D_iPSVM: Polynomial Kernel-Free Support Vector Machine

Soumadip Saha
Integrated M.Sc in Physics
IIT kharagpur
Kharagpur-721302, India
soumadipsaha2000@gmail.com

Meghashrita Das
Agricultural and Food engineering
IIT kharagpur
Kharagpur-721302, India
meghashrita99@gmail.com

Baishali Sow Mondal
Agricultural and Food engineering
IIT kharagpur
Kharagpur-721302, India
baishalism@gmail.com

Sobhan Sarkar
Information Systems & Business Analytics
IIM Ranchi
Jharkhand-834008, India
sobhan.sarkar@iimranchi.ac.in

J. Maiti
Industrial & System Engineering
IIT Kharagpur
Kharagpur-721302, India
jhareswar.maiti@gmail.com

***Abstract*—Support Vector Machine (SVM), a fast and dependable classification algorithm that performs very well with an amount of data to analyze in the field of machine learning. However, the kernel-based dependency of the SVM algorithm requires a long time to compute the support vectors for non-linear datasets. To remove the kernel, several types of functions are used with SVM. In earlier attempts, addition of kernel free approach in SVM caused major problems like repetitive feature space and long run time complexity. So, a non-linear function is introduced in this paper, namely i^{th} degree polynomial (D_iP). This function can directly identify non-linear features in a dataset. A kernel-free SVM model is proposed using D_iP function named as D_iPSVM in this paper. In D_iPSVM, the input feature space is first taken into a new higher order feature space by using the multi-variable Taylor's expansion of the input features up to i^{th} order to evaluate all the non-linear correlations among the input features. This method is implemented to check which specific ordered polynomial can increase the accuracy of the kernel free SVM model. Finally, sequential minimal optimization (SMO) is used for fast convergence to reduce the superiority of D_iPSVM is demonstrated over ten benchmark categorical and continuous datasets obtained from UCI machine learning repository. Results showed that D_iPSVM gives better accuracy and faster convergence than kernel-based SVM algorithms used on real life datasets.**

***Index Terms*—Support vector machine, Taylor's expansion, Sequential minimal optimization.**

I. Introduction

In the field of artificial intelligence (AI) and machine learning (ML), classification and regression-based models have a significant impact. Vapnik [3] proposed the support vector machine (SVM), which is a widely useful and successful method in those domains. The standard linear SVM model which is proposed by Vapnik separates data points in different classes using a hyper plane, while maximizing the margin of different classes. It works well on those data sets which have a linear decision boundary, but not on non-linearly decision boundaries. Therefore, by expanding the input feature space into a higher nonlinear dimension, various kernel functions [15] (linear, gaussian, polynomial) are applied with non-linear datasets.

With various kernel functions, it has attained numerous successes in diverse application areas, however, it also has a few drawbacks. First, in order to create the hyperplane, the kernel function needed to be chosen manually which is not a feasible method. Second, since solving the dual equation, the inverse of the kernel matrix is needed to be solved and it becomes computationally expensive. Third, if the kernel matrix is singular, it affects the learning rate of the model. Although the third issue has been resolved by using sequential-minimal-optimization (SMO) algorithm [16]. In order to resolve the drawbacks of standard kernel SVM models, a method is proposed to make a kernel-free SVM model in this paper. The objective of this study is to develop an algorithm for classification and regression, namely kernel free D_iPSVM which can handle the aforementioned issues of linear SVM [2] and kernel-based SVM [14]. The idea is to use i^{th} order polynomial surface for nonlinear separation directly. First, the feature making process is redesigned in such a way that it unable to build repetitive features in the feature space and increase the model complexity. Second, to address the issue of low accuracy score, a new cost function [7] is developed. For quick convergence, the SMO algorithm [9] with heuristics is also used.

A total of ten benchmark continuous and categorical datasets are used to test the hypothesis. Multi-class classification [12] problems can easily be converted into binary classification using one vs all method [17]. Hence, our primary focus in this paper is to use this kernel-free SVM model for binary classification which can be easily generalized for multi class classification problem.

