Support Vector Machine

Overview of an ML Algorithm

Chavez Lope, Janet

***Abstract*—Support Vector Machine (SVM) is a popular machine learning algorithm widely used for classification and regression tasks. In this paper, we provide a comprehensive review of the Support Vector Machine algorithm, covering its theoretical foundations, key concepts, and practical implementation. We explore the history of SVM, its mathematical formulation, the kernel trick, training process, and its applications in various domains. Additionally, we discuss the advantages and limitations of SVM and compare it with other popular machine learning algorithms. Through this review, we aim to provide a deep understanding of SVM, its capabilities and potential applications.**

***Index Terms*—keywords, svm, support vector machine, machine learning algorithms**

I. Introduction

MACHINE learning algorithms are computational models designed to automatically learn patterns and relationships from data without being explicitly programmed. These algorithms enable computers to make predictions or take actions based on input data and past experiences. Machine learning algorithms can be broadly categorized into three types, supervised learning, unsupervised learning and reinforcement learning.

Supervised learning algorithms learn from labeled training data, where each data instance is associated with a corresponding target or output label. The aim is to build a model that can generalize patterns in the training data to make accurate predictions or classifications on unseen data. A few examples of supervised learning algorithms include decision trees, random forests, support vector machines, and neural networks.

Unsupervised learning algorithms work with unlabeled data, where the input data has no corresponding output labels. These algorithms aim to discover patterns, structures, or relationships in the data without any predefined target. Its tasks include clustering (grouping similar data points), dimensionality reduction, and anomaly detection. Some examples of unsupervised learning algorithms include k-means clustering, hierarchical clustering, and principal component analysis (PCA).

Reinforcement learning algorithms learn through an agent interacting with an environment and receiving feedback in the form of rewards or punishments. The agent learns to take actions to maximize cumulative rewards over time, based on trial and error. Reinforcement learning is commonly used in tasks such as game playing, robotics, and optimization. Examples of reinforcement learning algorithms include Q-learning and deep reinforcement learning.

SUPPORT VECTOR MACHINE (SVM) is a popular supervised learning algorithm used for classification and regression tasks. It aims to find an optimal hyperplane that separates different classes of data points with the largest possible margin. The key idea is to transform the data into a higher-dimensional feature space, where a hyperplane can be constructed to maximize the separation between classes. SVM can handle linear and non-linear classification tasks. Non-linear classification is achieved by using a kernel function to implicitly map the data into a higher-dimensional space, where linear separation is possible. SVM has advantages such as handling high-dimensional data, robustness against over-fitting, and effectiveness in both linear and non-linear classification tasks. The training process involves solving a convex optimization problem to find the optimal hyperplane. SVM has been successfully applied in various domains such as text classification, image recognition, bio-informatics, and finance.

A. HISTORICAL DEVELOPMENT

Early development

The origins of Support Vector Machines can be traced back to the 1960s and 1970s when researchers were exploring pattern recognition and the idea of finding optimal decision boundaries. In particular, the work on the Perceptron Algorithm by Frank Rosenblatt in 1957 that laid the foundation for the concept of linear classifiers.

Vapnik and Cortes

Majors contributions to the development of SVM came from Vladimir Vapnik and his collaborator Corinna Cortes in the 1990s. They introduced the concept of Support Vector Machines as an extension of the earlier work on the theory of learning and statistical pattern recognition. Vapnik, a mathematician and computer scientist, had been researching the theory of learning in the 1960s, which focused on understanding the limitations of traditional statistical learning methods. He introduced the concept of structural risk minimization (SRM), emphasizing the importance of balancing the trade-off between model complexity and generalization ability. Cortes with Vapnik further developed the concept of SVM and introduced the maximum margin classifier, which formed the basis for SVM's approach to finding optimal decision boundaries. They proposed the idea of maximizing the margin between classes, which allows for better generalization and improved performance.

