Available online at www.ijournalse.org

Emerging Science Journal

(ISSN: 2610-9182)

Vol. 5, No. 5, October, 2021

Support Directional Shifting Vector: A Direction Based Machine Learning Classifier

Md. Kowsher [1*], Imran Hossen [2], Anik Tahabilder [3], Nusrat Jahan Prottasha [4], Kaiser Habib [5], Zafril Rizal M. Azmi [6]

[1] Department of Computer Science, Stevens Institute of Technology, Hoboken, NJ 07030, USA

[2] Department of Information and Communication Engineering, University of Rajshahi, Rajshahi 6205, Bangladesh

[3] Department of Computer Science, Wayne State University, Detroit, MI 48202, USA

[4] Department of Computer Science and Engineering, Daffodil International University, Dhaka 1207, Bangladesh

[5] Department of RET, University of Dhaka, Dhaka 1206, Bangladesh

[6] Faculty of Computing, College of Computing and Applied Sciences, Universiti Malaysia Pahang, Malaysia

Abstract

Machine learning models have been very popular nowadays for providing rigorous solutions to complicated real-life problems. There are three main domains named supervised, unsupervised, and reinforcement. Supervised learning mainly deals with regression and classification. There exist several types of classification algorithms, and these are based on various bases. The classification performance varies based on the dataset velocity and the algorithm selection. In this article, we have focused on developing a model of angular nature that performs supervised classification. Here, we have used two shifting vectors named Support Direction Vector (SDV) and Support Origin Vector (SOV) to form a linear function. These vectors form a linear function to measure cosine-angle with both the target class data and the non-target class data. Considering target data points, the linear function takes such a position that minimizes its angle with target class data and maximizes its angle with non-target class data. The positional error of the linear function has been modelled as a loss function which is iteratively optimized using the gradient descent algorithm. In order to justify the acceptability of this method, we have implemented this model on three different standard datasets. The model showed comparable accuracy with the existing standard supervised classification algorithm.

Keywords:

Supervised Machine Learning;
Classification;
Cosine Similarity;
Directional Vectors;
Angle Measurement.

Article History:

Received:	01	July	2021
Revised:	11	September	2021
Accepted:	21	September	2021
Published:	01	October	2021

1- Introduction

We always perform classification in our day-to-day life, even sometimes in the subconscious mind. For example, we usually receive a lot of emails every day. They can be classified as spam or non-spam. This classification is sometimes a two-class classification, and some other times, it's a multiclass classification. As human beings, while doing classification, we can differentiate or categorize things based on very explicitly sensible properties. However, we face difficulties in differentiating minor differences between two or more samples. Besides, if there are many parameters to consider, the human brain can't process and make perfect classification quickly. So, there have been efforts by the researcher to develop a computerized classification algorithm. With the recent development of machine learning, classification accuracy and robustness have been increased a lot. But there is still some area that has not been

* **CONTACT**: ga.kowsher@gmail.com

DOI: http://dx.doi.org/10.28991/esj-2021-01306

covered comprehensively. For example, the traditional classification methods may not perform well in all types of data. So, we have proposed a direction-based classification algorithm that is suitable for diverse kinds of data.

Human beings usually classify objects based on their visual and sensing system. On the other hand, computers or intelligent systems learn by analyzing data and then build a model based on the data, which finally acts as a classifier. By learning from vast amounts of spam data and user input, intelligent spam classifier quarantines the spam email. Similarly, advertising systems learn how to fit suitable advertisements to the right customers. Computer vision-based classification model can recognize diverse types of handwriting [1]. Financial fraud detection systems protect banks from cyber hackers. Machine learning classification approaches are being used in almost every intelligent classification system. So, the researchers are trying to develop various classification models that suits diverse types of data. These models are being used in machine learning and fully logic modeling for solving pattern recognition and classification problems [2].

R.A. Fisher, more than 60 years ago, suggested the first pattern recognition algorithm [8]. He proved the optimal (Bayesian) Solution as a quadratic decision function. He recommended a linear decision function for classification when the two distributions are not normal. Frank Rosenblatt introduced perceptrons or neural networks in the early1960s [21]. This algorithm was implemented as a piecewise linear separating surface. Therefore, from the very beginning, algorithms for pattern recognition were associated with linear decision surfaces.

In 1986 J.R. Quinlan proposed a method named decision tree, which is popularly known as Iterative Dichotomiser (ID3) [20]. He showed a methodology, dealing with information that is noisy and/or incomplete. He also mentioned the shortcoming of the basic algorithm and the solutions. The ID3 is an iterative algorithm; a subset of data is selected, known as a window, then a decision tree is formed.

Random forests are a combination of tree predictors in which each tree depends on the values of a random vector independently sampled, having the same distribution [5]. Yoav Freund and Robert E. Schapire used a random set of characteristics to split each node and found error rates that are comparable to Adaboost but more resistant in terms of noise [9]. They also showed Internal predictions monitor error, strength, and correlation, and these are used to illustrate the reaction to increasing the number of splitting features used. To assess variable value, internal projections are also used. These principles are also true for regression.

In 1995, Corinna Cortes, Vladimir Vapnik, and Lorenza Saitta introduced a simple classification algorithm named Support Vector Network [6]. They developed it for two classification problems. They implement three ideas in the paper: the solution technique from optimal hyperplanes, the concept of convolution of the dot-product, and the notion of soft margins.

The k-nearest neighbor (kNN) is a simple and effective algorithm that performs classification based on the property matching with the neighbor [7]. It is ubiquitously being used for classification and pattern recognition. However, choosing an optimum value of k is the main problem. Huawen Liu, Shichao Zhang, Jianming Zhao, Xiangfu Zhao, and Yuchang Mo proposed Mutual Nearest Neighbors (MNN), rather than the kNN [12]. The advantage of mutual neighbors is that the MNN does not take the pseudo nearest neighbors into account during the prediction process.

None of the algorithms above is suitable for all of the data types and applications. The performance of classification depends on the types of the dataset and the algorithm selection. This article proposes an angular measurement-based classifier named Support Directional Shifting Vector (SDSV), and it deals with the classification problems of linearly separable data. First, the cosine similarity is measured with the help of a Support Directional Vector (SDV) and Support Origin Vector (SOV). In the beginning, these two values are initialized randomly, and it gets optimized iteratively. The positional error of these vectors is modelled as loss function, and then the loss function value is minimized by gradient descent optimization technique. Finally, we have used different datasets to verify the performance of the proposed model. The model showed descent accuracy, which is comparable with the performance of similar standard classification models.

There exist a lot of research that involves machine learning-based classification. Most of them are well suited for non-angular data. But whole reviewing the literature, it was noticed that those methods are not best suitable for angular data like temporal data that repeats after a specific time interval. This challenge inspired us to make a classification that considers angle values for performing the categorization. Again, cosine similarity is usually non-parametric learning, and it can't be used in parametric machine learning.

The main contribution of this paper is summarized below in point:

- This classification method is best suitable for linearly separable angular data;
- It has enabled cosine similarity as parametric learning;
- In this method, a classification decision is made based on the direction of the data;
- It is capable of linearly separate data's using shifting direction and shifting origin vectors;

- We have used the gradient descent optimization technique to change the position of the plotting vector;
- The performance of this algorithm is comparable with other standard methods.

