Supervised Learning for Hit-Song and Customer Ad-Responsiveness Prediction

Shawn Egan
Georgia Institute of Technology
segan3@gatech.edu

I. Introduction

Three supervised learning algorithms are applied to two separate datasets to evaluate performance on two distinct classification tasks. The three algorithms are: Support Vector Machine (SVM), k-Nearest Neighbor (kNN), and Multi-layer Perceptron (MLP) or Neural Network (NN). Select parameters of these algorithms are investigated for both classification tasks. The algorithms are compared on classification performance and run time.

A. Spotify Classification Task

The first task involves using a multi-class classifier to predict the "size category" of a hit-song. "Size category" is defined by the number of streams; songs are grouped together if they have a similar number of streams. Five different streaming categories are defined. The dataset provides information about individual songs such as: artist, artist count, release date, number of streams, key, metrics on dance-ability and energy, and more. Additional features such as number of times the biggest artist on the song appears on the list were developed for the models to utilize in training. Data cleaning and analysis steps are included in Appendix A.

This task is interesting from a Machine Learning perspective because predicting song popularity is a non-trivial task [1]. If there were a way to predict the popularity of songs from analysis of the song itself, the music industry would be hyper-focused on optimizing characteristics of every song in order to generate "hit" songs.

B. Customer Classification Task

The second task involves using a binary classifier to predict whether a customer of a specific company is responsive to advertising campaigns given their history with the company and additional demographic information. Responsiveness to ads is based on the customer's responses to the previous six advertising attempts. The dataset provides information about individual customers such as: marital status, birth year, household income, amount spent on different product categories, and more. Additional features such as total purchases, total spent, spender category, and average purchase price were developed for the models to utilize in training. Data cleaning and analysis steps are included in Appendix A.

This task is interesting from a Machine Learning perspective because companies have been trying to predict and model consumer behavior since at least the 1940s [2]. Since the dawn of the digital age, companies have been using data-driven marketing to enhance engagement and Return On Investment. Predicting consumer behaviors is a non-trivial task [3].

C. Hypotheses

Hypothesis 1: *The MLP will outperform the SVM in the Customer classification task.*

Justification

Since the cleaned Customer data has 2208 samples, the MLP should be able to learn effectively given this sample size. In comparison, SVMs typically perform well with smaller datasets [4]. The Customer data has 30 features which should allow the MLP to generalize effectively given that the network is not too large and has proper regularization to avoid over-fitting the training data.

Hypothesis 2: *The SVM will outperform the MLP in the Spotify classification task.*

Justification

As stated in the justification for 1, SVMs perform better with smaller datasets; they also perform well with high-dimensional data [4], [7]. The Spotify classification task uses 32 features with only 815 samples. Figs. 3 and 6 of [7], show the performance of an MLP, linear SVM, and polynomial SVM on classification tasks as a function of both feature size and sample size. The SVM outperforms the MLP for a sample size of 200 and feature size of 30 for linear synthetic data with both correlated and uncorrelated features. The SVM also outperforms the MLP on a real classification task for a sample size of 40 and feature size of 30. The results from [7] support hypothesis 2.

Hypothesis 3: *The SVM and MLP will outperform the kNN on both classification tasks*

Justification

The Spotify dataset has 32 features, and the Customer dataset has 30 features. KNN models suffer from "the curse of dimensionality:" the amount of data required to generalize accurately grows *exponentially* as the number of features increase. Since SVMs and MLPs are equipped to handle a large number of features, and kNNs require exponentially more data for generalization, kNNs should under-perform the other two classifiers.

II. Methods

All 6 models are developed and tuned using the tools contained in the Python library Scikit-Learn.

A. Model Scoring

The Spotify classification models are scored using macro-averaged F1 score; the macro-average is used to weight all 5 classes equally. F-score is chosen as it is a good indication of overall classification performance, and F1 is chosen to weight precision and recall equally for all classes.

The Customer classification models are scored on positive-class (1) recall. This scoring metric is chosen for the Customer task because, in theory, the goal of the models for this task would be to identify the largest portion of customers that will respond to ads. This would be the goal for any marketing department to target advertisements more effectively.

B. Hyper-parameter Search

A hyper-parameter grid search with 5-fold cross validation is performed for all 6 models using the respective scoring metric for each task.

1) MLP

For the MLP Classifier, the initial search space includes hidden layer size, activation function, and regularization parameter (α). Initial hidden layer sizes are based on inspection of the predictor and target variables of the corresponding dataset. The standard activation functions ReLU, Identity, Logistic, and Tanh are included, and a standard range of $(10^{-4}, 10^{-1})$ is used for α.

The model for the Spotify data will utilize an input layer of 32 neurons and an output layer of 5 (one for each class), therefore the hidden layer search space is composed of networks with first layers of 8 or 16 neurons, final layers of 8 neurons, and depths of 2 or 3 layers to avoid over-fitting the smaller dataset.

