Dynamic Feature-Adaptive SVM: Enhanced Learning for High-Dimensional Data Classification using Gradient Boosting

Rishabh Chhabra and Nikhil Nair

01/08/2024

Yilmaz - Period 7

Quarter 2 Project

Abstract

As machine learning advances, the challenge of handling high-dimensional data, especially as the technology is introduced to novel applications, has become ever-important. Traditional kernelized Support Vector Machines (SVMs), while extensive, suffer from overfitting and inefficiency when facing a high-dimensional dataset. This work addresses this challenge by introducing an adaptive SVM approach based on a dynamic feature selection method and adaptive feature weighting. Our methodology focuses on identifying and leveraging the most informative features, which reduces dimensionality and improves the accuracy of the model. Our model is tested using diverse datasets including musk.csv, diabetes.csv, and inosphere.csv, which have already been used for evaluating alternative adaptive and linear SVMs. We evaluated the accuracy and effectiveness of our approach in mitigating overfitting by comparing its accuracy, precision, sensitivity, and AUC values on these datasets with previous adaptive solutions. The results highlight the potential of adaptive feature selection in high-dimensional data classification through its superior accuracy and precision, paving the way for more efficient and accurate classification models.

Introduction

In the evolving landscape of machine learning, Support Vector Machines (SVMs) have emerged as a robust and versatile tool, especially for classification tasks. However, using high-dimensional datasets, common in fields like genomics and image processing, poses unique challenges. Traditional SVMs, while effective in many scenarios, often struggle with the curse of dimensionality, leading to overfitting and computational inefficiency.

This project aims to address the challenge of effectively classifying high-dimensional data using SVMs. The significance of enhancing SVMs for such datasets lies in their widespread applicability, from medical diagnosis to complex pattern recognition in large-scale image datasets. Improved SVM performance in high-dimensional spaces has the potential to unlock deeper insights and more accurate predictions in these critical areas.

To tackle this challenge, we have employed an adaptive approach to SVMs. The cornerstone of this method is a dynamic feature selection mechanism, integrated with adaptive feature weighting. This innovative approach aims to selectively focus on the most informative features of the data, thereby reducing dimensionality while retaining essential information. Additionally, regularization techniques are incorporated to prevent overfitting, a common pitfall when dealing with high-dimensional data.

The datasets used for this project consist of musk.csv, diabetes.csv, and ionosphere.csv. For all three datasets, the input to our algorithm is a high-dimensional feature vector, molecular features for musk.csv, patient features for diabetes.csv, and radar signal features for

ionosphere.csv, and our enhanced SVM method outputs a predicted class label, musk/non-musk for musk.csv, positive/negative for diabetes.csv, and good/bad signals for ionosphere.csv.

Related Works

The challenge of class imbalance in machine learning, particularly in high-dimensional data scenarios, is a significant area of concern, especially for traditional classifiers like Support Vector Machines (SVMs). These classifiers often underperform due to a bias toward the majority class, leading to suboptimal performance in critical applications such as medical diagnosis (Gurbuz et al., 2011)

Cost-Sensitive SVM (CS-SVM) has emerged as a potential solution to this problem. It integrates misclassification costs into the training process, effectively weighing classes based on their importance. This approach is particularly beneficial in scenarios where the cost of misclassifying minority class instances is high. However, determining the appropriate cost parameters can be challenging and may require domain expertise (Tan et al., 2014).

In the realm of feature selection, techniques such as the Fisher Score and Recursive Feature Elimination SVM (RFE-SVM) are gaining prominence. The Fisher Score is effective even in scenarios with skewed class distributions. In contrast, RFE-SVM focuses on eliminating features in a backward stepwise manner, zeroing in on those that contribute most to class separation. These methods have proven effective in reducing dimensionality and improving model performance, especially in high-dimensional data. However, they can be computationally intensive, and in the case of RFE-SVM, there is a risk of excluding features that are individually weak but collectively strong predictors (Chongomweru et al., 2021).

The development of Kernel-Penalized SVM represents a novel approach to addressing class imbalance in high-dimensional domains. This method optimizes scaling factors in the SVM while penalizing their cardinality, providing a balance between model complexity and classification performance. It extends to both SVDD and CS-SVM, employing a quasi-Newton optimization scheme for efficient convergence. The complexity of this approach, however, might lead to longer training times, and it requires careful tuning of hyperparameters for optimal performance (Chung-Jui et al., 2007).

