A Machine Learning Approach for Targeting Lucrative Customer Segments

Leen Almajed[1], Haifa Almakhdhoub[2], Jana Alzeydan[3], Alaa Bin Sleem[4], Ouiem Bchir[5]
TAHAKOM, Riyadh, Saudi Arabia[1,2]
College of Computer and Information Sciences-Computer Science Department,
King Saud University, Riyadh, Saudi Arabia[1,2,3,4,5]

***Abstract*—Effective marketing plays a pivotal role in modern businesses, where strategic allocation of resources is essential for maximizing return on investment (ROI). Customer segmentation involves dividing users into distinct sub-groups based on common characteristics, enabling each segment to receive tailored promotions according to their behavior. However, identifying which customer segments to target can be financially challenging due to the significant costs associated with marketing campaigns. This study proposes a machine learning framework to forecast the profitability of various customer segments and optimize marketing strategies accordingly. The proposed solution leverages machine learning and deep learning techniques to classify customers based on their potential value. Specifically, conventional classifiers including AdaBoost, Gradient Boosting, Extreme Gradient Boosting (XGBoost), Linear Discriminant Analysis (LDA), Random Forest, Naïve Bayes, Support Vector Machines (SVM), and K-Nearest Neighbors (KNN) are evaluated. In addition, deep learning models, namely 1D ResNet and 1D DenseNet, are investigated. All models are trained and evaluated under a unified protocol that includes SMOTE-based class imbalance handling, systematic hyperparameter tuning, and a joint analysis of predictive performance and computational cost. The experimental findings reveal that traditional models, particularly XGBoost and Gradient Boosting, consistently outperform deep learning models in terms of accuracy, precision, and computational efficiency, with both achieving the highest weighted F1 score of 0.62 while requiring nearly six orders of magnitude fewer computations than DenseNet-121. These results provide concrete evidence that ensemble tree-based methods are better suited than deep architectures for moderately sized, imbalanced, tabular marketing datasets.**

Keywords*—Return on investment; customer segmentation; machine learning; deep learning; profitability prediction; tabular data; Gradient Boosting; XGBoost

I. Introduction

As the business environment grows more competitive, companies increasingly leverage data-driven targeted marketing strategies to drive financial growth [1]. Targeted marketing identifies a segment of people with similar needs or characteristics that a company chooses to serve, allowing messages and promotions to speak directly to the interests of each segment rather than relying on a one-size-fits-all approach [2]. Customer segmentation that incorporates psychological and behavioral dimensions enables businesses to tailor their marketing efforts and increase customer retention by aligning campaigns with customer preferences [3]. This approach improves the relevance and effectiveness of marketing efforts, resulting in higher engagement and conversion rates. By focusing on segments with the highest growth potential or profitability, companies can prioritize their marketing budgets, product development initiatives, and sales efforts to maximize return on investment [2]. The integration of data analytics techniques, such as machine learning and predictive modeling, allows these strategies to be refined further to meet specific market needs [4].

Despite these advantages, many companies still face significant challenges in implementing targeted marketing effectively. In practice, some organizations continue to rely on mass or untargeted approaches, which can lead to substantial financial loss when resources are spent reaching individuals who are uninterested in the products on offer [5]. The result is a wasted budget and a diminished return on investment (ROI), since the lack of focus prevents companies from engaging the segments most likely to respond. Although customer segmentation partially addresses this issue, accurately determining the profitability of each segment under a given marketing strategy remains difficult. This difficulty arises from the complexity of predicting how different customer groups will respond to campaigns and how those responses translate into financial outcomes. Traditional customer targeting techniques often rely on simplistic criteria such as demographic categories or basic purchasing metrics, which may not capture the complex patterns within customer data that determine profitability. The interactions between customer attributes, for example how the combination of income level, purchase frequency, and geographic location jointly influences campaign response, are inherently non-linear and difficult to model using rule-based approaches. Consequently, there is a clear need for an intelligent, data-driven approach capable of forecasting the profitability of different customer segments.

To address these challenges, machine learning offers a promising solution by analyzing large-scale customer data to uncover patterns that support accurate profitability prediction. By examining demographic, geographic, behavioral, and psychographic attributes, machine learning models provide actionable insights that enable businesses to allocate resources toward high-value segments and optimize marketing strategies, ultimately enhancing ROI and long-term customer relationship management (CRM). The ability of machine learning to process high-dimensional data and identify non-obvious relationships makes it particularly valuable for profitability prediction, where the interplay between multiple customer

This work was conducted while Leen Almajed and Haifa Almakhdhoub were affiliated with the Computer Science Department, College of Computer and Information Sciences, King Saud University.

attributes determines segment-level outcomes. Furthermore, the availability of diverse algorithmic approaches, ranging from interpretable linear models to complex deep learning architectures, enables practitioners to select models that best match the characteristics of their specific datasets and business requirements.

Building on this approach, this study proposes a machine learning framework for predicting customer segment profitability using historical customer data and advanced classification techniques. The framework is designed to support strategic decision-making and efficient resource allocation. To evaluate performance and robustness, multiple conventional classifiers are compared, including AdaBoost [6], Gradient Boosting [7], XGBoost [8], LDA [9], Random Forest [10], Naïve Bayes [11], SVM [12], and KNN [13]. Additionally, 1D deep learning architectures, namely ResNet [14] and DenseNet [15], are assessed to determine their effectiveness in predicting segment-level profitability.

The main contributions of this work are as follows: 1) a unified empirical benchmark of eight classical classifiers and two 1D deep learning architectures (1D ResNet and 1D DenseNet) applied to customer segment profitability prediction on a previously unexplored, moderately sized, imbalanced tabular dataset; 2) a domain-specific contribution to the marketing analytics literature showing that, contrary to several recent reports favoring deep learning for customer prediction [16], [17], [18], traditional ensemble methods outperform 1D deep architectures when applied to comparative two-group customer features under realistic data-volume and class-imbalance conditions; 3) a detailed computational cost analysis that quantifies the trade-off between predictive performance and resource requirements across all evaluated models, providing the GFLOPs and parameter counts needed to assess deployment feasibility in production marketing systems; and 4) practical recommendations for practitioners on model selection, hyperparameter behavior under class imbalance, and the limited gains obtainable from autoencoder-based feature compression or One-vs-Rest decomposition on this class of data.

II. Related Work

A. Traditional Machine Learning

Several studies applied traditional machine learning methods to predict customer behavior. Deniz and Bülbül [19] applied six models, including Logistic Regression [20], Random Forest [10], and Gradient Boosting [7] to predict purchasing behavior using attributes like age, gender, annual income, and number of purchases. A significant improvement was achieved through segmenting customers into high, medium, and low spenders. Random Forest achieved the highest accuracy at 94.4%, outperforming Logistic Regression at 82.2%.

Other studies [21], [22] showed that Random Forest outperformed other models achieving 90.4% and 88.63% accuracy, respectively, when predicting customer subscription behaviors and churn by analyzing features like call duration, contact frequency, and customer behavioral metrics.

