CAAM 564 Project: Optimization Algorithms for Support Vector Machines

Wei Wu

May 1, 2019

1 Introduction

The support vector machines (SVMs) are popular supervised machine learning models, first introduced in (Boser, Guyon and Vapnik 1992 [2], Cortes and Vapnik 1995 [5]). SVMs have since gained attention in the machine learning community and have been applied widely to classification problems such as text/spam classification [6] [12]. In this project, however, we take an under-the-hood exploration of the SVMs: we are primarily interested in the mathematical formulation of the model, and the numerical solution to its underlying quadratic programming problem. This project is partially inspired by my experience with one of the homework of COMP 540 Statistical Machine Learning at Rice University.

2 Mathematical Formulation

2.1 Hard Margin SVMs

The classic hard margin SVMs are usually posed in the following form

$$\underset{w,b}{\text{minimize}} \quad \frac{1}{2}\|w\|_2^2 \quad \text{subject to} \quad y_i(\langle w, x_i\rangle - b) \geq 1,\ i = 1, \ldots, n. \tag{1}$$

where each data point (x_i, y_i) consists of features vector $x_i \in R^m$ and class labels y_i. In this project, we consider the two-class support vector machines, and hence $y_i \in \{-1, +1\}$.

Rewrite the inequality constraints as

$$g_i(w) = -y_i(\langle w, x_i\rangle - b) + 1 \leq 0. \tag{2}$$

The hard margin SVM forms a quadratic programming problem, with its Lagrangian

$$L(w, b, \alpha) = \frac{1}{2}\|w\|_2^2 + \sum_{i=1}^{n} \alpha_i(1 - y_i(\langle w, x_i\rangle - b)). \tag{3}$$

The primal and dual solutions satisfy the Karush-Kuhn-Tucker (KKT) conditions. In particular, we have

the following

$$0 = \nabla_w L(w, b, \alpha) = w - \sum_i^n \alpha_i y_i x_i \tag{4a}$$

$$0 = \frac{\partial}{\partial b} L(w, b, \alpha) = \sum_i^n \alpha_i \tag{4b}$$

With the above identities, we yield the following expression of the dual function:

$$\begin{aligned} q(\alpha) &= \min_{w,b} L(w, b, \alpha) \\ &= \frac{1}{2}\left\| \sum_i \alpha_i y_i x_i \right\|_2^2 + \sum_i \alpha_i (1 - y_i \left\langle \sum_j \alpha_j y_j x_j, x_i \right\rangle + y_i b) \\ &= \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle - \sum_{i,j} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle + \sum_i \alpha_i (y_i b + 1) \\ &= \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \end{aligned} \tag{5}$$

This leads to our dual optimization problem $\max_{\alpha \geq 0} \min_{w,b} L(w, b, \alpha)$ with the following form

$$\max_\alpha \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle \text{ subject to } \alpha \geq 0, \text{ and } \sum_i \alpha_i y_i = 0. \tag{6}$$

Note that from the KKT conditions, we also get $\alpha_i > 0$ only for data points where the corresponding inequality constraints holds with equality, i.e. $g_i(w) = 0$. These x_i's are called the support vectors. The solution (w^*, b^*) to (7) gives an halfspace defined by $\langle w, x \rangle + b = 0$, which serves as the separating hyperplane (a decision boundary) to classify data points. If we add one more feature to each features vector, we can elimiate the bias term $b = 0$. In this case, the support vectors are at distance $1/||w^*||$ to the separating hyperplane, and in fact span w^*, a result by applying the Fritz-John (or Karush-John) optimality conditions [10], or simply from (4a) :

Let $I = \{i : |\langle w^*, x_i \rangle| = 1\}$. *Then there exists coefficients* $\alpha_1, ..., \alpha_n$ *such that* $w^* = \sum_{i \in I} \alpha_i x_i$

2.2 Soft Margin SVM

The hard-margin SVM does not have a solution if the data is not linearly separable. Further, even if the data is linearly separable, the solution is highly sensitive to outliers. To address this, we add a slack variable ξ and relax the hard inequality constraint, but penalize large violations:

$$\underset{w,b}{\text{minimize}} \quad \frac{1}{2}\|w\|_2^2 + C \sum_{i=1}^n \xi_i \quad \text{subject to} \quad y_i(\langle w, x_i \rangle - b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \ i = 1, \ldots, n. \tag{7}$$

Note that $C \geq 0$ is a hyperparameter to the model that regularizes the penalty of large slack variable.