Evolution and adoption

Researchers and practitioners have expanded upon the original SVM formulation, exploring various aspects such as handling non-linear data, incorporating kernel functions, and addressing multi-class classification problems. Bernhard Boser, Isabelle Guyon, and Vladimir Vapnik introduced the kernel trick that enabled SVM to efficiently handle non-linear classification tasks by implicitly mapping the data into a higher-dimensional feature space. This approach avoids explicitly computing the transformed feature space, making SVM computationally tractable. SVM gained popularity due to its solid theoretical foundations, versatility, and strong empirical performance over the years. It has been successfully applied in a wide range of domains, including text classification, image recognition, bio-informatics, and finance. SVM's ability to handle high-dimensional data, its robustness against over-fitting, and its effectiveness in both linear and non-linear classification tasks have contributed to its wide adoption. It has helped inthe advancements in optimization algorithms, scalability for large-scale datasets, and extensions to handle regression tasks (Support Vector Regression). Researchers continue to explore new techniques and variations of SVM, including multi-class SVM, ensemble methods, and deep support vector machines.

B. MATHEMATICAL FOUNDATIONS

Linear Separability and maximum margin principle

The SVM's goal is to find an optimal hyperplane that separates different classes of data points. Linear separability refers to the property where the data points of different classes can be perfectly separated by a hyperplane in the feature space. The maximum margin principle is the key concept in SVM, aiming to find the hyperplane that maximizes the margin between the classes. This margin is the distance between the hyperplane and the closest data points from each class. SVM seeks to find the hyperplane with the largest margin, as it provides better generalization and robustness to unseen data. The intuition behind the maximum margin principle is to create a decision boundary that is as far as possible from the training data, reducing the risk of miss-classification.

Formulation of SVM optimization problem

The SVM optimization problem involves finding the optimal hyperplane that maximizes the margin while minimizing the classification error. Mathematically, SVM formulates this as a constrained optimization problem. For linearly separable data, the optimization problem can be stated as follows:
minimize: $||\mathrm{w}||^2$
subject to: $yi(w * xi + b)1$ for all training samples (xi, yi)
Here, w represents the weight vector perpendicular to the hyperplane, b is the bias term, xi is a training sample, and yi is the corresponding class label $(+1 or -1)$. The inequality constraint ensures that the data points are correctly classified and lie outside the margin region.

Complexity and duality

Convexity is a crucial property in SVM optimization, which ensures that the problem has a unique global minimum. Convexity implies that any local minimum is also the global minimum. Duality in SVM refers to the transformation of the SVM optimization problem into its dual form, which allows for more efficient computation. The dual problem involves optimizing the Lagrange multipliers corresponding to the constraints of the primal problem. By solving the dual problem, we can obtain the optimal Lagrange multipliers, which in turn provide the support vectors and define the hyperplane. Duality also allows us to take advantage of the kernel trick, which implicitly maps the data into a higher-dimensional feature space. This mapping allows SVM to handle non-linearly separable data without explicitly computing the transformed feature space, making it computationally efficient. The dual problem in SVM is typically a quadratic programming problem that can be efficiently solved using optimization algorithms such as Sequential Minimal Optimization (SMO) or interior-point methods.

C. THE ALGORITHM

Feature Space and Decision Boundary

The feature space refers to the space where the input data is transformed or mapped to. The SVM algorithm aims to find an optimal decision boundary that separates different classes of data points in this feature space. For linearly separable data, the decision boundary is a hyperplane that maximizes the margin between the classes. The hyperplane is determined by a weight vector (w) perpendicular to it, and a bias term (b) determines its position. The decision boundary separates the feature space into different regions corresponding to different classes.

Soft-margin SVM and slack variables

Soft-Margin SVM allows for some miss-classifications by introducing slack variables (E) that measure the degree of miss-classification. The optimization objective is modified to minimize both the margin size and the classification errors. The introduction of slack variables relaxes the constraints in the optimization problem, allowing some data points to fall within the margin or even on the wrong side of the decision boundary. The trade-off between maximizing the margin and minimizing miss-classifications is controlled by a parameter (C), which determines the balance between model complexity and training error.

Non-linear SVM and the kernel trick

SVM can handle non-linearly separable data by using the kernel trick. The kernel trick implicitly maps the data into a higher-dimensional feature space, where linear separation may be possible. Rather than explicitly computing the transformed feature space, the kernel function calculates the inner product between pairs of data points in the original feature space. Common kernel functions include the linear kernel, polynomial kernel, radial basis function (RBF) kernel, and sigmoid kernel. The kernel trick allows SVM to efficiently perform computations in the high-dimensional feature space without explicitly dealing with the transformed data. This makes SVM computationally feasible even in cases where

the feature space dimension is very high or infinite.