Extending the approach to customer profitability prediction, Chen et al. [23] introduced a dynamic approach using the Recency, Frequency, and Monetary (RFM) model. Customers were segmented into profitability groups using k-means clustering, and models including regression, multilayer perceptron (MLP), and Naïve Bayes were applied. Open-loop models achieved an average accuracy of 84%, while closed-loop models reached 79%. The study highlighted that customer profitability groups remained largely stable over time, with transition probabilities below 6%.

Saeed et al. [24] employed an ensemble classifier combining Random Forest and Extra Trees to predict customer responses to a banking telemarketing campaign. The hybrid voting classifier improved accuracy by 1.6% over the best individual model, Random Forest, which achieved 94.02%.

In related studies, the gradient boosting decision tree (GBDT) demonstrated strong performance. Sun et al. [25] applied GBDT and Random Forest to predict Customer Lifetime Value (CLV), achieving mean square errors of 0.65 and 0.34, respectively, outperforming probability-based models. Similarly, Sun et al. [26] compared eight machine learning algorithms for customer segmentation based on lifecycle value, where AdaBoost achieved 94.60% accuracy and GBDT reached 93.80%.

Xiahou and Harada [27] utilized AdaBoost for churn prediction, evaluating it against a Backpropagation (BP) neural network. The AdaBoost model outperformed the BP neural network across all metrics, achieving accuracy, recall, and precision values of 0.96, 0.93, and 0.96, respectively. Mishra and Sharma [28] further validated AdaBoost for fraud detection, showing its ability to handle imbalanced datasets effectively.

Wong and Yeh [29] proposed hybrid classification algorithms that combine instance filtering with ensemble methods, demonstrating improved accuracy over standalone classifiers on customer behavior datasets. Uddin et al. [30] compared multiple supervised algorithms, including SVM, KNN, and Random Forest, finding that ensemble methods consistently outperformed individual classifiers for structured prediction tasks. Musa [31] conducted a comparative study between SVM and Logistic Regression, concluding that SVM achieves superior classification performance on high-dimensional customer datasets.

B. Deep Learning

Deep learning models have been increasingly adopted for customer prediction tasks. Xu et al. [16] employed a combination of ResNet-50 [14] and Gated Recurrent Unit (GRU) [32] to predict consumer behavior by combining visual and temporal data, achieving a prediction accuracy of 96.18%.

Keji et al. [17] applied a ResNet model to banking data [33] with Synthetic Minority Oversampling Technique (SMOTE) for class imbalance. ResNet achieved 93% accuracy, outperforming Random Forest at 90.78%. Nabi et al. [18] focused on profit prediction in e-commerce using deep learning. VGG16 [34] and ResNet50 [14] were compared to traditional machine learning algorithms. VGG16 emerged as the top-performing model with 92.55% accuracy.

Chaudhuri et al. [35] compared a Feed-forward Deep Neural Network (DNN) to traditional models for predicting customer purchase behavior on e-commerce platforms. DNN

outperformed all traditional algorithms with an accuracy of 89%. Singh et al. [36] compared MLP [37] to decision tree J48 and Naïve Bayes classifiers on a shopping center dataset for Customer Relationship Management (CRM). MLP outperformed both classifiers with an accuracy of 98.33%.

Dong et al. [38] proposed an improved 1D ResNet architecture for structured data classification, demonstrating that residual connections are effective even in non-image domains. Wang et al. [39] established that deep neural networks, including ResNet and fully convolutional networks provide strong baselines for time series and structured data classification, often matching or exceeding traditional approaches without extensive feature engineering.

While deep learning models excel with multi-modal and unstructured data, their application to structured tabular data remains less common. Traditional models are often preferred for their interpretability and suitability for smaller structured datasets. Recent benchmark studies have shown that gradient boosting methods frequently outperform deep learning on tabular data, particularly when datasets contain fewer than 10,000 instances. This study uses a dataset not previously explored in the literature and focuses on an empirical comparison of both paradigms for customer segment profitability prediction, providing evidence for the relative strengths of each approach under conditions of moderate dataset size and class imbalance.

III. Proposed Approach

This study proposes a machine learning framework for customer profitability prediction. The input to this framework is segmented historical customer data, structured based on customer attributes. This data undergoes a preprocessing pipeline that includes: 1) feature standardization to normalize input variables, ensuring that features with different scales do not disproportionately influence model training, 2) application of the Synthetic Minority Over-sampling Technique (SMOTE) to address class imbalance in the training set by generating synthetic instances of the minority class, and 3) for deep learning models, reshaping of input features to match the required format for 1D convolutional architectures. SMOTE is applied only to the training set to prevent data leakage, ensuring that the validation and test sets reflect the true class distribution.

The framework follows two main pathways. The first pathway trains and evaluates conventional machine learning models, including AdaBoost [6], Gradient Boosting [7], Extreme Gradient Boosting [8], Linear Discriminant Analysis [9], Random Forest [10], Naïve Bayes [11], Support Vector Machines [12], and K-Nearest Neighbors [13]. These models are trained using 10-fold cross-validation, where the data is partitioned into ten equal subsets, and each model is trained on nine folds while being validated on the remaining fold, rotating across all ten combinations. This strategy provides a robust estimate of model performance by reducing the variance associated with a single train-test split. The second pathway trains deep learning models, specifically 1D ResNet [14] and 1D DenseNet [15]. For deep learning, the dataset is split into 60% training, 20% validation, and 20% testing, with early stopping employed to halt training when validation performance ceases to improve, thereby preventing overfitting.

Following model training and testing, each model undergoes evaluation to assess performance. Based on the evaluation results, a final decision is made to select the best-performing model. The chosen model is then used to predict the profitability of each customer segment, providing actionable insights for targeting high-value customer segments and optimizing marketing resources.

The motivation for including conventional ML models stems from their advantages with structured datasets: they are computationally less intensive, easier to interpret, and generally less prone to overfitting with smaller datasets. Past research consistently demonstrates their effectiveness in customer-related prediction tasks [19], [21], [25], [27]. Additionally, conventional models allow for quicker training times, and their results are more interpretable, which is beneficial when presenting findings to stakeholders unfamiliar with deep learning concepts.

Deep learning models, specifically 1D ResNet and 1D DenseNet, are included due to their sophisticated feature extraction capabilities. ResNet uses residual learning with skip connections that alleviate the vanishing gradient problem [40], enabling more intricate pattern learning. In the context of customer data, this can mean better learning of hierarchical feature interactions across various attributes. DenseNet connects each layer to every subsequent layer in a feed-forward fashion, improving information flow and reducing redundant feature learning. This dense connectivity pattern allows DenseNet to retain critical customer information across layers, leading to more accurate predictions with fewer parameters compared to equivalently deep standard architectures. The choice of 1D architectures aligns with the structured nature of customer data, where each record represents a customer attribute vector, enabling the models to process tabular features with minimal transformations.

IV. Experiments

A. Dataset Description

The dataset [41], "Predicting Profitable Customer Segments", is a publicly available Kaggle benchmark that aggregates historical records from an online retail company to support predictive modeling for targeted marketing campaigns. Each row represents a pre-campaign observation that compares two customer groups, capturing their attributes and relational characteristics before any promotional intervention.

