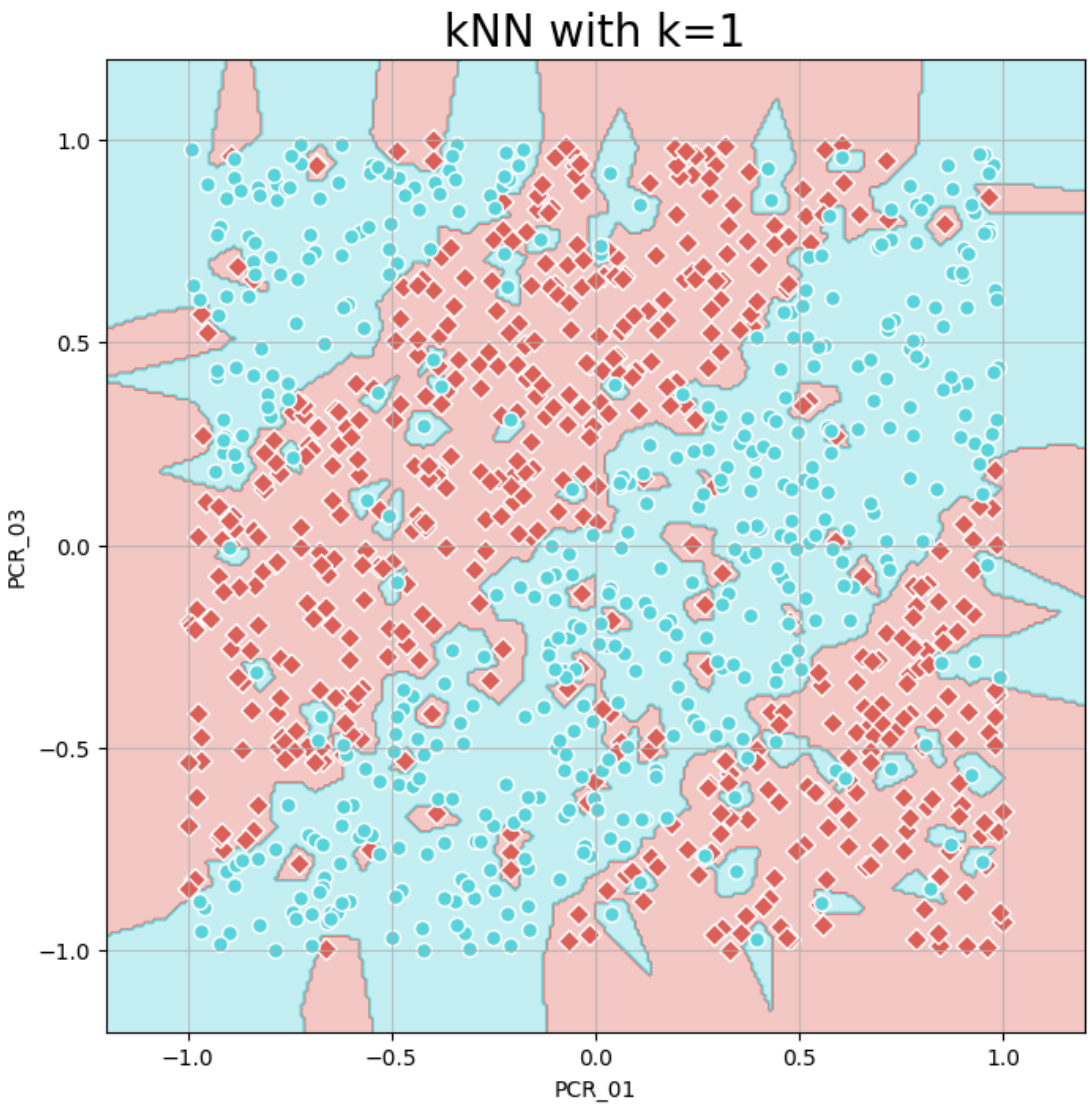
Intro to machine learning

$MHW2$ 325172351 – 318837317

Part 1: Basic model selection with k-Nearest Neighbors

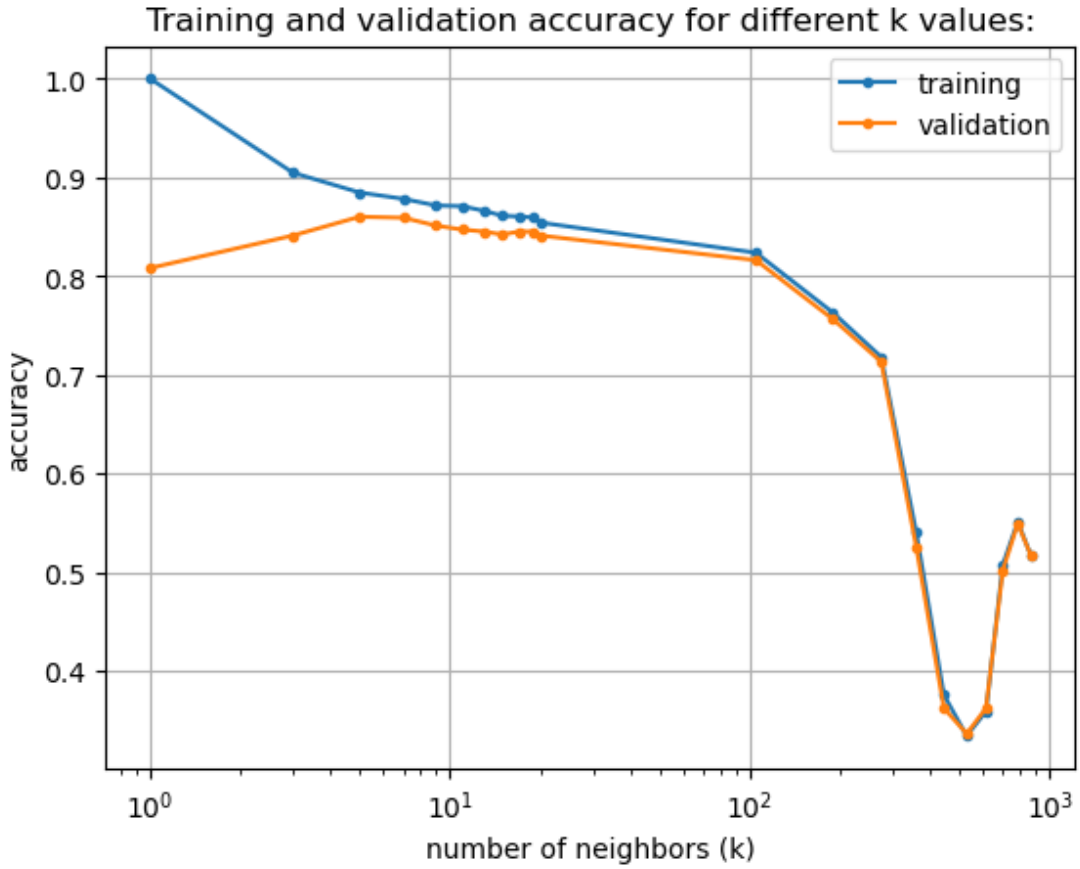
$Q1$:

the output after using visualize_clf:



$Q2$:

$validation\ curve$:

