Migration Letters

Volume: 21, No: S4 (2024), pp. 901-913

ISSN: 1741-8984 (Print) ISSN: 1741-8992 (Online)

www.migrationletters.com

Classification And Analysis Of Clustered Non-Linear Separable Data Set Using Support Vector Machines

Pavithra C[1], Saradha M[2]

Abstract

The Support Vector Machines is the pre-eminent methodology in supervised machine learning, is adeptly applied to classification tasks and extends its utility to regression challenges. SVMs endeavour to discern the optimal hyperplane for dichotomizing data into distinct classes. When confronted with clustered non-linearly separable data, the Nonlinear mapping function approach emerges as a strategic solution to enhance computational efficiency. By employing mapping functions, data is mapped into a feature space, thus revealing a discernible linear decision boundary for the categorization of non-linear data. In our research, the objective is to transmute clustered non-linearly separable data into a linearly separable format through developing the nonlinear mapping functions. We have illustrated an example in classifying the clustered nonlinear synthetic data which was generated using Generative Adversarial Network (GAN), enabling nonlinear models to adeptly represent the vector similarities within the feature space. Through the identification of nonlinear mapping functions denoted as Φ, the data undergoes transformation into a novel feature space where the discernment of the hyperplane separation becomes apparent. This nuanced approach not only provides a deeper understanding of the internal mechanics of these models but also facilitates the assessment of the pertinence of feature combinations.

Keywords: *Kernel trick, Feature Space, SVM, Nonlinear Separable data, Nonlinear mapping functions.*[1]

Introduction

SVM has emerged as the preferred tool for a multitude of machine learning researchers. Renowned for its exceptional classification accuracy, SVM consistently surpasses alternative classification techniques in various applications. In text categorization, SVM excels at discerning patterns within vast amounts of textual data. Its adaptability also extends to object detection, microarray gene expression data analysis, and diverse data classification tasks. One of the defining attributes of SVM is its knack for outperforming other supervised learning techniques. Research consistently highlights the superiority of SVM in terms of classification accuracy, making it a reliable choice for applications where precision and reliability are paramount. The capacity of SVM to navigate complex, high-

[1]Department of Mathematics, REVA University, Bangalore, India.
[1] Department of Mathematics, SRM IST, Ramapuram Campus, Chennai, India.
[2]Department of Mathematics, REVA University, Bangalore, India.

dimensional feature spaces, especially with the aid of the kernel trick, contributes to its success in scenarios where traditional approaches may fall short. Its consistent outperformance in comparison to other supervised learning techniques underscores its significance and positions it as a cornerstone in advancing the capabilities of machine learning systems. As the digital landscape continues to evolve, SVM's versatility and accuracy make it an indispensable asset for researchers and practitioners alike. [2]. However, the settings of the cost parameter and kernel parameters, particularly for specific datasets, can have a significant impact on the performance of SVM. So, the user must perform extensive cross validation, also known as model selection, to determine the best parameter value. Iterative testing of various SVM algorithm settings, such as the number of training samples, the Gaussian kernel, the corresponding weights for slack variables to manage the non-uniform distributed of labelled data, and the choice of kernel functions, are required to select a model. The outcomes could change as a result of certain parameters. One needs to have a fundamental understanding of support vector machines before diving into the concept, theories, and methodologies of SVM [1]. A conventional two-classification model called the Support Vector Machine (SVM) finds an appropriate hyperplane to partition the acquired data samples. To solve a given quadratic programming problem, the division's goal is to maximize the margin (including hard and soft margins). Based on the amount of the data, the suitable solution space is chosen, and the dual space solution is converted into the original space's classification surface before being split into three groups for calculation. An essential component of the Support Vector Machine application's working mechanism is its feature structure [3][4]. The theory behind the Support Vector Machine (SVM) algorithm is solid, and it has good generalization capabilities. It has drawn a lot of interest from academics both locally and abroad, resulting in ongoing research that has led to advancements. By the use of risk reduction principles, SVM, which is founded on the idea of small sample machine learning, successfully addresses issues including limited sample sizes, nonlinearity, and high dimensionality while preserving a minimal requirement for prior knowledge [5][10]. For networks with intrusion detection systems, the classification precision is ideal. It can be challenging to choose the right kernel function in real-world projects to make the training set linearly separable in the feature, despite the kernel function presupposes that the observations that must be trained match the linearly separable criteria in the feature space [7]. It is essential to use a nonlinear model for accurate classification of nonlinear problems because the linear separable Support Vector Machine could indeed handle them [6][12]. A symmetric function connected to a semi-positive definite kernel matrix is referred to as a "kernel function." Because when feature space is elevated and linearly separable, support vector machines operate at their best, however the feature space is heavily influenced by the kernel function selected. However, choosing the appropriate kernel function is crucial because it just specifies the feature space implicitly and it is uncertain how it maps the sample. The sample might be mapped to the incorrect feature space if the improper kernel function is selected, which would result in subpar performance and prevent the desired result from being realized [7]. A classification task is one that involves dividing two groups of things based on the traits they share. This problem can be completed using a variety of techniques, including Support Vector Machines (SVMs) [11]. A novel instance-based big margin classification technique called SVMs uses a collection of predefined functions to infer an implicit decision boundary [8]. There is a set of hyperplanes for Semi-Consistent Hyperplanes (SHs) that correctly classifies all of the data objects. With regard to the data, these hyperplanes are referred to as semi-consistent hyperplanes. The separable scenario,

in which the classes are precisely divided by a hyperplane, serves as the starting point for the consideration of SHs. Finally, utilizing kernels and slack variables in a manner akin to the Support Vector Machine (SVM) technique, the non-separable issue is addressed. [6][8].

