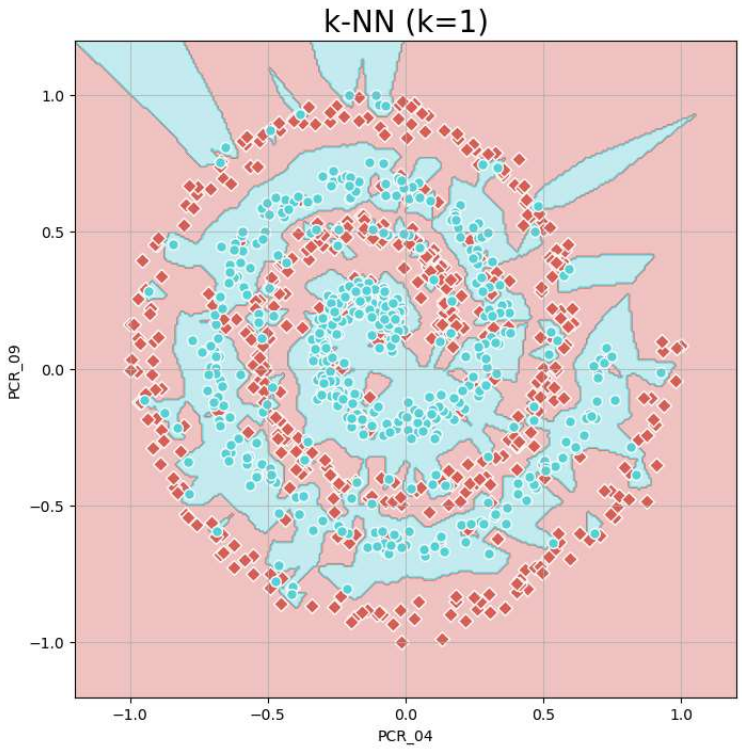
Machine Learning MAJOR_HW2

Majd Khazen (206708463)
Ayman Barham (212201057)

Q1:



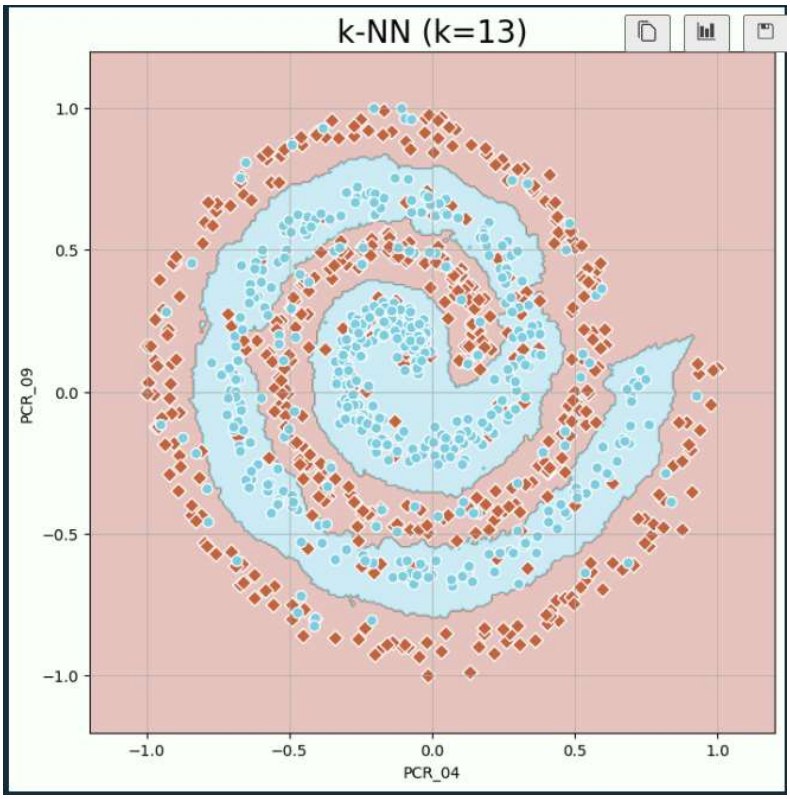
Q2:

based on the validation curve, the optimal k value is identified as 13.
note that he best k is the one that provides the highest validation accuracy. In the provided plot, this appears to be 13.
Average Training Accuracy: 84.13%
Average Validation Accuracy: 83.7%

what values of k are causing overfitting : Low k values (for example 2,3,5) , With low k, the model is highly flexible and can capture noise from the training data, leading to very high training accuracy but lower validation accuracy due to poor generalization to unseen data.

what values of k are casing underfitting : High valsue like $k > 50$ With high k, the model becomes too simple and smooth, failing to capture the underlying patterns in the data. Both training and validation accuracies are low because the model has high bias and cannot fit the data well.

