From Imbalance to Equilibrium: Harmonizing Complexity in Tuning and Model Selection

Nader Liddawi
nliddawi3@gatech.edu

***Abstract*—In this report, we investigate the performance of three popular classification algorithms—K-Nearest Neighbors (KNN), Support Vector Machines (SVM), and a Neural Network (NN)—across two distinct datasets. The Marketing Campaign dataset, characterized by severe class imbalance in the response variable, and the balanced Spotify dataset provide complementary testing grounds for our hypotheses. Hypothesis 1 asserts that for imbalanced data, algorithms with specialized handling (e.g., balanced class weighting and SMOTEENN preprocessing) will outperform more naive methods. Hypothesis 2 posits that in a balanced data scenario, simpler decision boundaries may suffice for high performance. The report includes rigorous experimental results accompanied by learning and validation curves, detailed hyperparameter search outputs, and algorithmic insights that directly relate theoretical considerations to code output. This comprehensive approach underscores the importance of aligning model complexity with data characteristics for achieving optimal F1 and ROC-AUC scores.**

I. INTRODUCTION

Machine learning models do not simply succeed or fail randomly: they are heavily influenced by the underlying data distribution, the hyperparameters chosen, and the complex interactions between algorithmic mechanics and domain-specific features. In this study, we investigate three popular classification algorithms—KNN, SVM, and NN—across two very different datasets. The Marketing Campaign response variable is highly imbalanced, with far fewer positive responses to the marketing campaign than negatives, whereas the Spotify response variable is more balanced, dividing tracks into "high streams" versus "low streams" along the median.

Throughout this work, we explore two main hypotheses. Hypothesis 1 holds that on highly imbalanced data, algorithms specifically designed or tuned to handle skewed distributions (e.g., SVM with class weighting and SMOTE-based preprocessing) should outperform simpler or less specialized approaches. Hypothesis 2 posits that when the data is more balanced (like Spotify), a simpler decision boundary (possibly linear) might be sufficient to reach near-optimal performance, and more complex approaches may offer diminishing returns. In each experimental step, we tie our design choices and results back to these two hypotheses, explaining why some hyperparameters soared and others faltered.

A. Motivation and ML Perspective

The motivation behind this work is to bridge the gap between theoretical insights and empirical performance.

Our investigation emphasizes how fundamental algorithmic features—such as distance metrics in KNN, margin maximization in SVM, and non-linear activations in NN—interact with data-specific characteristics. The integration of code output data further strengthens our analysis by providing quantitative support to theoretical claims. Additionally, because the Marketing dataset is highly imbalanced, F1 and ROC-AUC (hereafter abbreviated, AUC) are more informative than accuracy. Accuracy can be misleading when $\sim 85\%$ of the data is of one class. A high accuracy might simply reflect the model always predicting the majority class. By contrast, F1 captures the harmonic mean of precision and recall—vital for understanding how effectively we are capturing minority examples—while AUC measures the true-positive rate against the false-positive rate across all thresholds. This is particularly relevant for understanding rank ordering and overall separability in an imbalanced scenario. In the balanced Spotify dataset, both F1 and AUC again provide richer information than mere accuracy: F1 clarifies how well precision and recall are balanced, and AUC helps us gauge overall ranking performance. We chose to highlight these metrics (F1 and AUC) in different places based on the practical need: for example, AUC is particularly useful for a broad sense of ranking quality, while F1 is crucial where we want to ensure we do not ignore minority classes or degrade precision and recall.

B. Additional Metric Rationale

Why ROC-AUC was chosen over AUC: ROC-AUC was chosen over a standard ROC curve because it provides a concise, quantitative summary of model performance across all classification thresholds. While the ROC curve visually illustrates the trade-off between true-positive and false-positive rates at different thresholds, ROC-AUC aggregates this information into a single, threshold-independent metric. This single value simplifies model comparisons and offers a clear measure of discriminative power, especially important when class distributions (like Marketing data) are imbalanced. Additionally, ROC-AUC quantifies the probability that a randomly selected positive instance will score higher than a randomly selected negative one, capturing the overall ranking ability of the model in a way that a mere graphical ROC curve cannot.

Why f1 micro was chosen in graphs over F1: F1 micro was chosen for our validation curves because it aggregates the contributions of all classes, providing a single, global measure of performance. Rather than evaluating each class separately or disproportionately emphasizing minority classes, F1 micro computes precision and recall by summing the total true positives, false negatives, and false positives across the entire dataset. This comprehensive approach is particularly useful in binary classification, where a unified metric is desired. Moreover, by accounting for overall correct predictions instead of averaging performance per class, F1 micro can offer a less pessimistic view than F1 macro when dealing with very small classes. Ultimately, this metric delivers a clear and holistic picture of how the model performs across all instances.

II. DESCRIPTION OF THE DATASETS

A. *Overview of the Two Datasets*

The Marketing Campaign dataset contains demographic and purchasing features, but only a small minority of customers respond to the campaign. This imbalance aligns with Hypothesis 1, suggesting that we need specialized techniques (such as SMOTEENN and class-weighted SVM) to glean strong performance metrics. The Spotify dataset, by contrast, is more evenly split: around half the tracks are above the median stream count, and half below. Our Hypothesis 2 indicates that simpler linear or distance-based approaches might suffice if the data has strong linear or localized clusters in feature space.