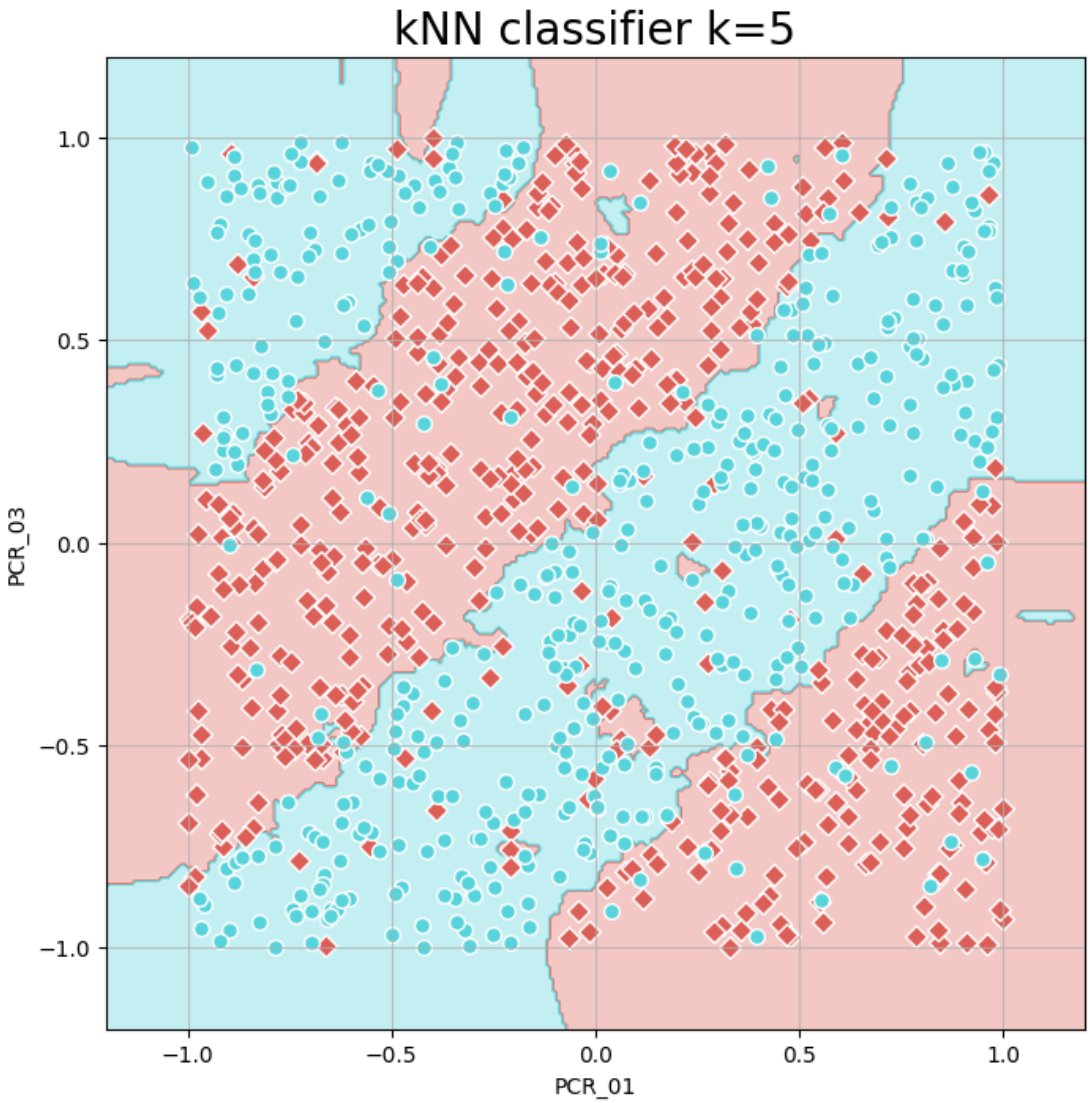


$turns\ out\ that\ k = 5\ is\ the\ best\ choice. average\ training\ accuracy\ is\ 0.884$
$and\ validation\ accuracy\ is\ 0.86.$

$small\ k\ values\ result\ in\ overfitting\ because\ it\ makes\ the\ classifier$
$only\ look\ on\ 1\ sample\ and\ not\ to\ consider\ a\ region, which\ can\ be\ seen\ in\ the\ graph$
$as\ the\ validation\ accuracy\ is\ big\ on\ the\ training\ set\ but\ low\ on\ the\ test\ set.\ \ too\ big\ k$
$values\ result\ in\ making\ the\ classifier\ close\ to\ a\ majority\ functionx, and\ it\ considers$
$regions\ of\ the\ data\ that\ are\ too\ big\ to\ be\ indicative, which\ can\ be\ seen\ in\ the\ graph$
$where\ both\ test\ and\ training\ sets\ have\ low\ accuracy.$

$Q3$:

$decision\ regions$:



$The\ test\ accuracy\ now\ is\ 0.844\ (better\ than\ k = 1\ which\ was\ 0.796)$

$Q4$:

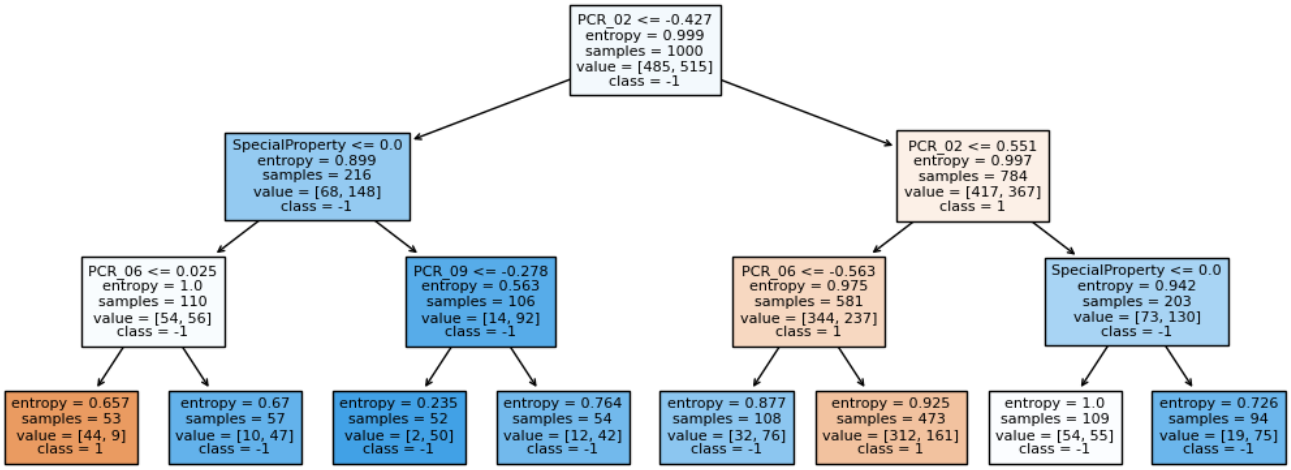
$first\ thing\ to\ notice\ is\ that\ for\ k = 5\ the\ boundries\ are\ more\ smooth.\ that\ goes\ along\ with\ a\ better\ validation\ accuracy\ and\ low\ overfit.\ The\ classifier\ in\ Q3\ acts\ better\ than\ the\ one\ in\ Q1\ in\ terms\ of\ overfit,\ and\ accuracy\ on\ the\ test\ set,\ but\ a\ little\ bit\ less\ better\ on\ the\ training\ set,\ due\ to\ existence\ of$ outliers, $and\ the\ fact\ that\ the\ distribution\ is\ not\ separable$.

