Mathematical Foundations of Support Vector Machine Method and Application Areas

Math 490 - Graduation Project

2025-2026 Fall Semester

19.01.2026

Author
Elif Berin YILMAZ

Supervisor
Assoc. Prof. Dr. Özlem DEFTERLİ

Department of Mathematics

Çankaya University

ABSTRACT

This study examines Support Vector Machines mainly from an optimization point of view. The learning problem is written as an optimization problem, and its primal and dual forms are discussed. The focus is on how the decision boundary is obtained through this process. Kernel functions are mentioned only to show how the same idea can be used for nonlinear problems. Some application areas are briefly noted at the end.

Keywords: Support Vector Machines, Optimization, Primal–dual formulation, Convex optimization, Kernel functions

Contents

1 Introduction

Support Vector Machines (SVMs) are commonly used in supervised learning for classification problems and are based on an optimization framework with well-specified mathematical constraints; concepts such as margin maximization, constrained optimization, and duality play a central role in explaining how decision boundaries are obtained. For this reason, understanding the optimization framework underlying SVMs is essential for explaining why the method performs well in practice.

The aim of this study is to examine the mathematical foundations of Support Vector Machines with an emphasis on the optimization process. The primal and dual formulations of the problem and the associated optimality conditions are discussed in detail. Kernel methods are introduced only to the extent necessary to explain how nonlinear problems fit into the same optimization framework.

In addition to the theoretical discussion, brief reference is made to application areas of SVMs in order to show where the presented mathematical concepts are used in practice.

1.1 Machine Learning

Machine learning is a data-driven subfield of artificial intelligence that focuses on algorithms which learn patterns from training data and use this experience to make decisions or predictions on new, previously unseen data. Fundamentally, learning processes are categorized into two main branches, supervised and unsupervised learning, depending on the structure of the data and the nature of the problem at hand [1].

1.2 Supervised Learning

In supervised learning, which is used for classification and regression problems, a model is first trained on labeled data. Labeled data, where the correct output value corresponding to the inputs is known, forms the basis of the training process, and in this way, the model reduces its error by minimizing its loss function, which measures

the difference between its own predictions and the correct labels; thus, it learns to improve its prediction accuracy [1, 2]. However, preparing labeled data is costly and laborious, especially in fields such as law, medicine, and engineering, as it requires expertise [3–5]. One of the fundamental problem types addressed by supervised learning is classification. In classification problems, the model learns from labeled training data and predicts a categorical target value, that is, a class, for a new sample. For example, determining whether a tumor is benign or malignant by analyzing the symptoms of previous patients can be considered a classification problem [1, 2]. Another fundamental problem type addressed by supervised learning is regression problems, which aim to express the relationship between inputs and a continuous (numerical) output using a mathematical model. Through this model, highly accurate predictions can be produced for previously unseen data [1, 2].

The output produced in regression is typically a real-valued number (often within a problem-specific range); it is used to predict values such as house prices, weather variables, and traffic flow [6–8].

In such regression tasks, methods such as decision trees, linear regression, KNN, and neural networks are commonly used; whereas in classification problems, approaches such as logistic regression, Naive Bayes, KNN, decision trees, neural networks, and SVM are used to assign observations to labeled classes in applications such as spam detection, sentiment analysis, face recognition, medical diagnosis, credit risk, and fraud detection [1, 2].

1.3 Unsupervised Learning

Unsupervised learning, which works with unlabeled data, is used to solve problems such as clustering, density estimation, dimensionality reduction, and outlier detection by revealing hidden structures and patterns within data [1, 9]. The fact that the model does not need labeled data is a significant advantage in terms of cost; however, since there are no labels as in supervised learning, additional validation methods, such as the Elbow Method or Silhouette Analysis, are needed to evaluate and verify the learning outcomes [10–12]. The K-means algorithm groups data

based on similarity and is widely used for customer segmentation as well as image segmentation, including medical imaging applications such as isolating tumors and lesions [13–15]. Hierarchical clustering is another option for segmentation: it can be used to form customer groups for personalized marketing and to split mixed patient populations into more consistent groups using diagnostic and biological data [16, 17]. PCA, on the other hand, is not a clustering method; it reduces the number of variables by projecting high-dimensional data into a lower-dimensional form, which is often convenient for visualization [18].

2 Support Vector Machines

The derivation and formulations presented in this section follow the MIT OpenCourseWare SVM lecture [19].