Section II described the literature review, and in Section III, we have depicted the theoretical background of the proposed method. Section IV formulated the mathematical model of the proposed model. In section V, we have discussed the experiment and results. We have finished this article in section VI by discussing the conclusion and discussing future research directions.

2- Literature Review

Every machine learning algorithm has its own application. One algorithm may be best suit for one dataset but not at all suitable for other datasets. In real life, we deal with directional data such as temporal period (e.g., time of day, week, month, year, etc.), orientation, rotation. Traditional classifiers may not be able to correctly classify these types of data because these classifiers are unaware of the angular nature. We need special types of algorithms to handle directional data.

A few researchers, including Diogo Pernes, Kelwin Fernandes, and Jaime S Cardoso used a non-probabilistic model for directional data [19]. Diogo Pernes and his co-researchers have proposed several directional-aware support vector machines: updated decision function of the SVM by considering parametric periodic mapping of directional variables using cosine and triangle waves, extended the model with triangular waves to allow asymmetric circle margins, and kernelized the different version of the SVM. Yet, the additional parameters involved in asymmetric SVMs periodically affect the boundary of the decision.

We generate a lot of directional data in our everyday life. However, the traditional statistical distribution is not well suited for this type of data. Therefore, different distributions and statistics should be used to deal with this type of data, such as the von Mises univariate and the von Mises-Fisher multivariate distributions. Pedro L. López-Cruz, Concha Bielza, and Pedro Larrañaga have demonstrated a one-directional Naive Bayes classifier predictive variable [13]. This model needs either von Mises univariate or von Mises univariate for implementation. The von Mises–Fisher and Gaussian distributions are used to model the predictive variables. In the hybrid setting, they demonstrated that the complexity of the decision surfaces depends on the parameters of the Gaussian distribution. They evaluated the von Mises naive Bayes (vMNB) classifier over eight datasets and compared it against the corresponding NB classifiers that use Gaussian distributions or discretization for modeling angular variables. Among others, the Selective NB classifier was the best ranking algorithm. David M Blei, Andrew Y Ng, and Jordan presented a generative probabilistic model called latent Dirichlet allocation (LDA) [3]. They showed how to estimate empirical Bayes parameters using effective approximate inference techniques based on variational methods and an EM algorithm. They used their findings in document modeling, text classification, and collaborative filtering. Nir Friedman, Dan Geiger, and Moises Goldszmidt showed that Tree Augmented Naive Bayes (TAN) outperformed naive Bayes [10]. Yet, the TAN maintained the same computational simplicity and robustness as the naive Bayes. Similar work has been done by Yeruva, S., Varalakshmi, M.S., Gowtham, B.P., Chandana, Y.H., and Prasad, P.K. for classifying Cell Anemia using a Deep neural network [23].

However, we consider directional vector formation to reduce the angle between the target class and maximize with the non-target class for classification. We have utilized cosine similarity between directional vectors and data for angle reduction and setting up directional vectors. Instead of making a decision boundary, a random vector is created in our approach to separate the classes.

3- Introduction to Support Directional Shifting Vector

Here, our proposed model can perform both binary classification and multiclass classification. We are considering a binary classification for primary understanding, where we have a dataset that contains data of two classes named target class and non-target class. The classification is performed based on the angle measured between two vectors, support direction vector, and support origin vector. Let us assume a vector space having two classes of the datapoints named A and B. Here, $V \in \mathbb{R}^{n*m}$ is a vector space with the datapoints $A \in V$ and $B \in V$. Let's $A \in V$ is the target class, and $B \in V$ is the non-target class. We need to find out such a direction vector of point $D \in V$ for which the angle of $angle \angle AOD$, the minimized and $angle\ \angle BOD$ gets maximized so that the target class makes a stronger relationship with the direction vector comparison to the non-target class. We have used dot product that is the measure of similarity between two non-zero vectors for the implementation. The value of cosine angle is in between -1 to 1 for angle 0 to 180 degrees. Angle 0 returns a maximum similarity of 1, and angle 180 bears a minimum similarity of -1. The similarity measurements rely on orientation rather than magnitude. For example, if two vectors are parallel, then the similarity index is 1. If they are opposite, then the similarity value is -1, and if they are perpendicular, then the value is 0. So, it is convenient to define the target class as '1' and the non-target class as '-1'. The characteristics of the target class are different from the non-target class. Figure 1 shows the target class and non-target classes visualization in vector space.

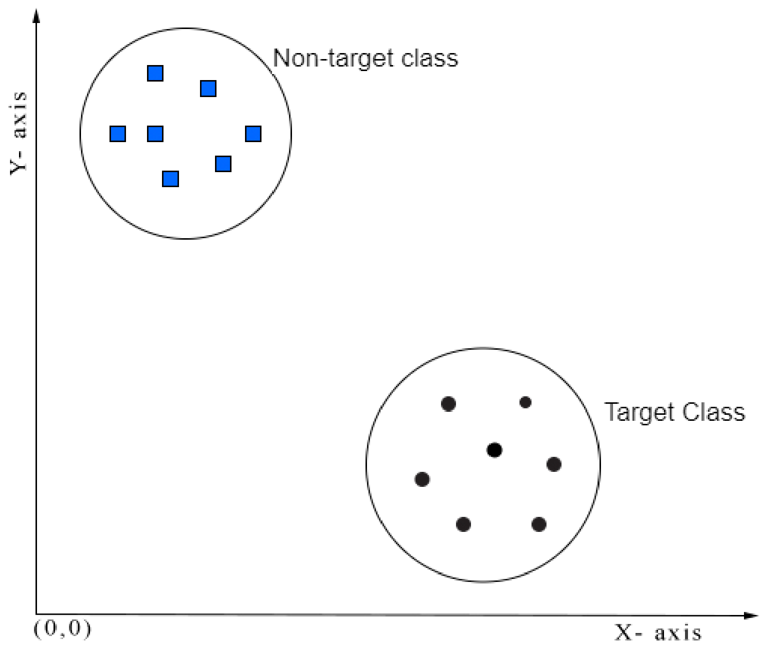

Figure 1. **Representation of dataset contains datapoints target class and non-target class.**