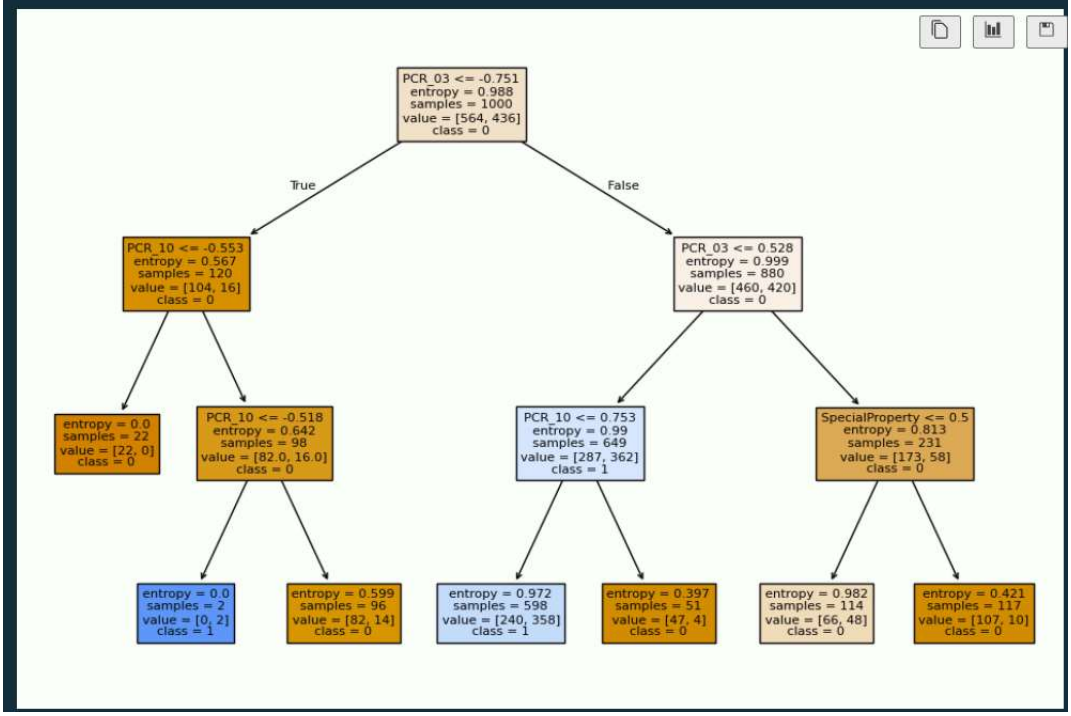
Q3 :
Test Accuracy: 82.80%



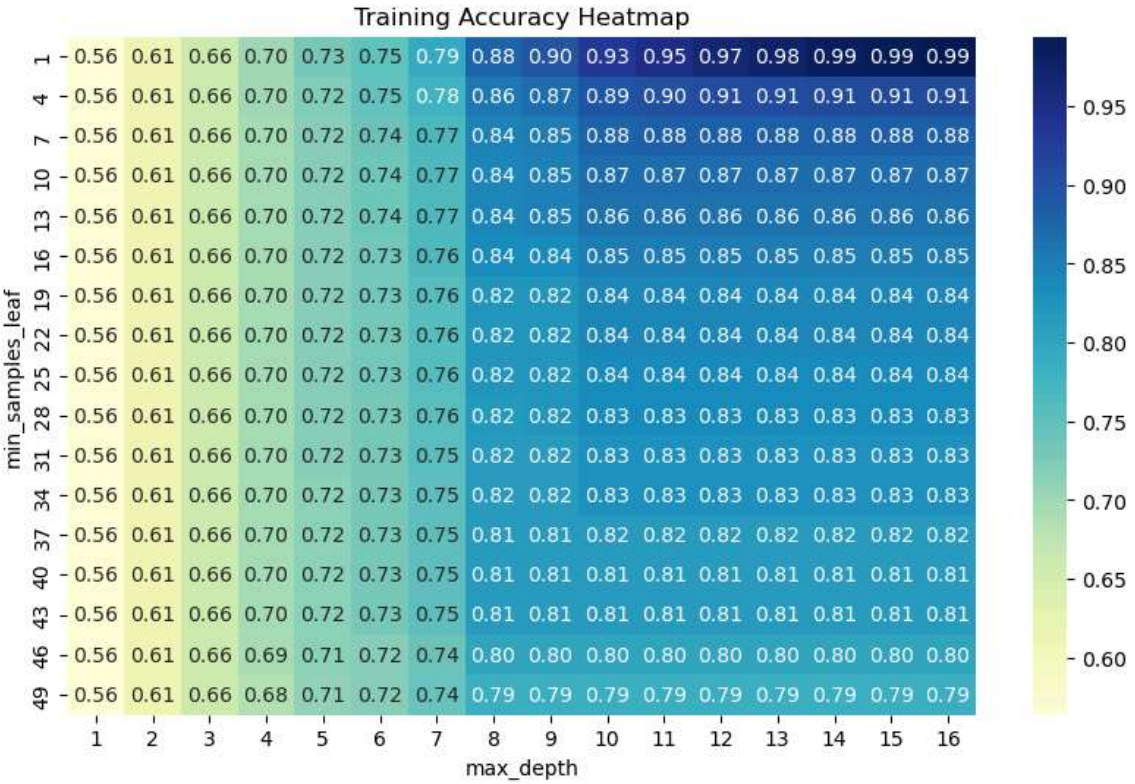
Q4 :
k-NN (k=1): The boundary is very complex and jagged, closely following the training data points.
This model exhibits overfitting, as it is highly sensitive to the training data, capturing noise and small variations.
k-NN (k=13): The boundary is smoother and more generalized compared to k=1.
the decision boundary is less complex, indicating that the model is less likely to overfit and is better at generalizing to unseen data.

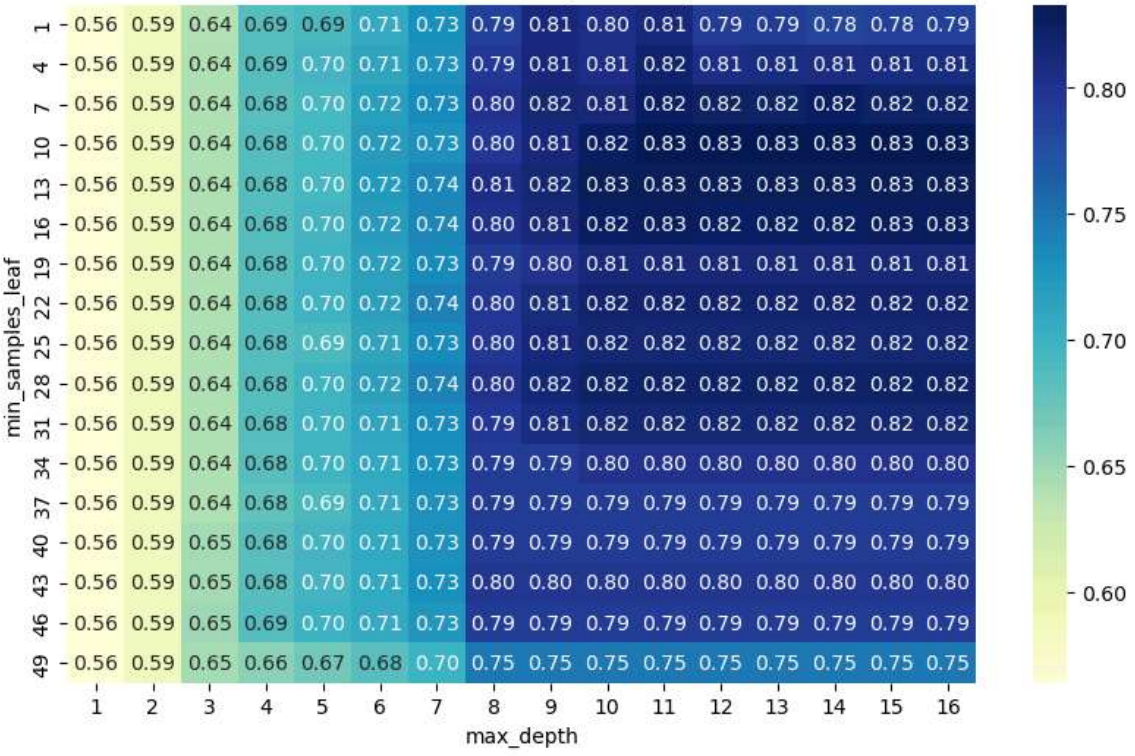
Q5 :
Training Accuracy: 68.40%

```
Training Accuracy: 68.40%
```



Q6:


Training Accuracy Heatmap
min_samples_leaf
1 0.56 0.61 0.66 0.70 0.73 0.75 0.79 0.88 0.90 0.93 0.95 0.97 0.98 0.99 0.99 0.99
4 0.56 0.61 0.66 0.70 0.72 0.75 0.78 0.86 0.87 0.89 0.90 0.91 0.91 0.91 0.91 0.91
7 0.56 0.61 0.66 0.70 0.72 0.74 0.77 0.84 0.85 0.88 0.88 0.88 0.88 0.88 0.88 0.88
10 0.56 0.61 0.66 0.70 0.72 0.74 0.77 0.84 0.85 0.87 0.87 0.87 0.87 0.87 0.87 0.87
13 0.56 0.61 0.66 0.70 0.72 0.74 0.77 0.84 0.85 0.86 0.86 0.86 0.86 0.86 0.86 0.86
16 0.56 0.61 0.66 0.70 0.72 0.73 0.76 0.84 0.84 0.85 0.85 0.85 0.85 0.85 0.85 0.85
19 0.56 0.61 0.66 0.70 0.72 0.73 0.76 0.82 0.82 0.84 0.84 0.84 0.84 0.84 0.84 0.84
22 0.56 0.61 0.66 0.70 0.72 0.73 0.76 0.82 0.82 0.84 0.84 0.84 0.84 0.84 0.84 0.84
25 0.56 0.61 0.66 0.70 0.72 0.73 0.76 0.82 0.82 0.84 0.84 0.84 0.84 0.84 0.84 0.84
28 0.56 0.61 0.66 0.70 0.72 0.73 0.76 0.82 0.82 0.83 0.83 0.83 0.83 0.83 0.83 0.83
31 0.56 0.61 0.66 0.70 0.72 0.73 0.75 0.82 0.82 0.83 0.83 0.83 0.83 0.83 0.83 0.83
34 0.56 0.61 0.66 0.70 0.72 0.73 0.75 0.82 0.82 0.83 0.83 0.83 0.83 0.83 0.83 0.83
37 0.56 0.61 0.66 0.70 0.72 0.73 0.75 0.81 0.81 0.82 0.82 0.82 0.82 0.82 0.82 0.82
40 0.56 0.61 0.66 0.70 0.72 0.73 0.75 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81
43 0.56 0.61 0.66 0.70 0.72 0.73 0.75 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81 0.81
46 0.56 0.61 0.66 0.69 0.71 0.72 0.74 0.80 0.80 0.80 0.80 0.80 0.80 0.80 0.80 0.80
49 0.56 0.61 0.66 0.68 0.71 0.72 0.74 0.79 0.79 0.79 0.79 0.79 0.79 0.79 0.79 0.79
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
max_depth
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.60


