East Asian Math. J.
Vol. 37 (2021), No. 3, pp. 333–354
http://dx.doi.org/10.7858/eamj.2021.023

COMPARATIVE STUDY OF THE PERFORMANCE OF SUPPORT VECTOR MACHINES WITH VARIOUS KERNELS

Seong-Uk Nam, Sangil Kim *, HyunMin Kim, and YongBin Yu

Abstract. A support vector machine (SVM) is a state-of-the-art machine learning model rooted in structural risk minimization. SVM is underestimated with regards to its application to real world problems because of the difficulties associated with its use. We aim at showing that the performance of SVM highly depends on which kernel function to use. To achieve these, after providing a summary of support vector machines and kernel function, we constructed experiments with various benchmark datasets to compare the performance of various kernel functions. For evaluating the performance of SVM, the F1-score and its Standard Deviation with 10-cross validation was used. Furthermore, we used taylor diagrams to reveal the difference between kernels. Finally, we provided Python codes for all our experiments to enable re-implementation of the experiments.

1. Motivation and Goal

SVMs are state-of-the-art machine learning techniques with their root in structural risk minimization [57, 58]. Additionally, SVMs have been successfully applied to a variety of real-world problems [4] such as particle identification, face recognition, text categorization, bioinformatics, and problems in civil and electrical engineering. However, except for some successful examples, SVM is underestimated with regards to its application to real world problems, because No one easily knows how to adapt SVM in their applications.

Here, all we want to say is that trying vaiouse kernel funtion is a simple solution to use SVM for thier applications. However, Not only do many studies use rbf kernels and polynomial kernels, but many libraries also provide only two kernels embedded in kernel functions. For this reason, we provide a summary of the support vector machines and kernel functions and show how different results can be obtained with various kernel functions.

Received May 3, 2021; Accepted May 31, 2021.
2010 *Mathematics Subject Classification.* 11A11.
Key words and phrases. Machine Learning, Support Vector Machine, Kernel Functions, Taylor Diagram.
This research was supported by PNU-RENovation(2018-2019).

Remainder of this paper is organized as follows. Section 2 summarizes various studies that have solved problems by applying a special kernel. It also classifies and describes several different types of kernels. Section 3 summarizes the experimental designs and results based on the UCI Dataset. Finally, Section 4 discusses the results of this study.

2. Previous Work

SVM, as a hard margin classifier, was proposed by Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik [1]. It has been developed continuously since it was first proposed. Many efforts have been made to apply it to various real problems [4], and to improve learning and inference time of SVM by enhancing the algorithm [82, 11] and using the GPU, FPGA device [80, 81]. New kernel functions have also been continuously suggested and studied [6, 9]. Furthermore, there are many helpful surveys on SVM [2, 3, 5, 7, 8]. In this section, we want to give a brief introduction of SVM including the notation, formula, and the kernel function.

2.1. Notation

Necessarily, we will use some datasets, hypothesis function, and cost functions. In this section, therefore, before the study detail of SVM, we will consider some basic definitions and concepts in the ordinal machine learning task.
In the ordinal machine learning task, we must first use some datasets. WLOG, the dataset can be represented by $n \times (m+1)$ matrix. Let the matrix D denote the dataset. Simply, n represents the number of examples of datasets, and $m+1$ expresses the number of variables. The variables are divided into m explanatory variables $X_{n\times m}$ and 1 target variable $Y_{n\times 1}$, simply X and Y. The equation can be expressed as

$$\begin{aligned} D &= \left[\begin{array}{c|c} X_{n\times m} & Y_{n\times 1} \end{array}\right] \text{, simply } \left[\begin{array}{c|c} X & Y \end{array}\right] \\ &= \left[\begin{array}{c|c} \mathbf{x}^{(1)} & y^{(1)} \\ \mathbf{x}^{(2)} & y^{(2)} \\ \vdots & \vdots \\ \mathbf{x}^{(n)} & y^{(n)} \end{array}\right] \\ &= \left[\begin{array}{cccc|c} \mathbf{x}_1^{(1)} & \mathbf{x}_2^{(1)} & \cdots & \mathbf{x}_m^{(1)} & y^{(1)} \\ \mathbf{x}_1^{(2)} & \mathbf{x}_2^{(2)} & \cdots & \mathbf{x}_m^{(2)} & y^{(2)} \\ \vdots & \vdots & & \vdots & \vdots \\ \mathbf{x}_1^{(n)} & \mathbf{x}_2^{(n)} & \cdots & \mathbf{x}_m^{(n)} & y^{(n)} \end{array}\right] \end{aligned}$$

where $(x^{(i)}, y^{(i)})$ denotes the i-th example with $x^{(i)} = (x_j^{(i)})_{j=1\cdots m}$, and $x_j^{(i)}$ means j-th variable of the i-th example. Occasionally, if there is no confusion, we omit the notation (i) of $x^{(i)}$, $x_j^{(i)}$, and $y^{(i)}$ as x, x_j , and y for convenience. We use x, x' to represent only two different examples (in section 2.3) rather $x^{(i)}, x^{(j)}$. And we call x as a pattern. Let a nonempty set $\mathbb{X}$ denote the domain, and $\mathbb{Y}$ denote the target space. $X_{n\times m}$ and $Y_{n\times 1}$, simply X and Y, are originated from the domain $\mathbb{X}$ and the target space $\mathbb{Y}$, respectivly. This means that X, Y is just a sample of $(\mathbb{X}, \mathbb{Y})$. We define a feature space by mapping the data into a higher-dimensional space, to obtain a better representation of the patterns. The feature map is denoted by Φ, and we have the feature space $\mathbb{H}$ as a co-domain of the map:

$$\Phi : \mathbb{X} \to \mathbb{H}$$

After transforming all the pattern $x \in X$ to the feature space, we get the new dataset D' with a high-dimension.

$$D' = \left[\begin{array}{cccc|c} \Phi(\mathbf{x}^{(1)})_1 & \Phi(\mathbf{x}^{(1)})_2 & \cdots & \Phi(\mathbf{x}^{(1)})_{m'} & y^{(1)} \\ \Phi(\mathbf{x}^{(2)})_1 & \Phi(\mathbf{x}^{(2)})_2 & \cdots & \Phi(\mathbf{x}^{(2)})_{m'} & y^{(2)} \\ \vdots & \vdots & & \vdots & \vdots \\ \Phi(\mathbf{x}^{(n)})_1 & \Phi(\mathbf{x}^{(n)})_2 & \cdots & \Phi(\mathbf{x}^{(n)})_{m'} & y^{(n)} \end{array}\right]$$

Definition 2.1. $(x, y, h_\theta(x)) \in X \times Y \times Y$ denote the triplet consisting of a example x, an observation y and a prediction $h_\theta(x)$. The map $c : X \times Y \times Y \to [0, \infty)$ with the property $c(x, y, y) = 0$ for all $x \in X$ and $y \in Y$ will be called a **loss function**.