The main contributions of our study are as follows:

(i) To combat the feature space overfitting [11] problem, the

polynomial function is expanded by using the Taylor's expansion formula. Use of Taylor's expansion [8] can reduce the dimension of new individual features by a huge number and decrease the over-fitting.

(ii) Standard quadratic programming problem [5] causes to slow down the optimization process [4] for large datasets. Thus, the well-known SMO algorithm [16] has been implemented on the proposed model for computational efficiency [19]. As feature space increases, it will help to reduce the time complexity of the model for large datasets.

The structure of the paper is organized as follows: Brief demonstration of SVM for binary classification in Section II. The proposed D_iPSVM algorithm is explained in Section III. Section IV shows in-depth experimental results of the D_iPSVM algorithm and a comparison of its performance. Finally, the conclusion including scope for future works are discussed in Section V.

II. SVM with Binary Classification

Binary classification helps to find a hyper plane which will separate different classes while maximizing the distance between the boundary (margin) of different classes [12]. Let D be a dataset consisting of two classes in Eq. (1).

$$D = (x_i, y_i),\ i = 1, \ldots, N\ \&\ x_i \in R^N,\ y_i \in \{-1, 1\} \quad (1)$$

where N denotes the count of examples, n denotes features count in each data point, $x_i = [x_{i1}, \ldots, x_{in}]^T \in R^{\mathrm{n}}$ is an example of a data point residing in an n-dimensional feature space. and y_{i} is the label of i^{th} of the data point.

When all of the data points cannot be separated in their respective classes or while doing that the model gets over-fit, the soft-margin concept is introduced [3] using the slack vector [19] in Eq. (2),

$$\xi = [\xi_1, \ldots, \xi_N] \in R^N \quad (2)$$

To reduce the overfitting of the model some data points which are difficult to classify without increasing the model complexity, are allowed to be labeled as misclassified data points. The modified cost function for soft-margin SVM [21] (SSVM), in Eq. (3).

$$min\ \frac{1}{2}||u||_2^2 + C\sum_{i=1}^{N}\xi_i. \quad (3)$$

Here, $C > 0$ penalises the model for misclassified examples and u is the weights for the model.

Kernels like, gaussian (RBF) [20] and quadratic [13] have been used on various real life data sets. kernel-based SSVM is also a convex QP equation [10], which is defined as the following Eq. (4).

$$\begin{gathered}\min_{\alpha}\ \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j K(x_i, x_j) - \sum_{i=1}^{N}\alpha_i \\ \sum_{i=1}^{N}\alpha_i y_i = 0 \text{ where } 0 \le \alpha_i \le C\ \forall i = 1, \ldots, N\end{gathered} \quad (4)$$

After creating the cost function in Eq. (4), in order to optimize the weights, the cost function must be minimized. This process can be done in two ways, analytically or numerically. SMO algorithm [18] uses the numerical approach to optimize the weights discussed in the next section.

III. D_iPSVM Model

In this section, first the mathematical background is explained then in the second section a new slack variable to calculate the cost function is introduced and the final D_iPSVM Model is proposed.

There are several types of i^{th} order polynomial function which can be used as a kernel-free function in SVM.

1) Polynomial Function: For calculation the default value of i is taken as 4 (Bi-quadratic polynomial) [1]. Moreover the algorithm is designed in such a way that i can take values from 1 to any range. Let G be a 4^{th} order polynomial from $R^n \to R$ in in Eq. (5),

$$G(x) = \frac{1}{2}\left(\frac{1}{2}||Bx - c||_2^2 - d\right)^2 + \frac{1}{2}x^T Ax + b^T x + q \quad (5)$$

where $B \in R^{m\times n}$, $c \in R^m$, $d \in R$, $A \in S^n$, $b \in R^n$, $q \in R$. Let us define $s_i = lvec(x_i)$, $w_B = hvec(B^T, B)$, $w_{Bc} = c^T B$, and $c_d = \frac{1}{2}c^T c - d$, then we have $\xi_i = s_i^T w_B - x_i^T w_{Bc} + c_d$. Let us define, $z_i = [s_i^T, x_i^T, 1]$, and $w_\xi = [w_B^T, w_{Bc}^T, c_d]^T$. Therefore G can be in Eq. (6),

$$G(x_i) = \frac{1}{2}z_i^T w_\xi w_\xi^T z_i + \frac{1}{2}x_i^T A x_i + b^T x_i + q \quad (6)$$

