Investigation

A detailed example of binary classification by quadratic kernel and its associated decision function

Un ejemplo detallado de clasificación binaria mediante kernel cuadrático y su función de decisión asociada

Yeisson Alexis Acevedo Agudelo
Oscar Mario Londoño Duque

Research Journal

G.I.E Pensamient Matemátic MAIC

Vol. XIV, Núm 1, pp. 005–017, ISSN 2174-0410
Recepción: 02 Mar'23; Aceptación: 17 Jul'23

April 2024

Resumen

En este artículo se pretende mostrar los detalles de cálculo internos a los algoritmos de clasificación binaria mediante máquinas de vectores de soporte (SVM). En particular, se presenta un ejemplo detallado del uso del algoritmo y el cálculo puntual de los parámetros asociados al sistema de entrenamiento mediante un kernel cuadrático. Las cuentas y definiciones presentadas en este trabajo, pueden ser de beneficio para los estudiantes de ciencias de la computación, ingeniería, matemáticas y para cualquier persona interesada en aprender sobre inteligencia artificial.

Palabras Clave: Clasificación Binaria, Funciones de Similitud, Función de Clasificación; Inteligencia Artificial.

Abstract

This article aims to illustrate the internal computation details of binary classification algorithms using Support Vector Machines (SVM). Specifically, a detailed example of the algorithm's usage and the precise computation of parameters associated with the training system using a quadratic kernel are presented. The computations and definitions presented in this work may be of benefit to students of computer science, engineering, mathematics, and anyone interested in learning about artificial intelligence.

Keywords: Binary Classification, Similarity Functions, Classification Function; Artificial Intelligence.

1. Introduction

Artificial Intelligence (AI) has transformed numerous fields, from medicine and industry to entertainment and education. The ability of AI to analyze vast amounts of data and learn complex patterns has led to significant advancements in problem-solving and decision-making [1]. Furthermore, AI can enhance the efficiency and productivity of businesses and organizations, potentially exerting a significant impact on the global economy.

Binary classification is one of the most common applications of AI [2, 3, 4], employed to segregate datasets into two distinct categories. However, this technique still poses significant challenges and issues, such as lack of precision and a tendency to overfit the training data. These issues can lead to inaccurate results and costly errors in decision-making.

To overcome these challenges, advanced binary classification algorithms, such as classification kernels, have been developed [5, 6]. These algorithms utilize similarity functions to map the data into a high-dimensional space, where it is easier to separate different categories. Furthermore, classification kernels are efficient in terms of computational resource usage, making them ideal for deployment in real-time systems and online applications.

This article will present a detailed example to comprehend the utilization of the quadratic kernel in the context of Support Vector Machines (SVM) for binary classification. In addition to exploring the basics of SVM, specific techniques and necessary steps for constructing a decision or classification function will be detailed, derived from the solution to the optimization problem with Lagrange coefficients. This article aims to serve as a practical guide for constructing decision functions using different kernels in SVM for binary classification problems. It is expected that, through understanding the internal structure of SVM and using different kernels, the concepts presented here can be applied in a variety of classification contexts and different programming languages. With this in mind, this article will not only present the detailed example but also the theoretical concepts necessary to comprehend the technical details behind constructing a decision function in SVM.

2. Preliminaries

In this section, we will establish some fundamental concepts necessary to understand binary classification using a separating hyperplane through Support Vector Machine (SVM).

Binary Classification Binary classification is a fundamental problem in the field of machine learning. It involves assigning instances to one of the two possible classes. For example, it could be the classification of emails as ′spam′ or ′non-spam′ or the detection of bank transactions as ′fraudulent′ or ′legitimate′. In our case, we will focus on the classification of instances into two classes labeled as ′positive′ and ′negative′.

Feature Vector. It is a mathematical entity representing a magnitude and direction in a multidimensional space. In the context of classification, each instance is represented by a vector in a vector space. A vector can have multiple components, representing the features of an instance. For example, if classifying images of fruits, the vector components could be the size, color, and texture of the fruit.