Literature Review

Piccialli, V et al., used the optimization of SVM performance transcends the mere application of a powerful algorithm, it requires a thoughtful and informed exploration of its parameter space. Through an iterative and informed process of model selection, where users can unlock the true potential of Support Vector Machines, adapting them to the intricacies of specific datasets and harnessing their capabilities for accurate and robust predictions in diverse scenarios [1].

Borah, P et al., proposed the combination of kernel trick, coupled with the exceptional capabilities of the Support Vector Machine, has positioned SVM as a standout tool in the machine learning landscape. Its ability to consistently outperform other supervised learning techniques across diverse applications underscores its significance and cements its reputation as a go-to solution for discerning patterns and making informed classifications in the ever expanding world of data analysis [2].

Gupta D and Borah P., proposed pivotal aspect of the SVM's functioning which is the selection of a suitable solution space based on the characteristics of the data. The dual space solution, derived from the quadratic programming problem, undergoes a transformation process. It is converted from the dual space into the original space, ultimately shaping the classification surface that segregates the data into its respective classes. The interplay between feature structures, solution spaces, and the classification surface forms the intricate separation that allows SVM to navigate and make sense of diverse datasets, making it a formidable tool in the arsenal of machine learning practitioners [3].

Gangopadhyay A et al., commenced with Support Vector Machine which was particularly renowned for its ability to establish a decision boundary in the data space through the identification of an optimal hyperplane. This hyperplane, a key element of the SVM's two-classification model, serves as the linchpin for partitioning acquired data samples into distinct classes. The SVM's objective is not merely classification but the identification of an appropriate hyperplane that maximizes the margin a critical concept encompassing both hard and soft margins [4].

Nalepa, J., & Kawulok, M., used one of the distinctive features that contribute to SVM's enduring appeal is its reliance on risk reduction principles. SVM operates on the fundamental concept of small sample machine learning, allowing it to navigate and extract meaningful insights from datasets with limited sample sizes. Beyond sample size constraints, SVM excels in addressing nonlinearity within data. Through the clever use of kernel functions, SVM can implicitly map data into higher-dimensional spaces, unravelling intricate patterns that may not be discernible in the original feature space. This nonlinear adaptability broadens the applicability of SVM to a diverse range of problems where complex relationships exist [5].

Lou, Y et al., Stated that while the linearly separable Support Vector Machine has its merits, acknowledging its limitations in handling nonlinear problems is essential, particularly in the domain of network intrusion detection. The selection of an appropriate kernel function becomes a strategic decision, influencing the algorithm's ability to accurately classify complex, real-world patterns. By embracing the nonlinear capabilities of SVM,

practitioners can enhance the precision of intrusion detection systems, ensuring a more robust defence against the evolving landscape of network threats [6].

Xingning, L et al., highlighted Support Vector Machines, which represents a novel and powerful approach to classification tasks. One distinctive feature that sets SVMs apart is their utilization of a novel instance-based big margin classification technique. Unlike some traditional methods that rely on proximity or probability estimates, SVMs introduce the concept of a margin which gives a clear separation between different classes. These functions facilitate the transformation of data into higher-dimensional spaces, revealing hidden patterns and allowing for the establishment of decision boundaries that may be intricate or nonlinear [7].

Preliminaries

Kernel trick in Support Vector Machine

The kernel trick is a transformative concept in the realm of Support Vector Machines (SVMs), amplifying their effectiveness in handling complex, nonlinear patterns within data. At its core, SVMs are known for constructing optimal hyperplanes in a high-dimensional feature space, aiming to separate different classes. However, many real-world datasets exhibit intricate relationships that are not linearly separable in the original feature space. This is where the kernel trick comes into play. The kernel trick is a technique that enables SVMs to implicitly map input data into higher-dimensional spaces, revealing nonlinear patterns without the need to explicitly compute the transformation. Instead of dealing with the computational burden of operating directly in the higher-dimensional space, the kernel function allows SVMs to compute the dot product between data points in this space efficiently. Commonly used kernel functions include the linear kernel for linearly separable data, the polynomial kernel for capturing polynomial relationships, and the radial basis function kernel for handling intricate, nonlinear patterns. These kernels effectively introduce a measure of similarity or distance between data points in the transformed space, enabling SVMs to discern complex structures and make accurate classifications. The kernel trick not only enhances the adaptability of SVMs to diverse datasets but also contributes to their ability to generalize well. This technique has found applications in various fields, including image recognition, bioinformatics, and natural language processing. The function, designated as K(x1, x2), has a few properties and is an inner product of x1 and x2 in a higher-dimensional space. To handle inner products between observations in the defined spaces of the kernel function, these functions are utilized in the kernel trick [2][7][8].