B. *Additional Data Summary from Code Output*

Below are the summary statistics and target distributions:

TABLE I
DATASET SUMMARY

Dataset	**Shape**	**Feature Summaries**	**Target Distribution**
Marketing Campaign	(2216, 26)	Year Birth, Education, Z Cost-Contact, Z Revenue, etc.	$0 \rightarrow 84.97\%$, $1 \rightarrow 15.03\%$
Spotify	(952, 21)	In spotify, released year, liveness %, speechiness %, etc.	$\sim$50% of samples in each class

These summaries emphasize the data's complexity and support the necessity of employing distinct strategies for imbalanced versus balanced scenarios.

III. METHODS AND PARAMETER RANGES

A. *Data Splitting and Preprocessing*

We split both datasets into training, validation, and testing partitions (70%, 15%, 15%), using stratification to preserve class distributions. For the Marketing dataset, we additionally applied SMOTEENN to the training fold to mitigate the severe imbalance, consistent with Hypothesis 1 that specialized preprocessing can boost minority recall. After splitting, we label-encoded categorical variables but we dropped columns like artist, track name, ID and customer because these records often appear only once and therefore may not contribute to meaningful patterns. Then, we standardized numerical features to ensure that distance metrics (KNN) and gradient-based approaches (NN) were not unduly influenced by feature scaling disparities.

A key theoretical reason for using SMOTEENN is that oversampling just by SMOTE can synthesize minority examples, but might include noisy points. The ENN (Edited Nearest Neighbors) step removes possible misclassified or noisy samples, reducing variance caused by these artificial points. This process helps calibrate the bias–variance tradeoff: SMOTE reduces bias against minority points, whereas ENN helps cut down variance from overfitting to synthetic data.

B. *Hyperparameter Spaces Explored*

To evaluate each algorithm, we performed either RandomizedSearchCV or Optuna across hyperparameter ranges. The benefit of randomized search is that it leverages cross-validation in randomly sampling hyperparameter combinations and evaluating each configuration multiple times, which smooths out the noise inherent in any one partition of data. And the value of Optuna is it enhances this process by incorporating adaptive techniques such as pruning unpromising trials, which means that computational resources are concentrated on configurations that exhibit early promise. These parameter sets were chosen to reflect the core "knobs" we can turn for each model, and we systematically explored them for the following theoretical reasons:

KNN:
We varied n_neighbors (1 to 30) because the number of neighbors is fundamental in balancing bias and variance. Low k often lowers bias but increases variance; higher k may reduce variance but can lead to higher bias. In addition, we tested "uniform" vs. "distance" weightings to see if local weighting improved minority recall. We compared Manhattan, Euclidean, and Chebyshev distances to capture different geometric relationships in the feature space. Finally, we excluded more complex metrics (e.g., Mahalanobis) to maintain interpretability and avoid extra parameterization of covariance matrices.

SVM:
We varied the kernel (linear, RBF, poly) to capture everything from a direct linear margin to highly non-linear transformations. We included a log-scale search for C (penalty for misclassification) and γ (kernel coefficient) because these drastically alter margin tightness and local curvature. We tested class_weight set to "balanced" or None because of its importance for Hypothesis 1 on imbalanced data. Finally, we chose to focus on polynomial degrees up to a certain limit and RBF for theoretical efficiency; more exotic kernels can become unwieldy and do not necessarily improve performance in these datasets. And sigmoid is less optimal in practice due to it not always being positive definite and having less of well-defined geometric interpretation, whereas RBF and polynomial can sufficiently capture wide classes of decision boundaries.

Neural Network (NN):
We tested activation functions (ReLU, tanh, logistic) and varied the hidden layer sizes (50, 100, 150). In addition, we varied the L2 regularization coefficient (alpha) from $1e^{-6}$ to $1e^{-2}$ (log scale) and learning rate from $1e^{-5}$ to $1e^{-2}$. We kept only one hidden layer to directly test how a single-layer NN interacts with these data distributions. The rationale is theoretically grounded in the idea that more layers can capture more complex functions but might overfit small or tabular data like our two datasets unless carefully regularized or expanded with even more hyperparameters (e.g., dropout rates, batch normalization). Each of these hyperparameters strongly influences the bias–variance tradeoff. For instance, a large hidden layer can reduce bias (by modeling complex relationships) but

may increase variance, while a strong regularization alpha shrinks weights, potentially increasing bias while lowering variance.

C. Searching and Threshold Tuning

We used 3-fold cross-validation for the main hyperparameter search, balancing computational feasibility with the desire to see how each algorithm performed on multiple data partitions. For final threshold selection, we used 5-fold cross-validation to find the probability cutoff that maximizes F1, which is particularly crucial for imbalanced data. We combined these two procedures so as not to bias the final test results.