Part 2: Decision trees

Visualization

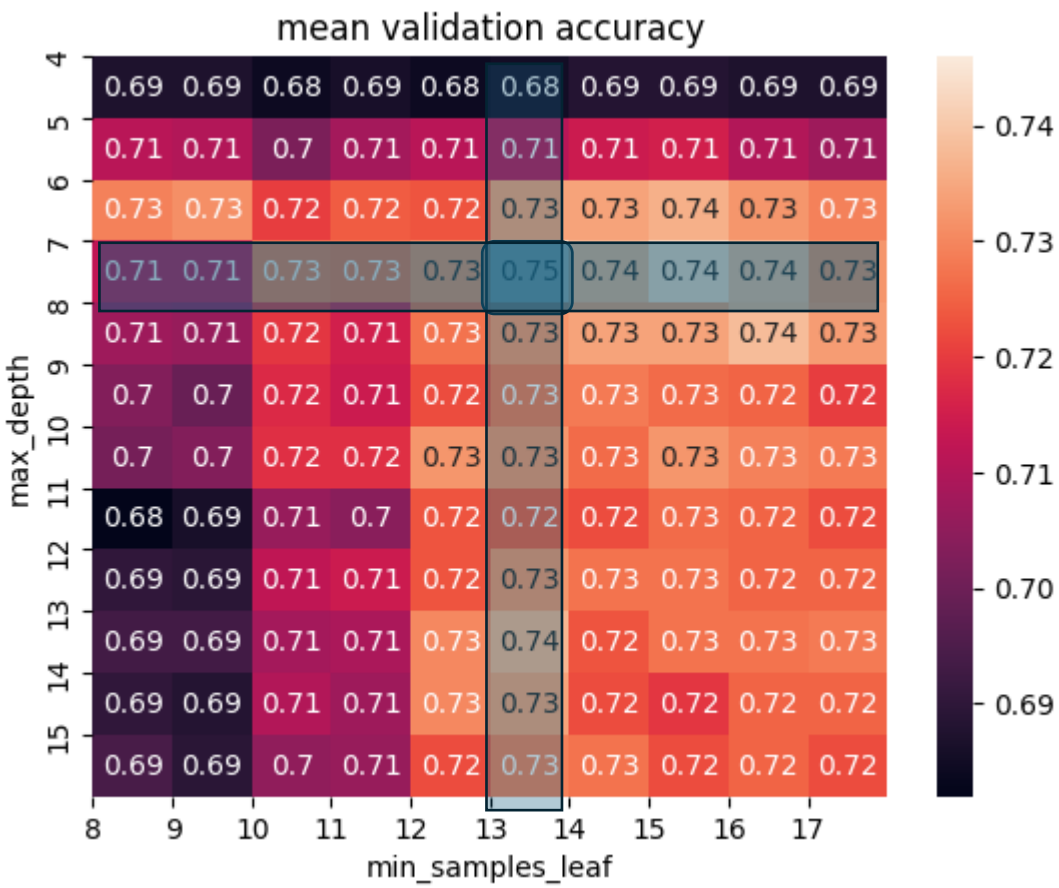
$Q5$:

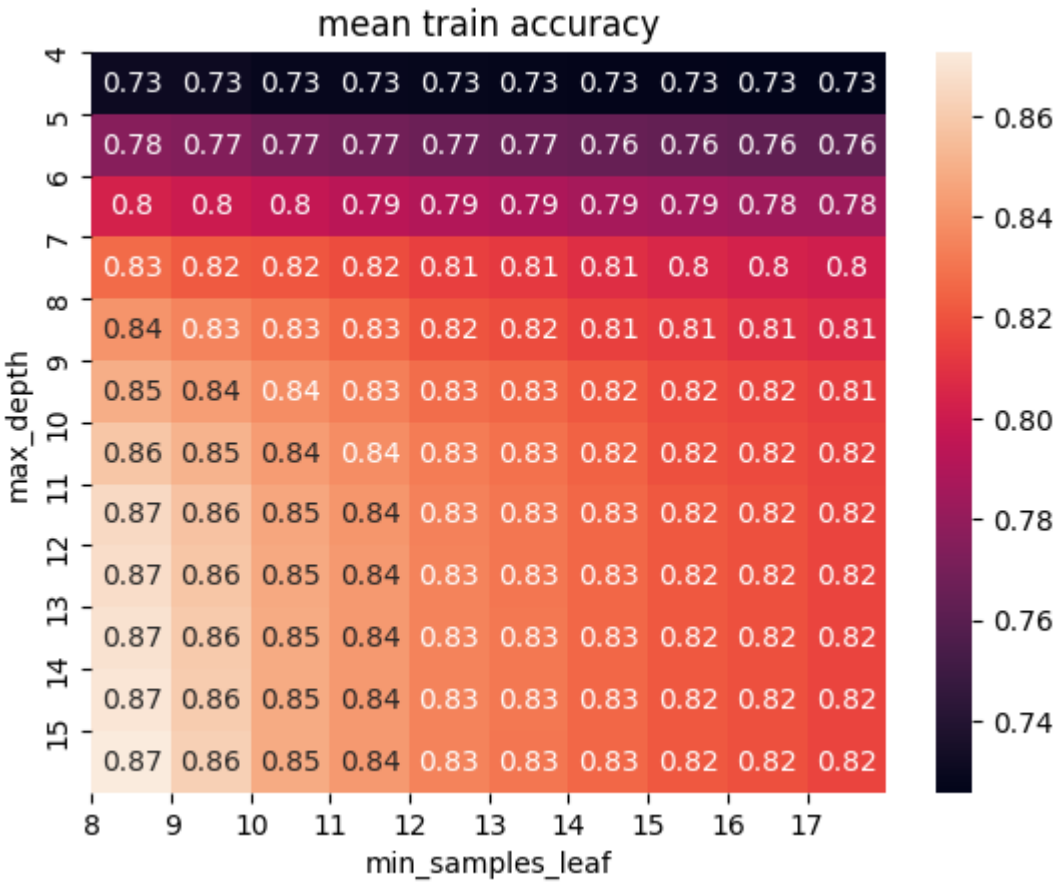
$the\ training\ accuracy\ is$: 0.701.



Model selection

$Q6$:





c: *turns out that the best combination of these two hyperparameters is*:
max $\_depth = 7$ *and* min $\_samples\_leaf = 13$

d: *underfitting case*: $depth = 4$ *and* $minSamplesLeaf = 17$
as can be seen in the heatmap, this combination gives a respectively low accuracy for both train set (0.73) *and validation set* (0.69)

e: *overfitting case*: $depth = 11$ *and* $minSamplesLeaf = 8$
as can be seen in the heatmap, this combination results in train accuracy of 0.87 *and validation accuracy of* 0.68.