Validation Accuracy Heatmap
min_samples_leaf
1 0.56 0.59 0.64 0.69 0.69 0.71 0.73 0.79 0.81 0.80 0.81 0.79 0.79 0.78 0.78 0.79
4 0.56 0.59 0.64 0.69 0.70 0.71 0.73 0.79 0.81 0.81 0.82 0.81 0.81 0.81 0.81 0.81
7 0.56 0.59 0.64 0.68 0.70 0.72 0.73 0.80 0.82 0.81 0.82 0.82 0.82 0.82 0.82 0.82
10 0.56 0.59 0.64 0.68 0.70 0.72 0.73 0.80 0.81 0.82 0.83 0.83 0.83 0.83 0.83 0.83
13 0.56 0.59 0.64 0.68 0.70 0.72 0.74 0.81 0.82 0.83 0.83 0.83 0.83 0.83 0.83 0.83
16 0.56 0.59 0.64 0.68 0.70 0.72 0.74 0.80 0.81 0.82 0.83 0.82 0.82 0.82 0.83 0.83
19 0.56 0.59 0.64 0.68 0.70 0.72 0.73 0.79 0.80 0.81 0.81 0.81 0.81 0.81 0.81 0.81
22 0.56 0.59 0.64 0.68 0.70 0.72 0.74 0.80 0.81 0.82 0.82 0.82 0.82 0.82 0.82 0.82
25 0.56 0.59 0.64 0.68 0.69 0.71 0.73 0.80 0.81 0.82 0.82 0.82 0.82 0.82 0.82 0.82
28 0.56 0.59 0.64 0.68 0.70 0.72 0.74 0.80 0.82 0.82 0.82 0.82 0.82 0.82 0.82 0.82
31 0.56 0.59 0.64 0.68 0.70 0.71 0.73 0.79 0.81 0.82 0.82 0.82 0.82 0.82 0.82 0.82
34 0.56 0.59 0.64 0.68 0.70 0.71 0.73 0.79 0.79 0.80 0.80 0.80 0.80 0.80 0.80 0.80
37 0.56 0.59 0.64 0.68 0.69 0.71 0.73 0.79 0.79 0.79 0.79 0.79 0.79 0.79 0.79 0.79
40 0.56 0.59 0.65 0.68 0.70 0.71 0.73 0.79 0.79 0.79 0.79 0.79 0.79 0.79 0.79 0.79
43 0.56 0.59 0.65 0.68 0.70 0.71 0.73 0.80 0.80 0.80 0.80 0.80 0.80 0.80 0.80 0.80
46 0.56 0.59 0.65 0.69 0.70 0.71 0.73 0.79 0.79 0.79 0.79 0.79 0.79 0.79 0.79 0.79
49 0.56 0.59 0.65 0.66 0.67 0.68 0.70 0.75 0.75 0.75 0.75 0.75 0.75 0.75 0.75 0.75
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
max_depth
0.80
0.75
0.70
0.65
0.60

- **Optimal Hyperparameter Combination :**
  Based on the validation accuracy heatmap, the optimal hyperparameter combination is with a validation accuracy of 83.3%. there is more than one slot for example: min_samples_leaf=10, max_depth=11
- **Hyperparameter Combination Causing Underfitting:**
  The combination of `max_depth=1` and `min_samples_leaf=1` results in underfitting, as seen from the low training accuracy of 56% and validation accuracy of 56% this shows that the model is too simple. therefore this model is underfitting.
- **Hyperparameter Combination Causing Overfitting**
  The combination of `max_depth=16` and `min_samples_leaf=1` results in overfitting, as indicated by the high training accuracy of 99.9% and lower validation accuracy of 79%.
- **Discussion on Underfitting and Overfitting**
  - Underfitting: With $max\_depth = 1$ and $min\_samples\_leaf = 1$, the model is too simple to capture the complexities of the data, also the depth of the tree is only 1, meaning it can only split the training data based on one feature, which makes it very simple and terrible at fitting the data.this leads to poor performance on both training and validation sets. This simplicity prevents the model from learning the underlying patterns.
  - Overfitting: With $max\_depth = 16$ and $min\_samples\_leaf = 1$, the model becomes overly complex, capturing noise and outliers in the training data. it can even have leaves that contain only one sample which makes it to specific to the training data. This results in excellent performance on the training set but poor generalization to the validation set, demonstrating classic overfitting behavior.

Q7:

The number of hyperparameter combinations are 2 :

- The max_depth ranges from 1 to 16 (inclusive), resulting in 16 possible values.
- The min_samples_leaf ranges from 1 to 49 jumps of 3 (inclusive), resulting in 17 possible values.

  Total combinations $= 16 \times 17 = 272$

If a third hyperparameter is added, the number of combinations will increase multiplicatively. Suppose the third hyperparameter, criterion, has 2 possible values :

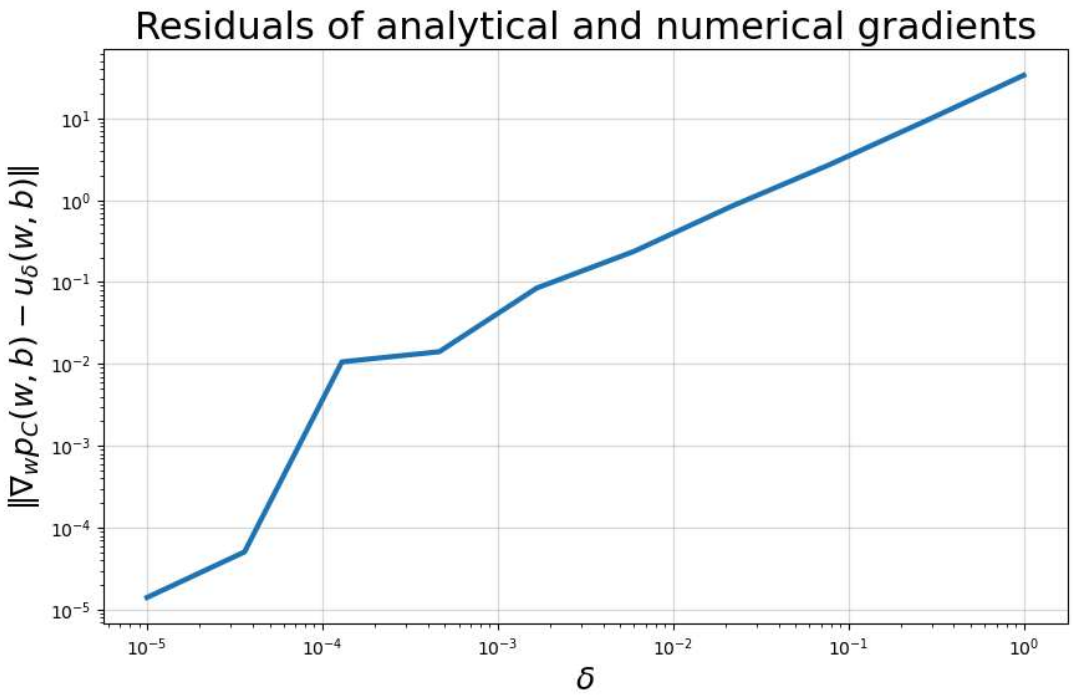
new Total combinations $16 \times 17 \times 2 = 544$

Searching over additional hyperparameters affects the total number of possible combinations exponentially. Each new hyperparameter with n values multiplies the total number of combinations by n . This results in a significantly larger search space, leading to increased computational cost and time required to find the optimal combination. While this can improve model performance by finding better hyperparameter settings, it also necessitates more

resources and can be computationally intensive.