Definition 2.2. For a given dataset D and a loss function c, we define the **cost function** $J(\theta)$ with the sum of loss over all examples, as

$$J(\theta) = \sum_{i=1}^{n} c(\mathbf{x}^{(i)}, y^{(i)}, h_\theta(\mathbf{x}^{(i)})) \tag{2.1}$$

Similarly, we define the **regularized cost function** $J_{reg}(\theta)$ as

$$J_{reg}(\theta) = \sum_{i=1}^{n} c(\mathbf{x}^{(i)}, y^{(i)}, h_\theta(\mathbf{x}^{(i)}) + \lambda\Omega(h_\theta(x)) \tag{2.2}$$

In the equation above, λ is a constant. Using this cost function, we can find the best parameter which gives more accurate predictions in sense of given cost function. We define the best parameter as optimal parameter, and it is denoted by θ^0

2.2. The Formulation

Most machine learning algorithms are determined by the hypothesis function, loss function, and regularization term. SVM may appear different from other

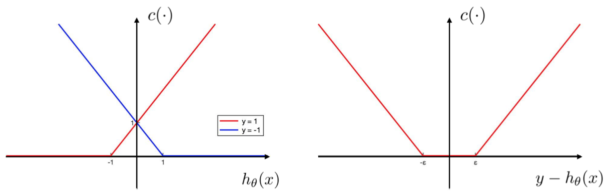

FIGURE 1. The left one represents the hinge loss for classification problem, defined as $c(x, y, h_\theta(x)) = \max(0, 1 - yh_\theta(x))$. Right one is for the epsilon intensive loss for regression problem, $c(x, y, h_\theta(x)) = \max(0, |y - h_\theta(x)| - \epsilon)$.

machine learning models as it needs to solve the constrained optimization problem. However, it can be generalized in the same way as follows [10]. The SVM classifier (regression) is obtained by:

$$\begin{cases} h_\theta(x) & = wx + b \\ c(x, y, h_\theta(x)) & = \max(0, 1 - yh_\theta(x)) \\ \Omega(h_\theta(x)) & = \frac{1}{2}\|w\|^2 \end{cases}$$

$$\begin{cases} h_\theta(x) & = wx + b \\ c(x, y, h_\theta(x)) & = \max(0, |y - h_\theta(x)| - \epsilon) \\ \Omega(h_\theta(x)) & = \frac{1}{2}\|w\|^2 \end{cases}$$

In the above mentioned equation, the loss function that defines the SVM classifier is known as hinge loss, and the loss function that defines the SVM regressor is known as an epsilon intensive loss fig. 1. In particular, there are several options for the loss function that defines the SVM regressor such as epsilon intensive loss, Laplacian loss, Gaussian loss, Hubers robust loss, polynomial loss, and piecewise polynomial loss [10, 2]. In this study, we only used the epsilon intensive loss. To train the SVM model in accordance with the data in these settings, the parameters must be trained to minimize the cost function. However, the gradient descent method cannot be simply applied due to the nature of the SVMs loss function. Therefore, it is needed to consider the dual problem corresponding to the original problem, as follows [10]:

$$\begin{cases} h_\theta(x) & = \sum_{i=1}^{m} \alpha_i y_i < x, x_i > +b \\ \text{maximize} & = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j < x_i, x_j > \\ \text{subject to} & \begin{cases} \alpha_i \geq 0, i = 1, \cdots, m, \\ \sum_{i=1}^{m} \alpha_i y_i = 0 \end{cases} \end{cases}$$

$$\begin{cases} h_\theta(x) & = \sum_{i=1}^{m} (\alpha_i^* - \alpha_i) < x_i, x > +b \\ \text{maximize} & \begin{cases} -\frac{1}{2} \sum_{i,j=1}^{m} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) < x_i, x_j > \\ -\epsilon \sum_{i=1}^{m} (\alpha_i^* + \alpha_i) + \sum_{i=1}^{m} y_i (\alpha_i^* - \alpha_i), \end{cases} \\ \text{subject to} & \sum_{i=1}^{m} (\alpha_i - \alpha_i^*) = 0 \text{ and } \alpha_i, \alpha_i^* \in [0, C/m] \end{cases}$$

2.3. Kernel Functions

The process of obtaining meaningful features from the given data determines the performance of the algorithm. In particular, achieving a high score in a machine learning algorithm competition such as Kaggle is determined in most cases, in which some features were found, rather than the model used. The process of creating these variables is easy when we use SVM, because it is closely related to the kernel function. Let us assume that $< x^{(i)}, x^{(j)} >$ is calculated by $< \Phi(x^{(i)}), \Phi(x^{(j)}) >$, considering a specific feature space. The kernel function can be understood as a function that calculates the inner product in a substituted feature space.

Definition 2.3. Let we define a similarity measure K of the form $K : X \times X \to \mathbb{R}$, which is a function that for given two patterns $x^{(i)}$ and $x^{(j)}$ returns a real number characterizing their similarity. For a given feature map Φ, if there exists a function K that satisfies

$$K(x^{(i)}, x^{(j)}) =< \Phi(x^{(i)}), \Phi(x^{(j)}) >,$$

we called such a function K as the kernel function.

There are many kernel functions (as summarized in 2.3.2). Kernel methods map the data onto higher dimensional spaces with the aim that the data could become more easily separated or better structured in this higher dimensional space. This is derived from the fact that, if we first map our input data onto a higher dimensional space, a linear algorithm operating in this space will behave non-linearly in the original input space, known as kernel trick (as summarized in 2.3.1). There are also some conditions on kernel functions to be satisfied for the kernel trick. It has to be symmetric and positive definite.

2.3.1. *The Kernel Trick.*

Theorem 2.4. *For any symmetric function $K : X \times X \to \mathbb{R}$ that satisfies the $\int f(x)K(x,x')f(x')dxdx' \geq 0$ for all $f \in L_2$, there exist functions $\Phi : X \to \mathbb{R}$ and $\lambda_i \geq 0$ such that $K(x,x') = \sum_i \lambda_i \Phi_i(x)\Phi_i(x')$ for all $x, x' \in X$.*

Therefore, for the kernel function that satisfies the symmetric and positive definite properties, the corresponding feature space can be used by calculating the $K(x^{(i)}, x^{(j)})$ in place of $< \Phi(x^{(i)}), \Phi(x^{(j)}) >$, even if the accurate equation for $\Phi()$ is unknown. Owing to this theorem, even infinite dimensional feature spaces can be used.

If the shape of the feature map can be predicted, trials and errors can be reduced by selecting and using an SVM kernel that is appropriate for the problem situation. Furthermore, in a situation that requires the creation of new features, an insight can be obtained by applying a different kernel of the SVM. One example of kernel functions for which the corresponding feature space has been revealed is a polynomial kernel. For example, if we consider the second-order polynomial kernel $k(x,y) =< x,y >^2$ for data with two variables, the corresponding feature map is $\Phi(x = (x_1,x_2)) \to (x_1^2, x_2^2, \sqrt{2}x_1x_2)$. And if we consider the rbf kernel; $k(x,y) = \exp((x-y)^2)$, the corresponding feature map is $\Phi(x = (x_1,x_2)) \to (x_1, x_2, x_1^2/2, x_2^2/2, \sqrt{2}x_1x_2/2,)$ because $\exp((x-y)^2) = \sum(k_d(x,y)/n!)$. Except for the above mentioned two kernels for which a specific feature map corresponds to the kernel function, several kernel functions described in 2.2.2 have not yet been analyzed. Many studies have been conducted on this subject [11, 12, 31]. In particular, Vedaldi, Andrea, Zisserman, and Andrew [11] analyzed which feature map the kernel corresponds if dataset has only one explanatory variable. For the stationary kernels and homogeneous kernel, they define the signature of kernel. Using this signature, they obtained the corresponding feature space. In fact, it can be applied to multiple variables cases by using addition and multiplication properties. However, from the case where the kernel is directly applied to multiple variables, different results are realized.

