Least Squares Support Vector Machine Classifiers

A-young Kang
School of Electrical Engineering and Computer Science
Oregon State University
Corvaliis, OR 97331
kanga2@oregonstate.edu

Abstract

In this paper, we discuss Support Vector Machines and a least squares version of Support Vector Machines for classification. The optimization problem of Support Vector Machines can be solved using a quadratic programming solver, while for Least Squares Support Vector Machines, the solution can be obtained by solving a set of linear equations.

1 Introduction

Support Vector Machines (SVM) are a sophisticated machine learning algorithm based on a very beautiful and solid theoretical background. The algorithm is easy to apply in many ways and has powerful performance so, it is practical and highly preferred. The problem to be solved in SVM is "how do we divide the space with decision boundaries?" SVM maximizes the margin around the separating hyperplanes, and the decision function is specified by the support vectors, which becomes a quadratic programming (QP) problem. Least Squares Support Vector Machines (LS-SVM) are a least-squares version of SVM. This involves solving a set of linear equations inestead of a QP problem. In classical SVM, many of the support values (Lagrange multipliers α_i's) are zero and only support vectors have non-zero values, while in LS-SVM, the support values are proportional to the errors.

In this paper, we review SVM for classification (Section 2), and formulate LS-SVM classifiers. (Section 3). In section 4, we implement SVM using `cvxopt` and LS-SVM using `np.linalg.solve` and use Optical Character Recognition (OCR) data to compare the optimal solutions found using both classifiers.

2 Support vector machines for classification

Given a training set $\{x_k, y_k\}_{k=1}^N$, where $x_k \in \mathbb{R}^n$ are input patterns and $y_k \in \mathbb{R}$ are class labels. The classifier can be expressed as

$$y_x = sign\left[\sum_{k=1}^{N} \alpha_k y_k \Psi(x, x_k) + b\right] \tag{1}$$

where α_k are positive real constants, b is a real constant, and Ψ is a kernel function.

If the training data is linearly separable, the hyperplans can be described as

$$y_k\left[w^T\varphi(x_k) + b\right] \geq 1, \quad k = 1, \cdots, N \tag{2}$$

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

where $\varphi(\cdot)$ is a nonlinear function that maps the input space into a higher dimensional space. For the non-separable training data, variables ξ_k are introduced and the optimization problem is

$$\begin{aligned} &\min \mathcal{J}(w, \xi_k) = \frac{1}{2} w^T w + c \sum_{k=1}^{N} \xi_k \\ &\text{subject to} \begin{cases} y_k \left[w^T \varphi(x_k) + b \right] \geq 1 - \xi_k, k = 1, \cdots, N \\ \xi_k \geq 0, k = 1, \cdots, N \end{cases} \end{aligned} \tag{3}$$

By constructing the Lagrangian function, we get the following quadratic programming (QP) problem:

$$\begin{aligned} &\max_{\alpha_k} \mathcal{Q} - \frac{1}{2} \sum \sum_{k,l=1}^{N} y_k y_l K(x_k, x_l) \alpha_k \alpha_l + \sum_{k=1}^{N} \alpha_k \\ &\text{such that} \begin{cases} \sum_{k=1}^{N} \alpha_k y_k = 0 \\ 0 \leq \alpha_k \leq c, k = 1, \cdots, N \end{cases} \end{aligned} \tag{4}$$

where $K(x_k, x_l) = \psi(x_k)^T \psi(x_l)$.

3 Least squares support vector machines for classification

The least-squares version of the SVM classifier is obtained by reformulating the minimization problem as

$$\min_{w,b,e} \mathcal{J}(w, b, e) = \frac{1}{2} w^T w + \gamma \frac{1}{2} \sum_{k=1}^{N} e_k^2 \tag{5}$$

subject to

$$y_k \left[w^T \varphi(x_k) + b \right] = 1 - e_k, \quad k = 1, \cdots, N \tag{6}$$

The Lagrangian

$$\mathcal{L}(w, b, e; \alpha) = \mathcal{J}(w, b, e) - \sum_{k=1}^{N} \alpha_k \{ y_k \left[w^T \varphi(x_k) + b \right] - 1 + e_k \} \tag{7}$$

with Lagrange multipliers α_k.

