Machine Learning - Support Vector Machine, Lab Assignment 2

Hani Hagash And Saly Al Masri

March 13, 2025

Question 1 and 2: Moving Clusters and Kernel Comparison

Question: Move the clusters around and change their sizes to make it easier or harder for the classifier to find a decent boundary. Pay attention to when the optimizer (minimize function) is not able to find a solution at all.

Experiment 1: Linear SVM with Different Spreads

Linear SVMs aim to find the maximum margin hyperplane that separates two classes. In this experiment, we vary the spread of clusters to observe how overlapping data affects linear separability.

Analysis of Results

With increasing spread, clusters overlap more, making it challenging for the linear SVM to find a separating hyperplane. For low spread values, the SVM achieves perfect separation. As the spread increases, the decision boundary becomes less accurate, leading to misclassifications.

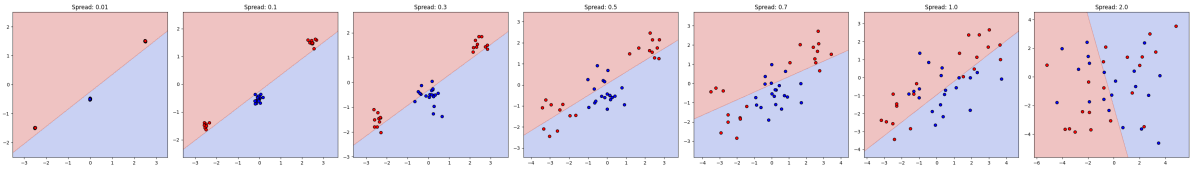


Figure 1: Effect of Cluster Spread on Linear SVM Decision Boundaries

Linear SVMs work well when the data is linearly separable. However, as the spread increases and clusters overlap, the SVM struggles to find an accurate boundary. This highlights the limitation of linear kernels in complex, non-linear data distributions. Linear SVMs are effective for well-separated clusters but fail when data overlap significantly. This motivates the need for non-linear kernels.

Experiment 2: 3D Feature Space and Non-Linear Separation

Mapping data to a higher-dimensional space can make it linearly separable. Here, we use the transformation $z = x^2 + y^2$ to observe how a 3D feature space affects separation.

Analysis of Results

In 2D space, the classes are not linearly separable. However, in 3D space, the transformation creates a non-linear boundary that perfectly separates the classes.

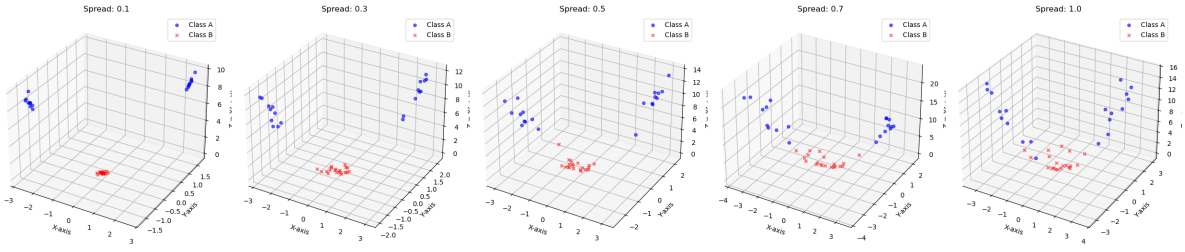


Figure 2: Non-linear Separation in 3D Feature Space

This experiment illustrates the power of mapping non-linear data to higher dimensions. It also demonstrates the effectiveness of the kernel trick in SVMs, which implicitly maps data to higher dimensions without computational cost. Mapping to higher dimensions enables linear separation of complex patterns, highlighting the power of non-linear kernels.

Experiment 3: RBF Kernel with Adversarial Noise

RBF kernel creates complex, flexible boundaries by emphasizing local data points. Adversarial noise challenges the model's generalization ability by introducing conflicting labels near the boundary.

Analysis of Results

The RBF kernel overfits to noise, creating wiggly boundaries that conform to noise points. As noise increases, decision boundaries become irregular, and overfitting worsens.