These approaches have been validated through practical applications, particularly in medical fields like cancer diagnosis. They demonstrate the potential of enhancing SVM classification accuracy in imbalanced, high-dimensional datasets. While these methods show real-world applicability and effectiveness in critical domains, their success can be highly data-dependent, and they may not generalize well across different types of datasets or problems (Balakrishnan et al., 2008)

Dataset and Features

We have employed three diverse datasets to explore the capabilities of SVMs in different contexts. These datasets are from two research papers.

4.1 Diabetes Dataset

The first dataset is Diabetes, which is a classification of diabetes in Pima Indians, where researchers conducted SVM analysis using a Ranker Search method. This dataset provides valuable insights into the application of SVMs in predicting diabetes outcomes among Pima Indians, showcasing the algorithm's potential in medical research and healthcare. We are going to run our Adaptive Feature Selection (AFS) using Gradient Boosting with the diabetes dataset to compare the results of the feature selection rather than the type of algorithm.

The Ranker-Search feature selection process consists of two phases. Initially, Support Vector Machine (SVM) attribute evaluation assigns weights to features, which are then ranked. In the second phase, a backward search removes the least-ranked features, one at a time, assessing their impact on classifier accuracy. The optimal feature subset is determined based on enhancements in accuracy. The model runs a Naive Bayes classification on the dataset, assesses accuracy, and then removes the least-ranked feature per the Ranker-Search algorithm until a consistent accuracy is maintained.

4.2 Musk and Ionosphere Datasets

The other two datasets were obtained from Tan et al. The first of the two additional datasets is the Musk dataset, a binary classification challenge, which aims to determine whether a molecule is a musk or not. The Ionosphere dataset, another binary classification dataset, involves identifying whether a given electron possesses an ionospheric structure. For these datasets, the authors deployed various SVM variants, including SVM-RFE (Recursive Feature Elimination), 0-norm SVM, 1-norm SVM, traditional p-norm SVM2, and an adaptive p-norm SVM that they developed. This comprehensive approach allows us to evaluate the effectiveness of SVMs across different datasets and problem domains, offering a robust analysis of SVMs in diverse real-world scenarios.

These scientists used actual SVM models rather than strict feature selection. This will allow us to get a more holistic view of how our AFS using Gradient Boosting will work by allowing us to compare it to variations on the traditional kernelized SVM.

4.3 Data Preparation and Preprocessing

Data is sourced from a CSV file and comprises various features with the last column as the class label. The dataset is divided into training and test sets, with stratification ensuring

consistent class distribution. Feature scaling is performed using StandardScaler, normalizing features to have zero mean and unit variance.

The major data cleaning that occurred was removing empty values from the musk dataset using WEKA.

The remainder of the preprocessing was through the feature selection algorithms themselves in order to minimize the placebo effect of scrutinized data preparation. This means that after using StandardScaler to normalize the data, we no longer had to do any more preprocessing.

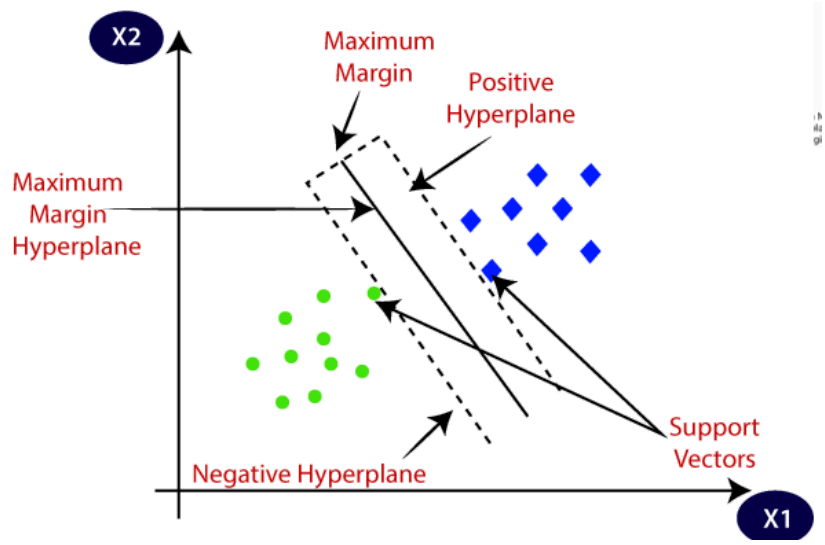
Methods

5.1 Support Vector Machine (SVM)

The Support Vector Machine (SVM) algorithm is employed for classification tasks in our project. The SVM operates by identifying an optimal hyperplane in an N-dimensional space (where N is the number of features) that distinctly classifies data points into different categories. The optimization objective of SVM is to minimize the following function:

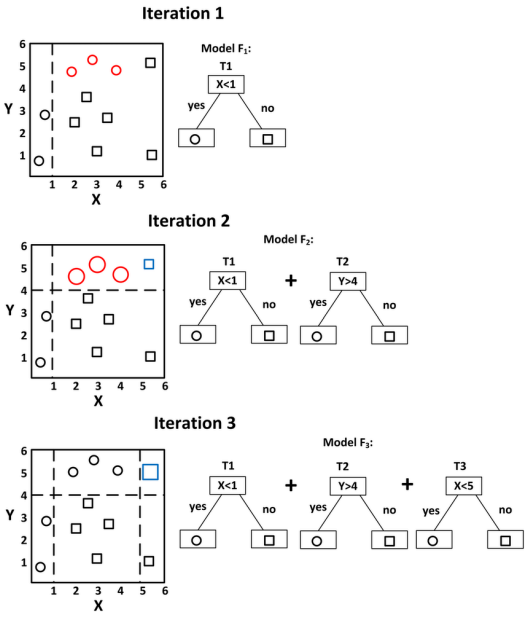
$$min_{w,b,\{\beta_n\}} \frac{1}{2}||w||_2 + C\sum_n \beta_n$$
$$s.t.\ \ y_n\left[w^T\phi(x_n) + b\right] \geq 1 - \beta_n\,;\ \forall n$$
$$\beta_n \geq 0\,,\ \forall n$$

On a two-dimensional plane where data points are plotted on feature-based axes, the SVM seeks an optimal hyperplane that categorically segregates data classes while maintaining a maximum margin. The closest points to the margin, known as 'Support Vectors', are instrumental in calibrating the hyperplane.



5.2 Adaptive Feature Selection using Gradient Boosting

Gradient Boosting, an ensemble learning technique, is at the heart of our adaptive feature selection process.



- Gradient Boosting: The algorithm begins with a base model (usually a simple decision tree) and sequentially adds trees that predict the residuals or errors of the previous trees. Each new tree is fitted on the residual errors of the combined existing ensemble of trees. The learning rate parameter controls the contribution of each tree to the final model. Lower rates typically require more trees but can lead to a more robust model.
- Training: The Gradient Boosting Classifier, set with 100 decision trees, is trained on the training dataset. This sequential correction strategy aims to continually improve the model's accuracy with each iteration.
- Feature Importance: After training, the Gradient Boosting model evaluates the importance of each feature, assigning scores based on each feature's contribution to reducing prediction error across all trees.

- Dynamic Selection: Features are ranked by importance, and a subset is dynamically selected to meet a cumulative importance threshold. This method focuses on retaining the most predictive features, reducing dimensionality, and potentially enhancing SVM performance.

5.3 SVM Training with Parameter Tuning

An SVM with a linear kernel is employed, utilizing GridSearchCV for tuning the regularization parameter 'C'. This approach seeks the optimal balance between model complexity and generalization capacity. The best-performing model is used for evaluation.

5.4 Cross-Validation and Model Evaluation

Model performance is evaluated using cross-validation on the test set, providing a varied metric compared to a single train-test split. We report on cross-validation scores, accuracy, precision, recall, and the F1-score, offering a comprehensive view of the model's predictive ability. We then compare the performance of our adaptive SVM to a linear kernelized SVM and other advanced SVM like p-norm SVM and SVM-RFE that were run on the respective datasets (musk.csv, and ionosphere.csv).

Additionally, in order to evaluate the efficiency of our AFS method, we paired it with a Naive Bayes classifier on the diabetes dataset used in Balakrishnan et al., as they similarly evaluated their feature selection method.

Experiments and Results

6.1 Finding Threshold Values

To test the effectiveness of our models, we first had to find the right threshold for the feature selection algorithm. We ran 100 tests from 0 to 1 (incrementing by 0.01) each time, ran the new dataset through Naive Bayes, and chose the threshold that yielded the highest accuracy. For the Naive Bayes classification, our best threshold was 0.71.

We did the same thing for the SVM on both the Ionosphere dataset and the Musk dataset. Once our test was completed, we found our best threshold for Ionosphere to be 0.97 and for Musk to be 0.99.

6.2 Comparisons

To create quantifiable success for our model, we wanted to compare it to a few other simulations. First, we wanted to compare our results to each of the basic models without any

feature selection (Naive Bayes on the Diabetes dataset, SVM on the Ionosphere dataset, SVM on the Musk dataset).

As mentioned in 4.1 and 4.2, we then compared our results to those found by Balakrishnan et al. for Naive Bayes and Tan et al. for SVM.