The dataset contains 6621 instances and 70 input features, organized into three semantically distinct categories of columns: the "g1_" columns describing absolute attributes of the first customer group (e.g., aggregated demographic and behavioral measurements), the "g2_" columns describing the corresponding attributes of the second group, and the "c_" columns capturing relational features that quantify pairwise differences between the two groups. The target column categorizes outcomes into three classes: neither group was profitable (0), Group 1 was more profitable (1), or Group 2 was more profitable (2). The class distribution is moderately imbalanced: class 1 (Group 1 more profitable) is the majority class with 3076 instances (46.5%), followed by class 2 (Group 2 more profitable) with 1877 instances (28.3%), and class 0 (neither profitable) as the minority class with 1667 instances

(25.2%). Although exact attribute semantics are not disclosed by the data provider for confidentiality reasons, the variable names and value ranges indicate that the features encompass demographic descriptors (such as age, income, and geographic indicators) and behavioral measures (such as transaction value and purchase frequency). The comparative "c_" columns encode the relative positioning of the two groups, for example the difference in average spending or the ratio of engagement frequencies between the groups. This dual representation, combining absolute group characteristics with relative inter-group metrics, creates a feature space where models can learn both what makes individual groups profitable and how the contrast between groups predicts campaign outcomes. The class imbalance poses an additional challenge, as classifiers tend to optimize for majority classes unless explicitly guided otherwise through class weighting or oversampling. We acknowledge that the limited public documentation of feature semantics constrains the depth of feature-level interpretation that can be performed, and we discuss the implications of this limitation in Section VI.

The implementation used Google Colab Pro with TPU support, TensorFlow/Keras [42], [43] for deep learning, Scikit-learn [44] for classical models, and the XGBoost library [8]. NumPy [45] was used for data manipulation and Matplotlib [46] for visualization. Initial experiments were conducted on local machines equipped with NVIDIA GeForce GTX 1660 Ti and Apple M1 Pro GPUs. However, the computational demands of training deep architectures such as 1D ResNet-50 and 1D DenseNet-121 required migrating to Google Colab Pro, which provided access to TPU and high-RAM environments that significantly accelerated training.

B. Experiment 1: Deep Learning

Two deep learning architectures were evaluated: 1D ResNet and 1D DenseNet. All models were trained using a consistent preprocessing pipeline including feature standardization, SMOTE for class imbalance, and input reshaping to conform to the expected input format of 1D convolutional networks. The dataset was split into 60% training, 20% validation, and 20% testing, with early stopping to prevent overfitting. Running these experiments introduced significant computational overhead, as deep learning architectures such as 1D ResNet-50 and 1D DenseNet-121 were inherently expensive to train in terms of both time and memory, requiring access to TPU-accelerated cloud environments.

1) 1D ResNet: Three variants (ResNet-18, ResNet-34, ResNet-50) were evaluated. The Adam optimizer was used with a batch size of 128 and early stopping. Class weighting was applied to address imbalanced learning. Multiple configurations were explored by varying the learning rate from 0.0001 to 0.1 across four settings per variant, yielding 12 configurations total. The number of training epochs ranged from 21 to 44 due to early stopping.

Among all variants and configurations, ResNet-18 with learning rate 0.1 achieved the best performance, as shown in Table I. The small decline between validation and test performance indicates good generalization. ResNet-34 and ResNet-50 peaked at validation F1 scores of 0.496 and 0.487, respectively, both with learning rate 0.01, falling short of ResNet-18's 0.515. The finding that deeper variants did not improve over ResNet-18 can be attributed to the dataset's relatively small size and low complexity, where a simpler architecture with fewer parameters is more efficient at capturing essential patterns without memorizing noise. The higher learning rate of 0.1 outperformed smaller rates across all variants, suggesting that the loss landscape for this task is relatively smooth, allowing aggressive gradient updates to converge faster without oscillating past optima.

TABLE I. RESNET-18 PERFORMANCE WITH BEST CONFIGURATION

	Precision	Recall	F1 Score	Accuracy
Validation	0.5119	0.5431	0.515	0.5431
Test	0.4971	0.505	0.501	0.4562

TABLE II. DENSENET-121 PERFORMANCE WITH BEST CONFIGURATION

	Precision	Recall	F1 Score	Accuracy
Validation	0.54	0.55	0.55	0.55
Test	0.54	0.54	0.54	0.54

2) 1D DenseNet: DenseNet-121 was evaluated across 11 configurations varying the optimizer (Adam, AdamW), learning rate (0.001 to 0.1), and class weighting strategies (none, automatically computed based on class distribution, and manually assigned). The batch size was fixed at 128 with early stopping.

The experiments revealed critical insights about training deep models on imbalanced data. Configurations without class weights achieved moderate accuracy but ignored the minority class 0 entirely, resulting in low F1 scores. For instance, configurations with no class weights produced biased predictions that favored majority classes, achieving an F1-score of only 0.47. Configurations with very low learning rates (0.001) performed poorly regardless of class weighting, with F1 as low as 0.30, suggesting the optimizer converged too slowly or became stuck in shallow local minima. In contrast, configurations with appropriately set class weights and higher learning rates achieved substantially better results. These findings demonstrate that both an appropriate learning rate and effective class weighting are jointly necessary to handle imbalanced datasets; neither alone is sufficient. The AdamW optimizer with automatic class weights achieved competitive F1 of 0.54, but did not surpass the Adam optimizer under optimal settings. Automatic class weights, computed proportionally from the class distribution, provided a reasonable approximation but were less effective than manually assigned weights that doubled the penalty for misclassifying the minority class.

The best performance was achieved with Adam optimizer, learning rate of 0.1, and manually assigned class weights $\{0 : 2.0, 1 : 1.0, 2 : 1.0\}$, which doubled the penalty for misclassifying the minority class, as shown in Table II. The consistent results across validation and test sets indicate good generalization and no overfitting.

Overall, DenseNet-121 outperformed all ResNet variants, achieving a test F1 of 0.54 compared to 0.50 for ResNet-18. DenseNet benefits from its dense connectivity pattern that creates shorter paths for gradient flow and enables more efficient feature reuse across layers. Its concatenation-based approach

TABLE III. NAÏVE BAYES PERFORMANCE RESULTS

	Precision	Recall	F1 Score	Accuracy
Training	0.50	0.49	0.49	0.48
Test	0.49	0.48	0.49	0.48

retains feature maps from all previous layers, allowing the model to access both low-level and high-level representations simultaneously. This is particularly beneficial for tabular data where feature interactions may span over different levels of abstraction. Deeper ResNet variants (ResNet-34, ResNet-50) did not improve over ResNet-18, suggesting that model depth should be proportional to the complexity and volume of the data. ResNet-50 achieved the lowest validation F1 among all variants (0.356 at learning rate 0.0001), indicating that the additional depth introduced unnecessary parameters that the limited training data could not support. The consistent finding that shallower architectures outperform deeper ones across both ResNet and DenseNet experiments suggests that the dataset's 6621 instances are insufficient to train deep networks effectively, and that the feature space does not contain hierarchical patterns complex enough to justify deep architectures.

