electronics

MDPI

Article

PFW: Polygonal Fuzzy Weighted—An SVM Kernel for the Classification of Overlapping Data Groups

Saman Shojae Chaeikar [1], Azizah Abdul Manaf [2], Ala Abdulsalam Alarood [2] and Mazdak Zamani [3,*]

1 Faculty of Computer Engineering, K. N. Toosi University of Technology, Tehran 1631714191, Iran; sschaeikar@mail.kntu.ac.ir
2 College of Computer Science and Engineering, University of Jeddah, Jeddah 21959, Saudi Arabia; aaabdmanaf@uj.edu.sa (A.A.M.); aasoleman@uj.edu.sa (A.A.A.)
3 School of Arts and Sciences, Felician University, Lodi, NJ 07070, USA
* Correspondence: zamanim@felician.edu

Received: 17 February 2020; Accepted: 1 April 2020; Published: 5 April 2020

Abstract: Support vector machines are supervised learning models which are capable of classifying data and measuring regression by means of a learning algorithm. If data are linearly separable, a conventional linear kernel is used to classify them. Otherwise, the data are normally first transformed from input space to feature space, and then they are classified. However, carrying out this transformation is not always practical, and the process itself increases the cost of training and prediction. To address these problems, this paper puts forward an SVM kernel, called polygonal fuzzy weighted or PFW, which effectively classifies data without space transformation, even if the groups in question are not linearly separable and have overlapping areas. This kernel is based on Gaussian data distribution, standard deviation, the three-sigma rule and a polygonal fuzzy membership function. A comparison of our PFW, radial basis function (RBF) and conventional linear kernels in identical experimental conditions shows that PFW produces a minimum of 26% higher classification accuracy compared with the linear kernel, and it outperforms the RBF kernel in two-thirds of class labels, by a minimum of 3%. Moreover, Since PFW runs within the original feature space, it involves no additional computational cost.

Keywords: data classification; SVM; machine learning; SVM kernel; feature; feature space

1. Introduction

Machine learning algorithms learn from observations whose class labels are already defined, and they make predictions about new instances based on the models constructed during the training process [1]. There are three main approaches to machine learning: supervised, unsupervised and semi-supervised [2].

Supervised learning infers classification functions from training labeled data. Major supervised learning approaches are multi-layer perceptron neural networks, decision tree learning, support vector machines and symbolic machine learning algorithms [3]. Unsupervised learning infers functions from the hidden structure of unlabeled data. The major unsupervised learning approaches are clustering (k-means, mixture models and hierarchical clustering), anomaly detection and neural networking (Hebbian learning and generative adversarial networks) [3]. Semi-supervised learning infers classification functions from a large amount of unlabeled data, together with a small amount of labeled data [4]. It thus falls between supervised learning and unsupervised learning. The main semi-supervised learning methods are generative models, low-density separation, graph-based methods and heuristic approaches [3].

Among supervised learning models, the support vector machine (SVM) proposed by Boser, Guyon and Vapnik in 1992 [5] is one of the best-known classification methods. SVMs are capable of solving complicated tasks, like local minima, overfitting and high dimension, while delivering an outstanding performance [6]. Important elements for the efficient use of SVMs are data preprocessing, the selection of the correct kernel, and optimized SVM and kernel setting [5]. The main limitation with SVMs is the difficulty of resolving quadratic programming when the number of data points is large [6].

SVM classification algorithms fall into two classes: linear and nonlinear [7]. A linear classifier is the dot product of a data vector and a weight vector, with the addition of a bias value. Assuming x is a vector, w is a weight vector and b is the bias value, Equation (1) is the discrimination function of a two-class linear classifier [5].

$$f(x) = w^T x + b \tag{1}$$

If $b = 0$, the points x such that $w^T x = 0$ are all vectors to w that go through the origin. In two dimensions, it is a line; and in three dimensions, a plane and, generally, a hyperplane. The bias b is a translation of the hyperplane from its origin [8].

If the data are not separable by a straight line (a hyperplane)—as shown in Figure 1B, above—a non-straight line separates the instances. This type of data is in nonlinear format, and the classifier for it is called nonlinear [5]. The advantage of linear classifiers is that the training algorithm is simpler and it scales with the number of training instances. Linear classifiers, as illustrated in Figure 2, can be converted into nonlinear ones by mapping the input data from space, x, to the feature space, F, by means of a nonlinear function, ϕ. Equation (2) is the nonlinear discrimination function in feature space, F [9].

$$f(x) = w^T \phi(x) + b \tag{2}$$

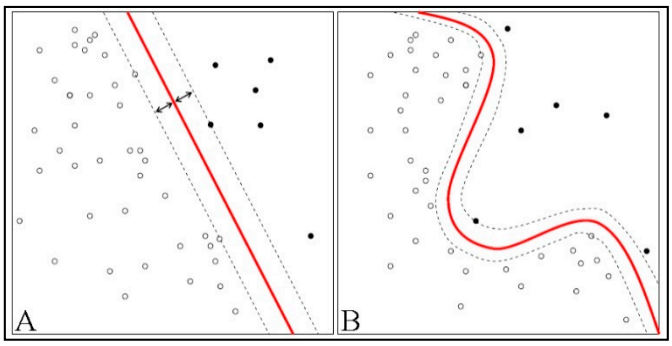

Figure 1. Illustration of (**A**) linear and (**B**) nonlinear classifiable input data [10].

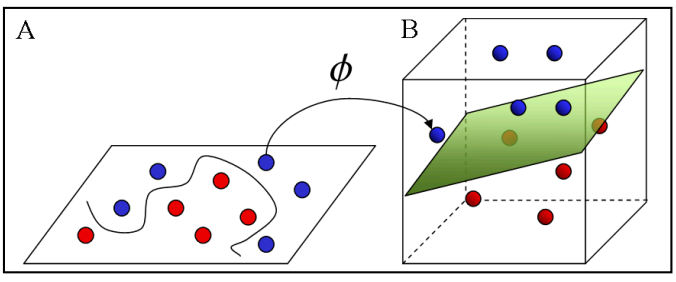

Figure 2. Illustration of mapping data from (**A**) input space to (**B**) feature space [11].

When mapping analogies are applied to input vectors in the d-dimensional space, for feature space, F, the dimensionality is quadratic in d. This entails a quadratic increase in the memory storage required for each feature and in the time needed to compute the classifier's discrimination function. For low data dimensions, this quadratic increase in complexity may be acceptable, but at higher data dimensions, it becomes problematic and, if higher degree monomials are used, completely

unworkable [7]. To overcome this issue, kernel methods are devised to avoid explicit mapping of the data into high-dimensional feature space [5].

In 1963, Vapnik initially proposed a linear classifier, using a maximum-margin hyperplane algorithm [3]. In 1964, Aizerman et al., for the first time, proposed the kernel method. Subsequently, in 1992, Boser et al. put forward the method of creating nonlinear classifiers by applying the kernel method to maximum-margin hyperplanes.