2.3.2. *Review of Kernels.*

Dot Product kernels

Linear (trivial) kernel is given by $k(x,y) =< x,y >$. This kernel function has the same meaning as the basic SVM. With regards to the linear kernel, many efforts have been made to shorten the time of training and inferencing parts of SVM, and it takes linear time with the number of training examples [71].

Polynomial kernel is given by $k(x, y) =< x, y >^{\alpha_1} + \alpha_2$. The order of the polynomial term is determined by the parameter α_1, and the feature space becomes α_1-th order terms. When $\alpha_1 = 1$, it perfectly matches the linear kernel. It is known to show particularly good performance when the data is normalized. It has been used in a study to classify different types of lymphoma [28], and demonstrated good performance in the termite detection problem [77]. In this study, we set a parameter α_2 as 0

Hyperbolinc tangent kernel is given by $k(x, y) = \tanh(< x, y >)$. This function is well known as the activation function. This function is conditionally positive definite. It has been used in a study to detect different types of lymphoma[72].

Vovks real polynomial, this kernel is given by $k(x, y) = \frac{1 - <x,y>^{\alpha_1}}{1 - <x,y>}$ [27]. This kernel can be written as $k(\xi) = \sum_{n=1}^{d} \xi^n$, hence all the coefficients equal to 1. This means that this kernel can be used regardless of the dimensionality of the input space. Similarly, we can analyze the an infinite power series.

Vovks infinite polynomial, this kernel is given by $k(x, y) = \frac{1}{1 - <x,y>}$ [27]. This kernel can be written as $k(\xi) = \sum_{n \geq 0} \xi^n$. hence, all the coefficients equal to 1. This suggests poor generalization properties.

Stationary Kernels

Rbf (gaussian) kernel is given by $k(x, y) = \exp(||x - y||^2 / 2\alpha_1{}^2)$. This corresponds to the infinite dimensional feature space. This kernel is known to show good performance in image classification problems such as hand-writing recognition. This kernel has been applied to classify different types of lymphoma [28], Arabic characters [29], and to predict coal price problem [36].

Laplacian kernel (or exponential kernel) is given by $k(x, y) = \exp(\frac{\|x-y\|}{\alpha_1})$. It is similar to the rbf kernel. It demonstrated good performance in classifying Arabic characters [29]. It has also been applied to problems related to object tracking [88].

Rational quadratic kernel (or cauchy kernel), this kernel is given by $k(x, y) = 1 - \frac{\|x-y\|^2}{\|x-y\|^2 + \alpha_1}$. It can be used as an alternative when the use of the Gaussian becomes significantly expensive. It is a long-tailed kernel and can be used to realize long-range influence and sensitivity over the high dimension space. It has been applied to the problem of classifying Arabic characters [29].

Multiquadric kernel, this kernel is given by $k(x, y) = \sqrt{\|x - y\|^2 + \alpha_1{}^2}$. It is summarized in the paper [92]. It has been applied to classify Arabic characters [29], and to detect images [73].

Inverse multiquadric kernel, this kernel is given by $k(x, y) = \frac{1}{\sqrt{\|x-y\|^2 + \alpha_1{}^2}}$. It is summarized in the paper [92]. It has also been applied in the classification of Arabic characters [29], and detection of images [73].

Circular kernel, this kernel is given by $k(x,y) = \frac{2}{\pi}\arccos(-T) - \frac{2}{\pi}T\sqrt{1-(T)^2}$ if $\|x-y\| < \alpha_1$, zero otherwise, where $T = \frac{\|x-y\|}{\alpha_1}$. This function is positive definite in R^2. The circular kernel is used in geostatic applications. It is summarized in [74]. In this study, it is applied to image recognition.

Spherical kernel, this kernel is given by $k(x,y) = 1 - \frac{3}{2}\frac{\|x-y\|}{\alpha_1} + \frac{1}{2}(\frac{\|x-y\|}{\alpha_1})^3$ if $\|x-y\| < \alpha_1$, zero otherwise. This function is positive definite in R^3. It is summarized in [74], where it is applied to image recognition.

Wave kernel, this kernel is given by $k(x,y) = \frac{\alpha_1}{\|x-y\|}\sin(\frac{\|x-y\|}{\alpha_1})$. It shows good performance in crude oil price data [75]. If $\|x-y\|$ equals to 0, we set $k(x,y)$ as 1.

Power kernel, this kernel is given by $k(x,y) = -\|x-y\|^{\alpha_1}$. It is an example of scale-invariant kernel [76] and it is only conditionally positive definite.

Log kernel, this kernel is given by $k(x,y) = -\log(\|x-y\|^{\alpha_1} + 1)$. The log kernel seems to be particularly appropriate for images. It has been applied to classify the domain of text [95].

generalized t-student kernel, this kernel is given by $k(x,y) = \frac{1}{1+\|x-y\|^{\alpha_1}}$

Other Kernels

ANOVA kernel is given by $k(x,y) = \sum(\exp(-\alpha_1(x-y)^2))^{\alpha_2}$. This kernel was suggested by [83]. It showed good performance in the classification of web pages [30] and the prediction of coal prices [36].

Spline kernel, this kernel is given by $k(x,y) = \prod_{i=1}^{d}[1+x_iy_i+x_iy_i\min(x_i,y_i) - \frac{x_i+y_i}{2}\min(x_i,y_i)^2 + \frac{\min(x_i,y_i)^3}{3}]$. This kernel is derived in [78].

Chi-square kernel, this kernel is given by $k(x,y) = 1 - \sum_{i=1}^{n}\frac{2x_iy_i}{(x_i+y_i)}$. It is has been applied for recognizing hand gestures [93].

Histogram intersection kernel, this kernel is given by $k(x,y) = \sum_{i=1}^{n}\min(x_i,y_i)$. The histogram intersection kernel is also known as the min kernel, and it has been proven useful in image classification [79].

Hellingerss kernel, this kernel is given by $k(x,y) = \sum_{i=1}^{n}\sqrt{x_iy_i}$. IIt has been applied to retrieve images [94].

There are other kernels such as **B-spline kernel** [91], **Bessel kernel** [90], **generalized histogram intersection kernel** [79], **Bayesian kernel** [35], **wavelet kernel** [89].

3. Experiment

3.1. Experimental Design

We planned this experiment to investigate which kernel functions are appropriate in various situations. Therefore, we measured the score of each kernel for specific data and summarized the results. For this experiment, the following was considered: sufficiently diverse benchmark dataset, possible parameters for each kernel, reasonable score for evaluation, and selection of diagrams to effectively show the results of the experiment. Finally, the issues that occurred during this experiment are summarized in this section. This experiment was designed to enable 100% re-implementation and was completely shared in the GitHub repository(https://github.com/000namc/python-sklearn-svm-kernels).

3.1.1. *Benchmark dataset.* Many studies have used the UCI dataset [14] to verify the performance of the proposed model [31, 32, 33, 34]. Among them, we selected five problems related to classification in the UCI dataset (breast cancer [13], yeast [20], segmentation, waveform [21], and leaf [15]) and five problems related to regression (wine [22], crime [16], airfoil [17], fire [18], and fish [19]). We choose a dataset which has adequate number of examples and columns so that it generally represents datasets as much as possible. Nevertheless, in the experiment, if a dataset has a lot of examples, the model training speed decreases sharply. Therefore, we used datasets with a smaller number of examples. Furthermore, a basic feature engineering such as one-hot-encoding was applied when necessary. All explanatory variables were normalized, and some meaningless columns were removed. Finally, the target variable of the regression dataset was also normalized in order to compare the results within the same range. All the codes for this preprocessing procedure are shared in the GitHub repository to enable re-implementation as mentioned above. The following Table 1 outlines the characteristics of each dataset, including the purpose of the data, the number of examples, the number of features, and the number of classes.

