Support Vector Machines (SVMs): How Kernel Choice Shapes the Decision Boundary

Student name: Avusali Mani Harshith
Student ID: 24005856
github link: https://github.com/Maniharshith18/SVM-Kernel-Tutorial.git

1. Introduction

Support Vector Machines (SVMs) are a powerful supervised learning algorithm used for classification. SVMs aim to find a hyperplane that maximizes the margin between classes, improving generalization on unseen data. Real-world data is often non-linear, and a linear hyperplane may fail to separate classes effectively.

Kernel functions allow SVMs to map data into higher-dimensional spaces where linear separation becomes possible. This tutorial explores how **linear, polynomial, and RBF (Gaussian) kernels** affect SVM decision boundaries. Using synthetic datasets, we visualize kernel effects and examine how parameters like **polynomial degree** and **RBF gamma** influence flexibility and overfitting.

Note on SVM Theory: The linear kernel produces a wide-margin hyperplane capturing the global trend, whereas polynomial and RBF kernels adjust local boundaries to accommodate non-linear patterns. Support vectors are crucial in defining these boundaries and illustrate the model's learning focus.

2. Datasets

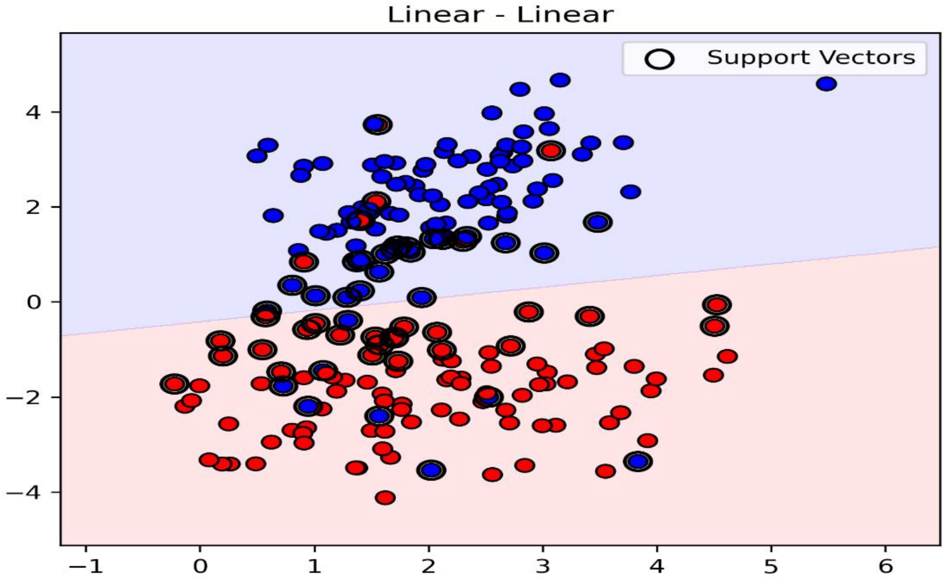
We use three 2D datasets:

1. **Linear dataset** – linearly separable points with slight noise.
2. **Moons dataset** – two interleaving half-moon shapes, moderately non-linear.
3. **Circles dataset** – concentric circles, highly non-linear.