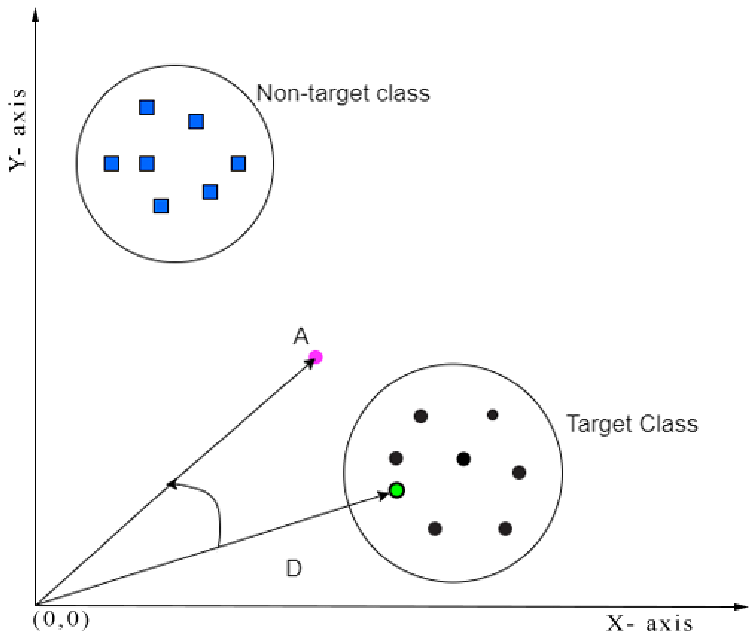

Figure 2. **Cosine similarity measurement with a particular data point A.**

In this model, we aim to separate the target class from the non-target class. In other words, we need to model a decision boundary. For example, the Support Vector Machine uses support vectors to form the decision boundary based on Euclidian distance. In the SVM algorithm, the better the margin, the better it suits for classifying unknown data. However, in this model, the boundary is not formed based on Euclidean distance rather, it is formed based on the angular measurement. This angle measurement is the measurement of cosine similarity, as shown in Figure 2. Ultimately, we build the Support Directional Shifting Vector in such a place from where the angular distance of all the data points of the target class will be smaller compared to the other non-target class. In other words, the angle measure of all target data points tries to place close to 0 degrees as possible. In contrast, the angle measure of all the data points of the non-target class attempts to remain as possible as close to 180 degrees.

Now, we need to determine whether the target point, A belongs to the target class or the non-target class. For this purpose, we measure the angle of the point 'A' with the directional vector using cosine similarity. If the measured angle value is smaller with respect to the target class (than with respect to the non-target class), then the test point 'A' is classified as '1' as target class; otherwise, the point 'A' is classified as '-1' for the non-target class.

Yet, in the proposed method, if some data points have a different location but the same angle with a linear function, they seem to be in the same class. But that may not be necessarily true in all cases. Figure 3 below illustrates this situation properly where we have a non-target class, and a target class datapoint and both of them make the same angle with the D vector.

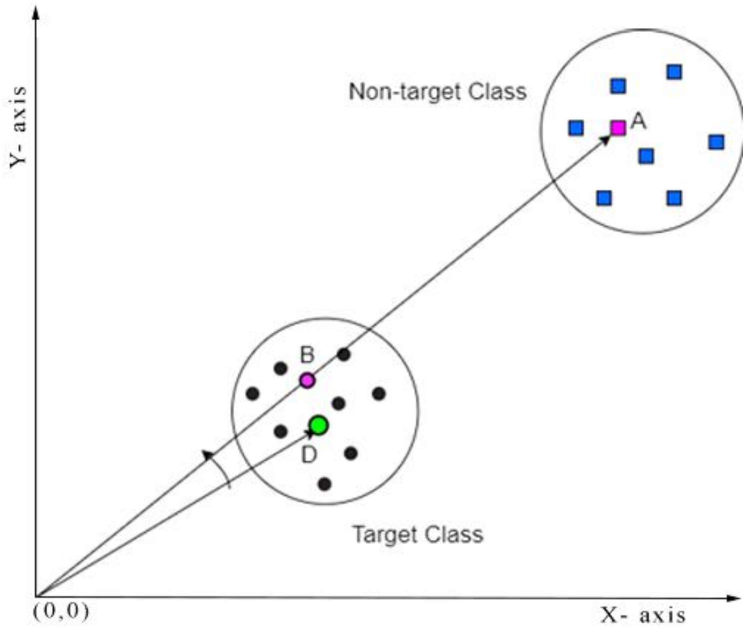

Figure 3. The problem in angle measurement for datapoint having different vector position but same direction with respect to the origin.

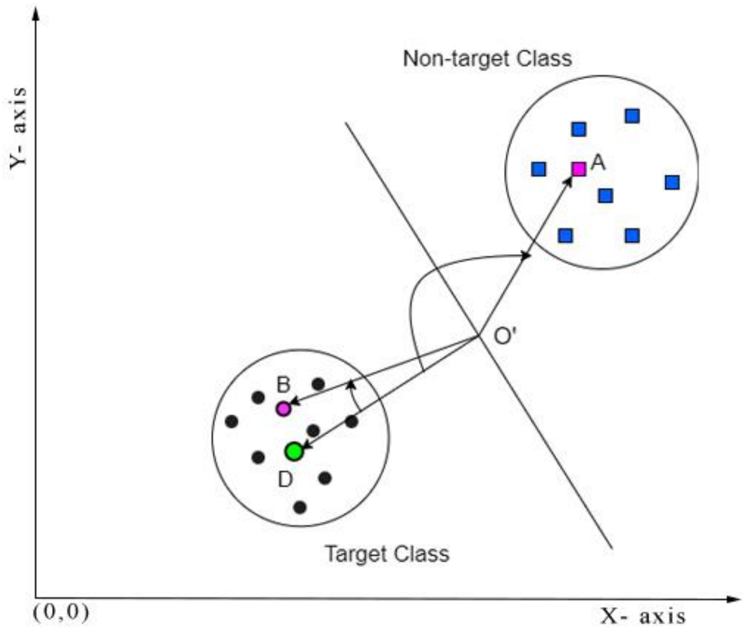

Figure 4. The solution of angle measurement problem by shifting the origin s.

However, the problem addressed in Figure 3 has been solved by shifting the origin from O to O' to make the angel different in a relevant way. The process of origin shifting has been illustrated in Figure 4, where we can observe that the origin shifting has changed the angle of both data points. This angle change has made it possible to classify both of the data points properly. From this figure, "B" is one of the data points of the target class, the ∠BO'D is less than 90 degrees, so the cosine similarity tells us that it is close to "1". Therefore, the data point "B" is in the target class. On the contrary, the data point "A" is on the opposite side of "B". And so, the cosine of the angle ∠AO'D is close to -1, and therefore it is classified as a non-target class.

4- Mathematical Model of SDSV

For developing a classification or regression model we actually build a function that maps the input to the output data. Mathematically, if there exists a relation between two variables x and y we need to develop a function f that can reproduce y if x is given. Classification and regression both are types of supervised machine learning techniques where classification is used to predict the label, and regression is used to predict the value. The mapping function is built by analyzing the relationship between the input variables(x) and the output variable(y). Mathematically, this can be represented by $y = f(x)$. This type of problem aims to measure the mapping function (f) as correctly as possible so that the output variable (y) for the dataset can be predicted whenever new input data (x) is available.

In supervised classification problems, the error in mapping function $f: x \rightarrow y$ is modelled using a loss function. The model accuracy is increased by reducing a loss function(L). When the loss of a model is minimum, the model can predict the classification with high accuracy. In the very first stage, it generates strategy function $g(x)$, and then passes is to an activation function $\mathcal{E}$ which generates probability assuming of the target. So, the model f can be defined as shown in Equation 1.

$$f(x) = \mathcal{E}\big(g(x)\big) \tag{1}$$

In gradient descent, for every possible move of function, the loss is computed to update Wight and to make an optimized solution. The loss(L) function can be defined as shown in Equation 2.

$$Loss = \sum_{i=1}^{n} (y, \hat{y}) \tag{2}$$

In our proposed model, two weights need to be updated during the loss calculation such as origin vector O' and directional vector(D). Eventually, both of the vectors will take such a position in vector space $O', D \in V$ that makes a linear function which is the strategy function $F: D, O' \rightarrow g$. At this stage, the angle with the target class will be the minimum possible value, an angle with the non-target class will the maximum possible value. The final output of this model will be a line that will separate the data points in two-dimensional vector space $V \in \mathbb{R}^2$.

