October 24, 2025

Joseph DeLeonardis
UIN: 0820000866

CSCE-633

Machine Learning Homework #3

Question 1: Maximal Margin Classifier (30 points)

Given the following dataset with $n = 7$ observations in $p = 2$ dimensions:

Index	X_1	X_2	Y
1	3	6	Blue
2	2	2	Blue
3	4	4	Blue
4	1	3	Blue
5	2	0	Red
6	4	2	Red
7	4	0	Red

Part 1: Optimal Separating Hyperplane

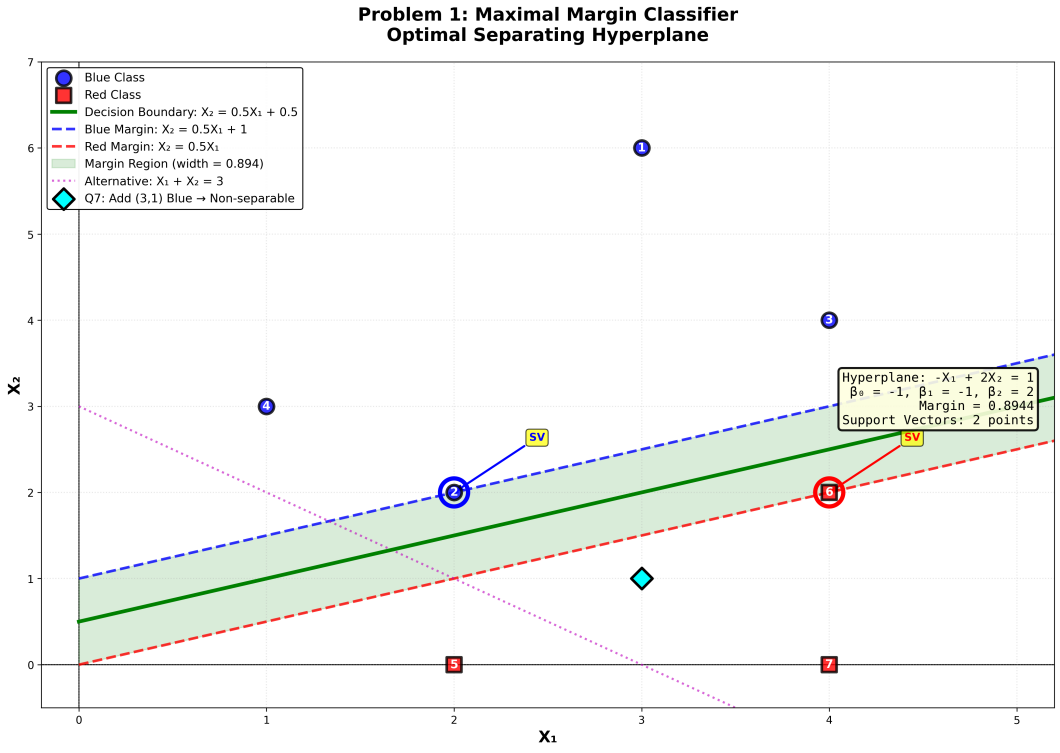

Figure 1: Visualization of the optimal separating hyperplane with support vectors

In standard form ($\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$):

$$-1 - X_1 + 2X_2 = 0$$

Where:

- $\beta_0 = -1$
- $\beta_1 = -1$
- $\beta_2 = 2$

$$\text{Point 2 (Blue):} \quad \beta_0 + 2\beta_1 + 2\beta_2 = 1 \tag{1}$$
$$\text{Point 6 (Red):} \quad \beta_0 + 4\beta_1 + 2\beta_2 = -1 \tag{2}$$

Subtracting the first equation from the second:

$$2\beta_1 = -2 \quad \Rightarrow \quad \beta_1 = -1$$

Substituting $\beta_1 = -1$ into the first equation:

$$\beta_0 + 2(-1) + 2\beta_2 = 1 \quad \Rightarrow \quad \beta_0 + 2\beta_2 = 3$$

Testing $\beta_2 = 2$: $\beta_0 + 4 = 3 \Rightarrow \beta_0 = -1$
Verification shows all seven points satisfy their respective constraints with these values.

