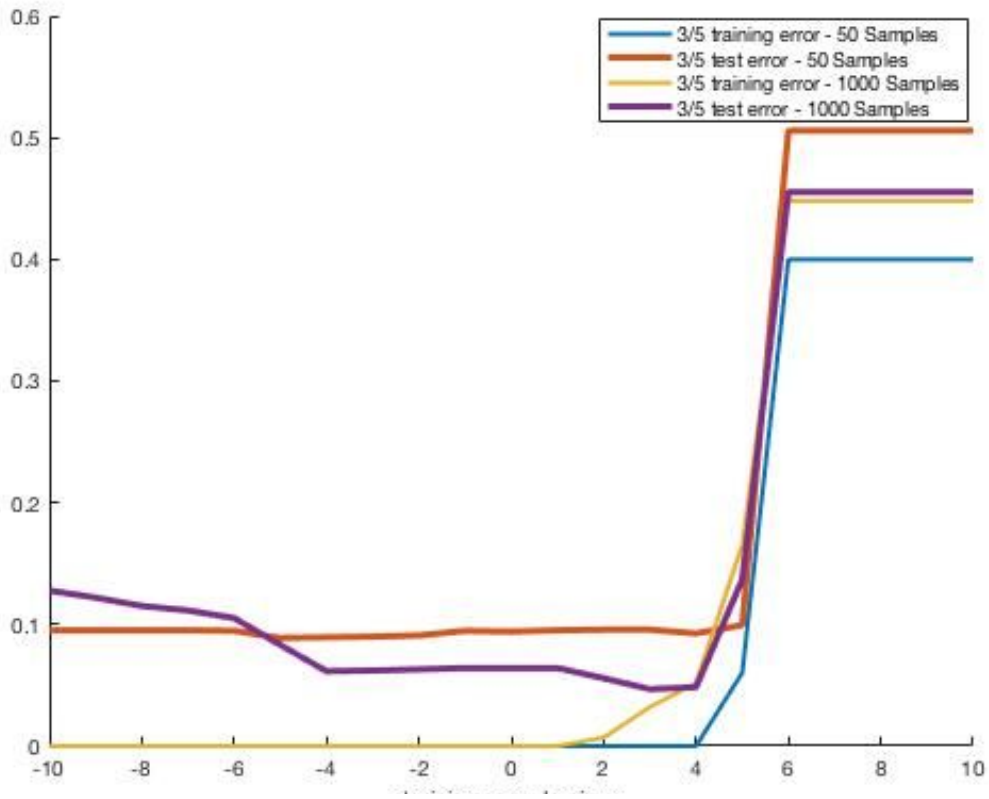
Assignment 2

Question 2. (a) (5 pts) Run your soft SVM implementation on a sample of size 50 with examples only of digits 3 and 5. Try the following values of λ: $\lambda = 10^n$, for $n \in \{-10, -9, \ldots, 0, 1, \ldots, 8\}$. Repeat the experiment with a sample of size 1000. Submit a plot of the test errors as a function of lambda with a line for each sample size. Plot λ on a log scale.

(b) (10pts) Based on what we learned in class, what would you expect the results to look like? Do the results you got match your expectations? In your answer address the following issues:

- The optimal value for λ in each of the sample sizes: which should be larger and why?
- The trend in the test error as a function of λ: should it be decreasing/increasing/something else? Why?

Answers 2. (a)



Answers 2. (b) We expected the result to be like they really appeared. Lambda is a parameter describing the penalty a vector w gets for its size. Therefore, the optimal lambda should punish large w that tries to "fool" the hinge loss, but in a way that will keep the hinge loss relevant and won't cause it to be totally ignored. If lambda is too big, the penalty from the norma of w will mask the penalty from the hinge loss and we will get result that is far from representing a solution for our problem. If it is too small, a large w could be chosen which is not optimal but gets the best result for our sample - meaning we chose a w which is overfit to our sample.
In Addition, the more samples we have, the less we should be bothered by overfitting and we can punish w less for its size and get sufficient results. This is why a smaller lambda is required with larger sample size.