C. Experiment 2: Traditional Machine Learning

Eight conventional machine learning classifiers were evaluated using 10-fold cross-validation during training. All models were trained on the same preprocessed dataset using feature standardization and SMOTE for class imbalance. For each model, multiple hyperparameter configurations were explored, and only the best-performing configuration is reported. The hyperparameters for each model were tuned systematically, varying key parameters such as learning rate, regularization strength, kernel type, and ensemble size to identify the optimal balance between model complexity and generalization.

1) Naïve Bayes: Naïve Bayes has no significant tunable hyperparameters and was evaluated using 10-fold cross-validation. Table III reports the results. The model shows consistent but low performance across training and test sets, with F1 below 0.5 reflecting its limited capacity on this dataset. This weak performance is likely due to the strong feature independence assumption, which rarely holds in real-world datasets where features tend to be correlated. The relational features in this dataset ("c_" columns quantifying group differences) are inherently derived from the group-specific features, creating strong inter-feature dependencies that directly violate this assumption. Despite this, the model's stability between training and test sets confirms that the issue is underfitting rather than overfitting. The near-identical training and test scores (0.49 vs. 0.49 for F1) indicate that the model has fully learned what its limited architecture permits, and additional data or features would not improve its performance without relaxing the independence assumption. While Naïve Bayes is the most lightweight model in our study, its weak predictive power limits its practical utility for this task.

2) K-Nearest Neighbors: KNN was tested with K values of 1, 2, 3, 5, 7, and 10 using 10-fold cross-validation. K=1 resulted in severe overfitting (training F1=1.0, test F1=0.71), as the model memorized the training data by returning the label of the single closest point. K=2 and K=3 achieved identical test F1 of 0.62 despite different training scores (0.84 and 0.80, respectively), highlighting diminishing returns from increased model complexity. K=5 yielded test F1 of 0.59, while K=7 dropped to 0.55. The generalization gap narrowed progressively from 0.29 (K=1) to 0.07 (K=10), illustrating the classic bias-variance tradeoff: lower K values produce models with low bias but high variance that are sensitive to noise, whereas higher K values produce smoother decision boundaries with higher bias but lower variance. K=10 provided the most balanced behavior, as shown in Table IV. Although its absolute test F1 of 0.52 is lower than K=2 and K=3, the reduced overfitting and more stable predictions make it a more reliable choice, particularly when the training set has been augmented with SMOTE-generated synthetic samples. The sensitivity of KNN to the choice of K underscores the importance of hyperparameter tuning and cross-validation in preventing misleading performance estimates.

TABLE IV. KNN PERFORMANCE RESULTS (K=10)

	Precision	Recall	F1 Score	Accuracy
Training	0.63	0.60	0.59	0.60
Test	0.56	0.53	0.52	0.53

TABLE V. RANDOM FOREST PERFORMANCE RESULTS

	Precision	Recall	F1 Score	Accuracy
Training	0.55	0.56	0.54	0.56
Test	0.53	0.54	0.53	0.55

3) Random Forest: Three configurations were evaluated varying max depth (5–10), max features (sqrt, log2), and min samples parameters with 5 trees. All configurations achieved similar test performance (F1 between 0.53–0.54), indicating that Random Forest is relatively robust to hyperparameter changes on this dataset. The first two configurations with max depth 10 both achieved training F1 of 0.56 with test F1 of 0.53, exhibiting slight overfitting as the deeper trees memorized training patterns that did not generalize. Using sqrt versus log2 for max features had negligible impact, suggesting that feature diversity among trees was sufficiently maintained by either approach. The most constrained configuration with max depth 5 and increased min samples split provided the best generalization, as shown in Table V. This suggests that on this dataset, controlling tree complexity is more beneficial than allowing the model to capture finer-grained patterns that may be noise. The relatively small ensemble size of 5 trees was chosen to reduce training time, though increasing this number could potentially improve performance at the cost of additional computation.

4) Support Vector Machine: Over 120 hyperparameter configurations were explored, varying the kernel type (linear, polynomial, sigmoid, RBF), regularization parameter C, and gamma. All top-performing configurations used the RBF kernel, indicating its effectiveness in modeling the complex, non-linear decision boundaries present in this dataset. The RBF kernel maps data into a higher-dimensional feature space using a Gaussian function, enabling the construction of non-linear decision boundaries that are necessary for datasets with over-

TABLE VI. SVM PERFORMANCE RESULTS (RBF, C=0.5, GAMMA=0.01)

	Precision	Recall	F1 Score	Accuracy
Training	0.58	0.58	0.58	0.58
Test	0.56	0.53	0.54	0.53

TABLE VII. ADABOOST PERFORMANCE RESULTS

	Precision	Recall	F1 Score	Accuracy
Training	0.55	0.56	0.55	0.56
Test	0.54	0.55	0.54	0.55

TABLE VIII. GRADIENT BOOSTING PERFORMANCE RESULTS

	Precision	Recall	F1 Score	Accuracy
Training	0.70	0.70	0.69	0.70
Test	0.63	0.62	0.62	0.62

TABLE IX. XGBOOST PERFORMANCE RESULTS

	Precision	Recall	F1 Score	Accuracy
Training	0.66	0.68	0.68	0.68
Test	0.60	0.61	0.62	0.62

lapping class distributions. A clear trend emerged: performance declined as C increased and gamma decreased, suggesting that overly large regularization values combined with very small gamma values harmed generalization. Specifically, the top six configurations all used RBF with C values ranging from 0.5 to 500 and gamma from 0.01 to 0.0001, with training F1 scores declining from 0.582 to 0.531 as C increased. The RBF kernel with C=0.5 and gamma=0.01 yielded the best results, as reported in Table VI. The moderate C value strikes a balance between maintaining a strict decision boundary and allowing flexibility for noisy data points. The training-test gap of 0.04 in F1 indicates moderate generalization. The linear, polynomial, and sigmoid kernels all performed below the RBF configurations, confirming that the class boundaries require the flexibility of a radial basis function.

5) AdaBoost: Three configurations were tested varying the learning rate (0.01, 0.1, 1.0) with 100 estimators and the SAMME algorithm [47]. A learning rate of 0.01 severely underfit with F1 of only 0.42 on both training and test sets, indicating that each weak learner's contribution was too small for the ensemble to converge within 100 iterations. This extreme consistency between training and test performance at a low score is a hallmark of underfitting, where the model has insufficient capacity to learn the underlying patterns. Learning rates of 0.1 and 1.0 achieved comparable test F1 of 0.54, with the higher learning rate (1.0) producing marginally higher training performance (0.55 vs. 0.54) without improving test results. The best result, shown in Table VII, demonstrates a minimal gap between training F1 (0.55) and test F1 (0.54), indicating no overfitting. This consistency highlights AdaBoost's strength in reducing variance through its sequential ensemble of weak learners, which iteratively increases the weights of misclassified instances to focus subsequent learners on harder cases. However, its absolute performance falls short of the more advanced boosting variants (Gradient Boosting and XGBoost), which employ gradient-based optimization of the loss function rather than AdaBoost's sample reweighting scheme, representing a 15% relative improvement in F1.