SVM is a powerful algorithm capable of detecting patterns in complex datasets that cannot be analyzed by simple methods; its core idea was introduced by Vapnik in 1979 [20]. In machine learning, classification problems are based on dividing the dataset given as input into predefined classes. In this context, Support Vector Machines are a classification method that models the separation between data points belonging to different classes through a geometric approach and performs it via a hyperplane defined as the decision boundary. Classification is examined separately as binary classification and multiclass classification; while in multiclass classification the data are divided into three or more categories, in binary classification the data are classified into one of two categories [1]. In this study, the working principle of Support Vector Machines is examined within the framework of binary classification. In binary classification, the set of values that the model output can take consists of two elements, and these values are referred to as class labels. Class labels can be defined, depending on preference, in the form $\{true, false\}$, $\{blue, red\}$, $\{0, 1\}$, $\{+1, -1\}$. In our study, these class labels are determined as $\{+1, -1\}$. The $+$ or $-$ symbols do not have a meaning in terms of positivity/negativity.

In other words, we are interested in estimators of this type;

$$f : \mathbb{R}^D \to \{+1, -1\} \tag{2.1}$$

- D: the number of features (features are the components of the vector representing an example)
- $\mathbb{R}^D$: the space in which each data point has D real-valued features

Let $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ denote the training dataset, where $\mathbf{x}_i \in \mathbb{R}^D$ represents the input feature vectors and $y_i \in \{+1, -1\}$ denotes the corresponding class labels [21].

2.1 Decision Boundary

In the SVM algorithm, a decision boundary, which is geometrically a hyperplane, is defined to classify the data. If the data can be perfectly separated by a linear decision boundary, it is said to be linearly separable. In two dimensions this boundary is a straight line, in three dimensions a plane, and in higher dimensions a hyperplane. To construct this decision boundary, two model parameters are used; the normal vector $\mathbf{w}$, which defines the orientation of the plane and is positioned perpendicular to the surface, and the bias term b, which is a scalar value that determines the position of the plane with respect to the origin. Optimizing these parameters not only separates the classes but also directly affects the margin width, which influences the model's performance. Consider a function $f(\mathbf{x})$. It is a linear function that processes an input data point $\mathbf{x}_i$ using the weight vector $\mathbf{w}$ and the bias term b learned by the model [21]. The explicit form of this function is as follows

$$f(\mathbf{x}_i) = \langle \boldsymbol{w}, \boldsymbol{x}_i \rangle + b \tag{2.2}$$

Here, $\langle \mathbf{w}, \mathbf{x}_i \rangle$ denotes the inner product between the vectors $\mathbf{w}$ and $\mathbf{x}_i$, which is equivalent to the matrix product $\mathbf{w}^T\mathbf{x}$ when vectors are represented in column form.

The value taken by this function determines the position of the corresponding data point in space and its membership to the classes.

The condition $f(\mathbf{x}) = 0$ represents points that lie exactly on the separating hyperplane (decision boundary). Moreover, for a sample $(\mathbf{x}_i, y_i)$, the constraint $y_i f(\mathbf{x}_i) \geq 1$ means the point is correctly classified and lies on or outside the margin. Equivalently, if $y_i = +1$ then $f(\mathbf{x}_i) \geq 1$, and if $y_i = -1$ then $f(\mathbf{x}_i) \leq -1$ [22].

The equation obtained as a result of combining the constraint equations belonging to the positive and negative classes with the target variables is given below

$$y_i\big(\langle \mathbf{w}, \mathbf{x}_i \rangle + b\big) \geq 1 \tag{2.3}$$

This equation is the optimization criterion that guarantees each data point is positioned at least a margin-boundary width away. If we examine it from a geometric perspective; there is a structure in which the decision boundary is first chosen as $f(\mathbf{x}) = 0$, and then this boundary is expanded in both directions until it touches the nearest data points belonging to the classes. The contact points where the closest data points of each class are located are the support vectors. These points form the planes $f(\mathbf{x}) = 1$ and $f(\mathbf{x}) = -1$ and define the margin boundaries (gutter). As a result, the decision boundary, which takes them as reference, lies exactly in the middle of these two parallel auxiliary planes. After the margin boundaries are determined, the decision for a new vector $\mathbf{x}$ follows directly from $f(\mathbf{x})$. A positive output corresponds to class $+1$, a negative output corresponds to class -1, and $f(\mathbf{x}) = 0$ indicates that $\mathbf{x}$ lies on the decision boundary [21, 23].

2.2 A Geometric View of the Margin

The success rate of the SVM algorithm tends to increase as the margin width between the classes increases.

The margin width is defined as the distance between the two margin boundaries (gutters) and is obtained by projecting the difference vector between a support vector from the positive class ($\mathbf{x}_+$) and a support vector from the negative class ($\mathbf{x}_-$) onto

the unit vector normal to the hyperplane, given by $\frac{\mathbf{w}}{\|\mathbf{w}\|}$, via the dot product.

$$\text{Margin Width} = (\mathbf{x}_+ - \mathbf{x}_-) \cdot \frac{\mathbf{w}}{\|\mathbf{w}\|} \tag{2.4}$$

Here, the Euclidean norm of the weight vector is defined as $\|\mathbf{w}\| = \sqrt{\langle \mathbf{w}, \mathbf{w} \rangle}$, which is equivalent to $\sqrt{\mathbf{w}^T \mathbf{w}}$ in matrix form [21].

