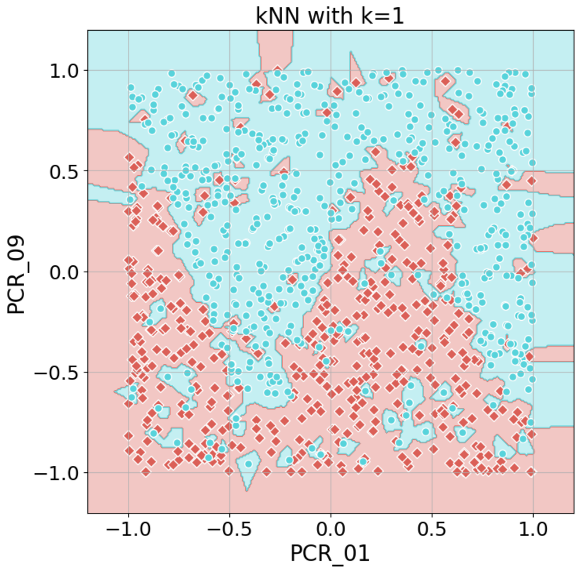
HW2: Algorithm Implementation and Basic Model Selection

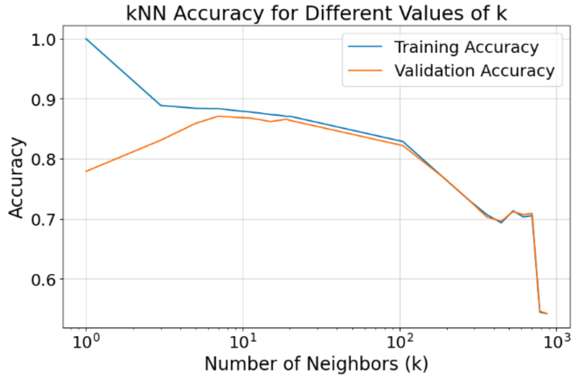
Ori Mintz & Almog Karif

Part 1: Basic model selection with k-Nearest Neighbors

Q1



Q2



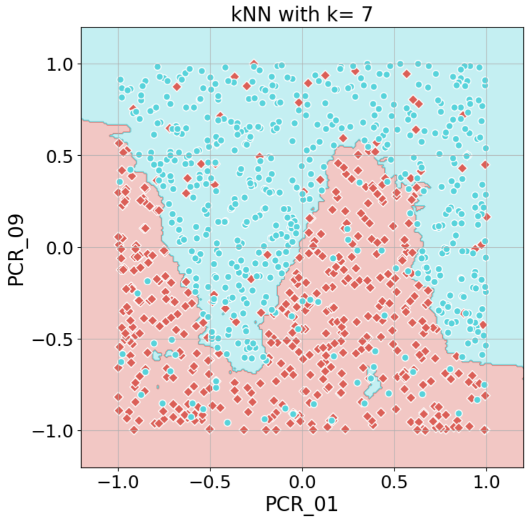
Based on the graph, it is evident that the optimal value of k for validation accuracy is 7. The average training accuracy for this value is 0.884, while the corresponding validation accuracy is 0.871. Furthermore, we observe that when $k \leq 3$, there is an instance of overfitting, indicated by a significantly higher training accuracy compared to the lower validation accuracy. This suggests that the predictions are excessively tailored to the training set, without considering potential errors. Conversely, when $k \geq 105$, both the training mean and validation accuracy experience a notable decline from approximately 0.83 to values below 0.8. This implies that the kNN model is underfitting the training set.

Q3

As we learned at the lecture and tutorial, as the k value increases we get a higher bias and a lower variance that happens because we get a less complex module that is smoother. That causes the prediction to be less precise on the data (lager bias) but the prediction for similar points are closer together (lower variance). Conversely, lower k values produce the opposite effects.

