Support Vector Machines in Smile detection: A comparison of auto-tuning standard processes in Gaussian kernel

1st João Gondim
Institute of Computing
Federal University of Bahia
Salvador, Brazil
gondimjoaom@gmail.com

2nd Mateus Maia
Hamilton Institute
Maynnooth University
Maynooth, Ireland
mateus.maiamarques.2021@mumail.ie

3rd Ana Caroline Lopes Rocha
Institute of Psychiatry
University São Paulo
São Paulo, Brazil
analopes_rocha@hotmail.com

4th Felipe Argolo
Institute of Psychiatry
University São Paulo
São Paulo, Brazil
felipe.c.argolo@hotmail.com

5th Anderson Ara
Department of Statistics
Federal University of Paraná
Curitiba, Brazil
ara@ufpr.br

6th Alexandre Andrade Loch
Instituto de Psiquiatria
Universidade de São Paulo
São Paulo, Brazil
alexandre.loch@usp.br

***Abstract*—Support Vector Machines are a set of machine learning models that have great performance in several tasks as well as on image classification and object recognition. However, the proper choice of model's hyperparameters has a great influence on the outcomes and the general capacity performance. In this paper, we explore some different traditional auto-tuning processes to estimate σ hyper-parameter for SVMs Gaussian kernel. These processes are common and also implemented on standard software of data science languages. The paper considers some different situations on smile detection. The results are composed by simulation study, two benchmark image applications and a real video data application.**

***Index Terms*—SVM, Gaussian Kernel, tuning, image detection, smile.**

I. Introduction

Support Vector Machines (SVMs) are machine learning methods in which input vectors are non-linearly mapped to a higher dimensional feature space where a decision boundary is constructed [6]. Its good capacity performance for classification and regression tasks made SVMs widely used for solving many image classification problems [11], [14],[15]. In the mental health field, it can be used, for instance, to recognize mood states from individuals. This is especially important given the stigma that is usually associated with mental disorders where through a subtle technology to recognize someone's mood states it would be possible to provide an earlier diagnostic. Shivaswamy *et al.* [19] attribute the success of support vector models mainly to four factors: i) rooted in the statistical learning theory, SVMs possess superior generalization capacity, i.e: yields a smaller generalization error to predict new observations; ii) a globally optimal solution is obtainable by solving a convex optimization problem, while the problems of local minima crack other contemporary approaches, such as neural networks; iii) using the so-called kernel trick, it is convenient to solve non-linear problems in arbitrarily high dimensional feature spaces; iv) only a part of training samples are involved in solution representation.

Even though SVMs have good prediction performance, they are sensitive to the hyperparameters used to fit the model, hence the need to proper set them before training [13]. Defining SVMs' hyperparameters might be time consuming depending on how the search is done, for example, Grid Search (GS) evaluates the performance of a predefined range of parameters in order to select the one with the best metrics.

GS for hyper-parameter tuning can take less time when there is a small range of values to search. In [4], the performance of different values for the hyper-parameter σ was evaluated to provide researchers with good initial choices for the search of a proper value, after experimenting with different datasets, they came up with an ideal starting point by defining Eq. 1,

$$\sigma_* = N \cdot M, \tag{1}$$

as a good value, to begin with, being *N* the dimensionality of the data and *M* the magnitude of the coordinates of the data.

In [5] the hyper-parameter γ (a different way to represent the width spread of the Gaussian kernel as we will explain later) is defined as Eq. 2:

$$\gamma = \frac{1}{\text{Number of features}}. \tag{2}$$

In this paper we investigate these two auto-tuning processes using two datasets of smiling and not smiling faces and evaluated on real data application gathered from a psychiatric study. The next sections are organized as follows: Section II describes the methodology, Section III describes the experimental setup, database and features used and the setup of our machine,

Section IV shows the results obtained when applying each investigated experiment and after that we conclude the paper with some final considerations.

