy (Source: own calculations based on information provided by Česká Spořitelna).

	Predicted not OPEN	**Predicted OPEN**
Actual not OPEN	59	41
Actual OPEN	51	50

Table 13: Campaign A's trained Pipeline Logistic Regression model's predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna).

	Precision	Recall	F1-score	Support
Not OPEN	0.53636	0.59000	0.56190	100
OPEN	0.54945	0.49505	0.52083	101
Macro average	0.54291	0.54252	0.54137	-
Weighted average	0.54294	0.54229	0.54127	-

Table 14: Campaign A's trained Pipeline Logistic Regression model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

The Pipeline Logistic Regression model proved to be more accurate than the simple Logistic Regression with its predictions, with a prediction accuracy rate of 54.23% projected on the actual database (Table 12). The higher than 50% accuracy means that the model is fit to a degree to be used for STO enhancement. While the Pipeline model was intended to improve predictive performance of the Logistic Regression, the results suggest the logistic approach cannot handle the erratic data appropriately. The Pipeline processing module adjusted one, despite being significantly more accurate, showcases that the Logistic Regression model alone is not an appropriate tool for metric-relevancy diagnostics or STO evaluation in Campaign A.

5.1.4. Random Forest model used on Campaign A

RANDOM FOREST (HYPERPARAMETER) MODEL	
Factor / Interaction	**Importance**
Account balance category + Client age	34.47%
Client's age	31.43%
Balance (categorized)	14.52%
Sent weekday	7.08%
Gender + sent weekday	5.48%
Pre 12:00 or post 12:00 sent time	3.52%
Gender	2.31%
Life phase (categorized as without (1-3) or with family (4-6))	1.15%

Table 15: Campaign A's trained Random Forest model's feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

Following in the footsteps of the academic literature, the next model to be used is a hyperparameter-tuned Random Forest model to see if a tree-based interpreter can handle Campaign A's recipient's behavioral pattern.

Model accuracy	51.74%

Table 16: Campaign A's trained Random Forest model's predictive accuracy (Source: own calculations based on information provided by Česká Spořitelna).

Random Forest's Confusion matrix prediction in a 201 unit sample (Table 17 and Table 18):

	Predicted not OPEN	**Predicted OPEN**
Actual not OPEN	57	43
Actual OPEN	54	47

Table 17: Campaign A's trained Random Forest model's predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna).

	Precision	**Recall**	**F1-score**	**Support**
Not OPEN	0.51351	0.57555	0.54028	100
OPEN	0.52222	0.46534	0.49214	101
Macro average	0.51786	0.51767	0.51621	-
Weighted average	0.51788	0.51741	0.51609	-

Table 18: Campaign A's trained Random Forest model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

5.1.5. Simple learner XGBoost model used on Campaign A

EXTREME GRADIENT BOOST (XGBOOST) MODEL	
Factor / Interaction	**Importance**
Account balance category + Sent weekday	13.13%
Gender + Sent weekday	12.46%
Balance (categorized)	12.06%
Account balance category + Client age	11.92%
Gender	11.85%
Sent weekday	11.83%
Pre 12:00 or post 12:00 sent time	9.82%
Life phase (categorized as without (1-3) or with family (4-6))	9.67%

Table 19: Campaign A's trained standard-scaler XGBoost model's feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

The Random Forest model's accuracy falls short of that of the Logistic Regression Pipeline model, with only 51.74% accuracy (Table 16), however its identification of certain interactions as potentially highly relevant reveals that there are interactive factors present as key influencers, unlike what the simple linear influence suggested with only considering individual factors, however this requires deeper analysing, so the next model used is a standard-scaler model with Extreme Gradient Boost (XGBoost).

Model accuracy	48.83%

Table 20: Campaign A's trained standard-scaler XGBoost model's accuracy (Source: own calculations based on information provided by Česká Spořitelna).

XGBoost's Confusion matrix prediction in a 301 unit sample (Table 21 and Table 22):

	Predicted not OPEN	**Predicted OPEN**
Actual not OPEN	75	75
Actual OPEN	79	72

Table 21: Campaign A's trained standard-scaler XGBoost model's predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna).

	Precision	Recall	F1-score	Support
Not OPEN	0.48755	0.5	0.49342	150
OPEN	0.48979	0.47682	0.48322	151
Macro average	0.48840	0.48841	0.48832	-
Weighted average	0.48840	0.48837	0.48830	-

Table 22: Campaign A's trained standard-scaler XGBoost model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

5.1.6. LightGMB model used on Campaign A

LIGHTGBM MODEL	
Factor / Interaction	**Importance**
Account balance category + Client's age	36.56%
Client's age	28.90%
Account balance category + Sent weekday	12.41%
Sent weekday	6.789%
Gender + Sent weekday	5.04%
Account balance category	4.36%
Gender	3.10%
Life phase (categorized as without (1-3) or with family (4-6))	2.23%

Table 23: Campaign A's trained standard-scaler XGBoost model's feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

Given the XGBoost model has identified further interaction-possibilities as important, however still fell short on the accuracy rating with a measly 48.83% (Table 20), means that idea is possibly right, but the tools the model is working with are insufficient. Therefore in accordance with the literature, the next model to be used is the LightGBM model.

LightGBM's accuracy is higher than the XGBoost model with 50.24% (Table 24), however it is still weaker than the Random Forest model or the Pipeline model. This means the predetermined interaction is an incorrect approach, however given it identifies mostly all the same interactions and factors as most important, a pattern can be detected. To try a new approach, we switch to a DL model, the NeuralNetwork model.

Model accuracy	50,24%

Table 24: Campaign A's trained LightGBM model's accuracy (Source: own calculations based on information provided by Česká Spořitelna).

LightGBM's Confusion matrix prediction in a 201 unit sample (Table 25 and Table 26):

	Predicted not OPEN	Predicted OPEN
Actual not OPEN	49	51
Actual OPEN	49	52

Table 25: Campaign A's trained LightGBM model's predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna).

	Precision	Recall	F1-score	Support
Not OPEN	0.50000	0.49000	0.49495	100
OPEN	0.50485	0.51485	0.50980	101
Macro average	0.50243	0.50243	0.50238	-
Weighted average	0.50244	0.50249	0.50241	-

Table 26: Campaign A's trained LightGBM model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

NeuralNetwork has a significantly better accuracy than previous non-linear models, with an accuracy of 53.23%. This is the first model to also reveal negative effect, which suggests that the factor/interaction harms the model's performance when included or that random shuffling of the feature improves predictions, either due to sheer irrelevancy, overfitting or collinearity (which makes sense given older clients tend to have equally higher savings).

5.1.7. Simple Logistic Regression model used on Campaign A

NEURALNETWORK MODEL	
Factor / Interaction	**Importance**
Gender	41.57%
Account balance category	22.29%
Life phase (categorized as without (1-3) or with family (4-6))	19.28%
Gender + Sent weekday	16.27%
Sent weekday	10.24%
Client’s age	9.64%
Pre 12:00 or post 12:00 sent time	-0.60%
Account balance category + Sent weekday	-7.23%
Account balance category + Client’s age	-11.45%

Table 27: Campaign A’s trained NeuralNetwork model’s feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

Model accuracy	53.23%

Table 28: Campaign A’s trained NeuralNetwork model’s accuracy (Source: own calculations based on information provided by Česká Spořitelna).

NeuralNetwork's Confusion matrix prediction in a 201 unit sample (Table 29 and Table 30):

	Predicted not OPEN	**Predicted OPEN**
Actual not OPEN	57	43
Actual OPEN	51	50

Table 29: Campaign A's trained NeuralNetwork model's predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna).

	Precision	**Recall**	**F1-score**	**Support**
Not OPEN	0.52778	0.57000	0.54808	100
OPEN	0.53763	0.49505	0.51546	101
Macro average	0.53271	0.53252	0.53177	-
Weighted average	0.53273	0.53234	0.53169	-

Table 30: Campaign A's trained NeuralNetwork model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

We can also see that it had more simple factor importance-identification than the previous models, similarly to the Linear Regression Pipeline model. This reveals that perhaps the models post-Pipeline but prior to NeuralNetwork have overestimated the significance of certain interactions and factors. However, it is seems increasingly likely that singular features alone cannot provide satisfactory explanations of OPEN rate influences. Because of this, a more complex, stacked ML+DL approach was designed to incorporate the better features of the non-linear models – by combining the NeuralNetwork model with the Random Forest (Hyperparameter) and the Extreme Gradient Boost models.

5.1.8. Mixed model used on Campaign A

NEURALNETWORK + RANDOM FOREST (HYPERPARAMETER) + XGBOOST (ML+DL)	
Factor / Interaction	**Importance**
Life Phase	33.50%
Pre 12:00 or post 12:00 sent time	18.23%
Account balance category + Client's age	14.29%
Account balance category + Client's age (polynomial transformation)	13.30%
Sent weekday	10.84%
Account balance category (polynomial transformation)	9.36%
Life phase (categorized as without (1-3) or with family (4-6))	8.87%
Client's age (polynomial transformation)	7.39%
Gender	-0.99%
Gender + Sent weekday	-3.94%
Account balance category + Sent weekday	-10.84%

Table 31: Campaign A's trained ML+DL model's feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

The ML+DL hybrid model reveals a few key factors (Table 31). First, the persistence of negative importance effects make the suggest certain elements not just unusable, but actually harmful on the prediction, it must be pointed out, however, that there is no consensus throughout models which factors are these, as while the plain NeuralNetwork suggested that Gender is the most important factor regarding OPEN rates, the hybrid model concluded that its actually a negative factor. Second, the results with polynomial transformation suggest that certain results, given unique conditions (like for example squaring up) display more tangible logistical significance (Table 32). This means result interpretation may be beneficial to also examine with polynomial conditioning.

Based on statistical significance, the results of the logistic model give well founded conjectures, and possibly may be worth placing more effort into.

Model accuracy	52.24%

Table 32: Campaign A's trained ML+DL model's accuracy (Source: own calculations based on information provided by Česká Spořitelna).

Model's Confusion matrix prediction in a 201 unit sample (Table 33 and Table 34):

	Predicted not OPEN	**Predicted OPEN**
Actual not OPEN	45	55
Actual OPEN	41	60

Table 33: Campaign A's trained ML+DL model's predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna).

	Precision	**Recall**	**F1-score**	**Support**
Not OPEN	0.52326	0.45000	0.48387	100
OPEN	0.52174	0.59406	0.55556	101
Macro average	0.52250	0.52203	0.51971	-
Weighted average	0.52249	0.52239	0.51989	-

Table 34: Campaign A's trained ML+DL model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

5.2. Campaign B: Send Time Optimization by Česká Spořitelna with Einstein

Campaign B used Einstein's suggestion for its STO. To also stress-test Einstein's capabilities, campaign B conducted a larger, 3,371 recipient campaign during the 2nd of December 2024 – 27th of December 2024 period. In campaign B's case, Einstein's STO made it so the sent time of day was 7:00am and 9:00am exclusively.

5.2.1. Campaign B metrics and statistics

Out of the 3,371 recipients 3,059 had accountable feedback. Out of these, 2,178 recipients opened the emails, marking a staggering 71.20% open rate, which is by 22.35% better than it was with Campaign A (Table 35).

Email metrics	Note	Calculation	Result
Send total	Number of emails sent	-	3,371
Sends	Number of emails with accountable feedback	-	3,059
Opens	Number of emails opened	-	2,178
Open Rate	Open rate	$\frac{Opens}{Sends}$	71.20%

Table 35: Campaign B's performance metrics (Source: own calculations based on information provided by Česká Spořitelna).

The gender-based performance metric once again shows that male recipients are more likely to engage with the emails sent by the bank (Table 36).

Gender	Sends	Opens	Opening rate
Male	1555	1123	72,22%
Female	1504	1055	70,15%

Table 36: Campaign B's gender-based performance metrics (Source: own calculations based on information provided by Česká Spořitelna).