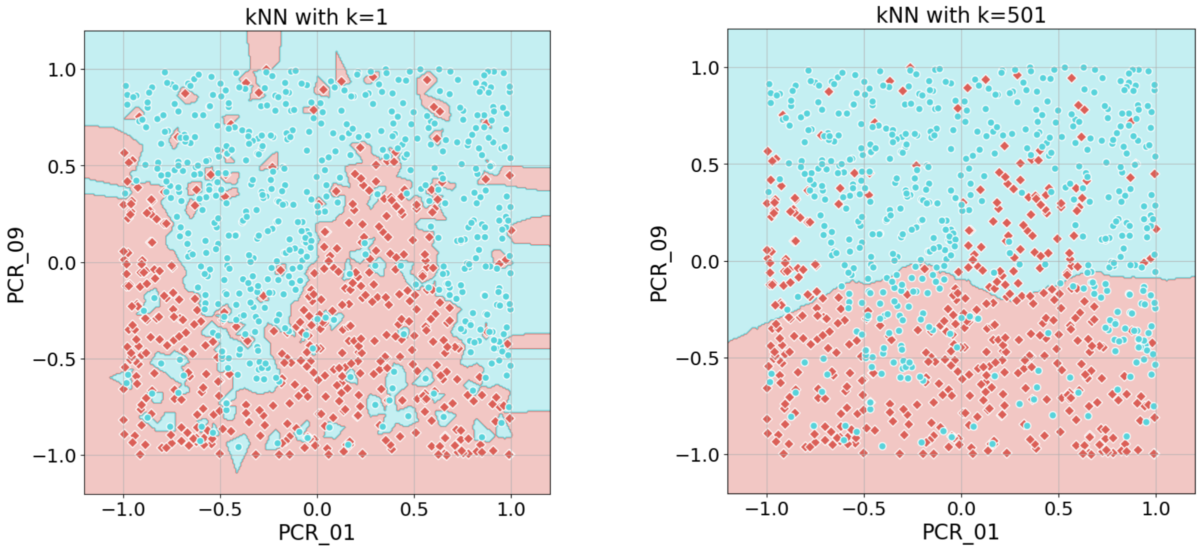
Q4

The score of the kNN module with k=7 is 0.872

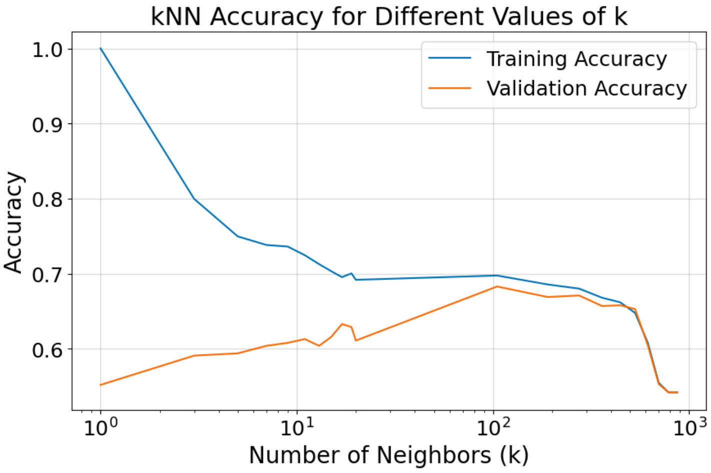


Q5

The visualization reveals distinct characteristics of the models based on the choice of k. The model with k=501 appears significantly smoother compared to the other model, but it is also observed to make numerous incorrect predictions. Conversely, when k=1, the model exhibits a larger number of blue gaps within the red region and red gaps within the blue region.



Q6



Based on the graph, the optimal value of k for validation accuracy is found to be 105, as opposed to 7 when only considering PCR_01 and PCR_09. However, the obtained training accuracy is 0.696 and the validation score is 0.683, both notably lower than the values mentioned in Q2 (0.884 and 0.871, respectively).

There are significant differences between the two graphs. Looking at the training accuracy, the curve of the kNN model using only PCR_01 and PCR_09 remains above 0.8 for k values slightly over 100. In contrast, the second graph shows the training accuracy falling below 0.8 before reaching $k = 3$. Additionally, while the validation accuracy of the first graph peaks at 0.87 and stays above 0.8 for k values between 3 and 105, the second graph only reaches a validation accuracy of 0.683.

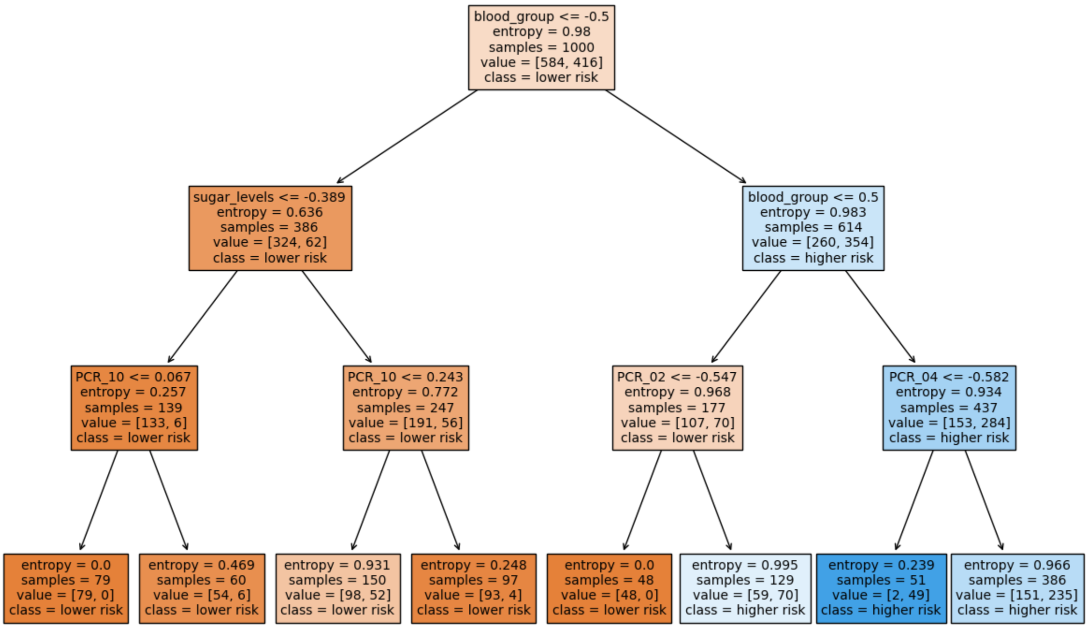
Both graphs exhibit some similarities around the value of 500 and above, where they experience underfitting as the validation accuracy dramatically declines. However, in the first graph, overfitting occurs until around $k = 3$, whereas in the second graph, overfitting is observed until approximately $k = 50$.

In our assessment, the disparity in results can be attributed to the consideration of all features, including less informative ones such as patient id, with equal weighting. Consequently, the impact of informative features like PCR_01 and PCR_02 is diminished. These features heavily influence the training set, resulting in even the best k value in the second graph (105) yielding a very low validation accuracy score. This suggests that the kNN model with such a large k value is likely too simplistic and lacks informative power.