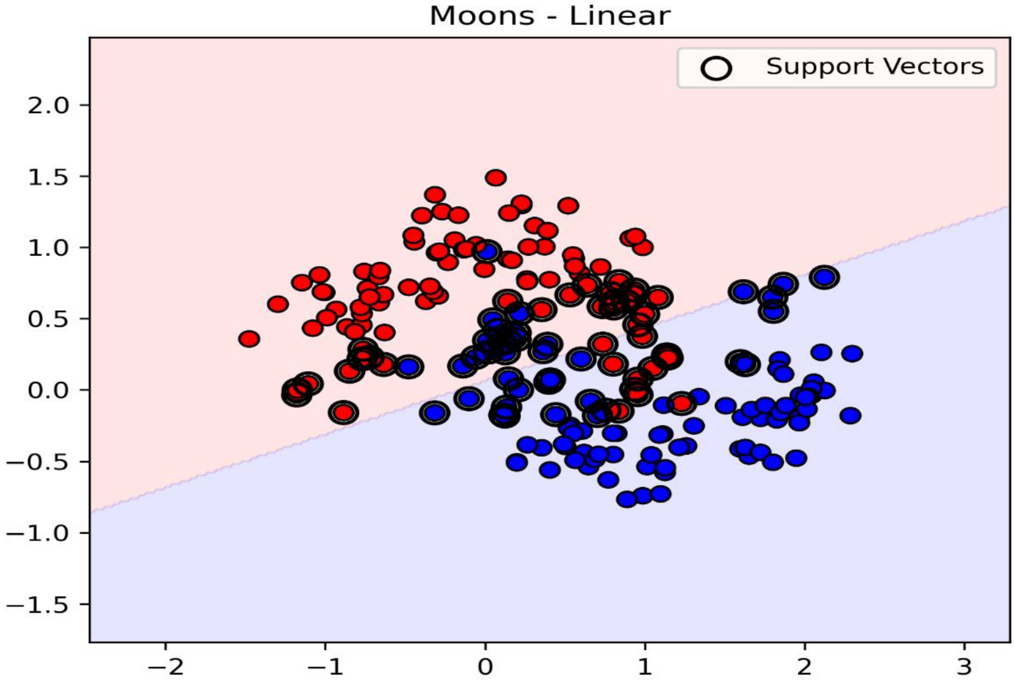
These datasets provide clear visualizations of decision boundaries.

Figures 1–3: Scatterplots of datasets

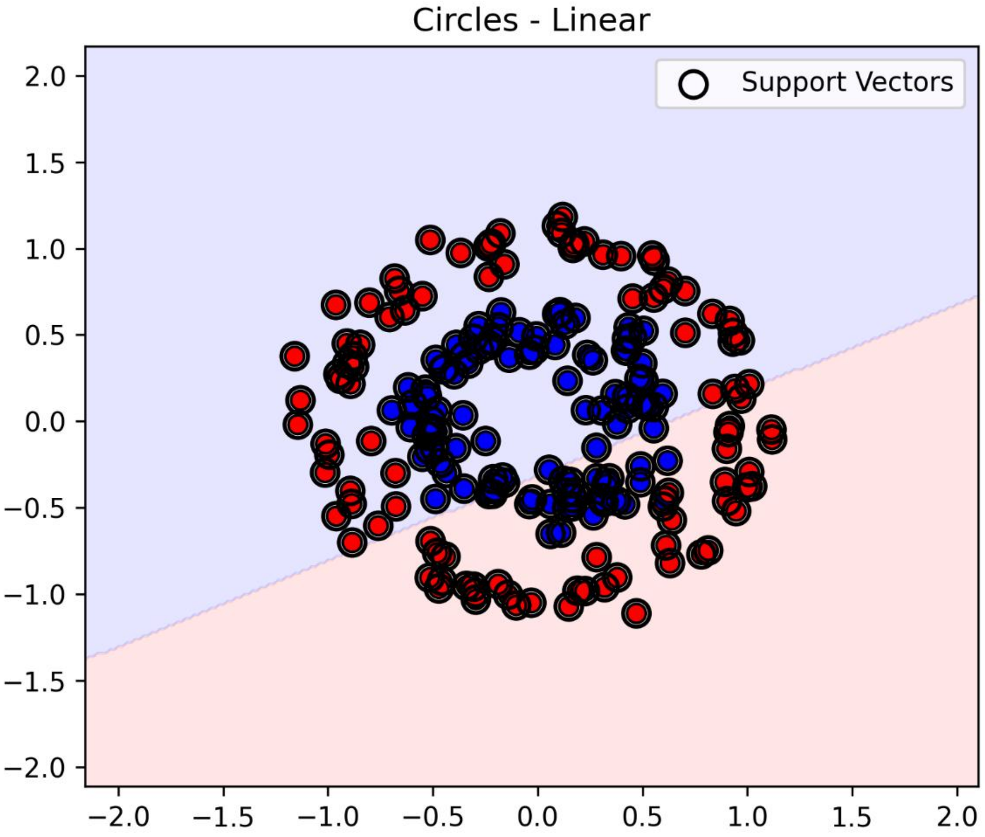
- *Figure 1: Linear dataset scatterplot*



- *Figure 2: Moons dataset scatterplot*



- *Figure 3: Circles dataset scatterplot*



---

3. Kernel Functions

3.1 Linear Kernel

- Computes a standard dot product between features.
- Suitable for linearly separable data.
- Fast and interpretable.
- Limitation: cannot model non-linear patterns.