The soft margin SVM in its primal form is equivalent to the following unconstrained optimization problem

$$\underset{w,b}{\text{minimize}} \quad \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{n} max(1 - y_i(\langle w, x_i\rangle - b), 0) \tag{8}$$

The dual form of the soft margin SVM could be found in a similar way where we obtain the dual form of the hard margin SVM. I omit the derivation and simply present the result below

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \langle x_i, x_j\rangle \text{ subject to } 0 \leq \alpha \leq C, \text{ and } \sum_i \alpha_i y_i = 0. \tag{9}$$

2.3 Usage of Kernel

The equations (6) and (9) alludes to the usage of kernel trick. If we apply a feature mapping ψ to each of the feature vector x, then we obtain a kernel function $K(x, x') = \langle \psi(x), \psi(x')\rangle$. By applying the feature mapping, and therefore the kernel function, we embed the input space into some high dimensional feature space, and hence enable our SVM to learn nonlinear decision boundary in the input space. Whether a kernel function is valid is governed by the Mercer's theorem, which states a kernel function is valid if its associated kernel matrix is positive-semidefinite.

Some common kernel functions include the class of linear kernels $K(x, y) = x^T y + b$ and the class of Gaussian kernels $K(x, y) = exp(-\frac{||x-y||^2}{2\sigma})$. From now on we assume a kernelized version of soft margin SVMs.

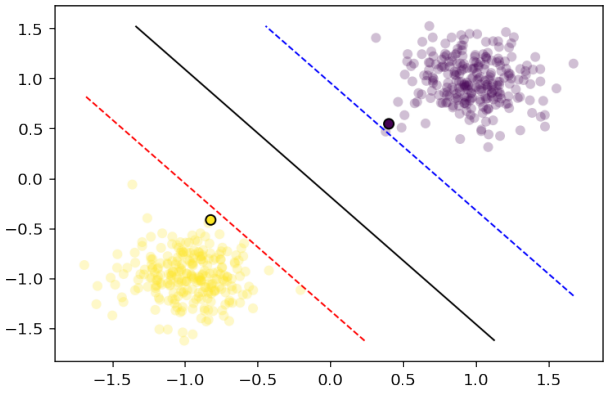
3 Optimization Algorithms

This primal formulation (8) gives rise to a simple stochastic (sub)gradient descent (SGD) algorithm for the soft margin SVM (e.g. [11]), since it fits exactly within the framework of solving regularized loss minimization problem with SGD. Other algorithms focused on solving the primal form were also proposed (e.g. [4]).

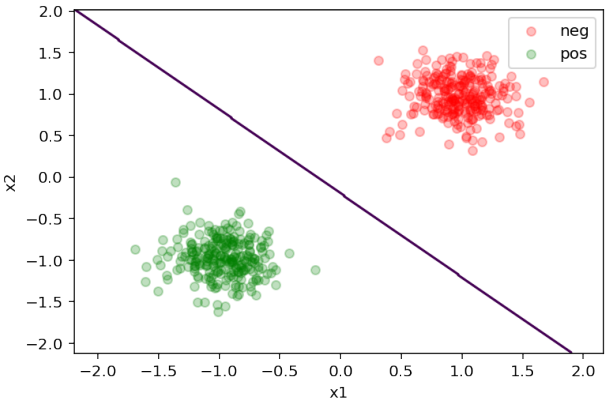
The numerical solution to the dual form (9) is more extensively, especially studied during the early 2000s. Some active sets algorithms are proposed in [9] and [13]. Both algorithms are much more complexed than the SGD-based algorithm. Another popular and simpler numerical algorithm to solve the dual form is the so-called Sequential Minimal Optimization, first proposed in [8], and its implementation *libsvm* [3] remains one of the SVM implementations (SVC) of the scikit-learn library [7]. One rudimentary implementation of SMO could be found here [1]

For this project, I implemented an SGD-based SVM, and also an SMO-based SVM by modifying a solver found online. From the practical perspective, one difference between the two algorithms is that the SGD-based SVM does not compute support vectors on the fly, while the SMO-based SVM gives the support vectors naturally by solving the dual problem. The support vectors could potentially be used by a machine learning practitioner to interpret the model.

