Video Transcript - **The Impact of Hard and Soft Margins in SVMs: How the Soft Margin Parameter C Influences Classification Performance**

Hello everyone, today we are going to discuss the impact of hard margin and soft margin in Support Vector machines.

Learning objectives

These are the learning objectives. Mainly to be familiar with hard margin and soft margin concepts in SVM.

Plan for today

This is what we are going to discuss today.

What Is SVM

First of all let's have a quick introduction of what SVM is. As you all know, it is a supervised learning algorithm used in classification and regression tasks. Mostly in classification problems. It works by finding the hyperplane that best separates data into different classes while maximizing the margin between them.

As you can see, hyperplane is the decision boundary that separate these two different classes. These are the margins. Margin is the distance between hyperplane and the nearest data points in each class. These nearest data points are called support vectors.

Hard Margin

Basically, there are two types, hard margin and soft margin. Hard margin perfectly separate the data by hyperplane. No points lie inside the margin or no misclassifications are allowed. This is the equation for hard margin SVM. But in reality a perfect separation may not be possible. Because data often contains noise or outliers. So the strict margin can lead to the overfitting.

Soft Margin

Soft margin allows some misclassifications to handle non separable data.

Here, we have to think balancing of maximizing the margin while minimizing the misclassifications. Here, what SVM does is, it finds the maximum margin and it adds a penalty each time a point crosses the margin.

Slack Variable

Each data point has its own slack variable to find how much it violates the margin. It takes the distance of a misclassified point from its correct margin boundary.

To make a soft margin equation we add 2 more terms to this equation which is zeta and multiply that by a hyperparameter 'c'

C parameter

C is a hyperparameter that controls the trade-off between maximizing the margin and minimizing margin violations. It acts as a penalty for slack variables.

- When you have a higher C value, that strongly penalizes margin violations leading to smaller margin with fewer misclassifications.
- A lower C value, less penalizes margin violations, focus on wider margin allowing more misclassifications.

Visualization - SVM_HardMargin_SoftMargin.ipynb

Here, I have changed the C parameter, let's go for the code to see how this looks in practice. In the first figure C = 0.1 and there is a wider margin with data points inside the margin and there are two misclassified points. The second figure with C value = 1, the margin gets slightly smaller and a few data points inside the margin with one misclassified point in the hyperplane. In higher C values you can see, smaller margin with perfect separation of data points. No misclassified points.

Choosing the better model

From these three figures which one is correct. There is no correct answer. That depends on your dataset. If there are outliers in your dataset, smaller C value helps you balancing between maximizing margin and minimizing classification errors. If you have a cleaned dataset, larger C will perfectly separate the data. But careful about overfitting.

So you can justify best model using Margin error + classification error

Optimal C

Optimal C value depends on your data set (the noise level and data distribution) + the kernel you have used. You can perform Grid search optimization or Cross validation to find the optimal C value.

Case Study

This is a case study of Medical Diagnostics. I have taken this from a research paper. This research is based on finding the hyper parameters C and gamma value for different kernels to identify the best combination which maximizes the classification accuracy.

This has chosen different kernels for testing (RBF and sigmoid) kernels with different C values and gamma values + linear kernel with different C values. Because it doesn't come with gamma parameter. And finally perform Grid search CV to find the best combination.

Practical - Classification_OnlineRetailData.ipynb

This is a practical application. for this practical application, I utilized the UCI Online Retail Dataset to classify customers as either frequent or infrequent based on their Recency and Monetary values.

After performing preprocessing steps, I applied feature scaling to standardize the dataset, preparing it for the SVM model. Here, I have used the min max scaler to standardize the data.

To address the imbalance between frequent and infrequent customers, I used SMOTE (Synthetic Minority Oversampling Technique). This produced synthetic samples of the minority class, balancing the dataset and enhancing the model's ability to classify both classes accurately.

I used a Support Vector Machine (SVM) model to perform the classification.

To optimize the model's performance, I conducted a GridSearchCV to identify the best combination of hyperparameters. The grid search tested for multiple values for C, Kernel and Gamma parameters.

The best combination of hyperparameters were C=100,gamma=scale and rbf kernel. The model achieved an accuracy of 99.7% on the test data. The confusion matrix demonstrated that the model misclassified only one instance.

Effective preprocessing, SMote's handling of class imbalance, and hyperparameter adjustment via GridSearchCV taken together let the model perform quite well.

References

This is the end of my presentation. I have included references here. And all the codes are in the github repository along with detailed instructions on how to use it, how to run the code, you simply download the files, install the libraries, and follow the instructions on the readme file. You can try changing the CCC values to see how it affects the model's behaviour. You can try them your selves, and thank you so much.

Reference List

- Novakovic, J. and Veljovic, A., 2011, September. C-support vector classification: Selection of kernel and parameters in medical diagnosis. In *2011 IEEE 9th international symposium on intelligent systems and informatics* (pp. 465-470). IEEE.
- Cherkassky, V. and Ma, Y., 2004. Practical selection of SVM parameters and noise estimation for SVM regression. *Neural networks*, *17*(1), pp.113-126.
- Analytics Vidhya, 2024. *Analytics Vidhya*. Available at: https://www.analyticsvidhya.com/ [Accessed 7 Dec. 2024].
- **GitHub Repository**: https://github.com/samindika/SVM-Tutorial