3.2 Polynomial Kernel

- Maps data into higher-dimensional polynomial space.
- Degree controls flexibility: higher degree = more complex boundaries.
- Risk of overfitting if degree is too high.

3.3 RBF (Gaussian) Kernel

- Maps data into infinite-dimensional space using an exponential function.
- Gamma controls the influence of a single training point.
- Highly flexible, works well for most non-linear problems.
- Sensitive to parameter tuning.

4. Training and Visualisation

We trained SVM classifiers on all three datasets using:

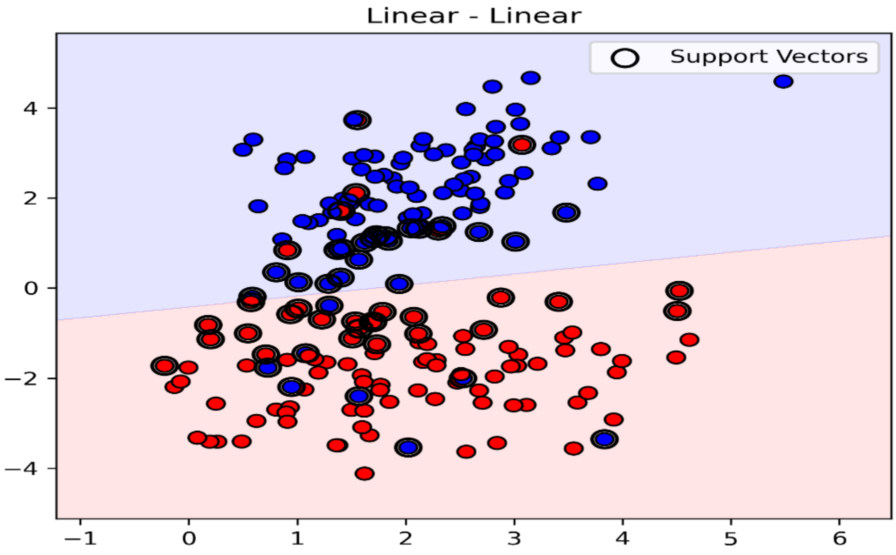
- Linear
- Polynomial (degree 3)
- RBF (gamma = 1)

Table 1: Effect of SVM Parameters on Model Behavior

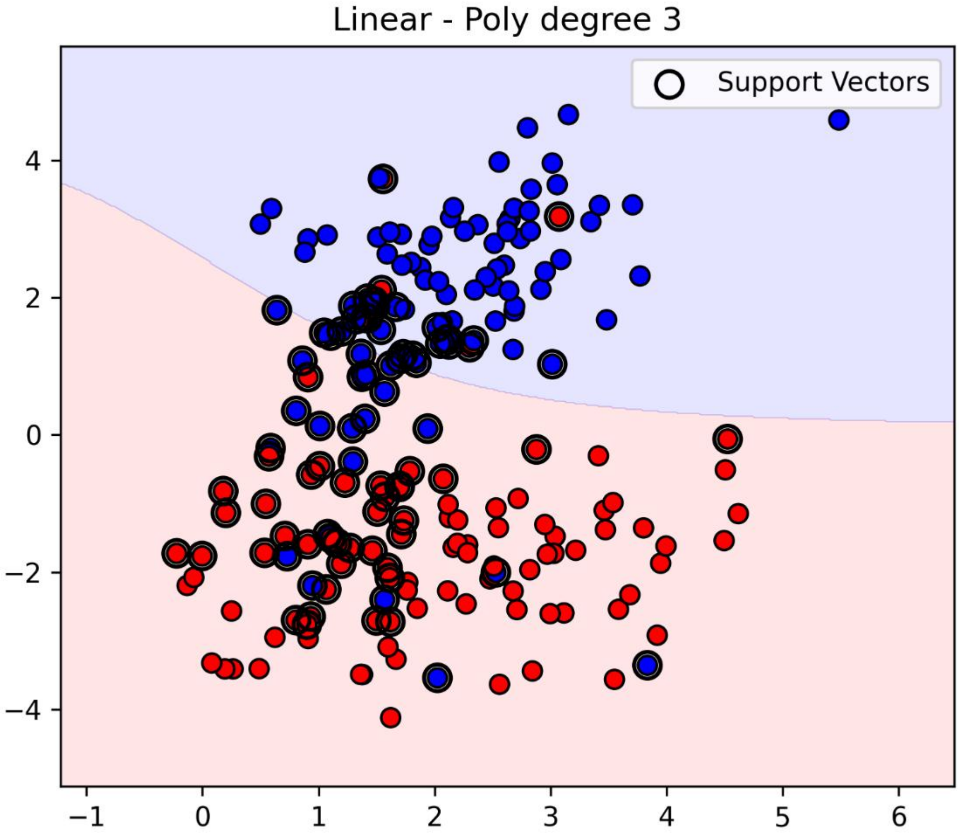
Parameter	Effect on model
C	Higher C → less regularization → tighter fit to training data
Degree (poly)	Higher degree → more complex decision boundary → risk of overfitting
Gamma (RBF)	Higher gamma → boundary fits closer to points → risk of overfitting