Answers 4. (a)
RBF soft SVM error rate on 5 fold validation:

Lambda **Sigma**	**0.01**	**0.1**	**1**
0.05	6.5%	6.5%	6.6%
1	6.5%	6.5%	6.4%
2	6.5%	6.4%	6.5%

Soft SVM error rate on 5 fold validation:

Lambda	**Validation Error**
0.01	6.5%
0.1	6.6%
1	6.3%

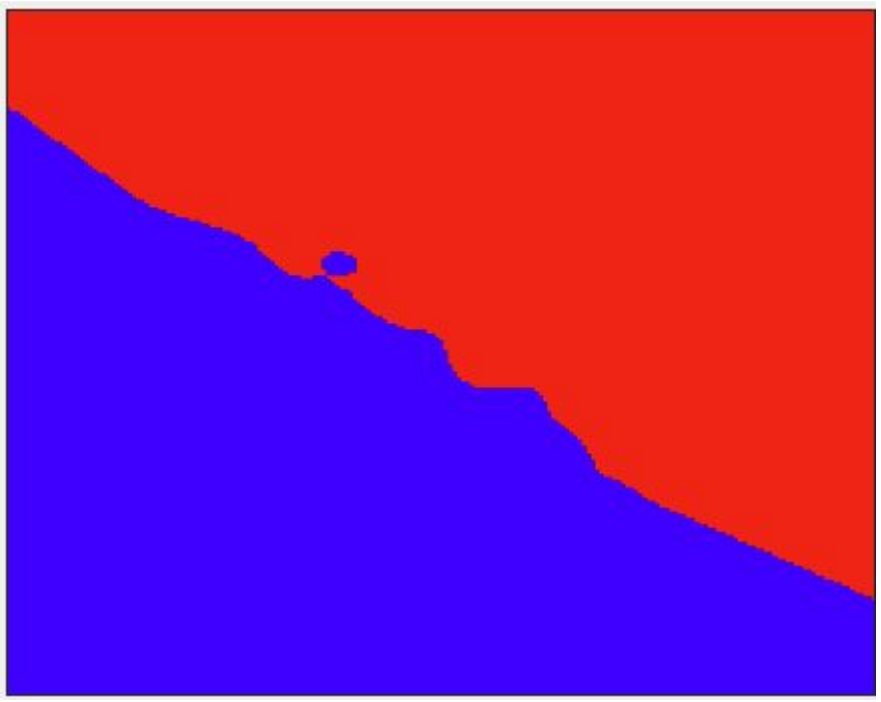
<u>Attention</u>: since the training samples were not distributed uniformly, meaning first occurred only positive samples and then only negative samples we shuffled the training set before performing the K fold validation process.

Answers 4. (b) Against our expectations, the soft SVM (6.3%) achieved a better validation error from RBF (6.4%). One explanation for that, and a reason why RBF might give worse validation error, is if RBF was configured with a parameter σ which is not optimal for the problem. In our situation we can see that the results are very close so we assume that the value of sigma is okay but can still be fine tuned.
A reason why soft RBF might give a better validation error is the fact that while in two-dimensions space the samples might don't have a linear separator, a linear separation for the samples might be able to be achieved in higher dimensional space, which is what RBF does (and this is why we thought it will get better validation error).

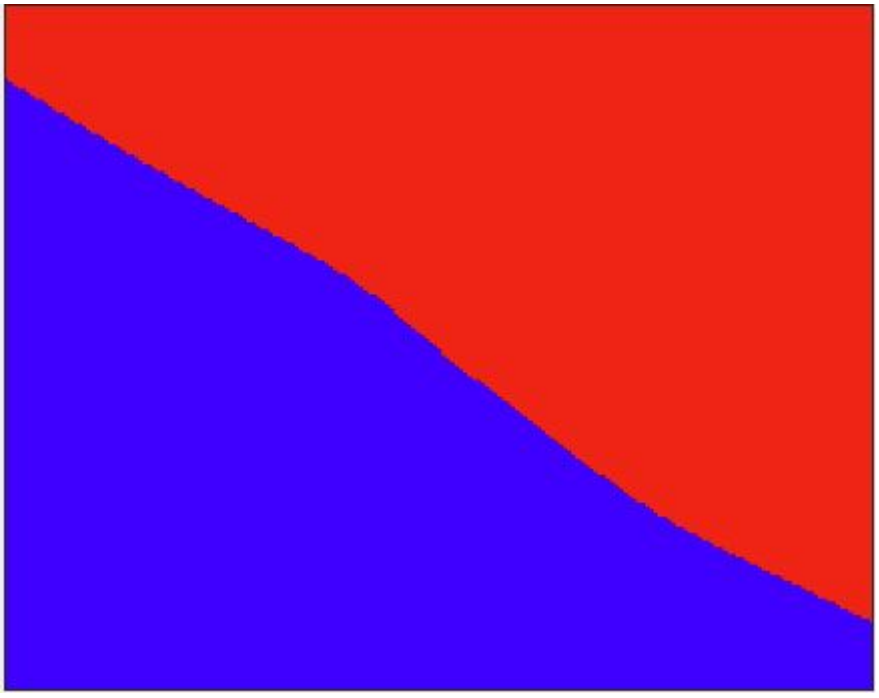
Answers 4. (c**)**

The heatmaps below are for range -5 to 5 in both axis, X and Y. The regions in red represents point on which our separator would classify as positive, and blue for negative.