The model for the Customer data has an input layer of 30 neurons, and an output layer of 1 neuron (binary classification), so the hidden layer search space is composed of networks with decreasing layer widths (8x4x2, 16x8x4, etc.), with first layers of 8-16 neurons, last layers of 2-8 neurons, and depths of 3 to 4 layers. The intuition for the decreasing layer widths comes from the fact that the final output layer of a single neuron will need to incorporate the output of every neuron in the last hidden layer, and therefore the last hidden layer should be small as to not overwhelm the output neuron with information. Additionally, the network needs to refine from 30 input features down to a single output neuron, necessitating the halving of layer widths to avoid networks that have too many layers. Avoiding deeper networks is necessary due to the Vanishing Gradient Problem [5].

A final hyper-parameter search is performed after parameter validation only for the MLP classifier, since the MLP has multiple additional parameters to optimize. Solver algorithm (lbfgs and adam), learning rate ($(10^{-4}, 10^{-2})$), and learning rate scheduling (constant, inverse scaling, and adaptive) are included in the final hyper-parameter search space.

2) SVM

A hyper-parameter search is performed for both SVM classifiers with a parameter space including kernel (linear, polynomial, radial basis function (RBF), and sigmoid), regularization parameter (C), and, in the case of the polynomial kernel, the degree of the polynomial kernel.

3) kNN

A hyper-parameter search is performed for both kNN classifiers with a parameter space including number of neighbors (k), algorithm (ball tree, kd tree, and brute force), weight type (uniform or distance), Minkowski distance metric power (p), and leaf size (for the tree algorithms). Number of neighbors is searched in the range [3,53). The 'uniform' weights treat all neighbors with equal weight in calculation of the predicted class, whereas 'distance' weights use the inverse of the distance to the neighbor as the neighbor weight. Distance weights give more weight to closer neighbors. Distance metric power values in the hyper-parameter space are 1, 2 and 3. The Minkowski distance metric is defined in [6].

C. Validation Curves of Select Hyper-parameters

After the initial hyper-parameter search, the performance of all 6 models is evaluated with 5-fold cross validation for selected ranges of hyper-parameters to investigate the influence that different parameters have on model performance. For the MLP Classifiers, the depth of the network and learning rate of the model are selected for investigation. For the SVM models, the degree of a polynomial kernel and the regularization parameter are selected. Finally, for the kNN models, the k-value and degree of the distance metric are selected for investigation.

D. Evaluation

1) Training Evaluation

Each model is trained with 5-fold cross validation on the full training set for two different activation functions, kernels, and k values for MLP, SVM, and kNN respectively.

2) Testing Evaluation

Classification reports including macro-average (Spotify task) and positive class (Customer) precision, recall, F1, and AUC are generated for all six models.

III. Results

All plots shown utilize an error rate of $1 - \text{F1}_{\text{macro}}$ for the Spotify multi-class classification task, and an error rate of $1 - \text{Recall}_1$ for the Customer binary classification task. All plots were generated with the Python library Matplotlib.

A. MLP

After hyper-parameter tuning, the optimal MLP classifiers have the characteristics shown in Table I.

TABLE I: MLP Hyper-parameters

Dataset	Layers	Activation	α	Batch Size	Time
Customer	(32,16,8,4)	Logistic	0.001	200	6m 44s
Spotify	(8,8,8)	ReLU	0.1	64	8m 39s

1) Network Depth

The blue and orange lines in Fig. 1 are for networks of width 8 and layers of depth 1 to 10 trained on the Spotify dataset. The red and green lines in Fig. 1 are for networks of width 32 and layers of depth 1 to 8 trained on the Customer dataset.

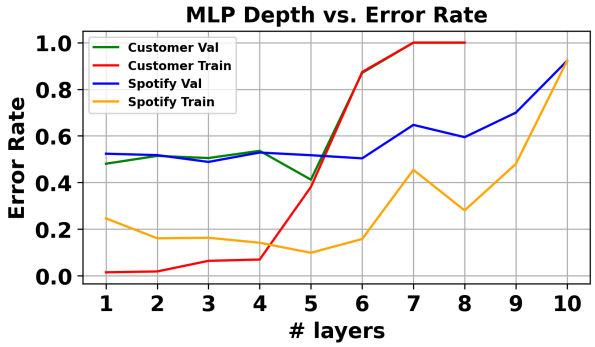

Fig. 1: Performance of MLPs on Spotify and Customer data for networks of increasing layer depth

2) Network Learning Rate

Fig. 2 shows the performance of both MLP classifiers as a function of learning rate in the range $(10^{-5},1)$. Vertical lines showing the optimal learning rates for both classifiers are given.