II. Support Vector Machines

SVMs classifies feature vectors by use of constructed optimal hyperplanes, a decision surface on high dimensional spaces that ensures high generalization for the network. The selected hyperplane is the one with greatest separation space between selected margin vectors, that can be written as some linear combination and represented by w, it can than classify a feature vector x, with b being a constant parameter. The hyperplane is defined as [6]:

$$w \cdot x + b = 0.$$

While linear data can be easier to classify, we often face problems where feature vectors are not linearly separable and, therefore, need nonlinear SVMs where the feature vectors input to the network model are mapped by use of a prior chosen function, a kernel [20] $\boldsymbol{K}(\boldsymbol{x}_i, \boldsymbol{x}_j)$ such that $K(\mathbf{x_i}, \mathbf{x_j}) = \phi(\mathbf{x_i}) \cdot \phi(\mathbf{x_j})$ replaces the inner product in the original networks machines formulation.

A map function ϕ operates as shown in

$$\phi : R^d \rightarrow H,$$

where the dimension of H is higher than d with the hyperplane now represented as

$$w \cdot \phi(x) + b = 0.$$

There are some established kernels used as mapping functions: linear, polynomial or Gaussian [9].

A. Gaussian kernel

In this paper, we investigate the use of the Gaussian kernel (also called Gaussian Radial Basis Function, or Gaussian RBF). The Gaussian kernel can be written as

$$k(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right), \quad (3)$$

with σ defining the width of the kernel, other representation of the Gaussian kernel is $k(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2)$, where $\gamma = \frac{1}{2\sigma^2}$.

The Gaussian kernel is responsible to transform the original feature space into an infinite-dimensional representation. The projections from the transformed data are given in an exponentially decaying function from the distances among the instances. Figure 1 represents how the choice of σ can influence the level of this behaviour. For greater values of σ the ratio is decaying slower, while for smaller values there is a sharp decrease.

B. Auto-Tuning

Among the types for auto-tuning a model parameters there are: the cost constant C, the exponent parameter of the polynomial kernel and the σ parameter of the Gaussian kernel [7]. We focus on the sigma parameter in our experiments.

The influence of σ on the Gaussian kernel is shown in Fig. 1, which expresses the behavior of the Eq. 3 and how sigma regularizes the squared difference $\|x_i - x_j\|^2$ in the Gaussian Kernel. Therefore, it may scale (or descale) distance values between vectors in the hyperlane projections.

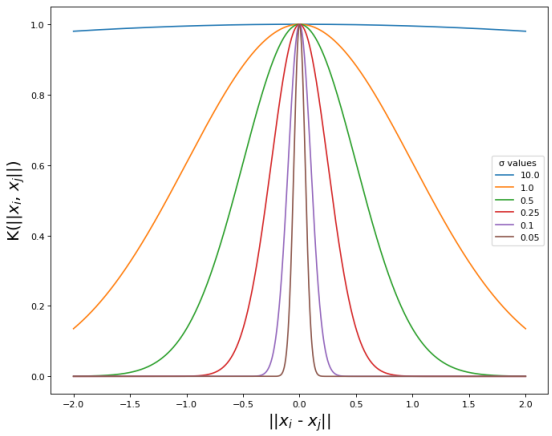

Fig. 1. Influence of σ on the Gaussian kernel width spread. Source: prepared by the authors.

If σ is assigned to a big value ($\gamma \rightarrow 0$) it might lead to underfit as every instance is classified into one class and if σ is close to zero ($\gamma \rightarrow \infty$) the SVM might overfit, as all the training instances are used as support vectors [17], as shown in Fig. 2. As seen, choosing the right value to σ is crucial to the proper training of an SVM model using a Gaussian kernel. On [4] experiments, (1) is shown to be a good value to start searching for the ideal value, but, for better-searching purposes, a range is set to ease where to look at. This range can be obtained by computing the values of $\|x_i - x_j\|^2$ and using the 0.1 and 0.9 quantiles to start searching.

