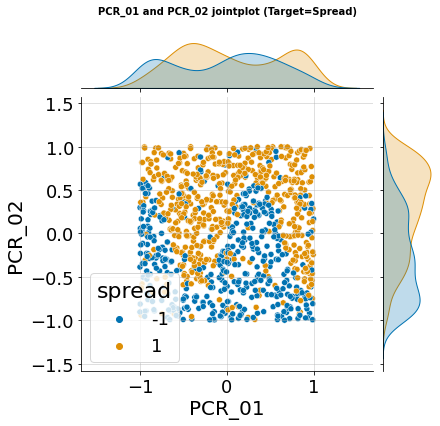
Major HW 1 – Data Exploration and Preparation

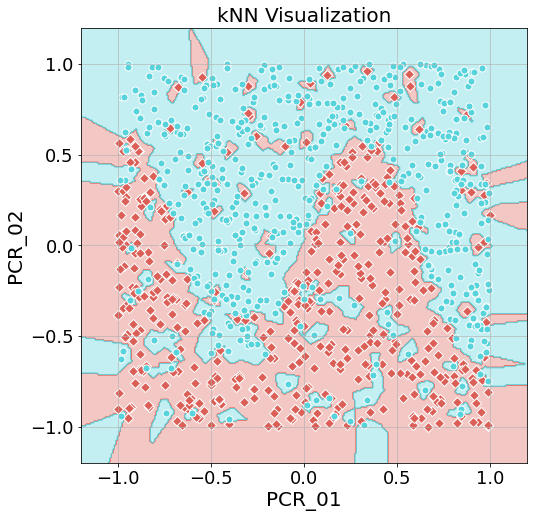
Orad Barel, 311288203, [oradbarel@campus.technion.ac.il](mailto:oradbarel@campus.technion.ac.il)

Ofir Manor, 316084623, [ofir.manor@campus.technion.ac.il](mailto:ofir.manor@campus.technion.ac.il)

**Q1.**



**Q2.**



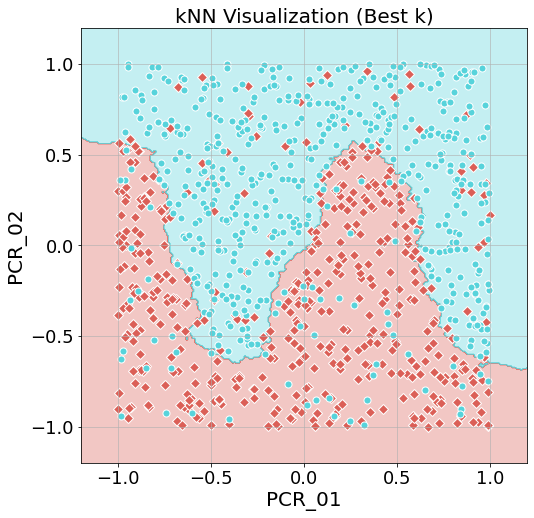
**Q3.**



The best value we found while doing the k-cross validation (which we interpreted as the value which led to the highest mean validation score as it best simulates the accuracy on unseen data) is 19, with a mean training accuracy of 0.863 and validation accuracy of 0.859.

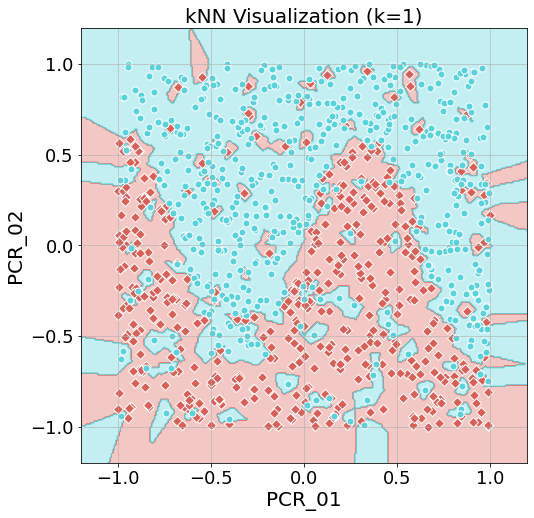
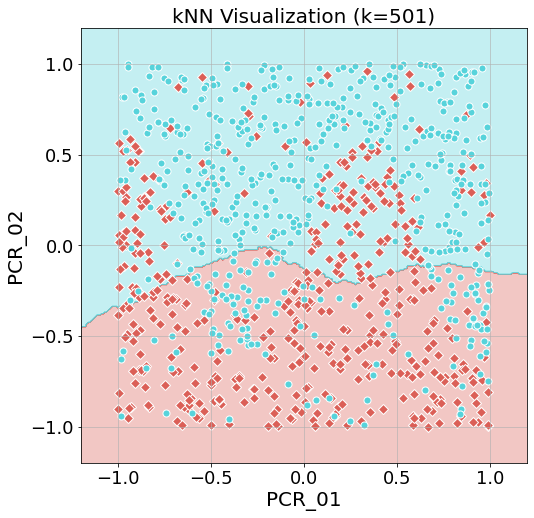
Lower values, such as 1 and 3, cause overfitting on the training data, as seen by the high training accuracy and low validation accuracy. This is cause by the minimal amount of neighbors taken into account. When looking at a small amount of neighbors, and testing on the training data, the original data point (whose Euclidian distance from itself is 0) is given great weight () of the decision, and thus the model easily classifies them. But as to unseen data, the model does not take into account the more general trend of the area surrounding (in Euclidian distance) of the tested datapoint, as only a few training datapoints are taken into account.