3.1.2. *Model parameter.* We aim to compare the results of each kernels. However, even with one kernel, different results can be obtained when the corresponding kernel parameters are changed. Therefore, we regard the highest score obtained with a varying parameter for a specific kernel as the score of that kernel. It is also impossible to investigate all possibilities because of the limited computing resources. Therefore, instead of using all the parameters, we selected one representative parameter that causes the largest change in a given kernel function, and the results were obtained while varying this parameter. The parameter that had the largest meaning is α_1. In the above mentioned definitions of the kernel functions, the parameter that has the largest meaning for each kernel is indicated as α_1. The following parameters for each kernel were used: linear: $[0]$, polynomial: $[2, 5, 8]$, hyperbolic tangent: $[0]$, Vovks real polynomial: $[2, 5, 8]$, Vovks infinite polynomial: $[0]$, Gaussian: $[0.5, 1, 5]$, Laplacian: $[0.5, 1, 5]$, rational quadratic: $[1, 10, 100]$, multi-quadratic: $[1, 5, 10]$, inverse multi-quadratic: $[1, 5, 10]$, circular: $[0.5, 1, 5]$, spherical: $[0.5, 1, 5]$, wave: $[0.5, 1, 5]$, power: $[2, 2.2, 2.5]$, log: $[2, 5, 8]$, generalized t-student: $[2, 5, 8]$, ANOVA: $[0.5, 1, 5]$, spline: $[0]$, chi square: $[0]$, histogram intersection: $[0]$, and Hellingers: $[0]$.

Data name	Purpose	Number of examples	Number of features	Number of labels	Type
Breast Cancer	Predicting breast cancer (whether it is benign or malignant)	683	9	2	Classification
Yeast	Predicting the cellular localization sites of proteins	1484	8	10	Classification
Segmenta tion	The instances were drawn randomly from a database of 7 outdoor images. Classifying the pictures	210	19	7	Classification
Waveform	Each class is generated from a combination of 2 of 3 "base" waves. Classifying the waves.	5000	21	3	Classification
Leaf	Predicting the species for given texture feature of the leaf	340	15	30	Classification
Wine	Predicting wine quality based on physicochemical tests	1599	11	-	Regression
Crime	Predicting Total number of violent crimes per 100K population	1994	100	-	Regression
Airfoil	Predicting of Sound Pressure Level in Various Experiments	1503	5	-	Regression
Fire	Predicting of burnt areas in case of fire	517	29	-	Regression
Fish	Predicting quantitatively acute aquatic toxicity towards the fish; Pimephales promelas	908	6	-	Regression

TABLE 1. data description

3.1.3. *Model validation.* To obtain reliable results, we performed a 10-fold-cross-validation for each case. In this process, we recorded two different measures: *accuracy* and the *standard deviation* of the accuracy in a 10-fold training set and the test set. For the classification problem, we used Accuracy Score as accuracy, and for the regression problem, we used $(3 - \text{RMSE})^+$, where $(x)^+ = \max(0, x)$, as accuracy. In other words, if RMSE exceeds three, the score is insignificant, and we regarded $(3 - \text{RMSE})^+ = 0$ as the worst case. We want to say that it is not enough to see only the accuracy to evaluate each case. It is also not a good result when the deviation is high no matter the accuracy. Therefore, we used a score of a slightly new perspective based on the cosine law of triangle ($c^2 = a^2 + b^2 - 2ab\cos\phi$): cosine-score with two different measures related by performance and deviation of the 10-fold.

$$\text{cosine-score}^2_{\text{pattern}} = A^2 + B^2 - 2 \cdot A \cdot B \cdot R$$

Where,

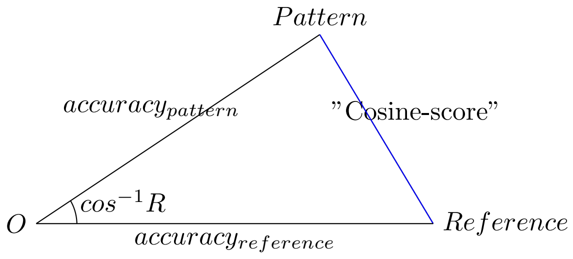

FIGURE 2. The cosine-score

$$A = \text{accuracy}_{\text{reference}}$$
$$B = \text{accuracy}_{\text{pattern}}$$
$$R = 1 - 1/k \times \min(\text{standard deviation}_{\text{pattern}}, k)$$

To understand the cosine-score, we will first consider the score of each model as a point consisting of an ordered pair (accuracy, R). R satisfies the condition $0 \le R \le 1$. If $R = \cos\pi$, it satisfies $0 \le \phi = \cos^{-1} R \le \frac{\pi}{2}$. In this equation, the reference point represents the highest possible score. The pattern point represents score for some data. cosine-score enables the evaluation of the model by simultaneously considering these pair. In other words, a good score is not given even if one measure is high. A constant k is used as a weight parameter between these pair: a higher value of k indicates a higher ratio of the accuracy in the cosine-score. In our experiment, $k = 0.2$ was used. In addition, the reference point was set to $(1, 1)$ and $(3, 1)$ for classification and regression problem. For accuracy, 100% is the perfect score, for $(3 - \text{RMSE})^+$, 3 is the perfect score, and for R, 1 is the perfect score. Based on this analysis, the scores can be represented as follows Fig. 2.

3.1.4. *Result Visualization.* The Taylor diagram provides a method of graphically summarizing the resemblance between a pattern (or a set of patterns) and the observations(or a reference) [23]. One characteristic of this graph is that it coordinates the two measures representing the graph and the distance to the reference point. This can be observed at a glance. Furthermore, unlike the data representation in the orthogonal coordinate system, the region close to the reference point is enlarged, thereby facilitating observation. Originally in the proposed study, the similarity between two patterns is quantified in terms of their correlation, the difference between their root-mean-square and the amplitude of their variations (represented by their standard deviations). However, in this study, we measured the similarity by using the average of the 10-fold accuracy and the R derived by standard deviation of 10-fold accuracy. we set a reference point positioned optimally in both measures: accuracy and variation.

We acquired a contour graph from it; hence, we had a good visualization of our proposed score.

We proposed another graph Top-N Counter we named. This diagram counts the number of experiments which recorded the top rank for each candidate. Here, the various datasets were the experiments, and the kernels candidates. The character N denotes the criterion to be in the top rank. For example, if $N = 5$, this counts the number of experiments to record in the top five ranks. Adequate value for N should be chosen depending on the given problem. In our case, as there were 20 candidates, we choose $N = 10$, and to draw this diagram, similar to the Taylor diagram, we considered the cosine-score.

3.1.5. *Used code.* Python is a good language for getting and sharing information because it is used by many people and many references exist. We also planned this experiment using the Python language. For data preprocessing, we used the pandas package [25], and for the SVM model, we used the scikit-learn package [24], which implemented the SVM based on the libsvm. The function provided by the scikit-learnis designed to use linear kernel, polynomial kernel, or Gaussian kernel by option. To use other kernel functions, we should calculate and input the gram matrix of the kernel that satisfies some conditions. The gram-matrix was used to store the kernel value of each example and it was defined as $G(i, j) = K(x^{(i)}, x^{(j)})$. Therefore, the process of calculating the gram matrix of each kernel is included in our experiment process in order to use various kernels that we have considered. Furthermore, with the Taylor diagram for effective visualization, we used a revised code shared in GitHub by Yannick Copin [26]. All the Python codes for this experiment using the above packages are shared in the GitHub repository that we provided to allow re-implementation.

