UNIVERSIDAD POLITÉCNICA DE CATALUNYA

Support Vector Machines: Primal and Dual

Ignasi Mañé y Joaquim Girbau

Profesor: Daniel Baena

Contents

1 Inroduction

The support vector machine problem consists in finding two parallel hyperplanes within a space $\mathbb{R}^n$ that separate two classes of m points $x_i \in \mathbb{R}^n$ such that we both minimize the classification error and maximize the distance between them. This problem can be formulated as an optimization problem as follows:

$$\begin{array}{lll} \min\limits_{(w,\gamma,s)\in\mathbb{R}^{N+1+m}} & \frac{1}{2}w'w + \nu \sum\limits_{i=1}^{m} s_i & \\ \text{subject to} & d_i(w'\Phi(x_i) - \gamma) + s_i \geq 1, & i = 1, \ldots, m. \\ & s_i \geq 0, & i = 1, \ldots, m. \end{array}$$

Where w is the orthogonal vector to the hyperplanes, γ is the independent term of the hyperplane, s_i is the slack of each constraint, which is 0 if the point x_i is correctly classified and greater than 0 if the point is incorrectly classified, ν is the penalization parameter, d_i is the class of point x_i, and Φ is a mapping $\Phi : \mathbb{R}^n \to \mathbb{R}^N$ applied to point x_i. In this work Φ will be the identity, so $\Phi(x_i) = x_i$.

The Support Vector Machines problem can also be formulated in its equivalent dual form as follows:

$$\begin{array}{ll} \max\limits_{\lambda} & \sum\limits_{i=1}^{m} \lambda_i - \frac{1}{2} \sum\limits_{i=1}^{m} \sum\limits_{j=1}^{m} \lambda_i d_i \lambda_j d_j K_{ij} \\ \text{subject to} & \sum\limits_{i=1}^{m} \lambda_i d_i = 0 \\ & 0 \leq \lambda_i \leq \nu, \quad i = 1, \ldots, m. \end{array}$$

Where λ_i are the Lagrange multipliers, d_i is the class (1 or -1) of point x_i, ν is the penalization parameter, and K_{ij} is the ith jth element of the kernel matrix K that depends of the transformation function applied to points x_i. To retrieve the weights from the dual solution λ, the following identity holds:

$$w = \sum_{i=1}^{m} \lambda_i d_i \Phi(x_i)$$

The independent term γ can be derived choosing some point x_i such that $s_i = 0$ and $\lambda_i > 0$, that is, x_i is a Support Vector, as follows:

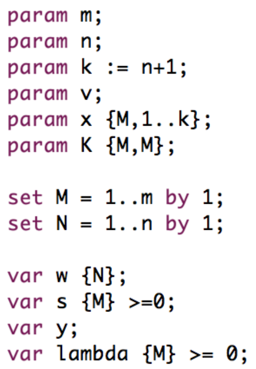
$$\gamma = w'\Phi(x_i) - \frac{1}{d_i}$$

2 AMPL implementation

To formulate and solve the presented problem we will use AMPL, a mathematical modelling language, and CPLEX, which is a solver capable of solving linearly and non linearly constrained programming problems.

2.1 Sets, Parameters and Variables

First, as it is shown in Figure 1, we define in the mod file the sets, the parameters and the variables that are going to be needed.



```
param m;
param n;
param k := n+1;
param v;
param x {M,1..k};
param K {M,M};

set M = 1..m by 1;
set N = 1..n by 1;

var w {N};
var s {M} >=0;
var y;
var lambda {M} >= 0;
```

Figure 1: Sets, parameters and variables

2.1.1 Sets

Set M is the set of observations and set N is the set of variables.

2.1.2 Parameters

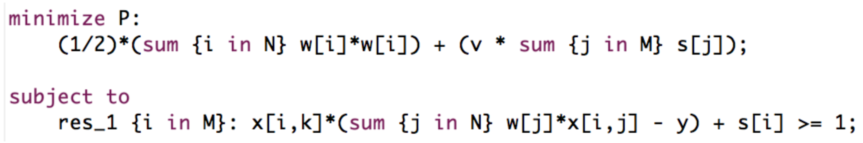
Parameter m is the number of observations or number of points, parameter n is the dimensions of those points used to solve the problem, k is an auxiliary parameter, v is the penalization parameter, x is the observations matrix, and K is the Kernel matrix. Notice that x has more columns than dimensions n because column n+1, which is equal to k, will contain the class value d_i.