Henceforth, the hyper-parameter σ, or γ equivalent, auto-tuning processes considered in this paper are named and defined as follows:

- ***auto***: definied as Eq. 2 or $\sigma = \frac{\sqrt{2N}}{2}$ and used in [5];
- ***median***: the median of the range defined by [4];
- ***90% quantile***: the 0.9 quantile value of the range from [4];
- ***quantile mean***: the mean of the 0.9 quantile and 0.1 quantile as shown in [4].

III. Simulation Study

To compare the standard methods of σ auto-tuning concerning their predictive capacity, we consider an artificial data generation with size 2000, different numbers of features ($n_f \in (10, 30, 100)$) and different proportion of class 1 balance ($\%class_1 \in (50, 10)$). The features were generated from a multivariate normal distribution as $\mathbf{X}_{Class1} \sim N_{n_f}(\mathbf{0}, \Sigma)$ and

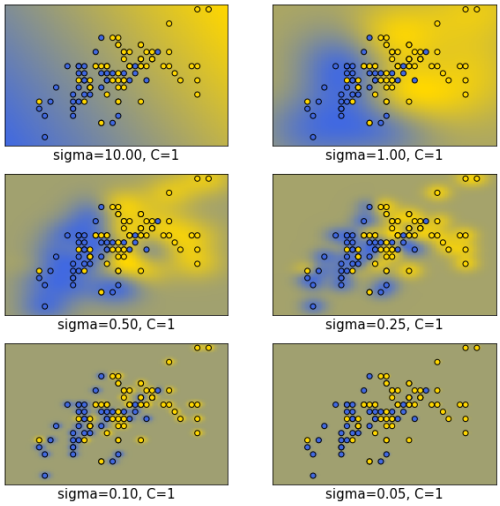

Fig. 2. Influence of σ on a SVM's decision function. As σ decreases, the model get more fitted to the data. Source: Adapted from [18] and prepared by the authors

$\mathbf{X}_{Class0} \sim N_{n_f}(\mathbf{1}, \Sigma)$, where Σ is the covariance matrix with 4 variance and 2 covariance for each component. All experiments were performed on a Intel Core i7, 16GB RAM, 5.4.0 Linux kernel.

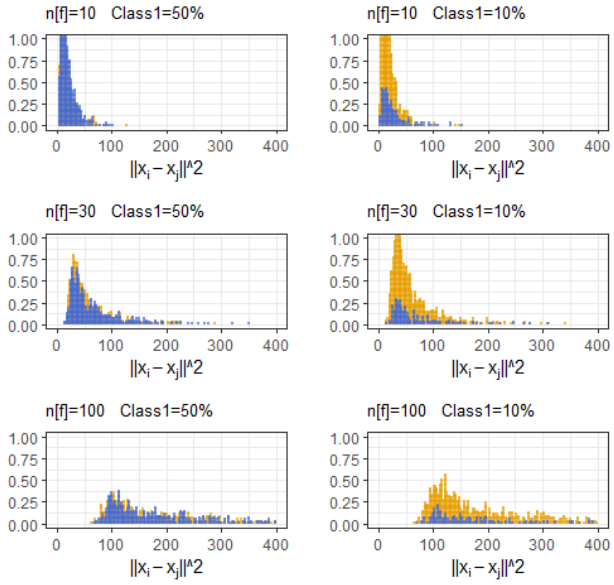

Fig. 3. Behavior of $||x_i - x_j||^2$ considering p, balanced (50%) and unbalanced (10%) artificial data. In blue observation on the same classes and in yellow observations on different classes

We used a repeated holdout validation method with 100 repetitions. In other words, to avoid having any influence on the random choice of train an test sets, 100 random shuffling of 70/30 splits were created. The performance measure used for reporting the simulation experiment is the Brier score loss (BS) [3]. The Brier score is a rigorous performance function used to measure the general accuracy of probabilistic estimates