3.1.6. *Implementation issue.* There were several issues with our experiments. First, the implementation of SVM, using the scikit-learn package considers max iteration, and error toleration as stopping criteria. It also takes a long time to satisfy the default stopping criterion in the experiment for some data with regression problems. To solve this problem, the max iteration was set to 10000. If it does not converge even under this condition, the lowest score was given, considering that this problem cannot be generalized using the kernel and parameter. The second problem is overflow error caused by a numerous or undefined calculations such as division by zero in the gram-matrix calculation process. The lowest score was given to this case as well. Thirdly, the data corresponding to $R = 0$ was excluded from performance comparison because this reveals the extreme poor performance in the definitions of R and accuracy. Fourth, the result is dependent on how to divide the 10-fold because our cosine-score is obtained by 10-fold-cross-validation.

3.2. Experiment results

We performed 10-cross-validation for each given dataset, kernel, and corresponding parameter. The results of our experiment could be listed in a table having about 600 rows for the number of datasets, the number of kernels, and the number of parameters of each kernel (the complete table can be obtained from in the GitHub repository). These results are summarized in Taylor diagrams and Top-10 Counter for classification and regression problems.

3.2.1. *Classification Problem.* In the Taylor diagram Fig. 3, the radius axis and the arc axis represent the Accuracy Score and R, respectively. The first notable result is that the cosine-score difference of the kernels is significant in every dataset as expected before designing this experiment. Each dataset had different sets of fitted kernels. The kernels that showed good scores were power kernel, rational quadratic kernel, and Laplacian kernel (breast cancer dataset); generalized t-student kernel, inverse multi-quadratic kernel, and Vovks real polynomial kernel (yeast dataset); gaussian kernel, histogram intersection kernel, and linear kernel (segmentation dataset); ANOVA kernel, Laplacian kernel, and log kernel (waveform dataset); log kernel, histogram intersection kernel, and power kernel (leaf dataset). Another finding was that the fitting level was different for each dataset. Based on the cosine-score, the maximum score for the waveform dataset was 0.271, 0.326 for the breast cancer dataset, 0.52 for the leaf dataset, 0.551 for the yeast dataset, and 0.614 for the segmentation dataset. Thus, the scores showed significant differences among the datasets. To examine them in detail, the multi-quadratic kernel, chi square kernel, and Vovks infinite polynomial kernel were not given good scores in general. Furthermore, unlike other datasets, the leaf dataset had significantly different scores for each kernel.

3.2.2. *Regression Problem.* In the Taylor diagram Fig. 4, the horizontal axis and the arc axis represent $(3 - \text{RMSE})^+$and R, respectively. As mentioned in the explanation of the datasets, each target value was normalized. Thus, regarding $(3-\text{RMSE})^+$, the graph represents scores from 3 points. This means fitting all data to point 0 for poor performance. In regression problem too, the first notable result was that the difference in cosine score of the kernels in every dataset is significant as expected before designing this experiment. The kernels that gave the best cosine-score for each dataset were log kernel, inverse multi-quadratic kernel, and ANOVA kernel (wine dataset); Laplacian kernel, histogram intersection kernel, and rational quadratic kernel (crime dataset); and ANOVA kernel, rational quadratic kernel, and Laplacian kernel (airfoil dataset). As an exception, the fire dataset did not show good scores in all kernels, and based on the simple RMSE, the histogram intersection kernel, gaussian kernel, and rational quadratic kernels showed good scores. In the case of the fish dataset, wave kernel, rational quadratic kernel, and Laplacian kernel showed good scores. Furthermore, the best score in each dataset was 1.926 for

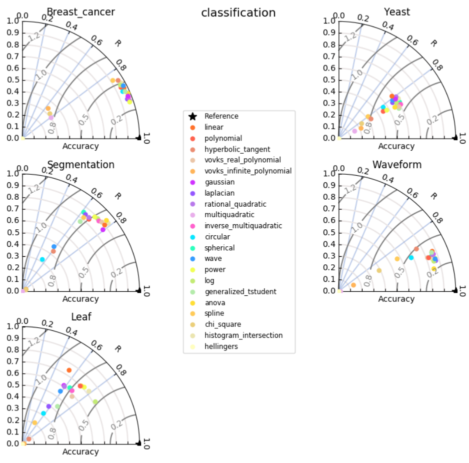

FIGURE 3. The Taylor diagram for the classification problem

fish, 1.830 for wine, 1.630 for crime, and 1.211 for airfoil. Another observation was that, the polynomial kernel always produced poor results. In the case of the fire dataset, it could be impossible to fit in the SVM model to solve this problem because the score was very poor and the reliability of the score was not very good either. The results showed a tendency of wider diffusion in the R axis than that of the classification problem, which could be interpreted that it is simply due to the different range of the accuracy.

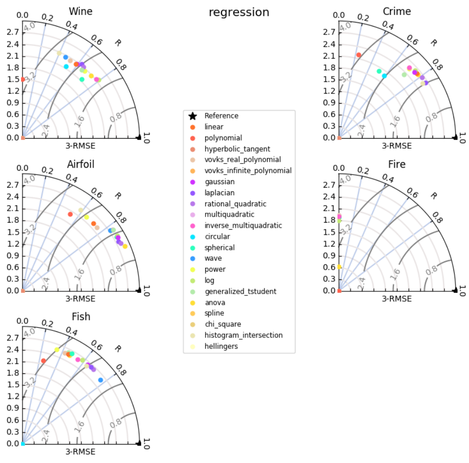

FIGURE 4. The Taylor diagram for the regression problem

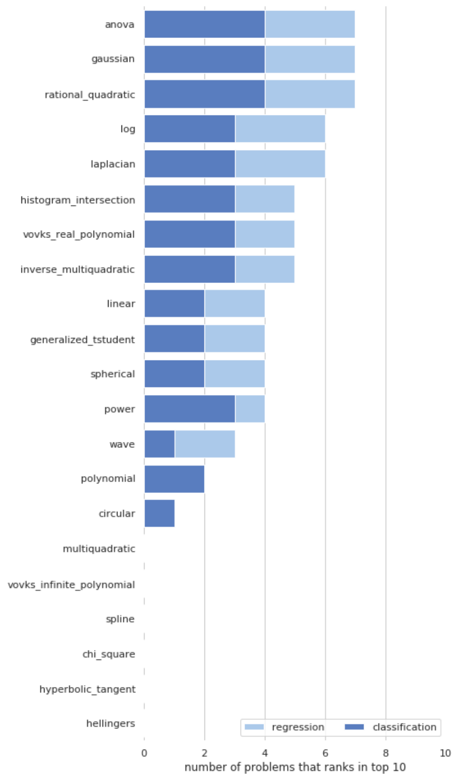

FIGURE 5. The Top-10 Counter

3.2.3. *Top-10 Counter.* In the following graph Fig. 5, the y-axis represents the kernel function, and the x-axis represents the number of datasets of which the corresponding kernel in the top 10 ranking. The first fact that it can be seen provides the differences between kernels. The chi square, spline, multi-quadratic, Vovks infinite polynomial, hyperbolic tangent, and Hellingers kernels have never been included in top 10 ranking for both classification and regression. Furthermore, the ANOVA kernel, gaussian kernel, rational quadratic kernel, log kernel, Laplacian kernel, histogram intersection kernel, Vovks real polynomial kernel, and inverse multi-quadratic kernel show relatively high scores. There was also no kernel function that was included in top 10 ranking for every dataset. In other words, there exists no kernel function that can represent every dataset well. We want to remark that among the basic kernel functions provided by the Python scikit-learn package, the linear kernel and polynomial kernel did not show good scores. The polynomial kernel has never been in top 10 ranking in the regression problem.