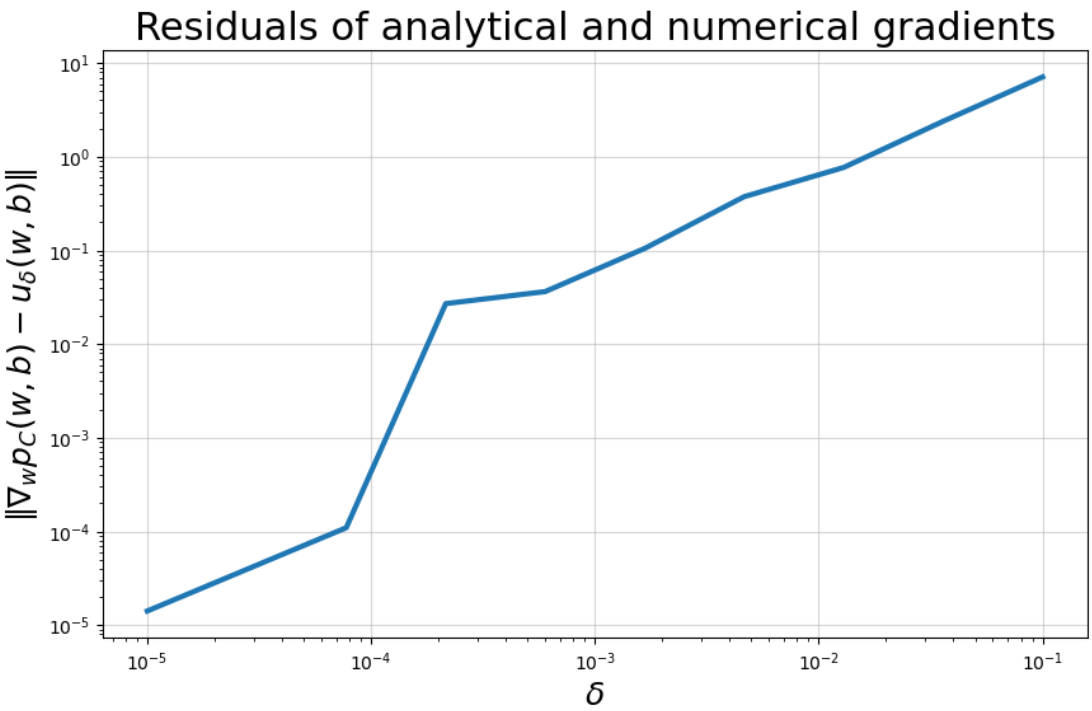
f: *the combination that caused overfitting is due to big depth, which will fit more to the train set, and relatively small leaf size, which demands more accuracy on the train set. the combination that causes underfitting is due to small depth, that does not give the tree any 'chance' to become epressive. also big leaf size which allows less accuracy.*

$Q7$:

the number of combinations is the multiplication of the sizes of the ranges we used: $10 \cdot 12 = 120$. *if we were to add a third parameter, it would multiply the amount of combinations by the size of its range. every addition of a parameters whivh is taken to consideration is making a great deal beacuse it add a whole new dimension to the combinatorical size of the possibilities space.*

$Q8$: *The test accuracy is*: 0.756

$Q9$:

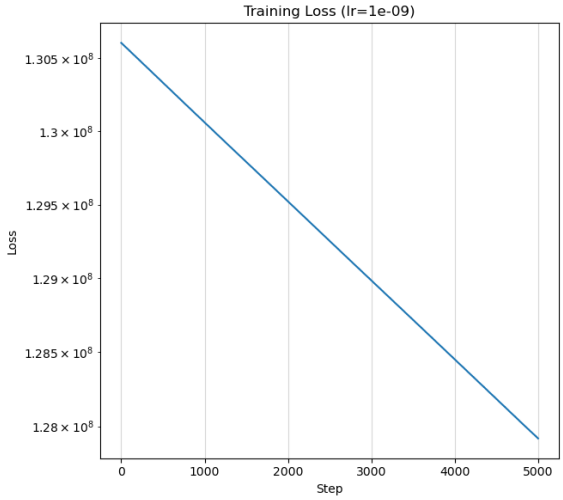


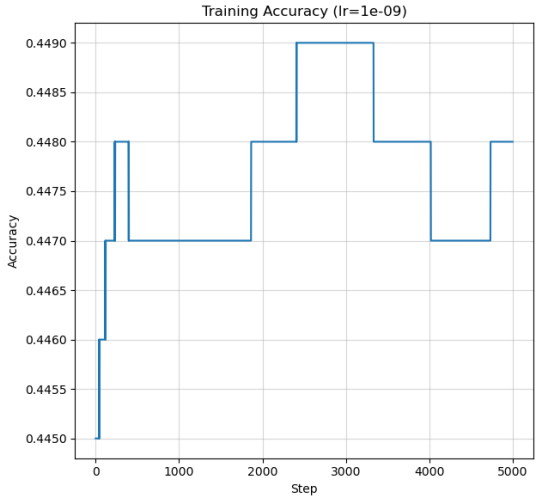
Plugging very small δ values is an approximation for the analytic derivative.
As shown In the plot, the bigger δ is, the bigger the difference between the approximation (numeric gradient) to the real value (analytic gradient) is (almost in a linear fashion), as we would expect.

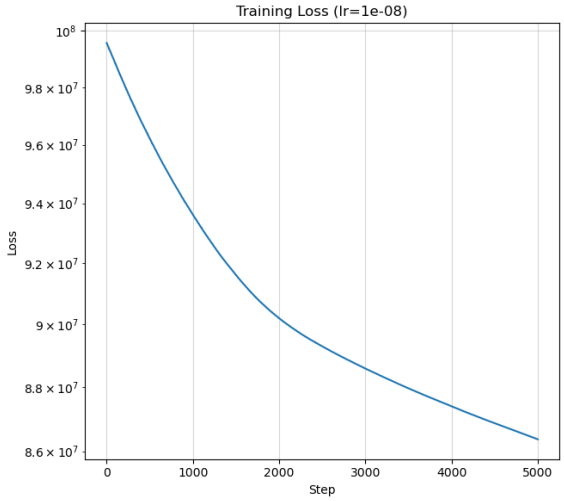
Q10:

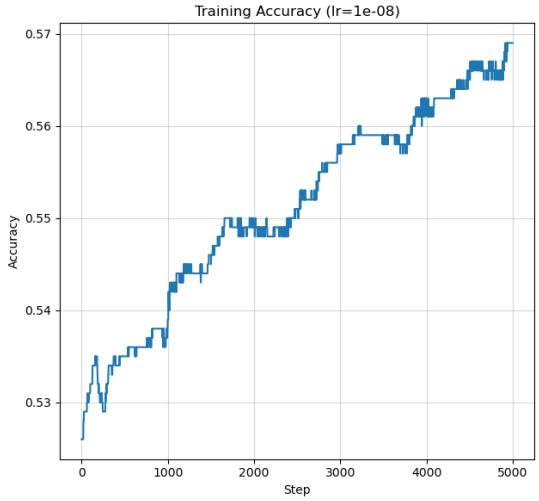
We are trying to minimize the loss, because we have chosen large C, we are very sensitive to the result and we might get overfitting.
When doing such low step it is promised to converage. We little by little will make the loss smaller and making the accuracy bigger.
Out data is not lineary seperable, so SVM does not work here and hence the accuracy is 0.5.
No the interaction is match out expectation, the accuracy does not go as up always. This might be because we know the data is not seperable.