Feature Space. It is a representation in a dimensional space where each instance is described by a feature vector. Each component of the vector represents a specific feature. By utilizing a feature space, we aim to transform instances into a space where it is easier to find a linear separation between classes.

Support Vector Machines (SVM). SVMs are a widely used technique in machine learning for binary classification. Their primary objective is to find a separating hyperplane in the feature space that maximizes the margin between instances of different classes. This hyperplane becomes a decision boundary for classifying new instances

Optimization and Training. The training of an SVM model involves finding the optimal parameters that define the separating hyperplane. This is achieved by formulating the problem as an optimization problem and employing optimization techniques, such as quadratic programming or maximizing the objective function. The goal is to find the hyperplane that generalizes well for new instances and minimizes classification errors.

3. Development

Suppose we want to classify with labels 1 or -1, the vectors

$$\mathbf{x} = (-2,1,2) \text{ y } \mathbf{y} = (2,1,3). \tag{1}$$

These could be, for example, the transformed coordinates for measurements obtained from two new patients who underwent a liver examination [7] (size, dimension, degree of alcoholism), and we want to classify whether each of these two patients has a fatty liver or not, using the labels $1, -1$, where 1 represents a healthy classification for a patient.

Additionally, suppose a quadratic kernel was selected for this task, given by $K(\mathbf{x},\mathbf{y}) = (\mathbf{x}\cdot\mathbf{y}+1)^2$, where $\mathbf{x}\cdot\mathbf{y}$ is the dot product between vectors $\mathbf{x}$ and $\mathbf{y}$.

Remark: According to the Mercer's theorem, not every function can be a kernel; in fact, the theorem states that the function K must be symmetric and positive semi-definite [8, 9]. This is: i) for all $\mathbf{x},\mathbf{y} \in \mathbb{R}^n$, it holds that $K(\mathbf{x},\mathbf{y}) = K(\mathbf{y},\mathbf{x})$, ii) for any finite set of n vectors $A = \mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3,\cdots,\mathbf{x}n$, the Gram matrix $Gij = K(\mathbf{x}_i,\mathbf{x}j)$, with $i,j = 1,2,3,...,n$; must be positive semi-definite (Eigenvalues greater than or equal to zero; or equivalently $\mathbf{x}i^T G\mathbf{x}_i \geq 0$). Various functions satisfy the conditions to be kernels; in [8], the reader can find different definitions of kernels: linear, quadratic, polynomial, Gaussian (RBF), sigmoid, among others.

Next, the previously selected kernel is applied to the vectors $\mathbf{x}$, $\mathbf{y}$ from (1):

$$K(\mathbf{x},\mathbf{y}) = K((-2,1,2),(2,1,3)) = [(-2)(2)+(1)(1)+(2)(3)+1]^2 = 16.$$

we can observe that $\mathbf{x}$ and $\mathbf{y}$ could be similar or likely to belong to the same class, but this value 16 is relative (two vectors will be similar or close if, when applying the kernel 'similarity function', a non-close-to-zero number is obtained [10]), and therefore, it is not decisive for classifying vectors $\mathbf{x}$, $\mathbf{y}$. Consequently, a decision function $f(\mathbf{x})$ must be constructed using a prior training set with known labels and, of course, grounded in the selected kernel $K(\mathbf{x},\mathbf{y})$.

By the above, let's consider the following three training vectors:

$$x_1 = (2,1,2);\ x_2 = (3,1,0);\ x_3 = (1,3,1) \text{ with their respective labels } y_1 = -1;\ \ y_2 = 1;\ \ y_3 = 1. \tag{2}$$

Remark:

It is common practice to randomly select 70 % from a previously known database for the training of the machine learning method (Machine Learning - AI). On the other hand, the remaining 30 % of the data, along with their respective labels $-1, 1$, is kept for evaluating the accuracy of the constructed model or selection function [11].