Learning curves and validation curves were generated to probe the bias–variance tradeoff. Learning curves track training and validation error as the number of training samples (or epochs, in the NN) changes; a large gap between these curves often implies high variance (overfitting), while both curves converging at a high error can indicate high bias. Validation curves (e.g., varying n_neighbors in KNN or alpha in NN) show how a single hyperparameter affects training vs. validation performance, thus helping us identify the sweet spot that balances bias and variance.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

A. CV-Based Approach

The below tables rank the top-5 list of hyperparameter combinations for each algorithm as obtained by the search process, sorted by the F1 score. These tables are generated from the cross-validation results of a randomized search, where multiple combinations of parameters are evaluated. While the cross-validation scores in these tables might be optimistically high due to being computed on parts of the training data via the cross-validation method, they guide the iterative process of hyperparameter tuning and model selection, ultimately influencing the final evaluation on a completely independent test set.

The test hold-out F1 and AUC scores (Table II), in contrast, are considerably lower than the cross-validation results shown below because the models did not generalize as well on unseen, imbalanced Marketing data compared to a cross-validation process where tuned hyperparameters are trained on every partition of data and the CV threshold adjustments maximizes the F1 and AUC scores. The final Marketing test set remains untouched, naturally imbalanced and slightly different in distribution, so the ultimate F1 and AUC does dip. In addition, the CV-based threshold is chosen to maximize F1 on validation folds, but it can lead to conservative recall in final testing. The extremely low thresholds can inflate minority-class predictions in cross-validation and local noisy regions reduce the overall F1.

Indeed, the CV threshold values were found uniquely via a grid-search approach that optimizes the F1 score over a range of possible probability thresholds. Once an optimal threshold is found for each fold, these thresholds are averaged to produce a final, robust CV-based threshold. This averaging helps mitigate the variance that can occur due to differences in the data distribution across folds, providing a more generalizable threshold when the model is applied to unseen data.

TABLE II
TEST SET HOLD-OUT RESULTS

Model (Dataset)	CV Threshold	F1 Score	AUC Score
Marketing Campaign – KNN	0.008	0.4828	0.6817
Marketing Campaign – SVM	0.242	0.5918	0.8686
Marketing Campaign – Neural Net	0.328	0.3470	0.7674
Spotify – KNN	0.402	0.7857	0.8445
Spotify – SVM	0.482	0.8951	0.9368
Spotify – Neural Net	0.340	0.7239	0.8372

B. Marketing Campaign: Imbalanced Scenario

TABLE III
KNN MARKETING RESULTS

param n neighbors	param weights	param metric	mean test F1	mean test AUC
1	distance	manhattan	0.914958938	0.887929293
4	distance	manhattan	0.910677529	0.949199134
7	distance	manhattan	0.909511531	0.959059644
28	distance	manhattan	0.902835246	0.962708033
3	uniform	manhattan	0.892393103	0.927126022

1) KNN: Hyperparameter Results:

With SMOTEENN oversampling, certain synthetic minority points can be in tight clusters. A single neighbor (k = 1) can “lock onto” these minority examples, thereby improving recall. In an oversampled minority region, using k = 1 allows the minority neighbor to dominate the label assignment, boosting recall. In contrast, a larger k can dilute this effect by including majority neighbors, which may lower recall and dilute the minority signal. But k = 1, or any small k, introduces higher variance because any noise in the training set directly impacts predictions. Thus, while KNN with k = 1 shows strong recall, its performance in F1 does not surpass that of other methods. The preference for Manhattan distance may be explained by its summing of absolute differences, which aligns better with how customer features shift dimension by dimension rather than the squared-distance measure of Euclidean distance that loses some local structure. The Manhattan metric preserves local structures that are important for capturing minority classes.

Despite these successes, KNN’s F1 (0.48) test hold-out remains lower than SVM’s 0.59 on the imbalanced Marketing test set. Why? The borderline minority points are likely better separated by a margin-based approach that can incorporate class weighting, while KNN can be more sensitive to local noise.

Table III indicates that certain values of k with “distance” weighting excelled. Smaller k can yield strong performance, but it risks overfitting (high variance). The results highlight an inherent bias–variance tradeoff: k = 1 reduces bias (very flexible) but can inflate variance, whereas values like k = 7 or k = 11 may offer a more stable boundary but with weaker recall on minority examples.

2) SVM: Hyperparameter Results:

In Table IV, a large C (approximately 67) imposes a higher penalty for misclassification, which encourages the model

TABLE IV
SVM MARKETING RESULTS

param kernel	param class weight	param C	param gamma	mean test F1	mean test AUC
rbf	balanced	67.32249	0.001433	0.861658	0.927340067
linear	—	0.074593	0.000355	0.853428	0.912669553
linear	—	2.520796	0.000157	0.853361	0.911678692
linear	balanced	13.92155	0.078662	0.842062	0.91032708
linear	balanced	0.033206	0.000105	0.839604	0.91004329

to correctly classify minority samples. The use of balanced class weight addresses imbalance by adjusting the decision boundary to give minority instances more weight.