The strength of the kernel method is that it can convert any algorithm that is expressible in terms of dot products between two vectors into nonlinear format [10]. A kernel function indicates an inner product within the feature space, and it is generally denoted as in Equation (3), below.

$$K(x, y) =< \phi(x), \phi(y) > \tag{3}$$

An applicable kernel function is symmetric and continuous, and it ideally has a positive definite Gram matrix: The kernel matrix holds only non-negative values. Algorithms which can operate with kernel functions include support vector machines (SVM), kernel perceptrons, Gaussian processes, canonical correlation analysis, principal component analysis (PCA), ridge regression, linear adaptive filters and spectral clustering [10]. The main SVM kernel types are linear, polynomial, radial basis function and sigmoid kernels [12,13].

Due to the popularity of SVM, a substantial number of specially designed solvers has been developed for SVM optimization [14]. Current software products for SVM training algorithms fall into two categories: (1) software for delivering SVM solver services, like LIBSVM [15] and SVMlight [16]; and (2) machine learning libraries that support various classification methods for essential tasks, like feature selection and preprocessing. Examples of some current popular SVM classifier products are Elefant [17], The Spider [18], Orange [19], Weka [20], Plearn [21], Shogun [22], Lush [23], PyML [24] and RapidMiner [25].

2. Literature Review

Kernels are designed to project input data into feature space, aiming to achieve a hyperplane that efficiently separates data of different classes [26]. Since this paper proposes an SVM kernel, we first review, below, relevant previous research on the SVM kernels currently in use, focusing on their description, the equations they employ, their applications and, finally, their advantages and disadvantages.

Linear kernels are the simplest kernel function, classifying data points in two classes by use of a straight separator line [26]. Since the classification is carried out in a linear format, no mapping of data points to feature space is required. The function is the inner product (x,y), with the addition of an optional constant, c, called bias [26] (Equation (4)).

$$K(x, y) = x^T y + c \tag{4}$$

The main advantages of linear kernels are the quickness of their training and classification processes (since they do not use kernel operations); their low cost; the lower risk of overfitting than with nonlinear kernels; and the lower number of optimization parameters required than with nonlinear kernels [3,10]. Linear kernels can outperform nonlinear ones when the number of features is large relative to the number of training samples, and also when there is a small number of features but the training set is large. Against that, their main disadvantage is that, if the features are not linearly separable—by a hyperplane—nonlinear kernels, such as the Gaussian one, will usually produce better classifications [3,10].

Polynomial kernels are a commonly used kernel function with SVMs that enable the learning of nonlinear models. In the feature space, these characterize similarities between training input vectors and polynomials of the original variables [3,10]. In addition to features of input samples, polynomial

kernels normally use a combination of features, which, for the purposes of regression analysis, are called interaction features [3,10].

Equation (5), below, denotes the polynomial kernel function when the degree of polynomials is N. The variables x and y are the input space vectors, and $c \geq 0$ is an influence tradeoff parameter between lower order and higher order polynomial terms. If $c = 0$, then the kernel is homogeneous. Numerical instability is the problem of polynomial kernels. If $x^T y + c < 1$, by increasing the value of N, the kernel function declines toward zero, while if $x^T y + c > 1$, it tends toward infinity.

$$K(x, y) = \left(x^T y + c\right)^N \tag{5}$$

A radial basis function (RBF), or Gaussian kernel, is a function which classifies data on a radial basis [3,10]. The separator function with two input space feature vectors of x and y is defined as in Equation (6), below.

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right) \tag{6}$$

$\|x - y\|^2$ is the squared Euclidean distance between the two input space feature vectors, while σ is an optional value parameter. Equation (7) is a simplified version of the RBF kernel function that substitutes $\frac{1}{2\sigma^2}$ with γ. The function value decreases by distance, ranging between zero (within the limit) and 1 (if $x = y$).

$$K(x, y) = \exp\left(-\gamma\|x - y\|^2\right) \tag{7}$$

How RBF kernels behave depends to a large extent on the selection of the gamma parameter. An overly large gamma can lead to overfitting, while a small gamma can constrain the model and render it unable to capture the shape or complexity of the data [3,10]. In addition, this type of kernel is robust against adversarial noise and in predictions. However, it has more limitations than neural networks [27,28].

Sigmoid kernels, also known as hyperbolic tangent and multilayer perceptron (MLP) kernels, are an SVM classifier inspired by neural networks. A bipolar sigmoid function is also used as an activation function for the artificial neurons [3,10]. In SVMs, sigmoid kernels are the equivalent of a two-layer perceptron neural network. A Sigmoid function kernel is shown in Equation (8), below.

$$K(x, y) = Tanh(\alpha(x.y) + c) \tag{8}$$

The value of parameter α is generally $\frac{1}{N}$, with N representing the data dimension and c the intercept constant. In certain ranges of α and c, a sigmoid kernel behaves like an RBF. The operation $x.y$ is dot product between the two vectors. The application of Sigmoid kernels is similar to that of RBFs and depends on the chosen level of cross-validation [27,28]. It is an appropriate kernel to use particularly for nonlinear classification in two dimensions or when the number of dimensions is high.

The main advantages of sigmoid kernels are (1) the differentiability at all points of the domain; (2) the fast training process; and (3) presenting a choice of different nonlinearity levels by choosing the sophistication and amount in the membership function [29]. The main drawback of sigmoid is their limited applicability and the fact that they only outperform RBFs in a limited number of cases [29].

The four SVM kernels listed above are not the only ones. There is also a range of less-frequently used SVM kernels, including fisher, graph, string, tree, path, Fourier, B-spline, cosine, multiquadric, wave, log, cauchy, Tstudent, thin-plate and wavelet-SVM kernels (Harr, Daubechies, Coiflet and Symlet) [30].

3. Problem Statement

Typically, linear SVM kernels are used to separate class labels by means of lines. However, the data groups involved are not always linearly separable. Kernel functions seek to resolve this

problem by projecting the data points from the original space to feature space, in order to enhance their separability. However, kernel functions suffer from limitations and cannot always provide an effective or cost-effective solution. A first problem is that they are not applicable to all datasets. In addition, the transformation process itself is expensive and increases both training and prediction costs.

To overcome the above difficulties, this study introduces an SVM kernel which can effectively classify linearly inseparable class labels, without using kernel functions.

4. Fuzzy Weighted SVM Kernel

A number of statistical and probability functions and the related abbreviations are used in this research. Table 1, below, lists and briefly describes these for clarity and consistency.

Table 1. Notations table.

Notation	Description	Notation	Description
μ	Mean	$L\sigma_i$	i^{th} left standard deviation
σ	Standard deviation	$R\sigma_i$	i^{th} right standard deviation
Pr	Probability	$LL\sigma_i$	Left border of i^{th} left standard deviation
Min	Minimum	$RR\sigma_i$	Right border of i^{th} right standard deviation
Max	Maximum	a, b, c, d, e, f	Pivot points in polygonal fuzzy classification
x_{f_i}	Item x in i^{th} feature	$P(x)$	Polygonal fuzzy membership degree of item x
μ_{f_i}	Mean point of i^{th} feature	$P(x_{f_i})$	Polygonal membership degree of item x in i^{th} feature
σ_{f_i}	Standard deviation of i^{th} feature	$\overline{P}(x)$	Average polygonal membership degree of item x
μ_A	Fuzzy membership function on set A	µs	Microsecond

4.1. Principal Probability and Statistical Functions

In the theory of probability, normal or Gaussian distribution is defined as a continuous probability distribution that is based on the premise that data distribution will converge to the normal, especially in natural science, if a sufficiently large number of observations is made. As this normal distribution takes the form of a bell when plotted on a graph, it is also often informally called the bell curve.