Where function F has two quadratic terms, z_i and x_i with respect to $R^{\frac{n(n+1)}{2}+2n+1} \to R$ and $R^n \to R$, respectively. Similarly let $k = \frac{n(n+1)}{2} + n + 1$ and,
$w_W = hvec(w_\xi, w_\xi^T) \in R^{\frac{k(k+1)}{2}}$; $w_A = hvec(A) \in R^{\frac{n(n+1)}{2}}$. Therefore, we can define some new variables v and r_i as follows:

$$v = [\boldsymbol{w}_W, \boldsymbol{w}_B]^T \in R^{\frac{k(k+1)+n(n+1)}{2}+n}$$
$$r_i = [\eta_i, s_i]^T \in R^{\frac{k(k+1)+n(n+1)}{2}+n}$$

Similarly, $G(x_i)$ is equivalent to the linear function G_l with respect to r_i and x_i in $R^{\frac{l(l+1)+n(n+1)}{2}+n}$,

$$G(x_i) = G_l[r_i, x_i]^T = r_i^T v + x_i^T b + q$$

This model of feature extraction up-to 4^{th} degree was proposed by Gao, Fang, Luo & Medhin [6]. In the proposed model, a new and more general way to create the feature space is introduced and explained next.

2) Redefinition of feature space: A multi-variable potential function will transform the input feature space to a higher dimension in order to find the non-linear relations among them. These relations are described in the following Eq. (7), Eq. (8), Eq. (9),

$$x_i = [x_i^1, x_i^2] \quad (7)$$
$$s_i = lvec(x_i) = [x_i^{1^2}, x_i^1 x_i^2, x_i^{2^2}] \quad (8)$$
$$z_i = [s_i, x_i, 1] = [x_i^{1^2}, x_i^1 x_i^2, x_i^{2^2}, x_i^1, x_i^2, 1] \quad (9)$$

While expressing η_i like above, there are some terms repeating, those are $x_1^2x_2^2$, $x_1^2x_2$, and others. In order to resolve this issue, one new model is proposed in this paper to create the feature space in D$_i$PSVM model. The idea is to use the expansion up to a degree defined by the user (default 4th degree) and thus, treating the non-linear function as a polynomial. In general, any function $f(x) : [x \in R^N]$ can be expanded into a polynomial using multi-variable Taylor expansion, as explained in Eq. (10),

$$f(x,y) \approx Q(x,y) = f(a,b) + f_x(a,b)(x-a) + f_y(a,b)(y-b) + \frac{f_{xx}(a,b)}{2}(x-a)^2 + f_{xy}(a,b)(x-a)(y-b) + \frac{f_{yy}(a,b)}{2}(y-b)^2 \quad (10)$$

This function is an example of such an expansion up to 2^{nd} order. Therefore, the error term will be of order 3. Since D$_4$P is a 4^{th} order polynomial, therefore the error term will be of order 5. Let $x = [x_1, x_2]$ be a datapoint consisting of two input features. If f is the original function to classify the datapoints, the Taylor expansion of f followed by Eq. (10) up-to 2nd order term is

$$\begin{aligned} f([x_1,x_2]) &= f([a,b]) + f_{x_1}([a,b])(x_1-a) + f_{x_2}([a,b]) \\ &\quad (x_2-b) + \frac{f_{x_1x_1}([a,b])}{2}(x_1-a)^2 \\ &+ f_{x_1x_2}([a,b])(x_1-a)(x_2-b) + \frac{f_{x_2x_2}([a,b])}{2}(x_2-b)^2 + \dots \\ &= c + w_1x_1 + w_2x_2 + w_3x_1^2 + w_4x_1x_2 + w_5x_2^2 + \dots \\ &= c + W_1x + W_2x^2 + \dots \end{aligned} \quad (11)$$

where a, b, and c are constants, and W represents the coefficients of the polynomial expansion.
In Eq. (10), each term of RHS represents the polynomial terms of various degrees. The second term of the RHS in Eq. (10) is a polynomial of order 1 whereas, the third term represents a second degree polynomial.
Therefore, in order to find the unique terms of each individual polynomial, the idea of Taylor's expansion can be used. The expansion is defined as expressed in Eq. (12).