Higher values, such as those above 100, cause underfitting, as seen by the downwards trend of the accuracy score of both the training accuracy and validation accuracy. This is caused by the large amount of training datapoint taken into consideration when deciding the label of the tested datapoint. The decision area slowly morphs into 2 continuous area (one for each label), only looking at where the vast majority of each label’s training datapoints lie, without taking into consideration any nuances.

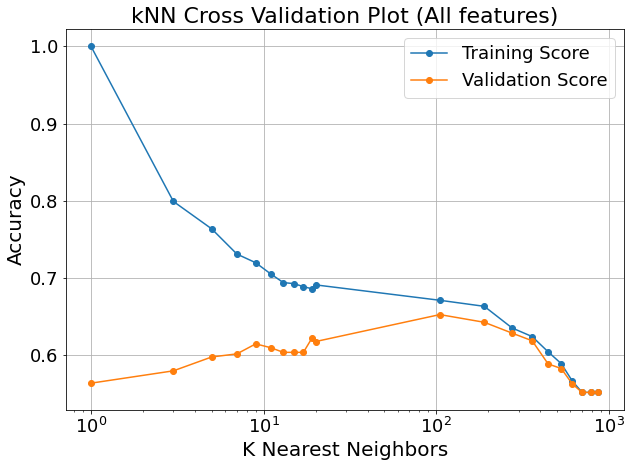
**Q4.**

The test accuracy is 0.912

**Q5.**

****

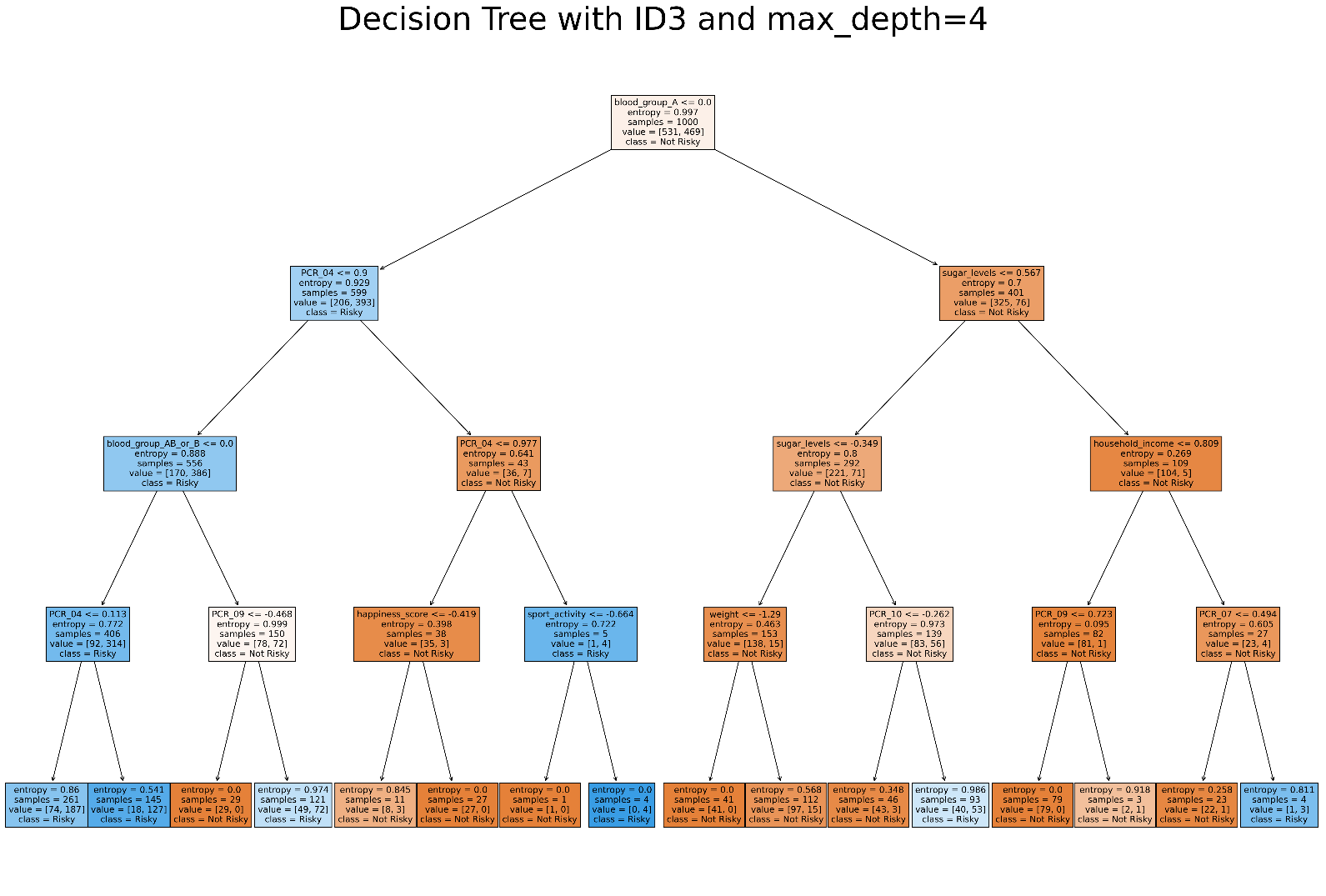
We find that for , as opposed to the optimal we found, the boundaries consist of many smaller decision areas, and while the training accuracy is 1, the test accuracy is 0.823 (lower than the optimal ), these are all signs of overfitting. As for , we find that the boundary between the decision area if far closer to a straight line for the optimal , and that both the training and test accuracies are lower than the optimal , with the test accuracy being 0.764, sign pointing to underfitting.