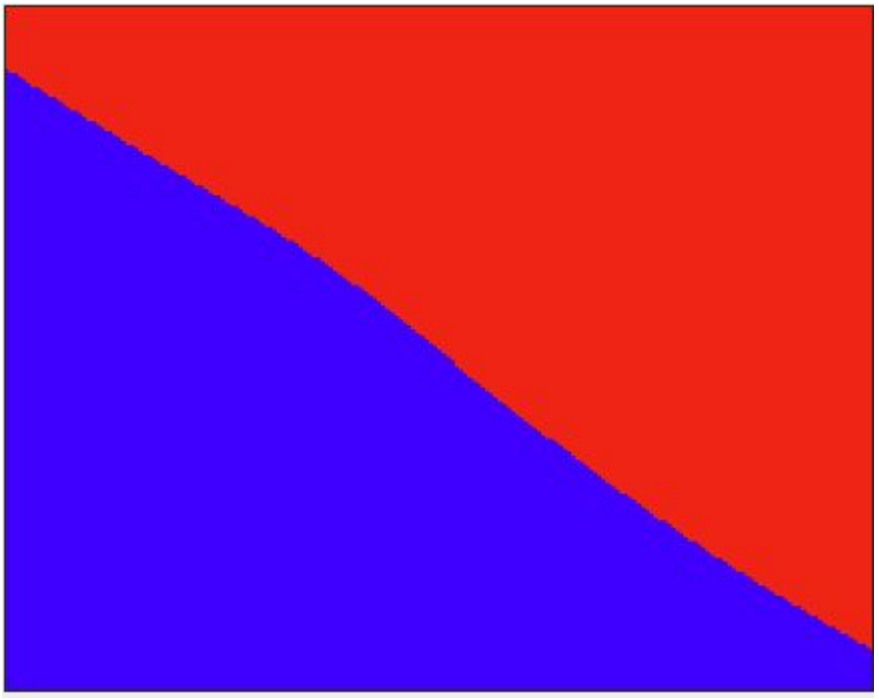
sigma = 0.05



sigma = 1



sigma = 2



Answers 4. (d) We can see in the plots as σ is getting higher the separator line is more linear.
We learned in class that sigma represents, intuitively, the radius of influence of each training sample. Therefore the smaller sigma is the larger estimation error we will get. It can be best seen in the first plot where sigma equals 0.05, the boundaries on this plot are very fine-tuned.In addition the bubble in the middle of the plot might be a good example of an overfitting to the training samples as we can expect when our estimation error increases(we can't know for sure if it really represents an overfitting because we don't know the real distribution, but it does look suspicious).
We also learned in class that the larger sigma is the more approximation error exists. This can be also be seen in the plots as the larger sigma is the smoother the graph is, meaning we pay less attention for few "distracting" points which were outbounded . The risk here is a smooth predictor that treat those samples as "distracting" and pay less attention to them, but in reality those points represents better the distribution than the other majority of points in the training sample.

Question 5. **Kernel functions.** Consider a space of examples $\mathcal{X} = \mathbb{R}^d$. Let $x, x' \in \mathcal{X}$.

(a) (5 pts) Prove that the following function *cannot be* a kernel function for any feature mapping ψ:

$$K(x, x') := -x(1)x'(1).$$

Answers 5. (a) Suppose negatively that a function ψ is a feature mapping, and there exists valid kernel function K, choosing $x, x' \in X$, *such that* $x = x'$:

Then $K(x,x') = \langle \psi(x), \psi(x') \rangle = \sum_{i=1}^{d} \psi(x) \cdot \psi(x') = \sum_{i=1}^{d} \psi(x) \cdot \psi(x) = \sum_{i=1}^{d} \psi^2(x) \geq 0 \Rightarrow K(x,x') \geq 0$.

But lets observe in the case of $x(1) = x'(1) = c$, $c \in \mathfrak{R}$, then by the defintion of K:

$K(x,x') := -x(1)x'(1) = -c^2 \Rightarrow K(x,x') < 0$ for $c \neq 0$. then $K(x,x') \neq \langle \psi(x), \psi(x') \rangle$.

Conradiction for the first claim, therfore there isn't any ψ is a feature mapping for this K.