2.1.3 Variables

Variable w contains the weights of the optimal hyperplane, variable ”s” contains the slack values, variable y is γ, and lambda contains the dual values of the Dual formulation.

2.2 SVM Primal

This is the AMPL formulation used to model the primal problem. Both the objective function and the constraints are equivalent to those presented in the introduction. Notice, though, that value d_i, as we already mentioned, is denoted by $x\,[i,k]$, that is, the value of column $n+1$.



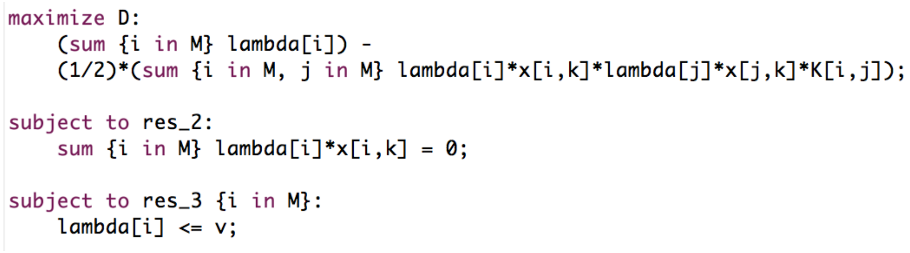
```
minimize P:
    (1/2)*(sum {i in N} w[i]*w[i]) + (v * sum {j in M} s[j]);

subject to
    res_1 {i in M}: x[i,k]*(sum {j in N} w[j]*x[i,j] - y) + s[i] >= 1;
```

Figure 2: AMPL Primal formulation

2.3 SVM Dual

This is the AMPL formulation used to model the dual problem. Again, both the objective function and the constraints are equivalent to those presented in the introduction.



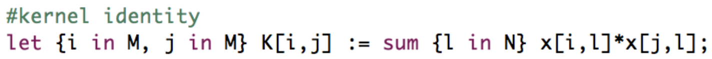
```
maximize D:
    (sum {i in M} lambda[i]) -
    (1/2)*(sum {i in M, j in M} lambda[i]*x[i,k]*lambda[j]*x[j,k]*K[i,j]);

subject to res_2:
    sum {i in M} lambda[i]*x[i,k] = 0;

subject to res_3 {i in M}:
    lambda[i] <= v;
```

Figure 3: Accuracy of the algorithm

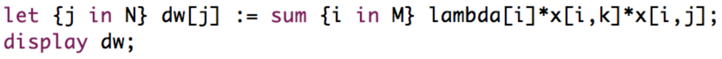
We used the identity Kernel, that is, no transformations were applied to any of the existing variables. Thus, it was calculated and assigned to parameter K as follows in all run files.



```
#kernel identity
let {i in M, j in M} K[i,j] := sum {l in N} x[i,l]*x[j,l];
```

Figure 4: Kernel Identity definition

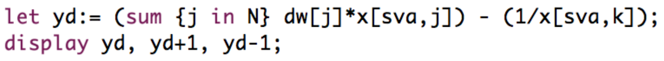
After obtaining the solution through the Dual form of the problem, the weights of the optimal hyperplane were derived using the following expression:



```
let {j in N} dw[j] := sum {i in M} lambda[i]*x[i,k]*x[i,j];
display dw;
```

Figure 5: Retrived weights for Dual form

Finally, after identifying a Suport Vector (observation=sva, that is: min($s = 0$ and $\lambda > 0$)), the independent term could be easily derived using the expression:



```
let yd:= (sum {j in N} dw[j]*x[sva,j]) - (1/x[sva,k]);
display yd, yd+1, yd-1;
```

Figure 6: Retrived independent term for Dual form

3 Primal Results

SVM.run file solves the Primal problem using 50 four dimensional observations generated with the generator program provided for the assignment (dataN50.dat). Since the penalization parameter might influence the optimal hyperplanes, we tried several values of ν to see what happens in both the train and validation set.