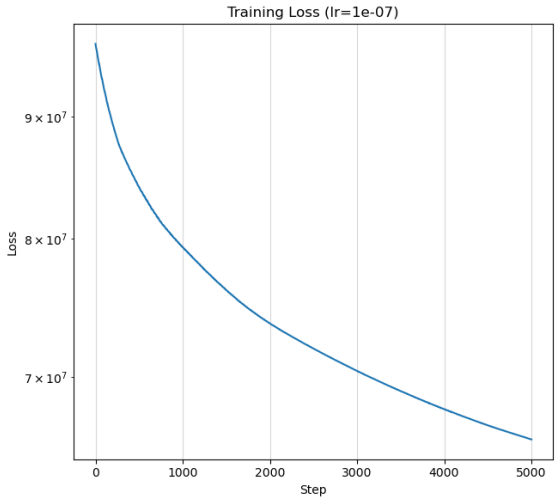
Q11:

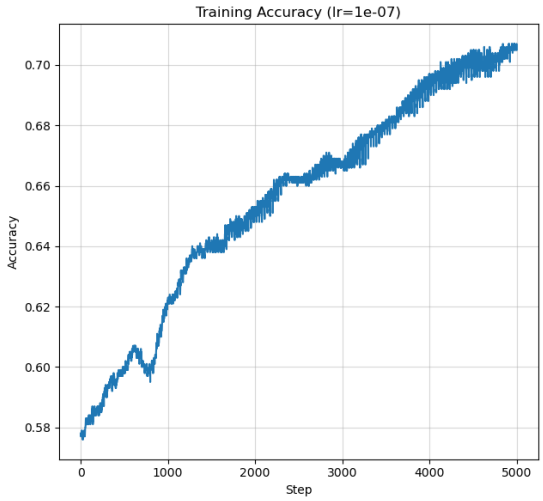

Training Loss (lr=1e-09)
Loss
Step
1.305×10^8
1.3×10^8
1.295×10^8
1.29×10^8
1.285×10^8
1.28×10^8
0
1000
2000
3000
4000
5000

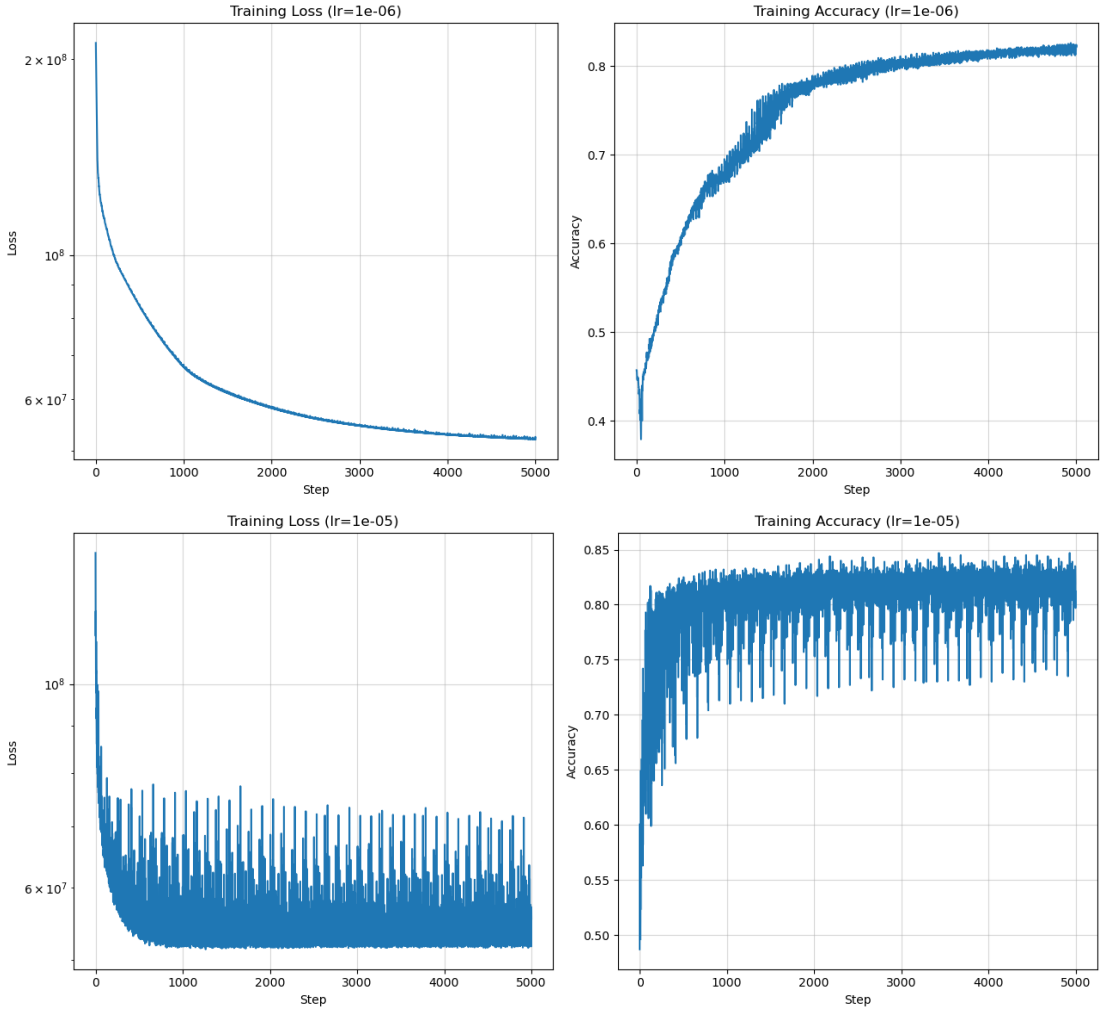

Training Accuracy (lr=1e-09)
Accuracy
Step
0.4490
0.4485
0.4480
0.4475
0.4470
0.4465
0.4460
0.4455
0.4450
0
1000
2000
3000
4000
5000


Training Loss (lr=1e-08)
Loss
Step
10^8
9.8×10^7
9.6×10^7
9.4×10^7
9.2×10^7
9×10^7
8.8×10^7
8.6×10^7
0
1000
2000
3000
4000
5000


Training Accuracy (lr=1e-08)
Accuracy
Step
0.57
0.56
0.55
0.54
0.53
0
1000
2000
3000
4000
5000