6) Gradient Boosting: Four configurations were tested varying the number of estimators (8–100), max depth (2–3), learning rate (0.1–1.0), and subsample ratio, all using Friedman-MSE criterion and log-loss. Configurations with fewer estimators or shallower trees showed progressively lower performance, indicating underfitting. The second configuration (10 estimators, max depth 3, learning rate 1.0) achieved test F1 of 0.59, while the third and fourth configurations (10 and 8 estimators, max depth 2) yielded 0.57 and 0.56, respectively. The best configuration used 100 estimators, max depth 3, and learning rate 0.1. As shown in Table VIII, Gradient Boosting achieved a test F1 score of 0.62, the highest among all models, tied with XGBoost. The gap between training F1 (0.69) and test F1 (0.62) is moderate, indicating that while some overfitting exists, the model still generalizes reasonably well. The combination of a moderate learning rate with sufficient estimators allowed the model to learn incrementally and capture subtle patterns, while max depth 3 prevented individual trees from becoming overly complex. The progressive performance improvement from the weakest to the strongest configuration demonstrates that the number of estimators and learning rate are the most influential hyperparameters for this task, with max depth playing a secondary role in controlling overfitting.

7) Extreme Gradient Boosting (XGBoost): Four configurations were evaluated varying max depth (2–4), learning rate (0.2–0.3), subsample ratio, and L1/L2 regularization. The best configuration used 100 estimators, max depth 4, learning rate 0.3, and reg_lambda 800 for strong L2 regularization. The high regularization parameter was critical in preventing overfitting despite the relatively deep trees, effectively penalizing large weights and keeping the model from becoming overly complex. The colsample_bytree of 0.8 introduced randomness by selecting a subset of features per tree, and the min_child_weight of 5 ensured that splits only occurred when sufficient data supported them. As shown in Table IX, XGBoost matched Gradient Boosting with a test F1 score of 0.62. Notably, a second configuration with only 50 estimators also achieved test F1 of 0.62 but with lower training performance (0.64 vs. 0.68), indicating that XGBoost can achieve comparable results with different hyperparameter combinations due to the compensatory relationships among its regularization parameters. The remaining two configurations with max depth 2 achieved lower test F1 scores of 0.57 and 0.59, confirming that deeper trees were necessary to capture the complexity of the dataset. The training-test gap of 0.06 is smaller than Gradient Boosting's gap of 0.07, suggesting slightly better generalization. XGBoost's built-in L1 and L2 regularization, combined with its optimized parallel processing and efficient handling of missing data, contributed to its strong performance while maintaining computational efficiency.

8) Linear Discriminant Analysis: Five configurations were tested varying the solver (SVD, LSQR, Eigen), shrinkage, prior probabilities, and tolerance. The SVD solver served as a baseline due to its computational efficiency and numerical stability, while the LSQR and Eigen solvers supported shrinkage to address potential feature redundancy. The Eigen solver

TABLE X. LDA PERFORMANCE RESULTS

	Precision	Recall	F1 Score	Accuracy
Training	0.54	0.53	0.53	0.53
Test	0.56	0.51	0.53	0.51

with automatic shrinkage yielded the best results, as shown in Table X. This combination provided the most balanced results across training and test sets, with the automatic shrinkage estimating an appropriate regularization level from the data. The SVD-based configurations, including one with uniform class priors, showed slightly lower performance, while increasing the tolerance to 0.5 produced the weakest results, suggesting that discarding too many dimensions reduced the model's discriminative ability. Despite testing multiple configurations, the performance differences across all LDA settings were relatively small (F1 ranging from 0.523 to 0.533), suggesting that LDA's linear projection is inherently limited for this dataset. The three-class problem with overlapping class distributions likely requires non-linear decision boundaries that LDA cannot model, which explains its lower performance compared to tree-based and kernel-based methods.

D. Additional Experiments

A One-vs-Rest (OvR) strategy was explored to improve classification of class 0, which the confusion matrix revealed was frequently misclassified as class 1 or class 2. This misclassification pattern was consistent across most models, indicating that class 0 (neither group profitable) shares significant feature overlap with both class 1 and class 2. Separate binary classifiers were trained to distinguish each class from the others, with a specific focus on isolating class 0. The rationale was that decomposing the multi-class problem into simpler binary tasks might allow classifiers to better learn the subtle differences between the minority class and the majority classes. However, the best F1 for class 0 reached only 0.4987, with other configurations yielding 0.4769, 0.4587, and 0.4335. These results suggest that the underlying class overlap and imbalance were difficult to overcome even with targeted binary modeling, and that the features available in the dataset may not be sufficiently discriminative for class 0.

Autoencoders [48], [49] were also tested as a preprocessing step to extract more meaningful features before classification. The hypothesis was that unsupervised feature learning could discover a compressed representation that better separates the classes. A shallow architecture (one hidden layer, 1000 neurons) achieved weighted F1 of 0.5245. However, increasing depth consistently degraded performance: a two-layer architecture yielded 0.5170, a three-layer version dropped to 0.5023, and further increasing the number of hidden nodes to 1500 also did not help, yielding 0.5023. Additional depth beyond three layers produced even lower scores of 0.4982, while reducing the number of nodes led to 0.4790. These results suggest that the dataset may not benefit from unsupervised compression, and that the original feature space already provides a sufficiently informative representation. The degradation with deeper architectures mirrors the findings from the ResNet experiments, reinforcing that additional model complexity does not improve results on this dataset.

TABLE XI. TEST PERFORMANCE COMPARISON OF ALL MODELS

Model	Prec.	Rec.	F1	Acc.	AUC
XGBoost	0.60	0.61	0.62	0.62	0.75
Gradient Boosting	0.63	0.62	0.62	0.62	0.74
AdaBoost	0.54	0.55	0.54	0.55	0.74
Random Forest	0.53	0.54	0.53	0.55	0.73
1D DenseNet-121	0.54	0.54	0.54	0.54	0.72
LDA	0.56	0.51	0.53	0.51	0.72
SVM (RBF)	0.56	0.53	0.54	0.53	0.66
KNN (K=10)	0.56	0.53	0.52	0.53	0.65
Naïve Bayes	0.49	0.48	0.49	0.48	0.69
1D ResNet-18	0.50	0.51	0.50	0.46	0.64

V. COMPARISON OF APPROACHES

Table XI summarizes the test performance of all evaluated models. XGBoost achieved the highest AUC of 0.75, followed by Gradient Boosting (0.74) and AdaBoost (0.74). Among deep learning models, DenseNet-121 performed best with an AUC of 0.72, while ResNet-18 reached 0.64.