Age-group based performance (Table 37) shows greater variety in the results, however a major distinction is that with this STO the 26-35 age group now stands as the second most engaged group, meaning that with Einstein's 07:00 AM or 09:00 AM approach is appealing to these groups. Similarly, however, at first place stands the 60+ age group, while the 18-25 years old recipients are at third place.

Age Group	Sends	Opens	Open Rate in %
18-25	361	249	68.98%
26-35	581	424	72.98%
36-45	630	429	68.10%
46-60	832	563	67.67%
60+	655	513	78.32%

Table 37: Campaign B's age group-based performance metrics (Source: own calculations based on information provided by Česká Spořitelna).

Life phase-wise once again the Retired personnel are the most likely to engage with the emails sent by Česká Spořitelna with 78.44% thanks to Einstein's STO (Table 38).

Life phase	Sends	Opens	Open Rate in %
1 - Student	130	88	67.69%
2 - Youth, living with parents	195	152	77.95%
3 - Single	588	416	70.75%
4 - Parent	704	460	65.34%
5 - Middle Aged	992	709	71.47%
6 - Retired	450	353	78.44%

Table 38: Campaign B's life phase-based performance metrics (Source: own calculations based on information provided by Česká Spořitelna).

Account balance category	Sends	Opens	Open Rate in %
Negative account balance (debt)	95	67	70.53%
0 - 1,000	20	14	70.00%
1,000 - 2,500	32	20	62.50%
2,500 - 5,000	65	45	69.23%
5,000 - 7,500	110	71	64.55%
7,500 - 10,000	109	88	80.73%
10,000 - 15,000	218	161	73.85%
15,000 - 20,000	208	149	71.63%
20,000 - 30,000	263	191	72.62%
30,000 - 40,000	219	159	72.60%
40,000 - 50,000	149	104	69.80%
50,000 - 75,000	264	189	71.59%
75,000 - 100,000	194	144	74.23%
100,000+	1113	776	69.72%

Table 39: Campaign B's account balance categorization-based performance metrics (Source: own calculations based on information provided by Česká Spořitelna).

Regarding the account balance category, the pool is once again highly varied. In the most likely to engage category stands the 7,500 - 10,000 CZK category with 80.73% Open rate. However, with this approach, even debtors show willingness to engage, with more than a 34.17% increase in engagement rate amongst them (Table 39).

5.2.2 Simple Logistic Regression model used on Campaign B

Since Einstein's STO made the send time strictly 07:00 AM or 09:00 AM, instead of the previous "Send day time" referring to before or after 12:00 PM, now its strictly whether it's 07:00 AM (=0) or 09:00 AM (=1).

Just as in Campaign A's case, the analysis of Campaign B starts with the trained simple Logistic Regression model.

LOGISTIC REGRESSION MODEL					
Factor	**Coefficient**	**Std. Error**	**Z value**	**Pr (>|z|)**	**Odds**
Client's age	-0.028	0.047	-0.586	0.558	0.973
Account balance category	-0.046	0.042	-1.097	0.273	0.955
Send day time	0.023	0.094	0.247	0.805	1.024
Gender	-0.103	0.081	-1.271	0.204	0.902
Life phase (categorized as without (1-3) or with family (4-6))	-0.051	0.093	-0.543	0.587	0.951

Table 40: Campaign B's trained simple Logistic Regression model's feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

Model accuracy	72.27%

Table 41: Campaign B's trained simple Logistic Regression model's predictive accuracy (Source: own calculations based on information provided by Česká Spořitelna).

	Predicted not OPEN	**Predicted OPEN**
Actual not OPEN	0	250
Actual OPEN	0	668

Table 42: Campaign B's trained simple Logistic Regression model's predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna).

	Precision	Recall	F1-score	Support
Not OPEN	0.00000	0.00000	0.00000	250
OPEN	0.72767	1.00000	0.84237	668
Macro average	0.36383	0.50000	0.42119	-
Weighted average	0.52950	0.72767	0.61297	-

Table 43: Campaign A's trained simple Logistic Regression model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

Interestingly for Campaign B the same model identified very few factors, and only predicted OPEN states in the much larger than previous 918 sample size, however it is also needs to be noted that it done it with an impressive 72.76% accuracy rating (Table 40 and Table 41).

5.2.3 Pipeline Logistic Regression model used on Campaign B

Using the same metrics the Logistic Regression Pipeline model gave the following metric-evaluation:

LOGISTIC REGRESSION PIPELINE MODEL					
Factor	**Coefficient**	**Std. Error**	**Z value**	**Pr (>|z|)**	**Odds**
Send weekday	0.04548	0.05292	0.85739	0.39123	1.04653
Client's age	-0.09696	0.06301	-1.53869	0.12388	0.90760
Account balance category	-0.06143	0.04654	-1.31481	0.18858	0.94041
Send day time	0.11071	0.11034	0.99997	0.31732	1.11708
Gender	-0.14254	0.09046	-1.60074	0.10943	0.86715
Life phase (categorized as without (1-3) or with family (4-6))	0.03255	0.10495	0.33211	0.73980	1.03309

Table 44: Campaign B's trained Pipeline Logistic Regression model's feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

Model accuracy	71.24%

Table 45: Campaign B's trained Pipeline Logistic Regression model's predictive accuracy (Source: own calculations based on information provided by Česká Spořitelna).

Confusion matrix prediction in a 612 unit sample (Table 46 and Table 47):

	Predicted not OPEN	Predicted OPEN
Actual not OPEN	0	176
Actual OPEN	0	436

Table 46: Campaign B's trained Pipeline Logistic Regression model's predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna)

	Precision	Recall	F1-score	Support
Not OPEN	0.00000	0.00000	0.00000	176
OPEN	0.71242	1.00000	0.83206	436
Macro average	0.35621	0.50000	0.41603	-
Weighted average	0.50754	0.71242	0.59278	-

Table 47: Campaign B's trained Pipeline Logistic Regression model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

Both the simple logistic regression and the Pipeline adjusted model showcase a high level of accuracy while the factor significance evaluation is shoddy at best. This means that the Einstein STO-d campaign has higher cohesion than the Campaign A, as expected (Table 44 and Table 45), with an accuracy rating of 71.24%

5.2.4 Random Forest model used on Campaign B

RANDOM FOREST (HYPERPARAMETER) MODEL	
Factor / Interaction	**Importance**
Account balance category + Client's age	35.96%
Client's age	25.95%
Account balance category	15.67%
Gender + Sent weekday	6.79%
Sent weekday	5.78%
Life phase (categorized as without (1-3) or with family (4-6))	3.55%
Gender	3.23%
Send day time	3.07%

Table 48: Campaign B's trained Random Forest model's feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

Model accuracy	71.24%

Table 49: Campaign B's trained Random Forest model's predictive accuracy (Source: own calculations based on information provided by Česká Spořitelna).

Random Forest's Confusion matrix prediction in a 612 unit sample (Table 50 and Table 51):

	Predicted not OPEN	**Predicted OPEN**
Actual not OPEN	0	176
Actual OPEN	0	436

Table 50: Campaign B's trained Random Forest model's predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna).

	Precision	**Recall**	**F1-score**	**Support**
Not OPEN	0.00000	0.00000	0.00000	176
OPEN	0.71242	1.00000	0.83206	436
Macro average	0.35621	0.50000	0.41603	-
Weighted average	0.50754	0.71242	0.59278	-

Table 51: Campaign B's trained Random Forest model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

In campaign B the Random Forest model's accuracy is the exact same as that of the Logistic Regression Pipeline (Table 45 and Table 49), therefore it means that with Einstein's improvement the more nuanced tree-based Random Forest also finds highly likely patterns (Table 48), confirming the relevance of the thought process behind the model-choices.

5.2.5. Simple learner XGBoost model used on Campaign B

EXTREME GRADIENT BOOST (XGBOOST) MODEL	
Factor / Interaction	**Importance**
Sent weekday	16.50%
Account balance category + Sent weekday	12.55%
Client's age	11.68%
Account balance category + Client's age	11.10%
Life phase (categorized as without (1-3) or with family (4-6))	10.96%
Account balance category	10.91%
Gender + Sent weekday	10.31%
Gender	8.29%
Send day time	7.70%

Table 52: Campaign A's trained standard-scaler XGBoost model's feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

Model accuracy	65.35%

Table 53: Campaign B's trained standard-scaler XGBoost model's accuracy (Source: own calculations based on information provided by Česká Spořitelna).

XGBoost's Confusion matrix prediction in a 801 unit sample (Table 54 and Table 55):

	Predicted not OPEN	**Predicted OPEN**
Actual not OPEN	33	231
Actual OPEN	87	567

Table 54: Campaign B's trained standard-scaler XGBoost model's predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna).

	Precision	Recall	F1-score	Support
Not OPEN	0.27500	0.12500	0.17188	264
OPEN	0.71053	0.86697	0.78099	654
Macro average	0.49276	0.49599	0.47643	-
Weighted average	0.58528	0.65359	0.60582	-

Table 55: Campaign B's trained standard-scaler XGBoost model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

XGBoost significantly weaker in accuracy than previous models in Campaign B, however at least here the model predicted non-open possibilities too.

5.2.6. LightGBM model used on Campaign B

LIGHTGBM MODEL	
Factor / Interaction	**Importance**
Account balance category + Client’s age	10.30%
Life phase	8.60%
Client’s age	3.60%
Account balance category + Sent weekday	2.90%
Gender + Sent weekday	2.90%
Account balance category	2.30%
Gender	1.60%
Sent weekday	0.20%

Table 56: Campaign B’s trained standard-scaler XGBoost model’s feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

Model accuracy	71.18%

Table 57: Campaign B’s trained LightGBM model’s accuracy (Source: own calculations based on information provided by Česká Spořitelna).

XGBoost’s Confusion matrix prediction in a 612 unit sample (Table 58 and Table 59):

	Predicted not OPEN	Predicted OPEN
Actual not OPEN	0	176
Actual OPEN	0	436

Table 58: Campaign B’s trained LightGBM model’s predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna).

	Precision	Recall	F1-score	Support
Not OPEN	0	0	0	176
OPEN	0.71241	1	0.83206	436
Macro average	0.35620	0.5	0.41603	-
Weighted average	0.50754	0.712418	0.59277	-

Table 59: Campaign B's trained LightGBM model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

LightGBM's accuracy is higher than the XGBoost model with 71.18% (Table 57), however once again the predictions only include opens, and it is still slightly weaker than the Random Forest model. This means the predetermined interaction is an incorrect approach, however given it identifies mostly all the same interactions and factors as most important, a pattern can be detected (Table 56).

5.2.7. Simple Logistic Regression model used on Campaign B

NEURALNETWORK MODEL	
Factor / Interaction	**Importance**
Gender	9.69%
Life phase	6.55%
Gender + Sent weekday	5.85%
Life phase (categorized as without (1-3) or with family (4-6))	2.87%
Client's age (polynomial transformation)	2.54%
Account balance category + Sent weekday	1.27%
Send day time	1.04%
Account balance category + Client's age (polynomial transformation)	1.00%
Sent weekday	0.57%
Account balance category (polynomial transformation)	0.53%
Account balance category + Client's age	-0.90%

Table 60: Campaign B's trained NeuralNetwork model's feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

The NeuralNetwork model has a staggering 90.84% accuracy rate (Table 61), which is an understandable near-extreme. The low values on the shown relevant factor suggests that certain factors unshown but understood that have some relevance, which causes this near-extreme positive accuracy rate, however model is also with negative importance rate by the "Account balance category + Client's age" interaction, thus it is possible the model would be more accurate should it be forced to ignore the "Account balance category + Client's age" interaction (Table 60).

Model accuracy	90.84%

Table 61: Campaign B's trained NeuralNetwork model's accuracy (Source: own calculations based on information provided by Česká Spořitelna).