Consider a line that passes through the point $O' = (x_{1,} y_{1,} z_1)$ and has a direction vector $\overrightarrow{O'} = (l, m, n)$ where l, m and n are non-zero real numbers.

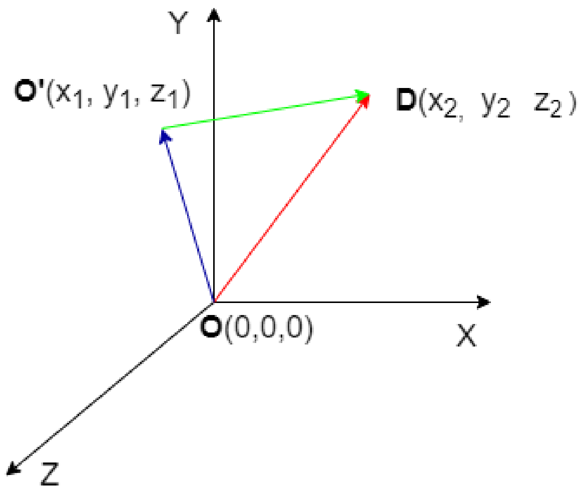

Figure 5. Representation of coordinate system in three-dimensional space.

Now let us consider $D = (x_2, y_2, z_2)$ be a random point on the line, and $\overrightarrow{O'D}$ will be parallel to $\overrightarrow{O'}$. See Figure 5. Now we can write the following way: $\overrightarrow{O'D} = t\overrightarrow{O'}$

$$\left(x_2 - x_1, y_2 - y_1, z_2 - z_1\right) = t(l, m, n)$$

$$t = \frac{x_2 - x_1}{l} = \frac{y_2 - y_1}{m} = \frac{z_2 - z_1}{n}$$

Therefore, any point $D = (x_2, y_2, z_2)$ on the line will satisfy Equation 3;

$$\frac{x_2 - x_1}{l} = \frac{y_2 - y_1}{m} = \frac{z_2 - z_1}{n} \tag{3}$$

The equation of a line, having the direction vector $\overrightarrow{O'} = (l, m, 0)$, passed through the point (x_1, y_1, z_1), is given by the two formulas $\frac{x_2 - x_1}{l} = \frac{y_2 - y_1}{m}$ and $z_2 = z_1$, where l, and m are non-zero real numbers. The line does not depend on its scalar but depends on its direction to measure the angle. Consequently, for any scalar value lambda (λ), it can be defined as Equation 4.

$$O'D = \lambda|O'D| \tag{4}$$

At this stage, we need to elect a proper activation function. We have selected the cosine function since we are working on an angular measurement-based model. We have set the classification threshold value of ω to classify the data. Usually, the threshold is $\omega = 90^0$ with respect to the angle and for cosine similarity, it returns $\cos(\omega) = 0$. Again, the cosine similarity value is bounded between -1 to 1. So, here +1 will indicate the maximum relationships, and -1 will indicate the minimum relationships. In general, using vanilla cosine similarity and considering the threshold value of $\omega = 0$, we can say 0 to 1 will refer to the target class and -1 to 0 will refer to the non-target class. In mathematics, cosine similarity is represented by the Equation 5.

$$cos\theta = \frac{A\cdot B}{|A|\times|B|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\times\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{5}$$

for ith features, the cosine similarity can be defined as Equation 6.

$$\hat{y} = \frac{\sum_i^n \overrightarrow{O'D_i}\cdot\overrightarrow{OX_i}}{\sqrt{\sum_i^n O'D_i^{\,2}} \times \sqrt{\sum_i^n O'X_i^2}} \tag{6}$$

In the matrix multiplication form, it can be defined as Equation 7.

$$\hat{y} = \frac{\sum_i^n O'D_i \cdot O'X_i^T}{\sqrt{\sum_i^n O'D_i^{\,2}} \times \sqrt{\sum_i^n O'X_i^2}} \tag{7}$$

By adding a bias, it can be written as Equation 8.

$$\hat{y} = \frac{\sum_i^n O'D_i \cdot O'X_i^T + b}{\sqrt{\sum_i^n O'D_i^{\,2}} \times \sqrt{\sum_i^n O'X_i^2}} \tag{8}$$

The element-wise multiplication of two matrices follows the rule of commutatively. Let us consider, n number of samples having a total of two classes and with ith features.

First, we have separated all the samples according to their classes like as $(1, 2, 3, \ldots k)$ as the target class and $k + 1, k + 2, \ldots n$ as a non-target class.

$X_{1,2,3\ldots k}, X_{k+1,k+2,\ldots n}$

Now the activation function with the target class is represented in Equation 9.

$$\angle\overrightarrow{O'D}_i \cdot \overrightarrow{O'X}_{1,2,\ldots k} \rightarrow 0° \tag{9}$$

And the activation function with non-target is represented in Equation 10.

$$\angle\overrightarrow{O'D}_i . \overrightarrow{O'X}_{k+1,k+2,\ldots.n} \rightarrow 180° \tag{10}$$

In similarity mode, they will be represented as shown in Equations 11 and 12.

$$S\left(\overrightarrow{O'D}_i, \overrightarrow{O'X}_{1,2,3,\ldots k}\right) \rightarrow 1 \tag{11}$$

$$S(\overrightarrow{O'D}_i\,, \overrightarrow{O'X}_{k+1,k+2,\ldots.n}) \rightarrow 0 \tag{12}$$

Here, the loss will be minimum in relation mode. The equation of loss is shown in Equations 13 and 14.

$Loss = 1 - cos\theta$ for target class (13)

$Loss = 1 + cos\theta$ for non-target class (14)

In general, if the target class is labeled as 1 and the opposite classes as -1, a discrete and nonlinear loss function can be defined as shown in Equation 15.

$$\text{Loss} = L\sum_{i=1}^{n} (y, \hat{y}) \tag{15}$$

if $\hat{y} \geq 0$ and $y = 1$ then 0

if $y < 0$ and $y = -1$ then 0

otherwise 1

For continuous value, usually cross-entropy loss is used. The categorical cross-entropy loss is represented in Equation 16.

$$Loss = -\frac{1}{n}\sum_{i=1}^{n} y_i \log(\widehat{y_i}) + (1 - y_i)\log(1 - \widehat{y_i}) \tag{16}$$

But before moving to the loss function, we need to scale the range [-1, 1] to [0, 1]. We can also define the absolute loss function as Equation 17. The square loss is also shown in Equation 18.

$$\text{Loss} = L(\sum_{i=1}^{n} |y - \hat{y}|) \tag{17}$$

$$\text{Loss} = L(\sum_{i=1}^{n} (y - \hat{y})^2) \tag{18}$$

Gradient descent is an optimization algorithm that is useful for every machine learning algorithm to minimize a cost function. It is a method that finds a minimum of a loss function by figuring out in which direction the function's slope is rising the most steeply and moving in the opposite direction. The loss function for SDSV is convex that means the gradient should be in global minima after completing the training where the gradient gets stuck. On the other hand, the loss function for a multilayer neural network is non-convex. The gradient may get stuck in the local minima and may not find the global minima. Let's consider a vector space V, and the optimization function can be defined by Equation 19.

$$\theta_t = \text{argmin} \frac{1}{n} \sum_{i=1}^{n} L\left(y^{(i)}, x^{(i)}; \theta\right) \tag{19}$$

where θ generally refers to weight (w) for origin vector O', directional vector(D), and bias (b). The x and y are training examples and labels, respectively. The notation n is the size of the datasets, and L is the loss function. To get an optimum $\theta,$ we need to update for several iterations. The update can be done by Equation 20.

$$\theta_{t+1} = \theta_t - \eta \nabla L(f(x, \theta), y) \tag{20}$$

where, η is the learning rate.