4. Discussion

The results of this study showed that it was effective to employ kernel functions for general datasets, such as ANOVA kernel, gaussian kernel, rational quadratic kernel, log kernel, Laplacian kernel, histogram intersection kernel, Vovks real polynomial kernel, and inverse multi-quadratic kernel. Therefore, we believe that the built in kernel functions in various machine learning packages including the scikit-learn of Python should be improved. Our results are based on the cosine-score, related to accuracy and the standard deviation of the 10-fold-validation results. Results with different weights of the two measures can be obtained through an adjustment of the value of parameter k.
We expect that continued research will be conducted on the kernel functions of SVM and their corresponding feature spaces. Methods to better utilize existing kernel functions and innovative new kernels should be developed continuously. For several kernels that reviewed in this study, the feature spaces represented by the kernel functions have not been revealed thus far. If these feature spaces are revealed, the existing kernels can be classified in new ways using common characteristics and insight on utilizing them can also be obtained. In addition, proposals for new kernels will be performed more systematically.
Continued efforts to shorten the running time of SVM are also required. In our experiment, the number of data were limited to 5000. Even with this number, the model training required a significant amount of time (it required about one day when using a 20-core CPU and 64GB memory to complete our experiment). This is still insufficient, considering that the size of dataset used in a general machine learning model can be as many as one million rows.

Our study requires further improvements in several aspects. First, additional parameters of the kernel functions need to be examined. Other model parameters as well as the regularization parameter should be adjusted. We could not investigate several possibilities due to the limited resources. However, this will be possible when the training speed of SVM becomes faster in the future. Second, additional candidates for the kernel functions should be collected and organized. Even though newly proposed kernel functions exist, newer possibilities need to be investigated. Third, the shared Python code needs to be optimized. It still has many parts that can be improved. In particular, the calculation of the gram matrix needs to be improved first. With regards to that, we always welcome pull requests for our code shared in GitHub.

References

[1] Boser, Bernhard E and Guyon, Isabelle M and Vapnik, Vladimir N, *A training algorithm for optimal margin classifiers*, Proceedings of the fifth annual workshop on Computational learning theory, ACM (1992), 144–152.

[2] Smola, *A tutorial on support vector regression*, Statistics and computing **14**, Springer (2004), no.3 199–222.

[3] Burges, Christopher JC, *A tutorial on support vector machines for pattern recognition*, Data mining and knowledge discovery **2**, Springer (1998), no.2 121–167.

[4] Guyon, I, *Svm application list*, URL http://www.clopinet.com/isabelle/Projects/SVM /applist.html, (1999).

[5] Wang, Guosheng, *A survey on training algorithms for support vector machine classifiers*, Fourth International Conference on Networked Computing and Advanced Information Management **1**, IEEE (2008), 123–128.

[6] Souza, Cesar R, *Kernel functions for machine learning applications*, Creative Commons Attribution-Noncommercial-Share Alike **3**, (2010), 29.

[7] Shawe-Taylor, John and Sun, Shiliang, *A review of optimization methodologies in support vector machines*, Neurocomputing **74**, Elsevier (2011), no.17 3609–3618.

[8] Nayak, Janmenjoy and Naik, Bighnaraj and Behera, H, *A comprehensive survey on support vector machine in data mining tasks: applications & challenges*, International Journal of Database Theory and Application **8**, (2015), no.1 169–186.

[9] Campbell, Colin, *Kernel methods: a survey of current techniques*, Neurocomputing **48**, Elsevier (2002), no.1-4 63–84.

[10] Smola, Alex J and Scholkopf, Bernhard, *Learning with kernels*, **4**, Citeseer (1998).

[11] Vedaldi, Andrea and Zisserman, Andrew, *Efficient additive kernels via explicit feature maps*, IEEE transactions on pattern analysis and machine intelligence **34**, IEEE (2012), no.3 480–492.

[12] Rahimi, Ali and Recht, Benjamin, *Random features for large-scale kernel machines*, Advances in neural information processing systems, (2008), 1177–1184.

[13] Mangasarian, Olvi L and Wolberg, William H, *Cancer diagnosis via linear programming*, University of Wisconsin-Madison Department of Computer Sciences, (1990).

[14] Dua, Dheeru and Graff, Casey, *UCI Machine Learning Repository*, University of California, Irvine, School of Information and Computer Sciences, (2017).

[15] Silva, Pedro FB and Marcal, Andre RS and da Silva, Rubim M Almeida, *Evaluation of features for leaf discrimination*, International Conference Image Analysis and Recognition, Springer (2013), 197–204.

[16] Redmond, Michael and Baveja, Alok, *A data-driven software tool for enabling cooperative information sharing among police departments*, European Journal of Operational Research **141**, Elsevier (2002), no.3 660–678.

[17] Brooks, Thomas F and Pope, D Stuart and Marcolini, Michael A, *Airfoil self-noise and prediction*, (1989).

[18] Cortez, Paulo and Morais, Aníbal de Jesus Raimundo, *A data mining approach to predict forest fires using meteorological data*, Data mining and knowledge discovery, APPIA (2007).

[19] Cassotti, M and Ballabio, D and Todeschini, R and Consonni, V, *A similarity-based QSAR model for predicting acute toxicity towards the fathead minnow (Pimephales promelas)*, SAR and QSAR in Environmental Research **26**, Taylor & Francis (2015), no.3 217–243.

[20] Nakai, Kenta and Kanehisa, Minoru, *Expert system for predicting protein localization sites in gram-negative bacteria*, Proteins: Structure, Function, and Bioinformatics **11**, Springer (1991), no.2 95–110.

[21] Breiman, Leo, *Classification and regression trees*, Routledge (2017).

[22] Cortez, Paulo and Cerdeira, Antonio and Almeida, Fernando and Matos, Telmo and Reis, Jose, *Modeling wine preferences by data mining from physicochemical properties*, Decision Support Systems **47**, Elsevier (1998), no.4 547–553.

[23] Taylor, Karl E, *Summarizing multiple aspects of model performance in a single diagram*, Data mining and knowledge discovery **106**, Journal of Geophysical Research: Atmospheres (1998), no.D7 7183–7192.

[24] Pedregosa, Fabian and Varoquaux, Gaël and Gramfort, Alexandre and Michel, Vincent and Thirion, Bertrand and Grisel, Olivier and Blondel, Mathieu and Prettenhofer, Peter and Weiss, Ron and Dubourg, Vincent and others, *Scikit-learn: Machine learning in Python*, Journal of machine learning research **12**, (2011), 2825–2830

[25] McKinney, Wes and others, *Data structures for statistical computing in python*, Proceedings of the 9th Python in Science Conference **445**, Austin, TX (2010), 51–56.

[26] Yannick Copin, *taylor diagram python code*, URL https://gist.github.com/ycopin/3342888, (2018).

[27] Saunders, Craig and Stitson, Mark O and Weston, Jason and Bottou, Leon and Smola, A and others, *Support vector machine-reference manual, Technical Report*, Department of Computer Science, Royal Holloway, University of London, Egham, UK, (1998).