Q8:
Test Accuracy: 81.60%

Q9:



we recall from calculus the definition of analytic derivative: $f'(x) = \lim_{\delta \to 0} \frac{f(x+\delta) - f(x)}{\delta}$
and in the graph that is exactly what we see, when δ is smaller the error gets smaller and smaller and the numeric derivative gets closer to the analytic derivative.

Q10:
Observing the training loss graph, it's evident that the loss starts at a substantial level and then swiftly diminishes within the initial several hundred iterations. Following this steep drop, the loss reduction pace moderates. This pattern can be attributed to the large C parameter value, set at $1e11$, which imposes a stringent emphasis on minimizing classification mistakes, thus accounting for the quick initial fall in loss.

The training accuracy chart illustrates notable variations in accuracy at the start, which gradually evens out as the training progresses. At first, the accuracy ascends to approximately 50%, then it takes a steep dive to roughly 42%, before climbing back up and leveling off in the vicinity of 48-50%. This phenomenon might be explained by the high C

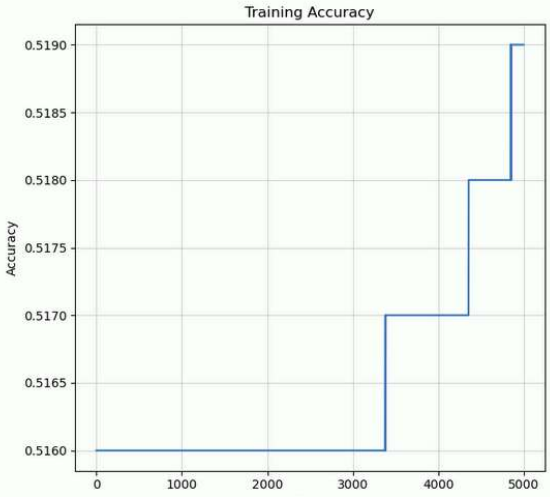
parameter value that the model uses to impose severe penalties on any misclassified examples. This heavy penalization causes the model to react strongly to particular instances in the training data, resulting in the initial fluctuations in accuracy.

naively we would expect that when the loss decreases, the accuracy increases, that is true in models that are like HardSVM. where the only thing we care for is to seperate the data/increase the accuracy of the model.
in SoftSVM however, we look to minimize 2 things together, trying to seperate the data, and minimizing the loss which is compromised of the margin violation. so a scenario like this is possible:
2 different SoftSVM models A and B, A has better accuracy but higher loss than B. (for example look at the graph after 500 steps (A) and after 1000 steps (B))
and sometimes a SoftSVM classifier can give one point more importance than 2 other points because one boldly misclassified point can contribute more to the loss than 2 barely misclassified points in terms of margin violation (of course depending on the loss function used)
This is not completely unexpected behavior as the fluctuating of the accuracy doesn't mean that the model is not learning (for example, the accuracy increases and the loss decreases, meaning the model is optimizing for the loss and learning but needs more time and steps for us to see the learning's results in accuracy). but in any case we would rather that the model learned faster and that it's accuracy increased while its loss decreased together, we would get faster convergence.

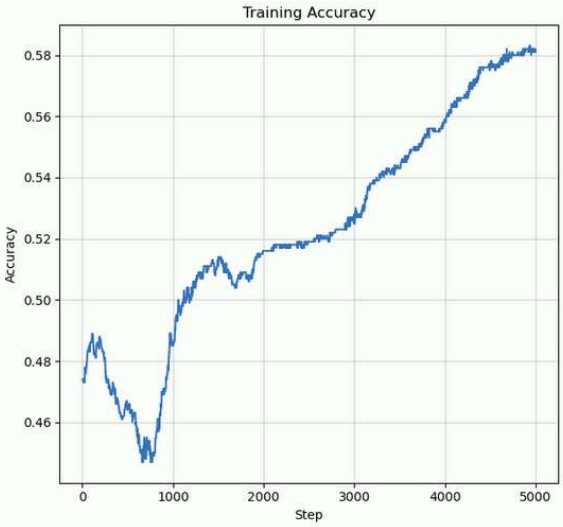
in order to settle this, we can try adjusting the regularization parameter C and the learning rate lr. Reducing C may lead to a more stable decision boundary and better generalization. Carefully increasing lr could help the model converge more quickly and stabilize the accuracy, if we increase too much, it may cause overshooting, and it may even not converge to the local minimum.
Additionally, performing cross-validation to tune these hyperparameters can ensure a better balance between bias and variance, leading to improved generalization performance.

Q11:

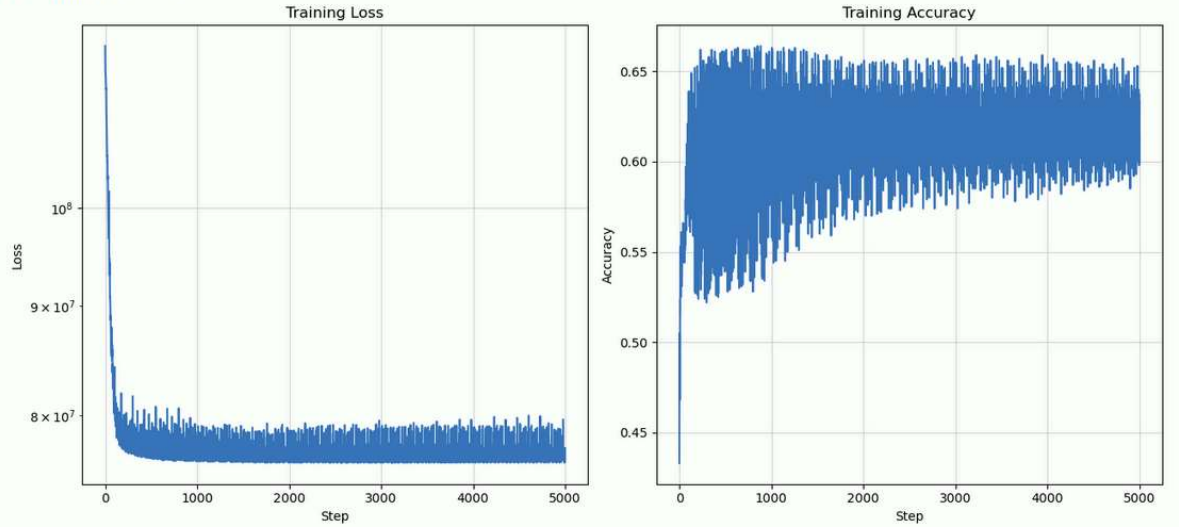
Learning rate: 1e-11



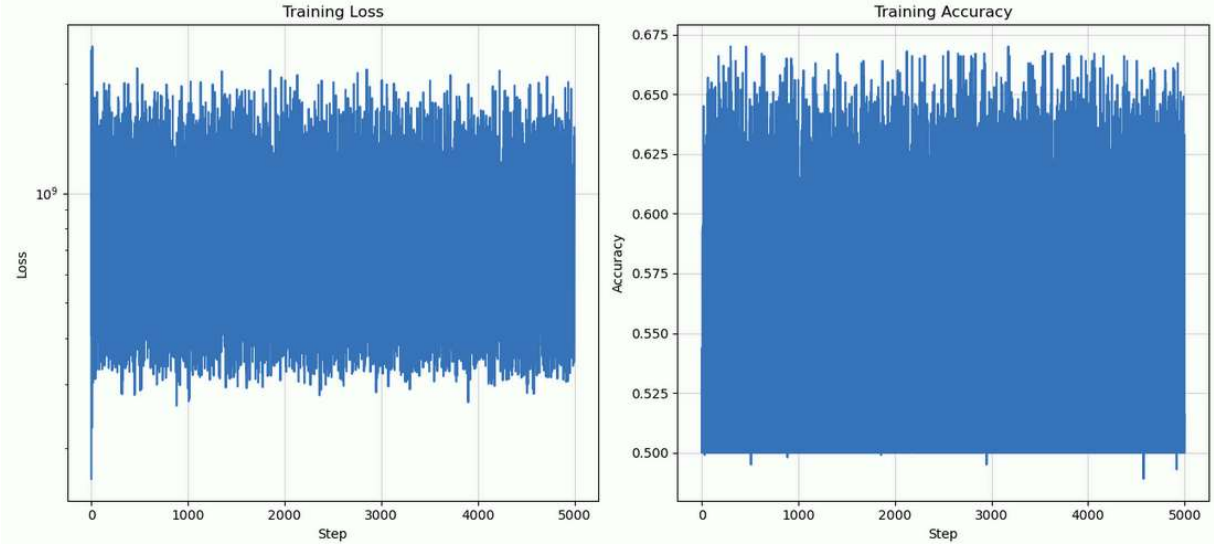
Learning rate: 1e-09

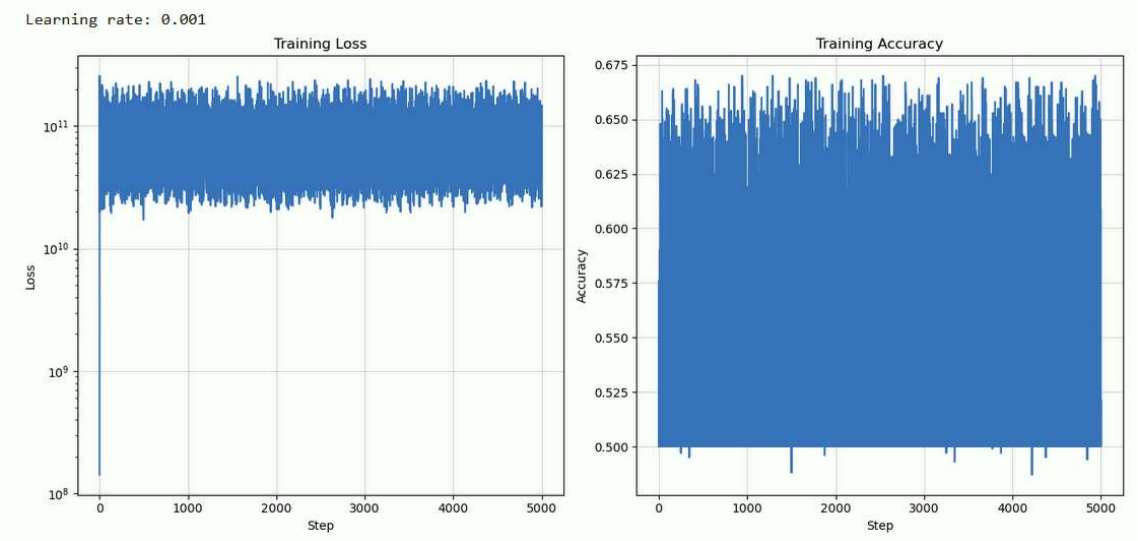


Learning rate: 1e-07



Learning rate: 1e-05





Learning rate: $1e-07$ is the best choice. let's compare it to each and every one of the other learn rates.

in comparison to $1e-11$:

we can see that $1e-11$ is much more stable, on the other hand though its converging very slowly and in the end of the training it reaches a very low accuracy, and very high loss (compared to $1e-07$), so all in all $1e-07$ is the better choice.

in comparison to $1e-09$:

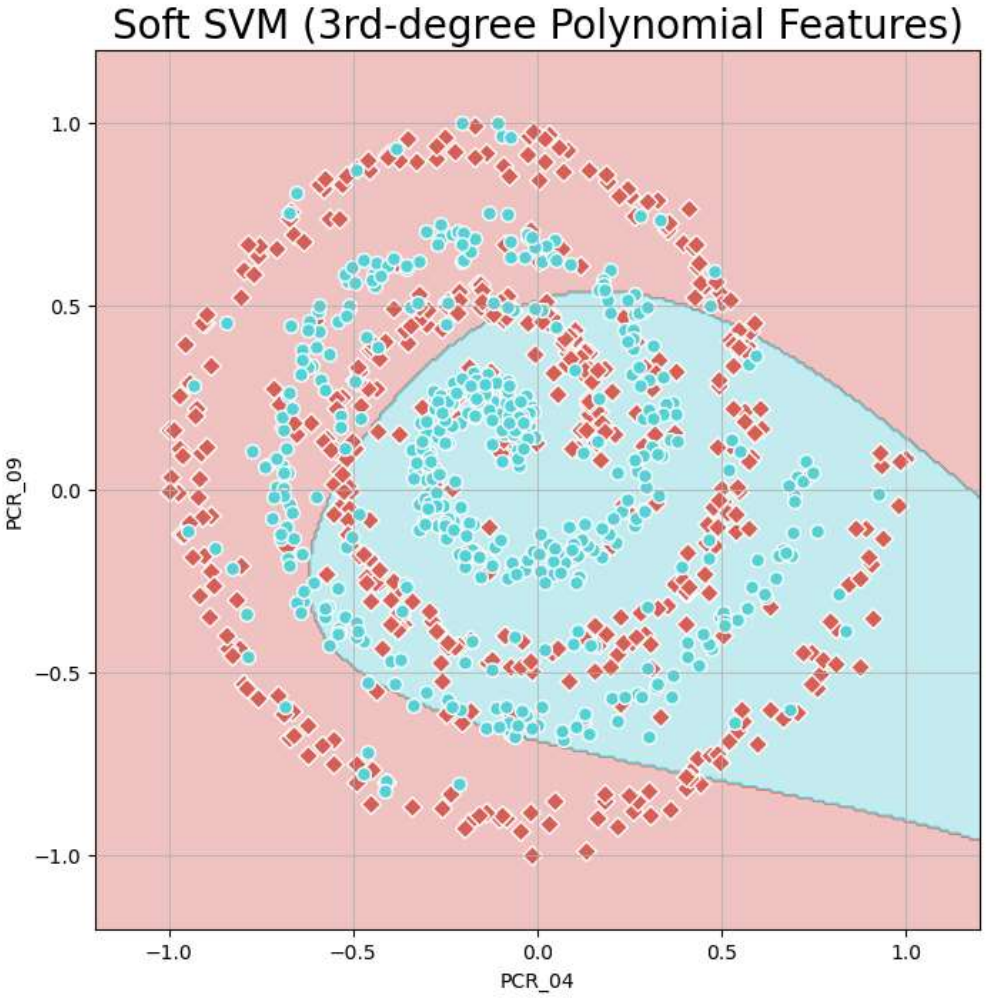
we can see that $1e-09$ is more stable, on the other hand though its converging very slowly and in the end of the training it reaches a low accuracy, and high loss (compared to $1e-07$), so all in all $1e-07$ is the better choice.

in comparison to $1e-05$ and $1e-03$:

here we can see that the learning rates are way too high and too unstable. the loss and accuracy in both cases go down to lower values than $1e-07$. so it's better to use $1e-07$

This learning rate offers a compromise between the very slow and steady progress of $1e-11$ and the fast but unstable convergence of $1e-05$, making it a practical choice for training the Soft SVM model.