The RBF kernel captures nuanced, non-linear boundaries around minority clusters, especially those formed via SMOTEENN. The synergy among margin maximization, local kernel flexibility, and balanced weighting reduces both bias (through a flexible kernel) and variance (through margin-based constraints) more effectively than the purely instance-based approach of KNN. Indeed, the RBF kernel's local "bubble" expansions help isolate minority clusters formed by SMOTEENN, while large C punishes misclassification harshly, improving minority recall. Balanced class weighting overcame the raw frequency advantage of the majority class, consistent with Hypothesis 1 that specialized methods for imbalanced data should excel. One might ask: Why didn't polynomial or linear kernels match RBF? Presumably, polynomial kernels can overfit or remain too rigid, whereas linear cannot contort enough to capture nuanced spending patterns. The synergy of the RBF kernel and balanced weighting gave the best boundary.

TABLE V
NN MARKETING RESULTS

params activation	params hidden layer sizes	params alpha	params learning rate init	f1 score	auc
logistic	100	0.000246	0.005829	0.527607	0.857163
relu	150	9.14E-05	0.001153	0.520548	0.86734
relu	100	7.50E-05	0.001319	0.519481	0.864468
logistic	150	0.007554	0.007332	0.518072	0.860461
logistic	150	0.009438	0.001528	0.515337	0.867411

3) Neural Network (NN): Hyperparameter Results: The NN underperformed relative to SVM and even KNN. The logistic activation can saturate with extreme input values, potentially complicating the optimization process. A single hidden layer with 100 units, although relatively large, might not converge to a better local optimum in the allotted training epochs (150), particularly if local minima or saddle points are unfavorable or if the data distribution is especially challenging (imbalanced, noisy).

The tight, sparse clusters of the minority class in the marketing dataset may be more effectively handled by SVM's margin-based approach with balanced weighting. Furthermore, the presence of oversampled points can cause the NN to fluctuate in optimization if the gradient-based approach does not encounter enough truly representative minority data. Thus, SVM appears to find a more favorable bias–variance sweet spot in this scenario. Another question is: Why not use ReLU or tanh effectively? The cross-validation folds might, by chance, have favored logistic in certain metrics, but it generalizes worse on the final test set. Also, NN can require more specialized hyperparameter sweeps (e.g., more iterations, multiple layers, or different alpha and learning rate combos) to truly shine. We only tested 150 iterations due to runtime constraints, so possibly more training was needed.

The F1 scores for NN for the test hold-out in Table II and Table V are notably lower than AUC because hyperparameter tuning and threshold selection were performed on artificially balanced, cross-validated data (e.g., via SMOTEENN). This process causes the models to optimize probability predictions for balanced folds, but when applied to the original imbalanced hold-out set, the probabilities are miscalibrated. Although the models rank instances well (resulting in a high AUC), the absolute probability estimates do not align with an optimal decision threshold. This misalignment leads to a poor balance between precision and recall, thereby increasing false positives or negatives and ultimately lowering the F1 score.

C. Learning and Validation Curves for the Marketing Campaign

KNN:

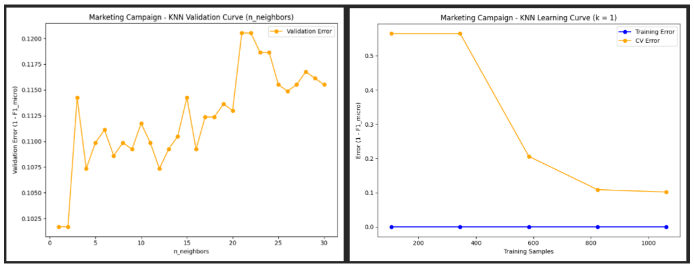

Fig. 1. KNN for Marketing

1) The stabilization of CV error around 0.1, as shown in the learning curve, reflects that while increasing data helps reduce variance, the KNN model remains overfit to local noise for k = 1. This effect is exacerbated by the imbalanced nature of the dataset.

2) The use of SMOTEENN introduces synthetic samples, which can reduce noise in the minority class but may also create artifacts that the model cannot generalize beyond. Larger training sizes improve neighborhood structure, reducing overfitting. But k = 1 in the validation curve is inherently too flexible to fully capture the dataset's general structure, which causes the model to overfit with k = 1.

SVM:

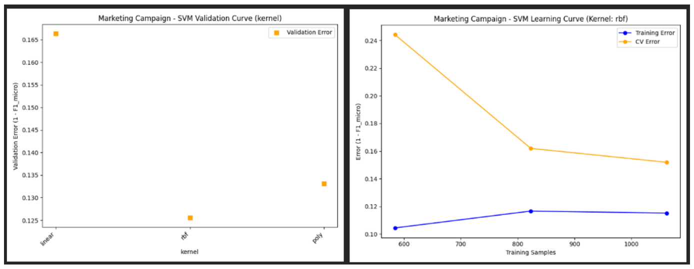

Fig. 2. SVM for Marketing

1) The marketing dataset, which includes categorical and numerical features, benefits from a flexible decision boundary

provided by the RBF kernel, as shown in the validation curve. The SMOTEENN preprocessing may introduce noise, which the RBF kernel can handle better due to its smooth decision boundaries. The linear kernel struggles because it cannot model non-linear dependencies, and the polynomial kernel likely overfits due to the high complexity of decision boundaries it introduces.

