Binary Class Linear SVMs

1 Understanding of SVM

1.1 Concept Of Binary Class Linear SVM:

Support Vector Machine is mainly used to solve classification problem. SVM divides the binary classified data as in positive and negative samples by hyperplane, which is optimized from the closest vectors that are called **Support Vectors**. The hyperplane separates the positive and negative data with the ***largest margin*** where margin is defined as the closest distance from positive/negative sample to the hyperplane.

1.2 Why max margin is good?

Line (max margin) separates minuses most from positives and vice versa. If it is near to negative points than it hits very soon to the negative side which means that hyperplane is not good as nearest negative vectors have more possibility of wrongly classified. Same way with positive side, we want to stay far from positive side as well negative sides which helps us to distinguish one class from another class in a better way. Not only that, it minimizes the possibility of getting errors/wrong classification between those different classes. In another words, distance from separating hyperplane corresponds to the "Confidence" of prediction. Therefore, maximum margin is good.

1.3 Understanding The Margin with formula:

Suppose, any two sets of samples are given with more than 1 feature where, we takes features as the dimensions for plotting the data, we can generate m * n matrix X in Euclidean 2 dimension space where n is columns of data as features and m is the different separate data collected in the rows. Now, for binary classification we try to separate those rows in two separate classes. To separate in two different groups, we can label each rows either as positive sample (Y[i]=+1) or negative sample(Y[i]=-1). Y is the separate labeled vector values for each data/rows.
Apart from labeling each data/row, we provide the weight depending on the feature, which is noted as W. Here, W is defined in a way that we can find a constant b, which provides the threshold for the datasets. When we get the constant we can find a hyperplane to separate these two set of samples with maximum margin.
To denote the formula associated with the sample data,

$$F(x) = W^t X + b = 0 \qquad (1)$$

Where, W = n*1 vector and perpendicular to the line, b = threshold value.

Therefore, distance D between one-sample vectors and hyperplane is denoted as: $D = |F(x)| / ||W||$

In SVM, we try to maximize the margin by finding the line(W,b) which has the distance of S

1 from negative and positive sample.

2 Linear Separable Case (Hard Margin)

Based on Linear separable case, data can be separated with linear line without the errors between those separate data samples by negative or positive labels.

So, we can write following equations for positive and negative distances:

$$W \cdot X + b >= +1 \ \forall \ Y(i) = +1 \quad (2)$$

$$W \cdot X + b <= -1 \ \forall \ Y(i) = -1 \quad (3)$$

Positive samples are labeled as +1 and negative samples are labeled -1, for mathematical convenience, following formula can be derived from (2) and (3).

$$Y(WX + b) >= 1 \quad (4)$$

As noted above that to maximize the margin between hyperplane and negative/positive sample, we need to find both side distance of 1 from the samples. So, if we want to find one sample vector's distance as 1, then our distance to the margin can be represented as

$$\frac{1}{||W||} \quad (5)$$

This way, we try to maximize the distance between one closest vector and hyperplane on both sides. So, our margin becomes as follows:

$$M = 2 / ||W|| \quad (6)$$

2.1 Primal Problem

As in maximizing the margin M in formula(6), w need to minimize the ||W|| which can be written as :

Minimizing:

$$F(W) = 0.5|| W ||^2 \quad (7)$$

s.t.

$$G(X) = Y_i(W*X(i)) + b) >= 1$$

To achieve this, Lagarangian Optimization method can be used and find the minimum ||W|| which can get the maximum margin between two sample datasets.

2.2 Dual Problem

As in convex problem, the above formula's (7) function F (W) and G (X) is composed in a convex set. In my understanding, whenever two formulas are depended on each other and it gives convex set, it is better to use as its dual problem.

We can write the primal problem as its dual form by subtracting the Lagrange Multiplier ($\alpha > 0$) by the conditions (G (X)>0)

$$f'(x) = f(x) - \alpha * g(x)$$
$$L_P = \frac{1}{2}||w||^2 - \sum_i \alpha(i)\{y(i)(x(i) \cdot w + b) - 1\}$$
$$st. \quad \alpha(i) >= 0 \quad \forall i \qquad g(x) = (x(i) \cdot w + b) - 1 >= 0 \quad \forall i \tag{8}$$

As in above formula, introducing the *α helps to weight the support vectors as more than 0 and all others are as 0 which helps us to minimize the W and maximize the α to find optimal hyperplane. That's why we use the dual problem where it eliminates the other samples by α and based on support vectors α, we can get maximum margin hyperplane.*