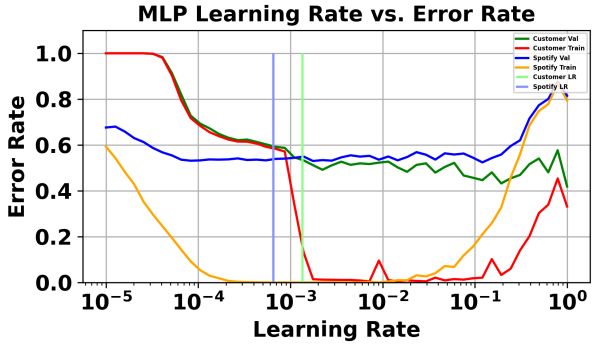

Fig. 2: Performance of MLPs on Spotify and Customer data for varying learning rate

3) Network Learning Curves

The learning curves for the Spotify classification problem took 3 minutes to generate, and for the Customer classification problem took 2 minutes and 15 seconds to generate.

The top plot of Fig 3 shows the learning curves of the Spotify MLP for the optimal (ReLU) activation function as well as the same optimal network with the Identity activation function. The bottom plot of the figure shows training time as a function of training size for both activation functions.

Fig. 3: Performance of MLP on Spotify data for different learning rates

The top plot of Fig 4 shows the learning curves of the Customer MLP for the optimal (Logistic) activation function as well as the same optimal network with the ReLU activation function. The bottom plot of the figure shows training time as a function of training size for both activation functions.

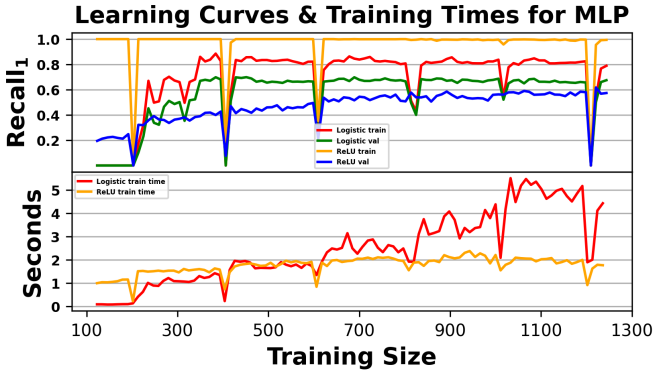

Fig. 4: Performance of MLP on Customer data for different learning rates

B. SVM

After hyper-parameter tuning, the optimal SVM classifiers have the characteristics shown in Table II.

TABLE II: SVM Hyper-parameters

Dataset	Kernel	Regularization (C)	Time
Customer	RBF	8.57	17s
Spotify	RBF	4.21	10s

1) Polynomial Kernel Degree

Fig. 5 shows error rate as a function of polynomial kernel degree for both SVM classifiers.

2) RBF Kernel Regularization

Fig. 6 shows error rate as a function of SVM Regularization magnitude for both SVM classifiers.

3) Learning Curves

The learning curves took 2.7s (Spotify) and 4.2s (Customer) to generate. Fig. 7 shows training performance

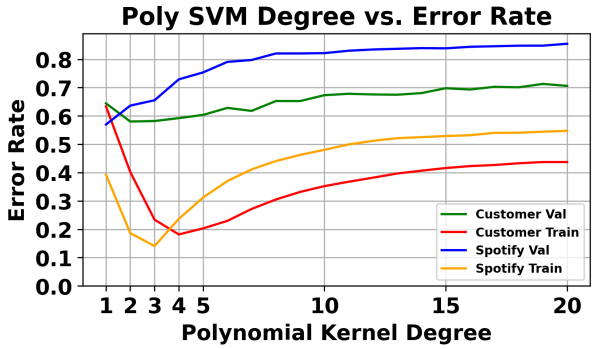

Fig. 5: Performance of polynomial-kernel SVM on Spotify and Customer data for different polynomial degrees

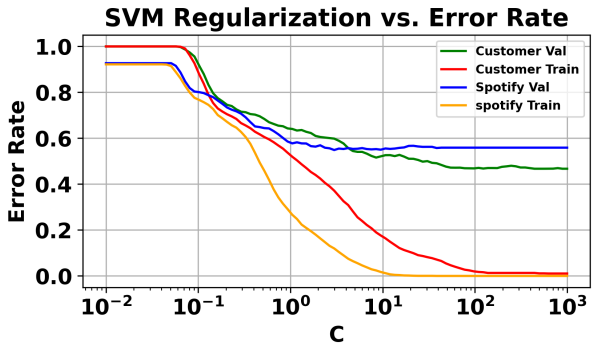

Fig. 6: Performance of RBF-kernel SVM on Spotify and Customer data for varying regulation parameter

on the Spotify classification task as a function of training size for linear- and RBF-kernel SVMs.