**Q6.**

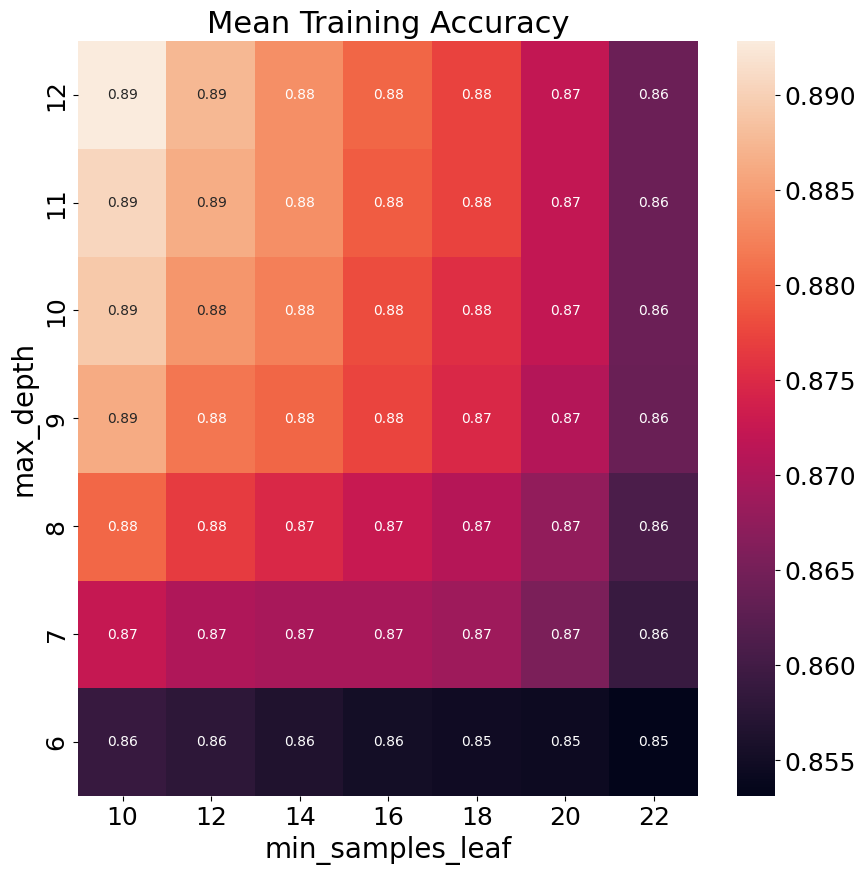
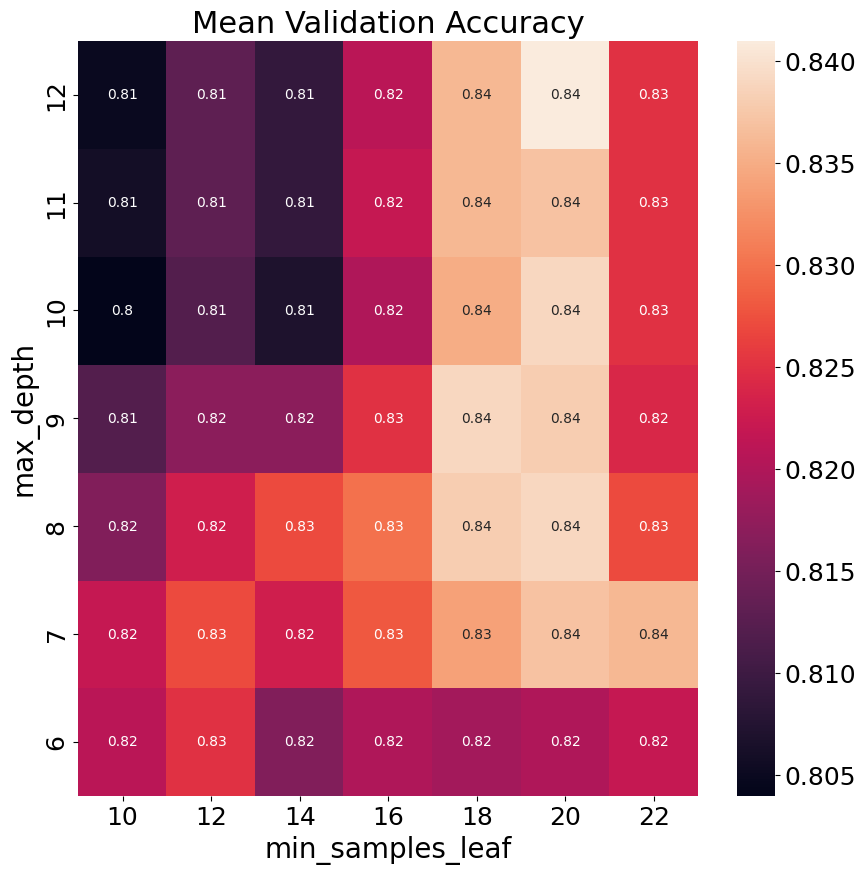
Here we see the training and validation accuracy curves when considering all features. We found that the optimal is 105 and that the training accuracy of that is 0.67 while the validation accuracy is 0.652. These are both considerably lower than when we used only PCR\_01 and PCR\_02. To us, this is logical considering how kNN works and why we originally chose PCR\_01 and PCR\_02.