Support Vector Regression (SVR)
In addition to classification, SVM can be extended to handle regression problems through Support Vector Regression (SVR). SVR aims to find a regression function that lies within a specified margin around the training data points, instead of trying to minimize classification errors. SVR uses a loss function that penalizes deviations from the desired margin, and the optimization problem is formulated to find a regression function that maximizes the margin while satisfying the specified error tolerance. The epsilon-insensitive loss function allows for a certain degree of error within the specified margin. SVR is particularly useful when dealing with non-linear regression tasks and can handle data with complex relationships.

D. TRAINING PROCESS

Data Preprocessing and feature scaling
Data preprocessing is an important step in SVM training. It involves cleaning and transforming the raw data to make it suitable for the SVM algorithm. Common preprocessing steps include handling missing values, encoding categorical variables, and removing outliers. Feature scaling is often necessary to ensure that all features contribute equally to the SVM model. Scaling techniques such as standardization (mean normalization) or normalization (min-max scaling) are commonly used to bring features to a similar scale. This step helps to prevent features with large values from dominating the learning process and ensures balanced contributions from all features.

Kernel functions selection
The choice of kernel function is a crucial aspect of SVM training. The kernel function determines how the data is implicitly mapped into a higher-dimensional feature space. Different kernel functions have different properties and can handle different types of data and decision boundaries. Common kernel functions include the linear kernel, polynomial kernel, radial basis function (RBF) kernel, and sigmoid kernel. The selection of the kernel function depends on the characteristics of the data and the complexity of the decision boundary. Experimentation and cross-validation techniques can help determine the most suitable kernel function for a given problem.

Parameter tuning and model selection
SVM models have hyperparameters that need to be tuned for optimal performance. Hyperparameters include the regularization parameter (C) that balances the margin size and training error, as well as kernel-specific parameters such as the degree of the polynomial kernel or the width of the RBF kernel. Parameter tuning is typically performed using techniques like grid search or randomized search, where different combinations of hyperparameters are evaluated using cross-validation. The goal is to find the set of hyperparameters that yield the best performance on the validation data. Model selection involves choosing the final SVM model with the optimal hyperparameters based on the performance evaluation.

Optimization algorithms for large-scale SVM
When dealing with large-scale datasets, traditional optimization algorithms may become computationally expensive. Several optimization algorithms have been developed to handle large-scale SVM training efficiently. Some popular approaches include:

- Sequential Minimal Optimization (SMO): This algorithm breaks down the optimization problem into a series of smaller sub-problems and solves them analytically, making it suitable for large datasets.
- Stochastic Gradient Descent (SGD): This optimization algorithm uses random subsets of the data to estimate the gradients, making it scalable for large datasets.
- Parallel SVM training: Distributed and parallel computing techniques can be employed to speed up the training process by distributing the computation across multiple machines or processors.

E. ADVANTAGES AND LIMITATIONS

Advantages

1) High Accuracy and Robustness: SVMs are known for their ability to achieve high accuracy in classification tasks, especially when dealing with complex and high-dimensional data. They can handle both linearly separable and non-linearly separable data, thanks to the use of kernel functions and the maximum margin principle. SVMs are also robust to noise and outliers in the data.
2) Handling Non-linear Data Using Kernel Trick: The kernel trick allows SVMs to handle non-linearly separable data by implicitly mapping it to a higher-dimensional feature space. This makes SVMs flexible in capturing complex decision boundaries and improves their ability to generalize well to unseen data.
3) Interpretability and Feature Selection: SVMs provide interpretability by identifying the support vectors, which are the data points closest to the decision boundary. These support vectors play a crucial role in defining the decision boundary and can provide insights into the important features or patterns in the data. Additionally, SVMs naturally perform feature selection by selecting the most relevant features that contribute to the decision boundary.