Part 2: Decision trees

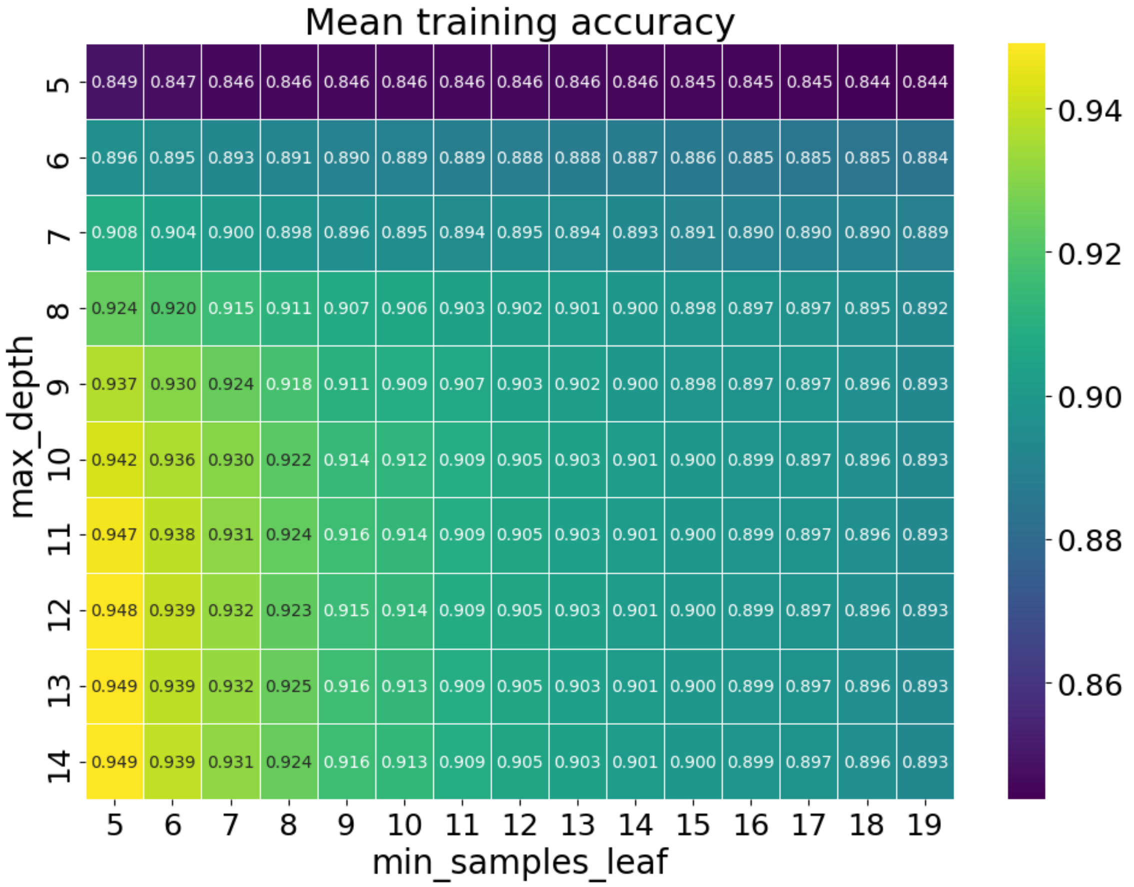
Q7

The training accuracy of of the decision tree is 0.726



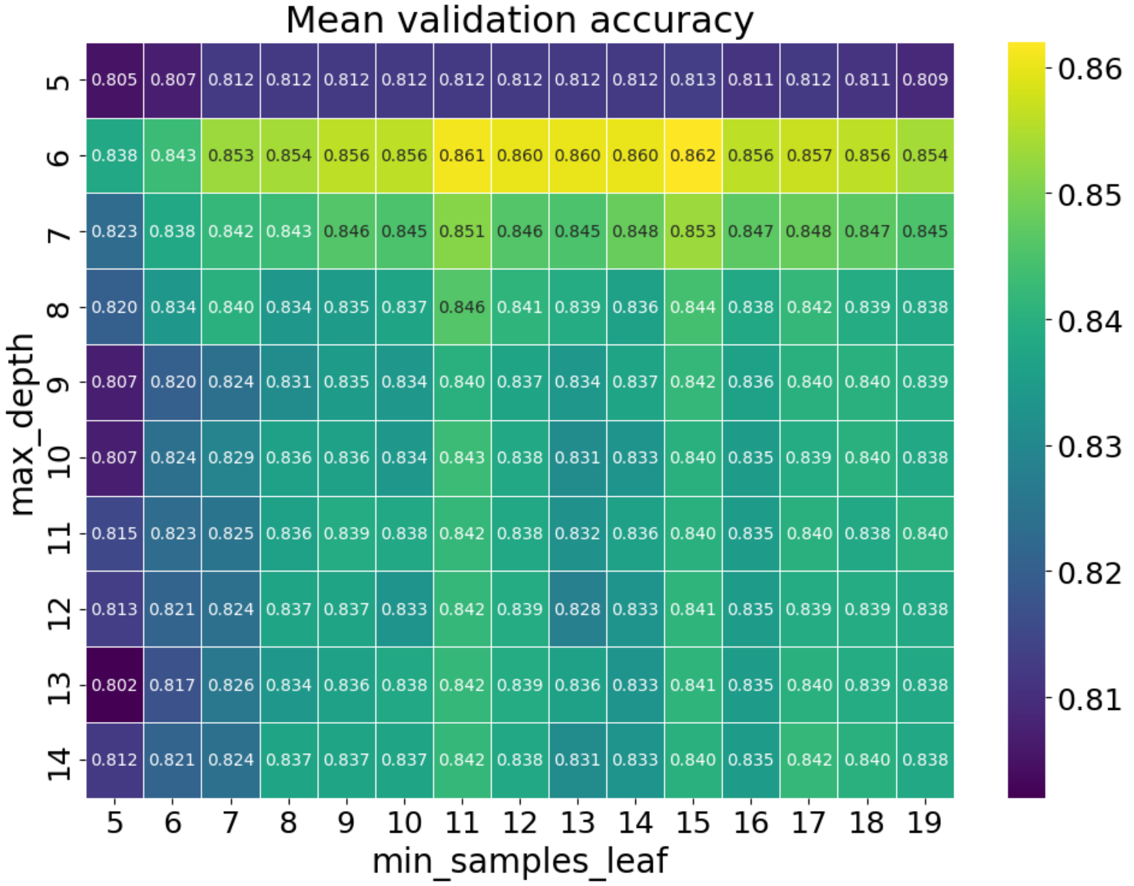
Decision Tree - ID3 with max_depth of 3

Q8



Mean training accuracy

max_depth \ min_samples_leaf	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
5	0.849	0.847	0.846	0.846	0.846	0.846	0.846	0.846	0.846	0.846	0.845	0.845	0.845	0.844	0.844
6	0.896	0.895	0.893	0.891	0.890	0.889	0.889	0.888	0.888	0.887	0.886	0.885	0.885	0.885	0.884
7	0.908	0.904	0.900	0.898	0.896	0.895	0.894	0.895	0.894	0.893	0.891	0.890	0.890	0.890	0.889
8	0.924	0.920	0.915	0.911	0.907	0.906	0.903	0.902	0.901	0.900	0.898	0.897	0.897	0.895	0.892
9	0.937	0.930	0.924	0.918	0.911	0.909	0.907	0.903	0.902	0.900	0.898	0.897	0.897	0.896	0.893
10	0.942	0.936	0.930	0.922	0.914	0.912	0.909	0.905	0.903	0.901	0.900	0.899	0.897	0.896	0.893
11	0.947	0.938	0.931	0.924	0.916	0.914	0.909	0.905	0.903	0.901	0.900	0.899	0.897	0.896	0.893
12	0.948	0.939	0.932	0.923	0.915	0.914	0.909	0.905	0.903	0.901	0.900	0.899	0.897	0.896	0.893
13	0.949	0.939	0.932	0.925	0.916	0.913	0.909	0.905	0.903	0.901	0.900	0.899	0.897	0.896	0.893
14	0.949	0.939	0.931	0.924	0.916	0.913	0.909	0.905	0.903	0.901	0.900	0.899	0.897	0.896	0.893



Mean validation accuracy

max_depth \ min_samples_leaf	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
5	0.805	0.807	0.812	0.812	0.812	0.812	0.812	0.812	0.812	0.812	0.813	0.811	0.812	0.811	0.809
6	0.838	0.843	0.853	0.854	0.856	0.856	0.861	0.860	0.860	0.860	0.862	0.856	0.857	0.856	0.854