Continuing with our goal of establishing the decision function $f(\mathbf{x})$, we must first obtain the following Gram matrix [12], applying the kernel to all training data given in (2):

$$G_{ij} = K(\mathbf{x}_i, \mathbf{x}_j) = \begin{bmatrix} K(x_1, x_1) & K(x_1, x_2) & K(x_1, x_3) \\ K(x_2, x_1) & K(x_2, x_2) & K(x_2, x_3) \\ K(x_3, x_1) & K(x_3, x_2) & K(x_3, x_3) \end{bmatrix} = \begin{bmatrix} 100 & 64 & 64 \\ 64 & 121 & 49 \\ 64 & 49 & 144 \end{bmatrix}. \tag{3}$$

We also need to obtain the cross-label matrix (or label Gram matrix) using (2):

$$Y_{ij} = \mathbf{y}_i \mathbf{y}_j = \begin{bmatrix} y_1 y_1 & y_1 y_2 & y_1 y_3 \\ y_2 y_1 & y_2 y_3 & y_2 y_3 \\ y_3 y_1 & y_3 y_2 & y_3 y_3 \end{bmatrix} = \begin{bmatrix} 1 & -1 & -1 \\ -1 & 1 & 1 \\ -1 & 1 & 1 \end{bmatrix}. \tag{4}$$

Now, it is important to present the decision function $f(\mathbf{x})$, which, according to [8], is given by:

$$f(\mathbf{x}) = Sign\left[\sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b\right] ; \text{ where:} \tag{5}$$

(I) α_i, are the Lagrange coefficients (They are found by solving the associated convex optimization problem with the training data; this quadratic optimization problem is solved using the method of Lagrange multipliers).

(II) y_i are the respective labels of the training data.

(III) $K(\mathbf{x}, x_i)$, is the similarity function (kernel) applied to the vector $\mathbf{x}$ to be classified with each of the training vectors.

(IV) b is the bias term (also known as the threshold or as the value of the intersection of the separating hyperplane with the vertical axis).

The main problem one always faces in (5) is determining the Lagrange coefficients α_i, which are necessary to obtain the decision function. In the example we are considering, we have vectors in $\mathbb{R}^3$, which, as will be seen below, will not be very difficult to solve for the optimization problem. However, the complexity will increase depending on the binary classification problem addressed, due to the amount of data, the choice of similarity function (Kernel), the number of vector components, and the number of training vectors available. Hence, the indispensable assistance of computers, especially as more and more training data is introduced to enhance the models. Performing the calculations manually becomes complex, and that's why support machines or assistance is required (we refer to such assistance as artificial intelligence (AI)).

To calculate the Lagrange coefficients α_i, we proceed with the following steps:

1. We formulate the objective function: The objective function for the optimization problem is known as the 'Lagrangian for optimization problems' [13], which, for estimating the separating plane, is given by:

$$L_P = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \lambda_i \left(y_i \left(\mathbf{w} \cdot \mathbf{x_i} + b\right) - 1\right), \tag{6}$$

Equation (6) is also known as the Primal Lagrangian (Initial separation region minimization problem), and its equivalent, the Dual Lagrangian [8], is given by:

$$L_D = \sum_{i=1}^{n} \lambda_i - \frac{1}{2}\sum_{i,j} \lambda_i \lambda_j y_i y_j K(\mathbf{x_i}, \mathbf{x_j}); \quad \text{with the constraint } \sum_{i=1}^{n} \alpha_i y_i = 0. \tag{7}$$

Calculating the Lagrange coefficients using the Dual (7) is always the best option, as the expression allows us to use the matrices (3) and (4), which are known data. Substituting $\lambda_i = \alpha_i$, we have, for our example, that expression (7) transforms into:

$$L(\alpha) = \sum_{i=1}^{3} \alpha_i - \frac{1}{2}\sum_{i=1}^{3}\sum_{j=1}^{3} \alpha_i \alpha_j y_i y_j K(x_i, x_j), \text{ with the constraint } \sum_{i=1}^{3} \alpha_i y_i = 0. \tag{8}$$

Remark: The original minimization problem for the primal Lagrangian L_P in (6) becomes a maximization problem for the dual Lagrangian L_D in (7) (See [14]).