Limitations

1) Computational Complexity and Memory Requirements: SVMs can be computationally expensive and memory-intensive, especially when dealing with large datasets. The training time and memory requirements increase with the number of data points, making SVMs less suitable for very large-scale applications without efficient optimization algorithms or parallel computing capabilities.
2) Sensitivity to Parameter Settings and Overfitting: SVMs have hyperparameters that need to be carefully tuned for

optimal performance. The choice of parameters such as the regularization parameter (C) and the kernel parameters can significantly impact the SVM's performance. Poorly chosen parameter settings can lead to overfitting (model overly adapted to the training data) or underfitting (model fails to capture the underlying patterns in the data).

3) Lack of Probabilistic Outputs: SVMs do not directly provide probabilistic outputs for classification. Instead, they produce binary decisions based on the location of data points relative to the decision boundary. If probabilistic outputs are desired, additional techniques such as Platt scaling or using alternative classifiers like Support Vector Probability Machines (SVPs) can be employed.

F. APPLICATIONS

Text Classification and Sentiment Analysis

SVMs are widely used in text classification tasks, where the goal is to categorize text documents into predefined categories or classes. They have been successfully applied in sentiment analysis to determine the sentiment or opinion expressed in textual data, such as product reviews, social media posts, and customer feedback. SVMs can effectively learn discriminative features from text data and generalize well to classify new and unseen text documents.

Image Recognition and Computer Vision

SVMs have shown promising results in image recognition and computer vision tasks, such as object detection, image classification, and face recognition. SVMs can learn complex decision boundaries and handle high-dimensional feature spaces, making them suitable for handling large sets of image features. By extracting relevant features from images and training SVM models, accurate classification and recognition of objects, scenes, or patterns can be achieved.

Bioinformatics and Genomics

SVMs have been widely used in bioinformatics and genomics for tasks such as gene expression analysis, protein structure prediction, and DNA sequence classification. SVMs can handle high-dimensional biological data and effectively classify and predict various biological properties. They have been applied to identify disease biomarkers, classify gene expression patterns, and predict protein-protein interactions, among other applications in the field of life sciences.

Financial Forecasting and Stock Market Prediction

SVMs have been employed in financial forecasting and stock market prediction tasks. By using historical financial data and market indicators as input features, SVM models can learn patterns and trends in the data to predict stock prices, market movements, or financial indicators. SVMs have been used to build trading models, portfolio optimization systems, and risk management tools in the finance industry.

G. COMPARISON: OTHER ML ALGORITHMS

Decision Trees and Random Forests

Decision trees are tree-like models that make predictions by splitting the data based on feature conditions. They are intuitive, interpretable, and can handle both numerical and categorical data. However, decision trees can be prone to overfitting and may not generalize well to unseen data. Random forests are an ensemble method that combines multiple decision trees. They reduce overfitting by averaging predictions from multiple trees. Random forests can handle high-dimensional data and provide feature importance measures. However, they may be computationally expensive and less interpretable compared to SVMs.

Logistic Regression and Neural Networks

Logistic regression is a linear model used for binary classification. It models the relationship between input features and the log-odds of the target variable. Logistic regression is computationally efficient, interpretable, and works well with linearly separable data. However, it may struggle with complex non-linear relationships. Neural networks are powerful models that consist of interconnected layers of artificial neurons. They can learn complex non-linear relationships and are highly flexible. Neural networks can handle large amounts of data and are effective for tasks such as image and speech recognition. However, they require a large number of parameters, extensive training, and can be prone to overfitting.

K-Nearest Neighbors (KNN) and Naive Bayes

K-Nearest Neighbors is a non-parametric algorithm that classifies data based on the majority vote of its neighboring data points. KNN is simple, easy to implement, and works well with small datasets. However, its performance can be sensitive to the choice of K and it may struggle with high-dimensional data. Naive Bayes is a probabilistic algorithm based on Bayes' theorem. It assumes that features are conditionally independent given the class label. Naive Bayes is computationally efficient, handles high-dimensional data well, and works well with text classification tasks. However, it may make the strong assumption of feature independence, which can limit its performance.

H. ADVANCES AND THE FUTURE

Kernel Approximation Techniques

Kernel methods, including SVM, often rely on the computation of kernel functions that can be computationally expensive, especially for large datasets. Recent advances have focused on developing efficient kernel approximation techniques to speed up the training and inference processes of SVM. These techniques aim to approximate the kernel matrix by using low-rank or sparse representations, enabling faster computations while maintaining good accuracy.