TABLE I
AUTO-TUNING PROCESSES WITH THE LOWEST BRIER SCORE ON THE SIMULATION STUDY

Setup	*auto*	*quantile mean*	*median*	*90% quantile*
$n_f = 2$				
50%	3	1	22	74
10%	42	28	13	17
$n_f = 30$				
50%	0	0	0	100
10%	40	16	11	33
$n_f = 100$				
50%	9	27	11	53
10%	49	7	18	26

and the real classes [16]. The BS for binary classification is given by the Equation:

$$\text{BS} = \frac{1}{n_{\text{samples}}} \sum_{i=1}^{n_{\text{samples}}} (y_i - \hat{p}_i)^2,$$

where $y_i \in \{0, 1\}$ is the real class value, $\hat{p}_i \in (0, 1)$ is the estimated probability by the SVM method for belong the $class_1$ and $n_{samples}$ is the sample size of the test data.

Fig 3 shows the estimated distribution behavior of the distance among each generated data observation in each setup. We can observe that with an increase in the number of features such distribution moves further and further away from zero, a problem related to the curse of dimensionality. Furthermore, and for this case, the increase of the dimension turns the distribution more and more right-skewed. The unbalance also affects the behavior of distances between observations that belong or do not belong to the same class. With a high imbalance, we have even greater asymmetry for observations that belong to the same class. These behaviors are explanatory for the results in Table I. The *auto* method is more efficient to the unbalanced dataset other methods are competitive in the balanced dataset.

IV. SMILE DETECTION

A. Benchmark Applications

To make the experiments, we used two datasets composed of smiling and not smiling faces, example are shown in Fig. 4. The first subset (Dataset 1) corresponds to selected smiling faces from [8] and is available online[1], the second (Dataset 2) is also a subset of [8], but smaller and balanced, it was gathered by [1], Table II shows the numbers of both datasets where it can be seen the imbalance of Dataset 1.

We used two features for training and experimenting: Histogram of Oriented Gradients and the landmarks of each face (retrivied using [10]), as seen in Fig. 5. The validation was also realized using a 100 repeated holdout. The metrics used for reporting these experiments are BS, cited in Section III,

[1]https://github.com/hromi/SMILEsmileD

Fig. 4. Examples of faces used on training. Source: [8].

TABLE II
QUANTITIES OF THE SMILE DATASETS.

Dataset	Smile	Not smile	n	p% Smiles	n_f
Dataset 1	3690	9475	13165	28.03	648
Dataset 2	600	600	1200	50	648

as well as the common measures Accuracy (ACC), F1-Score (F1), Matthews correlation coefficient (MCC) [12].

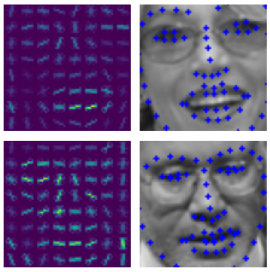

Fig. 5. Features used. Source: adapted from [8] and prepared by the authors.

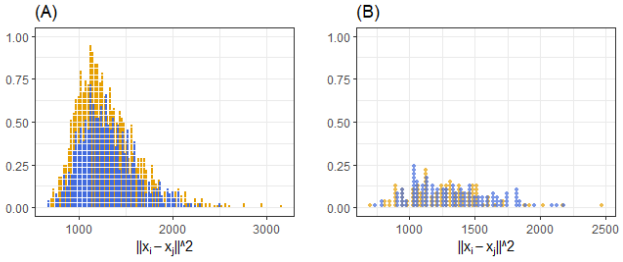

Fig. 6. Behavior of $||x_i - x_j||^2$ to the both data real data: (A) Dataset 1 and (B) Dataset 2.

