Assignment 1: Support Vector Machines

Moaz Mohamed
The University of Adelaide
Adelaide SA 5005
a1779177@student.adelaide.edu.au

1. Introduction

Over the last few years, Machine learning had a significant impact in the current technological landscape. The application of numerous Deep learning algorithms have been implemented in different sectors. In this paper Support vector machine (SVM) algorithm will be explored and its performance will be investigated on classification task on the provided dataset.In addition to a complete review of the SVM algorithm describing its strength and weakness.

2. Methodology and Background

2.1. Support Vector Machines

A famous approach to pattern recognition is using Support Vector Machines (SVM). Which in hindsight is very similar to the perceptron algorithm but SVM ensures finding a hyperplane that can separate between the two classes with maximum margin. the decision for either positive or negative classification follows the eq 1.

$$y_i\left(wx_i + b\right) \geq 1 \tag{1}$$

for the data points that lie exactly on the support vector boundary.

$$y_i\left(wx_i + b\right) - 1 = 0 \tag{2}$$

finding the width for the maximum margin will depend on finding the distance between the negative and positive points that exist on the boundary $(x_p - x_n)$ multiplying by $\frac{\bar{w}}{\|w\|}$ as unit vector provides the width.

$$\left(x_p - x_n\right) \times \frac{\bar{w}}{\|w\|} \tag{3}$$

using eq 2 the term that need to be maximized comes naturally as $\frac{2}{\|w\|}$ or for minimization as $\frac{1}{2} \times \|w\|^2$

$$\begin{gathered} \tfrac{1}{\|w\|} \times \left(x_p w - x_n w\right) \\ \tfrac{1}{\|w\|} \times \left([1-b] + [1+b]\right) \end{gathered} \tag{4}$$

$$\begin{gathered} \tfrac{2}{\|w\|} \\ \\ \tfrac{1}{2} \times \|w\|^2 \end{gathered} \tag{5}$$

thus the margin is

$$\frac{1}{2} \times \|w\|^2 \tag{6}$$

which can be constructed as a function to
minimize

$$j\left(w\right) = \frac{1}{2} \times \|w\|^2 \tag{7}$$

subject to

$$y_i \left(w^T x_i + b\right) \geq 1, \forall j \tag{8}$$

Such formulation does work on finding the optimal hyperplane and it can be solved with optimization tool such as CVXOPT, MOSER and SCS. a slack variable can also be added to relax the SVM with C as regularization parameter.

minimization of

$$\frac{1}{2} \times \|w\|^2 + c \times \frac{1}{n} \sum_{i=1}^{n} \xi_i \tag{9}$$

subject to

$$\begin{aligned} y_i \left(w^T x_i + b\right) \geq 1 - \xi i \\ \xi_i \geq 0, \forall n \end{aligned} \tag{10}$$

svm derive

$$L = \frac{1}{2} \|\bar{w}\|^2 - \sum \alpha_i \left[y_i \left(\bar{w} \cdot \bar{x}_i + b \right) - 1 \right] \tag{11}$$

$$\frac{\partial L}{\partial \bar{\omega}} = \bar{\omega} - \sum \alpha_i y_i x_i = 0 \Rightarrow \bar{\omega} = \sum_i \alpha_i y_i \bar{x}_i \tag{12}$$

$$\frac{\partial L}{\partial b} = -\sum \alpha_i y_i = 0 \Rightarrow \sum \alpha_i y_i = 0 \tag{13}$$

$$L = \frac{1}{2} \left(\sum \alpha_i y \bar{x}_1 \right) \left(\sum \alpha_j y_i \bar{x}_j \right) - \sum \alpha_i y_j x_i \left(\sum \alpha_j y_j x_j \right) - \sum \alpha_i y_i b + \sum \alpha_i (14)$$

Kuhn-Tucker theorem and Lagrange multipliers can be utilized to drive the previous problem from primal problem to dual problem.

$$L_D(\alpha) = \sum_i^n \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j < X_i^T X_j > \tag{15}$$

subject to for (non-separable case)

$$0 \leq \alpha_i \leq C/n, \forall i, \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{16}$$

or subject to for (separable case)

$$\alpha_i \geq 0, \forall i, \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{17}$$

and the $\overrightarrow{w}$ and the bias term can be calculated as follows.