The gradient of the loss with respect to weights (origin, direction, bias) will be $\frac{\partial L(y,\hat{y})}{\partial D}$, $\frac{\partial L(y,\hat{y})}{\partial O'}$ $\frac{\partial L(y,\hat{y})}{\partial b}$.

5- Experiment and Results

We have implemented the proposed method to solve various real-life problems that validate the acceptability of the proposed SDSV model. Furthermore, we have used different types of datasets. In this section, we will introduce the dataset, experimental method, training and testing procedure, result, and discussion. In Addition to these, we have shown performance comparison with other standard machine learning algorithms.

5-1- Dataset

To determine the performance of the proposed model, we have used it to classify supervised data of three different datasets. Two of them are from the NLP domain and the other dataset is from the ML domain. We will discuss the dataset property and the source and features in detail in this section.

The first dataset we have used is named 'Amazon Fine Foods reviews' which is a review collection of various food of a period of more than ten years [16]. This dataset includes the customer details, user information rating, and some plain text of the review. This dataset contains a total of 568,454 reviews of 74,258 products by 256,059 users. The average number per user is 260. The number of users who have contributed more than 50 reviews is 250.

The second dataset is again from the NLP domain that contains the text of movie review and is named 'Large Movie Review Dataset' [15]. This is a leveled dataset that is suitable for binary supervised machine learning classification. This dataset consists of 50 k observation amount these 25k is used for raining and the rest 25k is used for testing. There are some unlevelled data as well. In addition, this dataset comes with both unprocessed and preprocessed review text data. However, we have used the preprocessed data for our convenience.

We have also implemented a hits model on diabetics' dataset from Kaggle [22]. This data has been recorded using an automatic device and the record is kept on paper. This dataset contains four fields per observation including date of birth, time, code, and value. The code field contains a numerical value that can be deciphered to text. There is a total of 20 attributes listed in this dataset. This dataset is available for public use.

5-2- Experimental Setup

This article represents work that has been done completely in a simulation environment. We first came up with our mathematical model. In order to implement this model, we have built the model, we have developed a python script that has been executed on a google Colaboratory, a cloud computing platform, based python programming environment. We have used Google colab based GPU for faster and parallel computing. The total runtime for developing all the models was more than 200 hours. We have also released a python module of this model for public use that will help users to use this model in a very easy and efficient way. The implementation of the proposed model has been done in the following steps shown in the figure below.

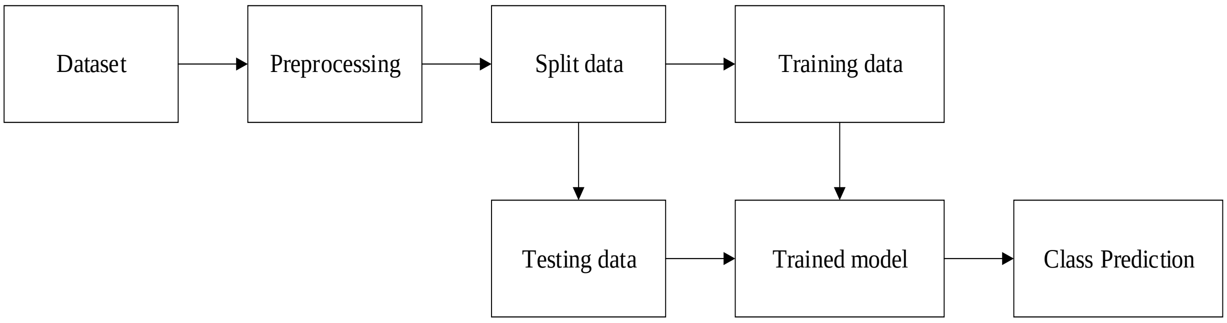

Figure 6. Flowchart of the proposed working methodology.

5-3- Training Method

Algorithm: Support Directional Shifting Vector (SDSV) needs two vectors of points. The first one is the support origin vector (SOV), the other one is the directional support vector (SDV). The angle is measured between the data point and the directional support vector from the origin vector. The parameters used in this method, during training time, are learning rate =0.01, batch size = 8, epoch =100, l2 regularization =0.03, categorical cross-entropy loss.

Training Method:

Input	:	Training example and training label
Output	:	Trained model
Step 1	:	Separate all the data points into two classes, such as target class and non-target class.
Step 2	:	The target class should be labeled as +1, and the non-target class should be -1, or 0 for cross-entropy
Step 3	:	Randomly need to be chosen two support vectors for the origin and direction
Step 4	:	Calculate the cosine similarity between the directional vector and all the data points from the origin vectors
Step 5	:	Scaling the calculated similarity to transfer the range [0, 1] from [-1, 1] for categorical cross-entropy
Step 6	:	Calculate loss
Step 7	:	Using gradient descent algorithm for optimization
Step 8	:	Update the position of origin and directional vector
Step 9	:	Go to step 4 until the loss is minimum as possible as
Step 10	:	Stop

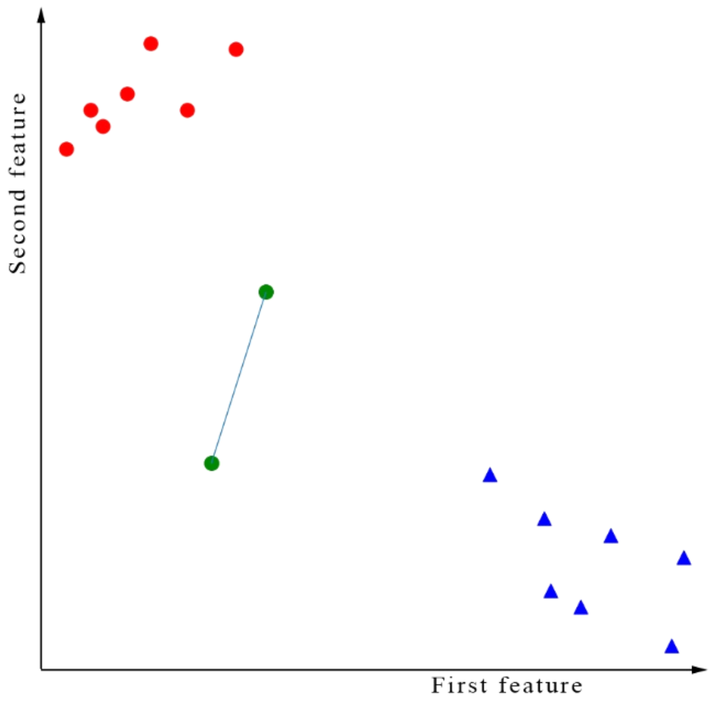

Figure 7. Random generation of shifting origin and shifting support direction vector in the first epoch.

