Two Different Frameworks for Support Vector Machines

Alessandro Gentili

July 24, 2024

1 Introduction

1.1 Problem

Support Vector Machines (SVMs) are powerful supervised learning models used for classification tasks. [11] They aim to find the optimal hyperplane that separates different classes in a dataset, maximizing the margin between the classes. SVMs have been widely adopted due to their effectiveness in high-dimensional spaces and their versatility achieved through the use of different kernel functions. We develop two different frameworks for implementing SVMs:

1. max margin
2. empirical risk minimization

We will analyze the mathematical foundations, implementation of respective learning algorithms, and experimental results for each framework using our custom-built SVM classifier.

1.2 Experiment Design

we analyze the two different frameworks independently. The datasets are created synthetically in order to achieve the maximum clearness and obtain a good visualization. We create two different synthetic datasets of 100 samples each:

1. linear separable
2. non linear separable

The dataset are the same for the two frameworks, except for labeling: the empirical risk minimization framework got $\{0, 1\}$ labels, the other $\{-1, 1\}$. We will analyze the mathematical foundations, implementation of respective learning algorithms, and experimental results for each framework using our custom-built SVM classifier. In both cases we perform kernel trick only with linear kernels. [6]

2 The Max Margin Framework

The max margin framework is a fundamental concept in Support Vector Machines (SVMs). The core idea is to find the hyperplane that maximizes the distance between the closest data points of different classes. This distance is referred to as the "margin".

2.1 Mathematical Background

For a hyperplane defined by $w^T x + b = 0$, where w is the normal vector to the hyperplane and b is the bias term, the margin is the perpendicular distance from the hyperplane to the nearest data point. Mathematically, this distance is given by:

$$\text{margin} = \frac{1}{\|w\|} \tag{1}$$

The primal form of the SVM optimization problem can be stated as:

$$\text{minimize} \quad \frac{1}{2}\|w\|^2 \tag{2}$$

$$\text{subject to} \quad y_i(w^T x_i + b) \geq 1, \quad \forall i = 1, \ldots, n \tag{3}$$

Here, we minimize $\frac{1}{2}\|w\|^2$ instead of maximizing $\frac{2}{\|w\|}$, which is equivalent but easier to optimize. Using Lagrangian multipliers [7], we can derive the dual formulation:

$$\text{maximize} \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j (x_i^T x_j) \tag{4}$$

$$\text{subject to} \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{5}$$

$$0 \leq \alpha_i \leq C, \quad \forall i = 1, \ldots, n \tag{6}$$

Where α_i are the Lagrange multipliers and C is the regularization parameter. [8] The KKT conditions [5] provide necessary and sufficient conditions for optimality:

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{n} \alpha_i y_i x_i = 0 \tag{7}$$

$$\frac{\partial L}{\partial b} = -\sum_{i=1}^{n} \alpha_i y_i = 0 \tag{8}$$

$$y_i(w^T x_i + b) - 1 + \xi_i \geq 0 \tag{9}$$

$$\alpha_i \geq 0 \tag{10}$$

$$\xi_i \geq 0 \tag{11}$$

$$\alpha_i(y_i(w^T x_i + b) - 1 + \xi_i) = 0 \tag{12}$$

$$(C - \alpha_i)\xi_i = 0 \tag{13}$$

These conditions are crucial for the SMO algorithm.

2.2 The SMO Algorithm

The Sequential Minimal Optimization (SMO) algorithm, proposed by John Platt [2], is an efficient method for solving the quadratic programming problem arising from the SVM optimization. The key idea is to break the problem into a series of smallest possible sub-problems, which are then solved analytically. Based on the provided implementation, the algorithm consists of the following main components:

1. **Initialization:** Set up initial parameters including regularization parameter C, numerical tolerance, and maximum number of passes.
2. **Main Loop:** Iterate through the training examples, examining and optimizing pairs of Lagrange multipliers.
3. **Example Examination:** For each example, check if it violates the KKT conditions and select a second example for joint optimization.
4. **Optimization Step:** Solve the constrained optimization problem for the selected pair of multipliers.
5. **Update:** Update the SVM model parameters based on the optimization results.