Question 5. (b) (5 pts) Prove that the following function *cannot be* a kernel function for any feature mapping ψ:

$$K(x, x') := (x(1) + x(2))(x'(3) + x'(4)).$$

Answers 5. (b) As same as in the first section, suppose negatively the existence function ψ is a feature mapping, as before choosing $x, x' \in X$, *such that* $x = x'$, then $K(x,x') \geq 0$.

But lets observe in the case of $x(1) = x(2) = x(3) = c > 0 \wedge x(4) = m < 0$, $c, m \in \mathfrak{R}$, then by the defintion of K:

$K(x,x') := (x(1) + x(2))(x'(3) + x'(4)) = (x(1) + x(2))(x(3) + x(4)) = (c + c)(c + m) = 2c(c + m) \Rightarrow K(x,x') < 0$

for $|m| > |c|$. then $K(x,x') \neq \langle \psi(x), \psi(x') \rangle$.

Conradiction for the first claim, therfore there isn't any ψ is a feature mapping for this K.

Question 5.

$$K(x, x') := (x(1) + x(2))(x'(3) + x'(4)).$$

(c) (10 pts) Recall that the Gaussian kernel is

$$K(x, x') := e^{-\frac{\|x-x'\|^2}{2\sigma}},$$

Prove that the Gaussian kernel is the correct kernel for the coordinate mapping which we gave in class: a coordinate is defined for each finite sequence z with values in $\{1, \ldots, d\}$, and the value of coordinate z in the mapping of x is:

Answers 5.

(c) **Proof:**

By defintion $< \psi(x), \psi(x') > \; = \sum_{n=0}^{\infty} \sum_{z \in \{1,..,d\}^n} \psi(x)(z) \cdot \psi(x')(z)$ define by the qustion

$\psi(x)(z) := \frac{1}{\sqrt{n!}} e^{-\frac{\|x\|^2}{2\sigma}} \prod_{i=1}^{n} x(z(i))/\sqrt{\sigma}$, therefore:

$$= \sum_{n=0}^{\infty} \sum_{z \in \{1,..,d\}^n} \frac{1}{\sqrt{n!}} e^{-\frac{\|x\|^2}{2\sigma}} \prod_{i=1}^{n} x(z(i))/\sqrt{\sigma} \cdot \frac{1}{\sqrt{n!}} e^{-\frac{\|x'\|^2}{2\sigma}} \prod_{i=1}^{n} x'(z(i))/\sqrt{\sigma} = e^{-\frac{\|x\|^2}{2\sigma} - \frac{\|x'\|^2}{2\sigma}} \cdot \sum_{n=0}^{\infty} \frac{1}{n!\sigma^n} \sum_{z \in \{1,..,d\}^n} \prod_{i=1}^{n} x(z(i)) \cdot x'(z(i))$$

And this is all the posiblity of multipiction in the space, therefore:

$$= e^{-\frac{\|x\|^2+\|x'\|^2}{2\sigma}} \cdot \sum_{n=0}^{\infty} \frac{1}{n!\sigma^n} \sum_{i=1}^{d} x(i) \cdot x'(i) = e^{-\frac{\|x\|^2+\|x'\|^2}{2\sigma}} \cdot \sum_{n=0}^{\infty} \frac{1}{n!\sigma^n} < x, x' >^n = e^{-\frac{\|x\|^2+\|x'\|^2}{2\sigma}} \cdot \sum_{n=0}^{\infty} \frac{<x,x'>^n}{n!\sigma^n} =$$

As we saw in class, this is a Taylor series $e^{-<x,x'>/2\sigma}$, therefore:

$$= e^{-\frac{\|x\|^2+\|x'\|^2}{2\sigma}} \cdot e^{-<x,x'>/2\sigma} = e^{-\frac{\|x\|^2+<x,x'>+\|x'\|^2}{2\sigma}} = e^{-\|x-x'\|^2/2\sigma} = K(x, x')$$

We find existence ψ such that $< \psi(x), \psi(x') > = K(x, x')$, as we see in class this claim is eqivlent to

$K(x, x') \; := \; e^{-\|x-x'\|^2/2\sigma}$ is a kernel function.

Question 6. (10 pts) **Hoeffding's bound.** Consider a learning algorithm with k numerical parameters. Suppose that we use a validation set of size n to decide which value to use for each of the parameters. For each numerical parameter we have r possible values, and we try each combination of values, by running the learning algorithm with this value combination on the training set, and calculating the validation error of the resulting classifier. Finally, we select the classifier that got the smallest validation error.