$$w = \sum_{i=1}^{n} \alpha_i y_j x_i \tag{18}$$

$$b = \frac{1}{y_j} - w^T x_j \tag{19}$$

It is interesting to note that in equation 18 the weight vector can be formulated as summation of alpha, label and the support vectors. Hence the number of the support vectors is depended on α_i as some values of α_i will be zero. instead of in the primal problem where the $\overrightarrow{w}$ is obtained directly from solving the primal optimization problem which can be computationally expensive as the number of dimensions increases hence in the primal problem all the points have to be queried unlike in Dual problem $\overrightarrow{w}$ can be obtained from only non zero values of α_i. Also such difference allows for the usage for something called the "Kernel Trick".

$$L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_j, x_i) \tag{20}$$

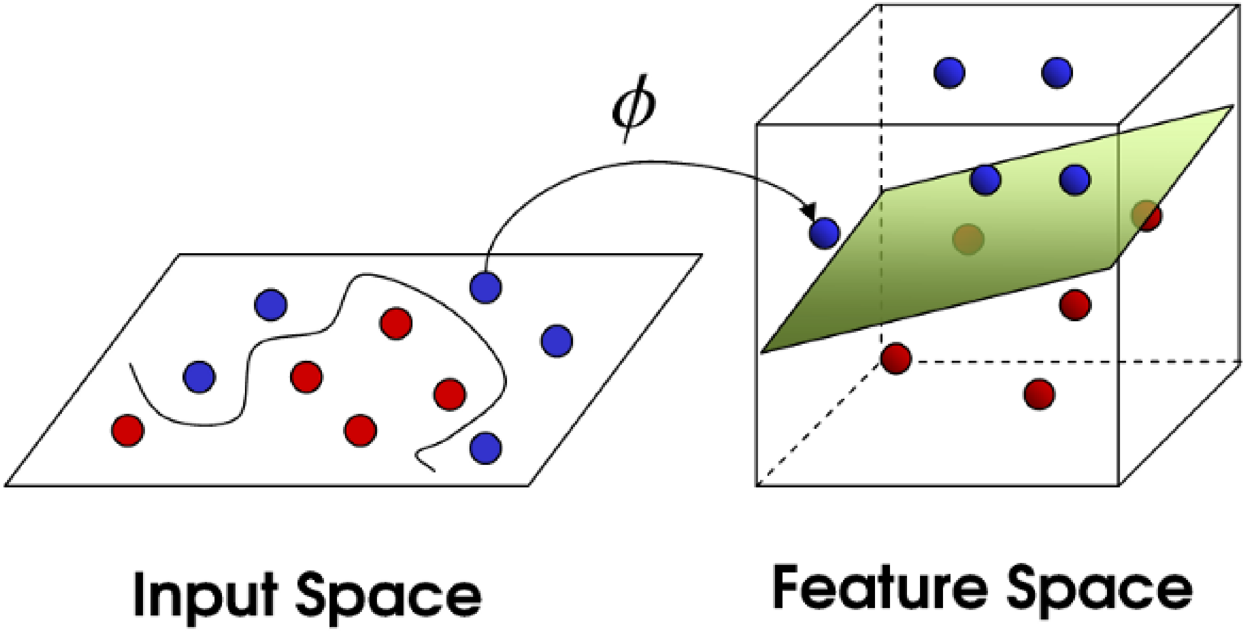

Figure 1: Mapping

$$K(x_j, x_i) = \left(x_i^T x_j + 1\right)^p \tag{21}$$

$$K(x_i, x_j) = \exp\left(-\frac{1}{2\sigma^2} \times \|x_j - x_i\|^2\right) \tag{22}$$

The optimization problems for both equations 20 and 15 except for the (x_j, x_i) in equation 20 is being applied to a kernel (Ex eq 21 and 22) and the data is being projected into higher dimensions. here the SVM is exploiting cover's theorem."A complex pattern-classification problem, cast in a high-dimensional space non-linearly, is more likely to be linearly separable than in a low-dimensional space, provided that the space is not densely populated." [4] Which increases the complexity of the SVM by applying the kernel and enabling it to classify classes even in non-separable cases as demonstrated in figure 1 and 4

3. Experimental Analysis.

3.1. SVM Primal and Dual

Sklearn toy datasets (classification and circles) have been used to for demonstration purposes. By utilizing the support vectors in eq 18 SVM is able to find the support vectors that can maximize the margin between the two classes as demonstrated in equation 15 but SVM without utilizing a kernel SVM has the same weakness as Perceptron [1] [3] [2] in that regard. But when a kernel is utilized project the data into higher dimensions SVM is capable of finding a hyperplane between the two classes as demonstrated in figure 4.