Finding the plane that can separate, classify, or split the data with the most margin is the fundamental tenet of SVM and Kernel Trick. Street width is another name for the margin. The separation measure from a point (x_0, y_0) to a line $Ax + By + c = 0$ is

$$\frac{|Ax_0 + By_0 + c|}{\sqrt{A^2 + B^2}} = 0 \tag{1}$$

Thus, the distance between H_0 and H_1 is then $\frac{|W.x + b|}{\|W\|} = \frac{1}{\|W\|}$ (2)

The total distance between H_1 and H_2 is thus $\frac{2}{\|W\|}$ (3)

To achieve the better and maximum distance of ||w|| with no data points existing in between H_1 and H_2 $x_i.W+b \geq +1$ when $y_i = +1$ and $x_i.W+b \leq -1$ when $y_i = -1$ the combined equation can be written as, $y_i(x_i.W) \geq 1$.

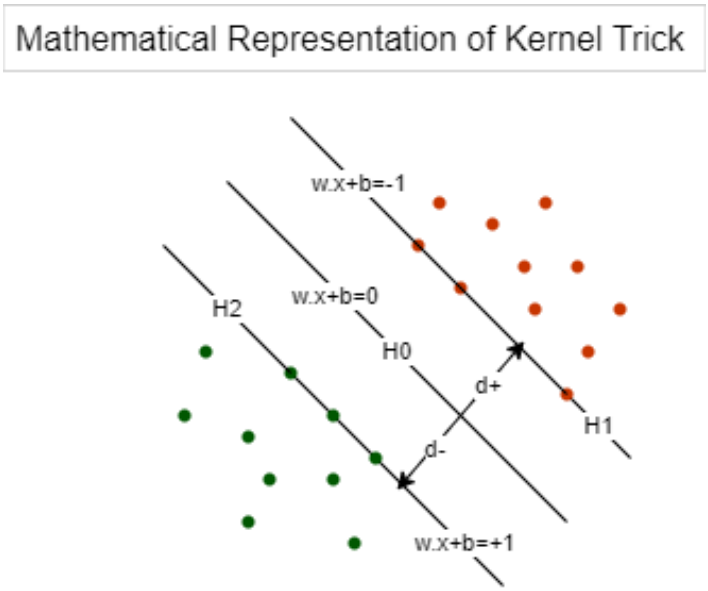

Fig 1: Kernel Trick

By maximizing the distance between support vectors, which are the nearest data points to the hyperplane that influence its position and orientation, SVM aims to maximize the margin or street width. This is accomplished by a constrained optimization procedure, in which the input values are chosen from among those that are permitted in order to determine the function's value. The requirement is that the support vectors must not be on, between, or near the street. Thus, the SVM optimization issue is an example of a restricted optimization problem [12].

Kernel trick for Separable data and Non-linear Separable data

A unique mathematical characteristic of the kernel function functions as a revised dot product. A linear boundary can be created in the higher dimensional space to demarcate the classes. The kernel trick is a machine learning method for dealing with non-linearly separable data. The transformed data can then be applied to the linear classifier to separate it, and the outcome of the discriminator in the high-dimensional space can be transmogrified back into the space to obtain the final result. It facilitates the application of a linear approach, such as the Support Vector Machine (SVM) algorithm, are applied to non-linear data by converting them to the new higher dimensional space where the data becomes linear data set. We have seen that raising the dimensionality of the data can help with precise categorization predictions. It is necessary to carry out operations with the modified feature vectors in high dimensions in order to train a support vector classifier and improve its performance. The compute cost of executing transformations requiring polynomial combinations of these features, however, might become excessively large and impractical

when working with real-world data that has a lot of features by transforming the information into a more dimensional space that is suitable for classification and then applying the standard algorithm to this transformed data, the kernel method, in principle, enables a linear algorithm to handle non-linear data [10].

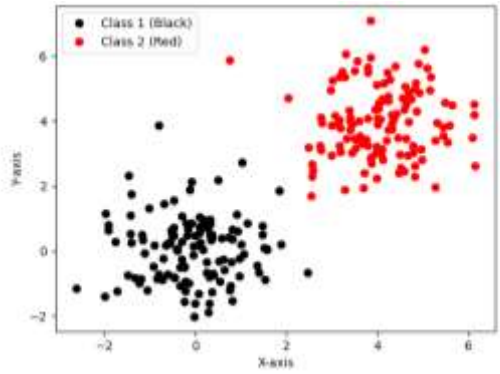

Fig 2: Linear Separable data

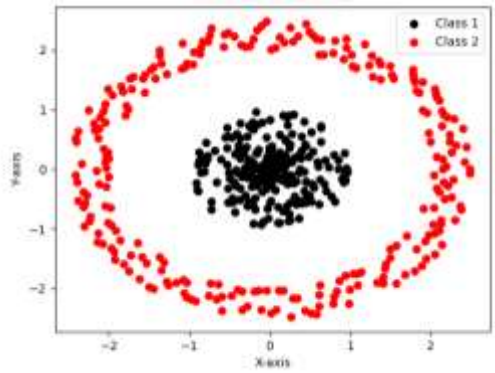

Fig 3: Nonlinear Separable data

When training a support vector classifier, the problem of high computing costs can be avoided using the kernel method. Kernel approaches merely describe the data through pairwise similarity comparisons between the observations, rather than explicitly performing modifications to the original data observations and representing the data in a higher dimensional space. Each item (i,j) in a kernel matrix of size n x n containing these comparisons is defined by the kernel function K. (xi,xj). By avoiding the requirement for explicit feature transformations, this method lessens the computational load. By translating non-linearly separable data into a higher dimensional space where it is linearly separable, the kernel trick is a machine learning technique for dealing with such data. The data is mapped into a new space using a kernel function so that a linear technique can be used to separate it. The kernel function separates binary classification issues by acting as a modified dot product and having a unique mathematical property. The results are then translated back into the original space to obtain the final result after the data has been separated in the high-dimensional space [3][7].