NeuralNetwork's Confusion matrix prediction in a 556 unit sample (Table 62 and Table 63):

	Predicted not OPEN	**Predicted OPEN**
Actual not OPEN	161	15
Actual OPEN	41	395

Table 62: Campaign B's trained NeuralNetwork model's predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna).

	Precision	**Recall**	**F1-score**	**Support**
Not OPEN	0.79703	0.91477	0.85185	100
OPEN	0.96341	0.90596	0.93381	101
Macro average	0.88022	0.91037	0.89283	-
Weighted average	0.91557	0.90850	0.91024	-

Table 63: Campaign B's trained NeuralNetwork model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

5.2.8 Mixed model used on Campaign B

The hybrid model is interestingly the weakest performing model of them all. This should mean that with Einstein's STO the hybrid model is plain overcompensation.

NEURALNETWORK + RANDOM FOREST (HYPERPARAMETER) + XGBOOST	
Factor / Interaction	**Importance**
Life phase	21.49%
Life phase (categorized as without (1-3) or with family (4-6))	16.74%
Gender + Sent weekday	11.48%
Gender	11.24%
Account balance category + Sent weekday	8.10%
Client's age (polynomial transformation)	7.48%
Account balance category (polynomial transformation)	6.39%
Send day time	5.40%
Account balance category + Client's age	4.54%
Sent weekday	3.86%
Account balance category + Client's age (polynomial transformation)	3.28%

Table 64: Campaign B's trained ML+DL model's feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

Model accuracy	59.64%

Table 65: Campaign B's trained ML+DL model's accuracy (Source: own calculations based on information provided by Česká Spořitelna).

Model's Confusion matrix prediction in a 612 unit sample (Table 66 and Table 67):

	Predicted not OPEN	Predicted OPEN
Actual not OPEN	44	132
Actual OPEN	115	321

Table 66: Campaign B's trained ML+DL model's predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna).

	Precision	Recall	F1-score	Support
Not OPEN	0.27673	0.25000	0.26269	176
OPEN	0.70861	0.73624	0.72216	436
Macro average	0.49267	0.49312	0.49242	-
Weighted average	0.58441	0.59641	0.59002	-

Table 67: Campaign B's trained ML+DL model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

5.3. NeuralNetwork on Campaign A and a combined database

The DL NeuralNetwork model trained on campaign B shown extremely promising results, therefore the observation needs to be made regarding how accurate the model is when dealing with the less efficient campaign, and the two databases combined.

5.3.1. Campaign B-trained NeuralNetwork used on Campaign A

CAMPAIGN B-TRAINED NEURALNETWORK MODEL ON CAMPAIGN A	
Factor / Interaction	**Importance**
Life phase	33.50%
Send day time	18.23%
Account balance category + Client's age	14.29%
Account balance category + Client's age (polynomial transformation)	13.30%
Sent weekday	10.84%
Account balance category (polynomial transformation)	9.36%
Life phase (categorized as without (1-3) or with family (4-6))	8.87%
Client's age (polynomial transformation)	7.39%

Table 68: Campaign B-trained NeuralNetwork projected onto campaign A, feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

Model accuracy	52.24%

Table 69: Campaign B-trained NeuralNetwork projected onto campaign A, predictive accuracy (Source: own calculations based on information provided by Česká Spořitelna).

Campaign B-trained NeuralNetwork's Confusion matrix prediction on campaign A in a 201 unit sample (Table 70 and Table 71):

	Predicted not OPEN	Predicted OPEN
Actual not OPEN	45	55
Actual OPEN	41	60

Table 70: Campaign B-trained NeuralNetwork projected onto campaign A, model's predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna).

	Precision	Recall	F1-score	Support
Not OPEN	0.52326	0.45000	0.48387	100
OPEN	0.52174	0.59406	0.55556	101
Macro average	0.52250	0.52203	0.51971	-
Weighted average	0.52249	0.52239	0.51989	-

Table 71: Campaign B-trained NeuralNetwork projected onto campaign A, model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

The campaign B-trained NeuralNetwork model used on the campaign A's dataset shows that despite the high accuracy level with campaign B, using on campaign A has lower accuracy rate than the model organically trained on campaign A, with an accuracy rate of just 52.24%, (Table 69).

5.3.2. Campaign B-trained NeuralNetwork used on combined dataset

CAMPAIGN B-TRAINED NEURALNETWORK MODEL ON THE COMBINDE CAMPAIGNS' DATASET	
Factor / Interaction	**Importance**
Gender + Sent weekday	13.10%
Account balance category (polynomial transformation)	11.38%
Account balance category + Client's age	11.06%
Life phase	10.57%
Account balance category + Sent weekday	10.39%
Account balance category + Client's age (polynomial transformation)	9.90%
Client's age (polynomial transformation)	9.82%
Gender	8.70%
Life phase (categorized as without (1-3) or with family (4-6))	8.35%
Sent weekday	5.97%
Send day time	0.75%

Table 72: Campaign B-trained ML+DL projected onto a combined campaign A and B dataset, feature importance evaluation (Source: own calculations based on information provided by Česká Spořitelna).

Model accuracy	60.15%

Table 73: Campaign B-trained ML+DL projected onto a combined campaign A and B dataset, model's predictive accuracy (Source: own calculations based on information provided by Česká Spořitelna).

NeuralNetwork's Confusion matrix prediction in a 813 unit sample (Table 74 and Table 75):

	Predicted not OPEN	Predicted OPEN
Actual not OPEN	78	198
Actual OPEN	126	411
TOTAL	**204**	**609**
RATE	**25.09%**	**74.91%**

Table 74: Campaign B-trained ML+DL projected onto a combined campaign A and B dataset, model's predictions projected on the actual database (Source: own calculations based on information provided by Česká Spořitelna).

	Precision	Recall	F1-score	Support
Not OPEN	0.38235	0.28261	0.32500	276
OPEN	0.67488	0.76536	0.71728	537
Macro average	0.52861	0.52399	0.52114	-
Weighted average	0.57557	0.60148	0.58411	-

Table 75: Campaign B-trained NeuralNetwork projected onto a combined campaign A and B dataset, model's prediction classification report (Source: own calculations based on information provided by Česká Spořitelna).

The campaign B-trained NeuralNetwork model used on a combined dataset has resulted in an accuracy rating of 60.14%, which is relatively low compared to the results of various models for campaign B, however higher than any models' performance with campaign A. This means that the interpretation of data is better, however overall the prediction is still overtly biased towards open rate, just shy of 0.09% the three-quarters of all predictions are OPEN.

6. CONCLUSIONS

From 2nd of December 2024 to the 27th of December 2024, the leading Czech bank Česká Spořitelna conducted two, subject- and content-wise identical marketing warm email campaigns, where one campaign, "Campaign A" lasted from 9th of December 2024 to the 22nd of December 2024, and the second one, "Campaign B", lasted from the full range of 2nd of December 2024 to 27th of December 2024. The key difference between the two, is that Campaign A used human curated Send Time Optimization (STO), while Campaign B used Česká Spořitelna's marketing partner SalesForce's AI "Einstein" to make STO optimizations. These campaigns were designed to test the STO capabilities of the Einstein AI. To track their email's performance metric, Česká Spořitelna is also in SalesForce's Feedback Loop List, which gives feedback data on email handling by clients, from how the email host categorized the email to whether the client opened it, irregardless of reaction to the email's content.

Campaign A's human-curated STO-strategy had emails sent out at a large variety of day time and hour. It reached 1,005 recipients, with an open rate of 48.85%, with male recipients being more likely to open. Age-wise the 18-25 years old group was the most likely to open their emails, the 60+ group the only other reaching more than 50%, while life-phase wise (as Česká Spořitelna keeps track of their client's life situation), appropriate to the age-grouping results 18+ year old students are the most likely to open, and appropriately the "Retired" groups the second. Account balance-wise the pool varied, but in comic irony the people in debt were the least likely to engage with the marketing email in Campaign A.

Campaign B, using Einstein's STO, was conducted through a longer period, and to stress-test the system it reached 3,371 customers, with 3,059 accountable feedback (the rest had missing or corrupt data, therefore those were ignored). Einstein's STO strategy was to send strictly at 07:00 AM or 09:00 AM to clients with unexpressed pattern of behaviour / metric, and it resulted in 71.20% open rate, which means Einstein's STO was 22.35% better, but if we also take into consideration that Einstein had a 3.35 times larger base to deal with, appropriately counting Einstein did $22.35 \times 3.35 = 74.87\%$ better. In every category, Einstein's STO had higher engagement rate than the highest rates in the Campaign A, meaning the AI's performance is the ultimate approach.

However, 71.20% is still not 100%. With this in mind, even Einstein STO needs to be improved upon. Looking at the research on advanced email marketing STO, it had been concluded that the popular attempt of further STO optimization is done using Machine Learning (ML) and Deep Learning (DL) statistical models to work with large datasets of historic email-engagement behaviour of clients.

Research into the topic in the early 2020s initially suggested that Machine Learning tools, such as Linear Regression, Random Forest and Gradient Boosted learners (XGBoost and LightGMB) were the superior way to go, while the Deep Learning tool NeuralNetwork may work as a supplementer, but ultimately cannot be a correct interpreter. With this in mind the Einstein AI developed too. Therefore that begged the question whether Einstein AI's STO can be optimized even further? And also, would the human-curated STO reach at least similar levels of success if they'd use more nuanced ML or DL models?

To test this, several ML and DL models were used with train/test approach in Python program language to examine both campaigns, to attempt to identify key features that contributes to positive engagement behaviour, aka open, and have the models attempt to predict open rates based on the features-assumed behavioural pattern, reflected at the actual databases.

Starting with Campaign A, at first two linear ML tools were used – basic Logistic Regression and Pipeline processor module inserted Logistic Regression model. Both model resulted in a prediction accuracy of around 50%, with the Pipeline Logistic Regression reaching the accuracy rate of 54.23%, however its identified predictors has resulted in little statistical relevance, which means that the logic was not based on tangible statistics. Therefore, according to the literature, the next step is a tree-based decision making ML model, the Random Forest.

The Random Forest (Hyperparameter) model resulted in a lower accuracy rating than that of the Pipeline Logistic Regression, however the Random Forest model identified certain feature interactions as major predictors, which means that it is impossible that the Pipeline Logistic Regression suggested strict linearity of individual feature influence on the outcome is real.

Nevertheless, the Random Forest model's low accuracy rating shows it didn't grasp the concrete pattern either. Moving on, the next step was using a weaker base learner, the standard scaler, but enhancing it with the Extreme Gradient Boost (XGBoost) booster, which resulted in a… lower performing model.

Moving on to the last ML used, the Light Gradient Boost Machine (LightGBM) model which once again uses a weaker learner only to be powered up, has yielded a modest 50.24% accuracy rating, nailing the coffin on the possibility that the decision-tree based learners can grasp the underlying patterns of what drives clients to engage with the emails sent by Česká Spořitelna. So, switching to DL, a NeuralNetwork ensemble had been used.

With a respectable rate of accuracy 53.23% shows that DL is in fact capable of grasping appropriate patterns, and while the Pipeline Logistic Regression had higher accuracy rate, the identifications made by the NeuralNetwork model seems very plausible statistically.

Lastly, building on the NeuralNetwork, an enhanced ML+DL model has been designed by combining the NeuralNetwork, Random Forest and XGBoost models. The results are not crazy, however still holds the second place within all of the used models with 52.24% of accuracy.

Next, the same models have been used on Campaign B's dataset. The results across the board were significantly better, with the simple Logistic Regression reaching 72.27%, both Pipeline Logistic Regression and Random Forest reaching 71.24%, XGBoost reaching 65.35% and LightGBM 71.18%. The biggest shock however was the results of the ensemble NeuralNetwork model, with an incredible 90.84% predictive rate. This means the DL model with the optimization provided by the Einstein AI has grasped the underlying logic. Attempting to upgrade upon this only made it weaker, with the ML+DL model only reaching 59.64% accuracy. However, the fact remains that in with the Einstein's STO in Campaign B, the NeuralNetwork deep learner has reached superior conclusivity power.