2. Expanding in (8), we obtain:
$L(\alpha) = \alpha_1 + \alpha_2 + \alpha_3 - \frac{1}{2}[(\alpha_1 y_1)(\alpha_1 y_1)K(x_1, x_1) + (\alpha_1 y_1)(\alpha_2 y_2)K(x_1, x_2) + (\alpha_1 y_1)(\alpha_3 y_3)K(x_1, x_3) + (\alpha_2 y_2)(\alpha_1 y_1)K(x_2, x_1) + (\alpha_2 y_2)(\alpha_2 y_2)K(x_2, x_2) + (\alpha_2 y_2)(\alpha_3 y_3)K(x_2, x_3) + (\alpha_3 y_3)(\alpha_1 y_1)K(x_3, x_1) + (\alpha_3 y_3)(\alpha_2 y_2)K(x_3, x_2) + (\alpha_3 y_3)(\alpha_3 y_3)K(x_3, x_3)]$,

   Using (3), (4), and simplifying, we obtain:

$$L(\alpha) = \alpha_1 + \alpha_2 + \alpha_3 - \frac{1}{2}(100\alpha_1\alpha_1 - 128\alpha_1\alpha_2 - 128\alpha_1\alpha_3 + 121\alpha_2\alpha_2 + 98\alpha_2\alpha_3 + 144\alpha_3\alpha_3). \tag{9}$$

3. Formulate the optimization problem: the optimization problem is to maximize the objective function $L(\alpha)$ in (9), subject to the constraints $\alpha_i \geq 0$ and $\sum_{i=1}^{3} \alpha_i y_i = 0$.

4. Solve the optimization problem: to find the stationary points, we differentiate $L(\alpha)$ with respect to each α_i and set them equal to zero. Thus, from (9), we have:

$$\frac{\partial L}{\partial \alpha_1} = 1 - 100\alpha_1 + 64\alpha_2 + 64\alpha_3 = 0.$$
$$\frac{\partial L}{\partial \alpha_2} = 1 + 64\alpha_1 - 121\alpha_2 - 49\alpha_3 = 0.$$
$$\frac{\partial L}{\partial \alpha_3} = 1 + 64\alpha_1 - 49\alpha_2 - 144\alpha_3 = 0.$$

   with $\sum_{i=1}^{3} \alpha_i y_i = 0$.

   Solving the above system of linear equations, we obtain:

$$\alpha_1 = \frac{25711}{818268}; \ \alpha_2 = \frac{3895}{204567}; \ \alpha_3 = \frac{984}{68189}. \tag{10}$$

   Note that $\alpha_1, \alpha_2, \alpha_3$ satisfy $\sum_{i=1}^{3} \alpha_i y_i \approx 0$, in general, they satisfy the equation (9), for $L(\alpha) = 0$.

Finally, we proceed to calculate the threshold (also known as bias or intersection term) b. This can be calculated using the values α_i from (10) and the expression (see section 7.1.2 in [15]):

$$b = \frac{1}{y_i} - \sum_{j=1}^{n} \alpha_j y_j K(x_i, x_j), \tag{11}$$

for any α_i such that $0 \leq \alpha_i$. In this case, we can use α_1, α_2 and α_3 as they are positive. Thus, in (11), it holds

$$b = \frac{1}{y_1} - \alpha_1 y_1 K(x_1, x_1) - \alpha_2 y_2 K(x_1, x_2) - \alpha_3 y_3 K(x_1, x_3),$$

$$b = -1 - \frac{25711}{818268}(-1)(100) - \frac{3895}{204567}(1)(64) - \frac{984}{68189}(1)(64) \approx 0.$$

Thus, with the threshold $b = 0$, we finally have the decision function $f(\mathbf{x})$ defined in (5), for our example, determined by

$$f(\mathbf{x}) = Sign\left[\sum_{i=1}^{3} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i)\right], \tag{12}$$

with $0 \leq \alpha_i$ given by (10).