Proposed System

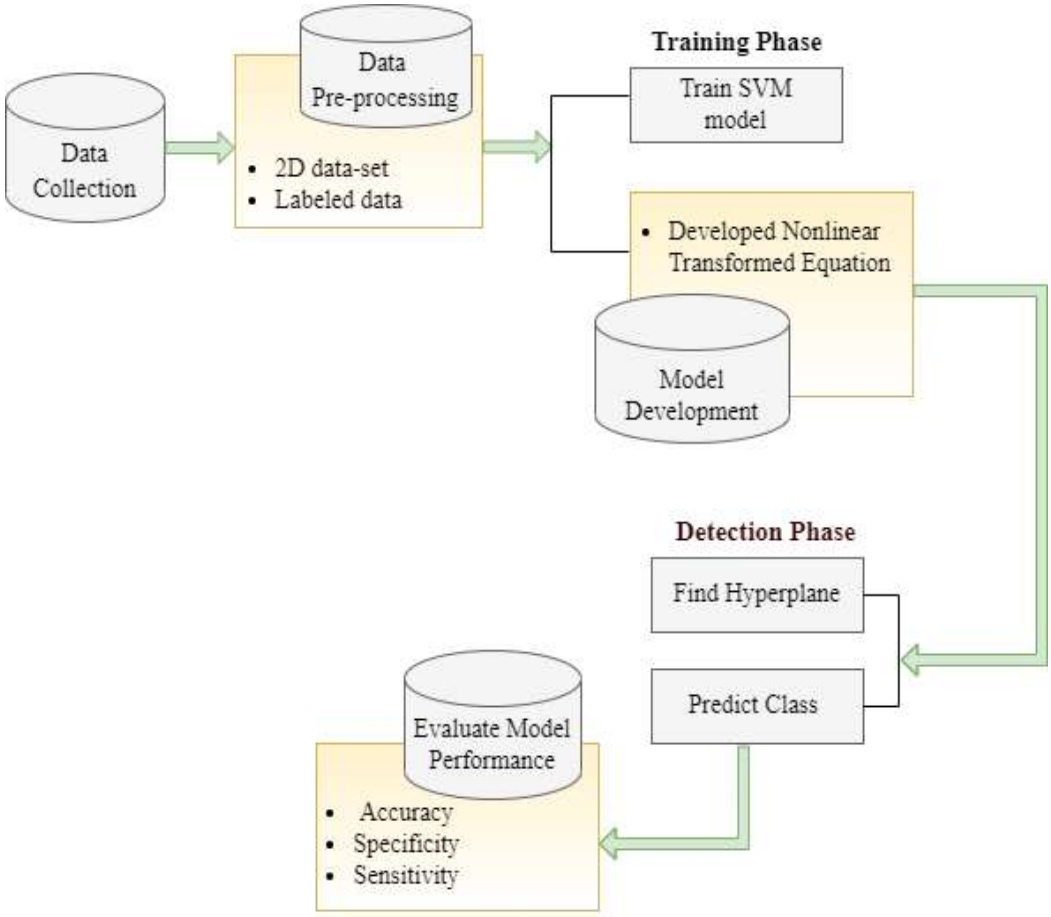

Fig 4: Structure of Proposed System

Fig 4 represents the structure of our proposed system. Collecting and pre-processing the data is the first step, here we collect data and pre-process it by cleaning and transforming it into a format suitable for SVM. The process begins with the representation of data points in a 2D space, utilizing the x and y axes to denote the features of the data. To navigate nonlinear patterns within the data, a nonlinear mapping function is selected based on the inherent characteristics of the dataset. This chosen transformation function facilitates the conversion of the data into a feature space where classes can be effectively separated by a hyperplane. The SVM model is then trained on this pre-processed data, aiming to find a hyperplane that maximizes the margin between the two classes within the feature space. This optimized hyperplane becomes instrumental in classifying new data points based on their features. In essence, the SVM's primary objective is to discover a decision boundary that optimally distinguishes positive and negative data points while maximizing the margin between these classes. The performance of the SVM model is subsequently assessed using weighted vectors, providing a comprehensive evaluation of its classification capabilities. Positive data points are labelled as +1 and negative data points are labelled as -1.

The hyperplane is represented as, $w_0 + w_1 \times x_1 + w_2 \times x_2 = 0$ (4)

Where w_0, w_1, and w_2 are the parameters of the hyperplane, x_1 and x_2 are the features of a data point, and with the bias term. The sign of the equation indicates the class of the data point. If the equation is positive, the data point is classified as positive (+1), and if the equation is negative, the data point is classified as negative (-1).