Standard deviation (generally denoted as σ) is a statistical function used to quantify the amount of variation or dispersion in a given dataset. Moreover, it is generally used to define the level of confidence in the accuracy of a given set of data. A high value for standard deviation means that the data are spread over a wide range, whereas a low value indicates that most of the data are close to the mean.

In statistics, the three-sigma, 68-95-99.7, or empirical rule, describes the density of data within standard deviation bands at both sides of the mean point. Specifically, in Gaussian distribution 68.27% (Equation (9)), 95.45% (Equation (10)) and 99.73% (Equation (11)) of the data values should be located within a distance of one, two and three standard deviation bands from both sides of the mean point, respectively.

$$\Pr(\mu - \sigma \leq X \leq \mu + \sigma) \approx 0.6827 \tag{9}$$

$$\Pr(\mu - 2\sigma \leq X \leq \mu + 2\sigma) \approx 0.9545 \tag{10}$$

$$\Pr(\mu - 3\sigma \leq X \leq \mu + 3\sigma) \approx 0.9973 \tag{11}$$

4.2. Data Classification

In the proposed kernel of this research, the role of standard deviation is to quantify the Gaussian distributed data, while the three-sigma rule defines the importance of the data placed in each band. According to the three-sigma rule, the data located in the first standard deviation bands are the most reliable. However, reliability of data declines from the second standard deviation bands toward the third bands, and any data lying beyond the third bands are considered as noise and, accordingly, ignored. Figure 3, below, presents an illustration of quantified Gaussian data distribution. The first, second and third standard deviation bands are highlighted in different colors, with the three right and left bands labeled as Rσ_1, Rσ_2 and Rσ_3 and Lσ_1, Lσ_2 and Lσ_3 respectively.

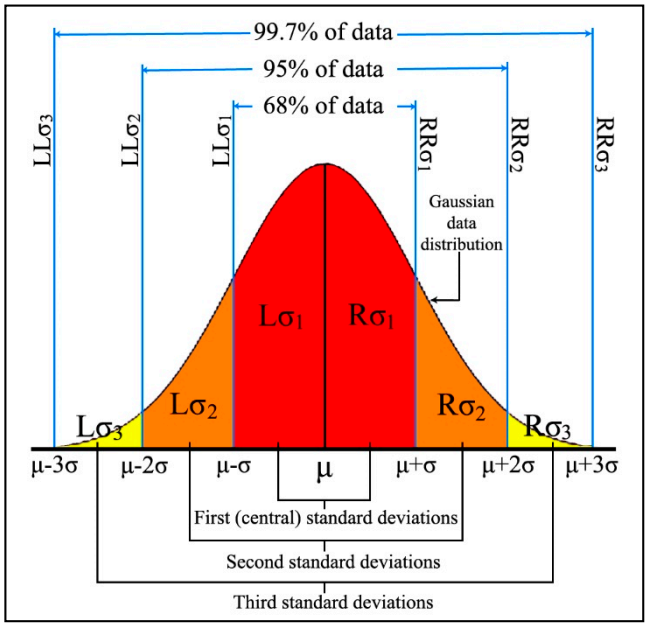

Figure 3. Illustration of data quantification and defining borders.

When data are normally distributed according to the three-sigma rule, the central (first) standard deviation bands hold the highest density, while this declines toward the outer bands. If, however, the data are abnormally distributed, they may not comply with the three-sigma rule: the data density in some of the bands might be higher or lower than the expected normal values, and some bands might not even exist.

Figure 4A, below, illustrates a normal data distribution, with a higher density within the Lσ_1 and Rσ_1 bands and a lower density in the more distant bands. Figure 4B, on the other hand, displays an instance of abnormally distributed data: a high density within Lσ_1, but the rest of the data dispersed only across the right bands. In Figure 4B, not only do the Lσ_2 and Lσ_3 bands not exist, but the LLσ_1 border has had to be relocated to the point of 0. However, it should be noted that sometimes border relocation, as is the case with LLσ_3 in Figure 4A, might be necessary, even if the data are overall normally distributed.

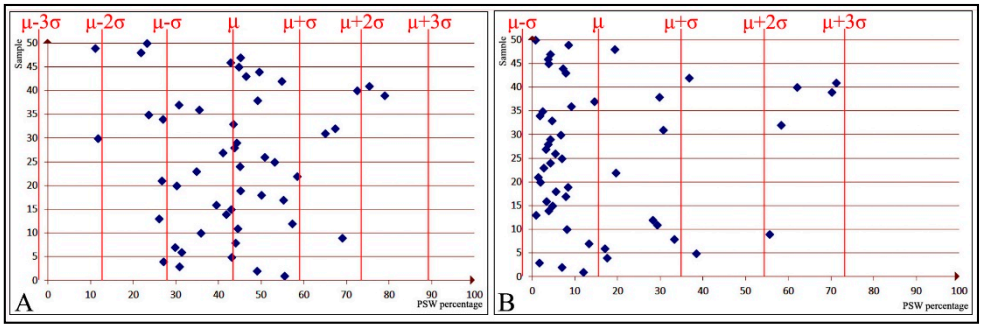

Figure 4. Illustration of (**A**) normal and (**B**) abnormal data distribution within standard deviation bands.

The criteria for calculating the left and right borders of the left and right standard deviation bands, respectively, are shown in Equations (12) and (13), below. The mean and max denote the need to relocate borders, while the word "Null" indicates that, due to abnormal data distribution, the band does not exist.

$$\mathrm{LL}\sigma_i = \begin{cases} \mu - (\sigma * i) & \text{if } \mu - (\sigma * i) \geq \min \\ \min & \text{else if } \mu - (\sigma * (i-1)) > \min \\ \text{Null} & \text{else} \end{cases} \tag{12}$$

$$\mathrm{RR}\sigma_i = \begin{cases} \mu + (\sigma * i) & \text{if } \mu + (\sigma * i) \leq \max \\ \max & \text{else if } \mu + (\sigma * (i-1)) < \max \\ \text{Null} & \text{else} \end{cases} \tag{13}$$

4.3. Membership Function

On the discoursed universe of X, the fuzzy membership function of set A is defined as $\mu_A \rightarrow [0,\ 1]$, where the mapping values of X elements varies between 0 and 1. This membership degree quantifies the extent of membership of element X within fuzzy set A. In fuzzy classification, when the membership criteria resemble a polygon, the function is normally denoted as $P(x)$ rather than μ_A.