4.1 Linear Dataset

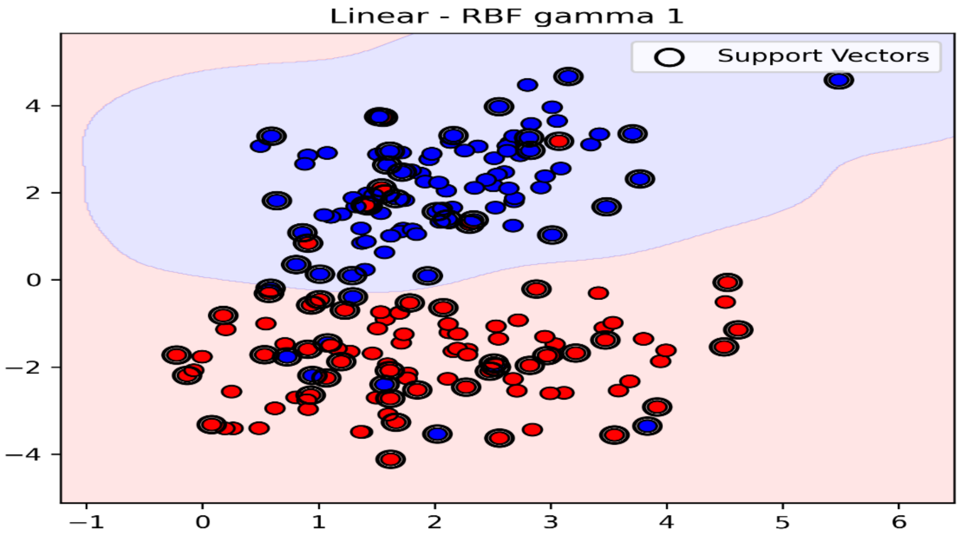
- *Figure 5: Linear dataset – Linear kernel*



- *Figure 6: Linear dataset – Polynomial (degree 3) kernel*



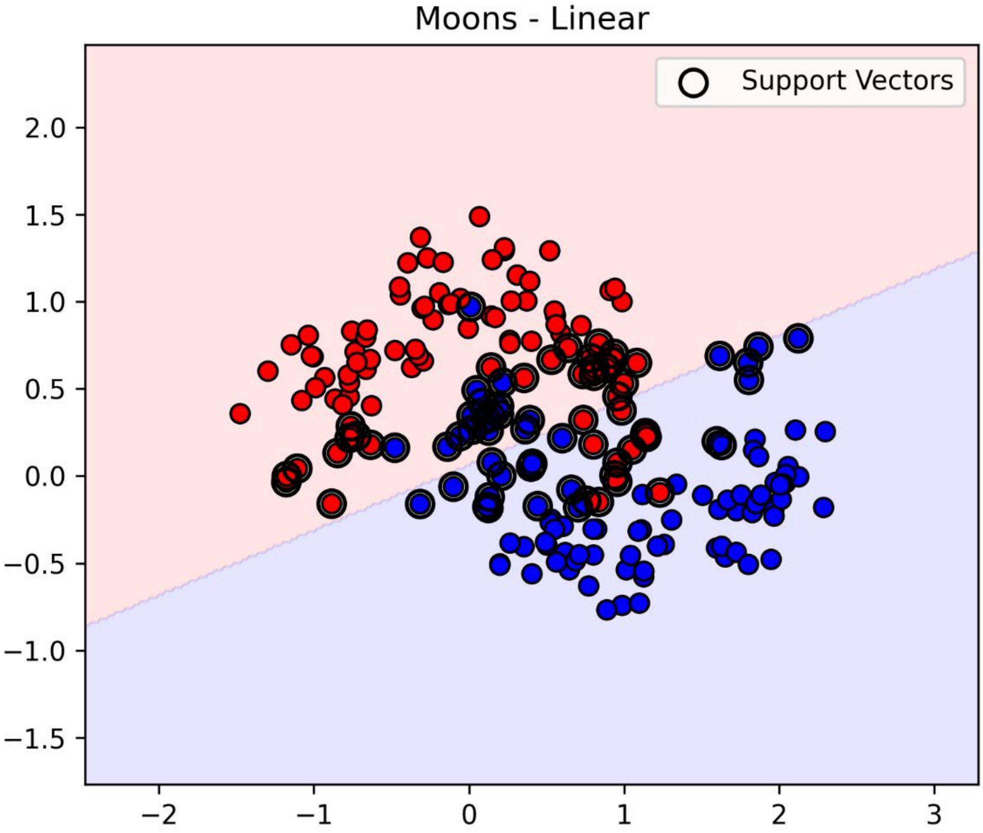
- *Figure 7: Linear dataset – RBF (gamma=1) kernel*