Methodology

We intend to categorise the non-linear separable data by transforming data sets to new feature space and applying SVM methods. Here, we handle a sizable data set using the nonlinear transformation method, which enables nonlinear models to express the similarity of vectors in the feature space. Non-linear separable data are a group of data that cannot be separated linearly and are not organised in a sequential manner. If all of the data are correctly classified throughout the separation process, we refer to it as a hard support vector machine otherwise, it is referred to as a soft support vector machine and is also known as partial nonlinear separable data. First, by identifying the most appropriate function to categorise the non-linearly separable data, we transform them to linearly separable data using the fundamental expansion approach. We start by creating the hyperplane for 2-dimensional space, the hyperplane for the 2D space is a 1-dimensioanl straight line. We execute the data transformation of the 2D space using transformation equation. Utilizing an appropriate function becomes essential to transition the data points into an alternative feature space. Our primary goal is to make a hyperplane that accurately separates the two classes, a task that proves challenging within the confines of the input space. In instances where a straightforward hyperplane is not discernible, the application of a nonlinear separable Support Vector Machine becomes imperative to facilitate the transformation of data from one feature space to another. Consider a scenario where the input space lacks a readily apparent hyperplane for accurate class distinction. In such non-trivial cases, envision a set of R2 data points featuring both positive and negative labels. To effectively handle this complexity and navigate nonlinear patterns inherent in the data, the deployment of a nonlinear SVM becomes necessary. This nonlinear SVM incorporates a mapping function that transforms input space into a feature space, providing the means to accurately represent and classify nonlinear data points. We can observe how modifies our data before to doing the dot products in order to construct the hyperplane. As a result, the data can be rewritten in feature space. The support vectors can be found as a further step. Consequently, we include bias input vectors that have been increased by 1 and discover values for the α_i.

Implementation and Analysis

An SVM model can be trained on a dataset of labelled clustered nonlinear data set samples using a binary classification method, where each sample is categorized as either dangerous or benign. Based on variables retrieved from the training dataset, such as file size, frequency of file access, and existence of particular code snippets, the SVM model learns to categories fresh samples. Different samples of data that are divided into the two classes "Infected" and "Non-infected" are represented by two-class data points. The attributes of each data point in the dataset, which each represent a file or an action, are derived from different sources, such system logs, network traffic, or file properties. The SVM model is trained to distinguish between Nonlinear data samples by examining these attributes, and it can then categories new data points appropriately. By recognizing and preventing malicious files or activities, this can assist in detecting and preventing anonymous assaults. Here we have illustrated the clustered non-linearly separable data related to infected and non-infected classification. We utilized Generative Adversarial Networks (GAN) to generate data, demonstrating the application of concepts and an evolved equation. This illustrates that the nonlinear equation efficiently transforms data into a new feature space without altering the dimension of the original dataset. This reduction in complexity facilitates the transformation of classification, making it more straightforward to identify a hyperplane for distinct classification in clustered nonlinear datasets.

The data consists of 300 observations of the number of hours of use and the number of crashes of nonlinear separable data. The class label indicates whether the data is considered "harmful/Infected" (class 1) or "non-harmful/Non-Infected" (class 0). However, when we plot the data as a scatter plot, we see that it's not possible to draw a straight line that separates the two classes. This means that the data is not linearly separable and a more sophisticated algorithm, such as a non-linear classifier, would be needed to separate the two classes effectively.

Algorithm 1: Input Space

Input: Input space (Class1 and Class2).
Output: Plot the points as a scatter plot.

Step 1: Two arrays "x" and "y" that represent two sets of points in the 2D space.
Step 2: Pass "x" as x-coordinate, "y" as y-coordinate, and "green" as color.
Step 3: Set x-axis label as "X" using "plt.xlabel" method..
Step 4: Set y-axis label as "Y" using "plt.ylabel" method
Step 5: Set the title of the plot as "Clustered Nonlinear data points" using "plt.title.
Step 6: Show the plot using plt.show().

The algorithm begins by taking two arrays, "x" and "y," representing coordinates in a 2D space, as input, where each array corresponds to a different class (Class1 and Class2). The primary objective is to create a scatter where points from Class1. The algorithm accomplishes this by passing the "x" array as x-coordinates, "y" array as y-coordinates, in the plt.scatter method. Furthermore, axis labels are set using corresponding codes to denote the x and y dimensions, respectively. The plot is then given a title, "Clustered Nonlinear data points," using plt.title for clarity. The final step involves displaying the plot with plt.show(). This algorithm proves useful for visually inspecting and comprehending the distribution and clustering patterns of nonlinear data points in a 2D space, providing valuable insights into the characteristics of the underlying data.