The reason why we do not use only $\|\mathbf{x}_+ - \mathbf{x}_-\|$ is that, although this expression represents the Euclidean distance between two points, this distance is not necessarily perpendicular to the hyperplanes. The vector $(\mathbf{x}_+ - \mathbf{x}_-)$ may contain components that lie along the hyperplane, i.e., components that are not aligned with the normal direction of the hyperplane, which does not conform to the definition of the margin. The formula used in our analysis measures only the component of the distance between the two points that is orthogonal to the hyperplane. When the inner product operation in the formula used to compute the margin width is carried out, we obtain

$$\text{Width} = \frac{\langle \mathbf{w}, \mathbf{x}_+ \rangle - \langle \mathbf{w}, \mathbf{x}_- \rangle}{\|\mathbf{w}\|} \tag{2.5}$$

For the margin hyperplanes corresponding to the positive and negative classes, respectively, we have:

$$\langle \mathbf{w}, \mathbf{x}_+ \rangle + b = 1, \qquad \langle \mathbf{w}, \mathbf{x}_- \rangle + b = -1. \tag{2.6}$$

When these two expressions are substituted into the margin width formula, it can be seen that the margin width depends only on the norm of the weight vector

$$\text{Width} = \frac{\langle \mathbf{w}, \mathbf{x}_+ \rangle - \langle \mathbf{w}, \mathbf{x}_- \rangle}{\|\mathbf{w}\|} = \frac{(1 - b) - (-1 - b)}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|}. \tag{2.7}$$

In Support Vector Machines, maximizing the margin is equivalent to minimizing the norm of the weight vector $\|\mathbf{w}\|$, since the margin is inversely proportional to this quantity. Instead of minimizing $\|\mathbf{w}\|$ directly, the objective function is formulated in terms of the squared norm $\frac{1}{2}\|\mathbf{w}\|^2$. This choice does not alter the optimal solution, as

$\|\mathbf{w}\|^2$ is a strictly increasing function of $\|\mathbf{w}\|$ for $\|\mathbf{w}\| > 0$. Moreover, $\|\mathbf{w}\|^2 = \mathbf{w}^T\mathbf{w}$ is a quadratic, convex, and everywhere differentiable function, which greatly simplifies the optimization process [24]. The inclusion of the factor $\frac{1}{2}$ eliminates the constant multiplier arising during differentiation, yielding a clean gradient expression. This formulation ensures both mathematical convenience and the existence of a unique global optimum under the imposed constraints. If this expression is minimized in an unconstrained manner, the solution will be $\mathbf{w} = 0$; however, this does not yield a meaningful separating hyperplane. The objective of the SVM model is not only to maximize the margin, but also to ensure that each data point lies on the correct side of the margin and belongs to the correct class. These conditions are enforced through the constraints added to the optimization problem. The mathematical formulation of these constraints was presented in subsection 2.1. In the following section, we examine how the objective function is solved under these constraints [21, 22].

2.3 Constrained Optimization via Lagrange Multipliers and KKT Conditions

In the previous section, it was stated that the expression $\frac{1}{2}\|\mathbf{w}\|^2$ must be minimized under a set of constraints. In order to handle a constrained optimization problem, the problem must first be reformulated as a single expression. This is achieved by incorporating the constraints into the objective function, resulting in the primal formulation of the problem. To include the constraints into the main equation, the method of Lagrange multipliers is employed. For each data point ($\mathbf{x}_i$), the constraints can be expressed mathematically as follows

$$y_i\big(\langle \mathbf{w}, \mathbf{x}_i\rangle + b\big) \geq 1. \tag{2.8}$$

In order to fit this equation into the standard optimization form $g_i(x) \leq 0$, we rewrite it as

$$g_i(\mathbf{w}, b) = 1 - y_i\big(\langle \mathbf{w}, \mathbf{x}_i\rangle + b\big) \leq 0. \tag{2.9}$$

For each constraint $g_i(x)$, a Lagrange multiplier (α_i) is introduced, and an L

function, known as the Lagrange function, is constructed [24].