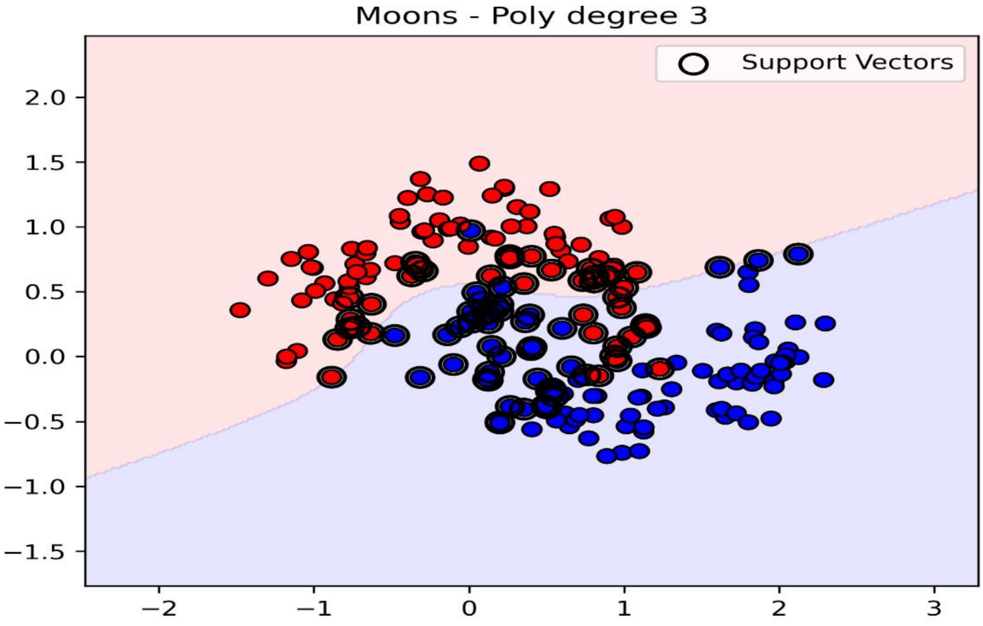
Observations: Linear kernel performs best; polynomial slightly overfits; RBF fits perfectly but is unnecessary.