$$(x_1 + \dots + x_m)^n = \sum_{k_1+\dots+k_m=n} \binom{n}{k_1, \dots, k_m} \prod_{t=1}^{m} x_t^{k_t} \quad (12)$$

The number of individual terms in the above polynomial expansion are,

$$\binom{n+m-1}{m-1} = \frac{(n+m-1)!}{n!(m-1)!} \quad (13)$$

Therefore, when the function $f(x)$ is expanded up-to degree 4, the number of unique individual terms can be calculated using the summation over Eq. (13) for degree up-to 4.

$$\text{Total number of terms in the } D_4P = \sum_{n=0}^{4} \frac{(n+m-1)!}{n!(m-1)!} = \binom{m+4}{4} \quad (14)$$

In order to calculate the unique Taylor expansion terms of this polynomial expansion, python in-built package "Comparison-with-replacement" has been used which computes all the combinations in the feature space up-to the order 4 (Section III).

IV. Computational Results

This section discuses the computation results and shows the comparison of various different state-of-the-art models with D$_i$PSVM model and finds the best result for each datasets. The proposed D$_i$PSVM algorithm is first tested on several public benchmark datasets. The value i is changed to see which dataset provides a greater accuracy with a specific degree polynomial. A run-time improvement analysis is also performed with optimized model.

TABLE I: Data set description.

Datasets	Types	Instances	[1]CA	[1]DC
Iris	Classification	150	4	3
Wholesale	Classification	440	8	3
Credit	Classification	690	15	3
TaxInfo	Classification	1004	10	3
Weather	Classification	367	22	2
Glass	Regression	214	11	_
Wine	Regression	4898	12	_
Car evaluation	Regression	1728	6	_
Abalone	Regression	4177	8	_
Forestfire	Regression	517	13	_

[1]CA = Conditional attributes, [1]DC = Decision classes

A. Experiment Settings

Ten regression and classification-based datasets are taken to measure the performance of the proposed model. Changing the degree in D$_i$PSVM model, the accuracy and time taken to compute the weights are calculated for each datasets.
Table I shows a complete description of the datasets. In Table II, the accuracies of various algorithms including the proposed model are recorded and it is observed that D$_4$PSVM outperforms every algorithms. Table III describes the run-time comparison of D$_4$PSVM (section III(A)) without SMO and with SMO algorithm. It describes the benefits of using SMO algorithm over traditional quadratic programming optimization methods.

B. Plots and sub-plots

Fig. 1 contains runtime and accuracy for all regression type datasets and similarly, Fig. 2 contains runtime and accuracy for all classification type datasets. Fig. 3 describes the CPU

TABLE II: Accuracy comparison for various models.

Datasets	SVM linear kernel	SVM RBF kernel	SVM poly kernel	D_iPSVM
Iris	61.34	89.34	98.67	100
Wholesale	75.97	75	75	89.77
Credit	100	75	53.75	72.5
TaxInfo	29.37	38.61	29.7	68.15
Glass	91.58	66.35	78.5	95.34
Wine	99.72	99.47	99.53	99.84
Car evaluation	73.08	100	100	79.18
Abalone	100	100	100	79.18
Forestfire	70.64	63.7	63.7	95.54
Weather	96.17	83.06	83.06	100