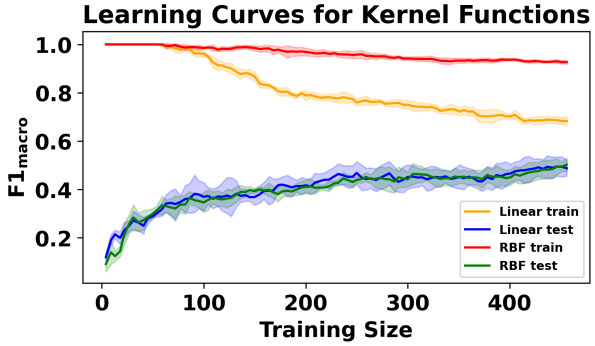

Fig. 7: Linear- and RBF-kernel SVMs training size vs $F1_{macro}$ for the Spotify classification task

Fig. 8 shows training performance on the Customer classification task as a function of training size for linear- and RBF-kernel SVMs.

C. kNN

The optimal kNN parameters from hyper-parameter tuning are shown in Table III.

1) Number of Neighbors

Fig. 9 shows training performance of the kNN classifiers for both classification tasks as a function of number of neighbors.

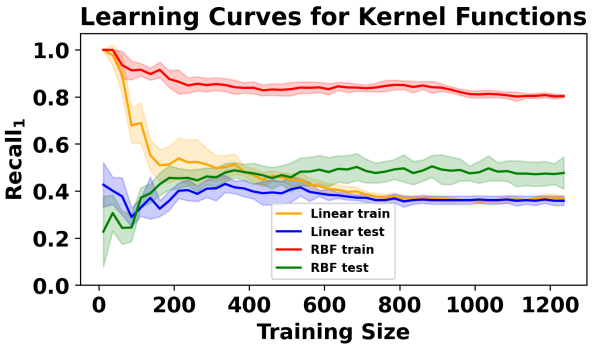

Fig. 8: Linear- and RBF-kernel SVMs training size vs positive recall for the Customer classification task

TABLE III: kNN Hyper-parameters

Dataset	Algorithm	k	p	weights	Time
Customer	ball tree	4	1	distance	2m 52s
Spotify	ball tree	14	1	distance	43s

2) Distance Metric

Fig. 10 shows the effects of distance metric power-parameter on training and validation for both classification tasks. The default distance metric used is the Minkowski metric.

3) Learning Curves

The learning curves took 0.2s (Spotify) and 2.3s (Customer) to generate. Figs. 11 and 12 show the comparison of optimal k-value and k=53 on both Spotify and Customer classification tasks. All learning curves start at a training size of 15% of the full training set, due to kNN needing at least k training examples for inference.

D. All Model Performance Scores

Table IV shows the macro-averaged precision, recall, F1 and AUC scores of all 3 models for the Spotify classification task. Table V shows the precision, recall, and F1 scores for all 3 models on the Customer classification task. The highlighted columns, $F1_{macro}$ and $Recall_1$ identify the scoring parameter that the models are optimized on.

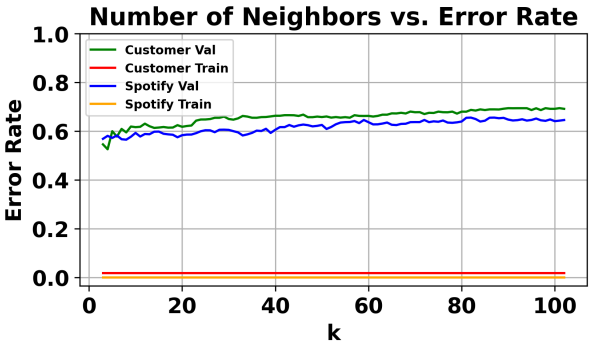

Fig. 9: Effect of k-value on training and validation error for the Spotify and Customer classification tasks

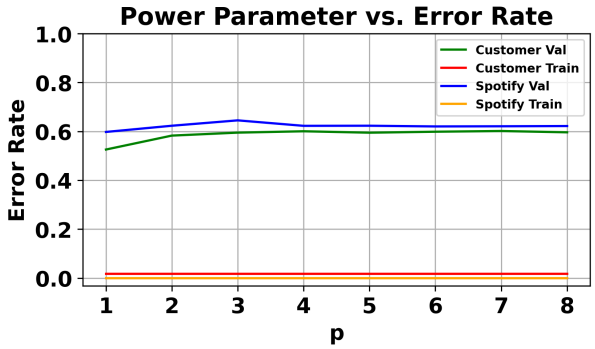

Fig. 10: Effect of distance metric power-parameter on training and validation error for the Spotify and Customer classification tasks