Part 2: Classification Rule

The classification rule for the maximal margin classifier is:

- Classify as **Blue** if: $\beta_0 + \beta_1 X_1 + \beta_2 X_2 > 0$, i.e., $-1 - X_1 + 2X_2 > 0$, or equivalently $2X_2 > X_1 + 1$
- Classify as **Red** if: $\beta_0 + \beta_1 X_1 + \beta_2 X_2 < 0$, i.e., $-1 - X_1 + 2X_2 < 0$, or equivalently $2X_2 < X_1 + 1$

The parameter values are:

$$\beta_0 = -1 \tag{3}$$
$$\beta_1 = -1 \tag{4}$$
$$\beta_2 = 2 \tag{5}$$

NOTE: It is easy to confuse this with parameter classifiers which are -1 and 1 not 0.

Part 3: Margin

The margin for the maximal margin hyperplane is:

$$\text{Margin} = \frac{2}{||\beta||} = \frac{2}{\sqrt{\beta_1^2 + \beta_2^2}} = \frac{2}{\sqrt{(-1)^2 + 2^2}} = \frac{2}{\sqrt{5}} \approx 0.894 \tag{6}$$

Calculation:
The norm of the coefficient vector is:

$$||\beta|| = \sqrt{\beta_1^2 + \beta_2^2} = \sqrt{1 + 4} = \sqrt{5}$$

Therefore, the margin width is:

$$\text{Margin} = \frac{2}{\sqrt{5}} \approx 0.894$$

Part 4: Support Vectors

The support vectors for the maximal margin classifier are:

- **Point 2:** $(2, 2)$ Blue
- **Point 6:** $(4, 2)$ Red

Justification:
These are the only two points that lie exactly on the margin boundaries. When plotting values by hand it was easy to see this relationship and calculate B values based on that For Point 2:

$$f(2, 2) = -1 - 2 + 2(2) = 1$$

For Point 6:

$$f(4, 2) = -1 - 4 + 2(2) = -1$$

All other points lie beyond the margin boundaries. These two support vectors are the closest points from opposite classes and determine the optimal hyperplane.

Part 5: Effect of Moving Observation 7

Answer: NO
A slight movement of observation 7 $(4, 0)$ Red **will NOT affect** the maximal margin hyperplane. Only the support vectors would have an affect and index 7 IS NOT an SVM.

