Pattern Recognition - ECE 5363

Project 2 Report

Soft SVM Implementation

By
Dan Ni Lin

Professor Name
Dr. Hamed Sari-Sarraf

Department of Electrical and Computer Engineering
Texas Tech University
Lubbock, Texas
2024

Contents

1 Answers

1.1 Part 1: Using CVXOPT Python package to implement Soft SVM

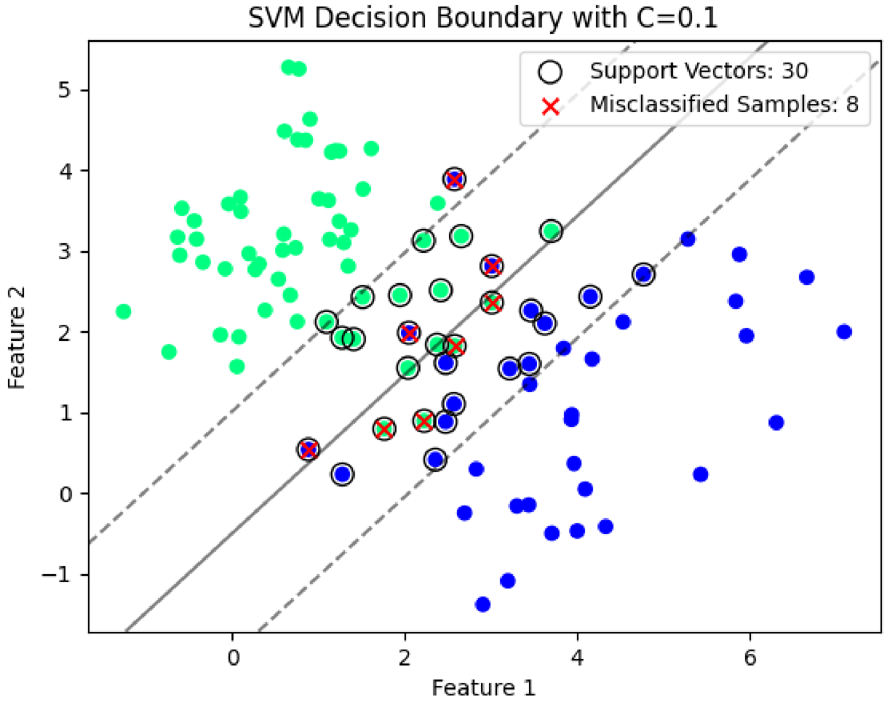

Figure 1: Soft SVM when C=0.1

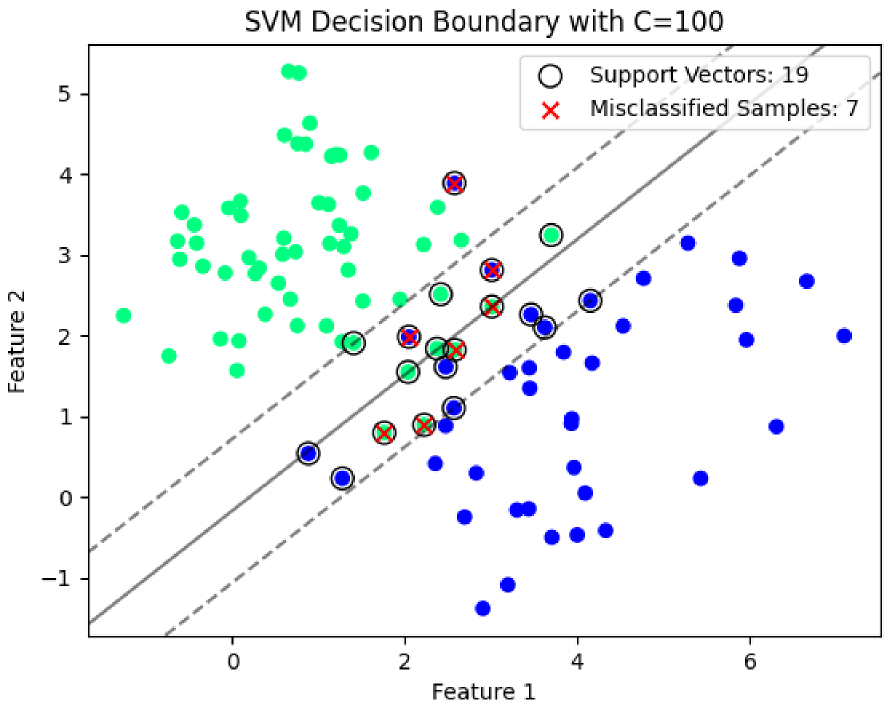

Figure 2: Soft SVM when C=100