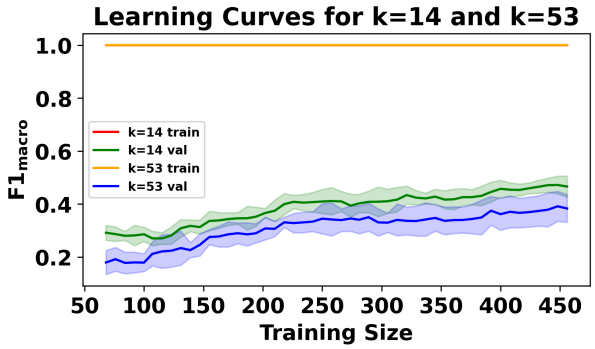

Fig. 11: Learning curves for optimal k and k=53 for the kNN classifier on the Spotify classification task

IV. Discussion

A. MLP

From Table I, the Spotify MLP utilizes a smaller architecture, larger α, and batch size of 64 in order to reduce over-fitting the smaller dataset. Larger α means more regularization, which reduces over-fitting. Smaller batch size has the same effect as larger α. The Spotify MLP also uses a learning rate an order of magnitude lower than the Customer MLP. The Spotify MLP is able to use a smaller learning rate due to the smaller batch size. Smaller batch sizes enable gradient descent to find "broader local minima," and smaller learning rate enables the gradient descent to search more of the optimization surface [8].

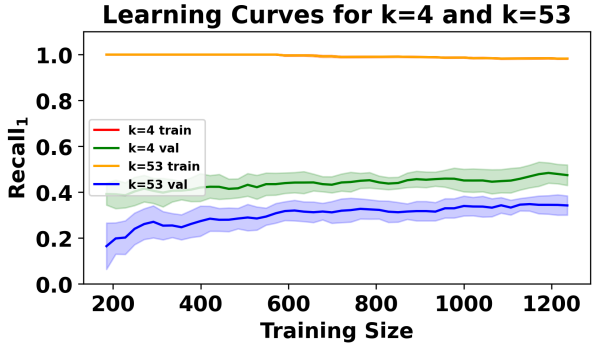

Fig. 12: Learning curves for optimal k and k=53 for the kNN classifier on the Customer classification task

TABLE IV: Spotify Classification Report

Model	Precision_{macro}	Recall_{macro}	F1_{macro}	AUC
MLP	0.52	0.51	0.50	0.80
SVM	0.53	0.54	0.52	0.83
kNN	0.44	0.42	0.41	0.78

TABLE V: Customer Classification Report

Model	Precision_1	Recall_1	F1_1	AUC
MLP	0.58	0.70	0.63	0.82
SVM	0.69	0.56	0.62	0.84
kNN	0.69	0.51	0.59	0.80

1) Network Depth

For the Spotify MLP in 1, Network depth does not change model validation performance up to a depth of 6 layers. Networks of depth 7 and 8 layers decrease validation performance. Networks of depth 10 layers and beyond do not provide any predictive ability. For the Customer MLP in 1, Network depth increases validation performance up to a depth of 5 layers. Networks of depths 6 and greater do not provide any predictive ability.

The Spotify MLP networks use layers of width 8, while the Customer networks use layers of width 32. This is the primary cause of the difference between network depth effects on the two models (Customer not learning beyond 5 layers vs. Spotify not learning beyond 10 layers). The Customer networks already have 4x the neurons in each layer, therefore adding layers to the Customer dataset results in 4x the model complexity. The inability for deeper networks to learn at all is also a known issue called the "Vanishing Gradient Problem." As networks deepen, the early layer gradients decrease in magnitude exponentially, which can result in instability and inability to learn in the worst cases[5].

2) Learning Rate

As shown in Fig 2, there are specific ranges of learning rate where the MLP learns for both classification tasks. For the Spotify data (blue and orange lines), the range is roughly $(10^{-4}, 10^{-1})$, and for the Customer data (red and green lines) the range is roughly $(10^{-3}, 1)$. The range for the Spotify MLP is roughly 1 to 2 orders of magnitude wider than that for the Customer data. Additionally, the Customer MLP sharply spikes on both ends of the graph, meaning the learner is very sensitive to learning rate. This preference for a specific range of learning rates is due to the relative size of the network in comparison to the Spotify MLP. The Customer MLP uses 2.5x the number of hidden neurons (60 vs 24) and an additional hidden layer. This results in over 3.8x more weights for the Customer MLP (1636 vs 424, including input/output layers). This additional size of the optimization space makes the Customer MLP much more sensitive to learning rate, as smaller or larger learning rates make traversing the optimization surface and

finding global minima more difficult. Smaller learning rates slow down optimization, and larger learning rates may miss optimization minima all together.

3) Learning Curves

a) Spotify Model

From Fig. 3, the Identity and ReLU activation functions have very similar validation performance. The Identify activation function's training curve shows a steeper decline in performance as training size increases, and the ReLU function shows a very small decline in training performance. The trajectory of both validation curves show that the model generalizes well (low bias and variance), and would likely perform better with more data.