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = F(x_1, x_2, \ldots, x_n) + \sum_{i=1}^{n} \alpha_i g_i(x_1, x_2, \ldots, x_n). \tag{2.10}$$

The function $F(x)$ represents the objective function, and in our model this function is $\frac{1}{2}\|\mathbf{w}\|^2$. Substituting the available information into the equation, we obtain the Lagrange (primal) formulation of the SVM algorithm

$$L = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \big[y_i \big(\langle \mathbf{w}, \mathbf{x}_i \rangle + b \big) - 1 \big]. \tag{2.11}$$

Lagrange multipliers are weight coefficients assigned to each data point. This weight represents the extent to which the corresponding data point affects the decision boundary [22, 23].

In minimization problems with constraints, the best solution point cannot be defined only by the condition that the derivative is zero, as in unconstrained problems. In such problems, the Lagrange multipliers approach makes the solution process possible by handling the objective function and the constraints within a common mathematical structure. In order to understand the activity of the constraints and the point at which the solution is achieved, the Karush–Kuhn–Tucker conditions are used.

2.3.1 Karush–Kuhn–Tucker (KKT) Conditions

2.3.1.1 Stationarity

This condition describes the point at which the gradient of the objective function and the gradients of the constraints balance each other. In our problem, two opposing effects are present. While the objective function aims to reduce its value, the constraints prevent the solution from leaving the feasible region. When the gradient of the Lagrange function with respect to the primal variables is set to zero, the solution reaches a stable point that satisfies the constraints. First, we take the partial

derivative of the Lagrangian with respect to the weight vector $\mathbf{w}$

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \tag{2.12}$$

Setting this derivative equal to zero yields

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \tag{2.13}$$

This expression indicates that the weight vector $\mathbf{w}$ is determined by the training samples and depends on their labels and the associated Lagrange multipliers. Next, we take the partial derivative of the Lagrangian with respect to the bias term b

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{n} \alpha_i y_i \tag{2.14}$$

Setting this derivative equal to zero gives the following condition

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \tag{2.15}$$

This constraint plays an important role in the derivation of the dual formulation [22–24].

2.3.1.2 Primal Feasibility

In SVM optimization, primal feasibility is the KKT condition that checks whether the solution satisfies the constraints. The decision boundary to be drawn should not leave any point on the wrong side and should not allow any point to enter the margin.

The constraint equation is given as follows

$$y_i\big(\langle \mathbf{w}, \mathbf{x}_i \rangle + b\big) \geq 1 \qquad \forall i \tag{2.16}$$

In order to write this constraint in the standard optimization form, it can be expressed as

$$g_i(\mathbf{w}, b) = 1 - y_i\big(\langle \mathbf{w}, \mathbf{x}_i \rangle + b\big) \leq 0 \tag{2.17}$$

Most of the points in the dataset are so far from the margin that they do not have any effect on determining the position of the decision boundary; these points are called *inactive constraints* ($g_i < 0$). Points that satisfy the constraints exactly at the boundary are called *active constraints* ($g_i = 0$); points that satisfy this condition are candidates to be support vectors [22–24].

2.3.1.3 Dual Feasibility

The objective function attempts to reduce $\|\mathbf{w}\|$, while the margin constraints defined for each data point act as a resistance in order to prevent the hyperplane from shrinking excessively. The multiplier α_i represents the magnitude of this resistance; mathematically, this quantity must be non-negative ($\alpha_i \geq 0$). If this condition is not satisfied, the Lagrangian/KKT framework breaks down and the resulting solution no longer corresponds to the intended maximum-margin problem [22–24].

2.3.1.4 Complementary Slackness

This condition mathematically formalizes the final effect of the points identified in the *Primal Feasibility* section on the model. Its mathematical expression is given as follows:

$$\alpha_i \, g_i(\mathbf{w}, b) = 0 \qquad \forall i \tag{2.18}$$

As stated in *primal feasibility*, if a data point lies outside the margin, that is, in the safe region, the constraint function is less than zero ($g_i < 0$). In this case, for the product to be equal to zero, it must be that $\alpha_i = 0$. Such points do not contribute to the formation of the decision boundary. If a data point lies on the margin line, that is, exactly on the boundary, the constraint function is equal to zero ($g_i = 0$). In this case, $\alpha_i > 0$ may hold, and these points are the support vectors that construct the decision hyperplane.

As a result of the complementary slackness condition, the constraint term being zero does not mean that the constraints are ineffective. Instead, it indicates that the

constraints and the objective function act in balance [22–24].

2.4 Dual Formulation

The SVM optimization problem is usually expressed in two equivalent formulations, called the Primal and the Dual. In the Primal case, the goal is to find suitable values for the weight vector $\mathbf{w}$ and the bias term b such that the margin between the classes is maximized. The Dual formulation approaches the same problem from a different angle by introducing Lagrange multipliers, and the solution depends directly on the training data rather than on the model parameters. The dual form is obtained by substituting the optimal values of the primal variables, which are found by differentiating the Lagrangian with respect to $\mathbf{w}$ and b and setting the results equal to zero, into the original equation. In the dual form, optimization becomes dependent on the dot products of training sample pairs, $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$, which enables working in high-dimensional feature spaces and provides a mathematical foundation for the kernel trick [22–24].