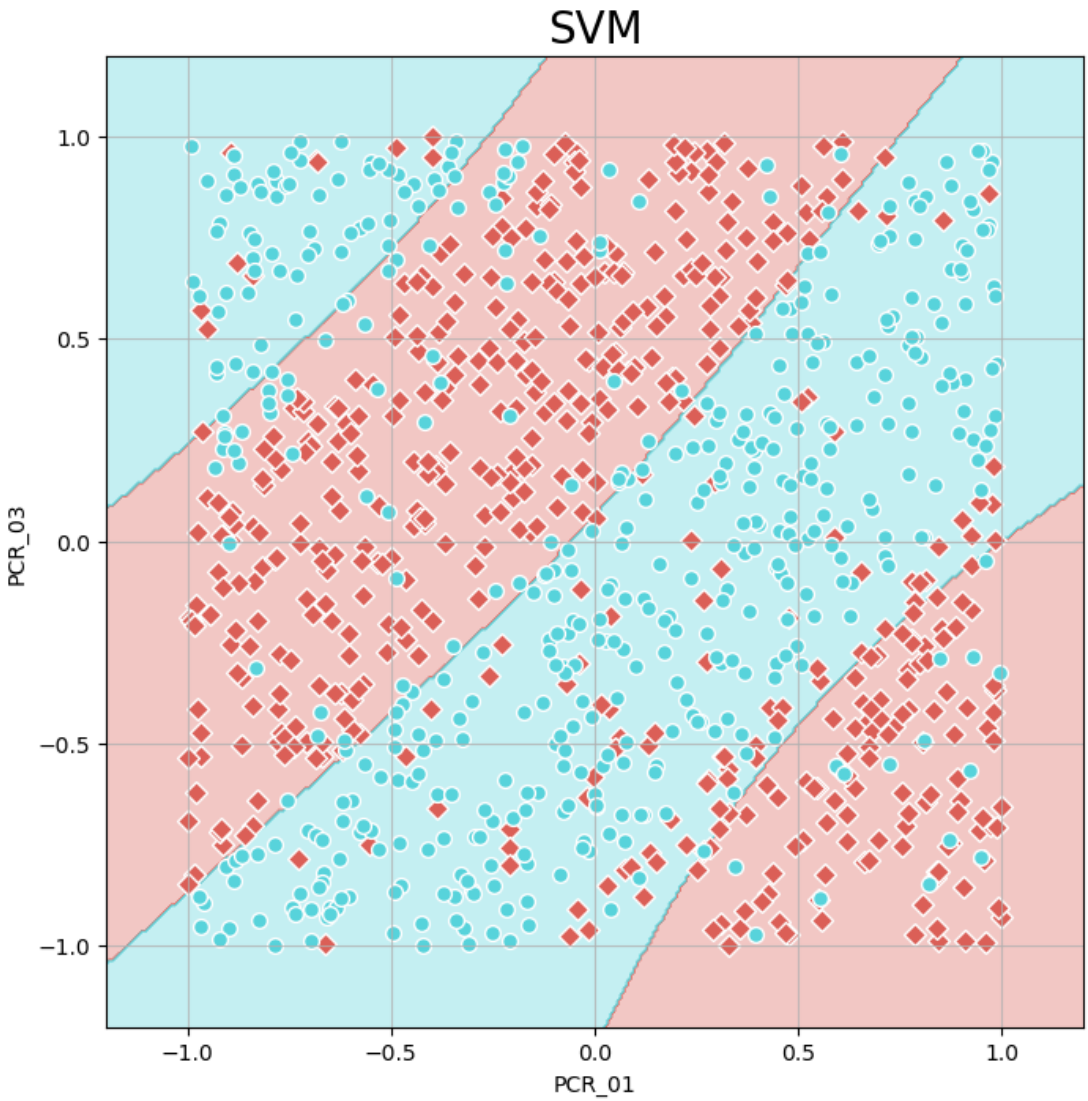

Training Loss (lr=1e-07)
Loss
Step
9×10^7
8×10^7
7×10^7
0
1000
2000
3000
4000
5000


Training Accuracy (lr=1e-07)
Accuracy
Step
0.70
0.68
0.66
0.64
0.62
0.60
0.58
0
1000
2000
3000
4000
5000



We chose lr=e^-06, we can see it converges and that the accuracy gets to maximum. lower than that, that converges is too slow. And bigger than that there is no good converages, but rather a pingpong.

Q12:



test accuracy: 0.816

train accuracy: 0.821

Q13:

A: $\lim_{\gamma\to\infty} sign(\sum_{i\in[m],\alpha i>0} \alpha_i y_i e^{-\gamma\|x-x_i\|_2{}^2}) = sign(\lim_{\gamma\to\infty}(\sum_{i\in[m],\alpha i>0} \alpha_i y_i e^{-\gamma\|x-x_i\|_2{}^2})) =$

$sign((\sum_{i\in[m],\alpha i>0} \alpha_i y_i \lim_{\gamma\to\infty} e^{-\gamma\|x-x_i\|_2{}^2}))_{norms\ is\ bounded} \overset{=}{} sign((\sum_{i\in[m],\alpha i>0} \alpha_i y_i \lim_{\gamma\to\infty} e^{-\gamma c1})) =$

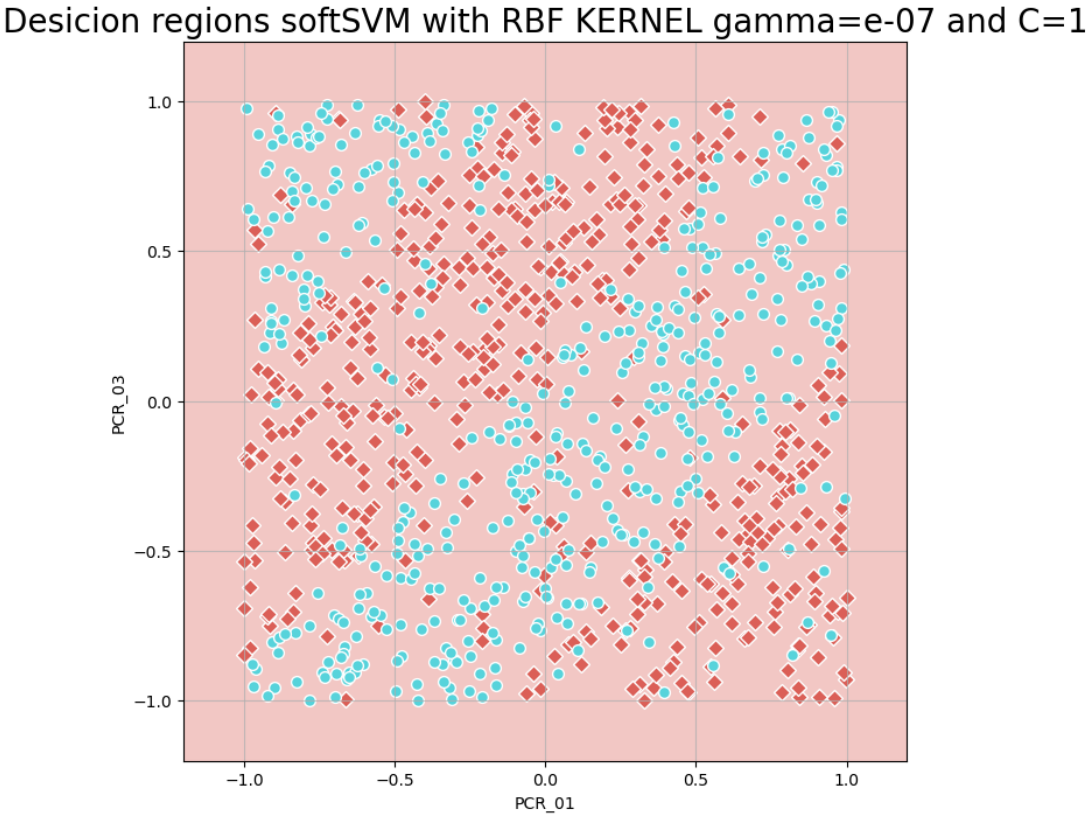
$$sign((\sum_{i\in[m],\alpha i>0} \alpha_i y_i)) =$$

$= \mathrm{argmax}_{y\in\{-1,+1\}} \sum_{\{i|y_i=y\}} \alpha_i$

When the last one is because each $\alpha_i y_i$ has the same power (because the factor of the norm can be abonded when $\gamma \to \infty$. So we only need to see which sum of α_i is bigger (the one of y_i=-1 or the one of y_i=1, because that's what tell us whether the sum will be negative or positive. And that's why we look of argmax with respect to sign of y.

