Module 6: SVM Example - Parameter Selection and Sentiment Analysis

Machine Learning Course

Contents

1 Introduction to SVM Parameter Selection

1.1 The Importance of Parameter Tuning

Support Vector Machines (SVMs) are powerful classification algorithms that find the optimal hyperplane to separate data points of different classes. However, their performance heavily depends on the proper selection of hyperparameters, particularly the regularization parameter C. This document explores a practical example of applying SVMs to sentiment analysis and demonstrates the critical role of parameter tuning in achieving optimal performance.

1.2 Review of the Soft-Margin SVM

Before diving into the example, let's briefly review the soft-margin SVM formulation:

Soft-Margin SVM Optimization Problem

$$\min_{w \in \mathbb{R}^d, b \in \mathbb{R}, \xi \in \mathbb{R}^n} \|w\|^2 + C \sum_{i=1}^{n} \xi_i \tag{1}$$

$$\text{subject to: } y^{(i)}(w \cdot x^{(i)} + b) \geq 1 - \xi_i \text{ for all } i = 1, 2, ..., n \tag{2}$$

$$\xi_i \geq 0 \text{ for all } i = 1, 2, ..., n \tag{3}$$

The soft-margin SVM introduces slack variables ξ_i that allow for some misclassifications in the training data. The parameter C controls the trade-off between maximizing the margin (minimizing $\|w\|^2$) and minimizing the classification error (minimizing $\sum_{i=1}^{n} \xi_i$).

2 Understanding the Regularization Parameter C

2.1 The Role of Parameter C

The parameter C in the soft-margin SVM formulation serves as a regularization parameter that controls the trade-off between two competing objectives:

- **Maximizing the margin**: Achieved by minimizing $\|w\|^2$
- **Minimizing training errors**: Achieved by minimizing $\sum_{i=1}^{n} \xi_i$

The value of C determines the relative importance of these objectives:

- **Small** C: Places more emphasis on maximizing the margin, even if it means allowing more training errors
- **Large** C: Places more emphasis on minimizing training errors, potentially at the cost of a smaller margin

2.2 Geometric Interpretation

The parameter C affects the geometry of the decision boundary:

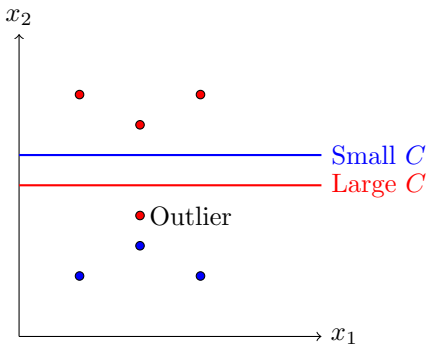

Figure 1: Effect of parameter C on the decision boundary. A small C (blue line) allows the outlier to be misclassified to maintain a larger margin. A large C (red line) adjusts the boundary to correctly classify the outlier, resulting in a smaller margin.

2.3 Practical Implications

The choice of C has significant practical implications:

- **Overfitting vs. Underfitting**: Large C values can lead to overfitting, while small C values might result in underfitting
- **Number of Support Vectors**: Larger C values typically result in fewer support vectors
- **Generalization Performance**: The optimal C value balances training accuracy and generalization to unseen data

3 Sentiment Analysis Case Study

3.1 Dataset Description

The case study uses a sentiment analysis dataset with the following characteristics:

- **Source**: Reviews from Amazon, Yelp, and IMDB
- **Labels**: Binary classification (positive or negative sentiment)
- **Representation**: Bag-of-words with a vocabulary of 4500 words
- **Size**: 2500 training sentences, 500 test sentences

3.2 Example Sentences

The dataset contains sentences like:

- "Needless to say, I wasted my money." (Negative)

- "He was very impressed when going from the original battery to the extended battery." (Positive)
- "I have to jiggle the plug to get it to line up right to get decent volume." (Negative)
- "Will order from them again!" (Positive)

3.3 Feature Representation

The bag-of-words representation transforms each sentence into a high-dimensional vector:

- Each dimension corresponds to a word in the vocabulary
- The value in each dimension represents the presence or frequency of the word in the sentence
- This creates a sparse vector representation (most entries are zero)

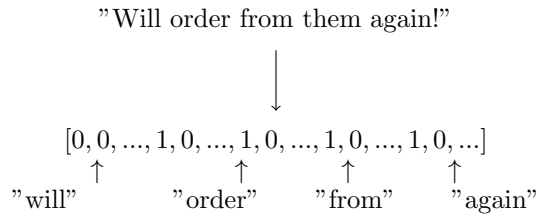

Figure 2: Bag-of-words representation of a sentence. Each word in the vocabulary corresponds to a dimension in the feature vector.