Multi-Class SVM and Ensemble Methods

Originally designed for binary classification, SVM has been extended to handle multi-class classification problems. Various approaches have been proposed, including one-vs-one

and one-vs-rest strategies. Additionally, ensemble methods that combine multiple SVM models have gained attention. These methods can enhance the performance and robustness of SVM by leveraging diverse models or incorporating ensemble techniques such as bagging or boosting.

Deep Support Vector Machines
Deep learning has revolutionized various domains, but traditional SVMs and deep learning techniques have different characteristics. Recent research has focused on bridging the gap between these two approaches by incorporating SVM principles into deep learning architectures. Deep Support Vector Machines (DSVMs) aim to combine the representation learning capabilities of deep neural networks with the robustness and interpretability of SVMs.

Interpretable and Explainable SVM Models
Interpretability and explainability are crucial in many real-world applications. While SVMs have a reputation for being interpretable due to the use of support vectors, recent efforts have focused on enhancing the interpretability of SVM models. This includes developing post-hoc interpretability methods that highlight the importance of features or support vectors in decision-making, as well as designing more interpretable kernel functions.

II. CONCLUSION

Support Vector Machines (SVM) have emerged as a powerful machine learning algorithm with several key findings and implications. Throughout this paper, we explored the historical background, mathematical foundations, training process, advantages and limitations, applications, and recent advances of SVM.

SVM utilizes the maximum margin principle and kernel functions to find optimal decision boundaries, enabling effective classification and regression tasks. It can handle both linearly separable and non-linearly separable data by employing the kernel trick to transform data into higher-dimensional feature spaces. It has shown high accuracy, robustness, and generalization capabilities across various domains, including text classification, image recognition, bioinformatics, and financial forecasting.

In conclusion, SVM has proven to be a versatile and effective machine learning algorithm. Its high accuracy, robustness, and interpretability make it a valuable tool in various domains. By addressing challenges related to scalability, interpretability, and handling complex data, future research in SVM has the potential to unlock even greater capabilities and expand its applications across a wide range of domains.

REFERENCES

[1] C. M. Bishop, "Pattern Recognition and Machine Learning," Springer, 2006.

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

[3] V. Vapnik, "Statistical Learning Theory," Wiley, 1998.

[4] N. Cristianini and J. Shawe-Taylor, "An Introduction to Support Vector Machines and Other Kernel-based Learning Methods," Cambridge University Press, 2000.

[5] C. J. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," Data Mining and Knowledge Discovery, vol. 2, no. 2, pp. 121-167, 1998.

[6] S. Boyd and L. Vandenberghe, "Convex Optimization," Cambridge University Press, 2004.

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

[8] T. M. Mitchell, "Machine Learning," McGraw-Hill, 1997.

[9] V. N. Vapnik and A. Lerner, "Pattern recognition using generalized portrait method," Automation and Remote Control, vol. 24, no. 6, pp. 774-780, 1963.

[10] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers," in Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pp. 144-152, ACM, 1992.

[11] V. N. Vapnik, "The Nature of Statistical Learning Theory," Springer Science Business Media, 1995.

[12] J. C. Platt, "Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines," Microsoft Research, 1998.

[13] B. Schölkopf and A. J. Smola, "Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond," MIT Press, 2002.

[14] B. Schölkopf, A. J. Smola, R. C. Williamson, and P. L. Bartlett, "New Support Vector Algorithms," Neural Computation, vol. 12, no. 5, pp. 1207-1245, 2000.

[15] L. Bottou and C. J. Lin, "Support Vector Machine Solvers," Large Scale Kernel Machines, pp. 301-320, 2007.

[16] Scikit-learn Documentation: Machine Learning in Python. [Online]. Available: https://scikit-learn.org/stable/

[17] "Support Vector Machines: A Visual Explanation with Example in Python." [Online]. Available: https://towardsdatascience.com/support-vector-machines-a-visual-explanation-with-example-in-python-3c7d3d34f536

[18] V. Cherkassky and Y. Ma, "Practical selection of SVM parameters and noise estimation for SVM regression," Neural Networks, vol. 17, no. 1, pp. 113-126, 2004.