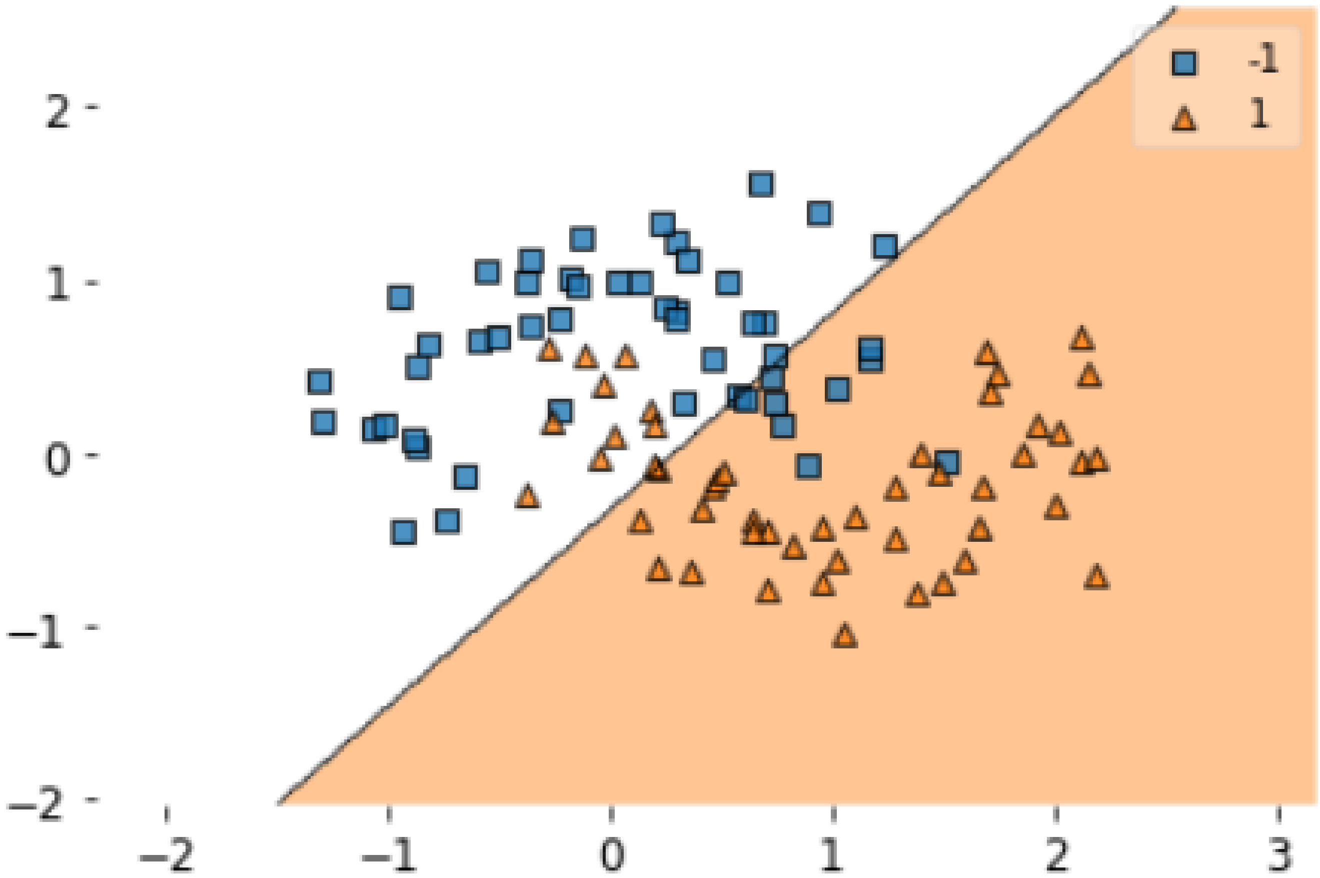

Figure 2: SVM decision boundary in non-linearly separable data

3.2. Performance on toy and provided data-set

In table 1 and 2 are the wights and biases produced by using CVXPY package to solve both primal and dual problems and the comparison with sklearn. As it can be seen in both tables the values are quite quite similar and acceptable error margin on the sklearn toy datasets. the performance on the provided dataset is shown in 3 with the same C parameters for all versions of SVM. with aggressive hyper-parameter tuning obtaining even a higher accuracy should be possible using grid-search with cross validation in mind but due to limited computation resources limited number of C values with cross-validation of 3 has been attempted and resulted in C value of (70,90) which corresponds to accuracy of (97.4%,97.2%) for both dual and primal respectively on the test-set. A more wide range of C value could be attempted with cross-validation of 5 or 10 but that have been abandoned to concentrate on the mathematical foundation of the algorithm itself.