7	0.823	0.838	0.842	0.843	0.846	0.845	0.851	0.846	0.845	0.848	0.853	0.847	0.848	0.847	0.845
8	0.820	0.834	0.840	0.834	0.835	0.837	0.846	0.841	0.839	0.836	0.844	0.838	0.842	0.839	0.838
9	0.807	0.820	0.824	0.831	0.835	0.834	0.840	0.837	0.834	0.837	0.842	0.836	0.840	0.840	0.839
10	0.807	0.824	0.829	0.836	0.836	0.834	0.843	0.838	0.831	0.833	0.840	0.835	0.839	0.840	0.838
11	0.815	0.823	0.825	0.836	0.839	0.838	0.842	0.838	0.832	0.836	0.840	0.835	0.840	0.838	0.840
12	0.813	0.821	0.824	0.837	0.837	0.833	0.842	0.839	0.828	0.833	0.841	0.835	0.839	0.839	0.838
13	0.802	0.817	0.826	0.834	0.836	0.838	0.842	0.839	0.836	0.833	0.841	0.835	0.840	0.839	0.838
14	0.812	0.821	0.824	0.837	0.837	0.837	0.842	0.838	0.831	0.833	0.840	0.835	0.842	0.840	0.838

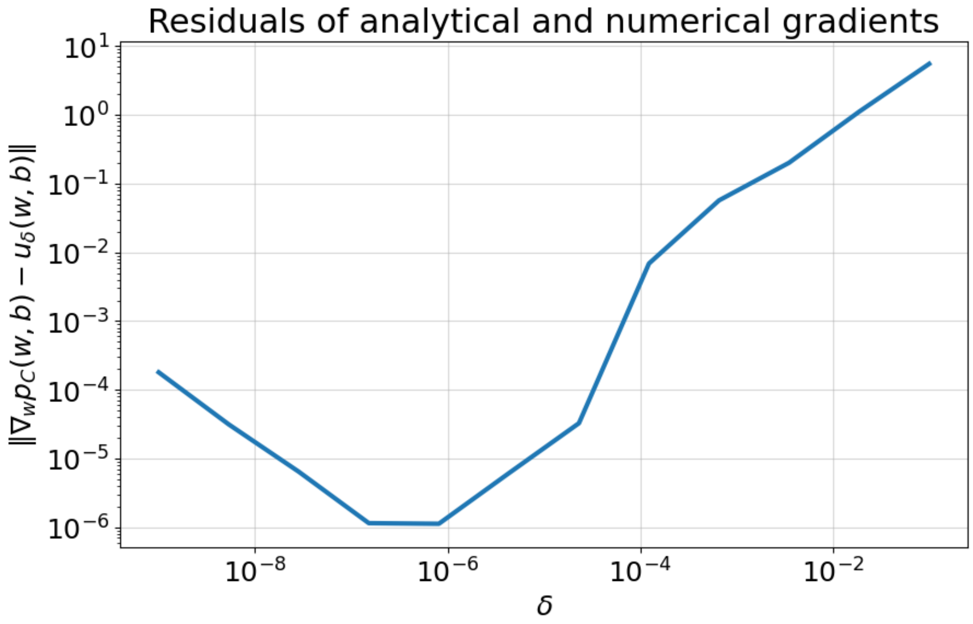
c. The plots indicate that the optimal hyperparameters are max_depth = 6 and min_samples_leaf = 15.
d. When the max_depth is less than 5, irrespective of the min_samples_leaf, the plots demonstrate underfitting.
e. The plots reveal overfitting when the min_samples_leaf is less than 7 and the max_depth is greater than 9.
f. In terms of underfitting, our observation suggests that when a decision tree has a very low maximum depth, it may struggle to distinguish complex data with numerous parameters that are not easily separable. Conversely, when the tree has a high depth without strong constraints on the minimum number of samples in a leaf, it becomes excessively tailored to the data, leading to overfitting.

Q9

The test accuracy of the model using the optimal hyperparameters is 0.887.