The core of SMO is the analytical solution for two multipliers. Given α_i and α_j, we compute the new value for α_j as follows:

$$\alpha_j^{new} = \alpha_j^{old} + y_j \frac{(E_i - E_j)}{\eta} \tag{14}$$

Where:

- $E_i = f(x_i) - y_i$ is the error for the i-th training example
- $\eta = K_{ii} + K_{jj} - 2K_{ij}$ is the second derivative of the objective function along the diagonal line
- $K_{ij} = K(x_i, x_j)$ is the kernel function (linear kernel in this implementation)

The implementation includes several key components:

1. **Error Caching:** Maintains an error cache to speed up computations.
2. **Second Choice Heuristic:** Selects the second example that maximizes the step size.
3. **Bound Constraints:** Ensures that the optimized values of α_i and α_j respect the box constraints.

4. **Threshold Updating:** Updates the bias term b after each optimization step.

5. **Stopping Criteria:** Uses a combination of examining all examples and only examining examples with non-bound α values.

The algorithm terminates when no examples violate the KKT conditions within the specified tolerance, or when the maximum number of passes is reached.

2.3 Results

We applied the SMO algorithm to train SVMs on both linearly separable and non-linearly separable datasets, using different values of the regularization parameter C.

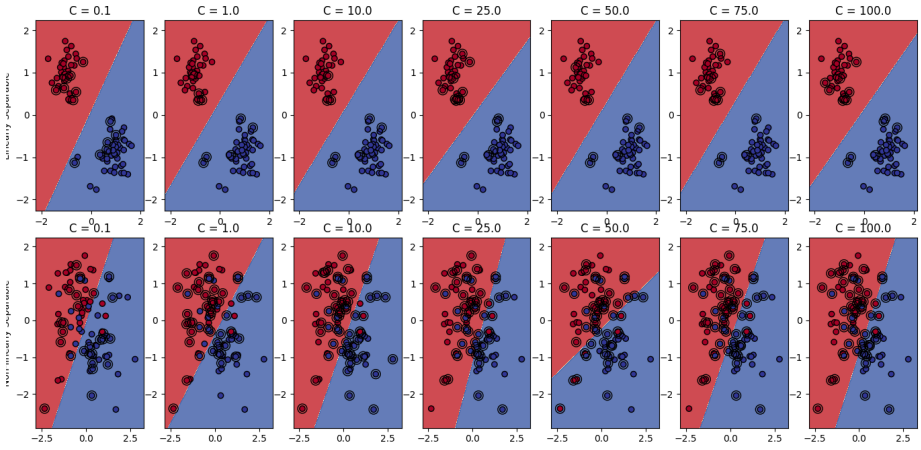

Figure 1: SMO classification performance

For the linearly separable dataset:

- The SVM achieved high accuracy
- As C increased, the margin became narrower, potentially leading to overfitting for very large C values.
- The number of support vectors decreased as C increased, until reaching a plateau.

For the non-linearly separable dataset:

- The SVM's performance was more sensitive to the choice of C.
- Moderate C values generally gave the best balance between bias and variance.

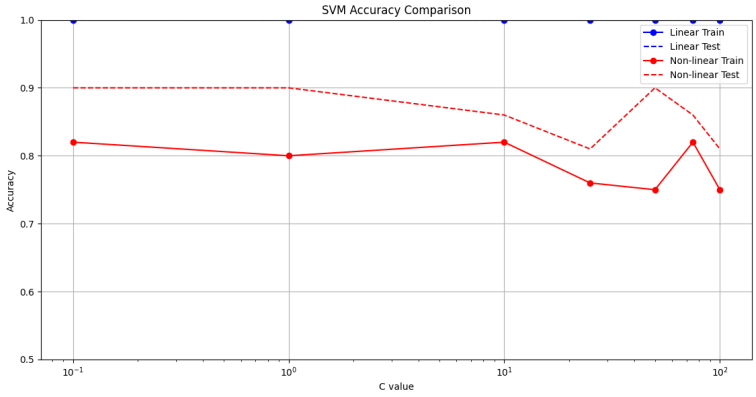

Figure 2: comparing SMO accuracy scores

- The number of support vectors was generally higher than in the linear case, reflecting the increased complexity of the decision boundary.