On our experiments, we trained all the 100 holdouts and show the results with the mean of ACC, F1-score, MCC and BS, results are shown in Table III. We also report the numbers of the probabilities estimated by the trained models with different hyperparameters tuning in a boxplot format, shown in Fig. 7 for the imbalanced dataset and Fig. 8 for the balanced dataset. The boxplot images show that the probability estimation of a model depends directly on whether the dataset used for training is balanced or not. For every auto-tuning method used, the median of inference's probabilities was closer to 100% on the models trained with Dataset 2 that also presents more flattened boxes, indicating less variability on the final predictions' probabilities. These probabilities corroborate with the results shown by Table III where all the metrics are better for Dataset 2, specifically BS, that analyses the probabilities given by a model.

TABLE III
RESULTS

Method	ACC	F1-Score	MCC	BS
Dataset 1				
auto	**91.37%**	**83.97%**	**78.23%**	**6.31**
median	91.18%	83.64%	77.76%	6.42
90% quantile	91.28%	83.82%	78.01%	6.36
quantile mean	91.19%	83.66%	77.78%	6.41
Dataset 2				
auto	95.06%	95.01%	90.12%	3.62
median	95.04%	94.99%	90.08%	3.64
90% quantile	**95.08%**	**95.03%**	**90.17%**	**3.60**
quantile mean	95.05%	95.01%	90.11%	3.63

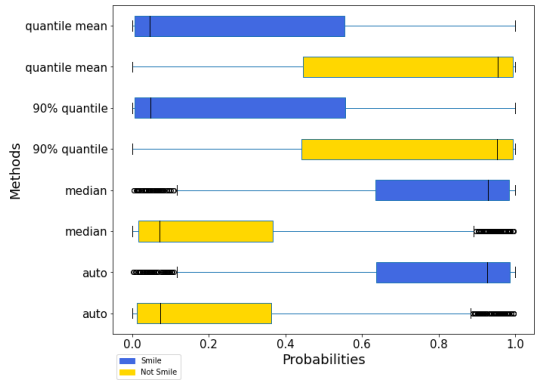

Fig. 7. Probabilities of each estimation on one holdout test set for Dataset 1.

In Table IV we compare every model trained with each method with themselves, evaluating them by counting how many times a model obtained greater metrics. Here, even

TABLE IV
NUMBER OF TIMES IN WHICH A METHOD OBTAINED GREATER METRICS THAN OTHERS. EACH LINE SUMMARIZES 100 HOLDOUT VALUES.

Measure	*auto*	*median*	*90% quantile*	*quantile mean*
Dataset 1				
ACC	78	2	15	5
F1-Score	72	2	21	5
MCC	74	2	19	5
BS	98	0	2	0
Dataset 2				
ACC	55	33	9	3
F1-Score	50	37	10	3
MCC	49	37	11	3
BS	40	14	33	13
Total	516	120	128	37

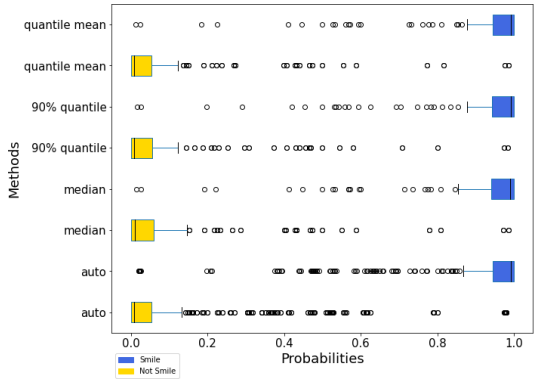

Fig. 8. Probabilities of each estimation on one holdout test set for Dataset 2.

though *90% quantile* had better means, *auto* has higher counts. To aid on the selection of the proper method for hyper-parameter we also present Table V showing the total number of outliers present on each of the above boxplots. This table depicts the higher number of outliers probabilities present on models trained with imbalanced datasets, *90% quantile* and *mean quantile* have 0 outliers but also taller boxes. From Table III is possible to observe that the performance over the first dataset was better using the $auto$ method, when compared with the other approaches. Given the unbalance among classes, this result was expected, since that yielded an asymmetry in the density of distances resulting in poor initial estimates for the other methods, as mentioned in the Simulation section. On the other hand, the same table shown a good performance of 90% quantile, but competitive among the others, once for the second dataset the density of distances is symmetric.