In Figure 5A, above, the polygon's limits are respectively labeled as a, b, c, d, e and f, where a < b < c < d < e < f. In Figure 5B,C, the polygonal shape of section A is mapped onto the Gaussian distribution, which is segmented according to standard deviation and the three-sigma rule. This mapping is an essential part of the designed classifier. The central standard deviation bands, as can be seen in the above figure, hold the most reliable data and are the densest bands. The inflation points of c and d are mapped onto the boundaries of the central bands. The whole area between these points receives the full membership degree of 1, as it holds the most reliable data. The lower limit a and upper limit f are respectively mapped onto $\mathrm{LL}\sigma_3$ and $\mathrm{RR}\sigma_3$, as the leftmost and rightmost borders of reliable data. The areas between $\mathrm{LL}\sigma_3$ and $\mathrm{LL}\sigma_1$ (a and c), as well as between $\mathrm{RR}\sigma_1$ and $\mathrm{RR}\sigma_3$ (d and f), are divided into two equal parts. The leftmost and rightmost bands (a to b and e to f) receive a membership degree equal or bigger than 0 and less than 0.5, as these hold the least reliable data, and the bands at sides of the central area (b to c and d to e) receive a membership degree equal or bigger than 0.5 and less than 1, as a medium-to-high reliable portion of data, depending on their precise location along the axis. Figure 5B,C represents the short-tail and long-tail Gaussian data distributions, respectively. In both distributions, because of the three-sigma rule, the two left and the two right polygon sides follow slope of the distribution, resulting in the achievement of a more accurate fuzzy membership degree.

This is designed to maximize classification accuracy. The criteria for calculating the polygonal fuzzy membership function are shown in Equation (14), below.

$$P(x) = \begin{cases} 0 & (x < a) \Big| (x > f) \\ \frac{x-a}{b-a} & a \le x < b \\ \frac{x-b}{c-b} & b \le x < c \\ 1 & c \le x \le d \\ \frac{e-x}{e-d} & d < x \le e \\ \frac{f-x}{f-e} & e < x \le f \end{cases} \tag{14}$$

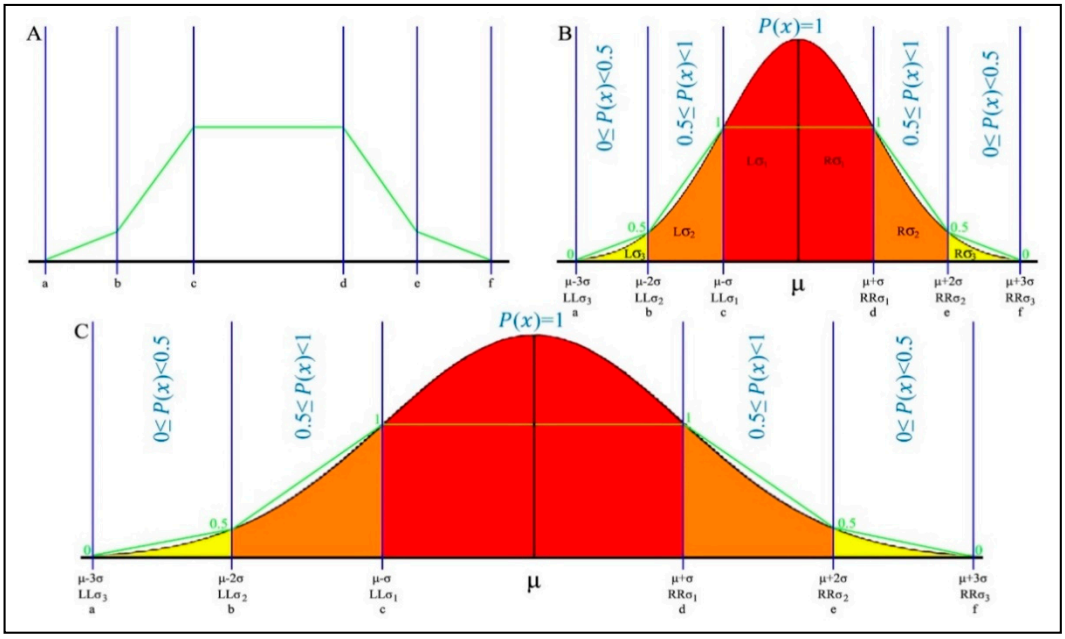

Figure 5. (**A**) Polygonal fuzzy membership; (**B**) mapping of polygonal fuzzy membership on short-tail segmented Gaussian distribution (bell curve) based on standard deviation and three-sigma rule; (**C**) mapping of polygonal fuzzy membership on long-tail segmented Gaussian distribution (bell curve) based on standard deviation and three-sigma rule.

Based on the mapping inflation points of Figure 5B,C, the equivalent values of the limits a, b, c, d, e and f are substituted in Equation (14) and then simplified to construct Equation (15).

$$P(x) = \begin{cases} 0 & (x < \mu - 3\sigma) \Big| (x > \mu + 3\sigma) \\ \frac{x-\mu+3\sigma}{\sigma} & \mu - 3\sigma \le x < \mu - 2\sigma \\ \frac{x-\mu+2\sigma}{\sigma} & \mu - 2\sigma \le x < \mu - \sigma \\ 1 & \mu - \sigma \le x \le \mu + \sigma \\ \frac{\mu+2\sigma-x}{\sigma} & \mu + \sigma < x \le \mu + 2\sigma \\ \frac{\mu+3\sigma-x}{\sigma} & \mu + 2\sigma < x \le \mu + 3\sigma \end{cases} \tag{15}$$

However, data do not always appear in a single feature. In such circumstances, the membership degree is the average of the membership degrees of all relevant features. To support multidimensional

features, Equation (15) is modified into Equation (16), below, which allows for the calculation of the membership degree of the given sample, x, in the i^{th} feature of k features.

$$P\left(x_{f_i}\right) = \begin{cases} 0 & \left(x_{f_i} < \mu_{f_i} - 3\sigma_{f_i}\right) \Big| \left(x_{f_i} > \mu_{f_i} + 3\sigma_{f_i}\right) \\ \frac{x_{f_i} - \mu_{f_i} + 3\sigma_{f_i}}{\sigma_{f_i}} & \mu_{f_i} - 3\sigma_{f_i} \le x_{f_i} < \mu_{f_i} - 2\sigma_{f_i} \\ \frac{x_{f_i} - \mu_{f_i} + 2\sigma_{f_i}}{\sigma_{f_i}} & \mu_{f_i} - 2\sigma_{f_i} \le x_{f_i} < \mu_{f_i} - \sigma_{f_i} \\ 1 & \mu_{f_i} - \sigma_{f_i} \le x_{f_i} \le \mu_{f_i} + \sigma_{f_i} \\ \frac{\mu_{f_i} + 2\sigma_{f_i} - x_{f_i}}{\sigma} & \mu_{f_i} + \sigma_{f_i} < x_{f_i} \le \mu_{f_i} + 2\sigma_{f_i} \\ \frac{\mu_{f_i} + 3\sigma_{f_i} - x_{f_i}}{\sigma_{f_i}} & \mu_{f_i} + 2\sigma_{f_i} < x_{f_i} \le \mu_{f_i} + 3\sigma_{f_i} \end{cases} \tag{16}$$