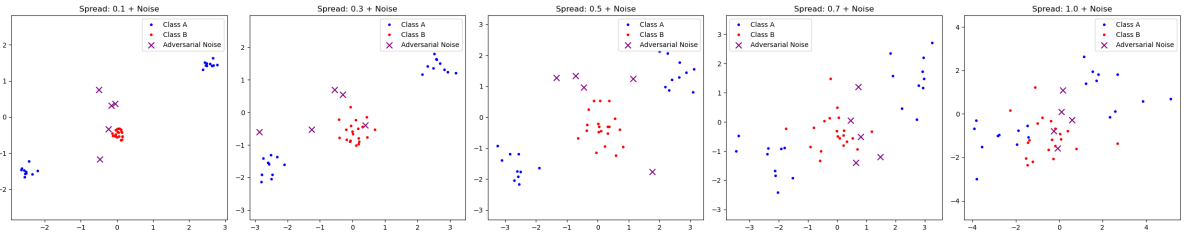


Figure 3: Effect of Adversarial Noise on RBF Kernel

The RBF kernel is highly sensitive to noise due to its locality. Regularization with slack parameter C can help control overfitting, balancing variance and bias. RBF kernel's flexibility comes with a risk of overfitting. Proper tuning of C is crucial for generalization.

Experiment 4: Multiclass SVM with RBF Kernel

One-vs-One SVM trains binary classifiers for each class pair, with majority voting for final classification. RBF kernel provides non-linear boundaries for each classifier.

Analysis of Results

With three classes, decision boundaries are complex and intersect in irregular patterns. As cluster overlap increases, conflicting predictions occur, leading to noisy decision regions.

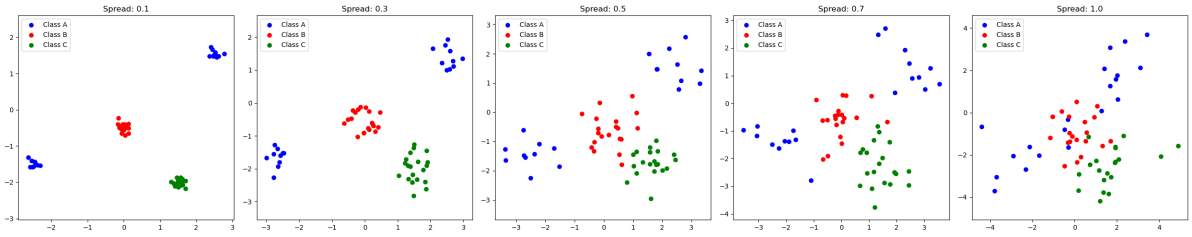


Figure 4: Multiclass Classification with RBF Kernel

The complexity of decision boundaries increases with more classes. Conflicting regions result from the intersection of one-vs-one classifiers. Adjusting σ and C is essential for controlling complexity. RBF Kernel is powerful for multiclass problems but requires careful tuning to balance complexity and generalization.

Question 3: Comparison of Kernels

Question: Compare the results when using different kernels.

1 Results: Question 2 and 3

The implementation of the non-linear kernels significantly impacted the classification performance, allowing the support vector machine to handle complex decision boundaries. The polynomial kernel, defined as $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T\mathbf{y} + 1)^p$, introduces non-linearity through the exponent p. The results demonstrated that for lower values of p, such as $p = 2$, the decision boundary remained relatively smooth, forming quadratic shapes such as ellipses and parabolas. As the value of p increased, the decision boundary became increasingly complex, adapting more closely to the data distribution but at the risk of capturing noise, thereby leading to overfitting.

Similarly, the radial basis function (RBF) kernel, defined as $K(\mathbf{x}, \mathbf{y}) = e^{-\frac{||\mathbf{x}-\mathbf{y}||^2}{2\sigma^2}}$, exhibited strong adaptability to the data. The parameter σ controlled the extent of influence exerted by each data point. When σ was large,

the decision boundary became smoother, leading to a high-bias, low-variance model that generalized well but lacked flexibility in distinguishing finer structures. Conversely, a small σ resulted in highly flexible boundaries, effectively separating even closely spaced data points. However, this introduced high variance, making the model sensitive to minor variations and noise.

