Supported Vector Machine with SAS

Qi Zhao

Lehigh University

Abstract. Supported Vectored Machine (SVM) is one of the most historical, but also most commonly used machine learning models in supervised learning. In this project, I built a SVM model with the Sequential Minimal Optimization (SMO) algorithm using SAS IML procedure. Also, I simulated some linearly separable data using data step and compared the result of the SVM model with the SAS build-in Logistic Procedure. Finally, I applied the model to a famous dataset called credit.

Keywords: SVM, SMO, Machine Learning, SAS

1 Summary

In this project, I generated a training set and a test set (following the same distribution) using SAS DATA step. Then I used SAS IML procedure to implement the Sequential Minimal Optimization Pseudo code, which allows me to approximate the parameters that we need in the Supported Vector Machine Model. Then utilizing the build-in Logistic procedure, I built a Logistic Model and used the confusion matrix and SGPLOT procedure to visualize the result. Next, I used the two models in a classic binary classification dataset, the credit dataset, to compare the performance of this algorithms and made some conclusions.

2 Supported Vector Machine

The original SVM model was invented by Vladimir Vapnik and Alexey Chervonenkis in 1963. In 1992, a more powerful model was proposed by Bernhard Boser, Isabelle Guyon and Vladimir Vapnik, which can create nonlinear classifiers by applying the kernel trick to maximum-margin hyperplanes. In this report, we will focus on the linear SVM in binary classification.

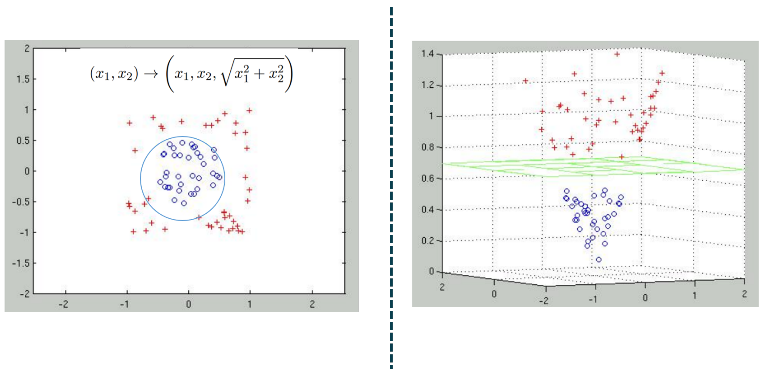


Fig. 1. mapping data into higher dimensional space

2.1 Linear SVM

First of all, consider the training set of n points $\{\mathbf{x}_i, y_i\}$ $i = 1, \ldots, n$ where $y_i \in \{-1, 1\}$ is the class of the point x_i. Here, we see the two classes as $y_i = 1$ and $y_i = -1$. We want to find a classifier (a hyperplane) which can map x_is into higher dimensional space so that the two different classes of points can be divided. Also we want the hyperplane to have the maximum-margin, which can maximize the distance of the hyperplane and nearest points from both groups. In linear cases, the hyperplane can be written as:

$$\mathbf{x}_i\mathbf{w} + b = 0$$

We want to find two parallel hyperplanes that can also separate the data and we want their distance to be as large as possible. Heres a way to describe them:

$$\mathbf{x}_i\mathbf{w} + b = +1$$

and

$$\mathbf{x}_i\mathbf{w} + b = -1$$

Its not difficult to prove that the distance between the two hyperplanes is $\frac{2}{||w||}$:

$$d_+ + d_- = \frac{|1-b|}{\|w\|} + \frac{|-1-b|}{\|w\|} = \frac{2}{\|w\|} \tag{1}$$

which means we want to minimize $\|w\|$ since it is always positive. Besides, we want for every $i \in (1, n)$, x_i and y_i follows the constraints:

$$\mathbf{x}_i\mathbf{w} + b \geq +1,\ y_i = +1 \tag{2}$$
$$\mathbf{x}_i\mathbf{w} + b \leq -1,\ y_i = -1 \tag{3}$$
$$\equiv \tag{4}$$
$$y_i(\mathbf{x}_i\mathbf{w} + b) - 1 \geq 0, \qquad \forall i \tag{5}$$

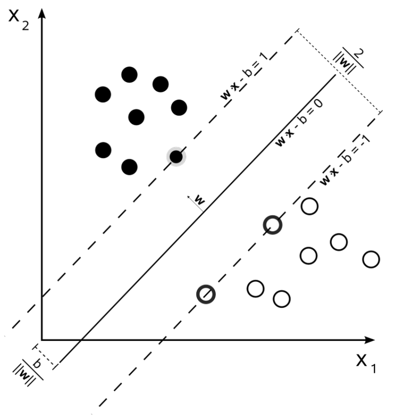


Fig. 2. 2 hyperplanes

2.2 Linear SVM

However, in most cases it is impossible for all the points in the training data to follow the rules. To solve this problem, people add this constraint as a regularization part in the object function which we want to minimize.