[28] Valentini, Giorgio, *Gene expression data analysis of human lymphoma using support vector machines and output coding ensembles*, Artificial Intelligence in Medicine **26**, Elsevier (2002), no.3 281–304.

[29] Fadel, Sayed and Ghoniemy, Said and Abdallah, Mohamed and Sorra, Hussein Abu and Ashour, Amira and Ansary, Asif, *Investigating the effect of different kernel functions on the performance of SVM for recognizing Arabic characters*, International Journal of Advanced Computer Science and Applications **7**, Citeseer (2016), no.1 446–450.

[30] Chen, Rung-Ching and Hsieh, Chung-Hsun, *Web page classification based on a support vector machine using a weighted vote schema*, Expert Systems with Applications **31**, Elsevier (2006), no.2 427–435.

[31] Kar, Purushottam and Karnick, Harish, *Random feature maps for dot product kernels*, Artificial Intelligence and Statistics (2012), 583–591.

[32] Deng, Wan-Yu and Ong, Yew-Soon and Zheng, Qing-Hua, *A fast reduced kernel extreme learning machine*, Neural Networks **76**, Elsevier (2016), 29–38.

[33] Wang, Benjamin X and Japkowicz, Nathalie, *Boosting support vector machines for imbalanced data sets*, Knowledge and information systems **25**, Springer (2010), no.1 1–20.

[34] Caruana, Rich and Niculescu-Mizil, Alexandru, *An empirical comparison of supervised learning algorithms*, Proceedings of the 23rd international conference on Machine learning, ACM (2006), 161–168.

[35] Alashwal, Hany and Deris, Safaai and Othman, Razib M, *A Bayesian kernel for the Prediction of Protein-Protein Interactions*, World Academy of Science, Engineering and Technology **51**, (2009), 928–933.

[36] BONITA, OLIVIA and MUFLIKHAH, LAILIL, *Comparison of Gaussian and ANOVA Kernel in Support Vector Regression for Predicting Coal Price*, 2018 International Conference on Sustainable Information Engineering and Technology (SIET), IEEE (2018), 147–150.

[37] Gish, Herbert, *A probabilistic approach to the understanding and training of neural network classifiers*, International Conference on Acoustics, Speech, and Signal Processing, IEEE (1990), 1361–1364.

[38] Zhang, Guoqiang Peter, *Neural networks for classification: a survey*, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) **30**, IEEE (2000), no.4 451–462.

[39] Adya, Monica and Collopy, Fred, *How effective are neural networks at forecasting and prediction? A review and evaluation*, Journal of forecasting **17**, Wiley Online Library (1998), no.5-6 481–495.

[40] Callen, Jeffrey L and Kwan, Clarence CY and Yip, Patrick CY and Yuan, Yufei, *Neural network forecasting of quarterly accounting earnings*, International Journal of Forecasting **12**, Elsevier (1996), no.4 475–482.

[41] Church, Keith B and Curram, Stephen P, *Forecasting consumers' expenditure: A comparison between econometric and neural network models*, International journal of forecasting **12**, Elsevier (1996), no.2 255–267.

[42] Connor, Jerome T and Martin, R Douglas and Atlas, Les E, *Recurrent neural networks and robust time series prediction*, IEEE transactions on neural networks **5**, IEEE (1994), no.2 240–254.

[43] Cottrell, Marie and Girard, Bernard and Girard, Yvonne and Mangeas, Morgan and Muller, Corinne, *Neural modeling for time series: a statistical stepwise method for weight elimination*, IEEE transactions on neural networks **6**, IEEE (1995), no.6 1355–1364.

[44] Faraway, Julian and Chatfield, Chris, *Time series forecasting with neural networks: a comparative study using the air line data*, Journal of the Royal Statistical Society: Series C (Applied Statistics) **47**, Wiley Online Library (1998), no.2 231–250.

[45] Fletcher, Desmond and Goss, Ernie, *Forecasting with neural networks: an application using bankruptcy data*, Information & Management **24**, Elsevier (1993), no.3 159–167.

[46] Gorr, Wilpen L, *Research prospective on neural network forecasting*, International Journal of Forecasting **10**, Elsevier (1994), no.1 1–4.

[47] Hippert, Henrique Steinherz and Pedreira, Carlos Eduardo and Souza, Reinaldo Castro, *Neural networks for short-term load forecasting: A review and evaluation*, IEEE Transactions on power systems **16**, IEEE (2001), no.1 44–55.

[48] Belli, MR and Conti, Massimo and Crippa, Paolo and Turchetti, Claudio, *Artificial neural networks as approximators of stochastic processes*, Neural Networks **12**, Elsevier (1999), no.4-5 647–658.

[49] Castro, Juan Luis and Mantas, Carlos Javier and Benitez, JM, *Neural networks with a continuous squashing function in the output are universal approximators*, Neural Networks **13**, Elsevier (2000), no.6 561–563.

[50] Funahashi, Ken-Ichi, *On the approximate realization of continuous mappings by neural networks*, Neural networks **2**, Elsevier (1989), no.3 183–192.

[51] Andrews, Robert and Diederich, Joachim and Tickle, Alan B, *Survey and critique of techniques for extracting rules from trained artificial neural networks*, Knowledge-based systems **8**, Elsevier (1995), no.6 373–389.

[52] Castro, Juan L and Mantas, Carlos J and Benítez, José Manuel, *Interpretation of artificial neural networks by means of fuzzy rules*, IEEE Transactions on Neural Networks **13**, IEEE (2002), no.1 101–116.

[53] Setiono, Rudy and Leow, Wee Kheng and Zurada, Jacek M, *Extraction of rules from artificial neural networks for nonlinear regression*, IEEE transactions on neural networks **13**, IEEE (2002), no.3 564–577.

[54] Setiono, Rudy and Thong, James YL, *An approach to generate rules from neural networks for regression problems*, European Journal of Operational Research **155**, Elsevier (2004), no.1 239–250.

[55] Lisboa, Paulo JG, *A review of evidence of health benefit from artificial neural networks in medical intervention*, Neural networks **15**, Elsevier (2002), no.1 11–39.

[56] Portney, Leslie Gross and Watkins, Mary P and others, *Foundations of clinical research: applications to practice*, Pearson/Prentice Hall Upper Saddle River, NJ **892**, (2009).

[57] Shawe-Taylor, John and Bartlett, Peter L and Williamson, Robert C and Anthony, Martin, *Structural risk minimization over data-dependent hierarchies*, IEEE transactions on Information Theory **44**, IEEE (1998), no.5 1926–1940.

[58] Vapnik, Vladimir, *Estimation of dependences based on empirical data*, Springer Science & Business Media, (2006).

[59] McCulloch, Warren S and Pitts, Walter, *A logical calculus of the ideas immanent in nervous activity*, The bulletin of mathematical biophysics **5**, Springer (1943), no.4 115–133.

[60] McClelland, James L and Rumelhart, David E and PDP Research Group and others, *Parallel distributed processing*, MIT press Cambridge, MA: **2**, (1987).

[61] Dietterich, Thomas G, *Ensemble methods in machine learning*, International workshop on multiple classifier systems, Springer (2000), 1–15.

[62] Rokach, Lior and Maimon, Oded, *Feature set decomposition for decision trees*, Intelligent Data Analysis **9**, IOS Press (1998), no.2 131–158.

[63] Kuncheva, Ludmila I and Whitaker, Christopher J, *Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy*, Machine learning **51**, Springer (2003), no.2 181–207.