Key observations:

- The SMO algorithm converged faster than gradient-based methods, especially for larger datasets.
- The algorithm was particularly effective for problems with many support vectors.

These results highlight the importance of proper hyperparameter tuning in SVM training, particularly the regularization parameter C and the kernel function choice for non-linear problems. The SMO algorithm proved to be an efficient and effective method for training SVMs, capable of handling both linear and non-linear classification tasks.

3 The Empirical Risk Minimization Framework

Empirical Risk Minimization (ERM) is a principle in statistical learning theory that forms the basis for many machine learning algorithms, including Support Vector Machines (SVMs). [3] The core idea is to minimize the average loss over the training data, which serves as an estimate of the expected loss over the entire data distribution. The objective to minimize is:

$$J(w, b) = \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(x_i)) + \lambda R(w) \tag{15}$$

Where:

- n is the number of training samples
- $L(y_i, f(x_i))$ is the loss function
- f if a function that map a input point into a class label. in our case it is $f(x_i) = w^T x_i + b$
- $R(w)$ is the regularization term
- λ is the regularization parameter that balances the trade-off between minimizing the loss and the complexity of the model

3.1 Mathematical Background

For SVMs [9], we typically use the hinge loss function [4]:

$$L(y_i, f(x_i)) = \max(0, 1 - y_i f(x_i)) = \max(0, 1 - y_i(w^T x_i + b)) \tag{16}$$

The hinge loss penalizes misclassifications and points inside the margin, encouraging a decision boundary with a wide margin.

For regularization [1], we consider three options:

1. L2 regularization:

$$R(w) = \|w\|_2^2 = \sum_{j=1}^{d} w_j^2 \tag{17}$$

2. L1 regularization:

$$R(w) = \|w\|_1 = \sum_{j=1}^{d} |w_j| \tag{18}$$

3. Elastic Net regularization:

$$R(w) = \alpha\|w\|_1 + (1 - \alpha)\|w\|_2^2 \tag{19}$$

where α is a mixing parameter between L1 and L2 regularization.

Combining these, our full optimization problem becomes:

$$\min_{w,b} \frac{1}{n} \sum_{i=1}^{n} \max(0, 1 - y_i(w^T x_i + b)) + \lambda R(w) \tag{20}$$

3.2 The Subgradient Descent Algorithm

The subgradient descent algorithm is an extension of gradient descent that can handle non-differentiable objective functions. [10] This is particularly useful in our case because the hinge loss function is not differentiable at the point where

the margin equals 1, but it is still convex. A subgradient of a function f at a point x is any vector g that satisfies:

$$f(y) \geq f(x) + g^T(y - x) \quad \forall y \tag{21}$$

In particular we can say that a function is differentiable if and only if the subgradient contains only the gradient, at each point. For the hinge loss, we can define the subgradient with respect to w as:

$$\nabla_w L = \begin{cases} -y_i x_i & \text{if } y_i(w^T x_i + b) < 1 \\ 0 & \text{otherwise} \end{cases} \tag{22}$$

And with respect to b:

$$\nabla_b L = \begin{cases} -y_i & \text{if } y_i(w^T x_i + b) < 1 \\ 0 & \text{otherwise} \end{cases} \tag{23}$$

The subgradient descent algorithm for training an SVM proceeds as follows:

1. Initialize w and b
2. For each epoch:
   (a) Shuffle the training data
   (b) For each mini-batch:
      i. Compute the subgradients of the loss and regularization terms
      ii. Update w and b:

$$w \leftarrow w - \eta(\nabla_w L + \lambda \nabla_w R) \tag{24}$$
$$b \leftarrow b - \eta \nabla_b L \tag{25}$$

      where η is the learning rate
3. Repeat until convergence or maximum epochs reached

3.3 Results

We applied our SVM implementation with subgradient descent to both linearly separable and non-linearly separable datasets, using different regularization terms (L1, L2, and Elastic Net) and various values of the regularization parameter C (inverse of λ).