6.3 Performance Metrics

To get a holistic view of our model's successes, we decided to look at the accuracies, confusion matrices, and receiver operating characteristic (ROC) curves of each of our models on the three datasets, both with and without AFS using Gradient Boosting feature selection (6 total tests). From our confusion matrices, we calculated precision and sensitivity (recall), as well as obtained the ROC Area under Curve (AUC). We then obtained the same metrics from our related research, although some variables were unable to be calculated or found.

Each of these metrics plays an important role in the success of our models. Accuracy is widely considered to be the best indicator of success for the initial, overarching performance of a model. We decided to find both precision and sensitivity as well, given that the necessity of one of these metrics over the other would depend on the data needed to be collected. Given that True Positives = TP, True Negatives = TN, False Positives = FP, and False Negatives = FN, the formulas for these metrics are:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

$$Precision = \frac{TP}{TP+FN}$$

$$Sensitivity = \frac{TP}{TP+FP}$$

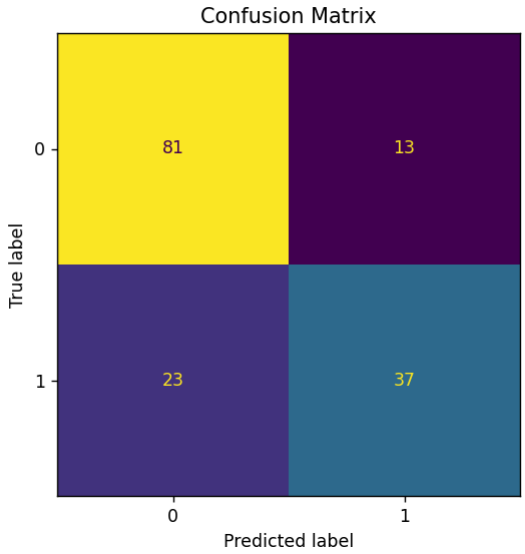
Our analysis found that sensitivity would be more important than precision for all three datasets because a False Negative has a more drastic effect than a False Positive. For Diabetes, predicting that a diabetic patient does not have diabetes could cause extremely dangerous health effects. For Musk, incorrectly categorizing a musk particle could mean that a filter lets said particle flow through into a body of water, potentially endangering wildlife. For Ionosphere, an underestimation of the number of ionic structures existing in the ionosphere would throw off existing models and could result in needed environmental awareness being overlooked.

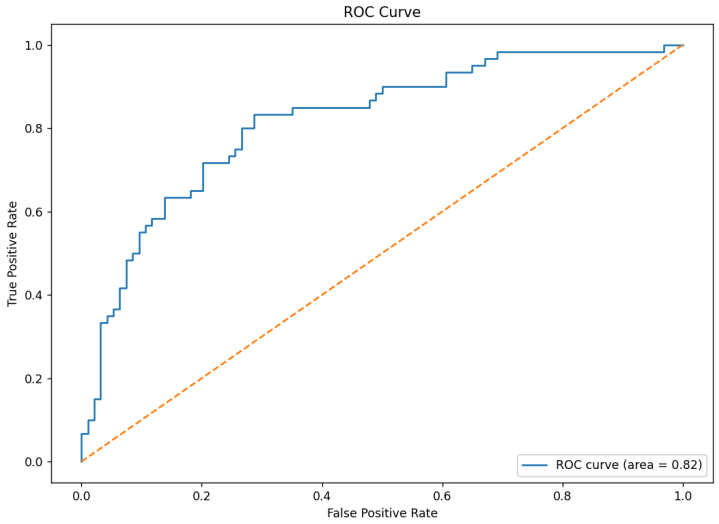
The ROC AUC also allows us to understand how random our model is, as well as how skewed our models are by class imbalances.

6.4 Results - Naive Bayes

Naive Bayes without feature selection:

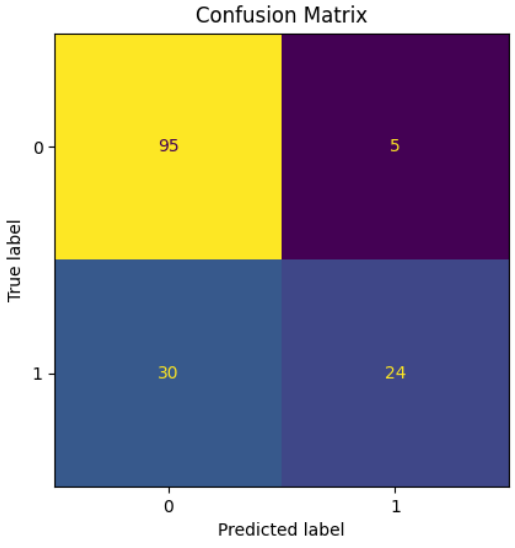
Accuracy: 0.7662337662337663

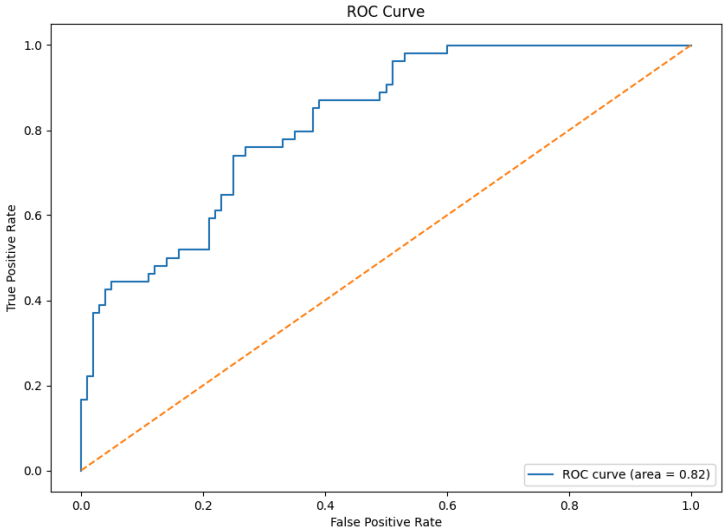




Naive Bayes with AFS using Gradient Boosting:

Best Accuracy: 0.7817028027498678
Best Threshold: 0.71

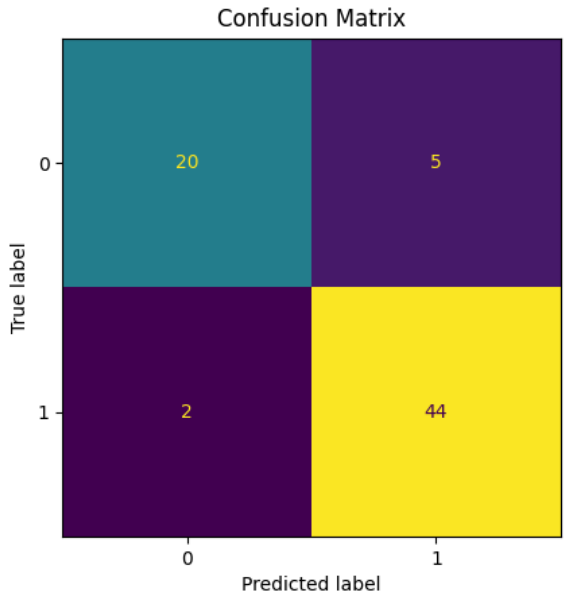


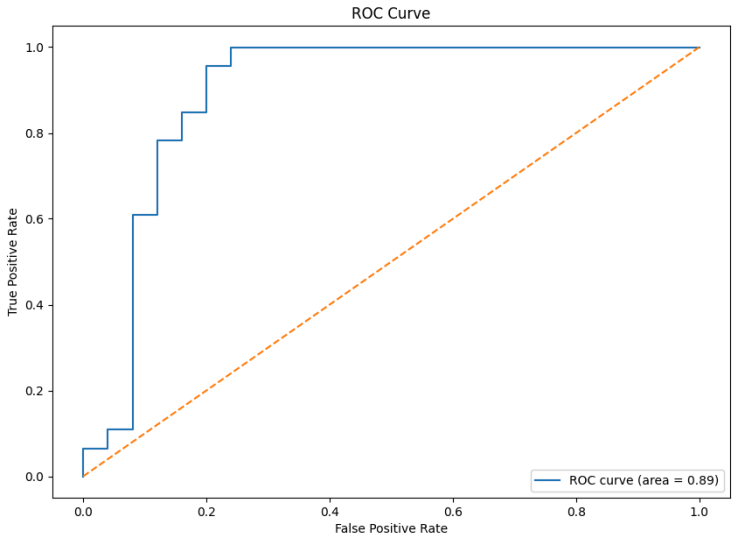


6.5 Results - SVM

SVM without feature selection on Ionosphere dataset:

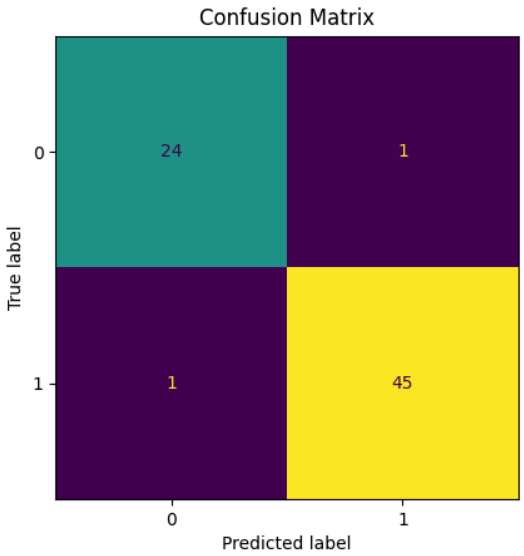
Mean 10-Cross Validation Score: 0.8589285714285715