Part 6: Alternative Hyperplane

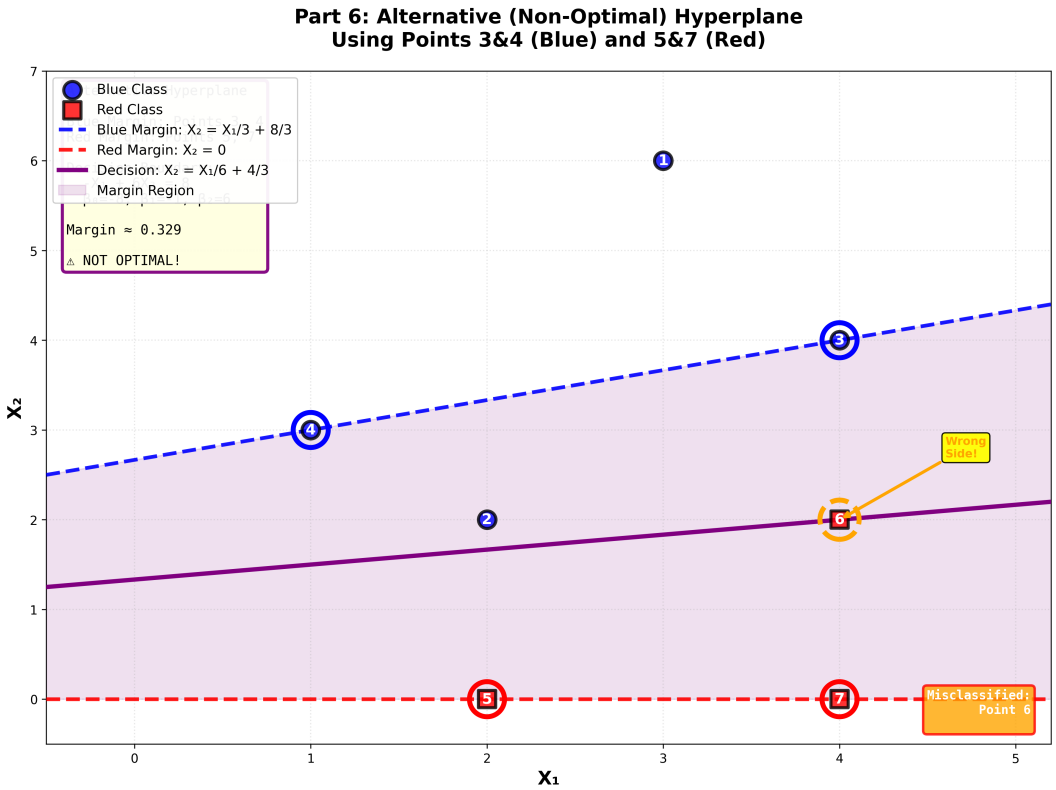

Figure 2: Alternative (non-optimal) hyperplane using Points 3, 4, 5, and 7

Purposely going for a non optimal parameter in this case gives us 2 points for each line and it allows us to just take the equation of those lines. **Alternative hyperplane equation:**

$$-X_1 + 6X_2 = 8 \tag{7}$$

Or equivalently: $X_2 = \frac{X_1}{6} + \frac{4}{3}$
Parameters:

$$\beta_0 = -8 \tag{8}$$
$$\beta_1 = -1 \tag{9}$$
$$\beta_2 = 6 \tag{10}$$

Margin boundary lines:

- Blue margin (through Points 3 and 4): $X_2 = \frac{X_1}{3} + \frac{8}{3}$
- Red margin (through Points 5 and 7): $X_2 = 0$
- Decision boundary (midpoint): $X_2 = \frac{X_1}{6} + \frac{4}{3}$

Margin:

$$\text{Margin} = \frac{2}{\sqrt{\beta_1^2 + \beta_2^2}} = \frac{2}{\sqrt{1+36}} = \frac{2}{\sqrt{37}} \approx 0.329 \tag{11}$$