To see this models accuracy rate, the NeuralNetwork model trained on Campaign B was tested on Campaign A, which did not have strong results, with an accuracy rate lower than the non campaign B trained NeuralNetwork model, and also on a combined dataset of both campaigns, which's result, albeit respectable compared to other results in Campaign A, still fell short of the overall results in Campaign B and especially the sole NeuralNetwork on Campaign B.

Overall these results suggest that the logic behind the STO provided by Einstein AI is very strong, and when paired with a proper system that can grasp it, it can be improved upon, as signified by the NeuralNetwork DL's performance on Campaign B. SalesForce's AI developers may want to consider the implications of this, and utilize the resulting knowledge.

7. LIST OF SOURCES

Abbas, Dalia Saleh and Al-Jailawi, Mustafa (2024): Data-Driven Strategies for Improving Email Campaign Engagement: A Send Time Optimization Approach. Lund University Press, Lund, Sweden. Downloaded from https://lup.lub.lu.se/luur/download?func=downloadFile&recordOId=9174474&fileOId=9174481 (03.10.2025)

Al-Ababneh, Hassan Ali (2024): Electronic Banking Marketing. Dar Al-Swaqe Alelmieh, Jordan. Downloaded from https://www.researchgate.net/publication/385319138_ELECTRONIC_BANKING_MARKETING (14.03.2025)

Araujo, Carolina; Soares, Cristophe; Pereira, Ivo; Coelho, Duarte; Rebelo, Miguel Angelo and Madureira, Ana (2022): A Novel Approach for Send Time Prediction on Email Marketing. Applied Sciences, 12(8), 1-13. https://doi.org/10.3390/app12168310

Bagui, Sikha; Nandi, Debarghya; Bagui, Subhash and White, Rober Jaimie (2021): Machine Learning and Deep Learning for Phishing Email Classification using One-Hot Encoding. *Journal of Computer Science*, 17(7), pp. 610-623.

Česká Spořitelna (2018): Valná hromada ČS rozhodla 3. října 2018 o přechodu všech akcií ostatních akcionářů České spořitelny, a.s. na hlavního akcionáře – Erste Group Bank AG. Downloaded from https://www.csas.cz/cs/o-nas/pro-media/tiskove-zpravy/2018/10/03/akcionari-n a-valne-hromade-rozhodli-o-prechodu-vsech-akcii-ostatnich-akcionaru-cs-na-spolecnost-erste-group-bank-ag (20.03.2025)

Česká Spořitelna (2021): Annual Report 2020. Downloaded from https://www.csas.cz/static_internet/en/Redakce/Ostatni/Ostatni_IE/Prilohy/vz-2020.pdf (25.03.2025)

Česká Spořitelna (2025): Annual Report 2024. Downloaded from https://www.csas.cz/static_internet/en/Redakce/Ostatni/Ostatni_IE/Prilohy/annual_report_2024.pdf (25.03.2025)

Czech Academy of Sciences (CAS) (2025): Headquarters of the CAS - History of the Building. Downloaded from https://www.avcr.cz/en/about-us/history/Headquarters-of-the-CAS/ (22.03.2025)

Deshmukh, Tushar (2024): A Research Paper On Effectivness Of E-Mail Marketing For Customer. Downloaded from https://www.researchgate.net/publication/384355097_A_RESEARCH_PAPER_ON_EFFECTIVNESS_OF_E-MAIL_MARKETING_FOR_CUSTOMER (10.03.2025) dictionary.cambridge.org (2025): spam. Downloaded from https://dictionary.cambridge.org/dictionary/english/spam (23.02.2025).

Gul, Samina; Rohma, Ahmad Rana and Kiran, Samra (2024): The Impact Of Email Marketing On Consumer Buying Decision Process In The Pakistani Market. *Journal for Business Education and Management*, 4(1), 209-226.

Gupta, Namit and Rana, Sudhir (2017): Perception toward Permission-based E-mails in Banks. *Jindal Journal of Business Research*, 6(1), pp. 1-10.

Hafeez, Nida Abdullah; Nasir, Muhammad Umar; Shabbir, Maryam and Mehmood, Shahid (2024): ersonalized Email Marketing: A Machine LearningApproach for Higher Engagement and Conversion Rates. Conference: October 2024 conference on Horizons of Information Technology and Engineering (HITE), at: Lahore, Pakistan. Downloaded from https://www.researchgate.net/publication/386585384_Personalized_Email_Marketing_A_Machine_Learning_Approach_for_Higher_Engagement_and_Conversion_Rates (04.03.2025)

Hruška, Zdeněk and Dvořáková, Lilia (2015): Development Milestones And Tendencies Of Banking In The Czech Lands In The Context Of Financial Literacy Development. *Trendy v Podnikání*, 5(2), pp. 13-20.

Irohlas, Javezš (2020): Česká spořitelna slaví 195 let. Chudší obyvatelé měli díky ní lépe hospodařit s penězi. Downloaded from https://www.irozhlas.cz/ekonomika/ceska-sporitelna- slavi-195-let-chudsi-obyvatele-meli-diky-ni-lepe-hospodarit-s_2002121748_ako (22.03.2025)

Kalmár, Péter; Zéman, Zoltán and Lukács, János (2015): Bank and controlling with a marketing attitude applied statistics in the service of controlling. *Financial and Economic Review*, 14(4), pp. 110-125.

Kaminyar, Kevin (2023): 6 Steps To The World's Best Cold Email. Downloaded from https://www.l-spark.com/wp-content/uploads/2024/10/6-Steps-To-The-Worlds-Best-Cold-Email-_-Entrepreneur.pdf (08.03.2025)

Le Plaisir, Morris (2024a): Specific Target: How Important Is Personalization In Cold Emails? American Institute for Research & Innovation, Sacramento, California, USA. Downloaded from https://www.researchgate.net/publication/378488355_SPECIFIC_TARGET_HOW_IMPORTANT_IS_PERSONALIZATION_IN_COLD_EMAILS (09.03.2025)

Le Plaisir, Morris (2024b): How To Build The Perfect Cold Email Strategy And Why Creating Campaigns With A Maximum Of 2 Emails. American Institute for Research & Innovation, Sacramento, California, USA. Downloaded from https://www.researchgate.net/publication/380216144_HOW_TO_BUILD_THE_PERFECT_COLD_EMAIL_STRATEGY_AND_WHY_CREATING_CAMPAIGNS_WITH_A_MAXIMUM_OF_2_EMAILS (09.03.2025)

Lutfil, Hardy (2021): Cold Email Marketing Effectiveness and Localisation in Esports. Acta Universitatis Danubius, Kajaani, Finland. Downloaded from https://www.theseus.fi/bitstream/handle/10024/512545/Cold%20Email%20Marketing%20Effectiveness%20and%20Localisation%20in%20Esports.pdf?sequence=2 (06.03.2025)

Macher, Flora (2018): The Austrian Banking Crisis Of 1931: One Bad Apple Spoils The Whole Bunch. Downloaded from https://eprints.lse.ac.uk/87151/1/wp274.pdf (24.03.2025)

Patil, Dimple (2024): Email marketing with artificial intelligence: Enhancing personalization, engagement, and customer retention. Hurix Digital, Andheri, India. Downloaded from https://www.researchgate.net/publication/385772630_Email_marketing_with_artificial_intelligence_Enhancing_personalization_engagement_and_customer_retention (15.03.2025)

Pavlidou, Dora (2025): Gary Thuerk: “People Make The Same Mistakes Over And Over Again”. Downloaded from https://moosend.com/blog/gary-thuerk-people-make-the-same-mistakes-over -and-over-again/ (22.02.2025).

Potrimba, Petru (2022): Time Sent Optimization: The Diennea Case. Downloaded from https://amslaurea.unibo.it/id/eprint/24111/1/potrimba_petru_thesis.pdf (03.03.2025)

Rabab’ah, Ghaleb; Yagi, Sane; Alghazo, Sharif and Malkawi, Rima Jamil (2024): Persuasive strategies in email marketing: An analysis of appeal and influence in business communication. *Journal of Intercultural Communication*, 24(2), pp.

Ravichandran, M. and Kumar, Kishore K. (2024): The Effectiveness Of Email Marketing For Engaging Customer. *International Journal of Research Publication and Reviews*, 5(12), pp. 621-625.

Sabbagh, Foued (2021): Email Marketing: The most Important Advantages and Disadvantages. *Journal of Economics and Technology Research*, 2(3), pp. 14-29.

Soukup, Marek; Žídek, Libor (2021) : Growth decomposition in the socialist Czech economy. *Law, Economics and Social Issues Review*, 12(1), pp. 1-12.

Tambe, Niranja and Solanki, Chetan (2022): What are the Most Significant Benefits and Drawbacks of Email Marketing? *International Journal of Advanced Research in Science, Communication and Technology*, 2(3), pp. 258-267.

Torossian, Ronn (2025): The Pitfalls of Email Marketing in 2025: What Went Wrong? Downloaded from https://ronntorossian.com/the-pitfalls-of-email-marketing-in-2025-what-went-wrong/ (05.03. 2025)

Tvrdá, Kateřina (2017): Heinrich Eduard Herz (1785–1849). Geschichte eines jüdischen Zuckerfabrikanten. In book: Němeček and Prokš edt.: Moderní Dějiny. Institute of History, Prague, Czechia. Downloaded from: https://www.hiu.cas.cz/user_uploads/vydavatelska_cinnost/periodika/moderni_dejiny/md_25_1_2017_fin.pdf (25.03.2025)

APPENDIX

Appendix I. ML and DL model Python scripts used for Campaign A.

Simple Logistic Regression model

```
import pandas as pd
import numpy as np
import statsmodels.api as sm
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
file_path = "D:\\CampaignA.xlsx"
df = pd.read_excel(file_path)
df = df.drop(columns=["ID", "Sent_time", "Chage_State_Time"])
X = df[["Sent_Day(/7)", "Morning_Evening", "GNDR", "Client_Age",
"Life_Phase_cat", "balance_mean"]]
y = df["OPEN?"]
scaler = StandardScaler()
X[["Client_Age", "balance_mean"]] =
scaler.fit_transform(X[["Client_Age", "balance_mean"]])
X = sm.add_constant(X)
model = sm.Logit(y, X)
result = model.fit()
summary_df = pd.DataFrame({
"Feature": X.columns,
"Coefficient": result.params,
"Std. Error": result.bse,
"Z value": result.tvalues,
"Pr (>|z|)": result.pvalues,
"Odds": np.exp(result.params)
})
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42)
y_pred = (result.predict(X_test) >= 0.5).astype(int)
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred, output_dict=True)
eval_results = pd.DataFrame({
"Metric": ["Accuracy"],
"Value": [accuracy]
})
output_file = "D:\\campalogre.xlsx"
with pd.ExcelWriter(output_file) as writer:
summary_df.to_excel(writer, sheet_name="Model Results", index=False)
eval_results.to_excel(writer, sheet_name="Evaluation", index=False)
pd.DataFrame(conf_matrix, columns=["Predicted 0", "Predicted 1"],
index=["Actual 0", "Actual 1"]).to_excel(writer, sheet_name="Confusion
Matrix")
```

```
pd.DataFrame(class_report).transpose().to_excel(writer,
sheet_name="Classification Report")
```