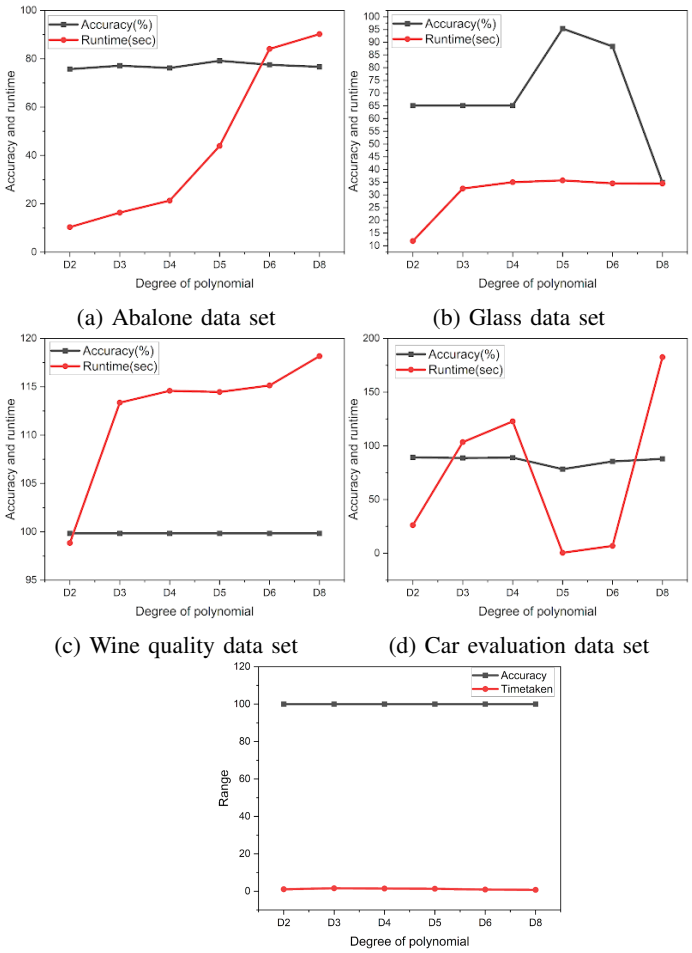
TABLE III: Runtime improvement calculation

Datasets	Runtime(s) without SMO	Runtime(s) with SMO
Iris	9.216	0.131
Wholesale	27.75	25.96
Credit	4.042	0.985
TaxInfo	274.36	8.7
Glass	14.713	2.323
Wine	896.259	0.98
Car evaluation	977.35	0.259
Abalone	1327.45	210.98
Forestfire	0.76	0.23
Weather	7.87	3.34

run-time comparison or how SMO algorithm improved the model. In Fig. 1, regression-based datasets have been trained on different order of the D_iPSVM models. The comparison shows that different models has a separation hyperplane of different order. The best degrees of the hyperplane function are finally used in Table II to compare with "RBF", "Linear" and "Polynomial" kernel-based SVM algorithms. Fig. 4 shows that D_iPSVM outperforms rest of the models with huge margin improvement. As explained in Table II and Fig. 3, runtime of D_iPSVM is further improved with the help of SMO algorithm.

V. Conclusion

A new kernel-free support vector machine algorithm is introduced in this paper using polynomial hyperplane to classify the non linearly separable datasets. The proposed D_iPSVM model performs better and also overcome the problems with kernel-based methods. This algorithm is tested with various public benchmark datasets to understand the required CPU time complexity and accuracy. Further SMO algorithm makes the model more optimized and computationally efficient.
Different datasets mentioned in this paper, follows different hyperplane. Therefore various degree of polynomials fits the various datasets. This can be achieved by this model by changing the degree. The best results of this model can be seen for credit score and prediction based datasets.
In the training process, it reduces the computation time since it doesn't need to compute the matrix elements in each optimization iteration and the entire non-linear function can be treated as simple linear function while consisting the same properties of non-linear decision boundaries. The proposed model in this paper still carries opportunity for future research.



(a) Abalone data set (b) Glass data set

(c) Wine quality data set (d) Car evaluation data set

(e) Forest data set

Fig. 1: Plots of regression-based results.

This model works better on small and medium sized data sets due to it's pre-computation. Therefore the first thing to do is that to find a better cache way to store the calculation for big datasets. Also this model can be used to classify images; this prospect of this model needs further investigation.

References

[1] Immanuel M Bomze, Chen Ling, Liqun Qi, and Xinzhen Zhang. Standard bi-quadratic optimization problems and unconstrained polynomial reformulations. *Journal of Global Optimization*, 52(4):663–687, 2012.