1.2 Part 2: Compare computational efficiency of soft SVM implementation with SMO approach

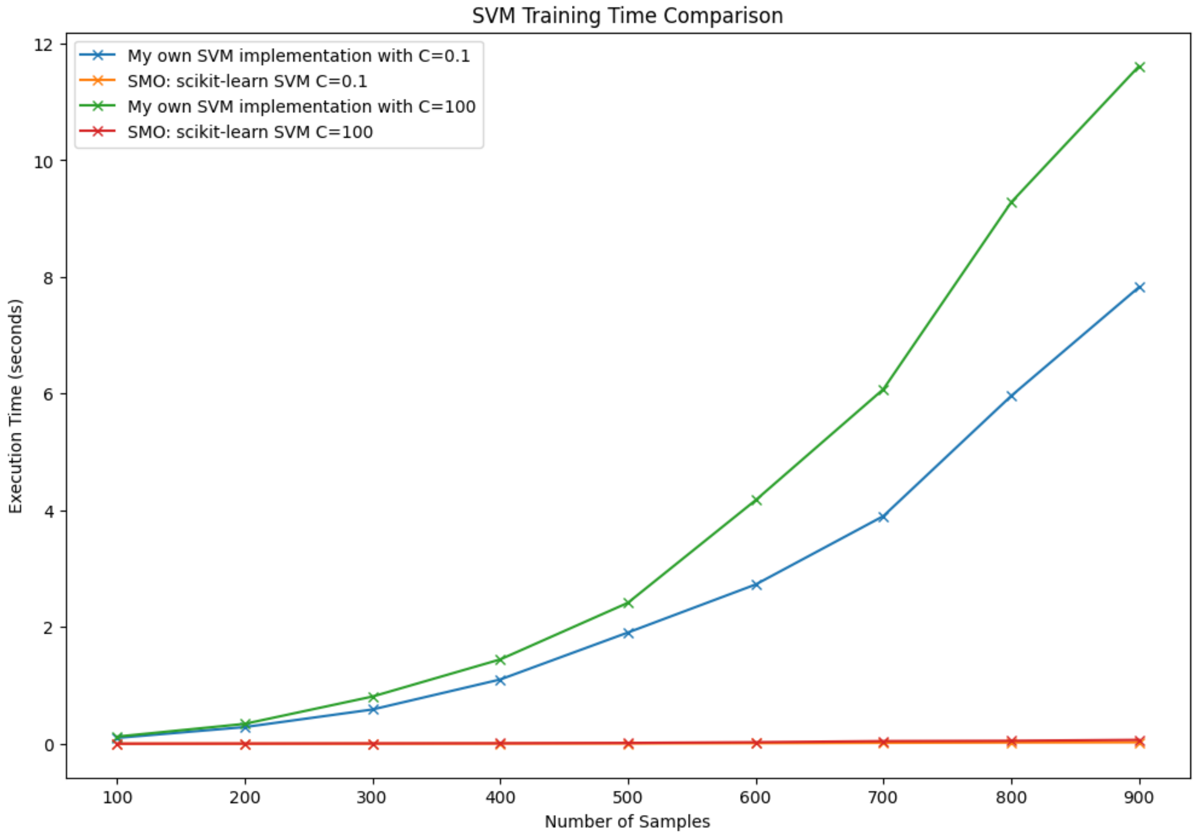

Figure 3: Comparison between SMO python package and our own Soft SVM implementation

2 Dataset Overview

The given dataset contains 100 data points without mentioning the source of the observations. The data provided contains 2 classes:

1. 60 data points that belongs to class 1 with label 1.
2. 40 data points that belongs to class -1 with label -1.