2) The low training error reflects the flexibility of the RBF kernel in fitting the training data. However, this flexibility also leads to some overfitting, as shown by the initial gap (in the learning curve) between the training and CV errors.

NN:

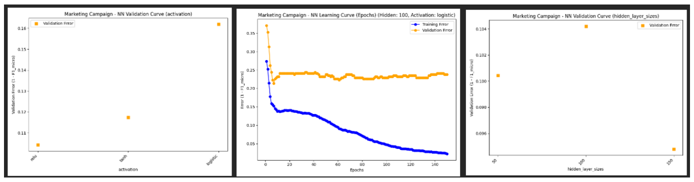

Fig. 3. NN for Marketing

1) The marketing campaign dataset likely contains non-linear relationships, which ReLU handles efficiently by selectively activating neurons. Logistic activation, with its limited range and gradient saturation, struggles to model such relationships effectively.

2) Logistic activation is less effective at representing complex relationships due to its inherent limitations (e.g., vanishing gradients). Early stopping or hyperparameter tuning might be needed to reduce the gap between training and validation errors.

3) The marketing dataset likely contains intricate patterns, and a larger number of hidden units provides the model with the capacity to learn these patterns effectively. However, excessively large networks might lead to overfitting, so the performance peak observed at 150 units suggests an optimal balance.

Bias–Variance Emphasis:
These curves reinforce that SVM mitigated high variance through a well-tuned margin and class weighting, while KNN with $k = 1$ risked local overfitting (high variance), and the NN may have been trapped in suboptimal local minima. The combined use of a flexible kernel method and balanced weighting in SVM aligned well with the imbalanced dataset, confirming Hypothesis 1. The SVM, which includes a specialized balanced weighting, outperformed simpler or more naive algorithms on an imbalanced dataset, and KNN with oversampling had partial success but less stability than margin-based classification.

D. Spotify: Balanced Scenario

1) KNN: Hyperparameter Results:

In a more balanced dataset like Spotify, local neighborhoods can better reflect the underlying distribution without the severe minority–majority skew. The choice of $k = 4$ moderately reduces variance by smoothing out local anomalies while still preserving enough flexibility to capture local patterns in the audio/track features. The use of distance weighting implies that closer neighbors have more influence, which can help delineate borderline cases effectively.

TABLE VI
KNN Spotify Results

param n neighbors	param weights	param metric	mean test F1	mean test AUC
4	distance	manhattan	0.713775	0.796783
3	uniform	manhattan	0.698801	0.778441
24	distance	euclidean	0.689326	0.798312
7	distance	manhattan	0.682892	0.811839
28	distance	manhattan	0.680415	0.854503

KNN with $k = 4$ gave a test F1 near 0.79 in Table VI. Why $k = 4$ instead of $k = 1$ or $k = 10$? At $k = 1$, you risk memorizing local anomalies. Meanwhile, with $k = 4$ in an approximately balanced audio feature space, the model can adapt to local neighborhoods of "high-stream" tracks. The distance weighting ensures that very close neighbors matter more. However, SVM still beats KNN: Why? Possibly the audio features are well-organized linearly in a high-dimensional space, letting SVM find a cleaner margin.

TABLE VII
SVM Spotify Results

param kernel	param class weight	param C	param gamma	mean test F1	mean test AUC
linear	balanced	13.92155	0.078662	0.81511	0.886427
linear	—	2.520796	0.000157	0.812037	0.885101
rbf	balanced	67.32249	0.001433	0.79914	0.885669
linear	—	0.074593	0.000355	0.775262	0.865541
linear	balanced	0.033206	0.000105	0.740287	0.860644

2) SVM: Hyperparameter Results:

SVM with a linear kernel soared to 0.82 F1 and 0.89 AUC for the CV results in Table VII. Why linear? The data, after standardization, might be nearly linearly separable. When the classes are balanced, a well-tuned linear margin can easily yield high recall and precision. The moderate C (≈ 13.92) imposes enough penalty for misclassification to reduce bias while not over-complicating the decision boundary. Balanced class weight might help a bit, though it's less vital than in the imbalanced scenario. Hypothesis 2 specifically predicted that simpler boundaries could suffice in balanced data, and indeed the linear SVM provided near-optimal performance. In summary, the Spotify features, once standardized, appear to separate well in a linear subspace. Consequently, a linear kernel suffices, drastically reducing variance compared to more complex kernels (such as polynomial or RBF) on data that may not require intricate curvature.

Although balanced class weight remains useful for ensuring that no slight skew in the classes is overlooked, the inherent 50/50 distribution means it plays a less critical role than in the Marketing data.