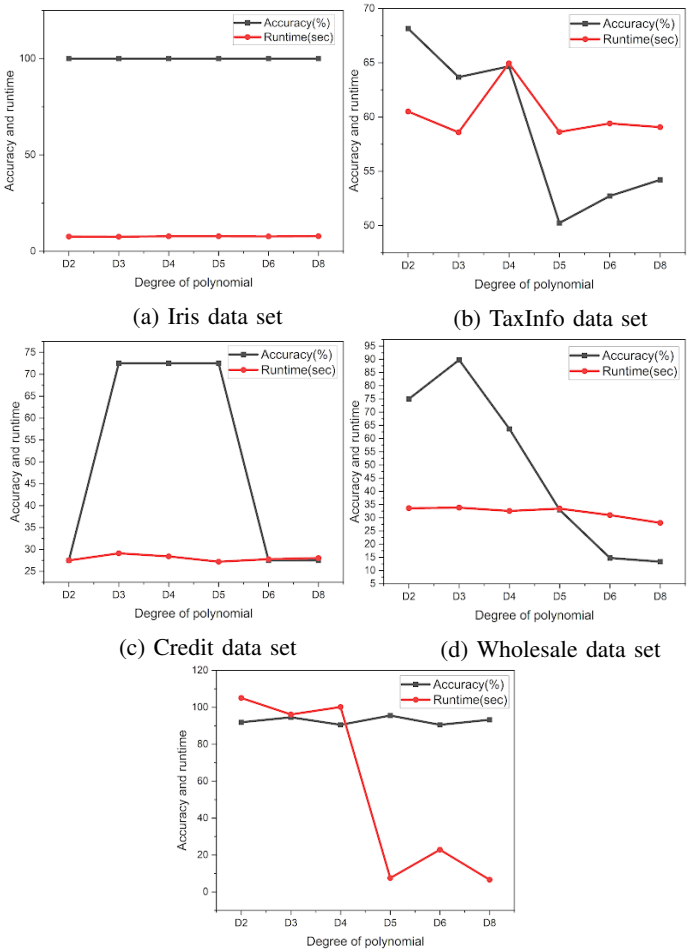
[2] Yin-Wen Chang and Chih-Jen Lin. Feature ranking using linear svm. In *Causation and prediction challenge*, pages 53–64. PMLR, 2008.

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

[4] Jian-xiong Dong, Adam Krzyżak, and Ching_Y Suen. A fast parallel optimization for training support vector machine. In *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, pages 96–105. Springer, 2003.

[5] Marguerite Frank, Philip Wolfe, et al. An algorithm for quadratic programming. *Naval research logistics quarterly*, 3(1-2):95–110, 1956.

[6] Zheming Gao, Shu-Cherng Fang, Jian Luo, and Negash Medhin. A kernel-free double well potential support vector machine with applications. *European Journal of Operational Research*, 290(1):248–262, 2021.



(a) Iris data set

(b) TaxInfo data set

(c) Credit data set

(d) Wholesale data set

(e) Weather data set

Fig. 2: Plots of classification-based results.