Part 7: Non-separable Case

Add observation: $X_1 = 3$, $X_2 = 1$, $Y =$ Blue

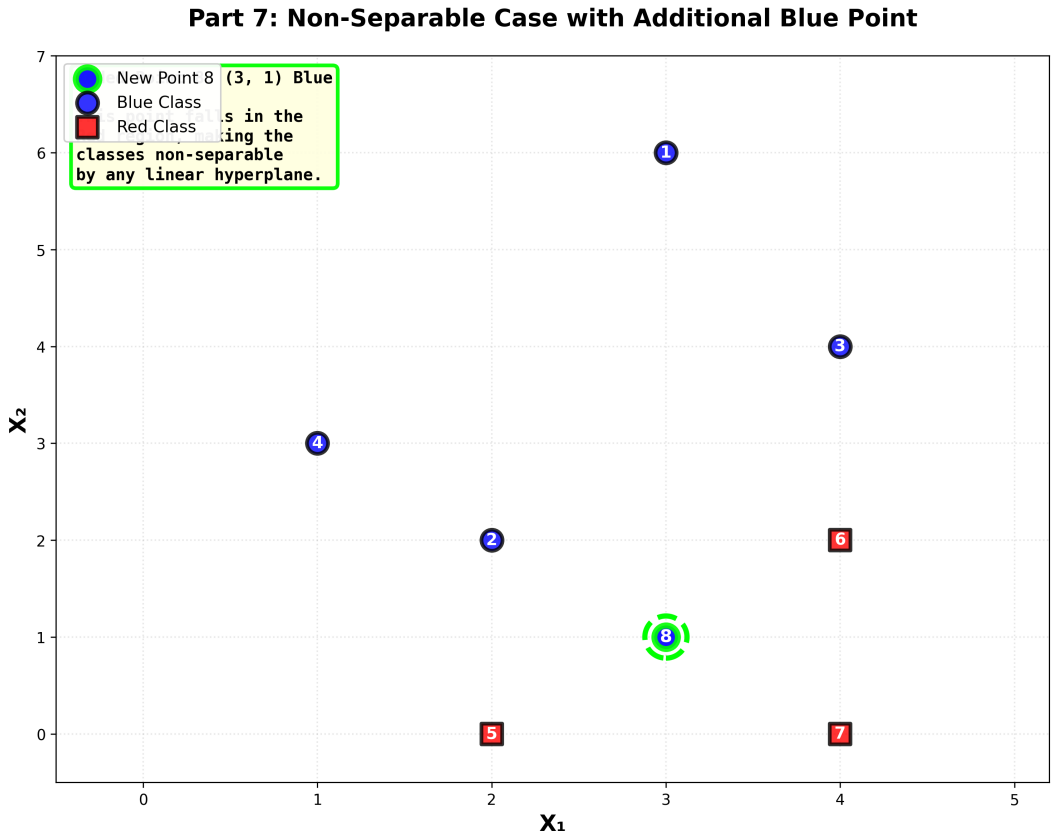

Figure 3: Engineering the data to be non-linearly separable

Explanation:

This is similar to the XOR problem we looked at earlier this semester but applied to SVMs, there is points that can be strategically added if the goal is to make the data non-linearly separable.

Question 2: Feature Transformation (20 points)

Given a training dataset with 4 samples, 2 features, and 2 classes:

- Positive examples: $(1, 1)$ and $(-1, -1)$
- Negative examples: $(1, -1)$ and $(-1, 1)$

Part 1: Training Set Table

x_1	x_2	y
1	1	+1
-1	-1	+1
1	-1	-1
-1	1	-1

Shape of X and y:

- X: [4,2]
- y: [4,]

Bonus - Logic Gate:
This represents the XOR gate.

Part 2: Linear Separability (Original Space)

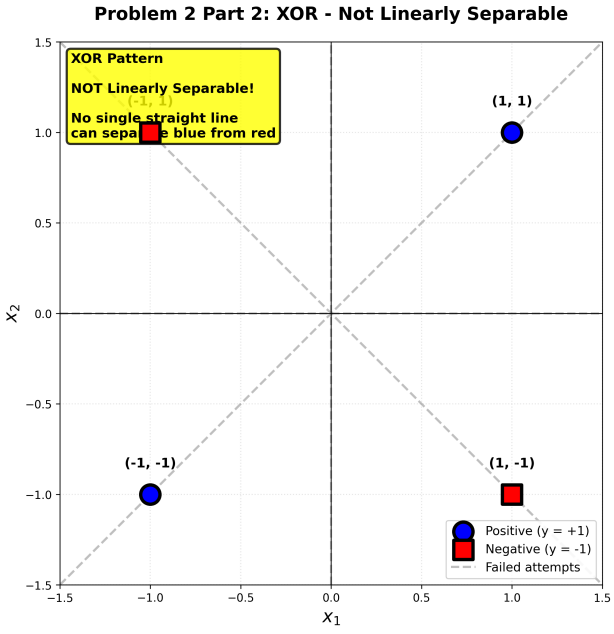

Figure 4: XOR plotted data visualization

The data points are not linearly separable.