The probability rule of sum is used when we calculate the probability of a union of events, by adding the distinct probabilities together. In other words, it gives the total probability of the final result when there are mutually exclusive events. The summation also could be interpreted as a weighted average—a value that is helpful for problem-solving. Using this rule, Equation (17), below, is the final equation which, along with Equation (16), calculates the normalized membership degree of the sample, x, in k features.

$$\overline{P}(x) = \frac{\sum_{i=1}^{k} P\left(x_{f_i}\right)}{k} \tag{17}$$

4.4. Noise Filtering

The designed classifier eliminates and mitigates natural data noise in two ways. First of all, it separates reliable data from noisy data, based on Gaussian data distribution and the three-sigma rule. In other words, it ignores any data beyond the reliable boundary. Second, it determines the influence of any given data in decisions, by computing their membership degree $\overline{P}(x)$. In simple terms, data located in the central bands are more influential than data located in the side bands.

4.5. Reference Profiles

The PFW kernel records the statistical properties of all classes of the training data and stores these based on the structure presented in Table 2, below. The number of profiles corresponds to the number of data classes, and these are then used to classify future given instances. The extracted statistics from each feature of each data group are divided into six standard deviation bands. Lσ_1 and Rσ_1 (LL$\sigma_1(f_i)$ $\le x_{f_i} \le$ RR$\sigma_1(f_i)$) hold the most trustworthy data for the i^{th} feature, and hence the membership degree $P(x_{f_i})$ for these bands equals 1. The membership degree for Lσ_2 (LL$\sigma_2(f_i) \le x_{f_i} <$ LL$\sigma_1(f_i)$) and Rσ_2 (RR$\sigma_1(f_i) < x_{f_i} \le$ RR$\sigma_2(f_i)$) is $0.5 \le P(x_{f_i}) < 1$; and for the third bands, Lσ_3 (LL$\sigma_3(f_i) \le x_{f_i} <$ LL$\sigma_2(f_i)$) and Rσ_3 (RR$\sigma_2(f_i) < x_{f_i} \le$ RR$\sigma_3(f_i)$), the value is $0 \le P(x_{f_i}) < 0.5$. In cases where a band does not exist, because of abnormal data distribution, the associated value range in the profile is replaced with "Null".

Table 2. Statistical structure of reference profiles for k features (i:{1..k}).

	Lσ_3.	**Lσ_2**	**Lσ_1 & Rσ_1**	**Rσ_2**	**Rσ_3**
	$0 \le \boldsymbol{P(x_{f_i})} < 0.5$	$0.5 \le \boldsymbol{P(x_{f_i})} < 1$	$\boldsymbol{P(x_{f_i})} = 1$	$0.5 \le \boldsymbol{P(x_{f_i})} < 1$	$0 \le \boldsymbol{P(x_{f_i})} < 0.5$
Feature 1	LL$\sigma_3(f_1) \le x_{f_1}$ < LL$\sigma_2(f_1)$|Null	LL$\sigma_2(f_1) \le x_{f_1}$ < LL$\sigma_1(f_1)$|Null	LL$\sigma_1(f_1) \le x_{f_1} \le$ RR$\sigma_1(f_1)$|Null	RR$\sigma_1(f_1) < x_{f_1} \le$ RR$\sigma_2(f_1)$|Null	RR$\sigma_2(f_1) < x_{f_1} \le$ RR$\sigma_3(f_1)$|Null
	⋮	⋮	⋮	⋮	⋮
Feature k	LL$\sigma_3(f_k) \le x_{f_k}$ < LL$\sigma_2(f_k)$|Null	LL$\sigma_2(f_k) \le x_{f_k}$ < LL$\sigma_1(f_k)$|Null	LL$\sigma_1(f_k) \le x_{f_k} \le$ RR$\sigma_1(f_k)$|Null	RR$\sigma_1(f_k) < x_{f_k} \le$ RR$\sigma_2(f_k)$|Null	RR$\sigma_2(f_k) < x_{f_k} \le$ RR$\sigma_3(f_k)$|Null

At classification time, PFW kernel calculates the membership degree of the given instance in comparison with each one of the profiles, according to Equation (17). Its class label is then detected based on the profile that achieves the highest membership degree.

5. Evaluation and Comparison

To examine the capabilities and classification accuracy of PFW kernel, we tested it against linear and RBF SVM kernels, in two approaches. First, the run-time of the kernels, and second, using the kernels as classification engines for PSW steganalysis [31], to compare against each other for accuracy. The sections below describe the results of time complexity evaluation, the classification experiment area, the chosen classification feature, the properties of the training and testing image sets, and, finally, a comparison of the achieved accuracy in classification, together with a discussion about the proposed kernel.

5.1. Run-Time Comparison

Using polygonal fuzzy membership in PFW lowers the complexity of training and classification processes to a set of very simple calculations, in Equation (15). To evaluate this improvement, the run-time of PFW, RBF and linear kernels is measured within an identical condition: to this end, a database in a single dimension and 1000 numerical feature values in two classes produced and fed to the kernels—80% for training and 20% for classification. Scikit-learn library in Python was used to implement linear and RBF kernels and the required code developed for PFW kernel. The experiment was performed by using a computer equipped with a Core i7 processor and 8 gigabytes RAM. As shown in Table 3, the training and classification processes consumed 46,865 microseconds for PFW, while this value rose to 421,775 and 437,390 microseconds for linear and RBF kernels, respectively—approximately nine times faster.

Table 3. Run-time comparison between linear, RBF and PFW kernels.

	Linear	**RBF**	**PFW**
Run-time	421,775 µs	437,390 µs	46,865 µs

5.2. Accuracy Evaluation

5.2.1. Classification Experiment Area

The classification problem tackled was distinguishing the least significant bit replacement (LSBR) steganographied images from clean ones. Steganography is the practice of hiding secret data within the body of digital media, like image and audio, to conceal its presence. A range of techniques is devised for this purpose, including least significant bit (LSB), discrete wavelet transform (DWT) and direct cosine transform (DCT). Amongst these, LSB is the most widely used method, as it is simple to implement and delivers a high capacity for data hiding. It replaces the least significant bits of the host file with the bits of the secret message. This replacement imposes some noise on the body of the host file.

As a response to steganography, steganalysis is the science of uncovering covert communications and thereby defeating those who create these, through a deep structural and technical understanding of steganographic methods [32,33]. Steganalysis, normally by means of an SVM classifier, measures image data anomalies for the likely signs of data embedding.

To evaluate the accuracy of the PFW classifier, we used it for classifying the extracted features in an image steganalysis technique called pixel similarity weight (PSW) [31]. Hiding external data within an image declines the color correlativity that exists between its pixels. To discover if an image is carrying a secret message, PSW steganalysis analyzes the color correlativity of all image pixels with their neighboring pixels, and from these then comes up with the final decision of whether the image is

clean or not. Sections 5.2.2 and 5.2.3, below, give more insight on how the features are extracted from the images. Full technical details are accessible in Reference [31].