We proceed then to fulfill our initial objective, classifying the vector $\mathbf{x} = (-2, 1, 2)$. Using $\mathbf{x}$, (2), (10) and the kernel operator in the decision function (12), with which we obtain:

$$f(\mathbf{x}) = Sign\left[\frac{25711}{818268}(-1)(2)^2 + \frac{3895}{204567}(1)(-4)^2 + \frac{984}{68189}(1)(4)^2\right],$$

$$f(\mathbf{x}) = Sign\left[\frac{27947}{68189}\right],$$

$$f(\mathbf{x}) = 1.$$

Similarly, for $\mathbf{y} = (2, 1, 3)$, we obtain

$$f(\mathbf{y}) = Sign\left[\frac{25711}{818268}(-1)(12)^2 + \frac{3895}{204567}(1)(8)^2 + \frac{984}{68189}(1)(9)^2\right],$$

$$f(\mathbf{y}) = Sign\left[-\frac{437204}{204567}\right],$$

$$f(\mathbf{y}) = -1.$$

Then, the proposed objective in our example was achieved, and the respective classifications are$f(\mathbf{x}) = 1, \ f(\mathbf{y}) = -1$.

The decision function $f(\mathbf{x})$ is, in reality, equivalent to establishing a separation function through a separating plane (or separating hyperplane in high dimensions). To corroborate this assertion, the following is a way to construct the separation plane in the feature space, using the Lagrange coefficients α_i obtained in (10) and the value of the bias term b:

Finding the separation hyperplane.

The separating plane in binary SVM classification algorithms is defined by a linear equation of the form $\mathbf{w}^T\mathbf{X} + b = 0$, where $\mathbf{w}$ is a vector normal to the hyperplane that defines the direction of the hyperplane, $\mathbf{X}$ is the feature vector (attributes or input) representing the object to be classified in the feature space, and b is a scalar indicating the distance from the hyperplane to the origin or decision threshold. The expression $\mathbf{w}^T$ denotes the transpose of the vector $\mathbf{w}$. In this context, the task of the SVM algorithm is to find the vector $\mathbf{w}$ and the scalar b that define the optimal separating hyperplane between the two classes.

The vector **w** defines the direction of the hyperplane and is perpendicular to it. Similarly, **w** maximizes the distance between the hyperplane and the points of each class in the feature space. According to [16], the vector **w** is calculated using the expression

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i x_i. \tag{13}$$

Substituting the data for our example, given in (2) and (10), we obtain in (13):

$$\mathbf{w} = \sum_{i=1}^{3} \alpha_i y_i x_i = \frac{25711}{818268}(-1)(2,1,2) + \frac{3895}{204567}(1)(3,1,0) + \frac{984}{68189}(1)(1,3,1),$$

$$\mathbf{w} = \left(\frac{3563}{409134}, \frac{8431}{272756}, -\frac{19807}{409134}\right).$$

Furthermore, as in our example $b = 0$, we finally have the separating plane $\mathbf{w}^T\mathbf{X} + b = 0$, it is:

$$\frac{3563}{409134}x_{.1} + \frac{8431}{272756}x_{.2} - \frac{19807}{409134}x_{.3} = 0, \tag{14}$$

for all vector $\mathbf{X} = \begin{pmatrix} x_{.1} \\ x_{.2} \\ x_{.3} \end{pmatrix}$ of attributes or inputs.