2.4.1 Stationary Conditions

The Lagrangian function for the SVM optimization problem is defined as

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \Big[y_i\big(\langle \mathbf{w}, \mathbf{x}_i \rangle + b\big) - 1 \Big]. \tag{2.19}$$

Expanding the Lagrangian yields

$$\mathcal{L} = \frac{1}{2}\langle \mathbf{w}, \mathbf{w} \rangle - \sum_{i=1}^{n} \alpha_i y_i\big(\langle \mathbf{w}, \mathbf{x}_i \rangle\big) - b\sum_{i=1}^{n} \alpha_i y_i + \sum_{i=1}^{n} \alpha_i. \tag{2.20}$$

To obtain the optimal solution, the partial derivatives of $\mathcal{L}$ with respect to the primal variables are set to zero

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i = \mathbf{0}. \tag{2.21}$$

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i. \tag{2.22}$$

$$\frac{\partial \mathcal{L}}{\partial b} = -\sum_{i=1}^{n} \alpha_i y_i = 0. \tag{2.23}$$

$$\sum_{i=1}^{n} \alpha_i y_i = 0. \tag{2.24}$$

[19, 22]

2.4.2 Expanding the Weight Vector Norm

In this subsection, the inner product of the weight vector with itself is written in summation form.

$$\langle \mathbf{w}, \mathbf{w} \rangle = \left\langle \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \ , \ \sum_{j=1}^{n} \alpha_j y_j \mathbf{x}_j \right\rangle. \tag{2.25}$$

$$\langle \mathbf{w}, \mathbf{w} \rangle = \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle. \tag{2.26}$$

[19]

2.4.3 Eliminating the Weight Vector from the Lagrangian

The term involving the inner product $\langle \mathbf{w}, \mathbf{x}_i \rangle$ is expressed as

$$-\sum_{i=1}^{n} \alpha_i y_i \langle \mathbf{w}, \mathbf{x}_i \rangle. \tag{2.27}$$

Recalling the derived definition of the weight vector $\mathbf{w}$,

$$\mathbf{w} = \sum_{j=1}^{n} \alpha_j y_j \mathbf{x}_j, \tag{2.28}$$

substituting this expression into the term above yields

$$-\sum_{i=1}^{n} \alpha_i y_i \left(\sum_{j=1}^{n} \alpha_j y_j \mathbf{x}_j \right) \cdot \mathbf{x}_i. \tag{2.29}$$

Applying the distributive property of the dot product, the expression expands into a double summation

$$-\sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left\langle \mathbf{x}_j, \mathbf{x}_i \right\rangle. \tag{2.30}$$

Since the dot product is commutative, this expression is equivalent to $-\mathbf{w} \cdot \mathbf{w} = -\|\mathbf{w}\|^2$. Consequently, when combined with the first term of the Lagrangian, the expression simplifies to

$$\frac{1}{2}\|\mathbf{w}\|^2 - \|\mathbf{w}\|^2 = -\frac{1}{2}\|\mathbf{w}\|^2. \tag{2.31}$$

[19]

2.4.4 The Final Dual Lagrangian Form

The resulting equation depends solely on the Lagrange multipliers and the dot products of the samples

$$L(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle. \tag{2.32}$$

In essence, this dual formulation shows us something very practical; to find the best boundary, we do not actually need to know the exact location of every single data point in space. Instead, the entire mathematical formulation reduces to how these points relate to each other through their inner products. This represents a significant advantage, since it implies that the complexity of the model depends on the number of samples rather than the dimensionality of the feature space. By expressing the problem solely in terms of dot products, the computational burden is greatly reduced. Moreover, this formulation forms the basis of kernel methods, enabling the solution of non-linear problems through kernel functions without explicitly mapping the data

into higher-dimensional spaces [19, 22, 23].

2.5 From Hard-Margin to Soft-Margin SVM

The formulation discussed so far corresponds to the hard-margin SVM, where the data are assumed to be perfectly separable. However, in real-world problems, data are usually noisy and cannot be separated exactly by a single linear boundary. To overcome this issue, the soft-margin SVM allows some violations of the margin by introducing slack variables ξ_i and a regularization parameter C. Slack variables ξ_i basically show how much a data point breaks the margin rule. If $\xi_i = 0$, that point is fine and stays outside the margin. If $0 < \xi_i \leq 1$, it is still on the correct side but it falls into the margin. If $\xi_i > 1$, it ends up on the wrong side, meaning it is misclassified. The parameter C decides how strict we are about these violations. When C is large, the model does not want to allow violations, so it tries hard to classify the training data correctly (but it can become sensitive to noisy/outlier points). When C is small, violations are not punished that much, so the model prefers a larger margin and becomes more tolerant to noise, even if that means making a few training mistakes.