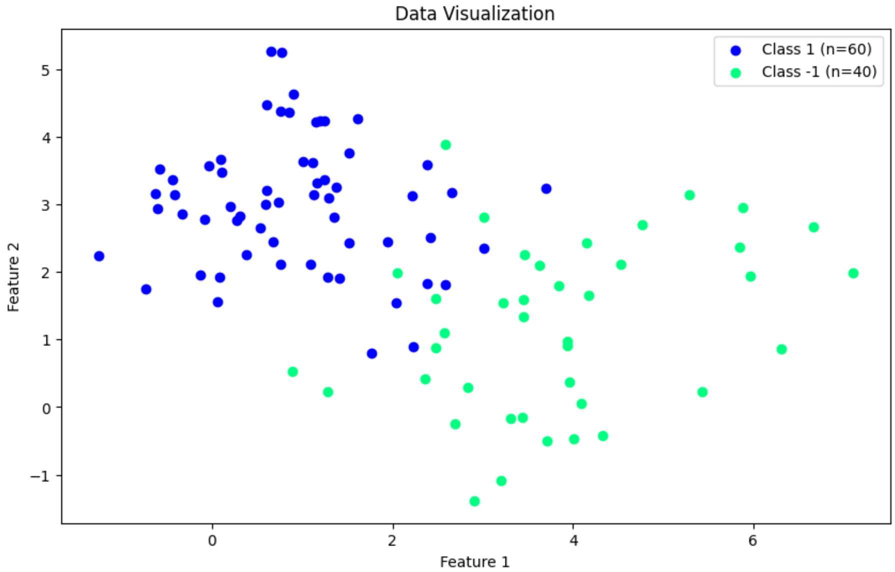

Figure 4: Visualization of the data given in the excel file.

Visually inspecting our data, we can notice that the data points are not linearly separable, then implementing soft SVM is a better option than hard SVM.

	Feature 1	Feature 2
Minimum	-1.26	-1.38
Maximum	7.10	5.27
Mean	2.10	2.27
Variance	3.43	1.98
Within Class Variance	1.41	1.20
Between Class Variance	2.02	0.80

Table 1: Statistical analysis of features from excel data

The statistical analysis of the dataset reveals that is optional to scale data due to the range of values for Feature 1 ($7.1 - (-1.26) = 8.36 \approx 8$) and Feature 2 ($5.27 - (-1.38) = 6.65 \approx 7$), there's no a huge difference in the features values range . By taking a look into the variance, we can observe that Feature 1 variance (3.51) is higher than that of Feature 2 (2.03), indicating a wider spread of data points around the mean.

About the within-class variance values (1.41 for Feature 1 and 1.20 for Feature 2) suggest that data points within each class are relatively almost same tightly clustered, but Feature 2 is slightly more clustered than Feature 1 (this is visually observable from Figure 1).

Finally, between-class variance (2.02 for Feature 1 and 0.80 for Feature 2) measures the degree to which class means differ from the overall mean. A higher between-class variance for Feature 1 suggests it may be more effective in distinguishing between classes when compared to Feature 2.

3 Dual Soft SVM Derivation

The primal problem of Soft SVM given in class is:

$$\min_{\mathbf{w}, w_{l+1}, \boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i$$

Subject to the constraints:

$$\sum_{i=1}^{n} y_i(\mathbf{w}^T \mathbf{x}_i + w_{l+1}) \geq 1 - \xi_i, \quad \forall i \in \{1, \ldots, n\}$$

$$\sum_{i=1}^{n} \xi_i \geq 0, \quad \forall i \in \{1, \ldots, n\}$$

Where:

- $\mathbf{w}$ is the weight vector perpendicular to the hyperplane.
- w_{l+1} is the bias term, which shifts the hyperplane away from the origin.
- $\mathbf{x}_i$ and y_i are the feature vectors and labels (+1 or 1), respectively, of the training data.
- ξ_i are the slack variables that allow misclassification.
- C is the regularization parameter that controls the trade-off between maximizing the margin and minimizing the classification error. It's a hyperparameter.