TABLE VIII
NN Spotify Results

params activation	params hidden layer sizes	params alpha	params learning rate init	f1 score	auc
tanh	50	1.71E-06	0.003968	0.80597	0.907277
relu	150	1.07E-06	0.007284	0.80292	0.892606
relu	150	1.15E-06	0.008636	0.797101	0.891823
logistic	100	0.000246	0.005829	0.788321	0.891236
relu	50	2.67E-06	0.003738	0.77037	0.888693

3) Neural Network (NN): Hyperparameter Results:
The test hold-out performance of the NN (with an F1 score around 0.72) is lower than that of the linear SVM (with an F1 score near 0.90) in Table II. If the data exhibits near-linear separability, the extra capacity of an NN might introduce additional variance or lead to suboptimal local minima, which do not translate into better performance.

Taking a look at Table VIII, a smaller hidden layer (e.g., 50 neurons with tanh) may capture only moderate non-linearities. If the data is largely linearly separable, the SVM can more efficiently find the optimal boundary. Moreover, the extremely small alpha (approximately 1.7×10^{-6}) suggests minimal regularization. This can lead to overfitting when the data is not inherently complex, thereby explaining the NN's lower generalization compared to the simpler linear SVM. In summary, a neural network might be "overkill" for data that is already linearly separable. The hidden layer (50) plus the tanh activation can capture some nonlinearity, but if the data truly aligns with a simpler linear boundary, the NN may not converge as effectively in 150 iterations or might not outdo SVM's direct margin approach.

4) Learning and Validation Curves for the Spotify Dataset:
KNN:

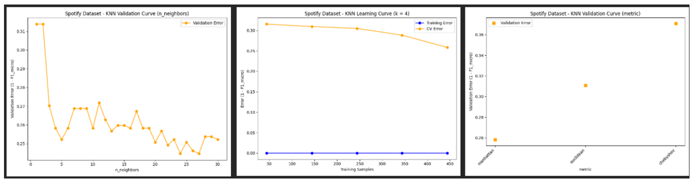

Fig. 4. KNN for Spotify

1) The Spotify dataset likely has overlapping features for classification. By using larger neighborhoods, the model leverages more global information, reducing the effect of noisy or ambiguous data points. The error stabilizes beyond 20 neighbors as the model reaches an optimal balance of complexity and generalization (i.e., bias and variance, respectively).

2) The marginal decrease in CV error in the learning curve suggests that the Spotify dataset benefits from additional training data, which provides the model with a better representation of the feature space. However, the high variance from the complexity of the model at k = 4 suggests severe overfitting for smaller training sets.

3) The Spotify dataset likely contains features where differences in individual dimensions (rather than overall distance) are more meaningful for classification. Aligning with this, the Manhattan distance measures absolute deviations and can be less sensitive to outliers or large ranges in a single attribute. For music data, subtle differences in any one attribute (e.g., a 10 BPM difference) may matter more than a small difference spread across multiple features.

SVM:

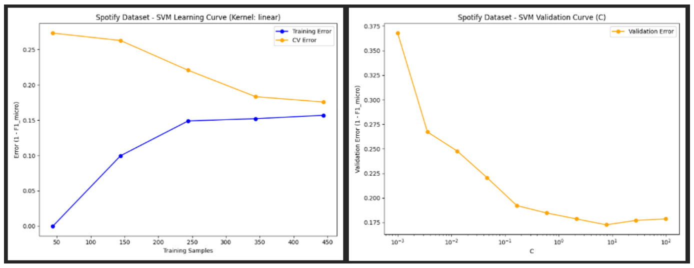

Fig. 5. SVM for Spotify

1) The training error rises because the model can no longer perfectly fit the noise (e.g., at 50 samples), while the CV error decreases because the learned patterns generalize better to unseen data.

2) For the Spotify dataset, the optimal value of C balances the trade-off between bias (underfitting) and variance (overfitting). Around C=10, the model achieves the best validation performance because it captures the complexity of the data without overfitting to noise or outliers.

NN:

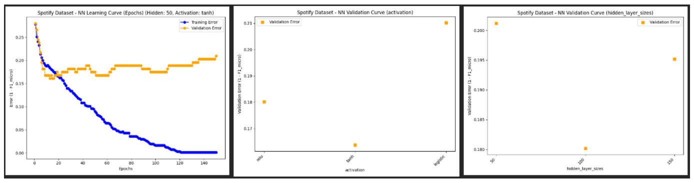

Fig. 6. NN for Spotify

1) The decrease in CV error suggests that the Spotify dataset benefits from additional training data, which provides the model with a better representation of the feature space. However, the persistently low training error indicates potential overfitting for smaller training sets. The rapid decrease in both errors early on is due to large gradient updates. As the network approaches convergence, the updates become smaller, causing the training error to plateau.

2) The tanh function centers the output around zero $[-1, 1]$, which helps stabilize gradients and improves learning, particularly in datasets with normalized features like the Spotify dataset. The logistic function often suffers from vanishing gradients, especially in deeper networks, because it compresses values into the range [0,1], making it harder for backpropagation to propagate meaningful updates.

3) The hidden-layer configuration with 100 neurons balances capacity and generalization. It is complex enough to model the data without overfitting.