The ReLU function is the same as the Identity function for positive values, but 0 for negative values. The Identity function makes learning complex tasks hard because most of the neurons are active in the network (positive or negative). The ReLU generates a more sparse network since only positive neurons are active, meaning it is able to learn more complex tasks with less over-fitting [9]. This is why the ReLU function seems more over-fit than the Identity function (ReLU has higher training scores with the same validation scores): the ReLU has a simpler network after training and is more-robust to over-fitting.

In addition, the Identity function is 4x faster than the ReLU function for training on the full training set. This shows the benefit of the Identity function since there is no activation to compute; the neuron output is simply a linear combination of the inputs with the weights.

b) Customer Model

From Fig. 4, the ReLU classifier displays 100% positive recall on the training set for the entire range of training sizes, whereas the Logistic classifier sharply jumps to over 80% positive recall for training sizes under 400 instances. The Logistic classifier also sharply increases validation set performance below 400 training set size. Both training and validation performance for the Logistic classifier stay around 80% and 65%, respectively, once training size passes beyond 400 instances. The ReLU classifier shows a much more gradual increase in validation performance over the full range of training set sizes, ending up with around 58% positive recall for the full training set. The trajectory of both validation curves again show that the model generalizes well (low bias and variance), and would likely perform better with more data.

In addition, the ReLU function is 2.5x faster than the Logistic function for training on the full training set. The Logistic activation function is more complex to compute than the simple maximum computation for the ReLU activation function, leading to this discrepancy.

B. SVM

1) Poly-kernel Degree

As shown in Fig. 5, a degree-1 polynomial (linear) kernel is optimal to avoid over-fitting for the Spotify classification task. The degree-2 polynomial kernel exhibits over-fitting since the training error decreases but the validation error increases. Degree-3 and larger polynomial kernels exhibit worsening training and validation performance as degree increases.

As shown in Fig. 5, a degree-2 kernel is optimal for the Customer classification task. Training error decreases until degree-2 polynomial kernel, and for degrees 3 and greater the training and validation error increase.

The linear-kernel preference for the Spotify task suggests the data are linearly separable. This is likely due to the incorporation of the correlated predictors listed in the Appendix A; the SVM can find an optimal hyperplane that separates the data based on charting and playlist numbers. The quadratic kernel preference for the Customer task suggests the data are not linearly separable.

2) Regularization

Fig. 6 shows that regularization has a large impact on training and validation performance for both classification tasks. Regularization below 0.1 exhibits little to no learning ability for both classification tasks. Regularization in the range $(0.1, 9)$ (Customer) and $(0.1, 5)$ (Spotify) result in models that are not over-fit. Regularization in the range 9 (Customer) and 5 (Spotify) to 20 display over-fit models as the training error continues to decrease while validation error stays relatively constant. Regularization beyond 100 shows no performance improvements for training or validation in either classification task. The Customer SVM classifier requires double the regularization of the Spotify task due to the larger dataset; more data means more opportunity to over-fit the training data. Regularization is one way to reduce over-fitting for the RBF-kernel SVM.

3) Learning Curves

a) Spotify Model

As seen in Fig. 7, linear- and RBF-kernel functions have very similar validation performance over the full range of training sizes for the Spotify classification task. The linear-kernel SVM displays a sharper decrease in training set performance for training sizes between 100 and 200 instances and a slow decline in training performance beyond 200 instances. The RBF-kernel shows a very slow decrease in training performance beyond training sizes of 100 instances. The trajectory of the validation curves shows that the SVM will benefit from more data for this classification task, the model generalizes well.

Linear- and RBF-kernels have similar validation performance for the Spotify classification task. The RBF-kernel is unable to outperform the linear-kernel due to the data being linearly separable, as was concluded in Section IV-B1.

b) Customer Model

As seen in Fig. 8, the RBF-kernel SVM outperforms the linear-kernel for training sizes over 100 instances for the Customer classification task. The RBF-kernel shows a sharp increase in validation performance up to training sizes of 200 instances, and the linear-kernel

shows a decrease, then increase, then slow decline in validation performance over the range of training sizes. The linear-kernel displays a sharp decrease in validation performance below 200 instances, whereas the RBF-kernel shows a slow decline in training performance over the full range of training sizes. Both validation curves flatten out around a training size of 600 samples, meaning the SVM for the Customer classification task would not benefit from additional data.

The RBF-kernel outperforms the linear-kernel for the Customer classification task. This confirms the conclusion in Section IV-B1: the Customer data are not linearly-separable. The Radial-Basis Function incorporates the squared Euclidean distance in the calculation, incorporating a form of distance calculation similar to kNN classifiers into the model.