The corresponding primal optimization problem becomes

$$\min_{\mathbf{w}, b, \boldsymbol{\xi}} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i. \tag{2.33}$$

subject to

$$y_i\big(\langle \mathbf{w}, \mathbf{x}_i \rangle + b\big) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, n. \tag{2.34}$$

The hinge loss value is simply a way to measure how "good" or "bad" the prediction is for a given example, and even when the label is predicted correctly, how well the margin requirement is satisfied. Let $f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b$. Rearranging the constraint $y_i f(\mathbf{x}_i) \geq 1 - \xi_i$ gives $\xi_i \geq 1 - y_i f(\mathbf{x}_i)$. In addition, since the condition $\xi_i \geq 0$ must also be satisfied, ξ_i has to meet two lower bounds simultaneously.

Since the objective penalizes $\sum_{i=1}^{n} \xi_i$, the optimal slack satisfies

$$\xi_i = \max\big(0,\ 1 - y_i f(\mathbf{x}_i)\big). \tag{2.35}$$

This expression is exactly the same as the hinge loss.

In the dual formulation, the effect of the regularization parameter C appears as an upper bound on the Lagrange multipliers

$$0 \leq \alpha_i \leq C, \quad i = 1, \ldots, n. \tag{2.36}$$

[22]

2.6 Kernels

The dual formulation (2.32) reveals that, in the SVM optimization problem, only the inner products between samples play a role. In other words, the solution relies on the relationships among data points rather than writing the parameters directly. This result is important because it shows that SVM can also solve non-linear problems using kernel methods. Based on this structure that emerges in the dual formulation, a non-linear feature transformation $\boldsymbol{\phi}(\mathbf{x}_i)$ can be defined for each data point $\mathbf{x}_i$. In this case, the inner product terms in the dual SVM, $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$, are expressed instead as $\langle \boldsymbol{\phi}(\mathbf{x}_i), \boldsymbol{\phi}(\mathbf{x}_j) \rangle$. With the kernel method, the data are represented in a higher-dimensional feature space, and the SVM continues to separate the classes there using a linear hyperplane. However, when this linear separation is mapped back to the original input space, the decision boundary takes on a curved (non-linear) form. The kernel method makes it possible to perform the feature transformation implicitly by defining a function that quantifies the similarity between two data points. In this context, a kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$ satisfies

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \boldsymbol{\phi}(\mathbf{x}_i),\ \boldsymbol{\phi}(\mathbf{x}_j) \rangle\,, \tag{2.37}$$

for some feature map $\boldsymbol{\phi}$. This technique is known as the Kernel Trick. For the kernel trick to be valid within the SVM framework, the kernel function must be positive definite, ensuring that the associated kernel matrix represents a valid inner product in the feature space. When a kernel is employed, the dual SVM objective function takes the following form

$$\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \, k(\mathbf{x}_i, \mathbf{x}_j) \tag{2.38}$$

In this approach, only the inner product terms are replaced by the kernel function, while the remaining structure of the optimization problem remains unchanged.

To illustrate the use of the kernel method for real-valued data, the mathematical expressions of commonly used kernel functions are given below.

The linear kernel is given by

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle. \tag{2.39}$$

The polynomial kernel is given by

$$k(\mathbf{x}_i, \mathbf{x}_j) = \big(\langle \mathbf{x}_i, \mathbf{x}_j \rangle + c\big)^d. \tag{2.40}$$

where the degree parameter d governs model complexity by setting the polynomial order of the decision function, and the constant c tunes the weight given to higher-order components.

The Gaussian, or radial basis function (RBF), kernel is defined as

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right). \tag{2.41}$$

where $\sigma > 0$ specifies the kernel width. It governs the rate at which the similarity value drops as two observations move farther apart.

Finally, the sigmoid kernel is expressed as

$$k(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\kappa \langle \mathbf{x}_i, \mathbf{x}_j \rangle + c)\,. \tag{2.42}$$

where κ controls the scaling of the inner product and c acts as a shift parameter. The parameter κ influences the sensitivity of the kernel to differences between data points, while c adjusts the operating region of the function. Although the sigmoid kernel provides a nonlinear similarity measure, it is not positive semidefinite for all parameter choices. Consequently, the convexity of the SVM dual optimization problem is not always guaranteed, and careful parameter selection is required.

These kernel functions are used directly in the dual SVM formulation by replacing the inner product terms [22].