When we chose those two features in the previous homework assignment, we saw that they divided the datapoints marked with and those marked with into nearly two homogonous areas divided by a sinuous line (as can be seen in the joint plot in Q1). Seeing as kNN works off the Euclidian distance between a datapoint to all other datapoints the model was trained on, this meant that, assuming the training data is indictive of the validation data, most datapoints would be found not far from other datapoints with the same label (in Euclidian distance). Thus, the results were relatively high.

When we now considered all features, we have no such promise that they can be divided into mostly homogenous areas based on the label. The different features might spread the data differently, thus causing the general area of each datapoint to be filled with datapoint of the opposite label. Hence, the accuracy is far lower when considering all features, and not just those specifically chosen.

**Q7.**

**Q8.**



We found that the optimal hyperparameter combination is max\_depth=12 and min\_samples\_leaf=20 achieving a validation accuracy of roughly 0.84.

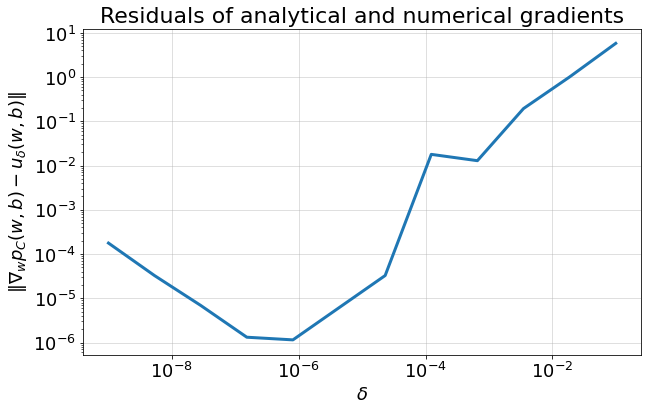
For the combination max\_depth=6 and min\_samples\_leaf=22 we found that the model is underfitting on the training data. This is because these hyperparameters do not allow the tree to be as expressive as others, creating very generalized decisions in its smaller hypothesis class.

For the combination max\_depth=12 and min\_samples\_leaf=10 we found that the model is overfitting on the training data. This is because we allowed the resulting tree to be very expressive (greater depth means the ability to utilize more features or further divide a feature, and smaller minimum number of samples allows leaves to represent less training datapoint). All of this allows for great accuracy on the training data, while producing decisions using features or parts if feature spaces which do not well represent the probability.

**Q9.**

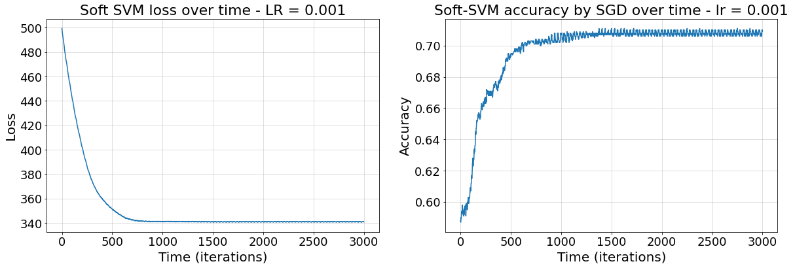
The test accuracy of our model using the optimal hyperparameter we found (reported in Q8) is 0.88

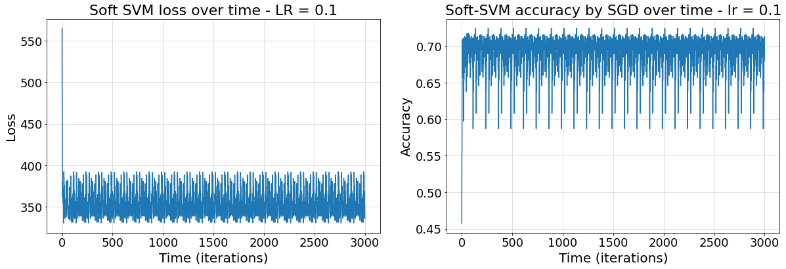
**Q10.**



We can see that from a certain point the larger is the larger the residuals are.

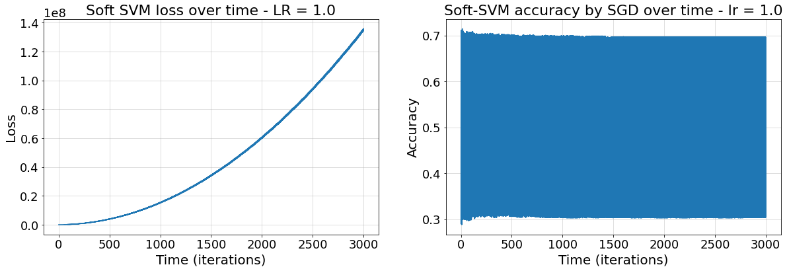
The larger becomes, the farther away we are from the definition of derivative, which states that , thus the numerical gradient becomes less accurate, while the analytical gradient continues to comply to the rules we learned in calculus and vector analysis. And similarly, for small values, we get small residuals too.