[64] Sollich, Peter and Krogh, Anders, *Learning with ensembles: How overfitting can be useful*, Advances in neural information processing systems, (1996), 190–196.

[65] Brown, Gavin and Wyatt, Jeremy and Harris, Rachel and Yao, Xin, *Diversity creation methods: a survey and categorisation*, Information Fusion **6**, Elsevier (2005), no.1 5–20.

[66] Adeva, Juan Jose Garcia and Beresi, U and Calvo, R, *Accuracy and diversity in ensembles of text categorisers*, CLEI Electronic Journal **9**, (2005), no.1 1–12.

[67] Krogh, Anders and Vedelsby, Jesper, *Neural network ensembles, cross validation, and active learning*, Advances in neural information processing systems, (1995) 231–238.

[68] Belkin, Mikhail and Hsu, Daniel and Ma, Siyuan and Mandal, Soumik, *Reconciling modern machine-learning practice and the classical bias–variance trade-off*, Proceedings of the National Academy of Sciences **116**, National Acad Sciences (2019), no.32 15849–15854.

[69] Nakkiran, Preetum and Kaplun, Gal and Bansal, Yamini and Yang, Tristan and Barak, Boaz and Sutskever, Ilya, *Deep double descent: Where bigger models and more data hurt*, arXiv preprint arXiv:1912.02292, (2019).

[70] Cybenko, George, *Approximation by superpositions of a sigmoidal function*, Mathematics of control, signals and systems **2**, Springer (1989), no.4 303–314.

[71] Joachims, Thorsten, *Training linear SVMs in linear time*, Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM (2006), 217–226.

[72] Boughorbel, Sabri and Tarel, J-P and Boujemaa, Nozha, *Conditionally positive definite kernels for svm based image recognition*, 2005 IEEE International Conference on Multimedia and Expo, IEEE (2005), 113–116.

[73] Nasrabadi, Nasser M and Kwon, Heesung, *Kernel spectral matched filter for hyperspectral target detection*, Proceedings.(ICASSP'05). IEEE International Conference on Acoustics, Speech, and Signal Processing **4**, IEEE (2005), iv–665.

[74] Boughorbel, Sabri and Tarel, Jean-Philippe and Fleuret, Francois and Boujemaa, Nozha, *The GCS kernel for SVM-based image recognition*, International Conference on Artificial Neural Networks, Springer (2005), 595–600.

[75] Chiroma, Haruna and Abdulkareem, Sameem and Abubakar, Adamu I and Herawan, Tutut, *Kernel functions for the support vector machine: comparing performances on crude oil price data*, Recent Advances on Soft Computing and Data Mining, Springer (2014), 273–281.

[76] Fleuret, Francois and Sahbi, Hichem, *Scale-invariance of support vector machines based on the triangular kernel*, 3rd International Workshop on Statistical and Computational Theories of Vision, (2003), 1–13.

[77] Achirul Nanda, Muhammad and Boro Seminar, Kudang and Nandika, Dodi and Maddu, Akhiruddin, *A comparison study of kernel functions in the support vector machine and its application for termite detection*, Information **9**, Multidisciplinary Digital Publishing Institute (2018), no.1.

[78] Gunn, Steve R and others, *Support vector machines for classification and regression*, ISIS technical report **14**, University of Southampton (1998), no.1 5–16.

[79] Maji, Subhransu and Berg, Alexander C and Malik, Jitendra, *Efficient classification for additive kernel SVMs*, IEEE transactions on pattern analysis and machine intelligence **35**, IEEE (2012), no.1 66–77.

[80] Vanek, Jan and Michálek, Josef and Psutka, Josef, *A Comparison of Support Vector Machines Training GPU-Accelerated Open Source Implementations*, arXiv preprint arXiv:1707.06470, (2017).

[81] Afifi, Shereen Moataz and GholamHosseini, Hamid and Poopak, S, *Hardware implementations of SVM on FPGA: A state-of-the-art review of current practice*, International Journal of Innovative Science Engineering and Technology (IJISET), (2015).

[82] Burges, Christopher JC, *Speed up SVM algorithm for massive classification tasks*, International conference on advanced data mining and applications, Springer (2008), 147–157.

[83] Burges, CJC and Vapnik, V, *A new method for constructing artificial neural networks*, Interim technical report, ONR contract, (1995).

[84] Abiodun, Oludare Isaac and Jantan, Aman and Omolara, Abiodun Esther and Dada, Kemi Victoria and Mohamed, Nachaat AbdElatif and Arshad, Humaira, *State-of-the-art in artificial neural network applications: A survey*, Heliyon **4**, Elsevier (2018), no.11.

[85] Saravanan, Kl and Sasithra, S, *Review on classification based on artificial neural networks*, International Journal of Ambient Systems and Applications (IJASA) **2**, (2014), no.4 11–18.

[86] Martinez-Porchas, Marcel and Villalpando-Canchola, Enrique and Vargas-Albores, Francisco, *Significant loss of sensitivity and specificity in the taxonomic classification occurs when short 16S rRNA gene sequences are used*, Heliyon **2**, Elsevier (2016), no.9.

[87] Abid, Faroudja and Hamami, Latifa, *A survey of neural network based automated systems for human chromosome classification*, Artificial Intelligence Review **49**, Springer (2018), no.1 41–56.

[88] Wang, Yaohui and Zhang, Jiyang, *Application of SVM in object tracking based on Laplacian kernel function*, 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC) **2**, IEEE (2016), 557–561.

[89] Zhang, Li and Zhou, Weida and Jiao, Licheng, *Wavelet support vector machine*, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **34**, IEEE (2004), no.1 34–39.

[90] Xiang, Li and Quanyin, Zhu and Liuyang, Wang, *Research of bessel kernel function of the first kind for support vector regression*, Information Technology Journal **12**, ANSINET (2013), no.14 2673–2682.

[91] Horvath, Gabor, *CMAC neural network as an SVM with B-Spline kernel functions*, Proceedings of the 20th IEEE Instrumentation Technology Conference (Cat. No. 03CH37412) **2**, IEEE (2003), 1108–1113.

[92] Aftab, Wasim and Moinuddin, Muhammad and Shaikh, Muhammad Shafique, *A novel kernel for RBF based neural networks*, Abstract and Applied Analysis, Hindawi (2014).

[93] Abadi, Wassila and Fezari, Mohamed and Hamdi, Rachid, *Bag of Visualwords and Chi-Squared Kernel Support Vector Machine: A Way to Improve Hand Gesture Recognition*, Proceedings of the International Conference on Intelligent Information Processing, Security and Advanced Communication, ACM (2015).

[94] Rao, Swathi, *Effects of Image Retrieval from Image Database using Linear Kernel and Hellinger Kernel Mapping of SVM*, International Journal of Scientific & Engineering Research **4**, no.5.

[95] Roul, *A Modified Cosine-Similarity based Log Kernel for Support Vector Machines in the Domain of Text Classification*, Proceedings of the 14th International Conference on Natural Language Processing, (2017), 338–347.

Seong-Uk Nam
Department of Mathematics, Pusan National University, Busan, Korea
E-mail address: `000namc@gmail.com`

Sangil Kim
Department of Mathematics, Pusan National University, Busan, Korea
E-mail address: `sangil.kim@pusan.ac.kr`

HyunMin Kim
Department of Mathematics, Pusan National University, Busan, Korea
E-mail address: `hynmin@pusan.ac.kr`

YongBin Yu
Department of Machine Learning Engineering, Silex, Seoul, Korea
E-mail address: `yongbin.yu@gmail.com`