The conditions for optimality are

$$\begin{cases} \frac{\partial \mathcal{L}}{\partial w} = 0 \quad \rightarrow \quad w = \sum_{k=1}^{N} \alpha_k y_k \varphi(x_k) \\ \frac{\partial \mathcal{L}}{\partial b} = 0 \quad \rightarrow \quad \sum_{k=1}^{N} \alpha_k y_k = 0 \\ \frac{\partial \mathcal{L}}{\partial e_k} = 0 \quad \rightarrow \quad \alpha_k = \gamma e_k \\ \frac{\partial \mathcal{L}}{\partial \alpha_k} = 0 \quad \rightarrow \quad y_k \left[w^T \varphi(x_k) + b \right] - 1 + e_k = 0 \end{cases} \tag{8}$$

These conditions can be written as

$$\left[\begin{array}{ccc|c} I & 0 & 0 & -Z^T \\ 0 & 0 & 0 & -Y^T \\ 0 & 0 & \gamma I & -I \\ \hline Z & Y & I & 0 \end{array} \right] \left[\begin{array}{c} w \\ b \\ e \\ \hline \alpha \end{array} \right] = \left[\begin{array}{c} 0 \\ 0 \\ 0 \\ \hline \vec{1} \end{array} \right] \tag{9}$$

where $Z = [\varphi(x_1)^T y_1; ...; \varphi(x_N)^T y_N], Y = [y_1; ...; y_N], \vec{1} = [1; ...; 1], e = [e_1; ...; e_N], \alpha = [\alpha_1; ...; \alpha_N]$. After eliminating w, e, we finally get the following dual problem

$$\left[\begin{array}{c|c} 0 & -Y^T \\ \hline Y & \Omega + \gamma^{-1} I \end{array} \right] \left[\begin{array}{c} b \\ \hline \alpha \end{array} \right] = \left[\begin{array}{c} 0 \\ \hline \vec{1} \end{array} \right] \tag{10}$$

where $\Omega = ZZ^T$ and Mercer's condition is applied

$$\begin{aligned} \Omega_{kl} &= y_k y_l \varphi(x_k)^T \varphi(x_l) \\ &= y_k y_l K(x_k, x_l). \end{aligned} \tag{11}$$

By solving the set of linear equations in equations (10)-(11), we can obtain the optimal solution. In equation (8), we can see that suppot values α_k are proportional to the errors.

4 SVM and LS-SVM for optical character recognition (OCR)

4.1 Data description

In Optical Chracter Recognition (OCR), for a given handwritten digit, we want to predict a number from zero to nine. we simplify the OCR into a binary classification task. Particularly, we consider predictions for 3 and 5. All the handwritten digits are obtained from `https://www.kaggle.com/c/digit-recognizer/data` and samples with only labels 3 and 5 are extracted. The training data set has 4888 samples and the test data set has 1629 samples. Our goal is to classify the handwritten digits of numbers 3 and 5.

4.2 SVM for OCR

We implemented hard-margin SVM using `cvxopt` package. Therefore, we need to match the solver's API which is of the form [2]:

$$\begin{aligned} &\min \frac{1}{2} x^T P x + q^T x \\ \text{subject to } &Gx \leq h \\ &Ax = b \end{aligned} \tag{12}$$

Let $\mathbf{H}$ be a matrix such that $H_{k,l} = y_k y_l x_k^T x_l$, then the dual problem for classical SVM becomes:

$$\begin{aligned} &\min \frac{1}{2} \alpha^T \mathbf{H} \alpha - 1^T \alpha \\ \text{subject to } &-\alpha_i \leq 0 \\ &y^T \alpha = 0 \end{aligned} \tag{13}$$

The above optimization problem gives us

- $P := H$
- $q := -\vec{1}$
- $G := -diag[1]$
- $h := \vec{0}$
- $A := y$
- $b := 0$

We utilize the below polynomial kernel with degree d:

$$K(x, y) = (1 + x^T y)^d \tag{14}$$

and plot the test accuracy versus d. As can been seen in Figure 1, the test accuracy decreases as d increases. Using a higher degree polynomial kernel (larger d value) is equivalent to working in a larger feature space, so it can lead to overfitting. In other words, the algorithm tries to fit the training data more and therefore results in lower accuracy on the test set. Therefore, the best value for d is 1 for this data set.

4.3 LS-SVM for OCR

We apply the kernel LS-SVM with different d values in [1, 2, 3, 4, 5] and observes the test accuracy versus d. Figure 2 shows how the test accuracy varies as the polynomial degree increases. In this case, when d = 3, the algorithm has the best test accuracy of 0.9877. We see the classifier starts overfitting when the degree of the polynomial kernel is greater than 3. As you can see, the results of both algorithms are not the same even though we apply them on the same dataset. This is because the formulation of both objective functions of SVM and LS-SVM are different, and therefore the results are not identical.