For the linearly separable dataset:

- All regularization methods performed well, achieving high accuracy (¿95
- L2 regularization showed slightly more stable performance across different C values.

- L1 regularization led to sparser models (smaller $\|w\|$) but sometimes at the cost of slightly lower accuracy.
- Elastic Net regularization provided a balance between L1 and L2, often achieving high accuracy with moderate sparsity.

For the non-linearly separable dataset:

- Overall performance was lower than for the linearly separable case, as expected.
- L2 regularization generally outperformed L1, especially for larger C values.
- Elastic Net often provided the best trade-off, achieving good accuracy while maintaining some sparsity.
- All methods showed improved performance with increasing C up to a point, after which overfitting became evident.

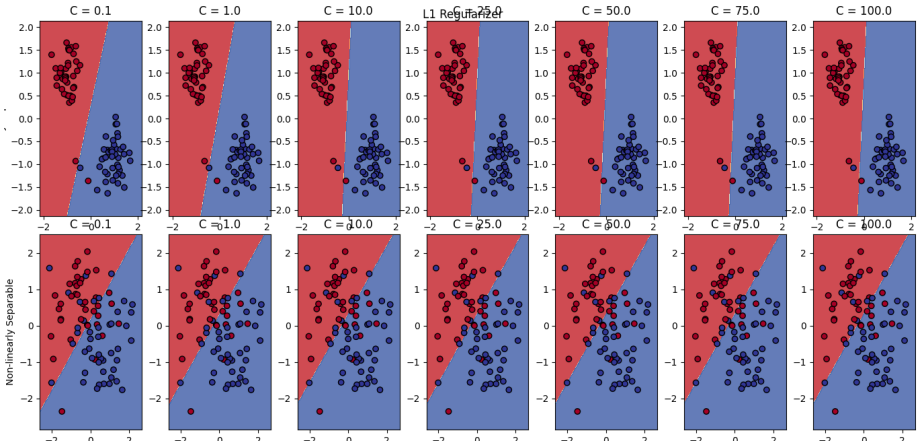

Figure 3: $L1$ regularization

We can observe that:

1. For both datasets, there's a general trend of increasing accuracy with increasing C, up to a certain point.
2. The gap between training and test accuracy tends to widen for very large C values, indicating overfitting.
3. L2 regularization (red lines) tends to provide more stable performance across different C values.
4. L1 regularization (blue lines) sometimes achieves high accuracy with lower C values, indicating effective feature selection.

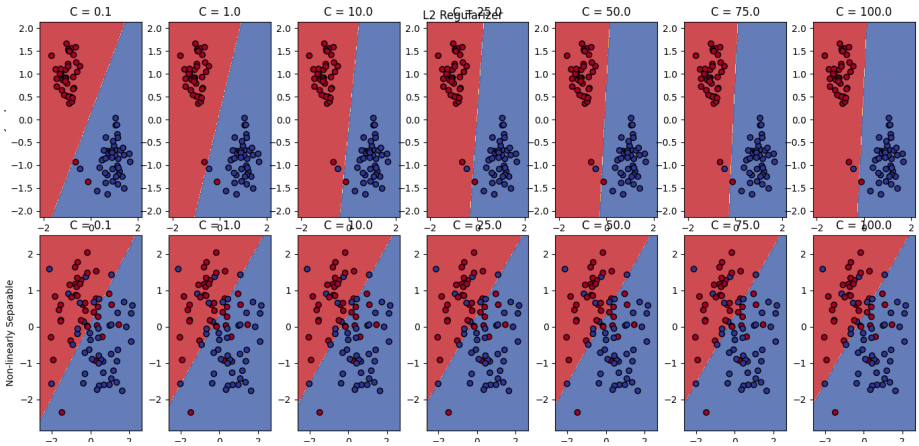

Figure 4: $L2$ regularization

Figure 5: $ElasticNet$ regularization

5. Elastic Net regularization (green lines) often provides a good balance, achieving high accuracy while avoiding extreme overfitting.