2.3 Lagrange Multiplier

As mentioned before, we want to maximize the margin, which is the same as minimize $\|w\|$. Also we want them data points to be properly classified.
Thus we want to solve the following optimization problem:

$$\text{Minimize } \tfrac{\|w\|^2}{2}$$
$$\text{s.t } y_i(\mathbf{x}_i\mathbf{w} + b) - 1 \geq 0$$

What we want is to find the w and b that can make this statement true. However, it is almost impossible to solve this equation directly. In order to solve this optimization problem, we need to introduce the Lagrange Multiplier.
In mathematical optimization, the method of Lagrange multipliers is a strategy for finding the local maxima and minima of a function subject to equality constraints.
For the case of only one constraint and only two choice variables, consider the optimization problem:

$$\text{Minimize } f(x,y)$$
$$\text{s.t } g(x,y) = 0$$

This looks exactly like the problem that we want to solve and we can switch the problem into a lagrangian problem. For convenience, I will skip the mathematical proof part and therefore we can change the problem into:

$$\text{Minimize } W_p(\alpha) = \tfrac{\|w\|^2}{2} - \sum_{i=1}^{l} \alpha_i y_i(\mathbf{x}_i\mathbf{w} + b) + \sum_{i=1}^{l} \alpha_i$$

We can see this as a convex quadratic programming problem with the dual:

$$\text{Maximize } W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{l} \alpha_i\alpha_j y_i y_j(\mathbf{x}_i\mathbf{x}_j)$$

The reason that we make this transformation is that the solution of dual problem can be easily approximate with computer. Although there will always be a small gap between the approximated result and the real answer, it is still very powerful and useful.

2.4 Finding Classifier

Usually, most α_i's are 0 and the points with $\alpha_i > 0$ are called "supported vectors".
After we find the α_i's, we need to use them to compute the vector w and the scalar b. But how? In the dual problem, we want to:

$$\text{maximize } W(\alpha)$$

where,

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

and the $\alpha \geq 0$ satisfying:

$$\sum_{j=1}^{n} \alpha_j y_j = 0$$

Using KKT-conditions for optimality,

$$\nabla_w L = 0, \text{i.e., } w = \sum_{j=1}^{n} \alpha_j y_j x_j$$

$$\nabla_b L = 0, \text{i.e., } \sum_{j=1}^{n} \alpha_j y_j = 0$$

Once an optimal α is obtained we find w as the linear classifier of the x_i's and we can also find b ,

$$b = \frac{-\sum_{i,j} \alpha_i \alpha_j y_i y_j < x_i, x_j >}{\sum_j \alpha_j}$$

3 Sequential Minimal Optimization

In order to approximate the dual problem in SAS there are many modern Algorithms like Sub-gradient descent and coordinate descent. In this project, I used a method called Sequential Minimal Optimization (SMO).
SMO method is very commonly used because there are many optimization inside this algorithm which makes it very fast to converge, which means it is very useful when the dataset is large.
In short, the SMO algorithm selects two parameters, α_i and α_j and optimizes the objective value jointly for both these αs. Finally it adjusts the b parameter based on the new αs. This process is repeated until the αs converge.

3.1 Simplified SMO

In this report, we use a simplified version of the Sequential Minimal Optimization algorithm for training support vector machines.The original SMO algorithm contains many optimizations designed to speed up the algorithm on large datasets and ensure that the algorithm converges even under degenerate conditions. The simplified version, however, is not even guaranteed to converge for all data sets, so if you want to use SVMs on a real-world application, you should either implement the full SMO algorithm, or find a software package for SVMs.The SVM model can be easily found in packages like Sk-learn in Python.

3.2 Main Steps

What we do is that we iterate over all $i, i = 1, ..., n$. If i does not fulfill the KKT conditions to within some numerical tolerance, we randomly select j from the remaining n-1 α's and try to optimize i and j together.

Firstly we want to find bounds L and H such that $L \leq j \leq H$ must hold in order for j to satisfy the constraint that 0 j C. It can be shown that these are given by the following:

if $y_i \neq y_j$, $L = max(0, \alpha_j - \alpha_i)$, $H = min(C, C + \alpha_j - \alpha_i)$

if $y_i = y_j$, $L = max(0, \alpha_j + \alpha_i - C)$, $H = min(C, \alpha_j + \alpha_i)$

Then we update and fix the α's into the upper and lower bounds. The SAS code will be provided in the appendix.

4 SAS Implementation

In this section, I use SAS to implement the SMO algorithm so I can build a SVM model using the training data. In SAS Enterprise Miner, there is a build-in SVM function but for convenience in this project I used logistic regression (the logistic procedure in SAS) to evaluate the result that I got using the SVM model.