**Q11.**



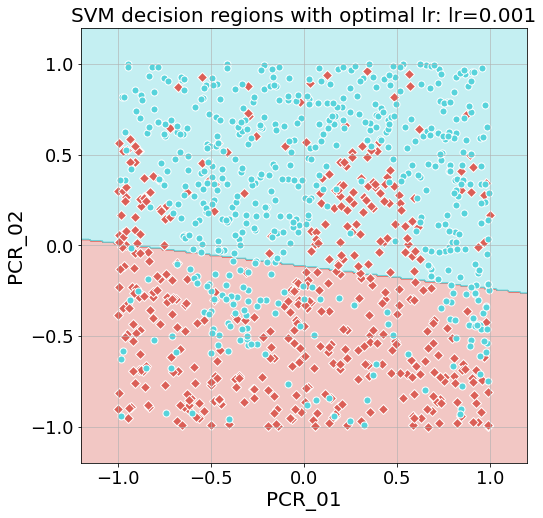
תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי



a) Denote that we set C = 1, since that for C = 0.1, we get the same clasification (same label) for all the train samples (it can be seen on the plot of the decision regions with visualize\_clf). Therefore, we increased a bit the `C` value.

As we can see, we converge the best (in terms of speed and stability) for `lr = 0.001`. Also, the best accuracy is for 0.001 and 0.01.

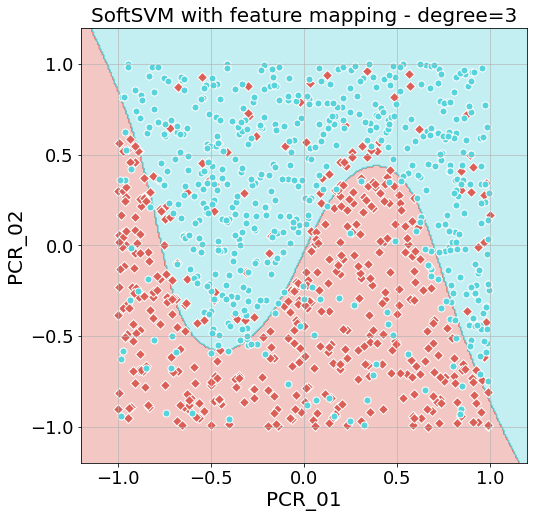
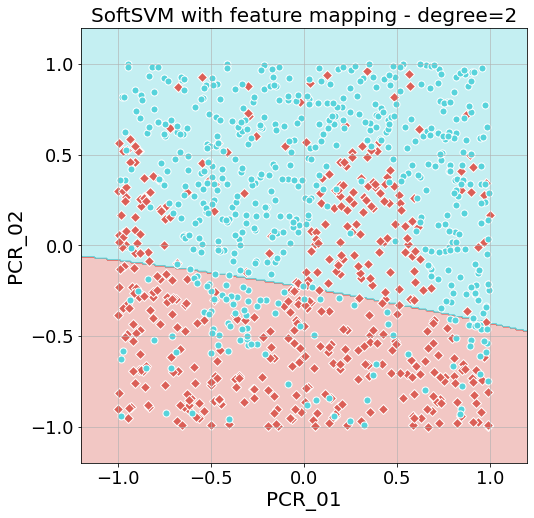
b)

As we can see, the maximum accuracy is 0.721 on step 563. The minimum loss is 340.7832 on step 1769. Are they attained at the same step? If so – must it be so? If not – how is it possible? Need to explain why.

**Q12.**

a) Using polynomial feature mapping with degree=2, we get a training accuracy of 0.704 and a test accuracy of 0.796.

Using polynomial feature mapping with degree=3, we get a training accuracy of 0.836 and a test accuracy of 0.884.