5.2.2. Classification Features

Images comprise three pixel classes: flat, smooth and edgy. Each of these has a certain percentage of pixels with an identical color in their neighboring zones. However, this level of correlativity falls when external data are embedded in the body of the image [31]. Thus, the selected classification feature in PSW steganalysis is the percentage of pixels with identical color that are immediately connected to the analyzed pixel, in the first, second and third neighboring zones.

Capacity in steganalysis is defined as the number of the replaced bits in one byte. If it exceeds a certain level, the generated artifacts reveal the presence of the hidden secret. Therefore, PSW steganalysis analyzes the images for 1-bit and 2-bit per byte embedding, up to a threshold that generates no artifacts. Table 4, immediately below, meanwhile presents the average statistical properties of the pixel classes in the first, second and third neighboring zones, before and after embedding. The numbers clearly show how increasing the embedding ratio progressively reduces the level of color correlativity of the pixels.

Table 4. Average color correlativity of flat, smooth and edgy pixel classes before and after LSB replacement [31].

	Pixel Class								
	Flat			**Smooth**			**Edgy**		
Used Capacity	**0-bit**	**1-bit**	**2-bit**	**0-bit**	**1-bit**	**2-bit**	**0-bit**	**1-bit**	**2-bit**
Neighboring zone 1	62.05%	8.02%	1.13%	27.64%	3.86%	0.652%	4.60%	0.64%	0.11%
Neighboring zone 2	52.94%	6.90%	1.01%	16.04%	2.32%	0.465%	2.41%	0.33%	0.05%
Neighboring zone 3	50.71%	6.60%	0.98%	13.09%	1.90%	0.396%	1.9%	0.26%	0.04%

While the average color correlativity of pixels noticeably drops by increasing percent of data embedding in a very sensible way, the falls only change the density of each data group. In other words, instead of having linearly classifiable groups, they are overlapped. This means that the results cannot reliably be classified linearly.

5.2.3. Training and Testing Image Sets Formation

In order to train the classifier, three distinct image sets, each consisting entirely of one of these pixel classes, are required. Therefore, what is required is a database meeting the three pixel-class definitions [34,35] manually constructed by 150 images and then embedded in 1-bit and 2-bit per byte—totally 450 images. The pixel similarity weight feature then extracted from the clean and embedded images to construct three reference profiles: 0-bit (clean), 1-bit, and 2-bit. Table 5, below, presents the fundamental structure of the PSW steganalysis profiles.

To perform steganalysis, PSW extracts the nine feature values from all pixels and then calculates the average of each one. In the next step, it computes the membership degree of the nine average feature values in comparison with the three profiles, according to Equation (16). To make the final decision, it then produces the average membership degree of each profile, using Equation (17)—three normalized membership degrees corresponding to the three profiles. The profile which achieves the highest membership degree detects which one of the classes that image belongs to: 0-bit, 1-bit or 2-bit embedded. More comprehensive technical details about these processes are available in our separate article entitled “PSW Statistical LSB Image Steganalysis” [31].

Table 5. Fundamental structure of a reference profile in PSW steganalysis [31].

Features	$L\sigma_3$	$L\sigma_2$	$L\sigma_1$ & $R\sigma_1$	$R\sigma_2$	$R\sigma_3$
	$0 \le P(x_{f_i}) < 0.5$	$0.5 \le P(x_{f_i}) < 1$	$P(x_{f_i}) = 1$	$0.5 \le P(x_{f_i}) < 1$	$0 \le P(x_{f_i}) < 0.5$
Flat pixels—zone 1	$LL\sigma_3(f_1) \le x_{f_1} < LL\sigma_2(f_1)\vert Null$	$LL\sigma_2(f_1) \le x_{f_1} < LL\sigma_1(f_1)\vert Null$	$LL\sigma_1(f_1) \le x_{f_1} \le RR\sigma_1(f_1)\vert Null$	$RR\sigma_1(f_1) < x_{f_1} \le RR\sigma_2(f_1)\vert Null$	$RR\sigma_2(f_1) < x_{f_1} \le RR\sigma_3(f_1)\vert Null$
Flat pixels—zone 2	$LL\sigma_3(f_2) \le x_{f_2} < LL\sigma_2(f_2)\vert Null$	$LL\sigma_2(f_2) \le x_{f_2} < LL\sigma_1(f_2)\vert Null$	$LL\sigma_1(f_2) \le x_{f_2} \le RR\sigma_1(f_2)\vert Null$	$RR\sigma_1(f_2) < x_{f_2} \le RR\sigma_2(f_2)\vert Null$	$RR\sigma_2(f_2) < x_{f_2} \le RR\sigma_3(f_2)\vert Null$
Flat pixels—zone 3	$LL\sigma_3(f_3) \le x_{f_3} < LL\sigma_2(f_3)\vert Null$	$LL\sigma_2(f_3) \le x_{f_3} < LL\sigma_1(f_3)\vert Null$	$LL\sigma_1(f_3) \le x_{f_3} \le RR\sigma_1(f_3)\vert Null$	$RR\sigma_1(f_3) < x_{f_3} \le RR\sigma_2(f_3)\vert Null$	$RR\sigma_2(f_3) < x_{f_3} \le RR\sigma_3(f_3)\vert Null$
Smooth pixels—zone 1	$LL\sigma_3(f_4) \le x_{f_4} < LL\sigma_2(f_4)\vert Null$	$LL\sigma_2(f_4) \le x_{f_4} < LL\sigma_1(f_4)\vert Null$	$LL\sigma_1(f_4) \le x_{f_4} \le RR\sigma_1(f_4)\vert Null$	$RR\sigma_1(f_4) < x_{f_4} \le RR\sigma_2(f_4)\vert Null$	$RR\sigma_2(f_4) < x_{f_4} \le RR\sigma_3(f_4)\vert Null$
Smooth pixels—zone 2	$LL\sigma_3(f_5) \le x_{f_5} < LL\sigma_2(f_5)\vert Null$	$LL\sigma_2(f_5) \le x_{f_5} < LL\sigma_1(f_5)\vert Null$	$LL\sigma_1(f_5) \le x_{f_5} \le RR\sigma_1(f_5)\vert Null$	$RR\sigma_1(f_5) < x_{f_5} \le RR\sigma_2(f_5)\vert Null$	$RR\sigma_2(f_5) < x_{f_1} \le RR\sigma_3(f_5)\vert Null$
Smooth pixels—zone 3	$LL\sigma_3(f_6) \le x_{f_6} < LL\sigma_2(f_6)\vert Null$	$LL\sigma_2(f_6) \le x_{f_6} < LL\sigma_1(f_6)\vert Null$	$LL\sigma_1(f_6) \le x_{f_6} \le RR\sigma_1(f_6)\vert Null$	$RR\sigma_1(f_6) < x_{f_6} \le RR\sigma_2(f_6)\vert Null$	$RR\sigma_2(f_6) < x_{f_1} \le RR\sigma_3(f_6)\vert Null$
Edgy pixels—zone 1	$LL\sigma_3(f_7) \le x_{f_7} < LL\sigma_2(f_7)\vert Null$	$LL\sigma_2(f_7) \le x_{f_7} < LL\sigma_1(f_7)\vert Null$	$LL\sigma_1(f_7) \le x_{f_7} \le RR\sigma_1(f_7)\vert Null$	$RR\sigma_1(f_7) < x_{f_7} \le RR\sigma_2(f_7)\vert Null$	$RR\sigma_2(f_7) < x_{f_1} \le RR\sigma_3(f_7)\vert Null$
Edgy pixels—zone 2	$LL\sigma_3(f_8) \le x_{f_8} < LL\sigma_2(f_8)\vert Null$	$LL\sigma_2(f_8) \le x_{f_8} < LL\sigma_1(f_8)\vert Null$	$LL\sigma_1(f_8) \le x_{f_8} \le RR\sigma_1(f_8)\vert Null$	$RR\sigma_1(f_8) < x_{f_8} \le RR\sigma_2(f_8)\vert Null$	$RR\sigma_2(f_8) < x_{f_1} \le RR\sigma_3(f_8)\vert Null$
Edgy pixels—zone 3	$LL\sigma_3(f_9) \le x_{f_9} < LL\sigma_2(f_9)\vert Null$	$LL\sigma_2(f_9) \le x_{f_9} < LL\sigma_1(f_9)\vert Null$	$LL\sigma_1(f_9) \le x_{f_9} \le RR\sigma_1(f_9)\vert Null$	$RR\sigma_1(f_9) < x_{f_9} \le RR\sigma_2(f_9)\vert Null$	$RR\sigma_2(f_9) < x_{f_1} \le RR\sigma_3(f_9)\vert Null$