In order to maximize $f'(\alpha)$, we get derivatives for w and b separately for formula (7).

$$w = \sum_i \alpha(i)y(i)x(i)$$
$$\sum_i \alpha(i)y(i) = 0 \tag{9}$$

If we substitute the W and b in the formula (8) from *α then it gives:*

$$\sum \alpha - \frac{1}{2}\sum_{i,j} \alpha(i)\alpha(j)y(i)y(j)x(i) \cdot x(j) \tag{10}$$

With the KKT constraints $G(x) >= 0$ and $\alpha(i)G(x) = 0$, It means the distance from the positive/negative points to the hyperplane WX+ b is 1. The points are called support vector machine. By the formula (10), we could find the *α* which are bigger than 0 and after putting *α*in the formula (W), it gives W.

3 Linear non-separable case (Soft Margin)

When positive samples and negative samples are separated but some of positive samples are on negative side or negative samples are on positive sides then we cannot get the linear hyperplane. Therefore, we need not to consider those points which are wrongly classified this way we can get the Soft Margin hyperplane where some errors are on both the sides due not able to do separation.

In order to deal with this errors, we involve a slack parameter $\varepsilon > 0$, and update formula (4) as follows. The formula will ignore the errors/samples, which are wrongly classified.

$$Y(WX + b) >= 1 - \varepsilon (i) \tag{11}$$

S

3.1 Primal Problem:

When we transform the primal for avoiding wrongly classified vectors, we gets following formula:

$$f(x) = \frac{1}{2}||w||^2 + C\sum \epsilon(i)$$
$$st. \quad g(x) = y(i)(w \cdot x(i) + b) >= 1 - \epsilon(i) \quad \forall i \qquad (12)$$
$$\epsilon(i) > 0 \quad \forall i$$

Where,
C is a positive constant can be called penalty parameter. Generally, C=1 by default.

3.2 Dual Problem:

According to KKT conditions, we could convert the above primal problem by Lagrange multiplier to its dual form.

$$f'(x) = f(x) - \alpha * g(x) - \beta * h(x)$$
$$f'(x) = \frac{1}{2}||w||^2 + C\sum_i \epsilon(i) - \sum_i \alpha(i)\{y(i)(x(i) \cdot w + b) - |1 + \epsilon(i)\} - \sum_i \beta(i)\epsilon(i)$$
$$st. \quad \alpha(i) >= 0 \quad \forall i \qquad (13)$$
$$g(x) = (x(i) \cdot w + b) - 1 + \epsilon(i) >= 0 \quad \forall i$$
$$\beta(i) >= 0 \quad \forall i$$
$$h(x) = \epsilon >= 0 \quad \forall i$$

solving dual problem Calculating the derivatives of w, b and *s*(*i*) from for- mula(13) separately, we get

$$w = \sum_i \alpha(i)y(i)x(i) \qquad (14)$$

$$-\sum_i \alpha(i)y(i) = 0 \qquad (15)$$

$$C - \alpha(i) - \beta(i) = 0 \quad \forall i \qquad (16)$$

As the distance between hyperplane and support vectors is 1 distance so it should satisfy following formula:

S

$$\alpha(i) * \{(y(i)x(i) + b) - 1 + \epsilon(i) = 0\} \quad (17)$$

Once we get the α *by solving the formula(13). As in constraints* $\alpha g(x)=0$ *and* $\alpha(i) > 0$ *will make sure G(X)=0. After combining we get the constraint as* $\beta h(x) = 0$.

The should be $\alpha(i) < C$, then $s(i) = 0$ gives us the b as threshold from the formula(17) We can get the W according to $0 < \alpha(i) < C$ because of formula(16) $\beta h(x) = 0$. This way we can get maximum soft margin in dual problem.

4 Generalization and duality

4.1 Concepts of generalization error and generalization bound

4.1.1 Generalization error

In machine learning, it is not only considered to get accurate outcume value from training data but mainly it considers about unseen data. We need the better accuracy in unseen data rather than getting in training data. So, when we train the data we try to measure the accuracy of the algorithm for predict outcome, which is called generalization error. As on training data, the prediction does not provide much information while after training the algorithm we can apply those trained variables on unseen data to predict the optimizing output/hyperplane in SVM. Avoiding under and over-fitting of those predicted variables in the algorithm could reduce generalized errors.