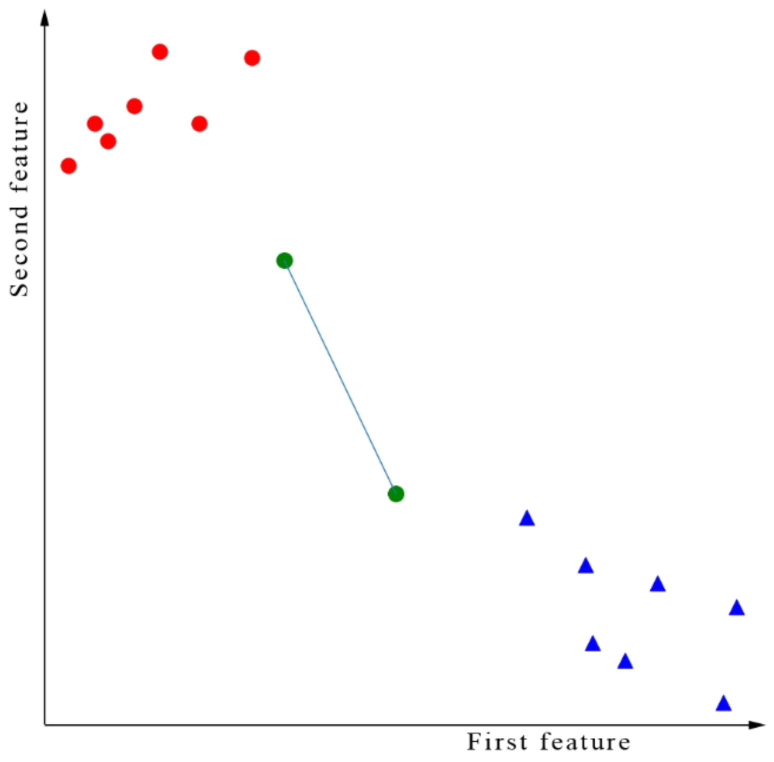

Figure 8. **Update of shifting origin and shifting support direction vector using gradient descent optimization.**

In the real-world scenario, the same type of data is categorized into the same class. There are divergent conceptual techniques proposed by the researchers to address the classification problems. Understanding these concepts we proposed for classification, for example, we may consider, two random vectors (in Figure 7) are being created for origin and direction. These two vectors are expressed as the direction from origin to direction vector to calculate the angle. Then, all the training examples' angle is measured with respect to the origin vector to the direction vector. After the first epoch using the gradient descent algorithm, the origin and directional vector shift to another position (in figure 8, and figure 9). In this way, the SDSV shifts to space another space until the training loss is optimum (in Figure 10).

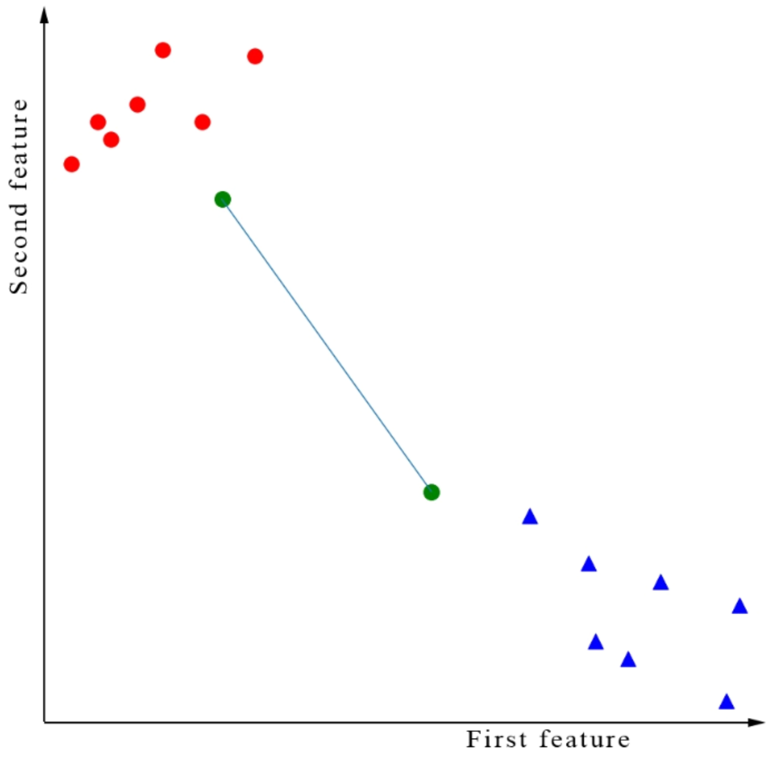

Figure 9. **Update of shifting origin and shifting support direction vector using gradient descent optimization.**

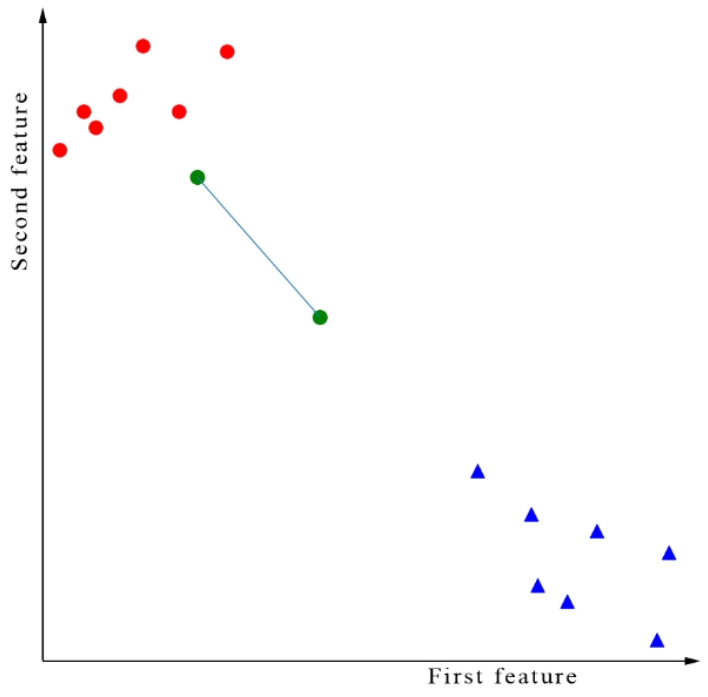

Figure 10. **Optimized shifting origin and shifting support direction vector.**

5-4- Testing Method

Input	:	Testing example
Output	:	Classified label
Step 1	:	Pass the same scale of data to the trained SDSV model
Step 2	:	Make the decision based on threshold and similarity
Step 3	:	Stop

For evaluating the trained model, let us consider, X_{test1} is the test data. Firstly, from figure 11, we calculated the angle measure between the X_{test1} and the trained directional vector from the origin, which is almost 20°. Thus, using cosine similarity, we have $cos\angle 20^{\circ} = 0.94$, which is a positive value and means 94%, similar to the target class. Secondly, for other data point X_{test2}, the angle is measured almost 100°. Similarly, we have $cos\angle 100^{\circ} = -0.17$, which is a negative value, means that this point is classified as the opposite class.