4.2 Moons Dataset

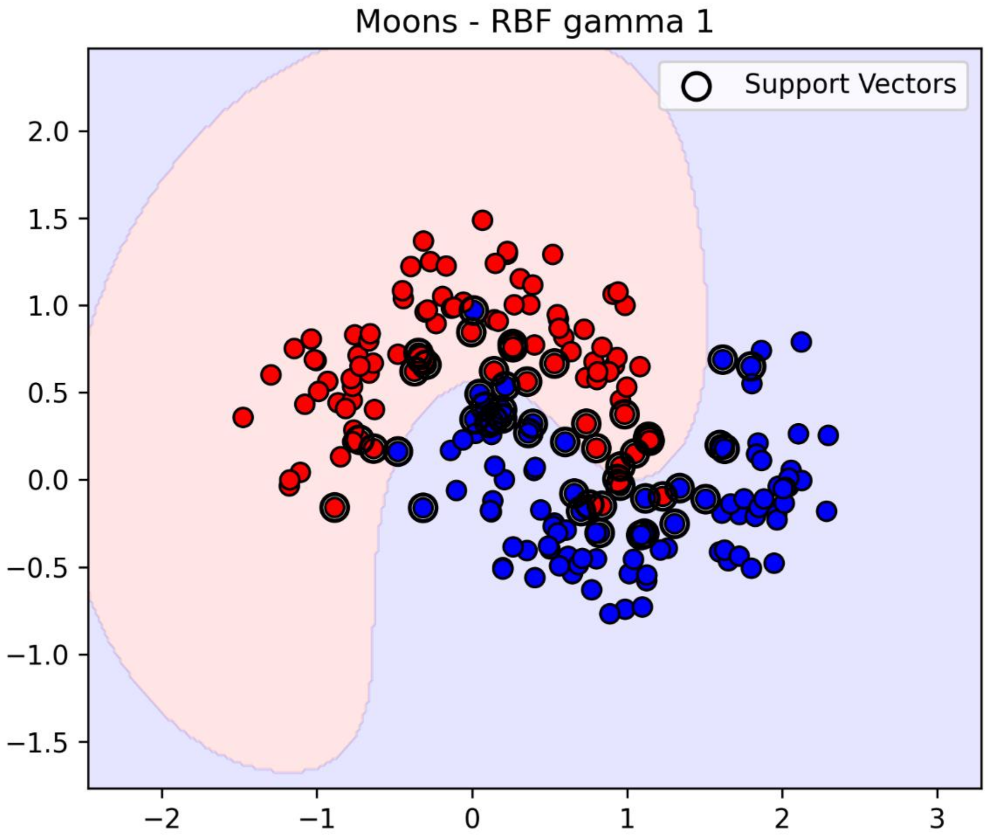
- *Figure 8: Moons dataset – Linear kernel*



- *Figure 9: Moons dataset – Polynomial (degree 3) kernel*



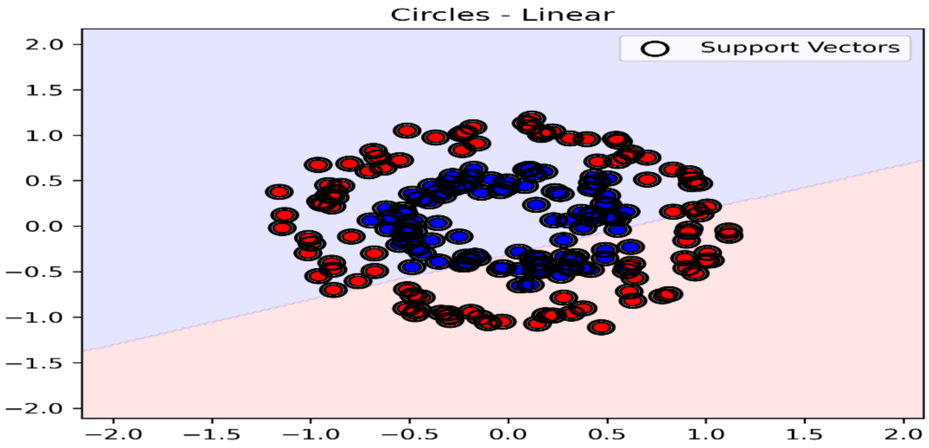
- *Figure 10: Moons dataset – RBF (gamma=1) kernel*