TABLE V
TOTAL NUMBER OF OUTLIERS FOR EACH METHOD.

Dataset	***auto***	***median***	***90% quantile***	***quantile mean***
Dataset 1	58	58	0	0
Dataset 2	25	21	23	23

B. Real Data application

To test the tuning methods studied, we applied the models trained with all the data from both datasets with 4 different tuning Methods: 1. Features from Dataset 1 trained using σ *auto*; 2. Features from Dataset 1 trained using σ *median*; 3. Features from Dataset 1 trained using σ *90% quantile*; 4. Features from Dataset 1 trained using σ *quantile mean*; 5. Features from Dataset 2 trained using σ *auto*; 6. Features from Dataset 2 trained using σ *median*; 7. Features from Dataset 2 trained using σ *90% quantile* and 8. Features from Dataset 2 trained using σ *quantile mean*.

We measured the count of smiles on videos collected from a psychiatric study. In the videos, interviewed participants were asked about recent dreams, their relationship with their parents, to talk about the day before or to tell a story based on images shown. Fig. 9 shows an example of a face (image half blurred to preserve identity of patient) found in the videos.

To make the experiments, we used [2] to get each frames from a video and transform the image to grayscale; using [10] library we detect the person's face and the landmarks used for inference among the face's HOG feature vector, after that we use each SVM classifier to find out if the features vector represents a smiling face or not, if we find a smiling face for 10 frames straight, we add one smile to the count.

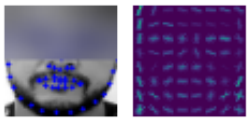

Fig. 9. Features of real case sample.

The data set includes 30 real patient videos recorded by Laboratorio de Neurociencias (LIM 27) (http://neurociencias.org.br/) from University of Sao Paulo (USP), Brazil. On the experiment we compared the number of smiles detected when using these considered the 8 methods. For better understanding of how each method behaves on real images, we calculated the Mean Average Error (MAE) for every 30 videos with:

$$\sum \frac{|y_{pred} - y_{true}|}{\text{Number of videos}}$$

The results with models trained with Dataset 1 are presented in Table VI and Table VII. On videos with no smiles, all the models trained with Dataset 1 performed well, but, when there are smiles present it is shown that different σ calculations result in different counting outcomes, also, on real data, the model with *auto* σ calculation on Dataset 1 still outperformed the other methods. On Dataset 2, the model performed poorly on real data (probably due to the lower quantity of images) with a maximum and minimum MAEs of 14.33 (Method 5) and 13.73 (Method 6) respectively.

V. FINAL COMMENTS

Auto-tuning methods are a fast and very used approach to fit machine learning methods, as there is a limitation on the investigation time to find the proper hyperparameters, e.g. GS. On the context of SVMs, the auto-tuning methods are similar, but there are differences among them. The *auto method* ($\sigma = \frac{\sqrt{2N}}{2}$) is, in general, superior compared with other ones, specially when there is a great number of features and high imbalanced data. However, methods based on σ estimation are potent for balanced data with no strong asymmetry on the Euclidean distance distribution, among them, *90% quantile* is the most adequate when such asymmetry is strong. Therefore, is important to highlight the importance of analyzing the density from the distances for a given dataset in order to achieve the most suitable estimation for ϕ. Finally, despite the presented guideline to select a good initial computation of σ, for obtaining the best performance over it is always recommended the use of GS or Bayesian optimisation methods to select the best value for the hyperparameters, despite the computational burden.