Algorithm	W	bias	accuracy
Dual Soft	[0.75893, -0.651570]	0.290155	59%
Primal Soft	[0.75884, -0.651509]	0.290233	59%
sklearn	[0.75886, -0.651530]	0.290229	59%

Table 1: SVM implementation for Soft margin

4. Conclusion

Different variations of Support vector machines algorithm have been explored and analyzed. mathematical formulation of all the algorithms have been studied. decision boundaries for have been made for a careful study of the behavior of various

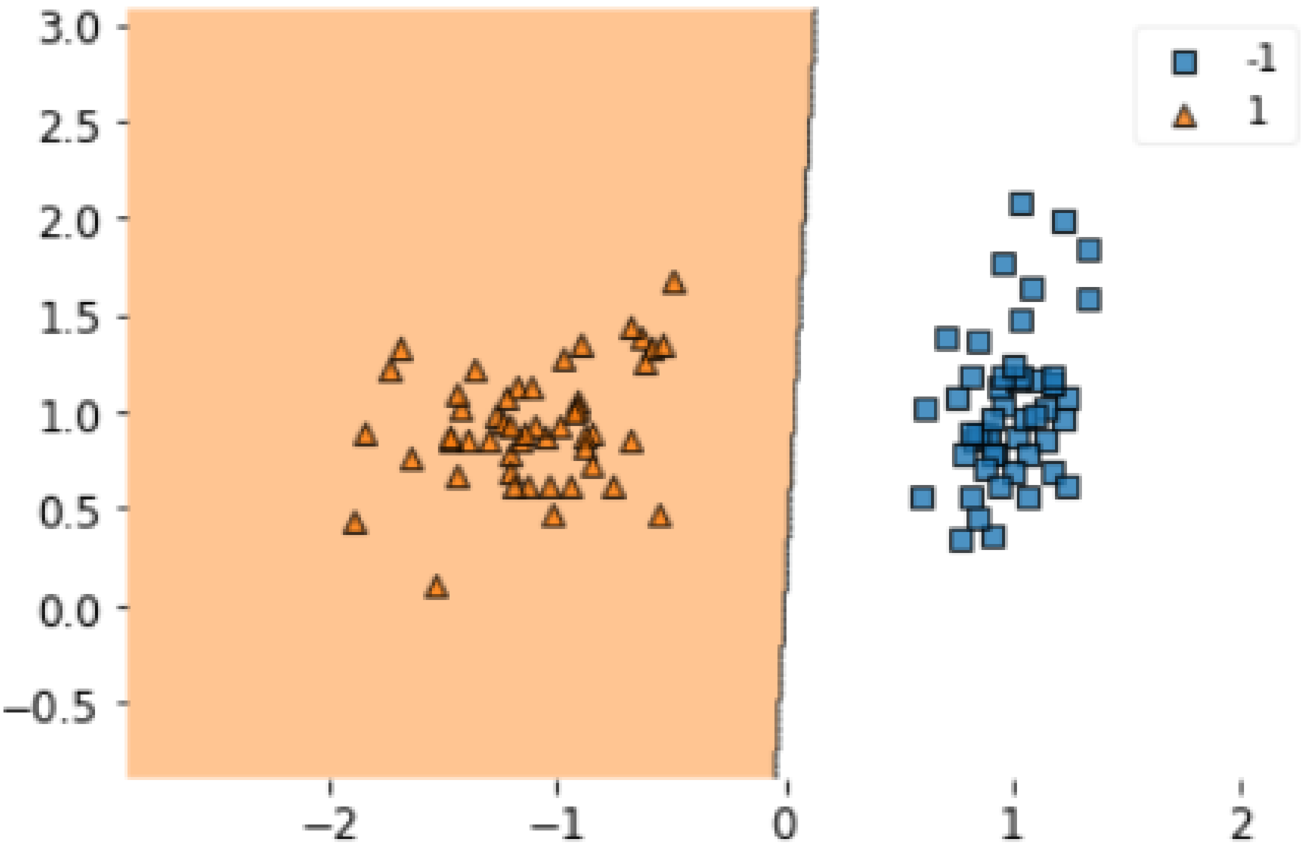

Figure 3: SVM decision boundary in linearly separable data

Algorithm	W	bias	accuracy
Dual Hard	[-1.75455, 0.077797]	0.00377	100%
Primal Hard	[-1.75451, 0.077783]	0.0038	100%
sklearn	[-1.74054, 0.077163]	0.0117	100%

Table 2: SVM implementation for Hard margin

Algorithm	training set	testing set	C
Dual Soft	97.74%	96.8%	1000
Primal Soft	97.71%	96.8%	1000
sklearn	97.75%	96.7%	1000

Table 3: SVM implementation for Soft margin on provided dataset

algorithms in different data situations. Advantageous and weakness have been identified for all the explored algorithms. SVM provides a very beautiful mathematical assay for its excellent performance especially for its ability to use different kernel to suit non-linear data. the math is elegant and provides excellent intuition for what is exactly the algorithm is doing to find the pattern in the data.