Now, we need to convert soft SVM primal form into dual form. Then we need to get the Lagrangian, but before that we need to adjust the primal form constraints:

$$-\sum_{i=1}^{n} \left(y_i(\mathbf{w}^T \mathbf{x}_i + w_{l+1}) + 1 - \xi_i\right) \leq 0, \quad \forall i \in \{1, \ldots, n\}$$

$$-\sum_{i=1}^{n} \xi_i \leq 0, \quad \forall i \in \{1, \ldots, n\}$$

Writing the Lagrangian:

$$L(w, w_{l+1}, \xi, \lambda, \mu) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \lambda_i \left(y_i(\mathbf{w}^T \mathbf{x}_i + w_{l+1}) + 1 - \xi_i\right) - \sum_{i=1}^{n} \mu_i \xi_i \tag{2.a}$$

Obtaining the KKT conditions:

$$\frac{d}{dw} = 0 \rightarrow w = \sum_{i=1}^{n} \lambda_i y_i x_i, \quad \forall i \in \{1, \ldots, n\} \tag{2.b}$$

$$\frac{d}{dw_{l+1}} = 0 \rightarrow \sum_{i=1}^{n} \lambda_i y_i = 0, \quad \forall i \in \{1, \ldots, n\} \tag{2.c}$$

$$\frac{d}{d\xi} = 0 \rightarrow C = \lambda_i + \mu_i, \quad \forall i \in \{1, \ldots, n\} \tag{2.d}$$

$$\lambda_i \left(y_i(\mathbf{w}^T \mathbf{x}_i + w_{l+1}) + 1 - \xi_i\right) = 0, \quad \forall i \in \{1, \ldots, n\} \tag{2.e}$$

$$\mu_i \xi_i = 0, \quad \forall i \in \{1, \ldots, n\} \tag{2.f}$$

$$\mu_i, \lambda_i \geq 0, \quad \forall i \in \{1, \ldots, n\} \tag{2.g}$$

Now expanding the Lagrangian in (2.a):

$$
\begin{aligned}
L(w, w_{l+1}, \xi, \lambda, \mu) &= \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\lambda_i\left(y_i(\mathbf{w}^T\mathbf{x}_i + w_{l+1}) + 1 - \xi_i\right) - \sum_{i=1}^{n}\mu_i\xi_i \\
&= \frac{1}{2}w^T w + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\mu_i\xi_i - \sum_{i=1}^{n}\lambda_i y_i x_i w^T - \sum_{i=1}^{n}\lambda_i y_i w_{l+1} + \sum_{i=1}^{n}\lambda_i - \sum_{i=1}^{n}\lambda_i\xi_i
\end{aligned}
$$

Then substitute (2.b) on above expression, we have:

$$
\begin{aligned}
L(w, w_{l+1}, \xi, \lambda, \mu) &= \frac{1}{2}(\sum_{i=1}^{n}\lambda_i y_i x_i)^T(\sum_{i=1}^{n}\lambda_i y_i x_i) + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\mu_i\xi_i \\
&- (\sum_{i=1}^{n}\lambda_i y_i x_i)(\sum_{i=1}^{n}\lambda_i y_i x_i)^T - \sum_{i=1}^{n}\lambda_i y_i w_{l+1} + \sum_{i=1}^{n}\lambda_i - \sum_{i=1}^{n}\lambda_i\xi_i \\
&= \sum_{i=1}^{n}\lambda_i - \frac{1}{2}(\sum_{i=1}^{n}\lambda_i y_i x_i)^T(\sum_{i=1}^{n}\lambda_i y_i x_i) + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\mu_i\xi_i - \sum_{i=1}^{n}\lambda_i y_i w_{l+1} - \sum_{i=1}^{n}\lambda_i\xi_i
\end{aligned}
$$