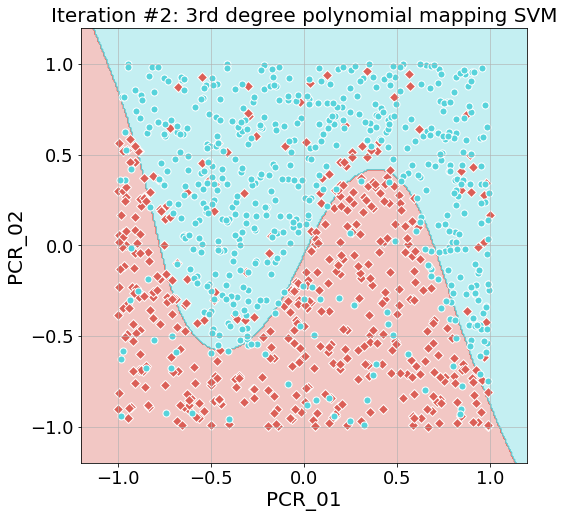
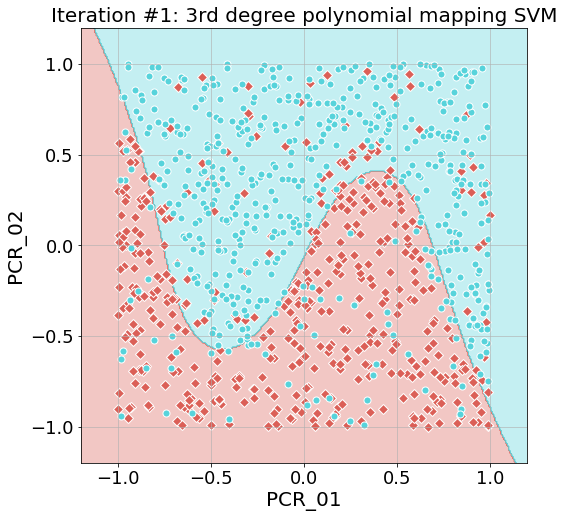
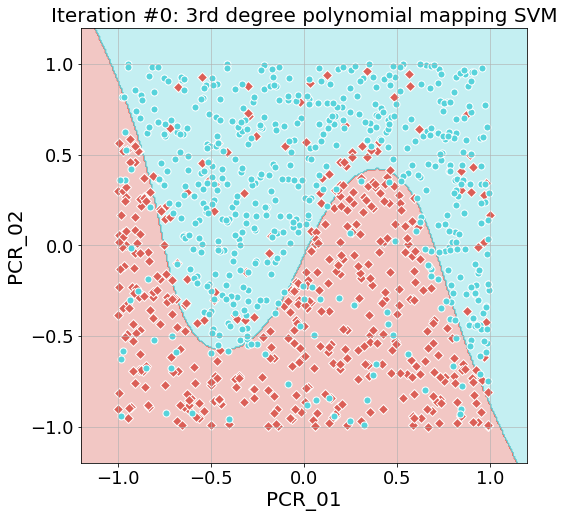
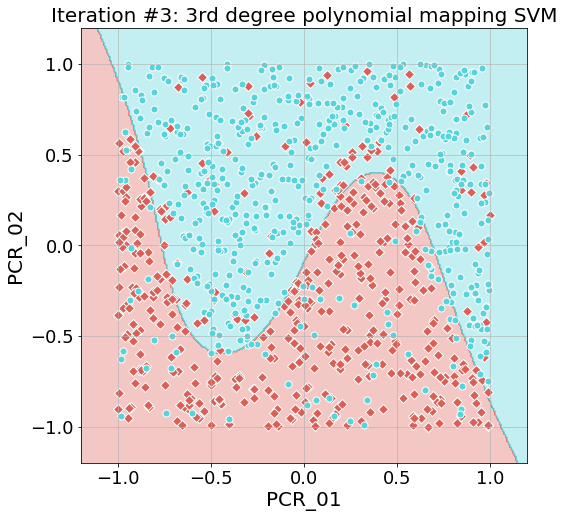
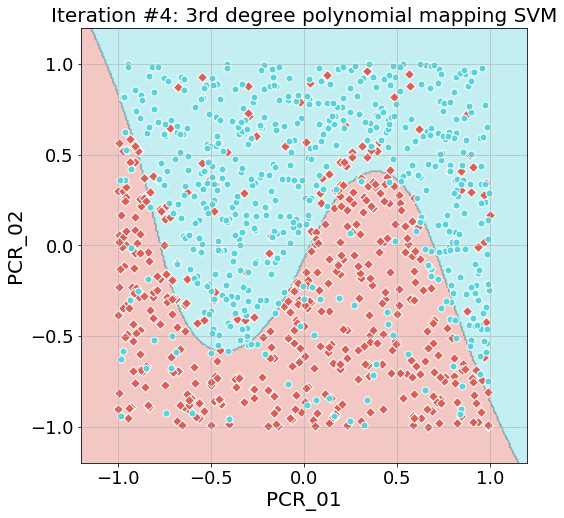
b)

c) The difference in the decision boundaries that we see is due to the fact that a 2nd degree polynomial has a parabolic shape (one extremum) while the 3rd degree polynomial has more 'flexible' shape (two extrema points). Therefore, the 3rd degree polynomial feature mapping can fit better to our data than the 2nd degree polynomial feature mapping.

**Q13.**

1. The accuracies are as following (in running order):

Their mean: . Their standard division: .

1. Following are the plots:
2. What is the reason to the variety?

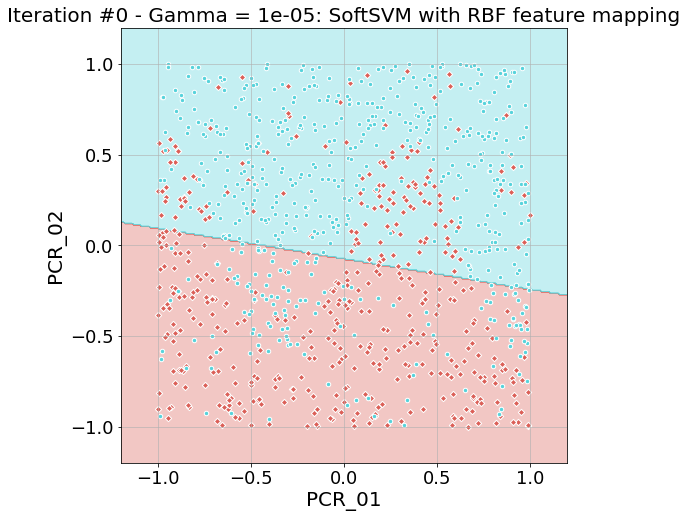
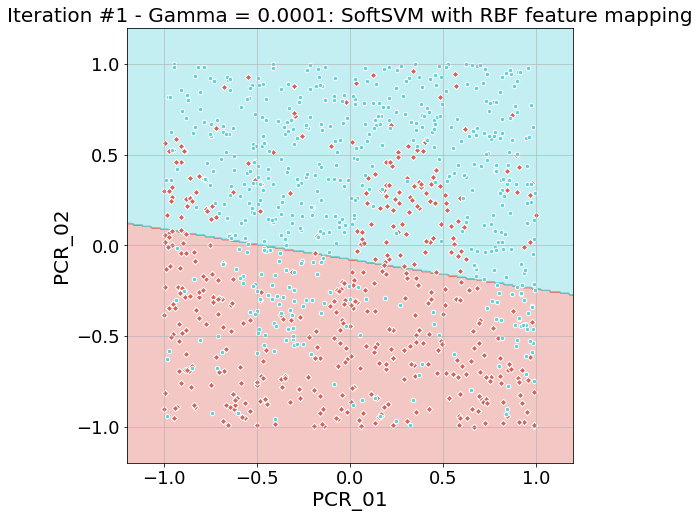
**Q14.**