4 Parameter Selection through Cross-Validation

4.1 The Cross-Validation Procedure

To find the optimal value of C, the study employs 5-fold cross-validation:

1. Divide the training data into 5 equal folds
2. For each value of C:
   (a) For each fold i (from 1 to 5):
      i. Train an SVM on all folds except fold i
      ii. Evaluate the model on fold i
   (b) Calculate the average error across all 5 folds
3. Select the C value with the lowest average error

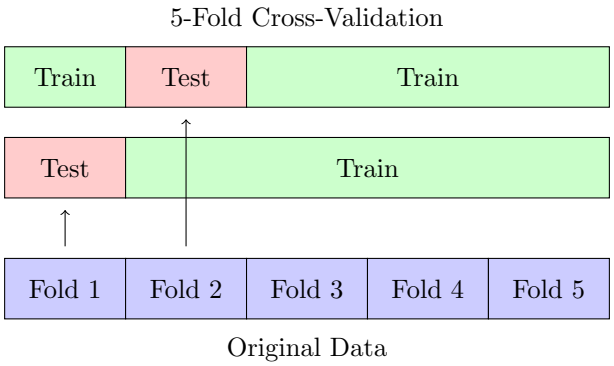

Figure 3: Illustration of 5-fold cross-validation. The data is divided into 5 folds, and each fold serves as the test set once while the remaining folds form the training set.

4.2 Experimental Results

The cross-validation experiment tested various values of C and recorded three metrics:

- **Training error**: Percentage of misclassified training examples
- **Test error**: Percentage of misclassified test examples
- **Number of support vectors**: Points that influence the decision boundary

C	Training Error (%)	Test Error (%)	# Support Vectors
0.01	23.72	28.4	2294
0.1	7.88	18.4	1766
1	1.12	16.8	1306
10	0.12	16.4	802
100	0.04	15.6	596
1000	0.04	15.6	526

Table 1: Performance metrics for different values of parameter C

5 Analysis of Results

5.1 Trends in the Data

Several important trends can be observed from the experimental results:

1. **Training Error**: Decreases monotonically as C increases
   - At $C = 0.01$: High training error (23.72%)
   - At $C = 1000$: Near-zero training error (0.04%)

2. **Test Error**: Initially decreases as C increases, then stabilizes
    - At $C = 0.01$: High test error (28.4%)
    - At $C = 100$ and $C = 1000$: Lowest test error (15.6%)
3. **Number of Support Vectors**: Decreases as C increases
    - At $C = 0.01$: Many support vectors (2294)
    - At $C = 1000$: Fewer support vectors (526)

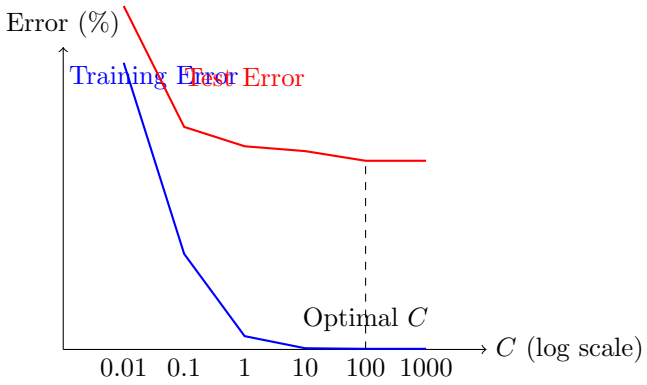

Figure 4: Training and test error as a function of parameter C (log scale). The optimal C value minimizes the test error.