Let $\delta, \epsilon \in (0, 1)$. What should the size of the validation set n be, to guarantee that with a probability of at least $1 - \delta$ over the choice of the validation set, the error of the classifier that we selected is at most ϵ more than the error of the best classifier in the set of classifiers that we tested? Your answer should depend on δ, ϵ, k, r. Prove your claim using Hoeffding's bound.

Answers 6. Define $\psi = \{(i_1.....i_r) | \ 1 \le i_j \le k, \ 1 \le j \le r\}$. Any element in ψ define a possible parameter combination for our learning algorithm. We will now show that for $n \ge \frac{log(2|\psi|/\delta)}{2*(0.5\varepsilon)^2}$, $(|\psi| = k^r) \rightarrow n \ge \frac{r*log(\frac{2k}{\delta})}{2*(0.5\varepsilon)^2}$ the classifier that we selected is at most ε more than the true error of the best classifier in probability of $1 - \delta$.

First we notice that from the conclusion of Hoeffding's bound that was presented in class we get:

$P[\forall \alpha \in \psi, |err(h_\alpha, V) - err(h_\alpha, D)| \le 0.5 * \varepsilon] \ \ge 1 - \delta$.

Let h be the best classifier on the distribution with some $\alpha_1 \in \psi$, and h' be the classifier we selected with $\alpha \in \psi$. Since we chose h' we know that $err(h', V) \le err(h, V)$ and since h is the best classifier we know that $err(h, D) \le err(h', D)$. Notice that we actually want to proof that: $P[err(h', D) \ \le \varepsilon + err(h, D)] \ge 1 - \delta$

We will now split into cases:

1. $err(h', V) \ge err(h', D)$

   This means: $err(h, V) \ge err(h', V) \ge err(h', D) \ge err(h, D)$.

   Therefore from Hoeffding's bound:

   $P[|err(h, V) \ - err(h, D)| \le 0.5 * \varepsilon] \ge 1 - \delta$.

   $err(h, V) \ge err(h, D) \ => \ \ P[err(h, V) \ - err(h, D) \le 0.5 * \varepsilon] \ge 1 - \delta$

   $P[err(h, V) \ \le 0.5 * \varepsilon + err(h, D)] \ge 1 - \delta$

   $err(h', D) \le err(h, V), \ 0.5\varepsilon \le \varepsilon \ => \ P[err(h', D) \ \le \varepsilon + err(h, D)] \ge 1 - \delta$.

2. $err(h', V) \le err(h', D)$

   a) $err(h, V) \le err(h, D)$

      this means: $err(h', V) \le err(h, V) \le err(h, D) \le err(h', D)$

      Therefore from Hoeffding's bound:

      $P[|err(h', V) - err(h', D)| \le 0.5 * \varepsilon] \ge 1 - \delta$

      $err(h', D) \ge err(h', D) \ => P[err(h', D) - err(h', V) \le 0.5 * \varepsilon] \ge 1 - \delta$

      $P[err(h', D) \le 0.5 * \varepsilon + err(h', V)] \ge 1 - \delta$

      $err(h', V) \le err(h, D), \ 0.5\varepsilon \le \varepsilon \ => \ P[err(h', D) \le \varepsilon + err(h, D)] \ge 1 - \delta$

b) $err(h, V) \geq err(h, D)$

Therefore from Hoeffding's bound:

$P[|err(h', V) - err(h', D)| \leq 0.5 * \varepsilon \ \wedge \ |err(h, V) - err(h, D)| \leq 0.5 * \varepsilon] \geq 1 - \delta$

$err(h, V) \geq err(h, D)$, $err(h', V) \leq err(h', D)$ =>

$P[err(h', D) - err(h', V) \leq 0.5 * \varepsilon \ \wedge \ err(h, V) - err(h, D) \leq 0.5 * \varepsilon] \geq 1 - \delta$ =>

$P[err(h', D) \leq 0.5 * \varepsilon + err(h', V) \ \wedge \ err(h, V) \leq 0.5 * \varepsilon + err(h, D)] \geq 1 - \delta$ =>

$err(h', V) \leq err(h, V)$ =>

$P[err(h', D) \leq 0.5 * \varepsilon + err(h, V) \ \wedge \ err(h, V) \leq 0.5 * \varepsilon + err(h, D)] \geq 1 - \delta$ =>

$P[err(h', D) \leq 0.5 * \varepsilon + 0.5 * \varepsilon + err(h, D)] \geq 1 - \delta$ => $P[err(h', D) \leq \varepsilon + err(h, D)] \geq 1 - \delta$