To evaluate the accuracy of our PFW kernel, the statistical features of the training image set were used both to construct PFW reference profiles and to train the linear and RBF SVM kernels. The PSW steganalysis technique [31] was then applied to the three kernels, to discriminate clean images from embedded ones. The accuracy of the kernels was then evaluated against the UW image database, a set of 1333 images in three versions of 0-bit (clean), 1-bit and 2-bit embedded. The dimensions of UW images vary between 512∗768 and 589∗883. The images are mainly of tourist sites, including buildings, objects, people and scenery, in different ranges of lighting, tone and contrast.

5.3. Comparison and Discussion

The accuracy results from our tests of the three kernels in classifying clean and embedded copies of the UW images are presented in Table 6, below. These show that, when the data groups have overlapping areas, the accuracy of the classical linear SVM kernel drops, whereas the PFW kernel can still effectively discriminate between different images. In fact, the PFW outperforms the linear kernel by a large margin, producing a minimum of 26.021% and a maximum of 40.425% greater accuracy. Comparing PFW and RBF kernels reveals that RBF outperforms PFW by 1.251% in 0-bit embedding ratio, while in the two other ratios (1-bit and 2-bit), PFW is more efficient, by 2.894% and 8.693%, respectively.

Table 6. Comparison of classification accuracy of PFW, RBF and linear SVM kernels, based on their performance in PSW steganalysis.

	0-bit (Clean)	1-bit	2-bit
PFW kernel	69.042%	77.004%	98.635%
RBF kernel	70.293%	74.11%	89.942%
Linear kernel	43.021%	49.771%	57.21%

To further illustrate the classification accuracy of each kernel, a sample of pixel classification is presented in Figure 6, below. Image A is the original image that was broken down into the three pixel

classes. The three images labeled B, C and D are the smooth pixel class sub-images produced by the PFW, RBF and linear kernels, respectively. These images clearly show that the classification of pixels by the PFW kernel is more accurate and smoother. Moreover, the PFW kernel has detected and classified a substantially larger portion of the pixels.

Figure 6. (**A**) original image; (**B**) smooth detected pixels by PFW kernel; (**C**) smooth detected pixels by RBF kernel; and (**D**) smooth detected pixels by linear kernel.

6. Conclusions

Supervised learning is a class of machine learning in which a function is inferred from labeled training data. The inferred function then determines the class label of unseen instances. One type of supervised learning, SVM (support vector machine), predicts the categories of new examples, based on a given set of training examples. SVM maps the data into a space in which the categories are dividable by a clear gap. However, such mapping is not always sufficient to produce clearly separable categories. In such cases, a further step is required to transform the data from their original space to feature space, increasing the computational cost of training and classification.

Our PFW (polygonal fuzzy weighted) kernel is designed to map data accurately, without transforming them to feature space, even when groups are not linearly separable. The kernel accurately predicts the class of new instances in the original space, even if the groups have overlapping areas. It is structured based on Gaussian distribution, standard deviation, the three-sigma rule and a polygonal fuzzy membership function. PFW generates one profile for each data class, based on the statistics extracted during the training process. For the purposes of classification, the new instances are then compared with the reference profiles, based on a polygonal fuzzy membership function: The highest degree of membership achieved defines the prediction result.

To test the accuracy of our PFW kernel, we used it together with RBF and conventional linear kernels as classification engines for PSW steganalysis of the same set of 1333 images, whose classification feature data had overlapping data groups. The results showed that the PFW clearly outperformed the linear kernel in predicting the class labels of the images—yielding a minimum of 26% and as much as 40% higher classification accuracy. PFW kernel in two out of three image classes outperformed RBF kernel by 3% and 9%, while RBF delivered better results, by 1% in one class label.