Because of (2.c), we know that $\sum_{i=1}^{n}\lambda_i y_i w_{l+1} = 0$, then:

$$
L(w, w_{l+1}, \xi, \lambda, \mu) = \sum_{i=1}^{n}\lambda_i - \frac{1}{2}(\sum_{i=1}^{n}\sum_{j=1}^{n}\lambda_i\lambda_j y_i y_j x_i x_j) + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\mu_i\xi_i - \sum_{i=1}^{n}\lambda_i\xi_i
$$

We can rewrite $-\sum_{i=1}^{n}\mu_i\xi_i - \sum_{i=1}^{n}\lambda_i\xi_i$ as $-C\sum_{i=1}^{n}\xi_i$ due to equation (2.d), so:

$$
\begin{aligned}
L(w, w_{l+1}, \xi, \lambda, \mu) &= \sum_{i=1}^{n}\lambda_i - \frac{1}{2}(\sum_{i=1}^{n}\sum_{j=1}^{n}\lambda_i\lambda_j y_i y_j x_i x_j) + C\sum_{i=1}^{n}\xi_i - C\sum_{i=1}^{n}\xi_i \\
&= \sum_{i=1}^{n}\lambda_i - \frac{1}{2}(\sum_{i=1}^{n}\sum_{j=1}^{n}\lambda_i\lambda_j y_i y_j x_i x_j)
\end{aligned}
$$

Looking at (2.d) again and (2.g), we know that

$$
0 \leq \lambda_i \leq C
$$

Finally, the soft SVM dual problem is formulated as:

$$
\max_{\lambda}\sum_{i=1}^{n}\lambda_i - \frac{1}{2}(\sum_{i=1}^{n}\sum_{j=1}^{n}\lambda_i\lambda_j y_i y_j x_i x_j)
$$

Subject to the constraints:

$$
\begin{aligned}
&\sum_{i=1}^{n}\lambda_i y_i = 0, \quad \forall i \in \{1, \ldots, n\} \\
&0 \leq \lambda_i \leq C, \quad \forall i \in \{1, \ldots, n\}
\end{aligned}
$$

4 Soft SVM Implementation

This section outlines the implementation of the soft-margin Support Vector Machine (SVM) as a Quadratic Programming (QP) problem, solvable by `cvxopt`.

4.1 QP Problem Formulation

The generic QP problem solved by `cvxopt` is formulated as:

$$\min_{\mathbf{x}} \frac{1}{2}\mathbf{x}^T P\mathbf{x} + \mathbf{q}^T\mathbf{x}$$

subject to:

$$G\mathbf{x} \preceq h,$$

$$A\mathbf{x} = b.$$

4.2 Adapting the SVM Dual Formulation

4.2.1 Objective Function

To adapt our soft SVM dual form formulation for minimization (as `cvxopt` inherently solves minimization problems), we convert the maximization problem into a minimization problem by multiplying by -1:

$$\min_{\lambda} \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\lambda_{\mathbf{i}}\lambda_{\mathbf{j}}y_i y_j x_i^\top x_j - \sum_{i=1}^{n}\lambda_{\mathbf{i}}$$

4.2.2 Constraints

The soft SVM dual formulation is subject to the following constraints:

$$\sum_{i=1}^{n}\lambda_{\mathbf{i}}y_i = 0,$$

$$0 \leq \lambda_{\mathbf{i}} \leq C, \quad \forall i \in \{1, \ldots, n\}.$$

4.3 Matrix Formulation

4.3.1 Objective Function

Writing the soft SVM dual formulation into the QP problem form, we define:

$$\min_{\lambda} \frac{1}{2}\lambda^\top(YXX^\top Y)\lambda - 1^\top\lambda$$

4.3.2 Constraints

The constraints matrix formulation is:

$$Y^T\lambda = 0$$

$$-I\lambda \leq 0$$

$$I\lambda \leq C$$

$$\lambda \geq 0$$

4.4 Input Matrices for `cvxopt`

Since our problem is to minimize over $\mathbf{x} = \boldsymbol{\lambda}$, then for the QP problem solved by `cvxopt` and by looking at Section 3.3, the input matrices are defined as follows:

- $P = YXX^\top Y$, which is equivalent to $y \otimes y * XX^\top$, where $\otimes$ denotes the outer product and $*$ is the element-wise multiplication.
- $q = -\mathbf{1}^\top$, indicating a column vector of ones made negative.
- $A = Y^\top$, representing the transpose of the label matrix Y.
- $b = 0$, a vector of 0's.