Pipeline Logistic Regression model

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
df = pd.read_excel("D:/CampaignA.xlsx")
df_clean = df.drop(columns=['ID', 'Sent_time', 'Chage_State_Time'])
df_clean.rename(columns={
'Sent_Day(/7)': 'Sent_Day',
'Morning_Evening': 'Morning_Evening',
'OPEN?': 'Opened',
'GNDR': 'Gender',
'Client_Age': 'Age',
'Life_Phase_cat': 'Life_Obligations',
'balance_mean': 'Balance'
}, inplace=True)
df_clean.dropna(subset=['Opened', 'Sent_Day', 'Morning_Evening',
'Gender', 'Age', 'Life_Obligations', 'Balance'], inplace=True)
X = df_clean[['Sent_Day', 'Morning_Evening', 'Gender', 'Age',
'Life_Obligations', 'Balance']]
y = df_clean['Opened']
preprocessor = ColumnTransformer(
    transformers=[
('num', StandardScaler(), ['Sent_Day', 'Age', 'Balance']),
('cat', OneHotEncoder(), ['Morning_Evening', 'Gender',
'Life_Obligations'])
]
)
model = Pipeline(steps=[
('preprocessor', preprocessor),
('classifier', LogisticRegression(max_iter=1000, random_state=42))
])
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42, stratify=y)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred, output_dict=True)
class_report_df = pd.DataFrame(class_report).transpose()

results_df = pd.DataFrame({
"Metric": ["Accuracy"],
"Value": [accuracy]
})
```

```
conf_matrix_df = pd.DataFrame(conf_matrix,
columns=[f'Predicted {i}' for i in range(conf_matrix.shape[1])],
index=[f'Actual {i}' for i in range(conf_matrix.shape[0])])
save_path = "D:/campapipe.xlsx"
with pd.ExcelWriter(save_path) as writer:
results_df.to_excel(writer, sheet_name="Model Results", index=False)
conf_matrix_df.to_excel(writer, sheet_name="Confusion Matrix")
class_report_df.to_excel(writer, sheet_name="Classification Report")
predictions_df = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
predictions_df.to_excel(writer, sheet_name="Predictions", index=False)
```

Random Forest (Hyperparameter)

```
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split,
RandomizedSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
import os
file_path = r"D:\CampaignA.xlsx"
df = pd.read_excel(file_path)
df['GNDR_Sent_Day'] = df['GNDR'] * df['Sent_Day(/7)']
df['Balance_ClientAge'] = df['balance_mean'] * df['Client_Age']
scaler = StandardScaler()
df[['balance_mean', 'Client_Age']] =
scaler.fit_transform(df[['balance_mean', 'Client_Age']])
features = ['Sent_Day(/7)', 'balance_mean', 'Client_Age', 'GNDR',
'Morning_Evening',
'Life_Phase_cat', 'GNDR_Sent_Day', 'Balance_ClientAge']
target = 'OPEN?'
X = df[features]
y = df[target]
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42, stratify=y)
param_dist = {
'n_estimators': [50, 100, 200, 300],
'max_depth': [5, 10, 20, None],
'min_samples_split': [2, 5, 10],
'min_samples_leaf': [1, 2, 4],
'class_weight': [None, 'balanced', 'balanced_subsample']
}
rf = RandomForestClassifier(random_state=42)
rf_cv = RandomizedSearchCV(rf, param_dist, n_iter=10, cv=5,
scoring='accuracy', n_jobs=-1, random_state=42)
rf_cv.fit(X_train, y_train)
best_rf = rf_cv.best_estimator_
y_pred = best_rf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred, output_dict=True)
```

```
feature_importance = pd.DataFrame({'Feature': features, 'Importance':
best_rf.feature_importances_})
feature_importance = feature_importance.sort_values(by='Importance',
ascending=False)
output_path = r"D:\camparandoforest_optimized.xlsx"
with pd.ExcelWriter(output_path) as writer:
pd.DataFrame({'Metric': ['Accuracy'], 'Value':
[accuracy]}).to_excel(writer, sheet_name='Evaluation', index=False)
feature_importance.to_excel(writer, sheet_name='Feature Importance',
index=False)
pd.DataFrame(conf_matrix, columns=['Predicted 0', 'Predicted 1'],
index=['Actual 0', 'Actual 1']).to_excel(writer, sheet_name='Confusion
Matrix')
pd.DataFrame(class_report).transpose().to_excel(writer,
sheet_name='Classification Report')
```

XGBoost (Standard-Scaler)

```
import pandas as pd
import numpy as np
import xgboost as xgb
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
from sklearn.preprocessing import StandardScaler
file_path = r"D:\CampaignA.xlsx"
df = pd.read_excel(file_path)
features = ["Sent_Day(/7)", "Morning_Evening", "GNDR", "Client_Age",
"Life_Phase_cat", "balance_mean"]
target = "OPEN?"
df["Balance_ClientAge"] = df["balance_mean"] * df["Client_Age"]
df["GNDR_Sent_Day"] = df["GNDR"] * df["Sent_Day(/7)"]
df["Balance_Sent_Day"] = df["balance_mean"] * df["Sent_Day(/7)"]
selected_features = features + ["Balance_ClientAge", "GNDR_Sent_Day",
"Balance_Sent_Day"]
X = df[selected_features]
y = df[target]
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42, stratify=y)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
model = xgb.XGBClassifier(objective="binary:logistic",
eval_metric="logloss", use_label_encoder=False)
model.fit(X_train_scaled, y_train)
y_pred = model.predict(X_test_scaled)
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred, output_dict=True)
feature_importance = pd.DataFrame({"Feature": selected_features,
"Importance": model.feature_importances_})
feature_importance = feature_importance.sort_values(by="Importance",
ascending=False)
```

```
output_path = r"D:\campaXGBoost.xlsx"
with pd.ExcelWriter(output_path) as writer:
feature_importance.to_excel(writer, sheet_name="Feature Importance",
index=False)
pd.DataFrame({"Metric": ["Accuracy"], "Value":
[accuracy]}).to_excel(writer, sheet_name="Evaluation", index=False)
pd.DataFrame(conf_matrix, index=["Actual 0", "Actual 1"],
columns=["Predicted 0", "Predicted 1"]).to_excel(writer,
sheet_name="Confusion Matrix")
pd.DataFrame(class_report).transpose().to_excel(writer,
sheet_name="Classification Report")
```

LightGBM

```
import pandas as pd
import numpy as np
import shap
import matplotlib.pyplot as plt
import seaborn as sns
from lightgbm import LGBMClassifier
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
file_path = "D:/CampaignAlite.xlsx"
df = pd.read_excel(file_path)
df["GNDR"] = df["GNDR"].astype(int)
df["Morning_Evening"] = df["Morning_Evening"].astype(int)
df["Life_Phase_cat"] = df["Life_Phase_cat"].astype(int)
df["Balance+Sent_Day"] = df["balance_mean"] * df["Sent_Day(/7)"]
df["GNDR+Sent_Day"] = df["GNDR"] * df["Sent_Day(/7)"]
df["Balance+ClientAge"] = df["balance_mean"] * df["Client_Age"]
X = df.drop(columns=["OPEN?"])
y = df["OPEN?"]
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42, stratify=y)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
lgbm = LGBMClassifier(verbose=-1, random_state=42)
param_grid = {
"n_estimators": [50, 100, 200],
"max_depth": [3, 5, 7],
"learning_rate": [0.01, 0.05, 0.1]
}
grid = GridSearchCV(lgbm, param_grid, cv=5, scoring="accuracy",
verbose=1, n_jobs=-1)
grid.fit(X_train_scaled, y_train)
best_model = grid.best_estimator_
y_pred = best_model.predict(X_test_scaled)
accuracy = accuracy_score(y_test, y_pred)
cm = confusion_matrix(y_test, y_pred)
```

```
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=["Not
Opened", "Opened"], yticklabels=["Not Opened", "Opened"])
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
explainer = shap.TreeExplainer(best_model)
shap_values = explainer.shap_values(X_test_scaled)
shap.summary_plot(shap_values, X_test, feature_names=X.columns)
feature_importance = pd.DataFrame({
"Feature": X.columns,
"Importance": best_model.feature_importances_
}).sort_values(by="Importance", ascending=False)
feature_importance.to_excel("D:/CampaignA_FeatureImportance.xlsx",
index=False)
```

NeuralNetwork

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neural_network import MLPClassifier
from sklearn.ensemble import RandomForestClassifier, StackingClassifier
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
from sklearn.inspection import permutation_importance
from imblearn.over_sampling import SMOTE
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import PolynomialFeatures
file_path = "D:/CampaignAlite.xlsx"
df = pd.read_excel(file_path)
datetime_cols = df.select_dtypes(include=["datetime64"]).columns
if len(datetime_cols) > 0:
print(f"Datetime columns found: {list(datetime_cols)} - Converting to
numerical format.")
for col in datetime_cols:
df[col] = df[col].astype(np.int64) // 10**9
convert_cols = ["GNDR", "7or9", "Life_Phase_cat"]
for col in convert_cols:
if col in df.columns and not np.issubdtype(df[col].dtype,
np.datetime64):
df[col] = df[col].astype(int)
df["Balance+Sent_Day"] = df["balance_mean"] * df["Sent_Day(/7)"]
df["GNDR+Sent_Day"] = df["GNDR"] * df["Sent_Day(/7)"]
df["Balance+ClientAge"] = df["balance_mean"] * df["Client_Age"]
df["Is_Weekend"] = df["Sent_Day(/7)"].apply(lambda x: 1 if x in [6, 7]
else 0)
X = df.drop(columns=["OPEN?"])
```

```
y = df["OPEN?"]
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42, stratify=y)
smote = SMOTE(random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train,
y_train)
preprocessor = ColumnTransformer([
("poly", PolynomialFeatures(degree=2, interaction_only=True),
["balance_mean", "Client_Age"])
], remainder="passthrough")
scaler = StandardScaler()
X_train_scaled =
scaler.fit_transform(preprocessor.fit_transform(X_train_resampled))
X_test_scaled = scaler.transform(preprocessor.transform(X_test))
rf = RandomForestClassifier(n_estimators=200, max_depth=7,
random_state=42)
xgb = XGBClassifier(n_estimators=200, learning_rate=0.1, max_depth=5,
random_state=42, use_label_encoder=False, eval_metric='logloss')
nn = MLPClassifier(hidden_layer_sizes=(128, 64, 32), activation='relu',
solver='adam', max_iter=1000, random_state=42)
stacking_clf = StackingClassifier(
estimators=[("rf", rf), ("xgb", xgb), ("nn", nn)],
final_estimator=RandomForestClassifier(n_estimators=100,
random_state=42)
)
stacking_clf.fit(X_train_scaled, y_train_resampled)
y_pred = stacking_clf.predict(X_test_scaled)
accuracy = accuracy_score(y_test, y_pred)
cm = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred, output_dict=True)
print(f"Accuracy: {accuracy:.4f}")
print("\nClassification Report:\n", classification_report(y_test,
y_pred))
plt.figure(figsize=(5,5))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=["Not
Opened", "Opened"], yticklabels=["Not Opened", "Opened"])
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
perm_importance = permutation_importance(stacking_clf, X_test_scaled,
y_test, scoring='accuracy', n_repeats=10, random_state=42)
feature_importance = pd.DataFrame({
"Feature": preprocessor.get_feature_names_out(),
"Importance (%)": (perm_importance.importances_mean /
perm_importance.importances_mean.sum()) * 100
}).sort_values(by="Importance (%)", ascending=False)
results_df = pd.DataFrame({
"Metric": ["Accuracy"],
"Value": [accuracy]
})
class_report_df = pd.DataFrame(class_report).transpose()
```

```
cm_df = pd.DataFrame(cm, index=["Actual_Not_Opened", "Actual_Opened"],
columns=["Predicted_Not_Opened", "Predicted_Opened"])
output_path = "D:/CampaignA_FeatureImportance_Stacking.xlsx"
with pd.ExcelWriter(output_path) as writer:
feature_importance.to_excel(writer, sheet_name="Feature Importance",
index=False)
results_df.to_excel(writer, sheet_name="Model Accuracy", index=False)
cm_df.to_excel(writer, sheet_name="Confusion Matrix")
class_report_df.to_excel(writer, sheet_name="Classification Report")
```