Once the separation plane (Hyperplane or decision boundary) has been found, it is also possible to use it to classify new points using the following three rules:

(a) Take the feature vector of the new point to be classified and substitute it into the equation of the separation plane.

(b) If, upon evaluation, the result is greater than zero, then the point is classified as belonging to the positive class $(+1)$, and if it is less than zero, then it is classified as belonging to the negative class (-1).

(c) If the result is equal to zero, then the point is exactly on the separation plane and can be classified as belonging to either of the two classes, depending on the convention being used (this situation rarely occurs).

It is possible to summarize the three rules above implicitly through the following expression:

$$f(\mathbf{x}) = Sign\left(\mathbf{w}^T\mathbf{X} + b\right).$$

As a result, the equivalence is finally fulfilled

$$f(\mathbf{x}) = Sign\left(\mathbf{w}^T\mathbf{X} + b\right) = Sign\left[\sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b\right].$$

4. Some considerations and model validation

It is possible that the classifications obtained with the separation hyperplane (derived from the Lagrange coefficients) do not match the classifications obtained directly from the decision function (obtained with the Lagrange coefficients and the selected kernel) on new points. This discrepancy can be attributed to various factors:

1. Approximation error: The kernel function uses an approximation to map the data to a higher-dimensional feature space. If this approximation, along with the estimation of Lagrange coefficients, is not accurate, the final decision function $f(\mathbf{x})$ may not capture all the complexities of the data, leading to classification errors.

2. Choice of kernel: The choice of the kernel can impact the model's generalization ability. Some kernels are more suitable for certain types of data than others, so choosing the wrong kernel can result in a suboptimal model.

3. Sample bias: The Lagrange coefficients and the separation plane derived from them are based on the training data. If the training data sample is biased or not representative of the population, the separation plane may not generalize well to new points.

In particular, this last case is the factor in which the example we are considering fails, as the data is not real in the sense that it was not selected from a particular database but rather chosen randomly to illustrate in detail the functioning of the binary classification algorithm using SVM (AI).

In general, it is important to remember that results obtained with machine learning models are always approximations, and practical results may slightly differ from theoretical results. It is always crucial to evaluate the performance of a model on independent test data to ensure that the model generalizes well to new data.

To evaluate the performance of a binary SVM model, it is necessary to use appropriate evaluation metrics for binary classifiers [17]. Some of the most common metrics include:

1. Accuracy:
$$E = \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}},$$
where TP: True Positives (positively labeled instances correctly classified). TN: True Negatives (negatively labeled instances correctly classified). FP: False Positives (negatively labeled instances incorrectly classified as positive). FN: False Negatives (positively labeled instances incorrectly classified as negative).

2. Precision:
$$P^* = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}\,.$$
Precision P^*, gauges the model's ability to accurately identify positive instances, disregarding instances of negative classification errors.

3. Sensitivity or True Positive Rate
$$S = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}\,.$$
Sensitivity measures the model's ability to accurately detect positive instances but does not account for incorrectly classified negative instances (false positives), which constitutes a significant limitation.

4. $F1$ Score:
$$F1\text{ score} = \frac{2\cdot\text{Precision}\cdot\text{Sensitivity}}{\text{Precision}+\text{Sensitivity}} = \frac{P^*.\,S}{P^*+S}\,.$$

The F1 Score combines both precision and sensitivity into a single measure, making it useful when seeking a balance between the two.

Validation

To validate the constructed model, for our example, let's recall that our decision function is given by $f(\mathbf{x}) = Sign\left[\sum_{i=1}^{3} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i)\right]$, as presented in (12). We will use the Accuracy measure E (also known as 'Exactitud' in Spanish [18]), with $0 \leq E \leq 1$, to assess the performance of our function $f(\mathbf{x})$ or binary classification SVM model.