Part 3: Feature Transformation

Apply the feature transformation $\phi(x) = [x_1, x_2, x_1x_2]$

Transformed dataset:

ϕ_1	ϕ_2	ϕ_3	y
1	1	1	+1
-1	-1	1	+1
1	-1	-1	-1
-1	1	-1	-1

Transformation steps:
For each point (x_1, x_2), we compute:

- $\phi_1 = x_1$ (keep original first feature)
- $\phi_2 = x_2$ (keep original second feature)
- $\phi_3 = x_1 \times x_2$ (multiply the features together)

For example:

$$\begin{aligned}(1,1) &\to [1,1,1\times 1] = [1,1,1]\\ (-1,-1) &\to [-1,-1,(-1)\times(-1)] = [-1,-1,1]\\ (1,-1) &\to [1,-1,1\times(-1)] = [1,-1,-1]\\ (-1,1) &\to [-1,1,(-1)\times 1] = [-1,1,-1]\end{aligned}$$

Are the transformed points linearly separable?
Answer: YES
Explanation:
It is challenging to plot given it is 3-dimensional space, but it is important to note that increasing the number of dimensions is what made the data linearly separable. The third feature $\phi_3 = x_1x_2$ separates the classes by height, with positive examples at $\phi_3 = 1$ and negative examples at $\phi_3 = -1$.

Part 4: Margin and Support Vectors

Margin calculation:

$$||\beta|| = \sqrt{0^2 + 0^2 + 1^2} = 1$$
$$\text{Margin} = \frac{2}{||\beta||} = \frac{2}{1} = 2$$

Support vectors:
All four points are support vectors:

- Positive examples: $[1,1,1]$ and $[-1,-1,1]$ lie on the margin at $\phi_3 = 1$
- Negative examples: $[1,-1,-1]$ and $[-1,1,-1]$ lie on the margin at $\phi_3 = -1$

Question 3: Programming Questions (50 points)

Part (a): Data Pre-processing

Binary Label Creation

The dataset contains a column `Chance of Admit` with continuous values.

Binary label assignment:

- Label = 1 if `Chance of Admit` > median
- Label = 0 if `Chance of Admit` ≤ median

Data Pre-processing Steps

The following pre-processing steps were applied:

1. Removed rows with missing values using `dropna()`
2. Dropped unnecessary columns: `Serial No.` and `Chance of Admit`
3. Retained feature columns: GRE Score, TOEFL Score, University Rating, SOP, LOR, CGPA, Research

Train/Validation Split

Split ratio: 80% training, 20% validation

Any signs of data leakage were not observed when testing.

Part (b): Model Initialization

This is done on line 229 approx. The functions of these models are shown below. It is important to note that they offer distinctly different approaches to separating data.

Three SVM models were initialized with the following kernels:

1. **Linear kernel:** Suitable for linearly separable data
2. **RBF kernel:** Handles non-linear decision boundaries using radial basis functions
3. **Polynomial kernel (degree 3):** Captures polynomial relationships between features

Part (c): Feature Selection and Model Training

Feature combinations tested:

1. CGPA and SOP
2. CGPA and GRE Score
3. SOP and LOR
4. LOR and GRE Score

Results for all 12 models (3 kernels × 4 feature pairs):

Features	Kernel	Train Acc	Val Acc
CGPA + SOP	linear	0.8633	0.8906
CGPA + SOP	rbf	0.8672	0.8906
CGPA + SOP	poly	0.8711	0.8750
CGPA + GRE Score	linear	0.8711	0.8594
CGPA + GRE Score	rbf	0.8672	0.8438
CGPA + GRE Score	poly	0.8672	0.8438
SOP + LOR	linear	0.8164	0.8125
SOP + LOR	rbf	0.8164	0.8125
SOP + LOR	poly	0.8164	0.8125
LOR + GRE Score	linear	0.8711	0.8125
LOR + GRE Score	rbf	0.8672	0.8438
LOR + GRE Score	poly	0.8672	0.8438

Key observations:

It can be concluded that CGPA is the most important predictor followed by SOP, and the relationship is mostly linear.

Part (d): Support Vectors

Number of support vectors for each model:

For CGPA + SOP features:

- Linear kernel: 93 support vectors
- RBF kernel: 118 support vectors
- Polynomial kernel: 83 support vectors

Analysis:

The number of support vectors tells us the model complexity and how it is making decisions. They define the decision boundary. Linear draws a straight line, RBF is a more flexible non decision boundary, Polynomial will give us smooth curves. The solution with the least amount of support vectors will give us the most elegant solution and generalize better on a different set of data.