Author Contributions: S.S.C. carried out conceptualization, data curation, methodology, and writing the manuscript. A.A.M. designed the research methodology and also supervised and administrated the project. A.A.A. performed implementation, validation, and visualization. M.Z. conducted formal analysis, investigation, and validation. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Iran's National Elites Foundation grant number B/1549.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Anzai, Y. *Pattern Recognition and Machine Learning*; Elsevier: Amsterdam, The Netherlands, 2012.
2. Paiva, J.S.; Cardoso, J.; Pereira, T. Supervised learning methods for pathological arterial pulse wave differentiation: A SVM and neural networks approach. *Int. J. Med. Informat.* **2018**, *109*, 30–38. [CrossRef] [PubMed]
3. Nalepa, J.; Kawulok, M. Selecting training sets for support vector machines: A review. *Artif. Intell. Rev.* **2019**, *52*, 857–900. [CrossRef]
4. Qian, P.; Xi, C.; Xu, M.; Jiang, Y.; Su, K.H.; Wang, S.; Muzic, R.F., Jr. SSC-EKE: Semi-supervised classification with extensive knowledge exploitation. *Inf. Sci.* **2018**, *422*, 51–56. [CrossRef] [PubMed]
5. Ben-Hur, A.; Ong, C.S.; Sonnenburg, S.; Schölkopf, B.; Rätsch, G. Support vector machines and kernels for computational biology. *PLos Comput. Biol.* **2008**, *4*, e1000173. [CrossRef] [PubMed]
6. Amini, S.; Homayouni, S.; Safari, A.; Darvishsefat, A.A. Object-based classification of hyperspectral data using Random Forest algorithm. *Geo-Spat. Inf. Sci.* **2018**, *21*, 127–128. [CrossRef]
7. Kecman, V. *Learning and Soft Computing: Support Vector Machines, Neural Networks, and Fuzzy Logic Models*; MIT Press: Cambridge, MA, USA, 2001.
8. Ni, T.; Gu, X.; Wang, J.; Zheng, Y.; Wang, H. Scalable transfer support vector machine with group probabilities. *Neurocomputing* **2018**, *273*, 570–572. [CrossRef]
9. Cristianini, N.; Shawe-Taylor, J. *An Introduction to Support Vector Machines and other Kernel-Based Learning Methods*; Cambridge University Press: Cambridge, UK, 2000.
10. "Support Vector Machine." Wikipedia, Wikimedia Foundation. 2 March 2020. Available online: en.wikipedia.org/wiki/Support-vector_machine (accessed on 25 March 2020).
11. "Reconhecimento De Padrões: Support Vector Machines". Reconhecimento De Padrões: Support Vector Machines. Available online: www.lapix.ufsc.br/ensino/reconhecimento-de-padroes/reconhecimento-de-padroessupport-vector-machines/ (accessed on 25 March 2020).
12. Cho, G.S.; Gantulga, N.; Choi, Y.W. A comparative study on multi-class SVM & kernel function for land cover classification in a KOMPSAT-2 image. *KSCE J. Civ. Eng.* **2017**, *21*, 1894–1904.
13. Bulut, S.; Günlü, A. Comparison of different supervised classification algorithms for land use classes. *Kast. Üniversitesi Orman Fakültesi Derg.* **2016**, *16*, 528–535.
14. Bordes, A.; Ertekin, S.; Weston, J.; Bottou, L. Fast kernel classifiers with online and active learning. *J. Mach. Learn. Res.* **2005**, *6*, 1579–1619.
15. Chang, C.C.; Lin, C.J. LIBSVM: A Library for Support Vector Machines. 2013 Software. Available online: https://www.csie.ntu.edu.tw/~{}cjlin/libsvm/ (accessed on 14 September 2019).
16. Ma, Y.; Guo, G. (Eds.) *Support Vector Machines. Applications*; Springer: New York, NY, USA, 2014.
17. Webers, C.; Gawande, K.; Smola, A.; Hui, H.; Javen, T.; Shi, Q.F.; Yu, J.; Mcauley, J.; Song, L.; Quoc, L.; et al. *Elefact User's Manual (Release 0.4)*; Technical Report; NICTA. Available online: http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=F22EF727ECFEA0E8F1454BF78C2E35B0?doi=10.1.1.170.9708&rep=rep1&type=pdf (accessed on 14 September 2019).
18. The Spider Toolbox. Available online: https://people.kyb.tuebingen.mpg.de/spider/main.html (accessed on 14 September 2019).
19. Demšar, J.; Zupan, B.; Leban, G.; Curk, T. Orange: From experimental machine learning to interactive data mining. In Proceedings of the European Conference on Principles of Data Mining and Knowledge Discovery, Antwerp, Belgium, 20 September 2004; Springer: Berlin/Heidelberg, Germany; pp. 537–539.
20. Witten, I.H.; Frank, E.; Hall, M.A.; Pal, C.J. *Data Mining: Practical Machine Learning Tools and Techniques*; Morgan Kaufmann: Burlington, MA, USA, 2016.
21. Plearn Toolbox. Available online: https://sourceforge.net/projects/plearn/ (accessed on 14 September 2019).
22. Sonnenburg, S.; Ra¨tsch, G.; Scha¨fer, C.; Scho¨lkopf, B. Large scale multiple kernel learning. *J. Mach. Learn. Res.* **2006**, *7*, 1531–1565. Available online: http://jmlr.csail.mit.edu/papers/v7/sonnenburg06a.html (accessed on 14 September 2019).
23. Bottou, L.; Cun, Y.L. Lush Reference Manual. 2002. Available online: http://lush.sourceforge.net (accessed on 14 September 2019).
24. PyML Toolbox. Available online: http://pyml.sourceforge.net (accessed on 14 September 2019).

25. Mierswa, I.; Wurst, M.; Klinkenberg, R.; Scholz, M.; Euler, T. Yale: Rapid prototyping for complex data mining tasks. In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, 20 August 2006; pp. 935–940.
26. Salazar, D.A.; Vélez, J.I.; Salazar, J.C. Comparison between SVM and logistic regression: Which one is better to discriminate? *Rev. Colomb. Estadística* **2012**, *35*, 223–237.
27. Naghibi, S.A.; Ahmadi, K.; Daneshi, A. Application of Support Vector Machine, Random Forest, and Genetic Algorithm Optimized Random Forest Models in Groundwater Potential Mapping. *Water Resour. Manag.* **2017**, *31*, 2761–2775. [CrossRef]
28. Nieto, P.G.; García-Gonzalo, E.; Antón, J.Á.; Suárez, V.G.; Bayón, R.M.; Martín, F.M. A comparison of several machine learning techniques for the centerline segregation prediction in continuous cast steel slabs and evaluation of its performance. *J. Comput. Appl. Math.* **2017**, *330*, 877–895. [CrossRef]
29. Camps-Valls, G.; Martín-Guerrero, J.D.; Rojo-Alvarez, J.L.; Soria-Olivas, E. Fuzzy sigmoid kernel for support vector classifiers. *Neurocomputing* **2004**, *62*, 501–506. [CrossRef]
30. Zhang, J. A Complete List of Kernels Used in Support Vector Machines. *Biochem. Pharmacol. (Los Angel)* **2015**, *4*, 2167-0501.
31. Chaeikar, S.S.; Zamani, M.; Manaf, A.B.; Zeki, A.M. PSW statistical LSB image steganalysis. *Multime Tools Appl.* **2018**, *77*, 805–835. [CrossRef]
32. Zamani, M.; Manaf, A.A.; Ahmad, R.; Jaryani, F.; Taherdoost, H.; Chaeikar, S.S.; Zeidanloo, H.R. A novel approach for genetic audio watermarking. *J. Inf. Assur. Secur.* **2010**, *5*, 102–111.
33. Zamani, M.; Manaf, A.B.; Abdullah, S.M.; Chaeikar, S.S. Correlation between PSNR and bit per sample rate in audio steganography. In Proceedings of the 11th International Conference on Signal Processing, Montreal, QC, Canada, 2 April 2012; pp. 163–168.
34. Chaeikar, S.S.; Ahmadi, A.; Ensemble, S.W. Image steganalysis: A low dimension method for LSBR detection. *Signal Process. Image Commun.* **2019**, *70*, 233–245. [CrossRef]
35. Chaeikar, S.S.; Ahmadi, A. SW: A blind LSBR image steganalysis technique. In Proceedings of the 10th International Conference on Computer Modeling and Simulation, Sydney, Australia, 8 January 2018; pp. 14–18.