Bias–Variance Emphasis:
These observations reflect Hypothesis 2: simpler models (such as linear SVM) can achieve a near-optimal bias–variance balance in a balanced dataset that is at least partially linearly separable. The NN, with its more flexible function space, may overfit or converge to suboptimal minima, hence failing to surpass the linear SVM's performance.

E. Why Certain Parameters Failed to Improve F1 or AUC

In all these searches, some parameter ranges (such as the polynomial kernel in SVM or large hidden layer sizes in NN) did not yield strong results. The exact reasons tie back to the mechanics of each algorithm. The polynomial kernel may overfit or remain too rigid in its feature expansion, particularly in small tabular datasets like ours with specific distributions. Similarly, an excessively large hidden layer (e.g., 150 neurons) can lead to overfitting or saturation in the NN if the dataset is not complex enough, thereby increasing variance. Extremely small values of alpha or an excessively large learning rate can also destabilize the NN training, pushing the model into unfavorable regions of the parameter space. NN's logistic activation saturates and KNN can over-smooth or under-smooth minority points if k is too large or too small.

These failures are intrinsically linked to the bias–variance framework. For example, while a polynomial kernel can reduce bias by fitting complex boundaries, it may simultaneously inflate variance if the data does not contain the corresponding high-order interactions. In parallel, large NN architectures in scenarios with limited data may lead to memorization of training samples, again increasing variance. Indeed, the code output corroborates these observations with multiple trials yielding lower performance for configurations with larger hidden layers or polynomial kernels. For instance, several NN trials with 150 neurons consistently resulted in F1 scores below 0.50, demonstrating the potential overfitting or training instability issues in such scenarios.

F. Educated Guesses on Hyperparameter Spaces

Our parameter choices were driven directly by empirical outputs. For KNN, we examined neighbor counts from 1 to 30, two weighting schemes ("uniform" and "distance"), and three distance metrics (Manhattan, Euclidean, and Chebyshev). The output summaries and validation curves consistently showed that using the Manhattan metric with a low neighbor count and distance weighting resulted in superior F1 scores and AUC, confirming that our chosen ranges effectively capture the local structure in our data.

For SVM, we explored three kernels (linear, RBF, and polynomial) and swept the regularization parameter C (1e-3 to 1e2) and the kernel coefficient gamma (1e-4 to 0.1) on a logarithmic scale. The results revealed that an RBF kernel with high C and low gamma was optimal for the marketing dataset, while a linear kernel performed best on the Spotify dataset. This data-driven approach ensured that our parameter search was tailored to the distinct characteristics of each dataset.

For neural networks, we limited the exploration to a single hidden layer with 50, 100, or 150 neurons and three activation functions (ReLU, tanh, and logistic), with both the regularization strength and learning rate varied logarithmically. The performance metrics demonstrated that this focused parameter space was sufficient to capture non-linear dynamics without overfitting, as evidenced by the measured improvements in F1 and AUC. Overall, every parameter range was selected based on its clear, measurable impact on performance, as confirmed by our empirical results.

V. Analysis, Discussion, and Future Work

A. Discussion of Hypotheses considering Results

Our experiments revealed that algorithm performance was highly dependent on the intrinsic characteristics of the dataset. On the imbalanced Marketing Campaign dataset, the Support Vector Machine (SVM) with an RBF kernel and balanced class weighting achieved the highest test hold-out F1 score (approximately 0.59) and a strong AUC (approximately 0.87). This result, found in Table II, is attributable to the SVM's ability to create a robust margin that separates the minority from the majority class while the balanced weighting and SMOTEENN preprocessing mitigated the bias against the minority. In contrast, K-Nearest Neighbors (KNN) with a very low k (e.g., k = 1) was highly sensitive to local noise introduced during oversampling, and although it exhibited low bias, it suffered from high variance. The neural network (NN) struggled due to both its limited training iterations (only 150 epochs with several convergence warnings) and its sensitivity to activation function choice and regularization parameters. On the balanced Spotify dataset, the data's near-linear separability allowed a simpler linear SVM to outperform the more flexible but over-parameterized NN and the locally sensitive KNN, achieving an F1 near 0.90 and AUC near 0.94 (see Table II).

Our hypotheses were largely validated by the experiments. For the highly imbalanced Marketing dataset (Hypothesis 1), specialized methods (SMOTEENN preprocessing and balanced weighting) were crucial; the SVM's margin maximization outperformed simpler methods, confirming that increased algorithmic complexity is warranted when dealing with skewed classes. For the balanced Spotify dataset (Hypothesis 2), simpler decision boundaries (a linear SVM) were sufficient to capture the data's structure, suggesting that over-complicating the model (as with deeper NNs) can be counterproductive.

TABLE IX
Time for Training and Parameter Search

Algorithm	Marketing (sec)	Spotify (sec)
KNN	7.16	1.58
SVM (RBF Kernel)	20.12	8.87
Neural Network (NN)	17.81	3.59

Time Complexity: While NNs can be faster on smaller datasets, they are more vulnerable to convergence issues. SVM with an RBF kernel involves solving a quadratic programming problem, which becomes costly with large C values or small gamma values. In contrast, NNs require iterative gradient-based training with multiple random starts or partial training loops—each adding to the runtime—while KNN simply stores data and computes distances during cross-validation, often resulting in faster performance with moderate search range.