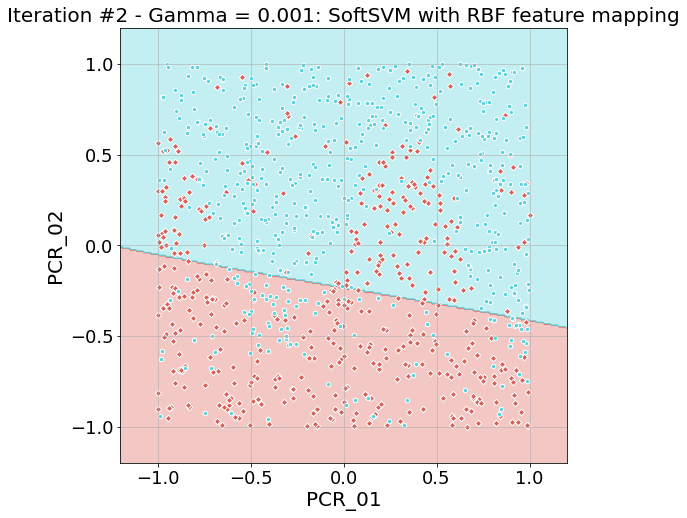
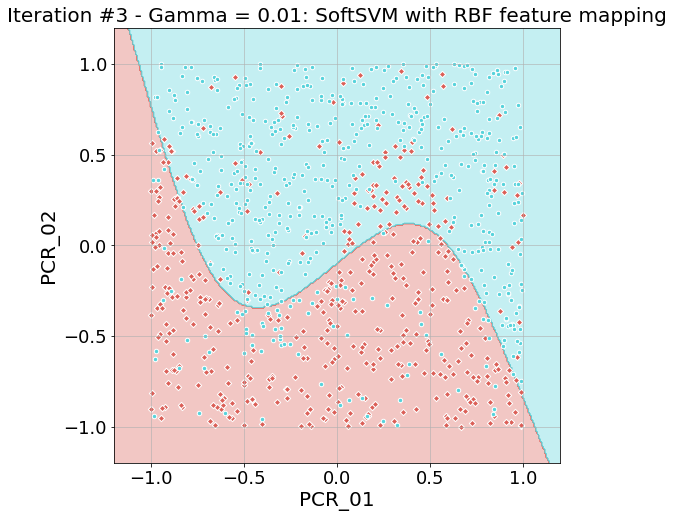
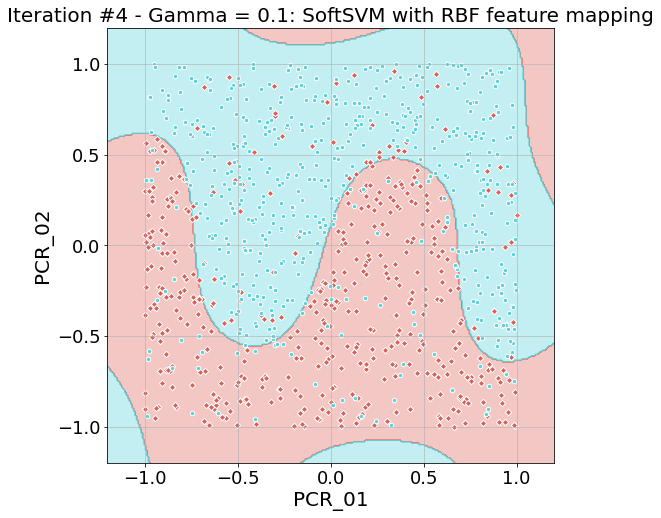
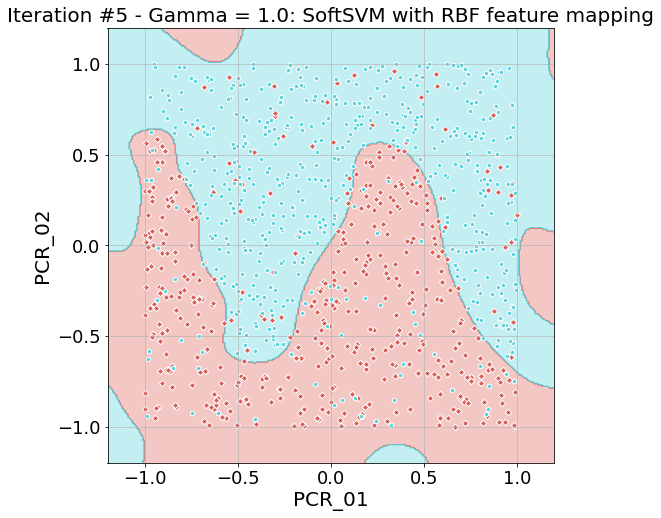
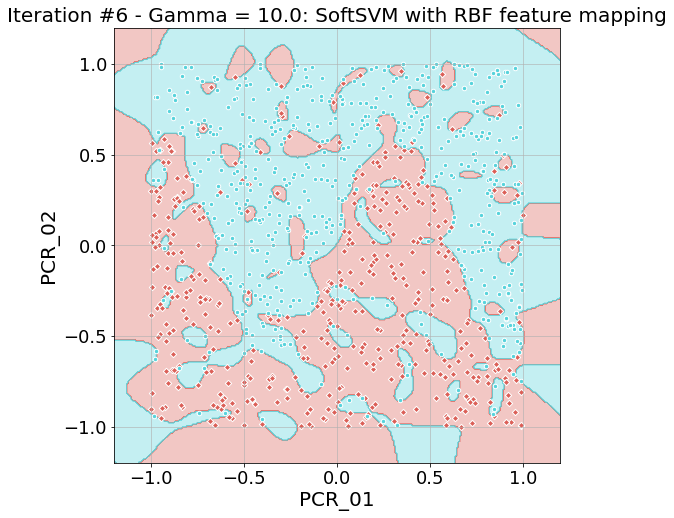
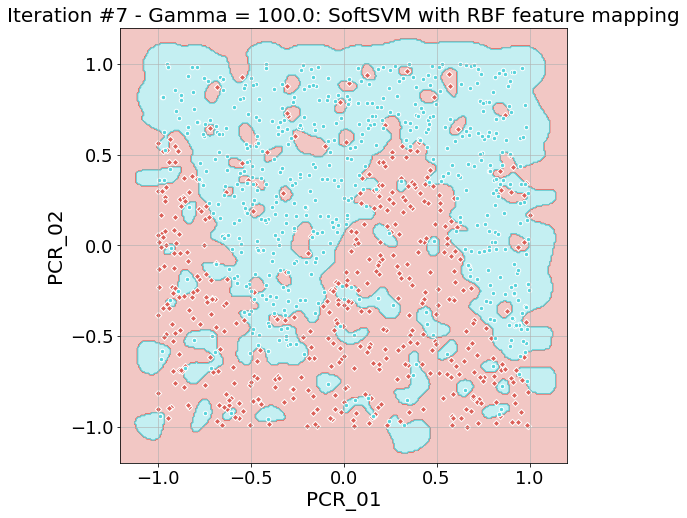