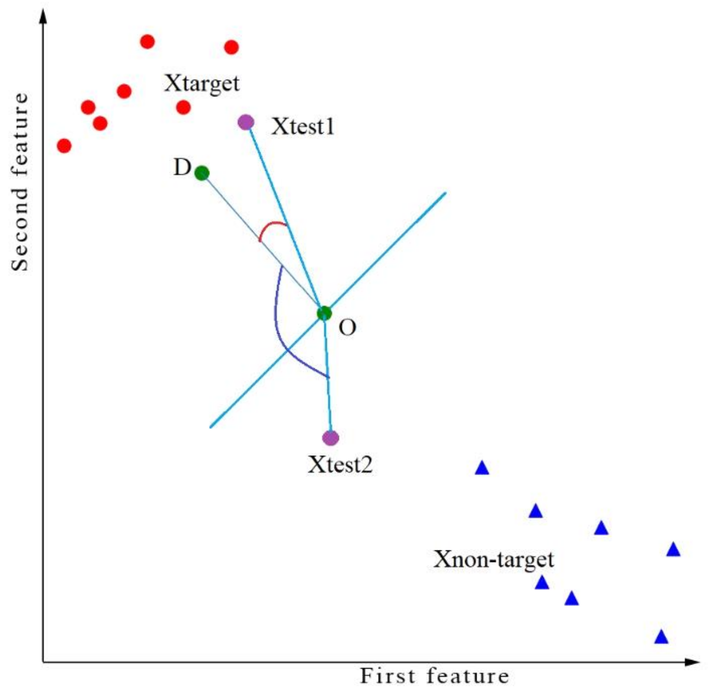

Figure 11. **Classification of the test data points using the trained model.**

5-5- Results and Discussion

In order to evaluate the proposed algorithm, we have tested our proposed algorithm for solving NLP problems as well as ordinary machine learning classification problems. The NLP problem we solved includes which is a binary classification problem, and we used a popular movie review dataset named “IMDB Dataset of 50K Movie Reviews”. To show the divergent applicability of the proposed method we also consider multiclass classification problems and the SDSV method is evaluated on “Amazon fine food review.” Our proposed technique can also be used for machine learning classification problems. We showed comparative results of our proposed method with other standard algorithms, and these results are listed in Table1 and Table 2, Table 3 respectively.

For the NLP problem, we used three-word representation techniques such as term frequency-inverse document frequency or TF-IDF [11,14], Word2Vec [17, 18], FastText [4]. The TF-IDF is the traditional technique to represent a word in the document in the corpus. The term frequency-inverse document frequency (tf–idf) is a statistical representation used in the NLP to reflect how essential a word is to a document in a corpus. Word embedding is a technique to represent words in the vector space using a numerical value. The popular techniques are Word2vec and FastText. Word2Vec is a software tool that used Skip-gram and continuous back of world (CBOW), neural network model, take classifier weight during training time to semantically represent words as word embeddings. The FastText works in the same way. Furthermore, it can handle out of vocabulary (OOV) words. We used these three-word represent techniques in evaluating the proposed method.

We have used two different model evaluation parameters such as accuracy, and F1 score. We have utilized some standard machine learning algorithms, including Random Forest (RF), Support Vector Machine (SVM), Logistic Regression (LR), K-Nearest Neighbor (KNN), Latent Dirichlet Allocation (LDA), Impact Learning (IL) [17], and SDSV to show comparative results for recommending our developed algorithm.

From Table 1, If we look at the accuracy field, we can see that the accuracy is somewhat similar. However, the accuracy of the LR, Adaboost, and our proposed method, SDSV, are comparable in some way. For model evaluation and performance analysis, the F1 score is a crucial factor to be considered. The more the F1 score, the better the model. When TF-IDF has used a word representation technique, the Logistic Regression outperformed other algorithms. Albeit the SDSV’s accuracy is comparable with it. When we used Word2Vec as a word embedding technique, our proposed algorithm outperformed another algorithm. After training the FastText, we see that it is the best suitable word representation for the SDSV. Using this word representation, we have gotten the highest accuracy and F1 score.

Table 1. Binary classification result: IMDB Dataset of 50K Movie Reviews.

Feature Extraction Method	**TF-IDF**		**Word2Vec**		**FastText**	
Name	**Accuracy**	**F1**	**Accuracy**	**F1**	**Accuracy**	**F1**
Logistic Regression	0.8772	0.8787	0.8592	0.86	0.8384	0.8384
Impact learning	0.7932	0.7948	0.8556	0.8674	0.8246	0.8277
Random Forest	0.8376	0.8372	0.8268	0.8259	0.7712	0.7713
KNN	0.8321	0.8215	0.7923	0.8011	0.78114	0.7312
LDA	0.8691	0.86991	0.8411	0.8481	0.8301	0.8312
AdaBoost	0.87131	0.87121	0.85001	0.8491	0.84031	0.8381
SDSV	0.87009	0.86912	0.8602	0.8621	0.84125	0.8431

Table 2. Multiclass classification result: Amazon fine food review dataset.

Feature Extraction Method	**TF-IDF**		**Word2Vec**		**FastText**	
Name	**Accuracy**	**F1**	**Accuracy**	**F1**	**Accuracy**	**F1**
Logistic Regression	0.8697	0.8436	0.83924	0.8547	0.8219	0.8424
Impact learning	0.83044	0.8433	0.84338	0.8548	0.82474	0.8325
Random Forest	0.84183	0.863	0.81258	0.834	0.78332	0.8125
KNN	0.81352	0.8215	0.78421	0.78913	0.76242	0.78252
LDA	0.8486	0.8395	0.84725	0.81673	0.82815	0.83151
AdaBoost	0.86134	0.86473	0.84592	0.8521	0.82515	0.83009
SDSV	0.86981	0.8671	0.84002	0.84913	0.82913	0.84001

To evaluate the proposed model, we used Amazon fine food review dataset, a multiclass labeled dataset, for recommending the applicability of the model in multiclass problems. From Table 2. It can be noticed that the accuracy and F1 score is the highest whining utilizing the tf-idf vectorizer. In addition to this, the accuracy is also the best when utilizing the FastText model. Furthermore, the accuracy and the F1 score are comparable with the highest accuracy and score such as Impact learning and LDA when we utilized the word2Vec technique. So, we concluded that this algorithm can be used for the NLP-based classification problem.

We have found the best result in diabetic detection in terms of accuracy and F1 score. It is clear from Table 3, that the accuracy (83%), F1 (71%), score of the SSDV outperformed all the other algorithms. According to the model evaluation parameters, our proposed algorithm SDSV is the best for diabetics' detection. So, we concluded that this algorithm can be the best algorithm used for machine learning classification problems today.

Table 3. Diabetes detection result: Pima Indians Diabetes Database.