Consider the following feature vectors and their respective labels as real data presented in the Table 1, for model validation:

Tabla 1: Validation data

Feature vector	*Label*
$(-2,1,2)$	1 $(Positive)$
$(2,1,2)$	-1 $(Negative)$
$(3,1,0)$	1 $(Positive)$
$(1,3,1)$	1 $(Positivo)$
$(0,2,1)$	-1 $(Negative)$
$(3,-1,2)$	1 $(Positive)$
$(2,1,3)$	-1 $(Negative)$

Tabla 2: Classified data through $f(\mathbf{x})$.

Feature vector	*Label with* $f(\mathbf{x})$	*Element of the confusion matrix*
$(-2,1,2)$	1 $(Positive)$	TP
$(2,1,2)$	-1 $(Negative)$	TN
$(3,1,0)$	1 $(Positive)$	TP
$(1,3,1)$	1 $(Positive)$	TP
$(0,2,1)$	1 $(Positive)$	FP
$(3,-1,2)$	-1 $(Negative)$	FN
$(2,1,3)$	-1 $(Negative)$	TN

Table 2, presents the classifications obtained through the decision function $f(\mathbf{x})$, along with their respective nomenclature as elements for a confusion matrix (through direct comparison with the Table 1).

Finally, based on the Table 2, we can measure the Accuracy E for the decision function $f(\mathbf{x})$ considered in our example:

$$E = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} = \frac{3+2}{3+2+1+1} = 0,71428571.$$

The accuracy of our decision function is reasonably high $(Aprox. 71,43\,\%)$, since it is above 50 %, which would be the expected accuracy for a model making random predictions in a binary classification.

Remark. An accuracy of 0 means that the model did not correctly predict any samples from either the positive or negative class, indicating that the model is extremely deficient in the classification task. On the other hand, an accuracy close to 1 indicates that the model is predicting correctly almost all samples from both the positive and negative classes, achieving highly accurate classification. It's important to note that accuracy alone can be misleading, especially if the classes are imbalanced, so it is crucial to complement its evaluation with other performance metrics.

5. Conclusions

In this paper, the technical details behind the construction of a decision function using SVM and the use of a quadratic kernel for binary classification problems have been presented. A detailed example was provided, illustrating how SVM can be used to classify vectors into two distinct categories $\{1, -1\}$. Through this example, it was observed that SVM is an efficient and accurate algorithm for binary classification, and its use within AI has multiple applications. Furthermore, it was demonstrated how SVM can be used to avoid problems such as overfitting of training data. In particular, the specific calculations were highlighted where computers can be an ideal support, leading to reduced times and improved accuracy in estimations

Finally, it is important to highlight that the concepts presented in this article can be of great use for students in computer science, engineering, mathematics, and anyone interested in learning about artificial intelligence and binary classification. Además, se espera que este artículo sirva como una guía práctica para la construcción de funciones de decisión utilizando diferentes kernels (acorde al teorema de Mercer) para SVM, así como su integración en diferentes lenguajes de programación.

As for future research, various areas can be further explored regarding the implementation of SVM and its use in binary classification. One of them is the exploration of different kernels, beyond the quadratic kernel used in this example, for specific problems that require greater complexity. Similarly, different techniques for the optimal selection of SVM algorithm parameters can be studied to achieve better performance and avoid overfitting or underfitting. Furthermore, possibilities of implementing SVM in the analysis of large datasets can be explored, which might involve the use of dimensionality reduction techniques such as Principal Component Analysis (PCA), which has gained high importance in AI.

Acknowledgments

Y. Acevedo and O.M.L. Duque are grateful to Universidad EAFIT, Colombia, for the financial support in the project 'Study and applications of diffusion processes of importance in health and computing' with code 11740052022.

Statement of Interests

The authors declare that there are no potential financial or personal conflicts of interest that could have influenced the results or interpretations presented in this study.