C. kNN

1) Number of Neighbors (k)

Fig. 9 shows zero training set error rate for both classification tasks, as is expected for a kNN classifier utilizing distance weighting since it utilizes all training data to make predictions. The Customer kNN validation error increases slowly beyond k=4, and the Spotify kNN validation error increases slowly beyond k=14. The Spotify kNN utilizes a larger k-value due to the smaller dataset size in combination with the multi-class classification task. The space of training instances is more sparse than the Customer data, and there are more classes that get a "vote" for the Spotify task, so more examples are needed to get an accurate prediction of the class of the instance. The large data size and binary-nature of the Customer classification task allows the kNN to make accurate predictions while only taking into account the nearest 4 instances.

2) Minkowski Distance Metric

From Fig. 10, the power parameter of the Minkowski distance metric has very little effect on model performance above $p = 4$ in both classification tasks. When $p = 1$ the Minkowski metric becomes the Manhattan distance, and when $p = 2$ the Minkowski metric is the standard Euclidean distance [6]. As shown in the figure, $p = 1$ is optimal for both classification tasks. Error rates jump slightly for $p = 2, 3$, and then stay mostly constant for $p > 3$. Power parameters larger than 4 have no effect on validation error rate due to the limit as $p \to \infty$ from [6]: as p grows, the distance approaches the maximum component distance of the vectors under consideration. When one component distance is much larger than all others, the Minkowski distance is approximated by that maximum component distance. For $p = 3$, the distance will already be dominated by the cube of the largest component distance, therefore $p > 3$ does not effect the distance calculation when finding nearest neighbors.

3) Learning Curves

Smaller k-values perform better in both classification tasks. Small k-values are more prone to over-fitting when data are noisy. The learning curves for the Spotify classification task in Fig. 11 show a more gradual increase in validation score up to around 50% training size, whereas the learning curves for the Customer classification task in Fig. 12 reach near-maximum validation scores around 20% training size. This difference in learning speed is due to the relative size of the datasets: the Customer dataset is larger and therefore has more examples to draw from when making inferences, therefore the validation error reaches near-maximum performance at smaller relative training size. The kNN model for the Customer classification task would not benefit from additional data, as the validation curves are nearly flat for the majority of the larger training sizes. The kNN model for the Spotify classification task may benefit from additional data, as the validation curves climb steadily for the full range of training sizes.

D. Model Comparison

From Tables IV and V, the MLP classifier beats the SVM and kNN classifiers on the Customer classification task for $F1_{macro}$ scoring. The SVM beats the MLP and kNN classifiers on the Spotify classification task for $Recall_1$ scoring. These results confirm Hypothesis 1 and Hypothesis 2. The SVM is more equipped for the smaller Spotify dataset, and the MLP is able to perform better on the larger dataset for the Customer classification task. Hypothesis 3 is proven by the result that the kNN classifier scored lowest on both classification tasks. All AUC scores in both tables (> 0.78) prove that all classifiers are able to learn from the data; all scores are significantly better than a random AUC of 0.5.

E. Conclusion

The results from this investigation show the strength of a simple Multi-layer Perceptron for datasets with enough data (Customer). Even with relatively small datasets, MLPs can outperform kNNs and SVMs for classification tasks. More complex Neural Networks with dropout and more complex topologies could provide even better performance on the classification tasks. Additionally, the investigation of network depth for MLPs highlights the limitation of the vanishing gradient problem for Neural Networks. The investigation of learning rate for MLPs identifies the importance of a specific range for learning rate values in hyper-parameter search and network training. Learning rates too large or too small hinder training time and performance and may halt learning capabilities all together.

The comparison of the Identity and ReLU activation functions for the Spotify classification tasks highlights the stark difference between the two activation functions. ReLU provides a proxy-dropout step: neurons that would otherwise be negative for the Identity activation function do not have weights in the network. This proxy-dropout allows ReLU networks to avoid over-fitting data while outperforming Identity networks.

The MLP hyper-parameter searches took significantly longer than the SVM and kNN hyper-parameter searches. The kNN ran faster for the Spotify dataset due to the smaller amount of data. The kNN had the fastest learning curve generation time, followed by the SVM, with the MLP coming in last. The SVM shows remarkable results given the very fast hyper-parameter search time and training times.

The results also show the strength of SVMs for data with many features but few samples; SVMs can outperform the more complex MLPs on classification tasks with limited data (Spotify). The comparison of polynomial SVM degree illuminates the difference between linearly separable and linearly non-separable data. Linear-kernel SVMs can perform as well as RBF-kernel SVMs if the data are linearly-separable. RBF-kernels provide increaed performance for data that are linearly non-separable. Investigation of regularization's effects on SVM classifiers shows how regularization needs differ based on data size. For larger datasets (Customer), more regularization is needed to avoid over-fitting the training data. Smaller datasets (Spotify) are more robust to over-fitting and therefore have a smaller window of regularization parameter in which the model is over-fit.