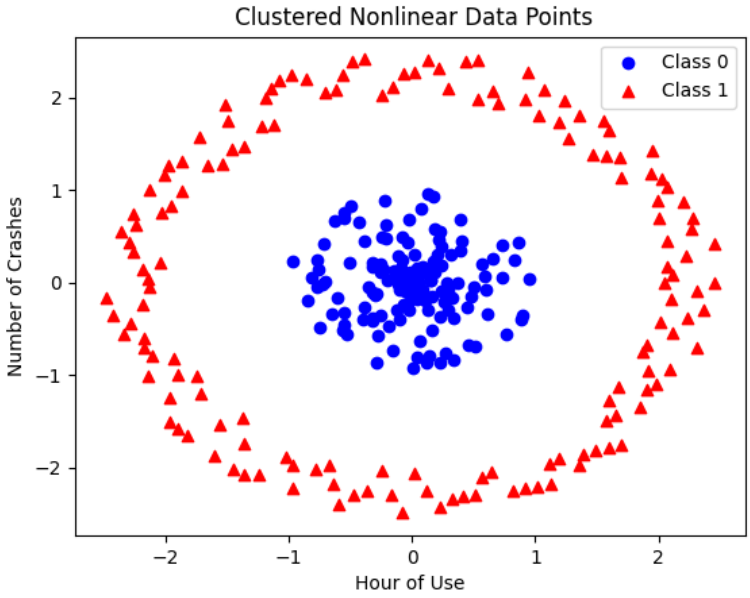

Fig 5: Input Space

Fig 5 represents the set of nonlinear data points, for the above set of nonlinear data sets, we form the desirable function to convert the nonlinear separable data points to linearly separable data points. Let us consider the mapping function,

$$\phi(x) = \begin{Bmatrix} n_R - x + |x - y| \\ n_R - y + |x - y| \\ \begin{pmatrix} x \\ y \end{pmatrix} \end{Bmatrix} \begin{matrix} if\ \sqrt{(x^2 + y^2)} \geq n \\ Otherwise \end{matrix} \quad (5)$$

Where, n is the highest values of corresponding class.

n_R is the randomly chosen number, where $n_R > n$.

By using the above transformation equation, we convert the class 2 data set to another feature space and this becomes the linearly separable data sets.

Algorithm 2: Transformed Feature Space

Input: Two sets of points (Class1 and Class2).
Output: Plot the points as a Transformed Feature Space.

Step 1: Import the required libraries, matplotlib.pyplot and numpy
Step 2: Define two sets of points ‘points1’ and ‘points2’ as a list of x and y coordinates.
Step 3: Store ‘x’ and ‘y’ as separate lists x1 and y1 for class 1, and x2 and y2 for class2.
Step 4: Plot the scatter plot of ‘class1’ and ‘class2’ (color 'Blue' and color 'Red').
Step 5: Add labels to x and y axis using plt.xlabel and plt.ylabel.
Step 6: Add a title to the plot using plt.title.
Step 7: Show the plot using plt.show.

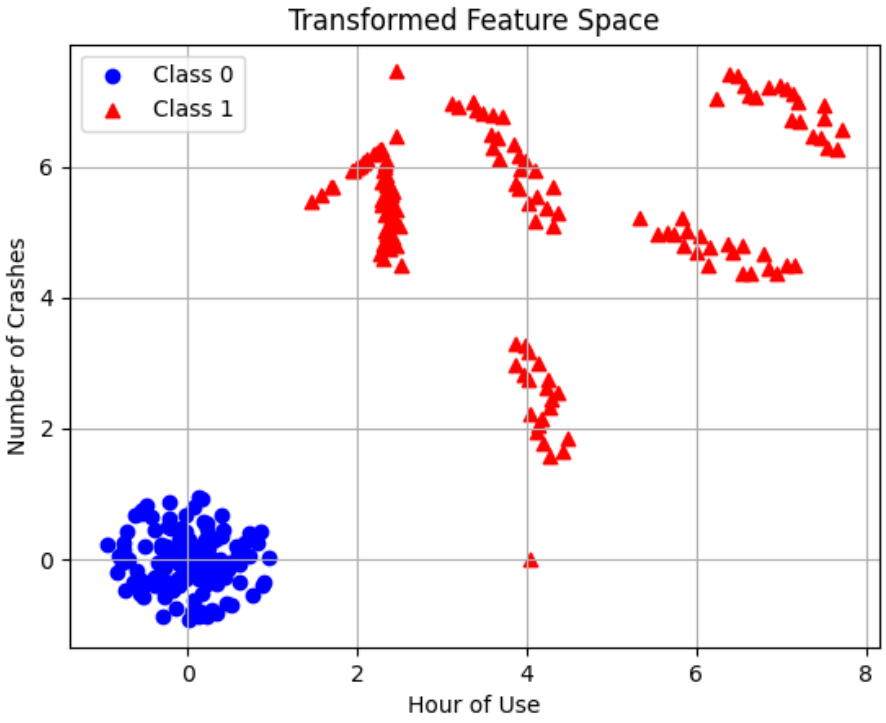

Fig 6: Transformed Feature Space

Fig 6 represents the feature space with linearly separable data. The following are the equations obtained using the transformed equation to compute the three parameters α_1, α_2 and α_3.

$$\alpha_1\Phi_1(s_1).\ \Phi_1(s_1) + \alpha_2\Phi_1(s_2).\ \Phi_1(s_1) + \ldots\ldots+\alpha_m\Phi_1(s_2).\ \Phi_1(s_1) = -1 \qquad \ldots(6)$$

$$\alpha_1\Phi_1(s_1).\ \Phi_1(s_2) + \alpha_2\Phi_1(s_2).\ \Phi_1(s_2) + \ldots\ldots+\alpha_m\Phi_1(s_2).\ \Phi_1(s_1) = +1 \qquad \ldots(7)$$

We choose the support vector from two classes, $s_1 = \begin{bmatrix} s_{11} \\ s_{12} \\ \vdots \\ s_{1n} \end{bmatrix}$ $s_2 = \begin{bmatrix} s_{21} \\ s_{22} \\ \vdots \\ s_{2n} \end{bmatrix}$ and $s_m = \begin{bmatrix} s_{m1} \\ s_{m2} \\ \vdots \\ s_{mn} \end{bmatrix}$.