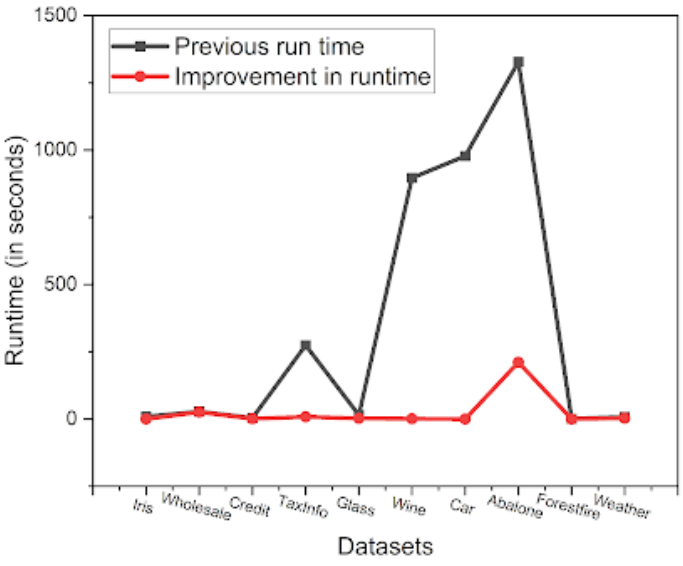


Fig. 3: Runtime comparison

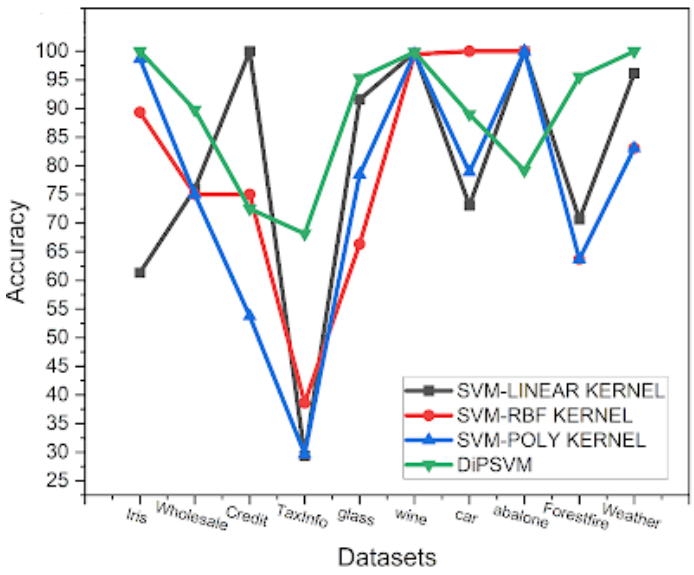


Fig. 4: Accuracy comparison

[7] Tarfa Hamed, Rozita Dara, and Stefan C Kremer. An accurate, fast embedded feature selection for svms. In *2014 13th International Conference on Machine Learning and Applications*, pages 135–140. IEEE, 2014.

[8] RP Kanwal and KC Liu. A taylor expansion approach for solving integral equations. *International Journal of Mathematical Education in Science and Technology*, 20(3):411–414, 1989.

[9] Tilman Knebel, Sepp Hochreiter, and Klaus Obermayer. An smo algorithm for the potential support vector machine. *Neural computation*, 20(1):271–287, 2008.

[10] Mikhail K Kozlov, Sergei P Tarasov, and Leonid G Khachiyan. The polynomial solvability of convex quadratic programming. *USSR Computational Mathematics and Mathematical Physics*, 20(5):223–228, 1980.

[11] Yong Liu, Shenggen Ju, Junfeng Wang, and Chong Su. A new feature selection method for text classification based on independent feature space search. *Mathematical Problems in Engineering*, 2020, 2020.

[12] Ajay Mathur and Giles M Foody. Multiclass and binary svm classification: Implications for training and classification users. *IEEE Geoscience and remote sensing letters*, 5(2):241–245, 2008.

[13] Ahmad Mousavi, Zheming Gao, Lanshan Han, and Alvin Lim. Quadratic surface support vector machine with l1 norm regularization. *arXiv preprint arXiv:1908.08616*, 2019.

[14] MN Murty and Rashmi Raghava. Kernel-based svm. In *Support vector machines and perceptrons*, pages 57–67. Springer, 2016.

[15] Arti Patle and Deepak Singh Chouhan. Svm kernel functions for classification. In *2013 International Conference on Advances in Technology and Engineering (ICATE)*, pages 1–9. IEEE, 2013.

[16] John Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. 1998.

[17] Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *The Journal of Machine Learning Research*, 5:101–141, 2004.

[18] Zhaonan Sun, Nawanol Ampornpunt, Manik Varma, and Svn Vishwanathan. Multiple kernel learning and the smo algorithm. *Advances in neural information processing systems*, 23:2361–2369, 2010.

[19] Fengzhen Tang, Peter Tiňo, Pedro Antonio Gutiérrez, and Huanhuan Chen. The benefits of modeling slack variables in svms. *Neural computation*, 27(4):954–981, 2015.

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

[21] Qiang Wu and Ding-Xuan Zhou. Svm soft margin classifiers: linear programming versus quadratic programming. *Neural computation*, 17(5):1160–1187, 2005.