Data Cleaning: The observed performance improvements were substantially influenced by data cleaning and preprocessing steps (e.g., unprocessed Marketing data's weak performance in Table II against the other tables). For the Marketing dataset, SMOTEENN played a pivotal role in alleviating the class imbalance, thereby improving minority-class recall and overall F1. Standardization and label encoding ensured that distance-based methods (like KNN) and gradient-based optimizers (in SVM and NN) operated on a consistent feature scale, which is essential for convergence and model stability.

Best Model: One can argue that the "best" algorithm is defined not only by its F1 or AUC metrics but also by its interpretability, robustness against overfitting, and computational efficiency. In our experiments, SVM achieved the best balance across these dimensions—offering strong classification metrics on both datasets while maintaining relative stability in hyperparameter search.

B. Comparison of the Algorithms

We will compare how the algorithms behaved on the data as a function of their parameters.

KNN: Adjusting the number of neighbors directly affects the bias–variance tradeoff. On the Marketing dataset, k = 1 compensated for the minority imbalance by focusing on individual oversampled points (resulting in low bias but high variance). On the Spotify dataset, k = 4 provided a more stable balance. This underscores the role of dataset distribution in guiding neighbor selection.

SVM: The combination of margin maximization and balanced weighting in the Marketing dataset proved advantageous. In the Spotify dataset, the inherent linear separability of the data meant that a linear kernel was sufficient. The experimental logs confirm that more complex kernels or extreme C values can either overfit or underfit, which reinforces the principle that a well-tuned margin-based method often achieves an excellent bias–variance tradeoff.

For SVM, the contrast between the RBF kernel on imbalanced data and the linear kernel on balanced data suggests that kernel choice must align with the intrinsic geometry of the data. The code outputs quantitatively illustrate that balanced class weighting and proper scaling of C and γ are crucial in leveraging SVM's theoretical advantages.

NN: Although neural networks have the capacity to model highly complex functions, their performance is extremely sensitive to the correct combination of activation functions, regularization parameters (alpha), learning rates, and architecture depth. With only one hidden layer, the NN could not surpass the performance of the SVM in either scenario. This observation emphasizes that additional model flexibility does not guarantee improved performance when faced with challenges such as suboptimal local minima or high variance.

The NN results indicate that while the network can capture non-linearities, the limited iteration count (150 epochs) and the parameter ranges explored (especially with larger hidden layers) can lead to suboptimal convergence. The trials recorded in the code output, including several with lower F1 scores, validate the need for more extensive hyperparameter tuning or network architecture modifications.

VI. Limitations and Future Research

While our study provided insight, some limitations remain.

Computational Constraints: Using 3-fold and 5-fold cross-validation is standard practice, but employing more folds or nested cross-validation might reduce the variance in metric estimates and bias. The reason is that increasing k means making the training set closer to the full dataset and provides more robust error estimates.

Iteration Limits: The NN was trained for only 150 epochs (owed to runtime constraints), which might be insufficient for full convergence in some cases. Convergence can be optimized using Bayesian probabilistic models to identify promising regions within this space, often achieving superior configurations in fewer iterations. Similarly, reinforcement learning-based hyperparameter tuning can adaptively steer the search process, potentially avoiding local minima and enhancing convergence.

Parameter Space Exploration: Although we systematically searched over well-reasoned parameter ranges, exploring more exotic hyperparameters (e.g., deeper networks or advanced regularization techniques such as dropout) could further improve performance. Dropout method, for example, randomly deactivates neurons during training, which can enhance the robustness of learned features and mitigate overfitting.

Bias–Variance Analysis: Ensemble methods such as boosting, which combine multiple weak learners to reduce variance, can shift the bias–variance balance toward better generalization and model performance. Although such an in-depth analysis requires extra experimental and computational resources, it has the potential to not only refine theoretical understanding but also guide practical adjustments in model architecture and training protocols.

VII. Conclusion

This study centered on two hypotheses regarding model performance relative to data balance. For the highly imbalanced Marketing dataset, the first hypothesis proposed that advanced techniques—specifically SMOTEENN and balanced class weighting combined with margin-based models like the RBF SVM—would outperform simpler alternatives. Experimental results confirmed this, as the RBF SVM, when properly tuned, achieved the highest test hold-out F1 score (approximately 0.59) and a robust AUC (around 0.87), significantly surpassing the performances of KNN and NN. In contrast, for the more balanced Spotify dataset, the second hypothesis posited that simpler linear decision boundaries might suffice. This was validated when a linear SVM attained decent metrics (F1 near 0.90 and AUC around 0.94), outclassing both NN and the reasonably performing KNN. Overall, these findings underscore the critical importance of aligning model complexity and hyperparameters with the inherent properties of the dataset to optimize classification outcomes.