Each vector is augmented with the value 1 as a bias input. Therefore, $\dot{s}_1 = \begin{bmatrix} s_{11} \\ s_{12} \\ \vdots \\ s_{1n} \\ 1 \end{bmatrix}$ $\dot{s}_2 = \begin{bmatrix} s_{21} \\ s_{22} \\ \vdots \\ s_{2n} \\ 1 \end{bmatrix}$

and $\dot{s}_m = \begin{bmatrix} s_{m1} \\ s_{m2} \\ \vdots \\ s_{mn} \\ 1 \end{bmatrix}$

$$\alpha_1\,\dot{s}_1.\dot{s}_1 + \alpha_2\,\dot{s}_2.\dot{s}_1 + \ldots\ldots.. + \alpha_m\,\dot{s}_m.\dot{s}_1 = -1 \qquad (8)$$

$$\alpha_1\,\dot{s}_1.\,\dot{s}_2 + \alpha_2\,\dot{s}_2.\dot{s}_2 + \ldots\ldots.. + \alpha_m\,\dot{s}_m.\dot{s}_2 = +1 \qquad (9)$$

$$\alpha_1 \begin{bmatrix} s_{11} \\ s_{12} \\ \vdots \\ s_{1n} \\ 1 \end{bmatrix}.\begin{bmatrix} s_{11} \\ s_{12} \\ \vdots \\ s_{1n} \\ 1 \end{bmatrix} + \alpha_2 \begin{bmatrix} s_{21} \\ s_{22} \\ \vdots \\ s_{2n} \\ 1 \end{bmatrix}.\begin{bmatrix} s_{11} \\ s_{12} \\ \vdots \\ s_{1n} \\ 1 \end{bmatrix} + \ldots\ldots.. + \alpha_m \begin{bmatrix} s_{m1} \\ s_{m2} \\ \vdots \\ s_{mn} \\ 1 \end{bmatrix}.\begin{bmatrix} s_{11} \\ s_{12} \\ \vdots \\ s_{1n} \\ 1 \end{bmatrix} = -1 \qquad (10)$$

$$\alpha_1 \begin{bmatrix} s_{11} \\ s_{12} \\ \vdots \\ s_{1n} \\ 1 \end{bmatrix}.\begin{bmatrix} s_{21} \\ s_{22} \\ \vdots \\ s_{2n} \\ 1 \end{bmatrix} + \alpha_2 \begin{bmatrix} s_{21} \\ s_{22} \\ \vdots \\ s_{2n} \\ 1 \end{bmatrix}.\begin{bmatrix} s_{21} \\ s_{22} \\ \vdots \\ s_{2n} \\ 1 \end{bmatrix} + \ldots\ldots.. + \alpha_m \begin{bmatrix} s_{m1} \\ s_{m2} \\ \vdots \\ s_{mn} \\ 1 \end{bmatrix}.\begin{bmatrix} s_{21} \\ s_{22} \\ \vdots \\ s_{2n} \\ 1 \end{bmatrix} = +1 \qquad (11)$$

By solving for the above equation, we get α_1, α_2 and α_m Subsequently, we compute discriminating decision boundary for the feature space. Where $\dot{w}$ is presented as the weighted vector.

$\dot{w} = X_i\ \alpha_i\ \dot{s}_i$

$\dot{w} = \alpha_1.\dot{s}_1 + \alpha_2.\dot{s}_2 + \ldots\ldots.. + \alpha_m.\dot{s}_m$

The equation of the hyperplane is given as, $\mathbf{y = \dot{w}\ x + b}$ with $\dot{w}$ and b. This gives the desired decision surface.

Algorithm 3: Scatter Plot with Hyperplane

Input: Create two sets of points (Class1 and Class2).
Output: Plot 2D scatter plot with Hyperplane.

Step 1: Import the required libraries, matplotlib.pyplot and numpy
Step 2: Extract the x and y coordinates for each set of points.
Step 3: Points1 are plotted as red circles and points2 are plotted as green circles.
Step 4: Define the hyperplane parameters (weights "w" and bias "b") for set of x values to create hyperplane.
Step 5: Calculate the corresponding y values for the hyperplane using the equation: y = -(w[0] * x_hyperplane + b) / w[1]
Step 6: Plot the hyperplane using the calculated x and y values.
Step 7: Label the x and y axis as "X" and "Y" respectively and add a title to the plot.
Step 8: Display the plot using plt.show().

The exploration of transformed nonlinear data with a decision boundary represents a pivotal aspect of contemporary research in machine learning and data analysis. This approach acknowledges the inherent complexities in real-world datasets, where relationships between features and outcomes are often intricate and nonlinear. By undertaking a nonlinear transformation of the feature space, we enhance our ability to perform and understand these intricate relationships.