Traditional machine learning models, particularly XGBoost and Gradient Boosting, consistently outperformed deep learning approaches. Both achieved the highest test F1 score of 0.62. XGBoost reached the highest AUC of 0.75, while Gradient Boosting followed closely at 0.74. Three interacting properties of the dataset jointly explain this gap. First, the data is tabular and heterogeneous: each feature has its own semantics and scale, so the translation-invariant local filters learned by 1D convolutional architectures have no clear analogue in this representation, whereas axis-aligned splits used by tree-based ensembles map directly onto the per-feature decision structure that is natural for this type of data. Second, the dataset is moderately sized: deep architectures typically require large amounts of training data to generalize effectively, and the 60/20/20 split further reduces the effective deep learning training set to approximately 3973 instances, whereas traditional models benefit from 10-fold cross-validation that uses 90% of the data for training in each fold. Third, the class distribution is imbalanced, and although SMOTE and class weighting mitigate this, the residual difficulty of separating the minority class 0 from classes 1 and 2 affects high-capacity deep models more severely than tree ensembles, which can isolate small but informative regions of feature space through targeted splits. These three factors, namely tabular structure, moderate sample size, and class imbalance, are precisely the conditions under which gradient boosting has been repeatedly shown to dominate deep learning on tabular benchmarks, and our results corroborate that pattern in the marketing analytics setting.

A consistent observation across all models is that the weighted F1 score appears to be bounded around 0.62. This bound is not a tuning artefact: it persists across systematic hyperparameter searches (over 120 configurations for SVM, multiple configurations for every other model), across an alternative classification strategy (One-vs-Rest, with the best class-0 F1 reaching only 0.4987), and across an alternative feature representation (autoencoder-based compression, peaking at 0.5245). The fact that two architecturally distinct top models (XGBoost and Gradient Boosting) converge to the same 0.62 score, and that neither deeper trees, nor stronger regularization, nor unsupervised feature extraction can push

past it, indicates that the limit is a property of the data rather than of the classifiers. The confusion-matrix analysis (see Section IV) confirms that the residual errors concentrate on class 0, which the feature space cannot reliably distinguish from the other two classes given the available attributes. This is consistent with the dataset's confidentiality-induced opacity: domain-relevant features that might disambiguate class 0 (such as marketing channel, prior campaign exposure, or product-category preference) may simply not be represented in the published feature set. Improving beyond F1 $\approx$ 0.62 would therefore most likely require richer input features rather than more expressive models, which we discuss as a direction for future work.

Among deep learning models, DenseNet-121 outperformed ResNet-18, achieving a test F1 score of 0.54 compared to 0.50. ResNet-18 outperformed deeper ResNet variants (ResNet-34 and ResNet-50), suggesting that a simpler architecture was better suited for this dataset. Notably, DenseNet-121 achieved performance comparable to several traditional models (AdaBoost at 0.54, SVM at 0.54) despite requiring orders of magnitude more computation, which raises questions about the practical value of deep learning for structured datasets of this scale.

An interesting observation is the relationship between AUC and F1 discrepancies: Naïve Bayes achieved AUC of 0.69 despite an F1 of only 0.49, indicating that while it can partially rank instances by class probability, its hard classification decisions are poor. This suggests that Naïve Bayes produces well-calibrated probability estimates despite its inability to set effective classification thresholds. LDA similarly showed a gap between AUC (0.72) and F1 (0.53), reflecting its strength in linear separability assessment despite limited classification accuracy. Conversely, XGBoost achieved the highest AUC alongside the highest F1, demonstrating both strong ranking ability and effective classification thresholds. This consistency between ranking and classification metrics indicates that XGBoost produces reliable probability estimates that translate directly into accurate predictions.

Several models incorporate implicit feature selection mechanisms. XGBoost, Random Forest, and AdaBoost evaluate feature importance during tree construction, naturally prioritizing informative features. LDA performs dimensionality reduction by projecting data onto a space that maximizes class separability. SVM with RBF kernel maps data into higher-dimensional spaces for better separation. Deep learning models rely on convolutional layers to learn hierarchical feature representations automatically. These built-in mechanisms allow each model to handle high-dimensional inputs differently, contributing to the observed performance differences.

In terms of computational efficiency, traditional models were significantly more resource-efficient. Table XII and Table XIII compare the computational requirements of all models. The contrast is substantial: DenseNet-121 required 121232 GFLOPs for training versus only 0.15 GFLOPs for XGBoost, a difference of nearly six orders of magnitude. In parameter count, DenseNet-121 (5.53M) had over 120 times more parameters than XGBoost (0.04M). Even the lighter deep learning model, ResNet-18 (0.23M parameters, 3582 GFLOPs), demanded over 24000 times more computation than XGBoost.

TABLE XII. SPACE COMPLEXITY (NUMBER OF PARAMETERS)

Model	Params (M)
1D DenseNet-121	5.5328
SVM	0.4856
KNN	0.4635
1D ResNet-18	0.2342
XGBoost	0.0445
Gradient Boosting	0.0213
LDA	0.0027
AdaBoost	0.0015
Random Forest	0.0004
Naïve Bayes	0.0002

TABLE XIII. TIME COMPLEXITY (GFLOPS)

Model	Training	Testing
1D DenseNet-121	121231.62	10099.90
1D ResNet-18	3582.26	303.06
SVM	1.96	0.10
Gradient Boosting	0.46	0.01
XGBoost	0.15	0.001
LDA	0.03	0.0001
AdaBoost	0.02	0.005
Random Forest	0.002	$<$0.001
Naïve Bayes	$<$0.001	0.001

Naïve Bayes was the most lightweight model (0.0002M parameters, $<$0.001 GFLOPs), but its weak predictive performance limits its practical utility. KNN, while requiring no formal training phase, incurs inference-time cost proportional to the dataset size (0.49 GFLOPs for testing), storing all 0.46M training instances for distance computation at prediction time. SVM, despite achieving moderate F1 of 0.54, required 1.96 GFLOPs for training due to the kernel computations needed for constructing the decision boundary in high-dimensional space. Among the top-performing models, XGBoost was notably faster than Gradient Boosting (0.15 vs. 0.46 GFLOPs for training) due to its optimized parallel processing and built-in regularization, while both achieved identical test F1 of 0.62. XGBoost struck a favorable balance with 0.04M parameters and strong predictive performance.

The results also reveal a clear performance hierarchy within the traditional ML models. The boosting family (XGBoost, Gradient Boosting, AdaBoost) consistently outperformed other approaches, with the gradient-based variants (XGBoost and Gradient Boosting) achieving the best results. This advantage stems from their iterative error correction mechanism, where each subsequent tree focuses on the residuals of the previous ensemble. Random Forest, despite being an ensemble method, achieved lower F1 (0.53) due to its independent tree construction that does not explicitly target misclassified instances. SVM and KNN achieved moderate performance (F1 of 0.54 and 0.52, respectively), while Naïve Bayes and LDA, both limited by strong distributional assumptions, fell short. This ordering aligns with the general principle that models capable of capturing non-linear interactions and iteratively refining predictions perform best on complex classification tasks.