The results are summarized in Table 1 in which optimal values for γ and w are shown for each tested value of ν.

v	y	w1	w2	w3	w4
1	3.51	1.92	1.90	1.25	2.52
2	4.21	2.32	2.02	1.67	2.95
3	4.65	2.55	2.14	2.03	3.13
4	5.03	2.71	2.23	2.33	3.35
5	5.74	3.02	2.47	2.63	3.74
6	5.74	3.02	2.47	2.63	3.74
7	5.96	3.12	2.50	2.89	3.86
8	6.21	3.23	2.71	2.96	3.99
9	6.77	3.64	3.03	3.22	4.12
10	7.11	3.96	3.12	3.44	4.10
11	7.21	4.06	3.17	3.49	4.12
12	7.21	4.06	3.17	3.49	4.12
13	7.59	4.23	3.38	3.62	4.32
14	7.98	4.43	3.54	3.79	4.50
15	7.98	4.43	3.54	3.79	4.50
16	7.98	4.43	3.54	3.79	4.50
17	7.98	4.43	3.54	3.79	4.50
18	7.98	4.43	3.54	3.79	4.50
19	7.98	4.43	3.54	3.79	4.50
20	8.03	4.47	3.53	3.84	4.52

Table 1: Primal algorithm results

4 Dual Results

SVM.run file solves the Dual problem using the same 50 four dimensional observations defined in the previous section. Again, we tried the same values of ν but in this case we wanted to confirm that both problems are equivalent and, therefore, that the same hyperplane is obtained for every value ν.

Table 2 confirms that the results obtained coincide with those presented in Table 1. For this case, values w and γ were obtained using the expressions presented in the introduction section.

v	y	w1	w2	w3	w4
1	3.51	1.92	1.90	1.25	2.52
2	4.21	2.32	2.02	1.67	2.95
3	4.65	2.55	2.14	2.03	3.13
4	5.03	2.71	2.23	2.33	3.35
5	5.74	3.02	2.47	2.63	3.74
6	5.74	3.02	2.47	2.63	3.74
7	5.96	3.12	2.50	2.89	3.86
8	6.21	3.23	2.71	2.96	3.99
9	6.77	3.64	3.03	3.22	4.12
10	7.11	3.96	3.12	3.44	4.10
11	7.21	4.06	3.17	3.49	4.12
12	7.21	4.06	3.17	3.49	4.12
13	7.59	4.23	3.38	3.62	4.32
14	7.98	4.43	3.54	3.79	4.50
15	7.98	4.43	3.54	3.79	4.50
16	7.98	4.43	3.54	3.79	4.50
17	7.98	4.43	3.54	3.79	4.50
18	7.98	4.43	3.54	3.79	4.50
19	7.98	4.43	3.54	3.79	4.50
20	8.03	4.47	3.53	3.84	4.52

Table 2: Dual algorithm results

5 Testing a new data set

We generated a new data set of 100 observations (dataN100.dat) to validate the results obtained in the previous sections. Using R and the results of the variables γ and w associated to the optimal hyperplane for each value of ν, we classified the new points using the following rule, and compared them to their theoretical class to compute the classification error.

$$Class = \begin{cases} +1 & \text{if} \quad w'x \geq \gamma + 1 \\ -1 & \text{if} \quad w'x \leq \gamma - 1 \end{cases}$$

Figure 7 shows the summarized results. For every value of $\nu \in \{1, 2, ..., 20\}$ the classification error in both the 50 and 100 observations datasets is represented. Commonly, the classification error computed in the set of observations used to obtain the optimal hyperplane is denoted as the training or in sample error, whereas that computed in any other set of observations is denoted as the out sample or test error. Notice that for large values of ν the difference between the errors is larger than for smaller ones. This is due to the well-known phenomena of over-fitting.