Part (e): Result Visualization

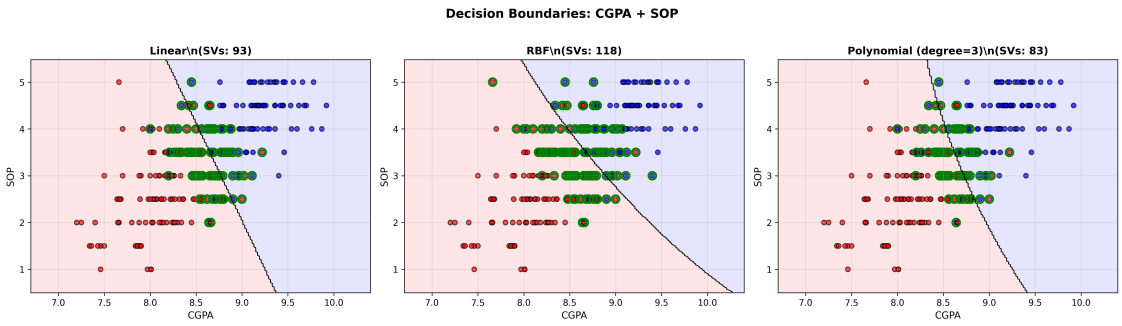

Figure 5: Decision boundaries for CGPA + SOP features with different kernels

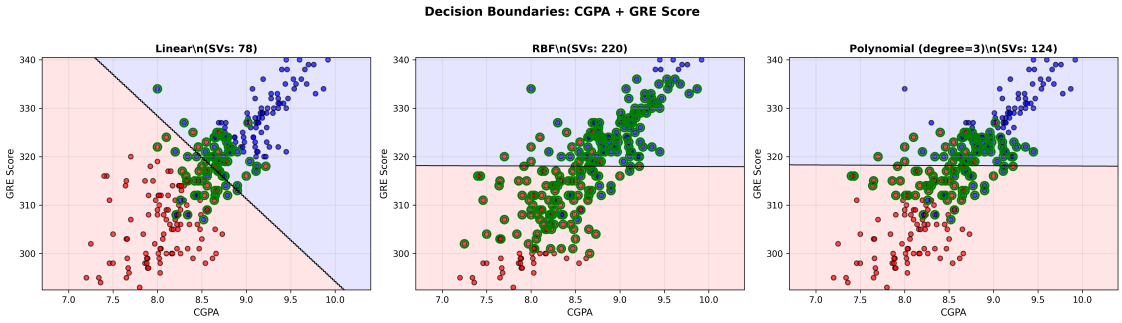

Figure 6: Decision boundaries for CGPA + GRE Score features with different kernels