B: each point has the same weight because alpha_i is 1 so in the sum we look of y_i, so we determine the result by the bigger group.
 it is like the model that looks on the biggest group and determines like it. i.e. if we had more points with label y=1 so the answer will be 1. It is KNN with K goes to infinity.

Q14:



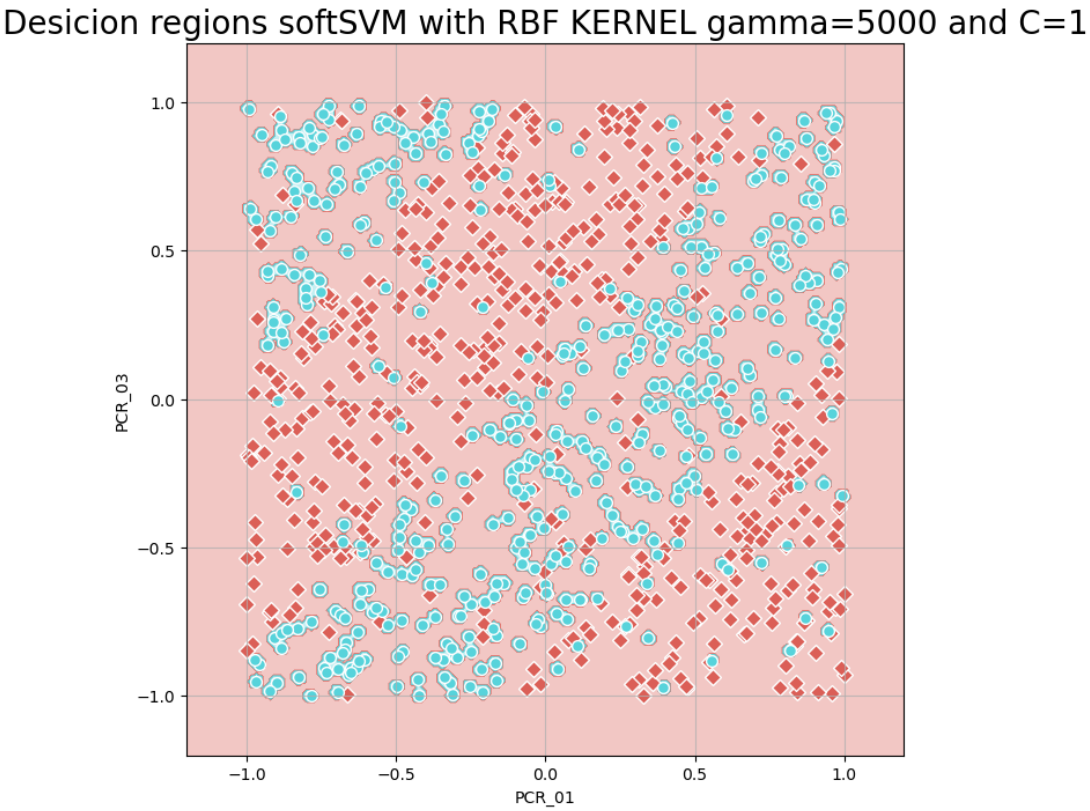
Yes it matches, we see that it goes by -1-negative because it is the larger group.

```
print(spread_col.value_counts()[1])
print(spread_col.value_counts()[-1])
```

[93] ✓ 0.0s

```
484
516
```

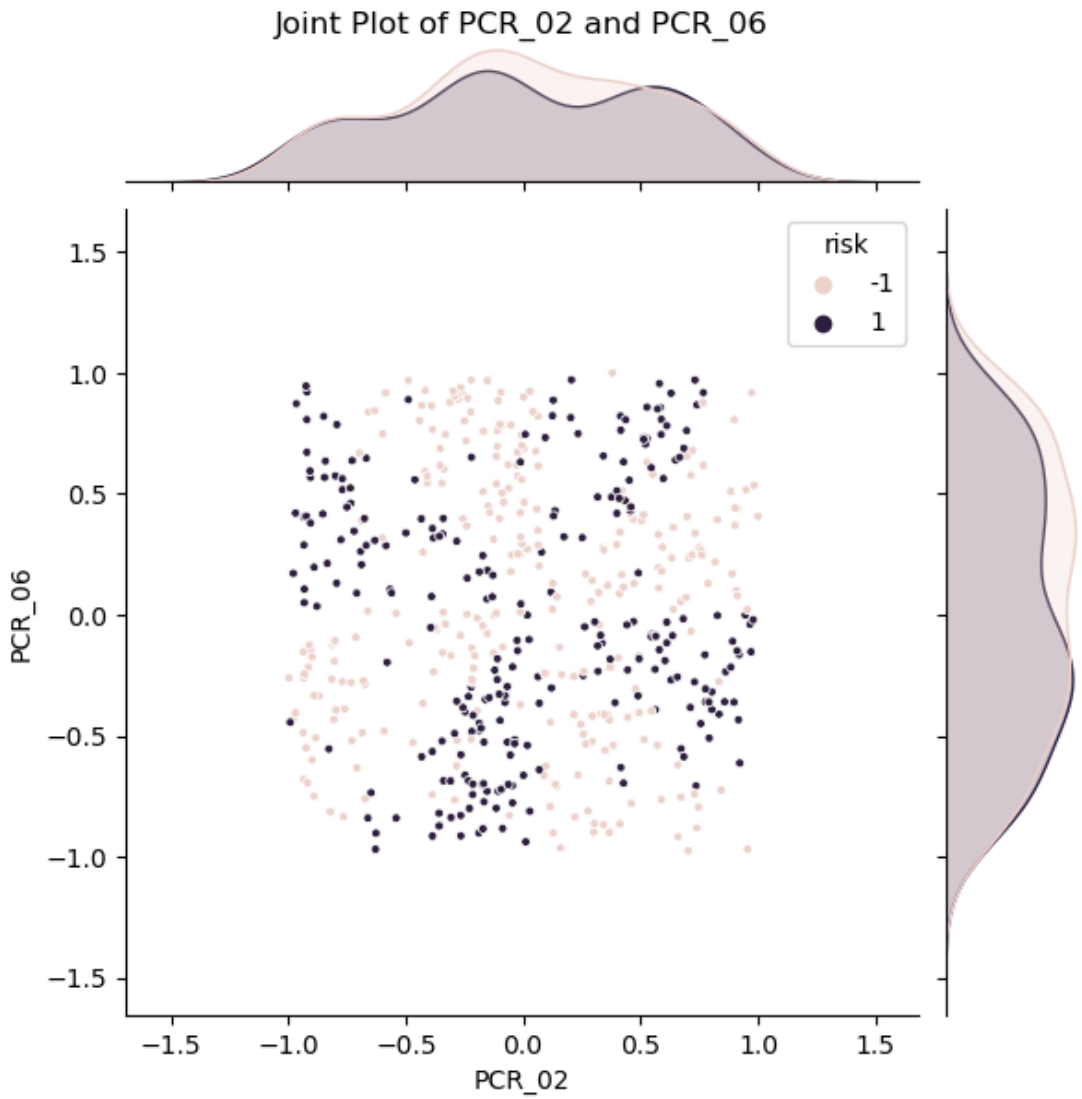
Q15:



SVM with RBF kernel looks with smoother decision boundary compared to 1-NN (You can barely see the blue areas. It is well surrounding the corresponding dots). This is because SVM fits a smooth decision boundary determined by support vectors, while 1-NN's decision boundary is more robust and follows the distribution of training points.