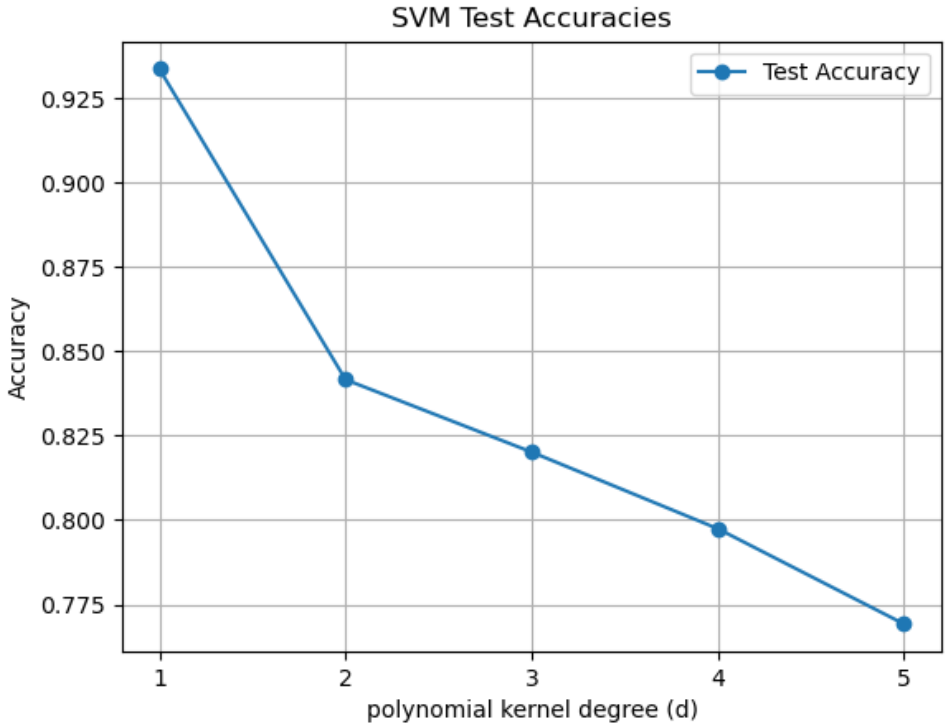

Figure 1: SVM Test Accuracies

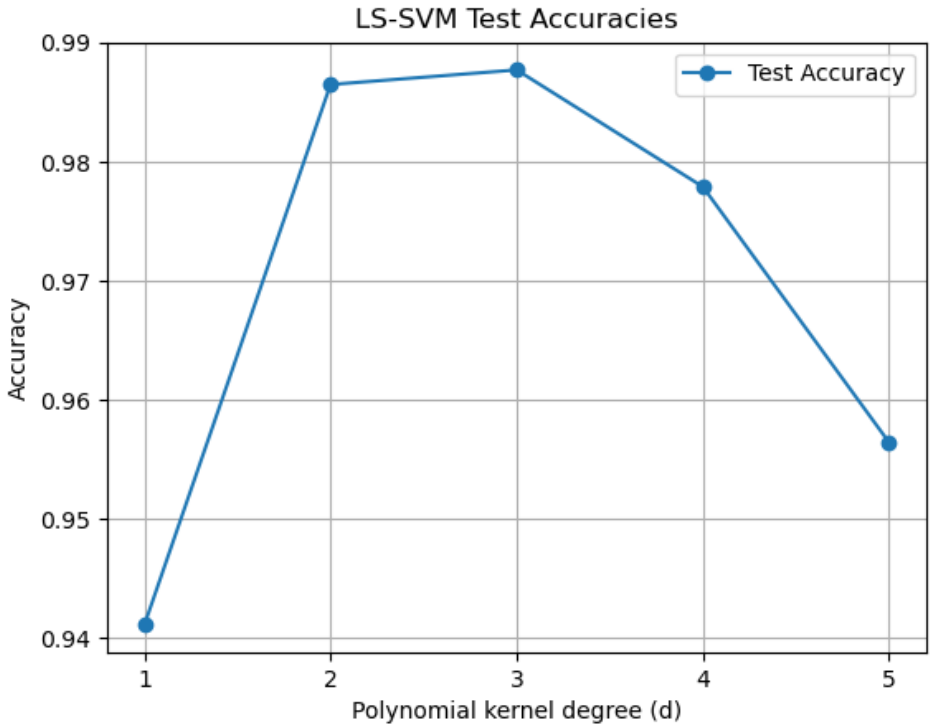

Figure 2: LS-SVM Test Accuracies

5 Conclusion

We discussed Support Vector Machine Classifiers and Least Squares Support Vector Machine Classifiers. The major difference is that LS-SVM requires us to solve a set of linear equations while SVM requires us to solve a quadratic programming problem. Moreover, LS-SVM's support values are proportional to the errors while in SVM, only support vectors have non-zero values. Since the formulation of both classifiers' objective functions are different, their results are not identical even when we apply them on the same dataset.

References

[1] Suykens, J.A.K. & Vandewalle, J. (1999) Least Squares Support Vector Machine Classifiers. *Neural Processing Letters*, pp. 609–616. **9**(3):293-300.

[2] Tristan, F. (2009) Support Vector Machines Explained. *Journal of Neuroscience*.