Neural Network + XGBoost + Random Forest

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neural_network import MLPClassifier
from sklearn.ensemble import RandomForestClassifier, StackingClassifier
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
from sklearn.inspection import permutation_importance
from imblearn.over_sampling import SMOTE
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import PolynomialFeatures
file_path = "D:/CampaignAlite.xlsx"
df = pd.read_excel(file_path)
df["GNDR"] = df["GNDR"].astype(int)
df["Morning_Evening"] = df["Morning_Evening"].astype(int)
df["Life_Phase_cat"] = df["Life_Phase_cat"].astype(int)
df["Balance+Sent_Day"] = df["balance_mean"] * df["Sent_Day(/7)"]
df["GNDR+Sent_Day"] = df["GNDR"] * df["Sent_Day(/7)"]
df["Balance+ClientAge"] = df["balance_mean"] * df["Client_Age"]
df["Is_Weekend"] = df["Sent_Day(/7)"].apply(lambda x: 1 if x in [6, 7]
else 0)
X = df.drop(columns=["OPEN?"])
y = df["OPEN?"]
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42, stratify=y)
smote = SMOTE(random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train,
y_train)
preprocessor = ColumnTransformer([
("poly", PolynomialFeatures(degree=2, interaction_only=True),
["balance_mean", "Client_Age"])
], remainder="passthrough")
scaler = StandardScaler()
X_train_scaled =
scaler.fit_transform(preprocessor.fit_transform(X_train_resampled))
X_test_scaled = scaler.transform(preprocessor.transform(X_test))
```

```
rf = RandomForestClassifier(n_estimators=200, max_depth=7,
random_state=42)
xgb = XGBClassifier(n_estimators=200, learning_rate=0.1, max_depth=5,
random_state=42, use_label_encoder=False, eval_metric='logloss')
nn = MLPClassifier(hidden_layer_sizes=(128, 64, 32), activation='relu',
solver='adam', max_iter=1000, random_state=42)
stacking_clf = StackingClassifier(
estimators=[("rf", rf), ("xgb", xgb), ("nn", nn)],
final_estimator=RandomForestClassifier(n_estimators=100,
random_state=42)
)
stacking_clf.fit(X_train_scaled, y_train_resampled)
y_pred = stacking_clf.predict(X_test_scaled)
accuracy = accuracy_score(y_test, y_pred)
cm = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred, output_dict=True)
plt.figure(figsize=(5,5))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=["Not
Opened", "Opened"], yticklabels=["Not Opened", "Opened"])
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
perm_importance = permutation_importance(stacking_clf, X_test_scaled,
y_test, scoring='accuracy', n_repeats=10, random_state=42)
feature_importance = pd.DataFrame({
"Feature": preprocessor.get_feature_names_out(),
"Importance (%)": (perm_importance.importances_mean /
perm_importance.importances_mean.sum()) * 100
}).sort_values(by="Importance (%)", ascending=False)
results_df = pd.DataFrame({
"Metric": ["Accuracy"],
"Value": [accuracy]
})
class_report_df = pd.DataFrame(class_report).transpose()
cm_df = pd.DataFrame(cm, index=["Actual_Not_Opened", "Actual_Opened"],
columns=["Predicted_Not_Opened", "Predicted_Opened"])
output_path = "D:/CampaignA_FeatureImportance_Stacking.xlsx"
with pd.ExcelWriter(output_path) as writer:
feature_importance.to_excel(writer, sheet_name="Feature Importance",
index=False)
results_df.to_excel(writer, sheet_name="Model Accuracy", index=False)
cm_df.to_excel(writer, sheet_name="Confusion Matrix")
class_report_df.to_excel(writer, sheet_name="Classification Report")
```

Appendix II. ML and DL model Python scripts used for Campaign B.

Simple Logistic Regression model

```
import pandas as pd
import numpy as np
```

```
import statsmodels.api as sm
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
file_path = "D:\\CampaignB.xlsx"
df = pd.read_excel(file_path)
df = df.drop(columns=["ID", "Sent_time", "Chage_State_Time"])
df.rename(columns={"7or9": "Send_Hour"}, inplace=True)
X = df[["Sent_Day(/7)", "Send_Hour", "GNDR", "Client_Age",
"Life_Phase_cat", "balance_mean"]]
y = df["OPEN?"]
X["Send_Hour"] = X["Send_Hour"].astype(str)
X["GNDR"] = X["GNDR"].astype(str)
X["Life_Phase_cat"] = X["Life_Phase_cat"].astype(str)
categorical variables
preprocessor = ColumnTransformer(
transformers=[
("num", StandardScaler(), ["Client_Age", "balance_mean"]),
("cat", OneHotEncoder(drop="first"), ["Send_Hour", "GNDR",
"Life_Phase_cat"])
]
)
X_transformed = preprocessor.fit_transform(X)
X_transformed = pd.DataFrame(X_transformed,
columns=preprocessor.get_feature_names_out())
X_transformed.insert(0, "Intercept", 1)
model = sm.Logit(y, X_transformed)
result = model.fit()
summary_df = pd.DataFrame({
"Feature": X_transformed.columns,
"Coefficient": result.params,
"Std. Error": result.bse,
"Z value": result.tvalues,
"Pr (>|z|)": result.pvalues,
"Odds": np.exp(result.params)
})
X_train, X_test, y_train, y_test = train_test_split(X_transformed, y,
test_size=0.3, random_state=42)
y_pred = (result.predict(X_test) >= 0.5).astype(int)
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred, output_dict=True)
eval_results = pd.DataFrame({
"Metric": ["Accuracy"],
"Value": [accuracy]
})
output_file = "D:\\CampaignBlogre.xlsx"
with pd.ExcelWriter(output_file) as writer:
summary_df.to_excel(writer, sheet_name="Model Results", index=False)
eval_results.to_excel(writer, sheet_name="Evaluation", index=False)
```

```
pd.DataFrame(conf_matrix, columns=["Predicted 0", "Predicted 1"],
index=["Actual 0", "Actual 1"]).to_excel(writer, sheet_name="Confusion
Matrix")
pd.DataFrame(class_report).transpose().to_excel(writer,
sheet_name="Classification Report")
```

Pipeline Logistic Regression model

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
df = pd.read_excel("D:\\CampaignB.xlsx")
df_clean = df.drop(columns=['ID', 'Sent_time', 'Chage_State_Time'])
df_clean.rename(columns={
'Sent_Day(/7)': 'Sent_Day',
'7or9': '7or9',
'OPEN?': 'Opened',
'GNDR': 'Gender',
'Client_Age': 'Age',
'Life_Phase_cat': 'Life_Obligations',
'balance_mean': 'Balance'
}, inplace=True)
df_clean.dropna(subset=['Opened', 'Sent_Day', '7or9', 'Gender', 'Age',
'Life_Obligations', 'Balance'], inplace=True)
X = df_clean[['Sent_Day', '7or9', 'Gender', 'Age', 'Life_Obligations',
'Balance']]
y = df_clean['Opened']
preprocessor = ColumnTransformer(
transformers=[
('num', StandardScaler(), ['Sent_Day', 'Age', 'Balance']),
('cat', OneHotEncoder(), ['7or9', 'Gender', 'Life_Obligations'])
]
)
model = Pipeline(steps=[
('preprocessor', preprocessor),
('classifier', LogisticRegression(max_iter=1000, random_state=42))
])
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42, stratify=y)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred, output_dict=True)
class_report_df = pd.DataFrame(class_report).transpose()
results_df = pd.DataFrame({
"Metric": ["Accuracy"],
"Value": [accuracy]
```

```
})
conf_matrix_df = pd.DataFrame(conf_matrix,
columns=[f'Predicted {i}' for i in range(conf_matrix.shape[1])],
index=[f'Actual {i}' for i in range(conf_matrix.shape[0])])
save_path = "D:\\CampaignBpipe.xlsx"
with pd.ExcelWriter(save_path) as writer:
results_df.to_excel(writer, sheet_name="Model Results", index=False)
conf_matrix_df.to_excel(writer, sheet_name="Confusion Matrix")
class_report_df.to_excel(writer, sheet_name="Classification Report")
predictions_df = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
predictions_df.to_excel(writer, sheet_name="Predictions", index=False)
```

Random Forest (Hyperparameter)

```
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split,
RandomizedSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
import os
file_path = r"D:\\CampaignB.xlsx"
df = pd.read_excel(file_path)
df['GNDR_Sent_Day'] = df['GNDR'] * df['Sent_Day(/7)']
df['Balance_ClientAge'] = df['balance_mean'] * df['Client_Age']
scaler = StandardScaler()
df[['balance_mean', 'Client_Age']] =
scaler.fit_transform(df[['balance_mean', 'Client_Age']])
features = ['Sent_Day(/7)', 'balance_mean', 'Client_Age', 'GNDR',
'7or9',
'Life_Phase_cat', 'GNDR_Sent_Day', 'Balance_ClientAge']
target = 'OPEN?'
X = df[features]
y = df[target]
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42, stratify=y)
param_dist = {
'n_estimators': [50, 100, 200, 300],
'max_depth': [5, 10, 20, None],
'min_samples_split': [2, 5, 10],
'min_samples_leaf': [1, 2, 4],
'class_weight': [None, 'balanced', 'balanced_subsample']
}
rf = RandomForestClassifier(random_state=42)
rf_cv = RandomizedSearchCV(rf, param_dist, n_iter=10, cv=5,
scoring='accuracy', n_jobs=-1, random_state=42)
rf_cv.fit(X_train, y_train)
best_rf = rf_cv.best_estimator_
y_pred = best_rf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
```

```
class_report = classification_report(y_test, y_pred, output_dict=True)
feature_importance = pd.DataFrame({'Feature': features, 'Importance':
best_rf.feature_importances_})
feature_importance = feature_importance.sort_values(by='Importance',
ascending=False)
output_path = r"D:/CampaignBrandoforest3.xlsx"
with pd.ExcelWriter(output_path) as writer:
pd.DataFrame({'Metric': ['Accuracy'], 'Value':
[accuracy]}).to_excel(writer, sheet_name='Evaluation', index=False)
feature_importance.to_excel(writer, sheet_name='Feature Importance',
index=False)
pd.DataFrame(conf_matrix, columns=['Predicted 0', 'Predicted 1'],
index=['Actual 0', 'Actual 1']).to_excel(writer, sheet_name='Confusion
Matrix')
pd.DataFrame(class_report).transpose().to_excel(writer,
sheet_name='Classification Report')
```

XGBoost (Standard-Scaler)

```
import pandas as pd
import numpy as np
import xgboost as xgb
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
from sklearn.preprocessing import StandardScaler
file_path = r"D:\\CampaignB.xlsx"
df = pd.read_excel(file_path)
features = ["Sent_Day(/7)", "7or9", "GNDR", "Client_Age",
"Life_Phase_cat", "balance_mean"]
target = "OPEN?"
df["Balance_ClientAge"] = df["balance_mean"] * df["Client_Age"]
df["GNDR_Sent_Day"] = df["GNDR"] * df["Sent_Day(/7)"]
df["Balance_Sent_Day"] = df["balance_mean"] * df["Sent_Day(/7)"]
selected_features = features + ["Balance_ClientAge", "GNDR_Sent_Day",
"Balance_Sent_Day"]
X = df[selected_features]
y = df[target]
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=42, stratify=y)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
model = xgb.XGBClassifier(objective="binary:logistic",
eval_metric="logloss", use_label_encoder=False)
model.fit(X_train_scaled, y_train)
y_pred = model.predict(X_test_scaled)
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred, output_dict=True)
feature_importance = pd.DataFrame({"Feature": selected_features,
"Importance": model.feature_importances_})
```

```
feature_importance = feature_importance.sort_values(by="Importance",
ascending=False)
output_path = "D:/CampaignBXGBoost.xlsx"
with pd.ExcelWriter(output_path) as writer:
feature_importance.to_excel(writer, sheet_name="Feature Importance",
index=False)
pd.DataFrame({"Metric": ["Accuracy"], "Value":
[accuracy]}).to_excel(writer, sheet_name="Evaluation", index=False)
pd.DataFrame(conf_matrix, index=["Actual 0", "Actual 1"],
columns=["Predicted 0", "Predicted 1"]).to_excel(writer,
sheet_name="Confusion Matrix")
pd.DataFrame(class_report).transpose().to_excel(writer,
sheet_name="Classification Report")
```