The matrix G and vector h are given by:

$$G = \begin{bmatrix} -I \\ I \end{bmatrix},$$

where I is the $n \times n$ identity matrix, and

$$h = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ C \\ \vdots \\ C \end{bmatrix},$$

with the first n entries being 0 for the constraint $-\lambda_i \leq 0$, and the next n entries being C for the constraint $\lambda_i \leq C$.

5 Computing the boundaries

By solving the dual problem, our output will be a vector of λ. The λ's determines which data point will be a support vector or not. Since the convex optimization package will not give a $\lambda = 0$, we need to set a threshold to determine which data points will be indeed a support vector, therefore in our case we set that:

$$\lambda = \begin{cases} \lambda & \text{if } \lambda > 10^{-4} \\ 0 & \text{otherwise} \end{cases}$$

The soft SVM decision boundarie is given by:

$$d(x) = \langle w, x \rangle + w_{l+1}$$

So we need to find the values of w and w_{l+1}.

5.1 Finding the weigth vector

We can find w from eq. (2.b) which refers to one of the KKT condition, then w is obtain by:

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

The matrix form is:

$$w = X^T \alpha * Y$$

,where $*$ stands for element-wise multiplication and the expected dimensions for X is $n_{\text{samples}} \times n_{\text{features}}$; for α is $n_{\text{samples}} \times 1$; for Y is $n_{\text{samples}} \times 1$ and for w is $n_{\text{features}} \times 1$.

5.2 Finding the offset

To obtain w_{l+1} we can look at the general equation to compute the marginal boundaries:

$$y_i(\mathbf{w}^T \mathbf{x_i} + w_{l+1}) = 1$$

Now let's isolate w_{l+1}:

$$\begin{aligned} y_i(\mathbf{w}^T \mathbf{x_i} + w_{l+1}) &= 1 \\ \mathbf{w}^T \mathbf{x_i} + w_{l+1} &= \frac{1}{y_i} \\ w_{l+1} &= \frac{1}{y_i} - \mathbf{w}^T \mathbf{x_i} \end{aligned}$$

Since y is 1 or -1 is equivalent to write that w_{l+1} is equal to:

$$w_{l+1} = y_i - \mathbf{w}^T \mathbf{x_i}$$

The above equation shows that we will have more than one possible w_{l+1}, therefore to pick the best choice of w_{l+1}, we can take the average of the w_{l+1}'s and pick the averaged $\bar{w}_{l+1}$ as our model offset, then:

$$\bar{w}_{l+1} = \frac{1}{SV} \sum_{i \in SV} (y_i - \langle w, x_i \rangle)$$

, $|SV| \in N$ stands for the number of support vectors from the model.

The matrix form will be:

$$\bar{w}_{l+1} = \frac{1}{SV}\left(Y - Xw\right)$$

,where the expected dimensions for X is $n_{\text{samples}} \times n_{\text{features}}$; for Y is $n_{\text{samples}} \times 1$; for w is $n_{\text{features}} \times 1$ and for $\bar{w}_{l+1}$ is 1×1 which stands for a scalar.

5.3 Decision boundaries equations

Once computed w and w_{l+1}, we can plot the soft SVM decision boundaries which are given by:

- The central decision boundary, where the decision function $d(\mathbf{x}) = 0$, is represented by:

$$\langle \mathbf{w}, \mathbf{x} \rangle + b = 0$$

- The positive margin, where the decision function $d(\mathbf{x}) = 1$, supporting the boundary for one class, is defined by:

$$\langle \mathbf{w}, \mathbf{x} \rangle + b = 1$$

- The negative margin, where the decision function $d(\mathbf{x}) = -1$, supporting the boundary for the other class, is defined by:

$$\langle \mathbf{w}, \mathbf{x} \rangle + b = -1$$

References

[1] Dr. Sarraf's Class notes.