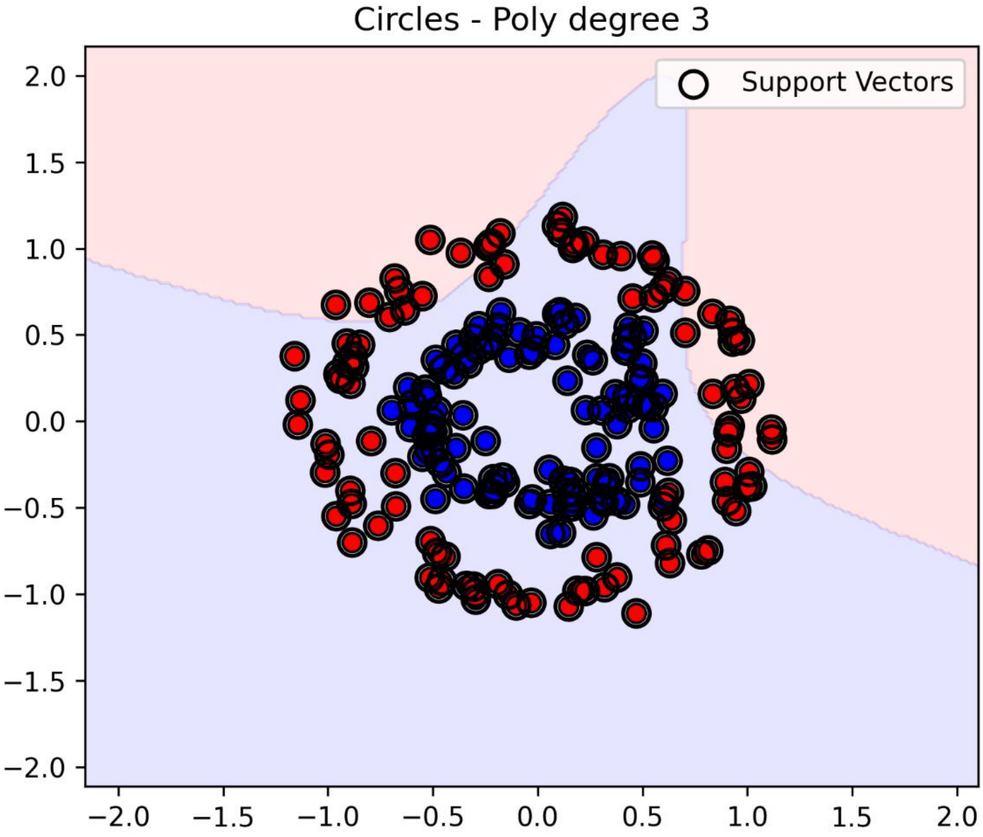
Observations: Linear kernel fails; polynomial kernel captures moderate curvature; RBF kernel fits non-linear shapes accurately.

4.3 Circles Dataset

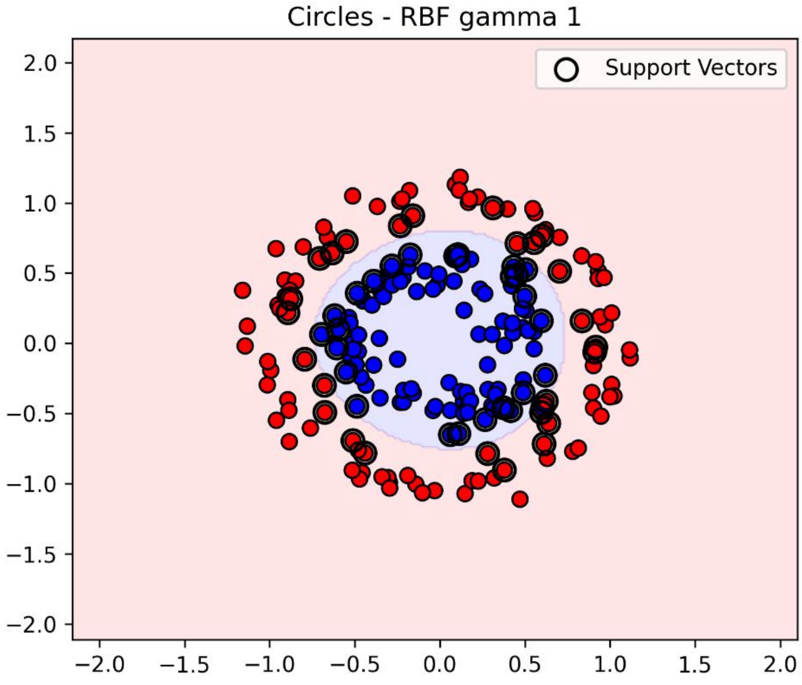
- *Figure 11: Circles dataset – Linear kernel*



- *Figure 12: Circles dataset – Polynomial (degree 3) kernel*



- *Figure 13: Circles dataset – RBF (gamma=1) kernel*



Observations: *Linear kernel fails completely; polynomial kernel partially separates; RBF kernel successfully separates concentric classes.*

---

5. Observations

- **Linear Kernel: Fast and interpretable; works for linear data; fails on non-linear datasets.**
- **Polynomial Kernel: Flexible; degree controls complexity; moderate non-linear patterns are captured; higher degrees risk overfitting.**
- **RBF Kernel: Highly flexible; gamma controls smoothness; excellent for complex non-linear patterns; sensitive to overfitting.**