Figure 6: comparison accuracy scores

These results demonstrate the importance of proper regularization and hyperparameter tuning in SVM training. The choice of regularization method can significantly impact model performance and characteristics, and should be selected based on the specific requirements of the problem at hand.

4 Possible Improvements

Our exploration of two different frameworks for Support Vector Machines - the Max Margin framework and the Empirical Risk Minimization framework - has provided valuable insights into their performance and characteristics. Based on our findings, we can identify several areas for potential improvements and further research:

1. **Kernel Optimization**: While our study focused primarily on linear kernels, expanding the implementation to include more sophisticated kernel functions (e.g., polynomial, radial basis function) could significantly improve performance on non-linearly separable datasets. Future work could explore automatic kernel selection or kernel parameter optimization techniques.

2. **Hyperparameter Tuning**: Both frameworks showed sensitivity to hyperparameters, particularly the regularization parameter C. Implementing advanced hyperparameter tuning methods such as Bayesian optimization or random search could lead to more robust and efficient model selection.

3. **Multi-class Classification**: Extend the current binary classification setup to handle multi-class problems using techniques like one-vs-one or one-vs-all approaches. This would broaden the applicability of our SVM implementations to a wider range of real-world problems.

4. **Online Learning**: Implement online versions of both algorithms to handle large-scale or streaming data more efficiently. This could involve techniques like stochastic gradient descent for the ERM framework or online SMO variants for the max margin approach.

5. **Interpretability**: Develop methods to enhance the interpretability of the SVM models, such as feature importance ranking or decision boundary visualization techniques, to provide more insights into the model's decision-making process.

6. **Robustness to Outliers**: Investigate and implement robust SVM variants that are less sensitive to outliers and noise in the training data.

7. **Convergence Analysis**: Conduct a more rigorous theoretical analysis of the convergence properties of both algorithms, particularly for the subgradient descent method with different regularization terms.

8. **Hybrid Approaches**: Explore the possibility of combining elements from both frameworks to create hybrid SVM algorithms that leverage the strengths of each approach.

9. **Benchmark Against State-of-the-Art**: Conduct a comprehensive comparison of our implementations against state-of-the-art SVM libraries and other machine learning algorithms on a diverse set of benchmark datasets.

By addressing these areas, we can further improve the performance, efficiency, and applicability of our SVM implementations. Future research in these directions will contribute to the ongoing development of SVM techniques and their practical applications in various domains of machine learning and data analysis.

References

[1] Daniel Lopez Martinez. Regularization approaches for support vector machines with applications to biomedical data. *CoRR*, abs/1710.10600, 2017.

[2] John Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. *Advances in Kernel Methods-Support Vector Learning*, 208, 07 1998.

[3] Wikipedia contributors. Empirical risk minimization — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Empirical_risk_minimization&oldid=1228458708, 2024. [Online; accessed 24-July-2024].

[4] Wikipedia contributors. Hinge loss — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Hinge_loss&oldid=1193339909, 2024. [Online; accessed 24-July-2024].

[5] Wikipedia contributors. Karush–kuhn–tucker conditions — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Karush%E2%80%93Kuhn%E2%80%93Tucker_conditions&oldid=1228996335, 2024. [Online; accessed 24-July-2024].

[6] Wikipedia contributors. Kernel method — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Kernel_method&oldid=1216509663, 2024. [Online; accessed 24-July-2024].

[7] Wikipedia contributors. Lagrange multiplier — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Lagrange_multiplier&oldid=1234844318, 2024. [Online; accessed 24-July-2024].

[8] Wikipedia contributors. Regularization (mathematics) — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Regularization_(mathematics)&oldid=1231615120, 2024. [Online; accessed 24-July-2024].

[9] Wikipedia contributors. Regularization perspectives on support vector machines — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Regularization_perspectives_on_support_vector_machines&oldid=1227520636, 2024. [Online; accessed 24-July-2024].

[10] Wikipedia contributors. Subgradient method — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Subgradient_method&oldid=1201924934, 2024. [Online; accessed 24-July-2024].

[11] Wikipedia contributors. Support vector machine — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Support_vector_machine&oldid=1235056835, 2024. [Online; accessed 24-July-2024].