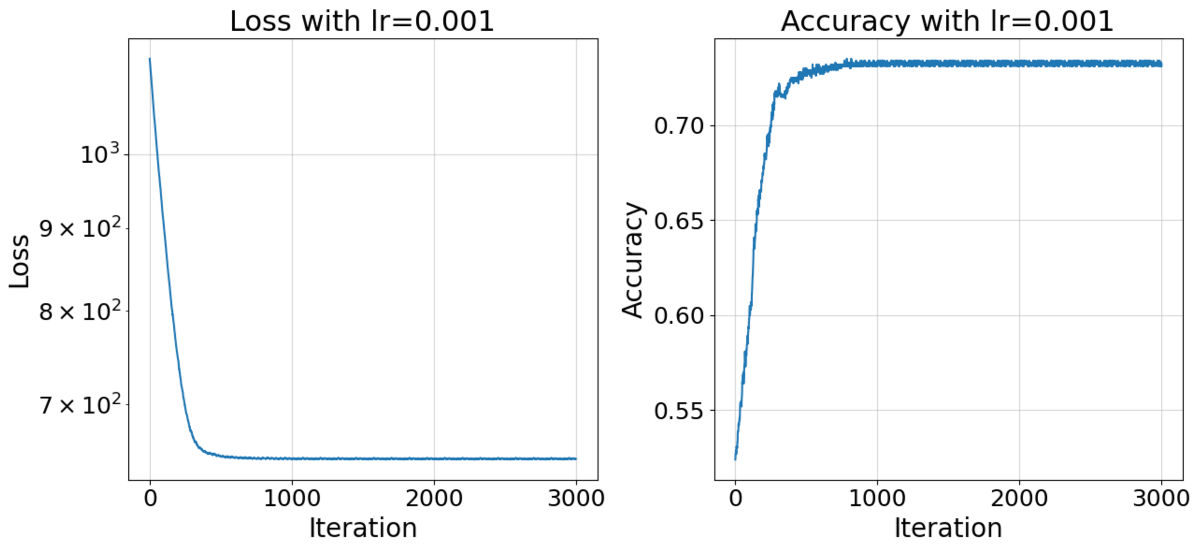
Part 3: Linear SVM and the Polynomial kernel

Q10

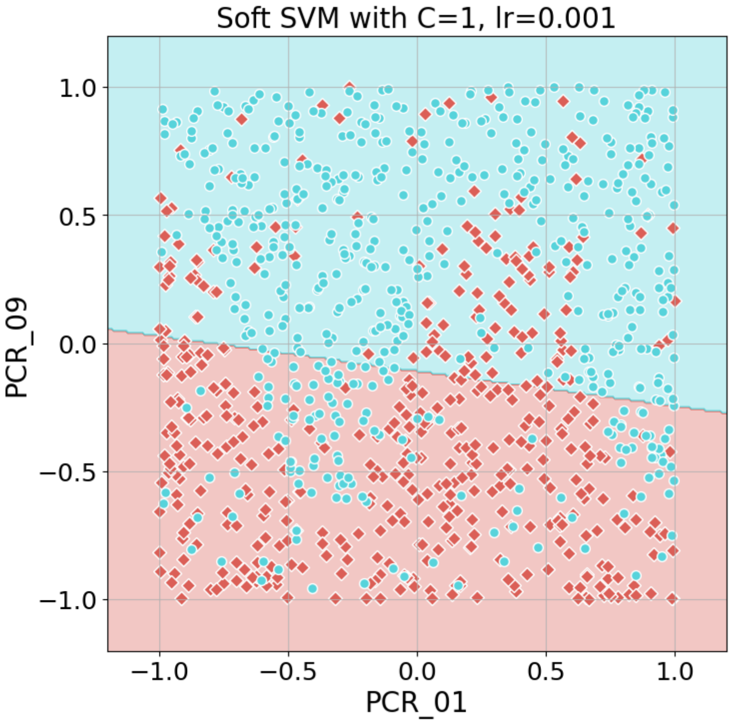


The graph's behavior is logical as it aligns with the nature of numerical approximations. When the value of δ is excessively large, the numerical approximation becomes less accurate due to the derivative's definition. On the other hand, as δ decreases to smaller values, the numerical approximation approaches the actual derivative more closely. However, when δ becomes too small, there are computational errors that arise from performing calculations on a computer.

Q11



a. We chose learning rate = 0.001 with $C = 1$.



b.

c. We got the following results:
   - The maximum accuracy is 0.735 and we got it in the 523th iteration.
   - The minimum loss is 646.495 and we got it in the 1122th iteration.

As evident, the achievement of high accuracy and low loss does not occur simultaneously during each step. This can be attributed to the fact that while accuracy is determined by comparing the prediction to the corresponding label for each data point, the loss quantifies the predicted accuracy for a new set of data. Ideally, we aim for a classifier with a high margin, which means even with 100% accuracy, the margin could still be small, resulting in a high loss.