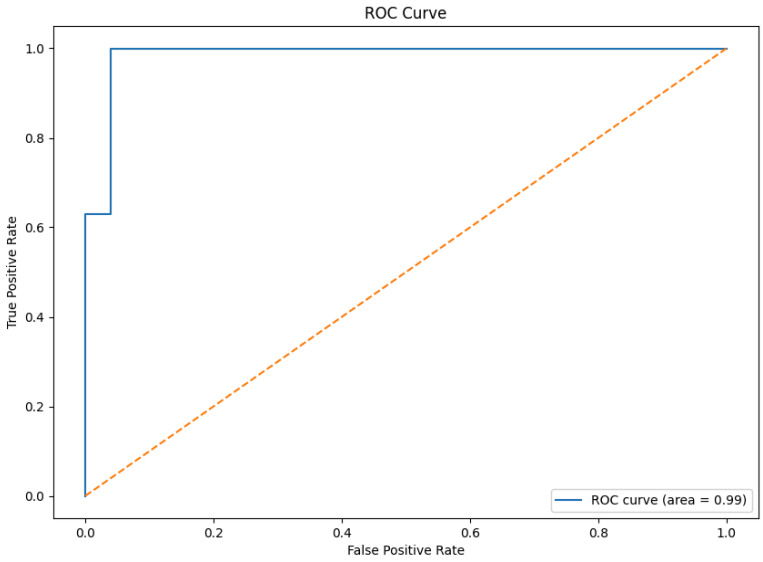




SVM with AFS using Gradient Boosting on Ionosphere dataset:

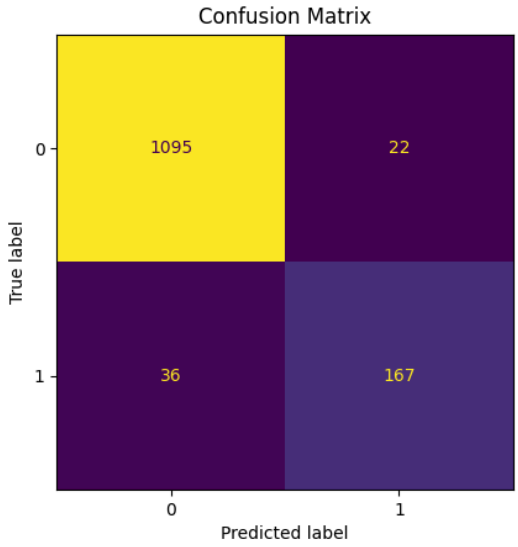
Best Accuracy: 0.971830985915493
Best Threshold: 0.97

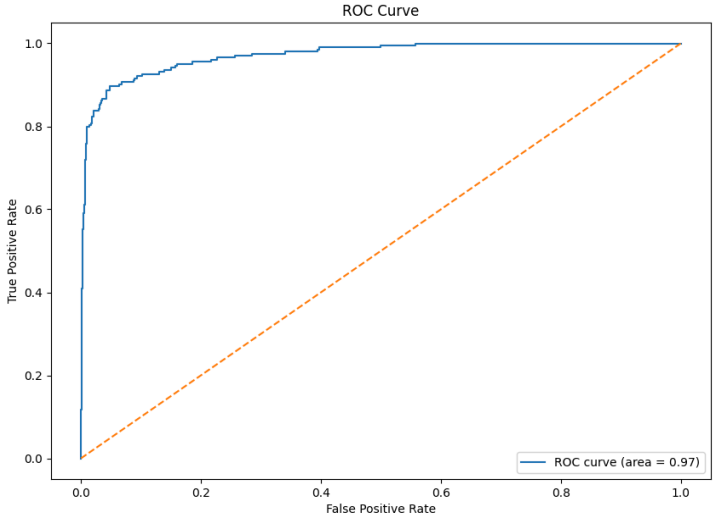




SVM without feature selection on Musk dataset:

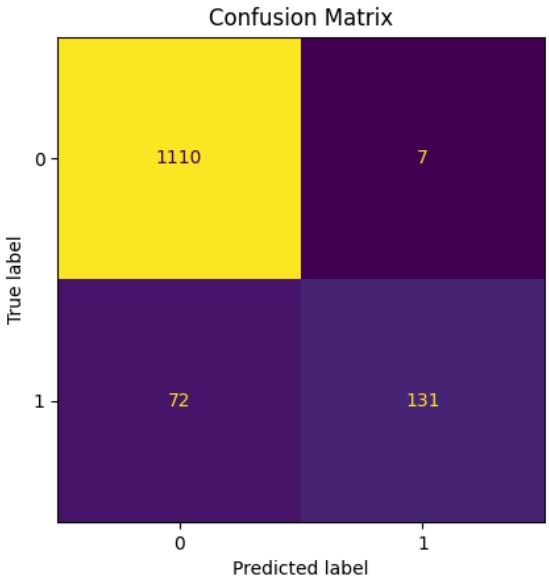
Mean 10-Cross Validation Score: 0.9045454545454545

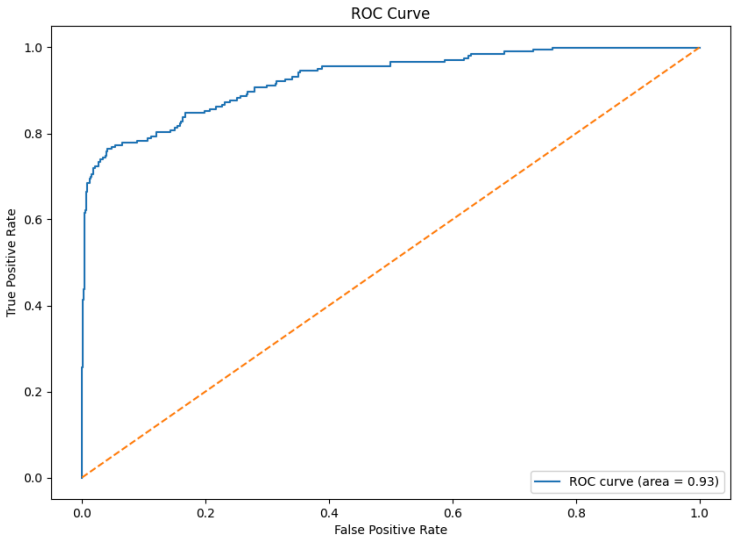




SVM with AFS using Gradient Boosting on Musk dataset:

Best Accuracy: 0.9401515151515152
Best Threshold: 0.99





6.6 Compiled Results

Diabetes (Naive Bayes)	**Sensitivity**	**Precision**	**Accuracy**	**AUC of ROC**
No feature selection	0.62	0.74	0.766	0.82
Ranking w/ Backwards Search			0.777	0.828
AFS using Gradient Boosting	0.44	0.83	0.782	0.82
Ionosphere (SVM)	**Sensitivity**	**Precision**	**Accuracy**	**AUC of ROC**
No feature selection	0.96	0.90	0.859	0.89
p-norm SVM	0.874	0.978	0.894	
p-norm SVM2	0.973	0.885	0.894	
0-norm SVM	0.874	0.951	0.881	
1-norm SVM	0.874	0.978	0.894	
SVM-RFE	0.859	0.951	0.869	
AFS using Gradient Boosting	0.98	0.98	0.972	0.99
Musk (SVM)	**Sensitivity**	**Precision**	**Accuracy**	**AUC of ROC**
No feature selection	0.82	0.88	0.905	0.97
p-norm SVM	0.819	0.874	0.863	
p-norm SVM2	0.797	0.826	0.832	
0-norm SVM	0.791	0.826	0.831	
1-norm SVM	0.778	0.744	0.796	
SVM-RFE	0.785	0.811	0.821	
AFS using Gradient Boosting	0.65	0.95	0.940	0.93

6.7 Discussion

As shown in our table of results, our models performed extremely well for the SVM model on both datasets, yet only showed slight improvements on the Naive Bayes model. This is likely because of the complexity of AFS using Gradient Boosting, since this feature selection

algorithm is very good at preventing overfitting and working through noise and hence is more successful with models that can make more flexible predictions like SVM due to the generalization it therefore provides. Due to its ensemble nature, the feature selection algorithm finds and chooses features based on dependencies between attributes, a method that is relatively unsuccessful within Bayesian networks due to the Naive Bayes' assumption of attribute independence.

We also found that our models with AFS using Gradient Boosting had lower sensitivity values for the Diabetes and Musk datasets, which we expected due to the larger class imbalance in both of those datasets as compared to the Ionosphere dataset. Gradient boosting algorithms tend to be highly affected by class imbalance due to the formation of trees, which often do not capture patterns in minority classes and tend to prioritize minimizing errors in the majority class.