References

[1] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain,” *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.

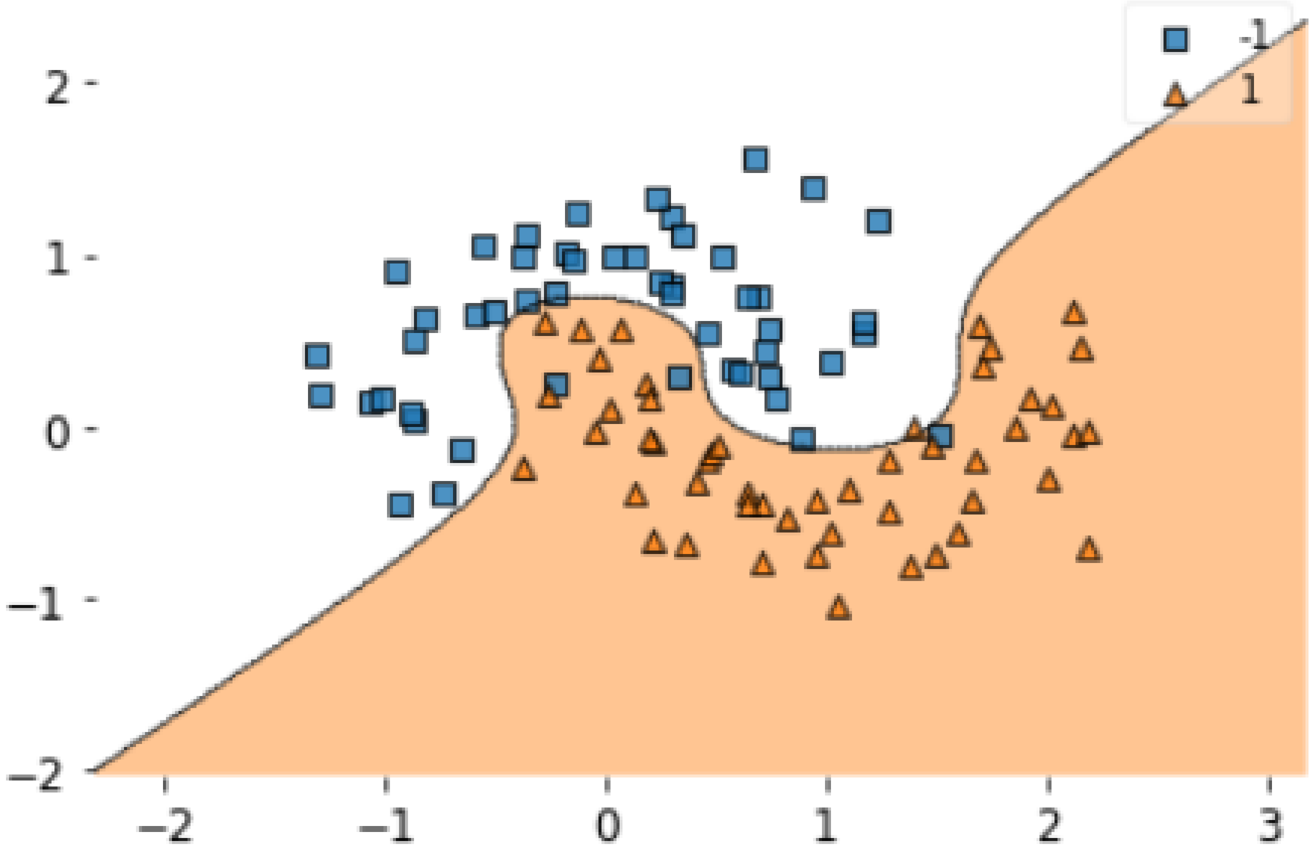

Figure 4: SVM with polynomial kernel applied. Decision boundary in non-linearly separable data

[2] C. Murphy, P. Gray, and G. Stewart, “Verified perceptron convergence theorem,” *MAPL 2017 - Proceedings of the 1st ACM SIGPLAN International Workshop on Machine Learning and Programming Languages, co-located with PLDI 2017*, pp. 43–50, 2017.

[3] M. Minsky and S. Papert, “Perceptrons: expanded edition,” *MIT Press Cambridge MA*, vol. 522, p. 20, 1969.

[4] M. Cover, “Inequalities Applications Pattern,” pp. 326–334, 1965.