Name	RF	SVM	NB	LR	K NN	DT	LDA	IL	SDSV*
Accuracy	0.79	0.79	0.79	0.82	0.79	0.70	0.82	0.83	0.83
F1	0.62	0.61	0.64	0.68	0.65	0.57	0.68	0.69	0.71

6- Conclusion and Future Work

In today's world, classification is an important issue for an intelligent system. From the very beginning of pattern recognition, many algorithms have been proposed. These algorithms are updating, or new algorithms are being developed to get good performance for a digital system. We introduced a robust classifier, Support Directional Shifting Vector, which gave a good result compared to other standard algorithms. Our approach introduces two vectors, a support Directional Vector, and Support Origin Vector, to measure cosine similarity. At the starting point, the first vector is formed randomly and is shifted to another location until the loss considering the target class and the non-target class is minimum. The origin vector is used to handle misclassification when the target data point and the non-target data point have the same angle measure with respect to the origin. We used a gradient descent algorithm for optimization and the right position of direction and origin vector.

Since the shifting vectors only consider linear combinations, the classification is performed linearly based on the angle. The nonlinear problems cannot be handled with these shifting vectors properly. However, we plan to use the linear separation in nonlinear issues, considering another variable that we are calling radius reduction on the state. The radius reduction will be the minimum radius on every separate state with separate values. In the future, we have a plan to execute the radius reduction theory for handling nonlinear problems in a vector space. Besides, we have a plan to build a high-label cloud computing-based API to use the SDSV in any machine learning or data science-based problems.

7- Declarations

7-1- Author Contributions

Conceptualization, mathematical architecture, model development, M.K.; methodology, validation, M.K., I.H. and A.T.; data curation, model comparison, N.J.P.; writing original draft preparation, I.H.; writing—review and editing, A.T. and Z.R.M.A.; visualization, K.H.; funding acquisition, supervision, Z.R.M.A.; All authors have read and agreed to the published version of the manuscript.

7-2- Data Availability Statement

Publicly available datasets were analyzed in this study. The datasets used here can be found in [15, 16, 22].

7-3- Funding

This research is funded by Ministry of Higher Education under Fundamental Research Grant Scheme (FRGS) No. FRGS/1/2019/ICT03/UMP/02/2 (University reference RDU1901194).

7-4- Conflicts of Interest

The authors declare that there is no conflict of interests regarding the publication of this manuscript. In addition, the ethical issues, including plagiarism, informed consent, misconduct, data fabrication and/or falsification, double publication and/or submission, and redundancies have been completely observed by the authors.

8- References

[1] Alkhateeb, Jawad Hasan. "An Effective Deep Learning Approach for Improving Off-Line Arabic Handwritten Character Recognition." International Journal of Software Engineering and Computer Systems 6, no. 2 (2020): 53-61.

[2] Aseri, Nur Azieta Mohamad, Mohd Arfian Ismail, Abdul Sahli Fakharudin, and Ashraf Osman Ibrahim. “Review of the Meta-Heuristic Algorithms for Fuzzy Modeling in the Classification Problem.” International Journal of Advanced Trends in Computer Science and Engineering 9, no. 1.4 (September 15, 2020): 387–400. doi:10.30534/ijatcse/2020/5691.42020.

[3] Blei, David M., Andrew Y. Ng, and M. Jordan. "Latent Dirichlet Allocation Michael I." Jordan. Journal of Machine Learning Research 3 (2003).

[4] Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov. “Enriching Word Vectors with Subword Information.” Transactions of the Association for Computational Linguistics 5 (December 2017): 135–146. doi:10.1162/tacl_a_00051.

[5] Breiman, Leo. "Random forests." Machine learning 45, no. 1 (October 2001): 5-32. doi:10.1023/A:1010933404324.

[6] Cortes, Corinna, and Vladimir Vapnik. “Support-Vector Networks.” Machine Learning 20, no. 3 (September 1995): 273–297. doi:10.1007/bf00994018.

[7] Cunningham, Padraig, and Sarah Jane Delany. "k-Nearest neighbour classifiers: (with Python examples)." arXiv preprint: 2004.04523 (2020).

[8] Fisher, R. A. “The Use of Multiple Measurements in Taxonomic Problems.” Annals of Eugenics 7, no. 2 (September 1936): 179–188. doi:10.1111/j.1469-1809.1936.tb02137.x.

[9] Freund, Yoav, and Robert E. Schapire. “A Decision-Theoretic Generalization of on-Line Learning and an Application to Boosting.” Computational Learning Theory (1995): 23–37. doi:10.1007/3-540-59119-2_166.

[10] Friedman, Nir, Dan Geiger, and Moises Goldszmidt. "Bayesian network classifiers." Machine learning 29, no. 2 (1997): 131-163. doi:10.1023/a:1007465528199.

[11] Sparck Jones, Karen. “A Statistical Interpretation of Term Specificity and Its Application in Retrieval.” Journal of Documentation 28, no. 1 (January 1972): 11–21. doi:10.1108/eb026526.

[12] Liu, Huawen, Shichao Zhang, Jianming Zhao, Xiangfu Zhao, and Yuchang Mo. “A New Classification Algorithm Using Mutual Nearest Neighbors.” Ninth International Conference on Grid and Cloud Computing (November 2010): 52–57. doi:10.1109/gcc.2010.23.

[13] López-Cruz, Pedro L., Concha Bielza, and Pedro Larrañaga. “Directional Naive Bayes Classifiers.” Pattern Analysis and Applications 18, no. 2 (July 4, 2013): 225–246. doi:10.1007/s10044-013-0340-z.

[14] Luhn, H. P. “A Statistical Approach to Mechanized Encoding and Searching of Literary Information.” IBM Journal of Research and Development 1, no. 4 (October 1957): 309–317. doi:10.1147/rd.14.0309.

[15] Maas, Andrew, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. "Learning word vectors for sentiment analysis." In Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies, (2011): 142-150.

[16] McAuley, Julian John, and Jure Leskovec. “From Amateurs to Connoisseurs.” Proceedings of the 22nd International Conference on World Wide Web - WWW ’13 (2013). doi:10.1145/2488388.2488466.

[17] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. “Efficient estimation of word representations in vector space.” arXiv Paper. 1301.3781 (2013).

[18] Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. "Distributed representations of words and phrases and their compositionality." In Advances in neural information processing systems, (2013): 3111-3119.

[19] Pernes, Diogo, Kelwin Fernandes, and Jaime Cardoso. “Directional Support Vector Machines.” Applied Sciences 9, no. 4 (February 19, 2019): 725. doi:10.3390/app9040725.

[20] Quinlan, J. Ross. "Induction of decision trees." Machine learning 1, no. 1 (1986): 81-106. doi:10.1023/A:1022643204877.

[21] Rosenblatt, Frank. “Principles of Neurodynamics. Perceptrons and the Theory of Brain Mechanisms.” Cornell Aeronautical Lab Inc. Buffalo NY, (1961).

[22] UCI. 2016. Pima Indians Diabetes Database | Kaggle. UCI Machine Learning. Available online: https://www.kaggle.com/uciml/pima-indians-diabetes-database (accessed on 30 June 2021).

[23] Yeruva, Sagar, M. Sharada Varalakshmi, B. Pavan Gowtham, Y. Hari Chandana, and PESN. Krishna Prasad. “Identification of Sickle Cell Anemia Using Deep Neural Networks.” Emerging Science Journal 5, no. 2 (April 1, 2021): 200–210. doi:10.28991/esj-2021-01270.