TABLE VI
RESULTS COMPARING THE NUMBER OF SMILES DETECTED ON EACH VIDEO AND THE GROUND TRUTH ON DATASET 1

ID	Method 1	Method 2	Method 3	Method 4	G. Truth
1	1	1	1	1	**4**
2	8	9	10	9	**7**
3	0	0	0	0	**6**
4	0	0	0	0	**5**
5	5	5	5	5	**8**
6	0	0	0	0	**0**
7	8	9	0	0	**1**
8	0	0	0	0	**1**
9	0	0	0	0	**0**
10	0	0	0	0	**0**
11	0	0	0	0	**1**
12	0	0	0	0	**0**
13	0	0	0	0	**0**
14	0	0	0	0	**0**
15	0	0	0	0	**0**
16	14	18	20	18	**5**
17	0	0	0	0	**0**
18	0	0	0	0	**0**
19	0	0	0	0	**2**
20	0	0	0	0	**0**
21	0	0	0	0	**1**
22	0	0	0	0	**2**
23	0	0	0	0	**0**
24	2	6	7	6	**3**
25	0	0	0	0	**2**
26	0	0	0	0	**1**
27	0	0	0	0	**0**
28	0	0	0	0	**1**
29	0	0	0	0	**3**
30	1	1	1	1	**2**

TABLE VII
MEAN AVERAGE ABSOLUTE ERROR FOR EACH METHOD.

Method	Method 1	Method 2	Method 3	Method 4
MAE	**1.67**	1.93	1.83	1.7

ACKNOWLEDGMENT

This work was supported by the Wellcome Trust [223139/Z/21/Z]. M.M.'s work was supported by a Science Foundation Ireland Career Development Award grant 17/CDA/4695.

REFERENCES

[1] O. Arigbabu. Smile detection from face images. Mendeley Data, 2017.

[2] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[3] G. W. Brier. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78:1–3, 1950.

[4] B. Caputo, K. Sim, F. Furesjo, and A. Smola. Appearance-based object recognition using svms: which kernel should i use? In *NIPS Proceedings, Whistler*, volume 2002, 2002.

[5] C.-C. Chang and C.-J. Lin. Libsvm: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27, 2011.

[6] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[7] L. H. Hamel. *Knowledge discovery with support vector machines*, volume 3. John Wiley & Sons, 2011.

[8] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. In *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*, 2008.

[9] T. Jebara. Multi-task feature and kernel selection for svms. In *Proceedings of the twenty-first international conference on Machine learning*, page 55, 2004.

[10] D. E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[11] P. Li, L. Dong, H. Xiao, and M. Xu. A cloud image detection method based on svm vector machine. *Neurocomputing*, 169:34–42, 2015.

[12] M. Maia, J. S. Pimentel, I. S. Pereira, J. Gondim, M. E. Barreto, and A. Ara. Convolutional support vector models: Prediction of coronavirus disease using chest x-rays. *Information*, 11(12):548, 2020.

[13] R. G. Mantovani, A. L. Rossi, J. Vanschoren, B. Bischl, and A. C. De Carvalho. Effectiveness of random search in svm hyper-parameter tuning. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. Ieee, 2015.

[14] U. Maulik and D. Chakraborty. Remote sensing image classification: A survey of support-vector-machine-based advanced techniques. *IEEE Geoscience and Remote Sensing Magazine*, 5(1):33–52, 2017.

[15] R. Mishra, S. Meher, N. Kustha, and T. Pradhan. A skin cancer image detection interface tool using vlf support vector machine classification. In *Computational Intelligence in Pattern Recognition*, pages 49–63. Springer, 2022.

[16] J. Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[17] C. Savas and F. Dovis. The impact of different kernel functions on the performance of scintillation detection based on support vector machines. *Sensors*, 19(23):5219, 2019.

[18] Scikit-learn. RBF SVM parameters. https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html, 2008. [Online; accessed 10-October-2021].

[19] P. K. Shivaswamy, W. Chu, and M. Jansche. A support vector approach to censored targets. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 655–660. IEEE, 2007.

[20] W. Wang, Z. Xu, W. Lu, and X. Zhang. Determination of the spread parameter in the gaussian kernel for classification and regression. *Neurocomputing*, 55(3-4):643–663, 2003.