[19] V. Vapnik, "The Nature of Statistical Learning Theory," Springer Science Business Media, 2013.

[20] C. J. Burges, "Tutorial on Support Vector Machines for Pattern Recognition," Data Mining and Knowledge Discovery, vol. 2, no. 2, pp. 121-167, 1998.

[21] T. Hastie, R. Tibshirani, and J. Friedman, "The Elements of Statistical Learning: Data Mining, Inference, and Prediction," Springer, 2009.

[22] B. Schölkopf, "The Kernel Trick for Dummies," Max Planck Institute for Biological Cybernetics Technical Report, 2001.

[23] K. R. Müller, A. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik, "An Introduction to Kernel-

Based Learning Algorithms," IEEE Transactions on Neural Networks, vol. 12, no. 2, pp. 181-201, 2001.

[24] R. E. Fan, P. H. Chen, and C. J. Lin, "Working Set Selection Using Second Order Information for Training Support Vector Machines," Journal of Machine Learning Research, vol. 6, Dec, pp. 1889-1918, 2005.

[25] L. Bottou and C. J. Lin, "Support Vector Machine Solvers," Large Scale Kernel Machines, pp. 301-320, 2007.

[26] B. Pang and L. Lee, "Opinion mining and sentiment analysis," Foundations and Trends in Information Retrieval, vol. 2, no. 1-2, pp. 1-135, 2008.

[27] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in European Conference on Machine Learning, 1998, pp. 137-142.

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

[29] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 1, 2005, pp. 886-893.

[30] Lin, X. Hu, and L. Zhang, "Kernel-based approaches for protein classification," Journal of Computational Biology, vol. 11, no. 5, pp. 563-574, 2004.

[31] A. Statnikov, L. Wang, and C. F. Aliferis, "A comprehensive comparison of random forests and support vector machines for microarray-based cancer classification," BMC Bioinformatics, vol. 9, 2008, p. 319.

[32] N. E. Huang et al., "The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis," Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences, vol. 454, no. 1971, pp. 903-995, 1998.

[33] Y. Li and L. Yu, "Ensemble-based stock price movement prediction using hybrid indicators," Expert Systems with Applications, vol. 39, no. 16, pp. 12755-12766, 2012.

[34] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5-32, 2001.

[35] J. R. Quinlan, "Induction of decision trees," Machine Learning, vol. 1, no. 1, pp. 81-106, 1986.

[36] C. M. Bishop, "Pattern Recognition and Machine Learning," Springer, 2006.

[37] T. Hastie, R. Tibshirani, and J. Friedman, "The Elements of Statistical Learning: Data Mining, Inference, and Prediction," Springer, 2009.

[38] T. Cover and P. Hart, "Nearest neighbor pattern classification," IEEE Transactions on Information Theory, vol. 13, no. 1, pp. 21-27, 1967.

[39] C. D. Manning, P. Raghavan, and H. Schütze, "Introduction to Information Retrieval," Cambridge University Press, 2008.

[40] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in Advances in Neural Information Processing Systems, 2008, pp. 1177-1184.

[41] F. R. Bach, "Sharp analysis of low-rank kernel matrix approximations," in Conference on Learning Theory, 2012, pp. 185-204.

[41] T. Hastie, S. Rosset, R. Tibshirani, and J. Zhu, "Multi-class AdaBoost," Statistics and its Interface, vol. 2, no. 3, pp. 349-360, 2009.

[43] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," Journal of Machine Learning Research, vol. 2, Nov, pp. 265-292, 2001.

[44] R. Collobert, S. Bengio, and Y. Bengio, "A parallel mixture of SVMs for very large scale problems," Neural Computation, vol. 14, no. 5, pp. 1105-1114, 2002.

[45] Y. Lee, B. Schölkopf, and R. C. Williamson, "A framework for learning predictive structures from multiple tasks and unlabeled data," Journal of Machine Learning Research, vol. 6, Oct, pp. 1817-1853, 2006.

[46] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why should I trust you?" Explaining the predictions of any classifier," in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 1135-1144.

[47] F. Wang, C. Rudin, D. Wagner, and R. Sevieri, "A kernelized support vector machine for structured data," Journal of Machine Learning Research, vol. 21, no. 29, pp. 1-54, 2020.