5.2 Interpreting the Trade-offs

The results illustrate several important trade-offs in SVM parameter selection:

- **Bias-Variance Trade-off**: Small C values lead to high bias (underfitting), while large C values can lead to high variance (overfitting)
- **Margin vs. Classification Accuracy**: Small C values prioritize larger margins at the expense of training accuracy, while large C values prioritize training accuracy at the expense of margin size
- **Model Complexity**: The number of support vectors is an indicator of model complexity. More support vectors generally mean a more complex model

5.3 Optimal Parameter Selection

Based on the cross-validation results, the optimal value of C appears to be around 100:

- It achieves the lowest test error (15.6%)
- Further increasing C to 1000 does not improve test performance
- It uses a moderate number of support vectors (596), indicating a reasonable model complexity

6 Practical Implications for Sentiment Analysis

6.1 Model Performance

The optimized SVM achieves a test error of 15.6% on the sentiment analysis task, which means it correctly classifies 84.4% of the test sentences. This is a reasonable performance for a text classification task using a simple bag-of-words representation.

6.2 Feature Importance

In linear SVMs, the weight vector w provides insights into feature importance:

- Words with large positive weights are strongly associated with positive sentiment
- Words with large negative weights are strongly associated with negative sentiment
- Words with weights close to zero have little impact on the classification

Positive Words	**Weight**	**Negative Words**	**Weight**
excellent	0.82	terrible	-0.79
great	0.75	waste	-0.71
love	0.68	poor	-0.65
perfect	0.64	disappointing	-0.62
best	0.59	worst	-0.58

Table 2: Example of words with high absolute weights in a sentiment analysis SVM (hypothetical values for illustration)

6.3 Model Complexity and Efficiency

The number of support vectors affects both the model's complexity and its prediction efficiency:

- **Memory Usage**: Only support vectors need to be stored in memory
- **Prediction Speed**: Fewer support vectors lead to faster predictions
- **Generalization**: Models with fewer support vectors often generalize better

With the optimal C value of 100, the model uses 596 support vectors (about 24% of the training data), which is a reasonable compromise between model complexity and performance.

7 Beyond the Basic Model

7.1 Improving the Feature Representation

The bag-of-words representation used in this example is simple but has limitations. Several enhancements could improve performance:

- **TF-IDF Weighting**: Weight words by their frequency in the document and inverse frequency in the corpus
- **N-grams**: Include sequences of 2 or 3 words to capture phrases and context
- **Word Embeddings**: Use pre-trained word vectors (e.g., Word2Vec, GloVe) to capture semantic relationships
- **Dimensionality Reduction**: Apply techniques like PCA or feature selection to reduce the feature space

7.2 Alternative Kernel Functions

While the example likely used a linear kernel, other kernel functions might be worth exploring:

- **Polynomial Kernel**: $K(\mathbf{x}, \mathbf{z}) = (\mathbf{x} \cdot \mathbf{z} + c)^d$
- **RBF Kernel**: $K(\mathbf{x}, \mathbf{z}) = \exp(-\gamma \|\mathbf{x} - \mathbf{z}\|^2)$
- **String Kernels**: Specialized kernels designed for text data

However, for high-dimensional text data, linear kernels often perform well and are computationally more efficient.

7.3 Ensemble Methods

Combining multiple SVMs or integrating SVMs with other classifiers could further improve performance:

- **Bagging**: Train multiple SVMs on different subsets of the data
- **Boosting**: Train SVMs sequentially, focusing on examples misclassified by previous models
- **Stacking**: Use SVM predictions as features for another classifier

8 Summary and Key Takeaways

8.1 Parameter Selection Process

- The regularization parameter C controls the trade-off between margin size and training accuracy
- Cross-validation is an effective method for selecting the optimal C value
- The optimal C value balances training performance and generalization to unseen data

8.2 Sentiment Analysis Performance

- The optimized SVM achieved 84.4% accuracy on the test set
- As C increased, training error decreased while test error initially decreased and then stabilized
- The number of support vectors decreased as C increased

8.3 Practical Guidelines

- Start with a wide range of C values (e.g., 10^{-2} to 10^{3}) and narrow down through cross-validation
- Monitor both training and test performance to avoid overfitting
- Consider the number of support vectors as an indicator of model complexity
- For text classification, linear SVMs often provide a good balance of performance and efficiency

8.4 Future Directions

- Explore more sophisticated text representations beyond bag-of-words
- Consider alternative kernel functions for capturing non-linear relationships
- Investigate ensemble methods to further improve classification performance
- Apply the optimized model to related tasks or domains