Referencias

[1] Acosta, Adan and Aguilar-Esteva, Verónica Carreño, Ricardo and Patiño, Miguel and Patiño, Julian and Martínez, Miguel A, *Nuevas tecnologías como factor de cambio ante los retos de la inteligencia artificial y la sociedad del conocimiento*, Rev. Espacios. vol 41 (05). 2020.

[2] Cano Lengua, Miguel Ángel *Un algoritmo multiplicador proximal para clasificación binaria en máquinas de vectores soporte*, Universidad Nacional Mayor de San Marcos. 2023.

[3] Meléndez Lorenzo, Adrián. *Estudio, desarrollo y evaluación de técnicas de aprendizaje automático en tareas de clasificación y/o predicción: detección de exoplanetas*, vol 1 (1). 2023.

[4] Fandiño Orjuela. Juan Camilo and others. *Desarrollo de una aplicación web para la clasificación de residuos a través de un modelo de machine learning*, thesis, Ingeniería de Sistemas, 2023.

[5] Pardo Bernardi, Lucas *Estudio, desarrollo y evaluación de técnicas de aprendizaje automático para la identificación de notas, instrumentos y/o compositores en archivos de música*, 2022.

[6] Van Vaerenbergh, S and Santamaría, I. *Métodos kernel para clasificación*, GTAS, Universidad de Cantabria, 2018.

[7] Castro, Lorena and Silva, Guillermo. *Hígado graso no alcohólico*, Revista Médica Clínica Las Condes. Elsevier. vol 26 (5), pag. 600-612, 2015.

[8] Felipe Bravo. *Clasificación Support Vector Machines*, https://felipebravom.com/teaching/svm.pdf. April. (Accessed on 04/11/2023) , 2023.

[9] Lyu, Siwei. *Mercer kernels for object recognition with local features*, in IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). vol 2, p. 233-299, 2005.

[10] Ossa Sánchez, Jorge Eduardo. *Elementos del álgebra lineal en el aprendizaje de máquina*, Pereira: Universidad Tecnológica de Pereira, 2016.

[11] Saxena, Pranshu Priya, Kanu Goel, Sachin Aggarwal, Puneet Kumar Sinha, Amit and Jain, Parita. *Rice Varieties Classification Using Machine Learning Algorithms*, p. 3762-3772, 2022.

[12] Drineas, Petros and Mahoney, Michael W *Approximating a gram matrix for improved kernel-based learning*, in Learning Theory: 18th Annual Conference on Learning Theory, Bertinoro, Italy, June 27-30, Springer p. 323-337. COLT 2005.

[13] Suárez, Enrique. *Tutorial sobre máquinas de vectores soporte (sVM)*, V. 1, p. 1-12. 2016.

[14] Andreas Christmann, Ingo Steinwart *Support Vector Machines*, Springer New York, p. 285-329. 2008.

[15] Bishop, Christopher M and Nasrabadi, Nasser M. *Pattern recognition and machine learning*, Springer, vol. 4 (4), 1 Ed. p. 328. 2006.

[16] Valenzuela González, Gema. *Aprendizaje Supervisado: Métodos, Propiedades y Aplicaciones*, Trabajo de revisión bibliográfica, Universidad de Málaga, p. 1- 63. 2022.

[17] Gónzalez Flores, P. G. *Análisis comparativo de algoritmos de clasificación para diagnosticar tipos de leucemia infantil*, Universidad Señor de Sipán, p. 1- 45. 2021.

[18] Mimura, M. Impact of benign sample size on binary classification accuracy, *Expert Systems with Applications*, V. 211, p. 118630. 2023.

About the authors:

Name: Yeisson Alexis Acevedo Agudelo
Email: yaceved2@eafit.edu.co
Institution: Universidad EAFIT
Orcid: orcid.org/0000-0002-1640-9084

Name: Oscar Mario Londoño Duque
Email: olondon2@eafit.edu.co
Institution: Universidad EAFIT
Orcid: orcid.org/0000-0002-5666-8224