3 Applications of Support Vector Machines

This section provides a brief overview of several application areas in which Support Vector Machines have been used. The examples presented below aim to illustrate how the theoretical framework discussed earlier appears in practical classification problems.

Here are some representative medical applications of Support Vector Machines. SVMs have been used with clinical records, such as electronic medical data, to build prognostic models for cardiovascular conditions and risk prediction using large datasets [25]. In medical imaging, SVM classifiers have been applied to MRI images to differentiate between types of tumors based on extracted features [26]. The use of SVM-based methods for diagnosis and outcome prediction in healthcare has been discussed in review studies [27]. SVM models have also been used for classification tasks involving biomedical text data and structured clinical records [28]. In medical classification, performance is frequently evaluated using the ROC curve. It relates sensitivity, meaning how many truly diseased patients are correctly detected, to the false positive rate, meaning how many truly healthy individuals are incorrectly labeled as diseased, as the classification rule is varied. Since SVM outputs

are not naturally given as probabilities, ROC analysis can be performed by training several weighted SVM models that put different emphasis on false positives versus false negatives, producing different operating points on the ROC plot. This idea and the construction of confidence bands for SVM-based ROC curves have been demonstrated in a healthcare prediction study, including an application on breast cancer patients for predicting treatment response [29].

Beyond medical applications, Support Vector Machines have also been used in image and pattern recognition tasks. In image classification problems, visual data are represented using feature vectors, and SVM models are applied to separate different object or pattern classes [23]. Common examples include face recognition and object classification. In these tasks, SVMs are used to separate visual categories based on extracted features [30]. These examples show that SVMs can work well for image-based problems with high-dimensional feature representations.

Support Vector Machines have also been used in text and document classification tasks. In these problems, documents are represented as high-dimensional feature vectors, and SVM models are used to assign them to predefined categories, such as in text categorization and document filtering [31]. This shows that SVMs can be applied effectively to text data with large feature spaces.

Another application area is security and anomaly detection. In this context, SVM models are applied to data such as network traffic or system activity in order to detect unusual behavior. One example is intrusion detection, where SVM-based approaches are used to separate normal activity from abnormal or suspicious patterns [32]. This shows that SVMs can be applied when the main goal is to identify deviations from normal behavior.

4 Conclusion

As a result of this study, it is shown that the SVM decision boundary can be obtained through a systematic optimization process. Through the derivation of the primal and dual formulations, it becomes clear how margin maximization and constraints determine the position of the decision boundary. In particular, the analysis shows

that the final solution depends on inner products between training samples, which explains how SVM focuses on the relationships between data points rather than their explicit coordinates.

References

[1] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. New York: Springer, 2006.

[2] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. 2nd ed. New York: Springer, 2009.

[3] Burr Settles. *Active Learning Literature Survey*. Tech. rep. 1648. University of Wisconsin–Madison, 2010.

[4] Dan Hendrycks et al. "CUAD: An Expert-Annotated NLP Dataset for Legal Contract Review". In: *arXiv preprint arXiv:2103.06268* (2021). DOI: 10.48550/arXiv.2103.06268.

[5] Adrian Krenzer, Kevin Makowski, Amar Hekalo, et al. "Fast machine learning annotation in the medical domain: a semi-automated video annotation tool for gastroenterologists". In: *BioMedical Engineering OnLine* 21 (2022), p. 33. DOI: 10.1186/s12938-022-01001-x.

[6] IBM. *Classification versus regression*. Accessed 2025-12-29. URL: https://www.ibm.com/think/topics/classification-vs-regression (visited on 12/29/2025).

[7] Microsoft Tech Community. *Training a Time-Series Forecasting Model Using Automated Machine Learning*. Aug. 2024. URL: https://techcommunity.microsoft.com/blog/educatordeveloperblog/training-a-time-series-forecasting-model-using-automated-machine-learning/4212384 (visited on 12/29/2025).

[8] Yaguang Li et al. “Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting”. In: *International Conference on Learning Representations (ICLR)*. 2018. URL: https://openreview.net/forum?id=SJiHXGWAZ (visited on 12/29/2025).

[9] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. Cambridge, MA: MIT Press, 2012.

[10] Maria Halkidi, Yannis Batistakis, and Michalis Vazirgiannis. “On Clustering Validation Techniques”. In: *Journal of Intelligent Information Systems* 17.2-3 (2001), pp. 107–145. URL: https://web.itu.edu.tr/sgunduz/courses/verimaden/paper/validity_survey.pdf (visited on 12/29/2025).

[11] Erich Schubert. “Stop using the elbow criterion for k-means and how to choose the number of clusters instead”. In: *ACM SIGKDD Explorations Newsletter* (2023). URL: https://kdd.org/exploration_files/p36-elbow.pdf (visited on 12/29/2025).