The investigation of k-value for kNN algorithms shows how smaller datasets can use more neighbors without over-fitting the data. The investigation of the Minkowski distance metric illuminates the role that the distance metric has for kNN algorithms while showing that the power parameter has little effect on kNN performance for these datasets and classification tasks.

F. Limitations and Future Work

The Spotify classification results are limited by the relationship between chart and playlist features with streaming count: all 3 models scored highest on category 4 classification (streaming count over 700M) (results not included for brevity). Further investigation is needed to verify performance of the three algorithms on data that do not include charting and playlist numbers, which are directly influenced by streaming numbers.

V. Resources

[1] Merritt, S. H., Gaffuri, K., and Zak., P. J. "Accurately predicting hit songs using neurophysiology and machine learning". *Frontiers in Artificial Intelligence*, vol. 6. (2023). Accessed: Feb. 7, 2025. doi: 10.3389/frai.2023.1154663 https://www.frontiersin.org/journals/artificial-intelligence/articles/10.3389/frai.2023.1154663

[2] Wikipedia. "Consumer behavior". *Wikipedia*. (2024). Accessed: Feb, 6, 2025. https://en.wikipedia.org/wiki/Consumer_behaviour

[3] El-Hajj, M. and Pavlova, M. "Predictive Modeling of Customer Response to Marketing Campaigns". *Electronics*, vol. 13., num. 19. (2024). Accessed: Feb. 7, 2025. doi: 10.3390/electronics13193953 https://www.mdpi.com/2079-9292/13/19/3953

[4] Idrees, H."Neural Networks vs. Support Vector Machines (SVM): Which Model Should You Choose?". *Medium*. (2024). Accessed: Jan, 25, 2025. https://shorturl.at/wHQfV

[5] Wikipedia. "Vanishing gradient problem". *Wikipedia*. (2025). Accessed: Feb. 7, 2025. https://en.wikipedia.org/wiki/Vanishing_gradient_problem

[6] Wikipedia. "Minkowski distance". *Wikipedia*. (2025). Accessed: Feb, 6, 2025. https://en.wikipedia.org/wiki/Minkowski_distance

[7] Jianping, H., Xiong, Z., Lowey, J., Suh, E., and Dougherty, E. R. "Optimal number of features as a function of sample size for various classification rules". *Bioinformatics*, vol. 21., iss. 8. (2005). Accessed: Jan. 7, 2025. doi: https://doi.org/10.1093/bioinformatics/bti171 https://academic.oup.com/bioinformatics/article/21/8/1509/249540

[8] Yastremsky, D. "Exploit Your Hyperparameters: Batch Size and Learning Rate as Regularization." *TowardsDataScience*. (2021). Accessed: Feb, 8. 2025. https://shorturl.at/8CHFW

[9] GeeksForGeeks. "Activation functions in Neural Networks". *GeeksForGeeks*. (2024). Accessed: Feb. 7, 2025. https://www.geeksforgeeks.org/activation-functions-neural-networks/

[10] Nakamura, K. "ML LaTeX Template". *GitHub*. (2023). Accessed: Feb. 5, 2025. https://github.com/knakamura13/cs7641-ml-study-materials-2023/tree/main

Appendix

A. Data Cleaning

Columns with many categories are mapped to fewer distinct categories. Columns without unique values are dropped. 'TotPurchases,' 'TotSpent,' 'AvgPurchasePrice,' 'Age,' and 'SpenderCategory' are calculated for the Customer dataset. An additional column is derived for the Spotify data to represent the number of times that the biggest listed artist is found in the data. Artist name is then dropped from the data. Release day, month, and year columns are used to generate a new column called 'days_since_release.' The Spotify target variable is generated by grouping songs of similar streaming count together. The categories are $<$100M, 100M $<$ s $<$ 200M, 200M $<$ s $<$ 400M, 400M $<$ s $<$ 700M, $>$ 700M, where M = millions of streams, and s = # of streams. The target variable for the Customer dataset, 'RespondsToAds,' is 1 from if any of 'AcceptedCmpX' is greater than 0. There are 602 customers that responded to ads, and 1606 customers that did not respond to any ad campaign.

All numerical columns are then standardized, and categorical columns are one-hot encoded for each dataset. The resulting Spotify data has 815 rows and 33 columns, and the Customer data has 2208 rows with 31 columns. The correlation between all numerical columns in the Spotify dataset and the target variable are computed. Highest correlations with the streaming category are the playlist and chart columns (0.22 to 0.61), as well as days since release (0.26). Highest correlations for the Customer dataset with the ad campaign responsiveness are amount spent on wines (0.42), number of catalog purchases (0.30), amount spent on meat products (0.29), income (0.28) and number of web purchases (0.22).