By subjecting the feature space to nonlinear transformations, we empower our equation to capture and elucidate the nuanced patterns of data. This not only leads to more accurate predictions and classifications but also facilitates a richer comprehension of the underlying structures governing the observed phenomena. Consequently, the adoption of nonlinear transformations becomes indispensable to extract meaningful insights from dataset characterized by intricate and nonlinear relationships.

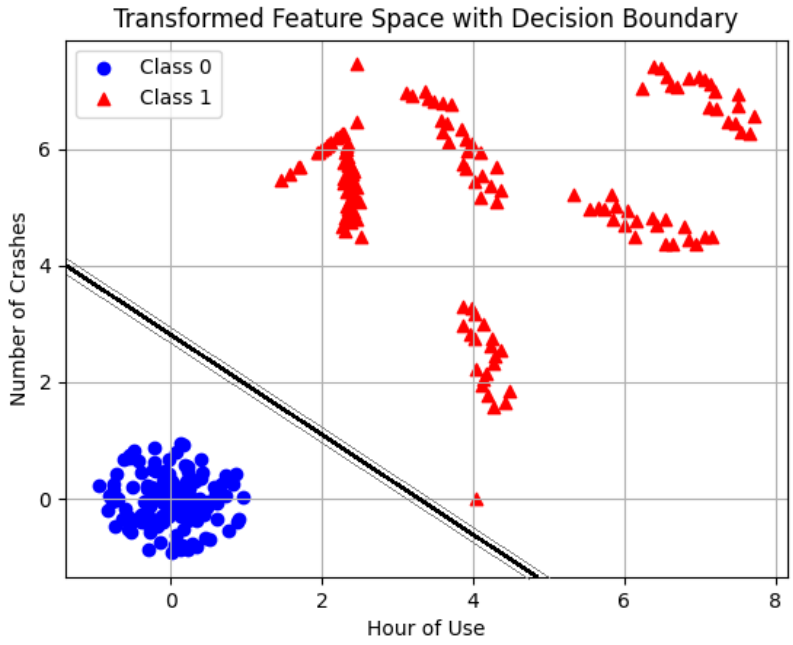

Fig 7: Transformed feature space with Decision Boundary

Conclusion

In this research we deploy Support Vector Machine for classification and regression. We create for the decision boundary that divides the features into different classes in the best way possible. Here we choose the support vectors, the points which are close to hyper planes. Finding the decision boundary for non-linear data categorization is made easier by the nonlinear transformation function. In this research paper, the identification of nonlinear mapping functions, denoted as Φ, plays a central role in transforming the data into a novel feature space, where the discernment of hyperplane separation becomes apparent. This nuanced approach not only deepens our understanding of the internal mechanics of these models but also facilitates a comprehensive assessment of the relevance of feature combinations. Our research contributes to advancing the application of SVMs in handling complex data patterns and underscores the significance of nonlinear mapping functions in achieving effective classification in the context of non-linearly separable data.

Acknowledgement: The author would like to thank REVA University for their encouragement and support in carrying out this research work.

References

[1] Piccialli, V., & Sciandrone, M. (2022). Nonlinear optimization and support vector machines. Ann Oper Res, 314, 15– 47https://doi.org/10.1007/s10479-022-04655-x

[2] Borah, P., & Gupta, D. (2020). Unconstrained convex minimization based implicit Lagrangian twin extreme learning machine for classification (ULTELMC). Applied Intelligence, 50(4), 1327– 1344.

[3] Borah, P., & Gupta, D. (2020). Functional iterative approaches for solving support vector classification problems based on generalized Huber loss. Neural Computing and Applications, 32(1), 1135–1139.

[4] Gangopadhyay, A., Chatterjee, O., & Chakrabartty, S. (2018). Extended polynomial growth transforms for design and training of generalized support vector machines. IEEE Transactions on Neural Networks & Learning Systems, 29(5), 1–14.

[5] Nalepa, J., & Kawulok, M. (2019). Selecting training sets for support vector machines: a review. Artificial Intelligence Review, 52(2), 857–900.

[6] Lou, Y., Liu, Y., Kaakinen, J. K., & Li, X. (2017). Using support vector machines to identify literacy skills: evidence from eye movements. Behavior Research Methods, 49(3), 887–895

[7] Li, Z. H. A. N. G., Xingning, L. U., & Conglin, L. U. (2017). National matriculation test prediction based on support vector machines. Journal of University of Science and Technology of China, 47(1), 1-9.

[8] Gu, W., Chen, W.-P., & Ko, C.-H. (2018). Two smooth support vector machines for -insensitive regression. Computational Optimization & Applications, 70(1), 1–29.

[9] Zhang, X., Li, Y., & Peng, X. (2016). Brain Wave Recognition of Word Imagination Based on Support Vector Machines. Chinese Journal of Aerospace Medicine, 14(3), 277-281.

[10] Veropoulos K., Cristianini N., and Campbell C., "The Application of Support Vector Machines to Medical Decision Support: A Case Study", ACAI99.

[11] Kernel Functions-Introduction to SVM Kernel & Examples. (2017, August 12).

[12] Chih-Wei Hsu, Chih-Chung Chang, and Chih- Jen Lin. “A Practical Guide to Support Vector Classification” 2007.