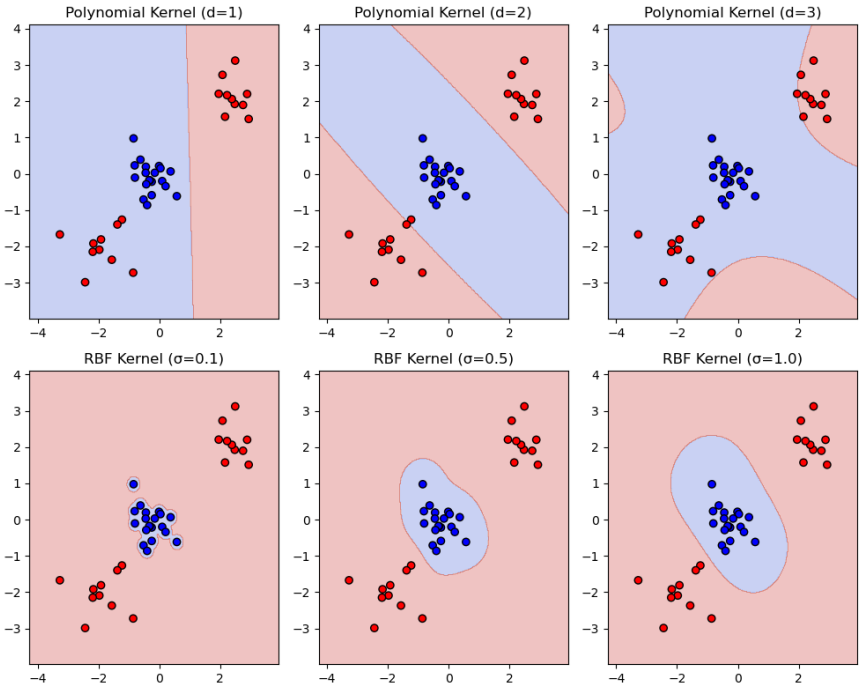
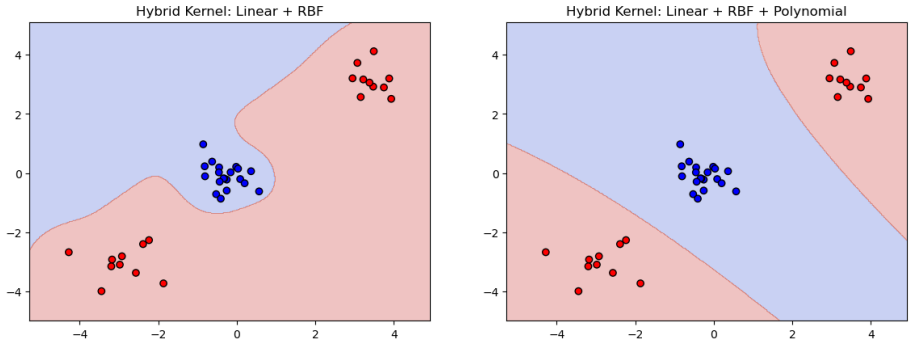


Figure 5: Enter Caption

The observed results align with the bias-variance trade-off, where an increase in model complexity reduces bias but raises variance. The polynomial kernel with a high degree and the RBF kernel with a small σ exhibited low bias but high variance, leading to overfitting in highly non-linear data. Conversely, a low-degree polynomial or a large σ produced a simpler, high-bias model with lower variance, which was more robust against noise but struggled to capture intricate patterns. The optimal choice of kernel parameters required a balance between these two extremes to ensure a model that generalizes well to new data while maintaining an effective decision boundary. These results highlight the importance of selecting appropriate kernel parameters based on the complexity and noise level of the data.

1.1 Hybrid Kernels

Combining multiple kernels can enhance classification performance by leveraging their individual strengths. Using a linear and RBF kernel provides a balance between generalization and local adaptability, making it suitable for data that is mostly linearly separable but has some local variations. The combination of a linear and polynomial kernel introduces curved decision boundaries while maintaining a simpler structure, making it effective for datasets with mild non-linearity. The RBF and polynomial kernel mix creates a highly flexible model capable of capturing complex patterns, though it requires careful tuning to prevent overfitting. Using all three kernels together provides the highest adaptability but increases computational complexity and risks overfitting. Selecting an appropriate hybrid kernel depends on the dataset's complexity, ensuring a trade-off between model flexibility and interpretability.



(a) Linear + RBF (b) Linear + RBF + Polynomial

Figure 6: Comparison of Hybrid Kernels

1.2 Manhattan Distance in RBF Kernel

Using the Manhattan distance instead of Euclidean distance in the RBF kernel alters how similarity between points is measured, resulting in decision boundaries that tend to be more axis-aligned. Unlike Euclidean distance, which computes straight-line distances and produces smooth, circular boundaries, the Manhattan distance sums absolute differences across dimensions, making it more robust to outliers and feature scaling variations. This characteristic makes it particularly effective in high-dimensional or sparse data scenarios, such as text classification or financial modeling, where features exhibit abrupt changes. The choice between Euclidean and Manhattan depends on the data structure, with Euclidean being preferable for smooth, continuous relationships and Manhattan excelling in cases where sharp distinctions between features exist.