Overall, XGBoost and Gradient Boosting demonstrated the best balance between accuracy, generalization, and computational efficiency, making them well-suited for the customer segment profitability prediction task. Their suitability for structured tabular data and their interpretability provide a practical

advantage for business decision-making, where understanding which customer group features drive profitability predictions is as valuable as the predictions themselves.

From a deployment perspective, the computational disparity reported in Table XII and Table XIII translates directly into operational consequences. XGBoost requires only 0.001 GFLOPs at inference time compared with 10099.90 GFLOPs for DenseNet-121, making it roughly ten million times faster at prediction time and enabling a profitability score to be produced for an incoming customer segment in essentially constant time on commodity CPU hardware. This makes XGBoost compatible with real-time campaign-optimization pipelines and with edge or in-database deployment. By contrast, DenseNet-121 effectively assumes GPU/TPU availability and introduces additional infrastructure cost, latency, and energy consumption that are hard to justify given the absence of any predictive advantage. The 0.04M parameter footprint of XGBoost also simplifies model versioning, A/B testing, and incremental retraining, which are routine requirements in marketing analytics platforms. Finally, the tree-based nature of XGBoost preserves direct compatibility with established interpretability tools (such as gain-based feature importance and SHAP values), supporting the auditability and explanation requirements that increasingly accompany automated marketing decisions. Taken together, these properties make XGBoost and Gradient Boosting the recommended candidates for production deployment in this domain, with deep architectures reserved for settings where dataset size and modality (for instance, image- or sequence-rich customer data) genuinely warrant them.

VI. Limitations

Several limitations of this study should be acknowledged explicitly. First, the dataset size of 6621 instances is moderate relative to the data volumes at which 1D ResNet and 1D DenseNet have historically demonstrated their advantages, and a fairer assessment of deep learning would require benchmarks with at least an order of magnitude more samples; our negative result for deep learning should, therefore, be interpreted as conditional on the data regime studied rather than as a general statement about deep architectures for marketing analytics. Second, the dataset attributes are described only at a category level ("g1_", "g2_", "c_") and exact feature semantics are not disclosed by the data provider for confidentiality reasons, which limits the depth of feature-level interpretation and constrains reproducibility at the level of individual attribute analysis. Third, the evaluation is restricted to classification metrics (precision, recall, F1, accuracy, AUC); business-oriented metrics such as expected ROI uplift, conversion rate gain, customer lifetime value impact, and campaign cost per acquired customer would more directly reflect the practical utility of the predictions, but require labelled campaign-outcome data and a deployment context that are outside the scope of this study. Fourth, although the framework was designed with interpretability in mind, post-hoc explainability techniques such as SHAP value analysis or permutation feature importance were not applied to the top-performing models; integrating such analyses would strengthen the link between model predictions and actionable marketing insights. Fifth, no external dataset or temporal validation was performed: the generalization claims pertain to the held-out split of this single Kaggle source, and replication on additional retail datasets with disclosed feature semantics would be needed to establish broader generalizability. These limitations are addressed in our future work plan in Section VII.

VII. Conclusion

This study explored the application of machine learning and deep learning techniques for predicting profitable customer segments to optimize targeted marketing strategies. A comprehensive framework was developed, incorporating conventional classifiers (AdaBoost, Gradient Boosting, XGBoost, LDA, Random Forest, Naïve Bayes, SVM, and KNN) and deep learning architectures (1D ResNet and 1D DenseNet).

The experimental results demonstrated that traditional machine learning models, particularly XGBoost and Gradient Boosting, consistently outperformed deep learning models across multiple metrics. Both achieved the highest weighted F1 score of 0.62 and the highest AUC values (0.75 and 0.74, respectively). The performance advantage of traditional models is attributed to the relatively limited dataset size and class imbalance, conditions under which deep learning models struggle to generalize effectively. Among the deep learning models, DenseNet-121 outperformed all ResNet variants with a test F1 of 0.54, while deeper ResNet architectures did not improve over the simpler ResNet-18, reinforcing that model complexity should be proportional to dataset size.

XGBoost offered additional advantages in terms of execution speed and computational efficiency while maintaining comparable predictive performance to Gradient Boosting. With only 0.04M parameters and 0.15 GFLOPs for training, XGBoost required nearly six orders of magnitude fewer computations than DenseNet-121 (121232 GFLOPs). Both models demonstrated effective generalization and robust handling of imbalanced data. As traditional machine learning models, they are also more interpretable than deep learning approaches, which is a key benefit for understanding model decisions and driving actionable business insights.

The additional experiments with One-vs-Rest classification and autoencoders did not yield significant improvements over the direct multi-class approach, suggesting that the underlying class overlap in the dataset poses a fundamental challenge that cannot be easily addressed through alternative classification strategies or unsupervised feature extraction.

Future work will pursue four directions in response to the limitations identified above. First, the framework will be evaluated on larger and more diverse customer datasets, particularly those exceeding tens of thousands of instances, to characterize the data-volume regime in which deep architectures begin to outperform tree-based ensembles for this task. Second, post-hoc explainability techniques, specifically SHAP value analysis and permutation-based feature importance, will be applied to XGBoost and Gradient Boosting to produce interpretable, per-prediction attributions that marketing teams can act on, addressing the interpretability dimension that the present study only partially covers. Third, the evaluation will be extended from purely classification-oriented metrics to business-oriented metrics, including expected ROI uplift, conversion rate gain, customer lifetime value impact, and cost per acquired customer, by partnering with marketing teams to

obtain labelled campaign-outcome data that link model predictions to realized financial returns. Fourth, hybrid ensemble methods that combine the strengths of gradient boosting with neural feature extractors will be investigated as a potential route to overcoming the F1 $\approx$ 0.62 ceiling observed on the current feature set, alongside richer input features (such as marketing channel, prior campaign exposure, and product-category preference) that may help disambiguate the residual confusion involving class 0. Robust outlier detection methods will also be incorporated as part of the preprocessing pipeline to further enhance data quality.

ACKNOWLEDGMENT

First and foremost, we thank Allah Almighty, the Most Gracious, the Most Merciful, for the countless blessings that led to the successful completion of this research. We are deeply grateful to our supervisor, Prof. Ouiem Bchir, for her continuous support, patience, and dedication throughout every stage of this work. Finally, we express our deepest gratitude to our families and friends for their unwavering support, love, and encouragement.

REFERENCES

[1] N. E. Koufi, A. Belangour, and M. Sadiq, "Toward a decision-making system based on artificial intelligence for precision marketing: A case study of Morocco," *Journal of Open Innovation: Technology, Market, and Complexity*, vol. 10, no. 1, p. 100250, 2024.

[2] M. Hafez, "Pioneering perspectives: Strategies and considerations in market segmentation and targeting," *SSRN Electronic Journal*, 2024.

[3] V. Kumar, A. R. Ashraf, and W. Nadeem, "AI-powered marketing: What, where, and how?" *International Journal of Information Management*, vol. 77, p. 102783, 2024.

[4] S. Adhikari, "Machine learning and customer behavior insights: Exploring the depth of predictive analytics in enhancing consumer interaction and engagement," *Journal of Empirical Social Science Studies*, vol. 8, no. 2, pp. 51–62, 2024.