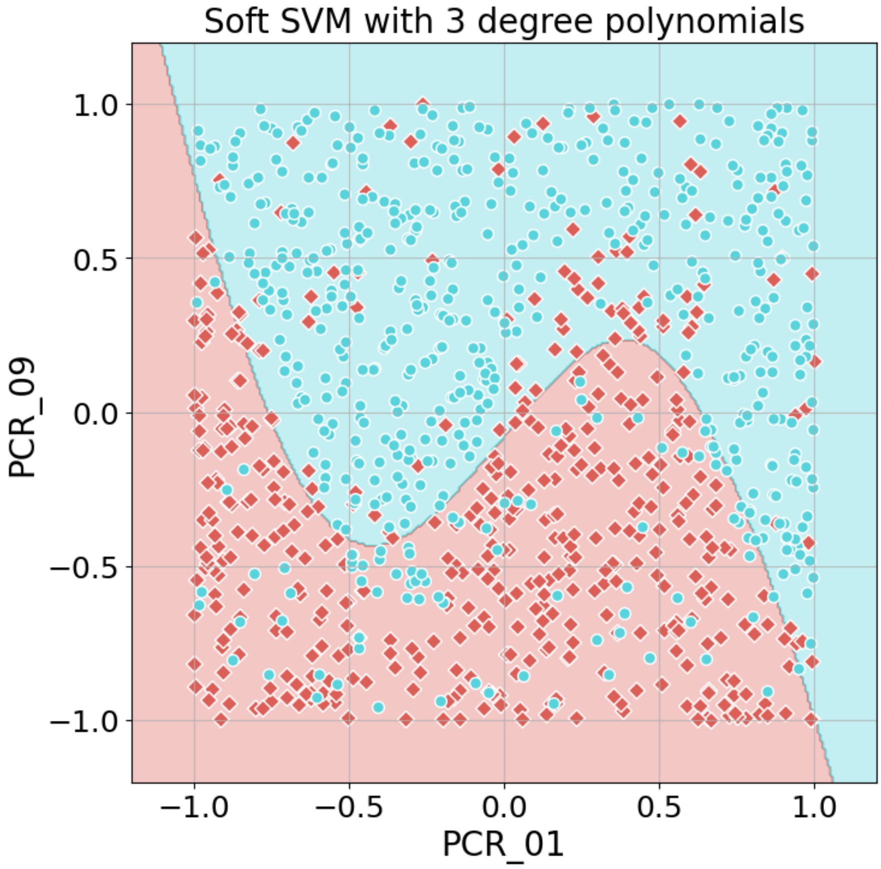
Q12

If we divide the learning rate by 10, the optimization algorithm (e.g., SGD) to take smaller steps during each iteration. That will lead to slower convergence speed but it may lead to a better solution (it may help fine-tune the solution).

If we multiply C by 10, we assigning more importance to the sum of the errors, mining less, emphasis on the margin. This change will make the model less sensitive to misclassification and it will lead to a model that fit the training data more closely. So it will reach a different solution.

Q13

a. The training score of SVM with 3 degree polynomials is 0.817
   The test score of SVM with 3 degree polynomials is 0.816



b.

c. The second model, in contrast to the previous one, demonstrates improved separation of regions. For instance, in the first model, the region between 0.0 and -0.5 in both features is primarily labeled as red. However, in the second model, this region is classified in a way that more accurately reflects the data because it is much more flexible.

Q14

a. The accuracies are 0.818, 0.821, 0.837, 0.819, 0.836
   The mean is 0.826 and the std is 0.008.

b. There are multiple factors contributing to the variation in our results.
   Firstly, during each iteration, the Stochastic Gradient Descent (SGD) algorithm randomly selects the data points to calculate gradients. This randomness causes slight differences in the results between runs since SGD completes after a specified number of iterations. Consequently, we do not achieve the global minimum, but

rather a point close to it, influenced by the stochastic nature of the problem. Additionally, the initialization of the parameter 'w' affects how close we start to the global minimum. Therefore, due to this initialization variation, we obtain different results. Furthermore, when employing a 3-degree polynomial as the kernel function, the problem may loses its convexity. Consequently, a global minimum no longer exists, and SGD converges to a local minimum instead. This local minimum can differ with each run.

Part 4: The RBF kernel

Q15

We need to prove that

$$\lim_{\gamma\to\infty}\left(\operatorname{sign}\left(\sum_{i\in[m],\alpha_i>0}\alpha_i y_i \exp\{-\gamma\|x-x_i\|\}\right)\right)=y_{i^*}$$

where $i^* = \operatorname{argmin}_{i\in[m],\alpha_i>0}\|x-x_i\|_2^2$

According to the information provided in the blog post, when the value of γ is sufficiently large, there is a significantly higher emphasis on the importance of x_{i^*}, compared to other data points. This implies that as the distance between data points increases, the influence of the RBF Kernel diminishes exponentially. In the case of a very large γ, the classification decision is primarily determined by y_{i^*}.

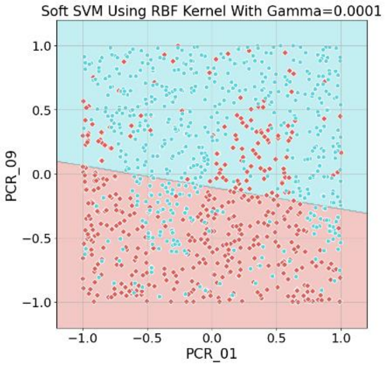
$$\lim_{\gamma\to\infty}\left(\operatorname{sign}\left(\sum_{i\in[m],\alpha_i>0}\alpha_i y_i \exp\{-\gamma\|x-x_i\|\}\right)\right)$$

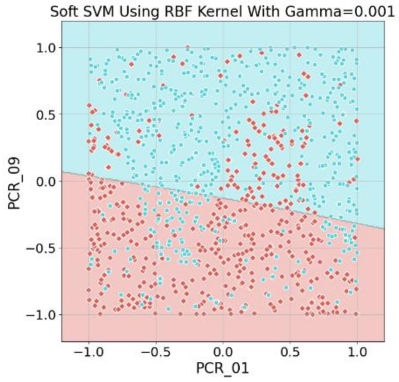
$$=\lim_{\gamma\to\infty}\operatorname{sign}\left(\alpha_{i^*}y_{i^*}\exp\{-\gamma\|x-x_{i^*}\|\}+\sum_{i^*\neq i\in[m],\alpha_i>0}\alpha_i y_i \exp\{-\gamma\|x-x_i\|\}\right)$$

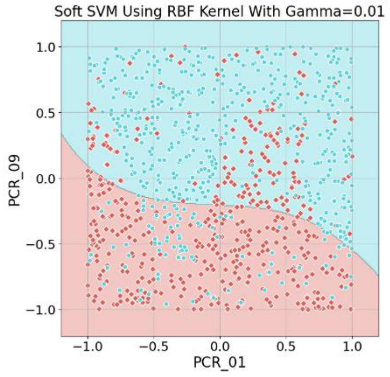
$$=\lim_{\gamma\to\infty}\operatorname{sign}\left(\alpha_{i^*}y_{i^*}\underbrace{exp\{-\gamma\|x-x_{i^*}\|\}}_{>0}+\epsilon(\gamma)\right)=\operatorname{sign}(\alpha_{i^*}y_{i^*})=y_{i^*}$$