4.1.2 Generalization Gap

The bounding value depends on the size of training data. Generalization gap is the predictive performance of the classes of algorithm. Assessing the capacity and risk of the main function/hypothesis on the training data could be denoted as the generalization Gap/bound.

4.2 Duality gap, Strong duality and Weak duality

4.2.1 Duality gap

In mathematical terms, Duality Gap is the different between the dual and primal solution and in SVM; it is difference between two hyperplanes too.
For primal x^* and d^* optimal values duality gap is $x^* - d^*$.
Suppose, F (X) for dual gives (X, X^*) as optimal value and F (Y) for dual gives (Y, Y^*) as optimal value than Duality gap is F (x, 0)- F (0,y) where x and y is all for X and Y values.

4.2.2 Strong duality and Weak duality

Weak duality is same as the optimization solution of the primal SVM. Because of duality gap is greater or equal to 0, which is same constraint to the primal optimization problem.

While, in the Strong duality the duality gap is opposed to the weak duality, which means it, gives different answer than primal optimization problem. Although in some cases only strong duality holds the same condition or constraint as the weak duality.

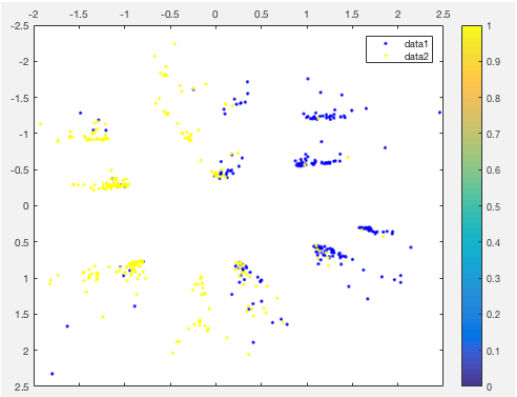
5 Experiments

5.1 Details for Matlab scripts

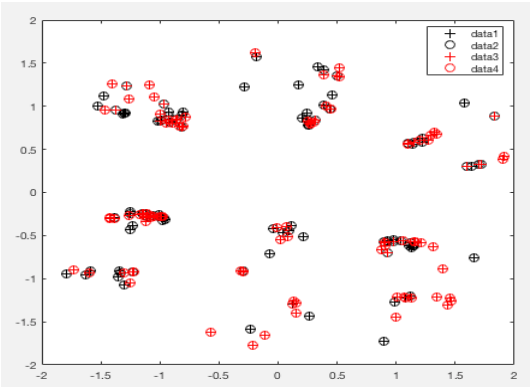
Separate file running for getting alphas, w and b values on data Australian_scale.txt:
1) Run Read_data.m by command [Y,X]=read_data() || Y is label and X is data matrix.
2) Run svmprimal_training for getting primal Ws and b by [model]=svmprimal_training(X, Y, 1, 0) wgere, C=1 as constant to maximize the margin and 0 is epsilon as it does not male difference. This will make a model where we can find alphas.
3) Run main_test.m to find the accuracy which is driven by the svm_predict function. To run command [model2]=main_test()
4) To use plotingData.m run plotingData(model2) where model2 has x and y Here, x is in the 2D data from multidimensional data.
5) Lastly to see boundary, we can run visualizeBountryLinear.m, which gives the optimal hyperplane, but program is not supporting to the dimensions.
Or directly running:
Run 'main_test.m' to check the experiment (based on Australian_scale data from Libsvm) Which gives my code accuracy.

2.1 Experiment

In the experiment, Australian_scale.txt is used as the data from the libsvm.



Fig(a)training samples

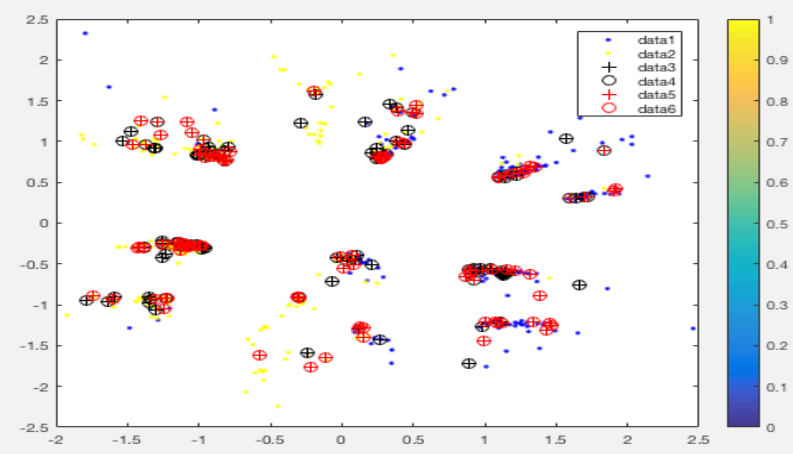


Fig(b)Testing Samples result.