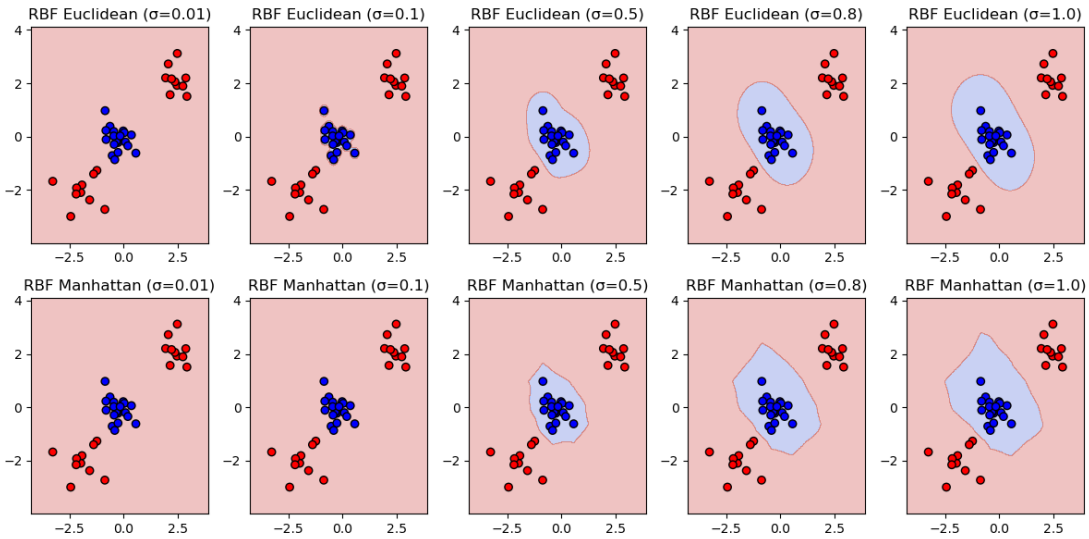


Figure 7: Decision boundaries using Manhattan distance in the RBF kernel

Question 4: Effect of Slack Parameter C

Question: Explore the role of the slack parameter C. What happens for very large/small values?

Answer: The slack parameter C controls the trade-off between maximizing margin and minimizing classification errors:

- **Small C:** Allows more slack, resulting in wider margins but potentially more misclassifications.
- **Large C:** Strictly penalizes misclassifications, leading to narrower margins and potentially overfitting.

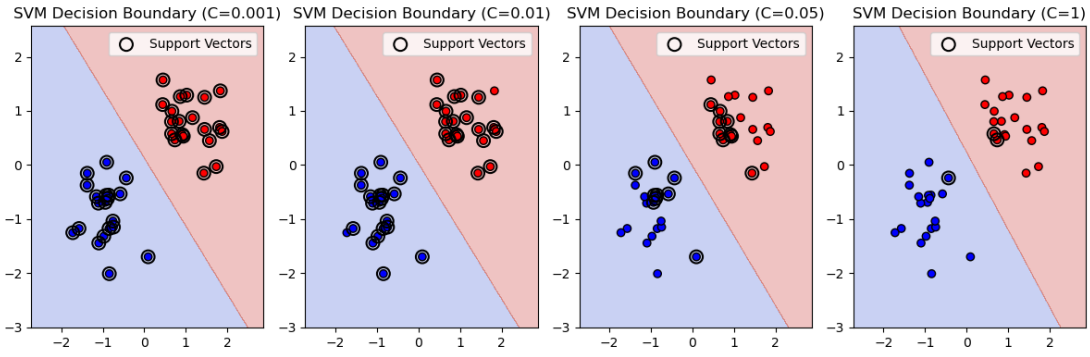


Figure 8: Effect of varying C on decision boundaries

Conclusion

This assignment demonstrates the importance of kernel choice and hyperparameter tuning in SVMs. By adjusting p, σ, and C, we can significantly influence decision boundaries and model performance.