Q12:



Training Accuracy: 58.60%
Test Accuracy: 56.8%

Q13:

a.

$$h(x) = sign\left(\sum_{j=1}^{m} \alpha_j y_j K(x, x_j)\right) \underset{\forall \alpha_i : \alpha_i = 1}{=} sign\left(\sum_{j=1}^{m} y_i \cdot exp\{-\gamma \|x_j - x\|_2\}\right)$$

b.

$$h(x) = sign\left(\sum_{j\in 1}^{m} y_j K(x, x_j)\right) \underset{divide\ to\ 2\ cases}{=} sign\left(\sum_{j\in\{i\in[m]|y_i=1\}} y_j K(x, x_j) + \sum_{j\in\{i\in[m]|y_i=-1\}} y_j K(x, x_j)\right)$$
$$= sign\left(\sum_{j\in\{i\in[m]|y_i=1\}} K(x, x_j) - \sum_{j\in\{i\in[m]|y_i=-1\}} K(x, x_j)\right)$$

lets assume there exists $\delta > 1$ that for every distinct $i, j \in [m] : \|x_i - x\|_2 > 3\delta$
and for every $x \in D$ there exists a tuple $(x_i, y_i) \in S$ such that: $y = y_i\ and\ \|x_i - x\|_2 < \delta - 1$

c. given x with classification $y = 1$.
there exists from above an index i^* such that:

$$\#(x_{i^*}, y_{i^*}) \in S :\ y = y_{i^*}\ and\ \|x - x_{i^*}\|_2 < \delta - 1$$
$$\sum_{j\in\{i\in[m]:\ y_i=1\}}^{m} K(x, x_j) = \sum_{\substack{j\in\{i\in[m]:\ y_i=1\} \\ j\neq i^*}}^{m} exp\{-\gamma\|x - x_j\|_2\} + exp\{-\gamma\|x - x_{i^*}\|_2\}$$
$$\underset{\forall z\in\mathbb{R}:\ exp\{z\}>0}{>} exp\{-\gamma\|x - x_{i^*}\|_2\} \underset{\substack{using\ \# \\ and\ exp\ is\ monotone\ increasing}}{>} exp\{-\gamma(\delta - 1)\}$$

d.

$$\sum_{j\in\{i\in[m]:\ y_i=-1\}}^{m} K(x, x_j) = \sum_{j\in\{i\in[m]:\ y_i=-1\}}^{m} exp\{-\gamma\|x - x_j\|_2\} =$$
$$= \sum_{j\in\{i\in[m]:\ y_i=-1\}}^{m} exp\{-\gamma\|x - x_{i^*} + x_{i^*} - x_j\|_2\}$$
$$\underset{norm\ triangle\ inequality}{\leqslant} \sum_{j\in\{i\in[m]:\ y_i=-1\}}^{m} exp\{-\gamma(\|x - x_{i^*}\|_2 + \|x_{i^*} - x_j\|_2)\}$$
$$\leqslant \sum_{j\in\{i\in[m]:\ y_i=-1\}}^{m} exp\{-\gamma(\|x - x_{i^*}\|_2 + 3\delta)\} = \frac{m}{2} exp\{-\gamma(\|x - x_{i^*}\|_2 + 3\delta)\}$$
$$\leqslant \frac{m}{2} exp\{-\gamma(-\|x - x_{i^*}\|_2 + 3\delta)\} \leqslant \frac{m}{2} exp\{-\gamma((\delta - 1) + 3\delta)\} = \frac{m}{2} exp\{-\gamma(2\delta + 1)\}$$
$$\leqslant \frac{m}{2} exp\{-\gamma(2\delta - 1)\}$$

e.

$$for\ \gamma = ln\left(\frac{m}{2}\right):$$

$$\sum_{j\in\{i\in[m]|y_i=1\}} K(x,x_j) - \sum_{j\in\{i\in[m]|y_i=-1\}} K(x,x_j) \underset{using\ (c)}{>}$$

$$> -\gamma(\delta-1) - \sum_{j\in\{i\in[m]|y_i=-1\}} K(x,x_j) \underset{using\ (d)\ and\ minus}{>} exp\{-\gamma(\delta-1)\} - \frac{m}{2}exp\{-\gamma(2\delta-1)\} =$$

$$> exp\{-\gamma(\delta-1)\} - \frac{m}{2}exp\{-\gamma(2\delta-1)\} = exp\{-\gamma(\delta-1)\} - \frac{m}{2}exp\{-\gamma - \gamma(2\delta-2)\}$$

$$\underset{\substack{\gamma=\ln\left(\frac{m}{2}\right),\ -\gamma=\ln\left(\frac{2}{m}\right),\\ exp(a+b)=exp(a)\cdot exp(b)}}{=} exp\{-\gamma(\delta-1)\} - \frac{m}{2}\cdot exp\left(\ln\left(\frac{2}{m}\right)\right)exp\{-2\gamma(\delta-1)\} \underset{exp(ln(z))=z}{=}$$

$$= (exp\{-\gamma(\delta-1)\} - exp\{-2\gamma(\delta-1)\} \underset{e^{-t}>e^{-2t}}{>} 0$$

$$so\ from\ here\ since:$$

$$h(x) = sign\left(\sum_{j\in\{i\in[m]|y_i=1\}} K(x,x_j) - \sum_{j\in\{i\in[m]|y_i=-1\}} K(x,x_j)\right)$$

$$we\ get\ that\ h(x) = 1$$

f.

given x, we devide to 2 cases:

- $y = 1$: already proved this case above.
- $y = 2$: the proof for this case is exactly the same as above but changes with respect to the sign of y (will provide the proof with the changes but it will duplicated code :))

proof for case of

given x with classification $y = -1$.

there exists from above an index i^* such that:

$$\#(x_{i^*}, y_{i^*}) \in S:\ y = y_{i^*}\ and\ \|x - x_{i^*}\|_2 < \delta - 1$$

$$\sum_{j\in\{i\in[m]:\ y_i=-1\}}^{m} K(x,x_j) = \sum_{\substack{j\in\{i\in[m]:\ y_i=-1\}\\ j\neq i^*}}^{m} exp\{-\gamma\|x-x_j\|_2\} + exp\{-\gamma\|x-x_{i^*}\|_2\}$$

$$\underset{\forall z\in\mathbb{R}:\ exp\{z\}>0}{>} exp\{-\gamma\|x-x_{i^*}\|_2\} \underset{\substack{using\ \#\\ and\ exp\ is\ monotone\ increasing}}{>} exp\{-\gamma(\delta-1)\}$$

d.

$$\sum_{j\in\{i\in[m]:\, y_i=1\}}^{m} K(x,x_j) \;=\; \sum_{j\in\{i\in[m]:\, y_i=1\}}^{m} exp\{-\gamma\|x-x_j\|_2\} =$$

$$= \sum_{j\in\{i\in[m]:\, y_i=1\}}^{m} exp\{-\gamma\|x-x_{i^*}+x_{i^*}-x_j\|_2\}$$

$$\underset{\text{norm triangle inequality}}{\leqslant} \sum_{j\in\{i\in[m]:\, y_i=1\}}^{m} exp\{-\gamma(\|x-x_{i^*}\|_2+\|x_{i^*}-x_j\|_2)\}$$

$$\leqslant \sum_{j\in\{i\in[m]:\, y_i=1\}}^{m} exp\{-\gamma(\|x-x_{i^*}\|_2+3\delta)\} = \frac{m}{2}exp\{-\gamma(\|x-x_{i^*}\|_2+3\delta)\}$$

$$\leqslant \frac{m}{2}exp\{-\gamma(-\|x-x_{i^*}\|_2+3\delta)\} \leqslant \frac{m}{2}exp\{-\gamma((\delta-1)+3\delta)\} = \frac{m}{2}exp\{-\gamma(2\delta+1)\}$$

$$\leqslant \frac{m}{2}exp\{-\gamma(2\delta-1)\}$$

e.