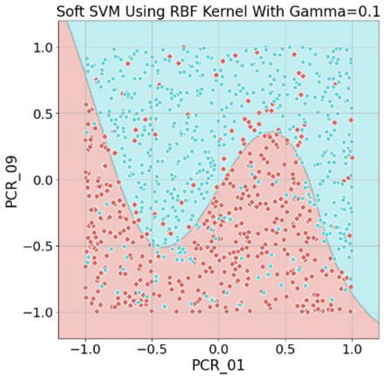
In the previous paragraph, it is mentioned that when γ is sufficiently large, the sum becomes negligible. In this case, the sign of the expression $\alpha_{i^*}y_{i^*}\exp\{-\gamma\|x-x_{i^*}\|\}$ is solely influenced by the sign of y_{i^*}. Regardless of the specific value of γ, the argument's sign always corresponds to y_{i^*}, leading to the conclusion that the limit is determined by the sign of y_{i^*}.

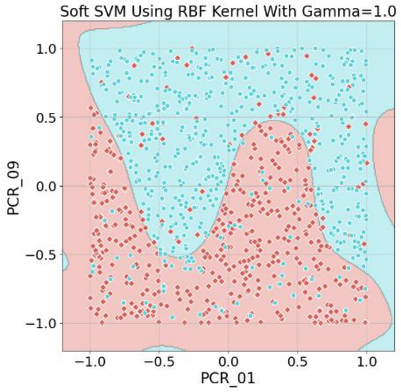
Q16

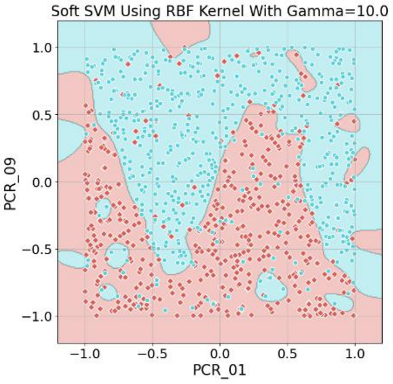

Soft SVM Using RBF Kernel With Gamma=0.0001
PCR_09
PCR_01
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0

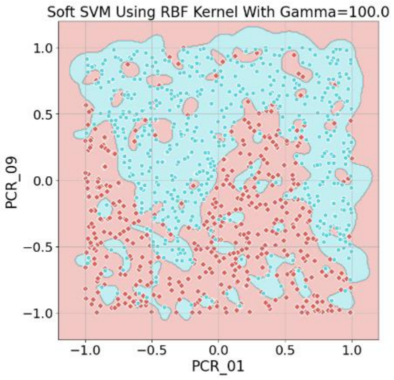

Soft SVM Using RBF Kernel With Gamma=0.001
PCR_09
PCR_01
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0

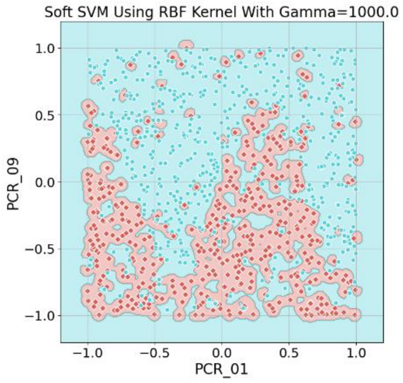

Soft SVM Using RBF Kernel With Gamma=0.01
PCR_09
PCR_01
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0

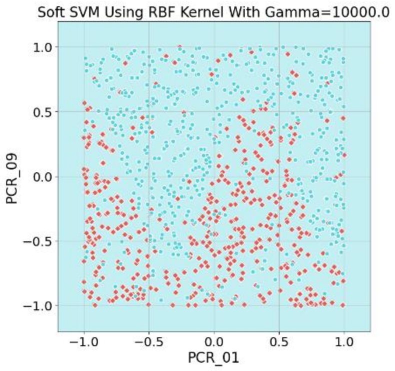

Soft SVM Using RBF Kernel With Gamma=0.1
PCR_09
PCR_01
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0


Soft SVM Using RBF Kernel With Gamma=1.0
PCR_09
PCR_01
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0

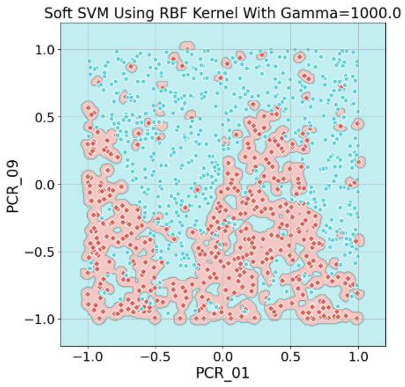

Soft SVM Using RBF Kernel With Gamma=10.0
PCR_09
PCR_01
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0

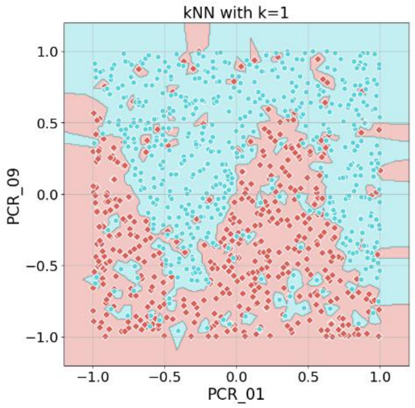

Soft SVM Using RBF Kernel With Gamma=100.0
PCR_09
PCR_01
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0


Soft SVM Using RBF Kernel With Gamma=1000.0
PCR_09
PCR_01
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0


Soft SVM Using RBF Kernel With Gamma=10000.0
PCR_09
PCR_01
1.0
0.5
0.0
−0.5
−1.0
−1.0
−0.5
0.0
0.5
1.0