Overall, our feature selection saw its most success on the Ionosphere dataset, with the highest performance in all four of our calculated metrics as compared to the non-preprocessed model and the findings of Tan et al. As mentioned earlier, SVMs are more successful with AFS using Gradient Boosting than Naive Bayes models, and the high dimensionality coupled with relatively low instances of the Ionosphere dataset provided a perfect environment for our feature selection algorithm to be successful. We faced some, albeit few, unsuccessful performances on the Musk dataset due to its very high instance count at a medium dimensionality.

Conclusion/Future Work

In our report, we successfully utilized Support Vector Machines (SVM) with Attribute-Frequency Selection (AFS) using Gradient Boosting, aimed at enhancing the performance of linear-kernel SVMs. Our extensive evaluation showcased the superiority of this approach over other models, including p-norm SVMs, SVM-RFEs, and alternative enhancements. Our model excelled on datasets characterized by high dimensionality and low instances.

One pivotal part of our study was the advantage of our feature selection technique when applied in conjunction with SVMs as compared to its low effectiveness on Naive Bayes. The AFS algorithm demonstrated its success by navigating through data noise, allowing for the identification of connections between attributes. This feature selection mechanism proved to be a key factor contributing to the overall success of our model. Furthermore, our AFS with Gradient Boosting on an SVM highlighted the model’s exceptional performance on datasets with high dimensionality and limited instances

Looking ahead, there are several avenues for future exploration. Firstly, due to time constraints, cross-validation was not conducted on our AFS-SVM model. Implementing cross-validation would serve to enhance the reliability of our findings by providing a more thorough assessment of the model’s generalized results. Additionally, we plan to extend our research by comparing the AFS-SVM model with various gradient boosting algorithms, such as XGBoost or AdaBoost, to figure out if AFS remains the optimal choice for boosting SVM

performance. As we continue to explore SVMs and AFS with our project, we hope to leave our mark on the field of data preprocessing and continue to explore the connection between academic skills and practical applications.

Contributions

Throughout the research process, work was divided evenly depending on both project member's strong suits. Nikhil designed and programmed the adaptive feature method and the regularization parameterization. Rishabh then applied these methods to a linear kernalized SVM and a Naive Bayes classifier. The paper was also written equitably as both partners picked up sections they were willing to write, and then the other would proofread and add information they felt was missing.

References

Balakrishnan, S., Narayanaswamy, R., Savarimuthu, N., & Samikannu, R. (2008). SVM ranking with backward search for feature selection in type II diabetes databases. 2008 IEEE International Conference on Systems, Man and Cybernetics, Singapore, 2628-2633. doi: 10.1109/ICSMC.2008.4811692.

Chongomweru, H., & Kasem, A. (2021). A novel ensemble method for classification in imbalanced datasets using split balancing technique based on instance hardness (sBal_IH). Neural Comput & Applic 33, 11233–11254. https://doi.org/10.1007/s00521-020-05570-7

Chung-Jui, Tu., Chuang, Li-Yeh., Jun-Yang, Chang., & Yang, Cheng-Hong. (2007). Feature selection using PSO-SVM. IAENG International Journal of Computer Science. 33.

Grandvalet, Yves & Canu, Stéphane. (2002). Adaptive scaling for feature selection in SVMs. Proceedings of the 15th International Conference on Neural Information Processing Systems (NIPS'02). MIT Press, Cambridge, MA, USA, 569–576.

Gurbuz, Emre & Kilic, Erdal. (2011). Diagnosis of diabetes by using Adaptive SVM and feature selection. 2011 IEEE 19th Signal Processing and Communications Applications Conference, SIU 2011. 10.1109/SIU.2011.5929582.

Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. Nature 585, 357–362 (2020). DOI: 10.1038/s41586-020-2649-2.

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90-95.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. Journal of machine learning research, 12(Oct), 2825-2830.

Radhika. (2023). Primal Form of SVM (Non-Perfect Separation). Analytics Vidhya. https://www.analyticsvidhya.com/blog/2020/10/the-mathematics-behind-svm/. Accessed 6 Jan. 2024.

Saini, Anshul. (2024). Support Vector Machine Diagram. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/. Accessed 6 Jan. 2024.

Tan, J., Zhang, Z., Zhen, L., et al. (2013). Adaptive feature selection via a new version of support vector machine. Neural Comput & Applic 23, 937–945. https://doi.org/10.1007/s00521-012-1018-y

"Exploring the clinical features of narcolepsy type 1 versus narcolepsy type 2 from European Narcolepsy Network database with machine learning." ResearchGate. https://www.researchgate.net/figure/A-simple-example-of-visualizing-gradient-boosting_fig5_326379229 [accessed 7 Jan, 2024]