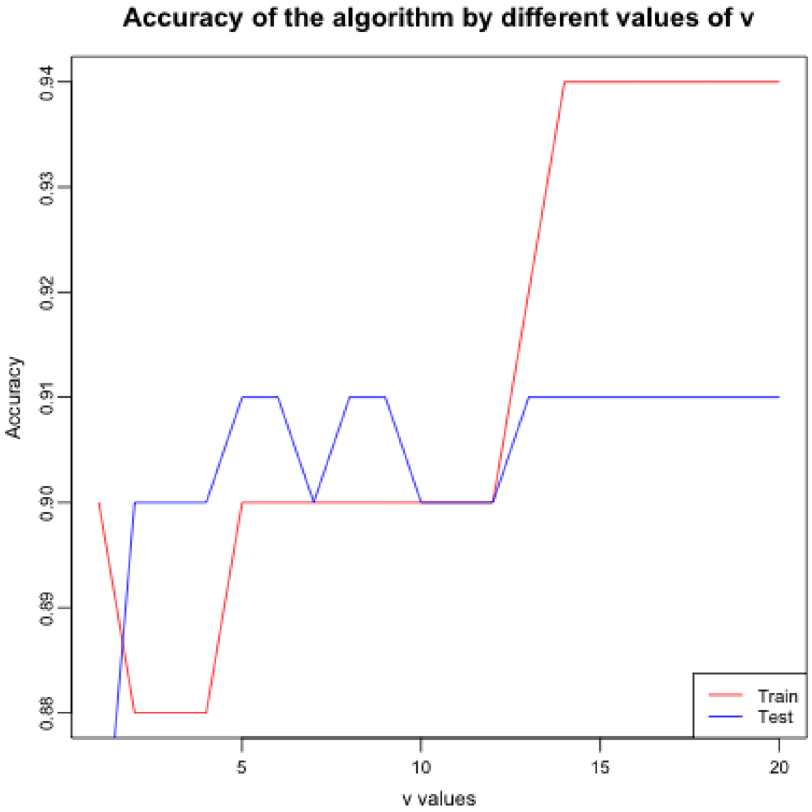


Figure 7: AMPL Dual formulation

6 New real dataset form kaggle

SVMv2.run applies the algorithm for $\nu = 20$ to 200 hundred real observations form this Kaggle dataset. The goal is to distinguish between males and females based on variables height and weight. The data can also be found in the file weight-height.csv attached along with the code and report.

Figure 8 shows a two dimensional plot of variables Height and Weight for each observation in which colour Green or Black identifies the class of each observation. Notice that males tend to be taller and heavier than Females and that it exists a clear positive correlation between both variables.

After applying the SVM algorithm fixing the parameter $\nu = 20$, we obtained the hyperplane represented in red. A total of three SV were also marked with + symbol. Notice that the obtained hyperplane correctly characterizes the classes Male and Female.

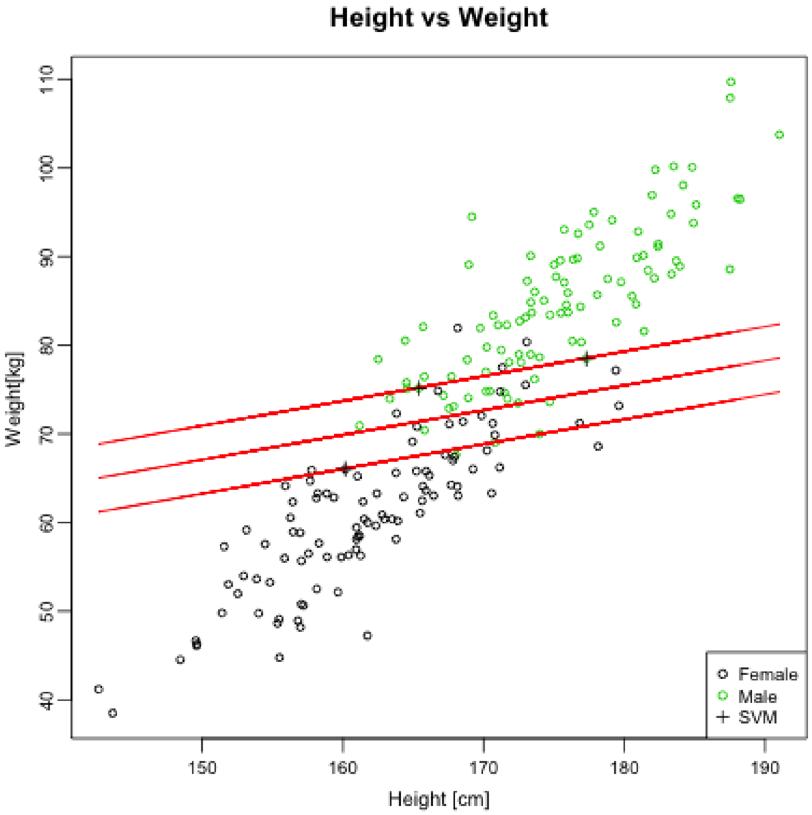


Figure 8: SVM applyed to classify Males and Females

7 Conclusions

In this assignment, the Support Vector Machine Problem has been formulated and solved in two different sets of data.

It has been shown that the problem admits two possible equivalent formulations and that the same results are obtained solving either the Dual or the Primal version.

We conclude that SVM is an computationally efficient and powerful technique that can correctly characterize two classes of observations. However, for linearly non separable data, the performance of the solution found in the train dataset in new samples might depend on the chosen value of ν and the Kernel of data transformation. Some values of this parameter, especially larger ones, might induce the algorithm to over fit the data. That means, correctly characterize the classes in sample, but incorrectly characterize them out of the sample.