$for\ \gamma = ln\left(\frac{m}{2}\right)$:

$$\sum_{j\in\{i\in[m]|y_i=1\}} K(x,x_j) - \sum_{j\in\{i\in[m]|y_i=-1\}} K(x,x_j) \underset{using\ (c)}{>}$$

$$> \sum_{j\in\{i\in[m]|y_i=1\}} K(x,x_j) - \gamma(\delta-1) \underset{using\ (d)\ and\ minus}{>} - exp\{-\gamma(\delta-1)\} + \frac{m}{2}exp\{-\gamma(2\delta-1)\} =$$

$$> -exp\{-\gamma(\delta-1)\} + \frac{m}{2}exp\{-\gamma(2\delta-1)\} = -exp\{-\gamma(\delta-1)\} + \frac{m}{2}exp\{-\gamma\ -\gamma(2\delta-2)\}$$

$$\underset{\substack{\gamma=\ln\left(\frac{m}{2}\right),\ -\gamma=\ln\left(\frac{2}{m}\right),\\ exp(a+b)=exp(a)\cdot exp(b)}}{=} -exp\{-\gamma(\delta-1)\} + \frac{m}{2}\cdot exp\left(\ln\left(\frac{2}{m}\right)\right)exp\{-2\gamma(\delta-1)\} \underset{exp(ln(z))=z}{=}$$

$$= -exp\{-\gamma(\delta-1)\} + exp\{-2\gamma(\delta-1)\} \underset{e^{-t}>e^{-2t}}{<} 0$$

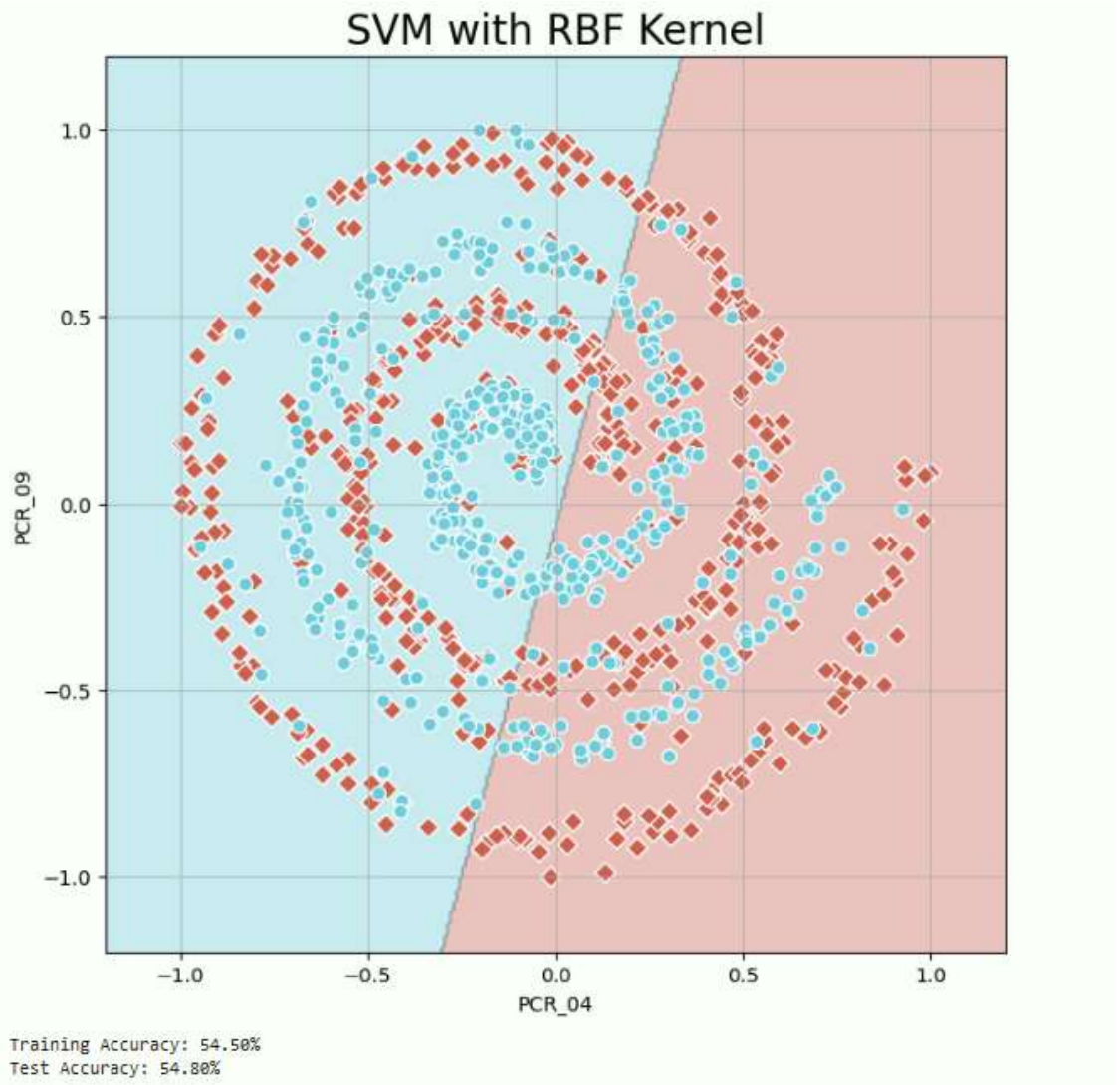
$$so\ from\ here\ since:$$

$$h(x) = sign\left(\sum_{j\in\{i\in[m]|y_i=1\}} K(x,x_j) - \sum_{j\in\{i\in[m]|y_i=-1\}} K(x,x_j)\right)$$