Q17



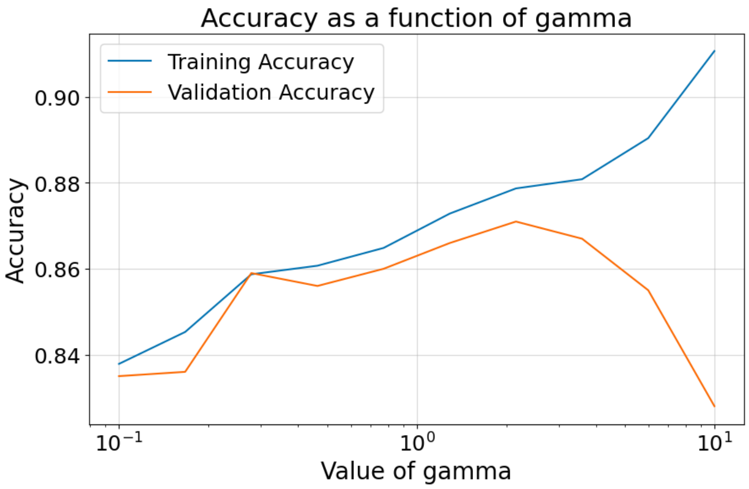


Although we have demonstrated in Q15 that as γ approaches infinity, the SVM with RBF kernel becomes similar to kNN with k=1, there are still significant differences between them. The SVM with RBF kernel and $\gamma = 10^4$ exhibits smoothness, whereas kNN with $k = 1$ does not.

Additionally, the SVM model exhibits distinct circular decision regions around each red point, while kNN lacks this feature. This dissimilarity arises because kNN assigns labels solely based on the nearest neighbor, even if it is "distant" in terms of L_2 distance (as observed in the bottom left corner). In contrast, the SVM model with a very high γ value ($\gamma = 10^4$) forms circular decision boundaries around each point, with smaller radii for larger γ values. Regions outside these circles are labeled as blue by default.

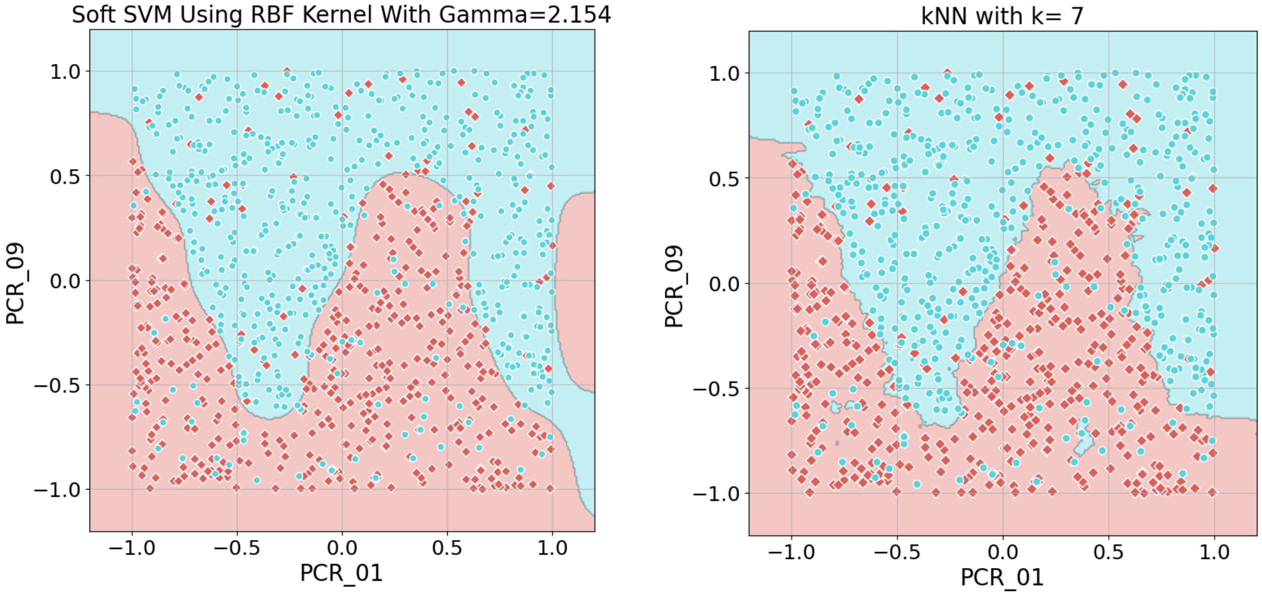
The kNN model appears to be more sensitive to outliers, as outliers have a greater influence on the surrounding area of the point. For instance, consider the area in the box of PCR_09 above 0.8 and PCR_01 between -0.5 and 0. In the kNN model, the surrounding area for these points is larger and more affected by outliers, whereas in the SVM model, the area is smaller and less sensitive to outliers. This behavior stems from the same reason as before: the RBF model assigns labels within a small circle around each point, while the kNN model assigns labels to the entire area.

Q18



The graph above demonstrates that the optimal value for gamma is 2.154. The corresponding mean train score is 0.879, and the validation score is 0.871. To identify the best γ value, our search was limited to the range $[10^{-1}, 10]$. Therefore, the graph does not provide an example of extreme underfit. However, through testing in 2 folds, we discovered that values below 10^{-1} result in underfitting. Similarly, values above 4 lead to overfitting. Underfitting occurs with values below 10^{-1} because the points considered are too distant from the margin, resulting in an overly simplistic model for data classification. On the other hand, for values above 4, the model incorporates edge cases that negatively impact validation performance. The graph clearly illustrates a deterioration in the validation score while the training score remains close to 1.

Q19



The SVM with RBF kernel achieves a test score of 0.876, whereas the kNN test score is 0.872. Solely considering these scores, the SVM with RBF kernel model outperforms kNN by a small margin. Additionally, the SVM with RBF kernel exhibits a notably smoother behavior, suggesting that it may possess better generalization capabilities for the data.