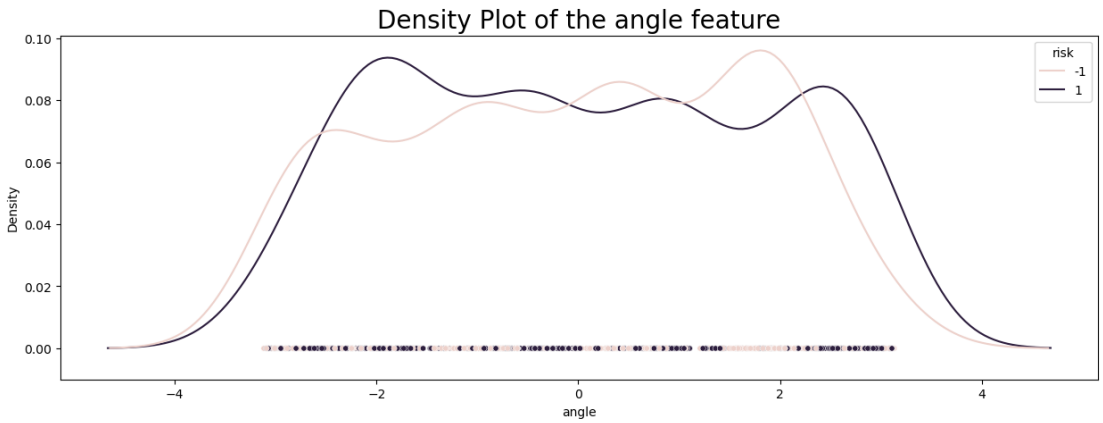
When the distance between data points is not very small, it means that neighboring data points have relatively large distances from each other in the feature space. This means thatn the RBF kernel will assign low similarity scores to these pairs of points. This means you need to be very close in order to get high similarity scores – like 1-NN.

Q16:



We observe the pattern of 8 regions, with each containing unique risk value.

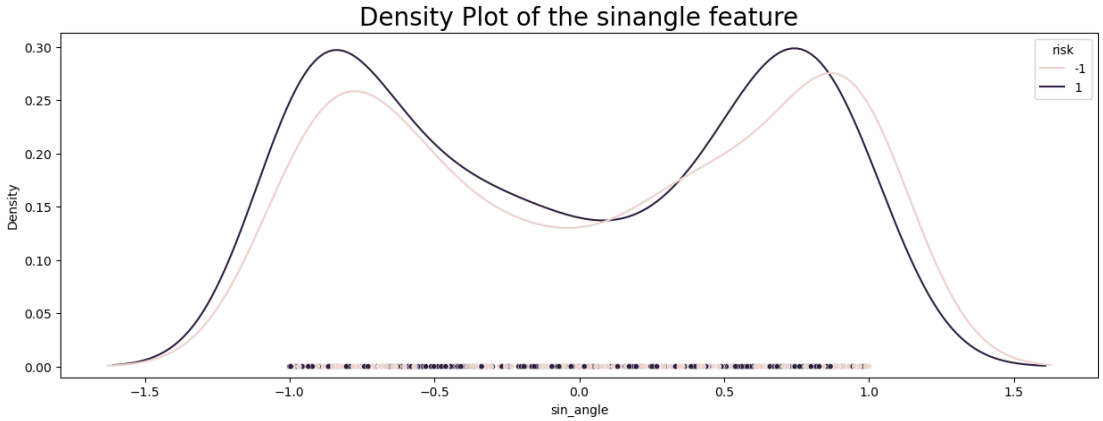
Q17:



It does not seem to be linearly separable after the mapping.

We see that the density of each risk value is about the same.

Q18:



It does not seem to be linearly separable after the mapping.

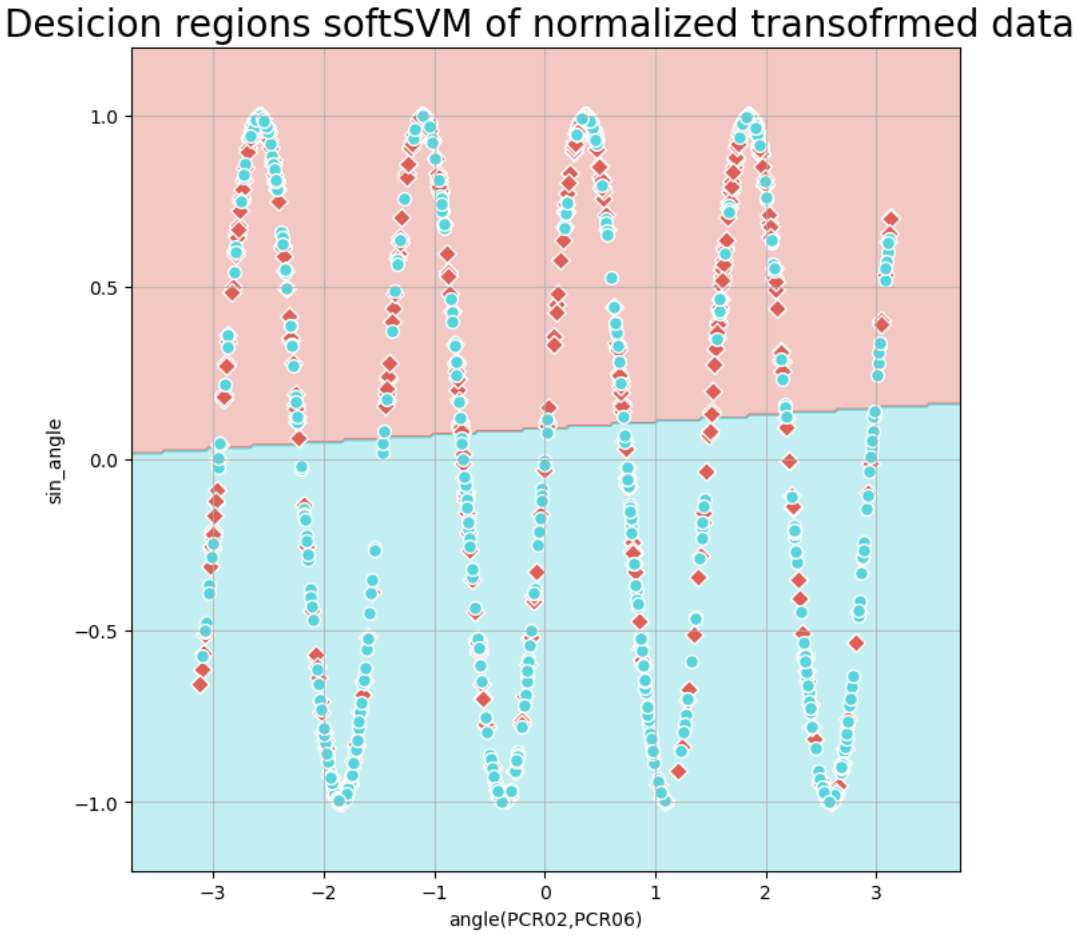
Q19:

Beta should be around 4, because we saw the risk data value changes every 45 degrees.

We did look to find the best one (by iteration) and we got that beta=4.26 is suitable for us.

Training Accuracy: 0.673

Test Accuracy: 0.704



It seems to be better separation than the former.