The code for both implementations could be found in the Github repository for this project. The file *SMO.py* contains the implementations of the SMO-based model. The files *linear_svm.py* and *linear_classifier.py* contain the implementation of the SGD-based model. Both implementations uses helper functions from *utils.py*. The IPython notebook *SVM.ipynb* contains demo code of both models. Some visualization results on small synthetic datasets could be found on the next page. Note that the decision boundaries are subject to hyperparameters tunning.

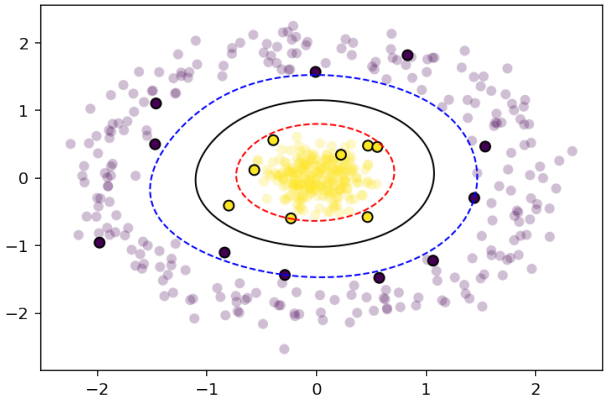


(a) SMO result with decision boundary + support vectors (highlighted)

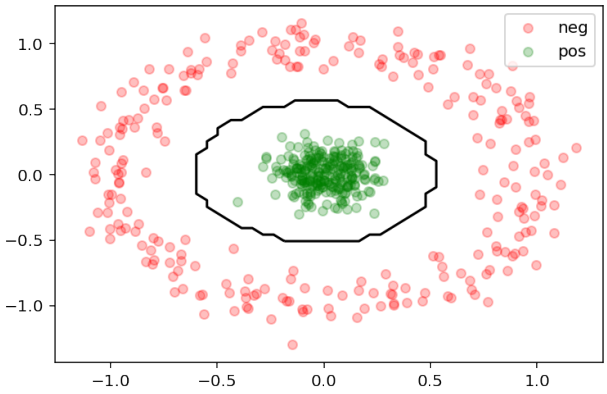


(b) SGD result with decision boundary

Figure 1: Linearly Separable Data



(a) SMO result with decision boundary + support vectors (highlighted)



(b) SGD result with decision boundary

Figure 2: Non-Linearly Separable Data

4 Conclusion/Reflection

In this project, we examined the mathematical formulation of Support Vector Machines, and implemented two of the commonly used algorithms to solve the underlying optimization problem.

Originally I wished to implement the active sets method presented in [9] to further my understanding of the active sets method learned in CAAM 564, but it turned out to be too ambitious. A more thorough examination of and an own implementation of the SMO algorithm is left as my future work.

References

[1] Implementing a support vector machine using sequential minimal optimization and python 3.5. `https://jonchar.net/notebooks/SVM/`.

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

[3] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[4] Olivier Chapelle. Training a support vector machine in the primal. *Neural computation*, 19(5):1155–1178, 2007.

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

[6] Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer, 1998.

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

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

[9] Katya Scheinberg. An efficient implementation of an active set method for svms. *Journal of Machine Learning Research*, 7(Oct):2237–2257, 2006.

[10] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, New York, NY, USA, 2014.

[11] Shai Shalev-Shwartz and Yoram Singer. Online learning: Theory, algorithms, and applications. 2007.

[12] Simon Tong and Daphne Koller. Support vector machine active learning with applications to text classification. *Journal of machine learning research*, 2(Nov):45–66, 2001.

[13] Michael Vogt and Vojislav Kecman. Active-set methods for support vector machines. In *Support vector machines: Theory and applications*, pages 133–158. Springer, 2005.