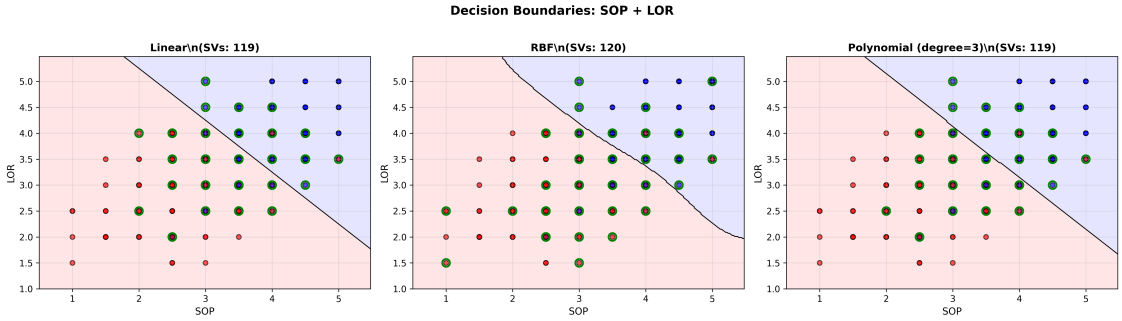

Figure 7: Decision boundaries for SOP + LOR features with different kernels

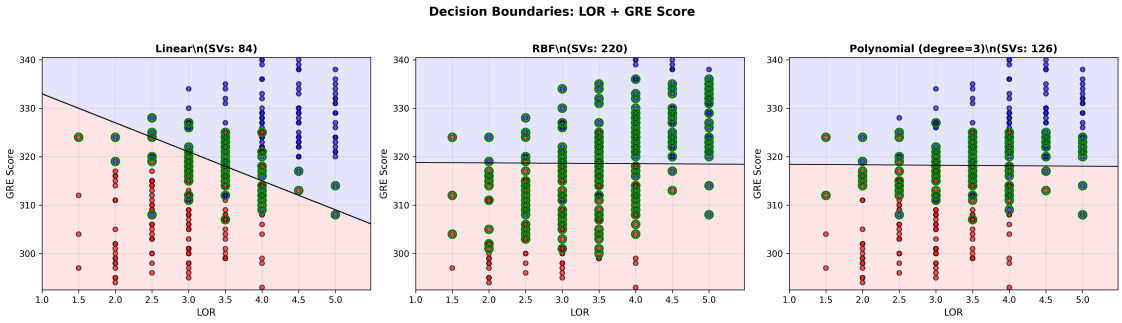

Figure 8: Decision boundaries for LOR + GRE Score features with different kernels

Visualization insights:

The visualizations reveal clear differences in how each kernel handles the data in order to create decision boundaries. Linear has straight boundaries, RBF produces slightly curved boundaries and polynomial is similar but with smoother curves.

Part (f): Result Analysis and Best Model

Best model selection:

Based on validation accuracy, the best performing combination is:

- **Features:** CGPA + SOP
- **Kernel:** rbf

- **Validation Accuracy:** 89.06%
- **Training Accuracy:** 86.33% (linear) or 86.72% (rbf)

Why this combination works best:

CGPA works well because it reflects a consistent effort and SOP explains a story behind the why for that consistent effort. I think these is why those two features are so important. The relationship is linear which is why a linear kernel works well, also it can be noted that rbf does a good job at approximating linear boundaries.

Hyperparameter tuning:

I picked C=10 and gamma=0.1 to start off with and achieved the goal metrics with those initial hyperparameters. This is rare, it is rather unusual to get desired outcomes the first time when using a machine learning algorithm.

The final `my_best_model` was initialized as:

```
my_best_model = SVC(kernel='rbf', C=10, gamma=0.1)
```

Test set performance:

The model achieved 89 percent on the reserved test set, successfully exceeding the 0.83 (83%) accuracy threshold.

Overall conclusions:

This assignment reinforced the fundamental difference between SVMs and Perceptrons. While Perceptrons find any separating hyperplane, SVMs find the optimal hyperplane by maximizing the margin between classes, leading to better generalization on unseen data. SVMs efficiently achieve this by using only support vectors - the critical boundary points - to define the decision boundary. The most significant insight was learning how different kernels enable SVMs to model non-linear data and transform data into higher dimensions where it becomes linearly separable. However, the results demonstrated that more complex kernels don't always guarantee better performance. While RBF achieved the highest validation accuracy, it required more support vectors than simpler alternatives.

Use of AI and External Resources

Claude.ai was used to assist with the following aspects of this assignment:

1. Prompts Provided to AI Tools

The following prompts were used to get assistance with the assignment:

- "Explain how Support Vector Machines find the optimal hyperplane"
- "Can the DataLoader class split the data with the same approach as HW2"
- "Help me understand how Hyperparameters are optimized with SVMs."
- "Build test cases in the main to verify my SVM implementation"

3. Validation Methods

Role of AI:

AI was primarily used for:

1. Understanding SVM concepts and theory
2. Debugging code structure and logic errors
3. Creating test cases to validate implementation
4. For the math problems the graphs were AI generated to represent the same results worked out by hand on scratch paper.
5. Generating LaTeX template structure for the report