[12] Peter J. Rousseeuw. “Silhouettes: A graphical aid to the interpretation and validation of cluster analysis”. In: *Journal of Computational and Applied Mathematics* 20 (1987), pp. 53–65. DOI: 10.1016/0377-0427(87)90125-7. URL: https://wis.kuleuven.be/stat/robust/papers/publications-1987/rousseeuw-silhouettes-jcam-sciencedirectopenarchiv.pdf (visited on 12/29/2025).

[13] J. A. Hartigan and M. A. Wong. “A K-Means Clustering Algorithm”. In: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 28.1 (1979), pp. 100–108. DOI: 10.2307/2346830.

[14] Kayalvily Tabianan, Shubashini Velu, and Vinayakumar Ravi. “K-Means Clustering Approach for Intelligent Customer Segmentation Using Customer Purchase Behavior Data”. In: *Sustainability* 14.12 (2022). DOI: 10.3390/su14127243. URL: https://www.mdpi.com/2071-1050/14/12/7243 (visited on 12/30/2025).

[15] Dzung L. Pham, Chenyang Xu, and Jerry L. Prince. “Current methods in medical image segmentation”. In: *Annual Review of Biomedical Engineering* 2 (2000), pp. 315–337. DOI: 10.1146/annurev.bioeng.2.1.315.

[16] Charu C. Aggarwal and Chandan K. Reddy, eds. *Data Clustering: Algorithms and Applications*. Boca Raton, FL: CRC Press, 2014.

[17] Michael B. Eisen et al. "Cluster analysis and display of genome-wide expression patterns". In: *Proceedings of the National Academy of Sciences* 95.25 (1998), pp. 14863–14868. DOI: 10.1073/pnas.95.25.14863.

[18] I. T. Jolliffe. *Principal Component Analysis*. 2nd ed. New York: Springer, 2002.

[19] Patrick H. Winston. *Lecture 16: Learning: Support Vector Machines*. 6.034 Artificial Intelligence, Fall 2010 (video lecture). MIT OpenCourseWare. 2010. URL: https://ocw.mit.edu/courses/6-034-artificial-intelligence-fall-2010/resources/lecture-16-learning-support-vector-machines/ (visited on 12/30/2025).

[20] Dustin Boswell. *Introduction to Support Vector Machines*. Unpublished tutorial/notes (PDF). Aug. 2002. URL: https://pzs.dstu.dp.ua/DataMining/svm/bibl/IntroToSVM.pdf (visited on 12/30/2025).

[21] Marc Peter Deisenroth, A. Aldo Faisal, and Cheng Soon Ong. *Mathematics for Machine Learning*. Cambridge University Press, 2020. URL: https://mml-book.github.io/book/mml-book.pdf (visited on 12/30/2025).

[22] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.

[23] Christopher J. C. Burges. "A Tutorial on Support Vector Machines for Pattern Recognition". In: *Data Mining and Knowledge Discovery* 2 (1998), pp. 121–167. DOI: 10.1023/A:1009715923555. URL: https://www.di.ens.fr/~mallat/papiers/svmtutorial.pdf (visited on 12/30/2025).

[24] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. URL: https://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf (visited on 12/30/2025).

[25] X. Zhou et al. "Support Vector Machine-Based Mining of Electronic Medical Records to Predict the Prognosis of Acute Myocardial Infarction". In: *Journal of Healthcare Engineering* (2022). Available on PubMed Central.

[26] R. Alrais and N. Elfadil. *Support Vector Machine (SVM) for Medical Image Classification of Tumorous Data*. Available on ResearchGate. 2020.

[27] R. Guido. "Support Vector Machines in Healthcare: Review and Applications". In: *Information* (2024). Open access (MDPI).

[28] "Comparative Analysis of Machine Learning Algorithms for Biomedical Text Classification". In: *Medicine Science* (2021). Open access.

[29] Daniel J. Luckett et al. "Receiver Operating Characteristic Curves and Confidence Bands for Support Vector Machines". In: *Biometrics* 77.4 (Dec. 2021). Published online 31 August 2020, pp. 1422–1430. DOI: 10.1111/biom.13365. URL: https://academic.oup.com/biometrics/article/77/4/1422/7446213.

[30] Bernd Heisele, Purdy Ho, and Tomaso Poggio. "Face Recognition with Support Vector Machines: Global versus Component-Based Approach". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2001.

[31] Thorsten Joachims. "Text Categorization with Support Vector Machines: Learning with Many Relevant Features". In: *Proceedings of the European Conference on Machine Learning*. 1998.

[32] Pavel Laskov, Christin Schafer, and Igor Kotenko. "Learning intrusion detection: Supervised or unsupervised?" In: *Image Analysis and Processing* (2005).