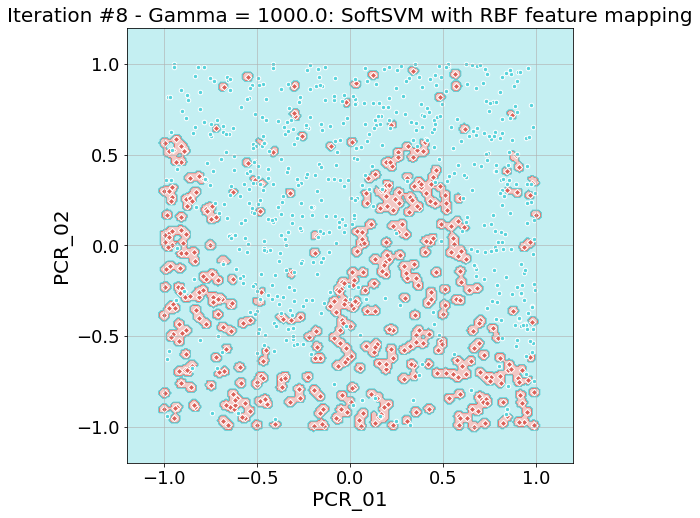
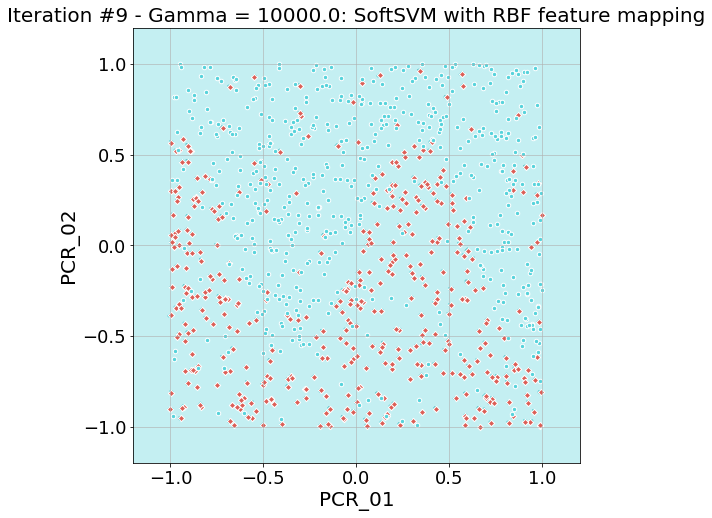
Let . Thus (and because we can assume ) , for each : .

Thus, we get that:

Now, let us complete the derivation:

**Q15.**

****Following are the plots:

****