LightGBM

```
import pandas as pd
import numpy as np
import shap
import matplotlib.pyplot as plt
import seaborn as sns
from lightgbm import LGBMClassifier
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
file_path = "D:/CampaignBlite.xlsx"
df = pd.read_excel(file_path)
for col in df.select_dtypes(include=["datetime64"]):
print(f"Dropping datetime column: {col}")
df.drop(columns=[col], inplace=True)
categorical_cols = ["GNDR", "7or9", "Life_Phase_cat"]
for col in categorical_cols:
df[col] = pd.to_numeric(df[col], errors="coerce").fillna(0).astype(int)
df["Balance+Sent_Day"] = df["balance_mean"] * df["Sent_Day(/7)"]
df["GNDR+Sent_Day"] = df["GNDR"] * df["Sent_Day(/7)"]
df["Balance+ClientAge"] = df["balance_mean"] * df["Client_Age"]
X = df.drop(columns=["OPEN?"])
y = df["OPEN?"]
print("Data types after fixes:\n", X.dtypes)
X_train, X_test, y_train, y_test = train_test_split(
X, y, test_size=0.2, random_state=42, stratify=y
)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
lgbm = LGBMClassifier(verbose=-1, random_state=42)
param_grid = {
"n_estimators": [50, 100, 200],
"max_depth": [3, 5, 7],
"learning_rate": [0.01, 0.05, 0.1]
}
```

```
grid = GridSearchCV(lgbm, param_grid, cv=5, scoring="accuracy",
verbose=1, n_jobs=-1)
grid.fit(X_train_scaled, y_train)
best_model = grid.best_estimator_
y_pred = best_model.predict(X_test_scaled)
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.4f}")
conf_matrix = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred, output_dict=True)
sns.heatmap(
conf_matrix, annot=True, fmt="d", cmap="Blues",
xticklabels=["Not Opened", "Opened"],
yticklabels=["Not Opened", "Opened"]
)
output_path = "D:/campbLightGBM.xlsx"
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
explainer = shap.TreeExplainer(best_model)
shap_values = explainer.shap_values(X_test_scaled)
shap.summary_plot(shap_values, X_test, feature_names=X.columns)
feature_importance = pd.DataFrame({
"Feature": X.columns,
"Importance": best_model.feature_importances_
}).sort_values(by="Importance", ascending=False)
with pd.ExcelWriter(output_path) as writer:
feature_importance.to_excel(writer, sheet_name="Feature Importance",
index=False)
pd.DataFrame({"Metric": ["Accuracy"], "Value":
[accuracy]}).to_excel(writer, sheet_name="Evaluation", index=False)
pd.DataFrame(conf_matrix, index=["Actual 0", "Actual 1"],
columns=["Predicted 0", "Predicted 1"]).to_excel(writer,
sheet_name="Confusion Matrix")
pd.DataFrame(class_report).transpose().to_excel(writer,
sheet_name="Classification Report")
```

NeuralNetwork

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neural_network import MLPClassifier
from sklearn.ensemble import RandomForestClassifier, StackingClassifier
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
from sklearn.inspection import permutation_importance
from imblearn.over_sampling import SMOTE
from sklearn.pipeline import Pipeline
```

```
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import PolynomialFeatures
file_path = "D:/CampaignBlite.xlsx"
df = pd.read_excel(file_path)
datetime_cols = df.select_dtypes(include=["datetime64"]).columns
if len(datetime_cols) > 0:
print(f"Datetime columns found: {list(datetime_cols)} - Converting to
numerical format.")
for col in datetime_cols:
df[col] = df[col].astype(np.int64) // 10**9
convert_cols = ["GNDR", "7or9", "Life_Phase_cat"]
for col in convert_cols:
if col in df.columns and not np.issubdtype(df[col].dtype,
np.datetime64):
df[col] = df[col].astype(int)
df["Balance+Sent_Day"] = df["balance_mean"] * df["Sent_Day(/7)"]
df["GNDR+Sent_Day"] = df["GNDR"] * df["Sent_Day(/7)"]
df["Balance+ClientAge"] = df["balance_mean"] * df["Client_Age"]
df["Is_Weekend"] = df["Sent_Day(/7)"].apply(lambda x: 1 if x in [6, 7]
else 0)
X = df.drop(columns=["OPEN?"])
y = df["OPEN?"]
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42, stratify=y)
smote = SMOTE(random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train,
y_train)
preprocessor = ColumnTransformer([
("poly", PolynomialFeatures(degree=2, interaction_only=True),
["balance_mean", "Client_Age"])
], remainder="passthrough")
scaler = StandardScaler()
X_train_scaled =
scaler.fit_transform(preprocessor.fit_transform(X_train_resampled))
X_test_scaled = scaler.transform(preprocessor.transform(X_test))
rf = RandomForestClassifier(n_estimators=200, max_depth=7,
random_state=42)
xgb = XGBClassifier(n_estimators=200, learning_rate=0.1, max_depth=5,
random_state=42, use_label_encoder=False, eval_metric='logloss')
nn = MLPClassifier(hidden_layer_sizes=(128, 64, 32), activation='relu',
solver='adam', max_iter=1000, random_state=42)
stacking_clf = StackingClassifier(
estimators=[("rf", rf), ("xgb", xgb), ("nn", nn)],
final_estimator=RandomForestClassifier(n_estimators=100,
random_state=42)
)
stacking_clf.fit(X_train_scaled, y_train_resampled)
y_pred = stacking_clf.predict(X_test_scaled)
accuracy = accuracy_score(y_test, y_pred)
cm = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred, output_dict=True)
print(f"Accuracy: {accuracy:.4f}")
```

```
print("\nClassification Report:\n", classification_report(y_test,
y_pred))
plt.figure(figsize=(5,5))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=["Not
Opened", "Opened"], yticklabels=["Not Opened", "Opened"])
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
perm_importance = permutation_importance(stacking_clf, X_test_scaled,
y_test, scoring='accuracy', n_repeats=10, random_state=42)
feature_importance = pd.DataFrame({
"Feature": preprocessor.get_feature_names_out(),
"Importance (%)": (perm_importance.importances_mean /
perm_importance.importances_mean.sum()) * 100
}).sort_values(by="Importance (%)", ascending=False)
results_df = pd.DataFrame({
"Metric": ["Accuracy"],
"Value": [accuracy]
})
class_report_df = pd.DataFrame(class_report).transpose()
cm_df = pd.DataFrame(cm, index=["Actual_Not_Opened", "Actual_Opened"],
columns=["Predicted_Not_Opened", "Predicted_Opened"])
output_path = "D:/CampaignB_FeatureImportance_Stacking.xlsx"
with pd.ExcelWriter(output_path) as writer:
feature_importance.to_excel(writer, sheet_name="Feature Importance",
index=False)
results_df.to_excel(writer, sheet_name="Model Accuracy", index=False)
cm_df.to_excel(writer, sheet_name="Confusion Matrix")
class_report_df.to_excel(writer, sheet_name="Classification Report")
```

Neural Network + XGBoost + Random Forest

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neural_network import MLPClassifier
from sklearn.ensemble import RandomForestClassifier, StackingClassifier
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
from sklearn.inspection import permutation_importance
from imblearn.over_sampling import SMOTE
from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import PolynomialFeatures
file_path = "D:/CampaignBlite.xlsx"
df = pd.read_excel(file_path)
datetime_cols = df.select_dtypes(include=["datetime64"]).columns
df = df.drop(columns=datetime_cols)
```

```
df["GNDR"] = df["GNDR"].astype(int)
df["7or9"] = df["7or9"].astype(int)
df["Life_Phase_cat"] = df["Life_Phase_cat"].astype(int)
df["Balance+Sent_Day"] = df["balance_mean"] * df["Sent_Day(/7)"]
df["GNDR+Sent_Day"] = df["GNDR"] * df["Sent_Day(/7)"]
df["Balance+ClientAge"] = df["balance_mean"] * df["Client_Age"]
df["Is_Weekend"] = df["Sent_Day(/7)"].apply(lambda x: 1 if x in [6, 7]
else 0)
X = df.drop(columns=["OPEN?"])
y = df["OPEN?"]
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42, stratify=y)
smote = SMOTE(random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train,
y_train)
preprocessor = ColumnTransformer([
("poly", PolynomialFeatures(degree=2, interaction_only=True),
["balance_mean", "Client_Age"])
], remainder="passthrough")
scaler = StandardScaler()
X_train_scaled =
scaler.fit_transform(preprocessor.fit_transform(X_train_resampled))
X_test_scaled = scaler.transform(preprocessor.transform(X_test))
rf = RandomForestClassifier(n_estimators=200, max_depth=7,
random_state=42)
xgb = XGBClassifier(n_estimators=200, learning_rate=0.1, max_depth=5,
random_state=42, use_label_encoder=False, eval_metric='logloss')
nn = MLPClassifier(hidden_layer_sizes=(128, 64, 32), activation='relu',
solver='adam', max_iter=1000, random_state=42)
stacking_clf = StackingClassifier(
estimators=[("rf", rf), ("xgb", xgb), ("nn", nn)],
final_estimator=RandomForestClassifier(n_estimators=100,
random_state=42)
)
stacking_clf.fit(X_train_scaled, y_train_resampled)
y_pred = stacking_clf.predict(X_test_scaled)
accuracy = accuracy_score(y_test, y_pred)
cm = confusion_matrix(y_test, y_pred)
class_report = classification_report(y_test, y_pred, output_dict=True)
plt.figure(figsize=(5,5))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=["Not
Opened", "Opened"], yticklabels=["Not Opened", "Opened"])
plt.title("Confusion Matrix")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.show()
perm_importance = permutation_importance(stacking_clf, X_test_scaled,
y_test, scoring='accuracy', n_repeats=10, random_state=42)
feature_importance = pd.DataFrame({
"Feature": preprocessor.get_feature_names_out(),
"Importance (%)": (perm_importance.importances_mean /
perm_importance.importances_mean.sum()) * 100
```

```
}).sort_values(by="Importance (%)", ascending=False)
results_df = pd.DataFrame({
"Metric": ["Accuracy"],
"Value": [accuracy]
})
class_report_df = pd.DataFrame(class_report).transpose()
cm_df = pd.DataFrame(cm, index=["Actual_Not_Opened", "Actual_Opened"],
columns=["Predicted_Not_Opened", "Predicted_Opened"])
output_path = "D:/CampaignBNNXGBRF.xlsx"
with pd.ExcelWriter(output_path) as writer:
feature_importance.to_excel(writer, sheet_name="Feature Importance",
index=False)
results_df.to_excel(writer, sheet_name="Model Accuracy", index=False)
cm_df.to_excel(writer, sheet_name="Confusion Matrix")
class_report_df.to_excel(writer, sheet_name="Classification Report")
```