Here, Fig(a) is showing the plotted training samples in different colors in 2D where training samples are in ration (1: 511). Here, yellow filled data is negative samples.

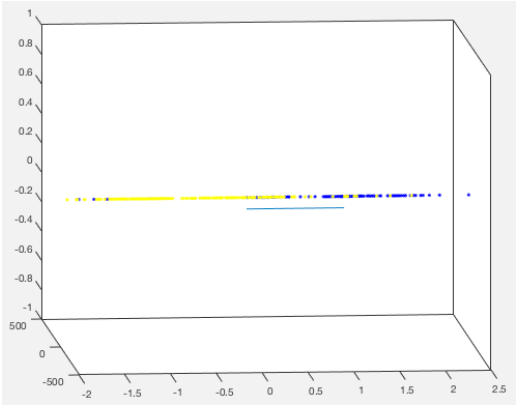
While Fig(b) is illustrating the testing samples(514:690). Here, "+" is used for predicting labels and "O" is used for truth labels where red are negative data and black are positive testing samples.

In Fig(c), the below shown graph is combination of Fig (a) and Fig (b) which gives overview of testing and training data samples.

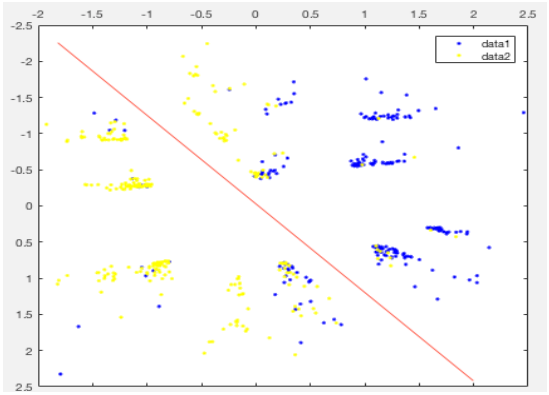


Fig(c) Combined graph of Fig(a) and Fig(b)

After training the samples, I got different Ws and b.
Where, Ws for dual svm on Australian_scale.txt are : -0.0015, -0.0110,-0.0001, 0.0072, 0.0187, -0.0033, 0.0371, 1.0027, 0.0068, 0.0339, -0.0037, 0.0257, -0.0099, 1.0099
And b= 1.0480
My code gave something weird output when I tried to do plot data and visulise the bountry which is down in fig(d):



Fig(d) My Code SVM Dual boundary (wrong)



Fig(e) SVM Boundary from online resource

To solve my hyperplane, I made model2 associated with main_test function which gives me 2D data from 14D and I put it on hold on and tried to draw hyperplane by xp = linspace(min(X(:,1)), max(X(:,1)), 100); yp = (w(1)*xp + b)/w(2); where I plot(xp,yp). Still it did not plot hyperplane properly.

2.2 Comparison with LibSVM

For the experiment, I use the same dataset with libsvm, which is "Australian_scale". I got two group of w and b from primal and dual problems.

1.accuracy of libsvm: 85.7971% (592/690) 2.accuracy of my code: 0.856522 (591/690)
Reason for difference: In my code, it is not checking for cross check which is making some difference.

S

3 Comparison between W and b of Primal and Dual

3.1 Experiment

1 primal problem:

w:

-0.0015, -0.0110, -0.0001, 0.0072, 0.0187, -0.0033, 0.0371, 1.0027, 0.0068, 0.0339, -0.0037, 0.0257, -0.0099, 1.0099
b: 1.0480
Accuracy: 0.8429(152/177)

2 Dual problem
(epsilon=1e-06)
w:
--0.0015, -0.0110,-0.0001, 0.0072, 0.0187, -0.0033, 0.0371, 1.0027, 0.0068, 0.0339, -0.0037, 0.0257, -0.0099, 1.0099
b: 1.0480
Accuracy: 0.858757 (152/177)

3. Dual problem
(epsilon=0)
w:
-0.0027, 0.0050, -0.0128, 0.0189, 0.0218, 0.0126, 0.0232, 1.0054, 0.0042, 0.0485,-0.0024, 0.0188, -0.0474, 1.0168, 0.5539
b=1.0239
Accuracy: 0.856522 (591/690)