$$we\ get\ that\ h(x) = -1$$

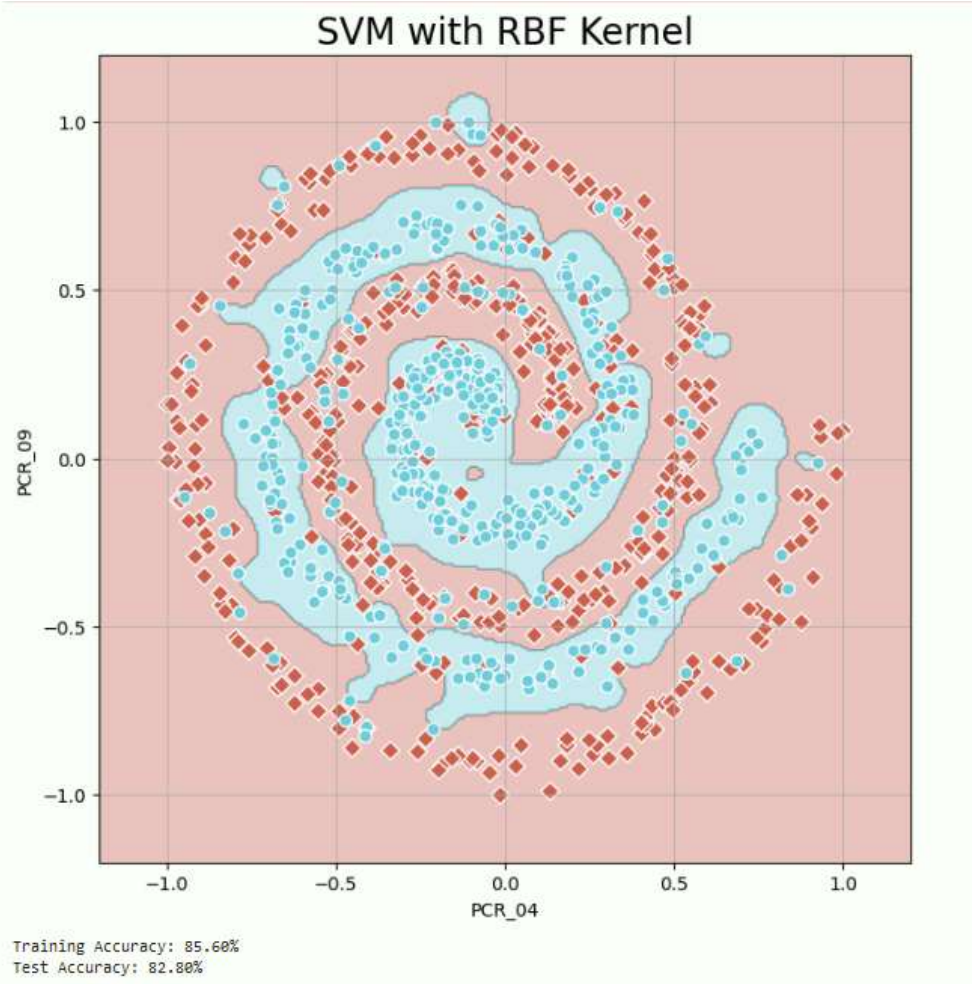
so we proved for all possible cases on y that h classifies right.

Q14:



the SVM model with the RBF kernel is underfitting. The decision boundary is too simple and doesn't capture the complex spiral patterns of the red and blue data points effectively. Many points are misclassified, showing that the model fails to learn the underlying structure of the data. Therefore, this model is underfitting.

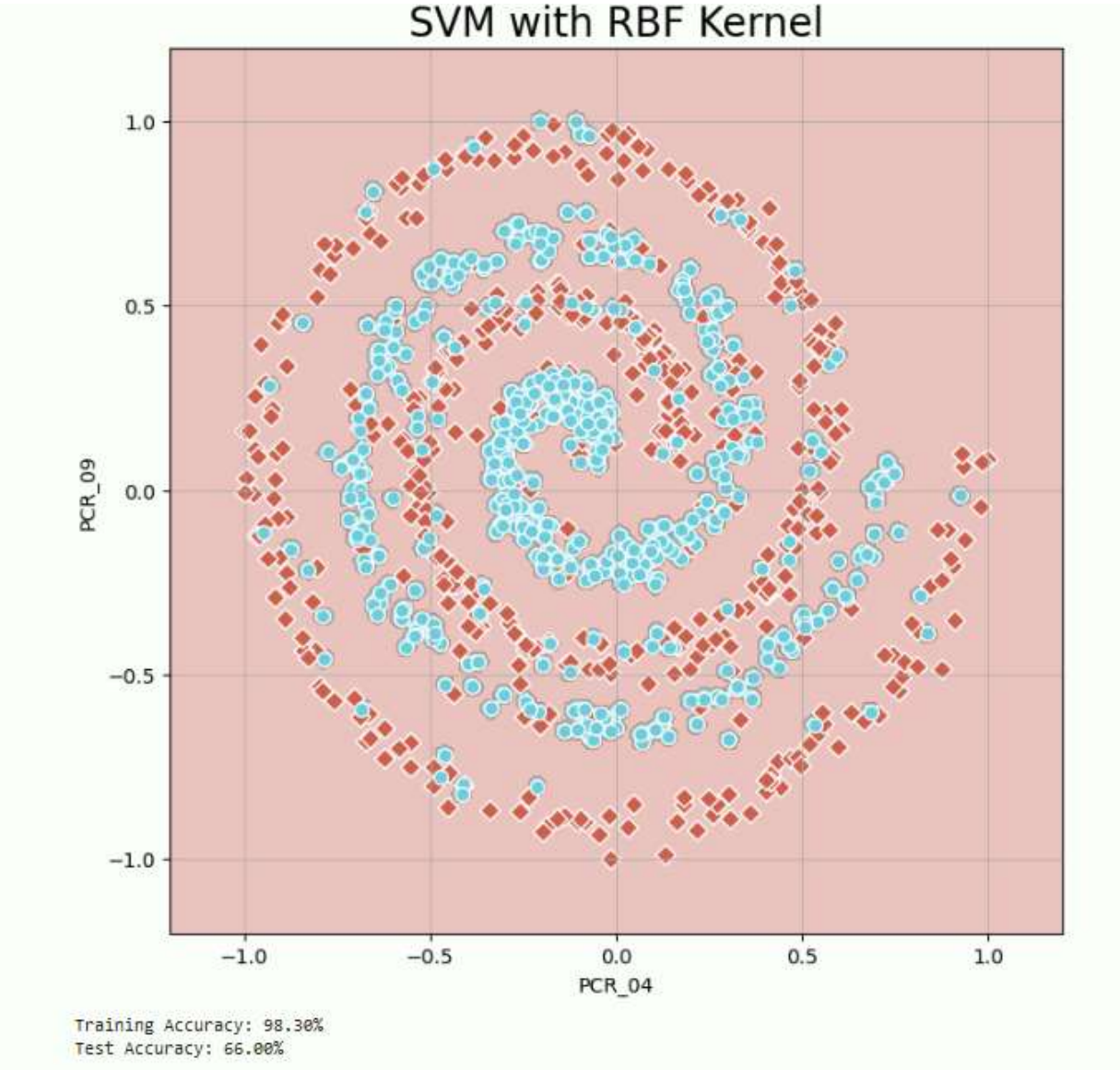
Q15:



The decision boundary observed in this scenario and in Question 3 are remarkably similar, to the extent that they yield identical test accuracies. However, a closer inspection reveals that the decision boundary in this case is marginally smoother. Additionally, there's a region where the feature PCR_09 exceeds 1.0 that is categorized as blue, which is not present in the decision boundaries from Question 3.

To delve deeper, the gamma value in a model dictates how much sway each individual training example has. With a high γ value, such as 200, the impact of each data point is highly localized. This results in a decision boundary that is very responsive to each specific data point, creating a more intricate and convoluted boundary. When the distance between a point x and a training point x_i is not extremely small, the term $exp(-\gamma ||x - x_i||_2)$ diminishes rapidly towards zero. Consequently, the decision function is predominantly affected by data points that are in close proximity, leading to a decision boundary that tightly encircles clusters of data.

Q16:



The plot shows a complex decision boundary that might be too finely tuned to the specific examples in the training set (which due to the hight γ) . The high training accuracy combined with the relatively low test accuracy suggests that the SVM model with the RBF kernel is overfitting.