Table 2: Kernel Comparison

Kernel	**Strengths**	**Weaknesses**	**Recommended Use**
Linear	Fast, interpretable	Cannot model curves	Linearly separable, high-dimensional data
Polynomial	Flexible, moderate non-linear patterns	Sensitive to degree	Moderate non-linear datasets
RBF	Highly flexible, handles complex patterns	Sensitive to gamma	Default choice for complex non-linear problems

---

6. Accessibility

- All figures include descriptive alt-text
- Colorblind-friendly palettes (e.g., Seaborn colorblind theme)
- Proper heading hierarchy for screen readers
- Mathematical expressions written in LaTeX (screen-reader compatible)
- High contrast between text and background
- Captions summarise the key insight of each figure
- Code blocks use accessible monospace fonts

7. Conclusion

Kernel choice dramatically affects SVM decision boundaries:

1. **Linear Kernel** – simple, fast, interpretable; best for linear data.
2. **Polynomial Kernel** – captures moderate non-linear patterns; higher degree = more flexible, risk of overfitting.

3. **RBF Kernel** – highly flexible; gamma controls smoothness and overfitting.

Tuning parameters (degree, gamma, C) is essential for balancing **bias and variance** to achieve optimal SVM performance.

---

8. Advanced SVM Theory

8.1 Soft-Margin SVM (Mathematical Formulation)

Real-world datasets are noisy and often overlap. Soft-margin SVM balances margin maximization with classification error using slack variables ξ_i.

Primal Formulation:

$$\min_{\mathbf{w},b,\xi} \frac{1}{2} \parallel \mathbf{w} \parallel^2 + C \sum_{i=1}^{n} \xi_i$$

Subject to:

$$y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \xi_i \geq 0$$

- Large **C** → stricter classification, smaller margin (risk of overfitting)
- Small **C** → wider margin, allows errors (less overfitting)

8.2 Dual Formulation and the Kernel Trick

In the dual problem, all computations involve **dot products**:

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \, \alpha_j y_i y_j K(x_i, x_j)$$

Subject to:

$$0 \leq \alpha_i \leq C, \sum_i \alpha_i \, y_i = 0$$

The kernel replaces the dot product:

$$K(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$$

9. Additional Kernel Functions

9.1 Sigmoid Kernel

$$K(x, x') = \tanh\left(\alpha\, x^\top x' + c\right)$$

- behaves like a neural network activation
- works for some text and speech tasks

9.2 Laplacian Kernel

$$K(x, x') = \exp\left(-\frac{\| x - x' \|_1}{\sigma}\right)$$

- less sensitive to outliers
- good for sparse or noisy data

9.3 Example of a Custom Kernel

$$K(x, x') = \cos^2(x^\top x')$$

You can define custom kernels when domain-specific similarity measures improve performance.

10. Computational Complexity of SVMs

10.1 Training Time

Kernel SVMs scale poorly because they rely on solving a quadratic optimization problem.

$$O(n^3)$$

This makes kernel SVMs impractical for very large datasets.

10.2 Memory Cost

$$O(n^2)$$

They must compute and store an $n \times n$Gram matrix.

10.3 Practical Implications

SVM Type	Complexity	Best Use Case
Linear	O(nd)	Large, high-dimensional datasets
Kernel	O(n^3)	Small/medium datasets with nonlinear structure

11. Visualising Kernel-Induced Feature Spaces

Kernels implicitly lift data into higher-dimensional spaces.
For example, a polynomial kernel of degree 2:

$$\phi(x_1, x_2) = (x_1^2,\ x_2^2,\ \sqrt{2}x_1x_2,\ \sqrt{2}x_1,\ \sqrt{2}x_2,\ 1)$$

A 2D dataset becomes **6-dimensional**.
Even a simple 3D plot of the first three components shows how linear separation becomes possible.

12. References

1. Cortes, C., & Vapnik, V. (1995). *Support-vector networks*. Machine Learning, 20, 273–297.
2. scikit-learn documentation: https://scikit-learn.org/stable/modules/svm.html
3. Raschka, S. (2018). *Python Machine Learning*. Packt Publishing.
4. Distill.pub. *Visualizing the Kernel Trick*. https://distill.pub/2016/kernel
5. Medium. *Understanding the SVM Kernel Trick*. https://medium.com