Appendix III. Campaign A database exemplified

Hard to r

ID	Sent_time	Sent_Day(/7)	Morning_Evening	Chage_State_Time	OPEN?	GNDR	Client_Age	life_phase	Life_Phase_cat	balance_mean	balance	balance_range
1	12/9/24 9:03	1	0	12/9/24 23:01	0	0	61	5	1	3750	4887.76	2500-5000
2	12/9/24 9:03	1	0	12/10/24 23:00	0	1	56	5	1	25000	29052.32	20000-30000
3	12/9/24 9:03	1	0	12/9/24 13:37	1	0	68	6	1	100000	181980.59	100000+
4	12/9/24 9:03	1	0	12/11/24 7:02	0	0	72	6	1	17500	18133.24	15000-20000
5	12/9/24 9:03	1	0	1/10/25 2:02	1	1	78	6	1	87500	98091.67	75000-100000
6	12/9/24 9:03	1	0	12/13/24 12:00	1	0	70	6	1	8750	7900.93	7500-10000
7	12/9/24 9:03	1	0	12/14/24 6:11	1	0	43	5	1	-1	-42392.18	debt
8	12/9/24 9:03	1	0	12/12/24 2:01	0	0	77	6	1	100000	545102.91	100000+
9	12/9/24 9:03	1	0	12/10/24 0:01	0	0	67	6	1	62500	63334.53	50000-75000
10	12/9/24 9:03	1	0	12/11/24 1:01	0	1	45	4	1	3750	3712.03	2500-5000
11	12/9/24 9:03	1	0	12/12/24 6:06	1	0	70	6	1	25000	27032.95	20000-30000
12	12/9/24 23:01	1	1	12/12/24 5:02	0	1	69	6	1	62500	54086.05	50000-75000
13	12/9/24 23:01	1	1	12/11/24 9:03	0	0	70	6	1	100000	131391.95	100000+
14	12/9/24 23:01	1	1	12/10/24 8:02	0	1	72	6	1	100000	248736.23	100000+
15	12/9/24 23:01	1	1	12/11/24 9:03	0	0	38	2	0	-1	-191870.46	debt
16	12/9/24 23:01	1	1	12/9/24 23:01	0	0	49	4	1	87500	98555.40	75000-100000
17	12/9/24 23:01	1	1	12/11/24 5:02	0	1	66	6	1	12500	12803.18	10000-15000
18	12/9/24 23:01	1	1	12/11/24 5:02	0	1	61	5	1	17500	17660.27	15000-20000
19	12/9/24 23:01	1	1	12/11/24 23:00	0	1	49	5	1	25000	21429.08	20000-30000
20	12/9/24 23:01	1	1	12/13/24 14:55	1	1	36	4	1	62500	57061.57	50000-75000
21	12/9/24 23:01	1	1	12/12/24 2:52	1	0	73	6	1	12500	14780.48	10000-15000
22	12/9/24 23:01	1	1	12/12/24 2:48	1	0	49	5	1	17500	19433.19	15000-20000
23	12/9/24 23:01	1	1	12/10/24 5:02	0	1	78	6	1	12500	13416.86	10000-15000
24	12/9/24 23:01	1	1	12/13/24 3:30	1	0	74	6	1	25000	25178.50	20000-30000
25	12/9/24 23:01	1	1	12/18/24 12:05	1	0	73	6	1	45000	48627.96	40000-50000
26	12/9/24 23:01	1	1	12/11/24 23:00	0	1	56	5	1	25000	21538.70	20000-30000
27	12/9/24 23:01	1	1	12/16/24 23:26	1	0	62	5	1	25000	21310.11	20000-30000
28	12/9/24 23:01	1	1	12/10/24 4:02	0	0	34	3	0	17500	16980.17	15000-20000
29	12/9/24 23:01	1	1	12/11/24 23:00	0	1	53	4	1	100000	167254.34	100000+
30	12/9/24 23:01	1	1	12/11/24 6:02	0	0	65	5	1	87500	94932.71	75000-100000
31	12/9/24 23:01	1	1	12/9/24 23:01	0	1	73	6	1	17500	17261.02	15000-20000
32	12/9/24 23:01	1	1	12/11/24 1:48	1	1	68	6	1	100000	111551.31	100000+
33	12/9/24 23:01	1	1	12/12/24 15:56	1	0	49	5	1	87500	91335.56	75000-100000
34	12/9/24 23:01	1	1	12/19/24 2:45	1	0	75	6	1	8750	7711.70	7500-10000
35	12/9/24 23:01	1	1	12/10/24 6:02	0	1	60	5	1	35000	39840.98	30000-40000
36	12/9/24 23:01	1	1	1/13/25 2:44	1	1	73	6	1	35000	36811.31	30000-40000
37	12/9/24 23:01	1	1	12/11/24 6:19	1	0	73	6	1	12500	12706.34	10000-15000
38	12/9/24 23:01	1	1	12/12/24 8:02	0	1	64	5	1	6250	6972.33	5000-7500
39	12/9/24 23:01	1	1	12/11/24 6:57	1	1	76	6	1	100000	168845.82	100000+
40	12/9/24 23:01	1	1	12/11/24 23:00	0	1	88	6	1	62500	60731.03	50000-75000
41	12/9/24 23:01	1	1	12/12/24 9:52	1	0	61	5	1	25000	22125.62	20000-30000
42	12/9/24 23:01	1	1	12/12/24 8:03	1	0	57	5	1	45000	40495.97	40000-50000
43	12/9/24 23:01	1	1	12/11/24 23:00	0	0	55	5	1	17500	16996.06	15000-20000
44	12/9/24 23:01	1	1	12/11/24 23:00	0	0	45	4	1	100000	258031.47	100000+
45	12/9/24 23:01	1	1	12/27/24 9:21	1	0	52	5	1	62500	72168.29	50000-75000
46	12/9/24 23:01	1	1	12/11/24 9:13	1	1	78	6	1	25000	23953.82	20000-30000
47	12/9/24 23:01	1	1	12/12/24 2:45	1	1	55	5	1	100000	114773.04	100000+
48	12/9/24 23:01	1	1	12/12/24 1:01	0	1	60	5	1	12500	10778.90	10000-15000
49	12/9/24 23:01	1	1	12/21/24 9:55	1	0	50	5	1	17500	18924.00	15000-20000
50	12/9/24 23:01	1	1	12/11/24 13:04	1	1	46	5	1	62500	61503.85	50000-75000
51	12/9/24 23:01	1	1	12/12/24 2:01	0	0	74	6	1	17500	17461.61	15000-20000

Appendix IV. Campaign B database exemplified

ID	Sent_time	Sent_Day(/7)	7or9	Chage_State_Time	OPEN?	GNDR	Client_Age	life_phase	Life_Phase_cat	balance_mean	balance	balance_range
1	12/2/24 7:30	1	7	12/3/24 9:11	1	0	18	3	0	100000	107856.26	100000+
2	12/2/24 7:30	1	7	12/16/24 7:46	1	1	18	3	0	62500	58422.44	50000-75000
3	12/2/24 7:30	1	7	12/5/24 11:25	1	1	18	6	1	87500	75625.89	75000-100000
4	12/2/24 7:30	1	7	12/3/24 10:47	1	1	18	4	1	100000	116478.21	100000+
5	12/2/24 7:30	1	7	12/4/24 2:45	1	1	18	3	0	100000	207429.35	100000+
6	12/2/24 7:30	1	7	12/22/24 8:34	1	1	18	3	0	100000	241948.38	100000+
7	12/2/24 7:30	1	7	12/3/24 16:03	1	0	18	3	0	12500	14218.49	10000-15000
8	12/2/24 7:30	1	7	12/3/24 9:58	1	0	18	4	1	100000	450784.11	100000+
9	12/2/24 7:30	1	7	12/20/24 11:41	1	1	18	6	1	87500	91733.47	75000-100000
10	12/2/24 7:30	1	7	12/27/24 7:32	1	0	18	3	0	100000	301479.64	100000+
11	12/2/24 7:30	1	7	12/27/24 9:21	1	1	18	4	1	25000	25436.36	20000-30000
12	12/2/24 7:30	1	7	12/16/24 8:56	1	1	18	6	1	12500	10877.14	10000-15000
13	12/2/24 7:30	1	7	12/3/24 10:36	1	1	18	4	1	12500	13613.38	10000-15000
14	12/2/24 7:30	1	7	12/4/24 5:23	1	0	18	5	1	3750	3133.47	2500-5000
15	12/2/24 7:30	1	7	12/12/24 9:03	0	1	18	4	1	17500	19544.3	15000-20000
16	12/2/24 7:30	1	7	12/3/24 18:46	1	1	18	4	1	17500	17390.98	15000-20000
17	12/2/24 7:30	1	7	12/3/24 9:03	0	0	18	6	1	45000	44721.35	40000-50000
18	12/2/24 7:30	1	7	12/3/24 9:03	0	0	18	3	0	12500	13109.15	10000-15000
19	12/2/24 7:30	1	7	12/3/24 22:42	1	1	18	4	1	25000	21538.05	20000-30000
20	12/2/24 7:30	1	7	12/3/24 9:23	1	0	18	3	0	100000	312117.81	100000+
21	12/2/24 7:30	1	7	12/16/24 15:36	1	1	18	2	0	100000	724881147.7	100000+
22	12/2/24 7:30	1	7	12/3/24 9:03	0	1	18	4	1	1750	1195.86	1000-2500
23	12/2/24 7:30	1	7	12/3/24 9:03	0	1	18	4	1	6250	7386.69	5000-7500
24	12/2/24 7:30	1	7	12/3/24 9:03	0	0	18	5	1	100000	339601.36	100000+
25	12/2/24 7:30	1	7	12/3/24 9:03	0	1	18	4	1	25000	25809.7	20000-30000
26	12/2/24 7:30	1	7	12/7/24 9:00	1	0	18	3	0	100000	135529.55	100000+
27	12/2/24 7:30	1	7	12/3/24 10:26	1	0	18	6	1	8750	8814.25	7500-10000
28	12/2/24 7:30	1	7	12/4/24 3:16	1	1	18	1	0	100000	3457573.18	100000+
29	12/2/24 7:30	1	7	12/29/24 4:22	1	1	18	6	1	87500	77616.14	75000-100000
30	12/2/24 7:30	1	7	12/10/24 1:57	1	0	18	6	1	25000	20392.54	20000-30000
31	12/2/24 7:30	1	7	12/6/24 9:03	0	0	18	5	1	-1	-9460.9	debt
32	12/2/24 7:30	1	7	12/9/24 8:01	1	1	18	4	1	3750	2765.69	2500-5000
33	12/2/24 7:30	1	7	12/3/24 9:53	1	1	18	3	0	100000	169993.2	100000+
34	12/2/24 7:30	1	7	12/3/24 9:03	0	0	18	5	1	100000	248778.9	100000+
35	12/2/24 7:30	1	7	12/12/24 1:20	1	0	18	5	1	62500	63730.96	50000-75000
36	12/2/24 7:30	1	7	12/10/24 9:15	1	0	18	3	0	6250	7124.97	5000-7500
37	12/2/24 7:30	1	7	12/9/24 11:08	1	0	18	1	0	100000	1088693171	100000+
38	12/2/24 7:30	1	7	12/3/24 9:03	0	1	18	6	1	100000	205418.8	100000+
39	12/2/24 7:30	1	7	12/9/24 15:52	1	1	19	5	1	45000	41844.07	40000-50000
40	12/2/24 7:30	1	7	12/18/24 7:53	1	1	19	4	1	45000	40892.97	40000-50000
41	12/2/24 7:30	1	7	12/12/24 11:11	1	0	19	5	1	100000	117437.21	100000+
42	12/2/24 7:30	1	7	12/7/24 11:57	1	1	19	5	1	8750	9823.1	7500-10000
43	12/2/24 7:30	1	7	12/3/24 9:03	0	1	19	3	0	100000	1440751073	100000+
44	12/2/24 7:30	1	7	12/5/24 9:03	0	0	19	3	0	100000	65864831.9	100000+
45	12/2/24 7:31	1	7	12/5/24 9:03	0	0	19	4	1	100000	211854.33	100000+
46	12/2/24 7:31	1	7	12/3/24 9:03	0	0	19	2	0	100000	394605.85	100000+
47	12/2/24 7:31	1	7	12/8/24 1:33	1	0	19	3	0	100000	315864.01	100000+
48	12/2/24 7:31	1	7	12/3/24 9:03	0	1	19	4	1	6250	5174.31	5000-7500
49	12/2/24 7:31	1	7	12/4/24 0:24	1	0	19	5	1	8750	9409.62	7500-10000
50	12/2/24 7:31	1	7	12/3/24 9:03	0	0	19	3	0	100000	748199.54	100000+
51	12/2/24 7:31	1	7	12/10/24 3:48	1	0	19	5	1	87500	98122.58	75000-100000