[5] A. R. Thomas, "The end of mass marketing: or, why all successful marketing is now direct marketing," *Direct Marketing: An International Journal*, vol. 1, no. 1, pp. 6–16, 2007.

[6] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[7] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001.

[8] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794.

[9] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. Springer, 2009.

[10] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[11] B. Steele, J. Chandler, and S. Reddy, *Algorithms for Data Science*. Switzerland: Springer, 2016.

[12] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[13] A. Burkov, *The Hundred-Page Machine Learning Book*. Andriy Burkov, 2019.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[15] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2261–2269.

[16] A. Xu, Y. Li, and P. K. Donta, "Marketing decision model and consumer behavior prediction with deep learning," *Journal of Organizational and End User Computing*, vol. 36, no. 1, pp. 1–25, 2024.

[17] A. A. Keji, O. Fakeye, N. N. Onochie, and O. Sangotoki, "Predicting long-term deposit customers using convolutional neural network and data conversion technique," *African Scientific Reports*, p. 192, 2024.

[18] N. Nabi *et al.*, "Unleashing deep learning: Transforming e-commerce profit prediction with CNNs," *Journal of Business and Management Studies*, vol. 6, no. 2, pp. 126–131, 2024.

[19] E. Deniz and S. Ç. Bülbül, "Predicting customer purchase behavior using machine learning models," *Information Technology in Economics and Business*, 2024.

[20] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 3rd ed. O'Reilly Media, Inc., 2022.

[21] R. D. Krishna, M. Mahadev, S. Hariprasad, S. Abhishek, and T. Anjali, "Cultivating customer purchase intent: Leveraging machine learning for precise predictions," in *2023 12th International Conference on System Modeling and Advancement in Research Trends (SMART)*, 2023, pp. 374–379.

[22] I. Ullah, B. Raza, A. K. Malik, M. Imran, S. U. Islam, and S. W. Kim, "A churn prediction model using random forest: Analysis of machine learning techniques for churn prediction and factor identification in telecom sector," *IEEE Access*, vol. 7, pp. 60 134–60 149, 2019.

[23] C. Daqing, K., and L. B. Guo, "Predicting customer profitability dynamically over time: An experimental comparative study," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*. Springer International Publishing, 2019, pp. 174–183.

[24] S. E. Saeed, M. Hammad, and A. Alqaddoumi, "Predicting customer's subscription response to bank telemarketing campaign based on machine learning algorithms," in *2022 International Conference on Decision Aid Sciences and Applications (DASA)*, 2022, pp. 1474–1478.

[25] Y. Sun, D. Cheng, S. Bandyopadhyay, and W. Xue, "Profitable retail customer identification based on a combined prediction strategy of customer lifetime value," *Midwest Social Sciences Journal*, vol. 24, no. 1, pp. 104–127, 2021.

[26] Y. Sun, H. Liu, and Y. Gao, "Research on customer lifetime value based on machine learning algorithms and customer relationship management analysis model," *Heliyon*, vol. 9, no. 2, 2023.

[27] X. Xiahou and Y. Harada, "Customer churn prediction using AdaBoost classifier and BP neural network techniques in the e-commerce industry," *American Journal of Industrial and Business Management*, vol. 12, no. 3, pp. 277–293, 2022.

[28] A. Mishra and C. Sharma, "Credit card fraud detection using AdaBoost and bagging ensemble techniques," in *Proceedings of the International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, 2021, aUTHORS: please verify this entry matches the exact paper you intended to cite; the original .tex used [28] without a corresponding .bib entry. Adjust title/venue/year if needed.

[29] T.-T. Wong, N.-Y. Yang, and G.-H. Chen, "Hybrid classification algorithms based on instance filtering," *Information Sciences*, vol. 520, pp. 445–455, 2020.

[30] S. Uddin, A. Khan, M. E. Hossain, and M. A. Moni, "Comparing different supervised machine learning algorithms for disease prediction," *BMC Medical Informatics and Decision Making*, vol. 19, no. 1, p. 281, 2019.

[31] A. B. Musa, "Comparative study on classification performance between support vector machine and logistic regression," *International Journal of Machine Learning and Cybernetics*, vol. 4, no. 1, pp. 13–24, 2013.

[32] R. Dey and F. M. Salem, "Gate-variants of gated recurrent unit (GRU) neural networks," in *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, 2017, pp. 1597–1600.

[33] S. Moro, P. Rita, and P. Cortez, "Bank marketing – UCI machine learning repository," 2014. [Online]. Available: https://archive.ics.uci.edu/dataset/222/bank+marketing

[34] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the International Conference on Learning Representations*, 2015.

[35] N. Chaudhuri, G. Gupta, V. Vamsi, and I. Bose, "On the platform but will they buy? Predicting customers' purchase behavior using deep learning," *Decision Support Systems*, vol. 149, 2021.

[36] N. Singh, P. Singh, K. K. Singh, and A. Singh, "Machine learning based classification and segmentation techniques for CRM: a customer analytics," *International Journal of Business Forecasting and Marketing Intelligence*, vol. 6, no. 2, p. 99, 2020.

[37] S. S. Haykin, *Neural Networks and Learning Machines*. Prentice Hall/Pearson, 2009.

[38] L. Dong, C. Wang, G. Yang, Z. Huang, Z. Zhang, and C. Li, "An improved ResNet-1d with channel attention for tool wear monitor in smart manufacturing," *Sensors*, vol. 23, no. 3, 2023.

[39] Z. Wang, W. Yan, and T. Oates, "Time series classification from scratch with deep neural networks: A strong baseline," in *Proceedings of the International Joint Conference on Neural Networks*, 2017, pp. 1578–1585.

[40] D. Wu, Y. Wang, S.-T. Xia, J. Bailey, and X. Ma, "Skip connections matter: On the transferability of adversarial examples generated with ResNets," Proceedings of the International Conference on Learning Representations (ICLR), 2020.

[41] Tsiaras, Christos (contributor) and Kaggle, "Predicting profitable customer segments," Kaggle Datasets, 2024, tabular dataset, 6621 instances with grouped customer attribute features (g1_, g2_, c_) and a three-class profitability target. Accessed: 2026-05-15. [Online]. Available: https://www.kaggle.com/datasets/tsiaras/predicting-profitable-customer-segments

[42] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: A system for large-scale machine learning," in *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16)*. Savannah, GA, USA: USENIX Association, 2016, pp. 265–283, software available from https://www.tensorflow.org. Version 2.x used in this work.

[43] F. Chollet *et al.*, "Keras: Deep learning for humans," Software library, 2015–2024, version 2.x used in this work. Available from https://keras.io/. [Online]. Available: https://keras.io/

[44] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, 2011.

[45] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020.

[46] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[47] J. Zhu, H. Zou, S. Rosset, and T. Hastie, "Multi-class AdaBoost," *Statistics and Its Interface*, vol. 2, no. 3, pp. 349–360, 2009.

[48] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[49] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.