4.1 Data Simulation

I simulated 1200 data points using 2 different probability density functions so that we can test this model. In order to visualize the data, I only used 2 variables X_1 and X_2 so that the points can be plotted in a scatter plot. Using the IML

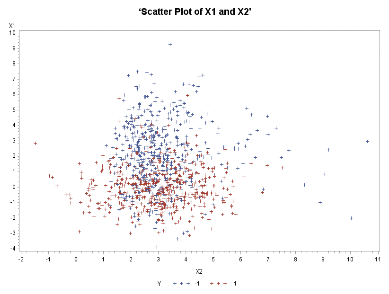


Fig. 3. 2 hyperplanes

procedure in SAS, we can implement the SMO algorithm to the simulated data. Following is the confusion matrix of the test set using the SVM model.
After that, I used the build-in Logistic Regression model to Compare the two different method. Fig.4. is the result of 2 models. As we can see, the two different model almost have the same performance, which means the SVM model using the SMO algorithm does work well with the simulated data.
Therefore, we are going to apply the model to a real-world data set.

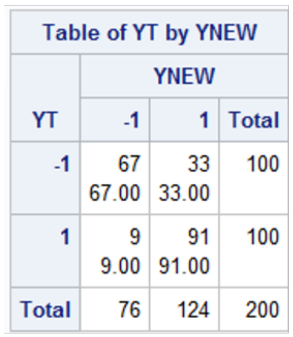


Table of YT by YNEW

	YNEW		
YT	-1	1	Total
-1	67 67.00	33 33.00	100
1	9 9.00	91 91.00	100
Total	76	124	200

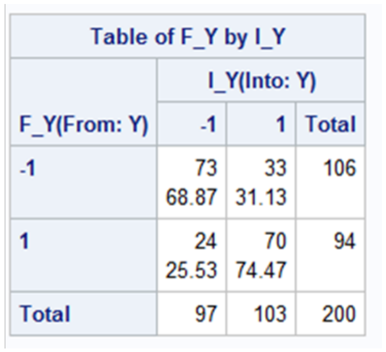


Table of F_Y by I_Y

	I_Y(Into: Y)		
F_Y(From: Y)	-1	1	Total
-1	73 68.87	33 31.13	106
1	24 25.53	70 74.47	94
Total	97	103	200

Fig. 4. the confusion matrix of SVM model and Logistic Regression model

4.2 Credit Approval Dataset

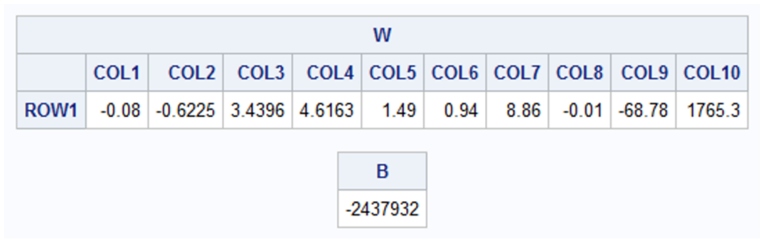
The credit approval dataset is a very famous dataset in UCI machine learning repository. concerns credit card applications. All attribute names and values have been changed to meaningless symbols to protect confidentiality of the data.

Processing Data First of all, I read all the data into SAS, and realized that there aren't many observations with missing values. Thus, I simply deleted all the observations with missing values.
Next, since there are only about 650 data left, I simply deleted all the character variables rather than using dummy variable. For the 'T' (True) and 'F' (False) variables, I changed them into '1' and '0'.
Finally, I separated the data into training set (about 70%) and testing set (about 30%).

Building the model The IML read the data into a matrix and update every time. I iterated 2000 times and got the result in SAS.
One thing I need to mention is that the way I computed the 'b' parameter is as follows:

$$b = \frac{-\min(wx_i|y_i = 1) - \max(wx_i|y_i = -1)}{2}$$

Results After trained the model using the training set, I used the data in testing set to see the result. However the result isn't that satisfying comparing to the Logistic Regression model.
The error rate is only 16.8% in the Logistic Regression model, but it is as high as 35.5% in the simple linear SVM model.



W										
	COL1	COL2	COL3	COL4	COL5	COL6	COL7	COL8	COL9	COL10
ROW1	-0.08	-0.6225	3.4396	4.6163	1.49	0.94	8.86	-0.01	-68.78	1765.3

B
-2437932

Fig. 5. the parameters

Conclusion We are curious about why the SVM model doesn't perform well in this new data set.
First of all, the SMO algorithm we use is a simplified version, and we simply iterate 2000 times which may affect the estimate of the parameters.
Secondly, the SVM model is only linear model, which may not be appropriate when the data is more complicated. Therefore it may perform better if we use some kernel tricks.
Finally, this model is sensitive to outliers since we didn't processed the outliers.

Notes and Comments. The SVM is still powerful in today's world and I have learned a lot through this project. If I have more time, I would try to dig deeper into the dual problem since that is the core part in SVM. and I am still learning how to use gradient descent to solve this dual problem.

References

1. SAS: IML
2. Platt, John.: Fast Training of Support Vector Machines using Sequential Minimal Optimization , in Advances in Kernel Methods Support Vector Learning, B. Scholkopf, C. Burges, A. Smola, eds., MIT Press (1998).
3. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines. Cambridge University Press, Cambridge (2000)