Assigment 2 FMAN45

Kalle Josefsson ka5532jo-s

April 25, 2024

1 Task 1

I used the formulas provided in the lab notes on how to calculate the values in the kernel. First we need to use the formula for the matrices called $\phi(x)$ using the values given in lab. The equation to calculate $\phi(x)$ is presented below.

$$\phi(x) = \begin{pmatrix} x \\ x^2 \end{pmatrix} \tag{1}$$

Then to calculate the value of a Kernel we use the equation equation as below.

$$k(x, y) = \phi(x)\phi(y)^T \tag{2}$$

The task was to calculate the value of the kernel k(x,x) which was done and got the following result.

$$K = k(x_i, x_j) = \begin{pmatrix} 20 & 6 & 2 & 12 \\ 6 & 2 & 0 & 2 \\ 2 & 0 & 2 & 6 \\ 12 & 2 & 6 & 20 \end{pmatrix} \tag{3}$$

2 Task 2

The second part of the task was to solve the maximization problem presented below.

$$\begin{aligned} \max \quad & \left\{ \sum_{i=1}^{4} \alpha_i - \frac{1}{2} \sum_{i=1}^{4} \sum_{j=1}^{4} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \right\} \\ \text{subject to} \quad & \alpha_i \geq 0, \\ & \sum_{i=1}^{4} y_i \alpha_i = 0 \end{aligned} \tag{4}$$

This can be very tricky but we were allowed to assume that $\alpha = \alpha_1 = \alpha_2 = \alpha_3 = \alpha_4$ and did not have to provide a proof for it so I did not. This however makes the optimization problem much easier to solve since we can rewrite the equation as below.

$$\max \left\{ 4\alpha - \frac{1}{2}\alpha^2 \sum_{i=1}^{4} \sum_{j=1}^{4} y_i y_j k(x_i, x_j) \right\} \tag{5}$$

Now we just remove the summation mark by doing the vector/matrix multiplication with the kernel we calculated in task 1 and we get the following polynomial to maximize with respect to α.

$$\max \left\{ 4\alpha - \frac{1}{2}\alpha^2 (20 - 6 - 2 + 12 - 6 + 2 + 0 - 2 - 2 + 0 + 2 - 6 + 12 - 2 - 6 + 20) \right\} \tag{6}$$

Which can be simplified to:

$$\max\left\{4\alpha - 18\alpha^2\right\} \tag{7}$$

We differentiate with respect to α and get:

$$\begin{aligned} f(\alpha) &= 4\alpha - \frac{1}{2}\alpha^2 \times 36, \\ \frac{d}{d\alpha}f(\alpha) &= 4 - 36\alpha, \\ 0 &= 4 - 36\alpha, \\ \alpha &= \frac{4}{36} = \frac{1}{9}. \end{aligned} \tag{8}$$

To check weather we have a minima or maxima we take the second derivative which is just -36 which is greater than zero and hence we have a maxima and so finally we have $\alpha = 1/9$

3 Task 3

For task three we want to reduce our classifier into its most simple form. But first we need an expression for it which is presented below.

$$g(x) = \sum_{j=1}^{4} \alpha_j y_j k(x_j, x) + b \tag{9}$$

Now we insert the calculated α, the newly calculated kernel $k(x_j, x)$ and the y-values presented in the lab. This give the new equation:

$$g(x) = \frac{1}{9}\left((4x^2 - 2x) - (x^2 - x) - (x^2 + x) + (4x^2 + 2x)\right) + b = \frac{2}{3}x^2 + b \tag{10}$$

But b is still unknown but fortunately we have a condition which will help us calculate b. This is presented below.

$$y_s\left(\sum_{j=1}^{4} \alpha_j y_j k(x_j, x_s) + b\right) = 1 \tag{11}$$

Now we insert the calculated g(x) into equation 11 and use the first data point which was $(x_1, y_1) = (-2, 1)$ and get the following result.

$$\begin{aligned} y_s\left(\frac{2}{3}x^2 + b\right) &= 1 \\ 1\left(\frac{2}{3}(-2)^2 + b\right) &= 1 && \text{(Substitute } y_s = 1 \text{ and } x = -2\text{)} \\ \frac{8}{3} + b &= 1 \\ b &= 1 - \frac{8}{3} \\ b &= \frac{3}{3} - \frac{8}{3} \\ b &= -\frac{5}{3} && \text{(Simplify to find } b\text{)} \end{aligned}$$

Now that we have be we get the final solution for g(x) in equation 12.

$$g(x) = \frac{2}{3}x^2 - \frac{5}{3} \tag{12}$$

4 Task 4

The dataset to be handled in task four just happens to be a superset of the dataset we have in task one through three. The data points $x_2, x_3, x_5 and x_6$ are all datapoints which we used before. And if we use see how the other newly added data points fits into our classifier function we can see that it still works fine at getting the correct classification, this since the value one gets when plugging our new data points in are all either greater than one or less than one. Also worthy to note is that no datapoint has an x-value in the margin which removes that problem.

5 Task 5

From the linear soft margin classifier primal formulation (13, 14, 15) we want to derive the dual Lagrange problem.

$$\min_{\omega,b,\xi}\left(\frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{n}\xi_i\right) \tag{13}$$

subject to

$$y_i(\omega^T x_i + b) \geq 1 - \xi_i, \quad \forall i, \tag{14}$$

$$\xi_i \geq 0, \quad \forall i \tag{15}$$

In order to solve this task we begin with expressing the Lagrange function.

$$L(\omega, b, \xi) = \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\alpha_i(1 - \xi_i - y_i(\omega^T x_i + b)) - \sum_{i=1}^{n}\lambda_i\xi_i \tag{16}$$

subject to

$$\alpha_i\lambda_i \geq 0, \quad \forall i \tag{17}$$

For us to solve this we need to minimize L with respect to ω, b, ξ which is done by differentiation and setting equal to zero.

$$\frac{\partial L}{\partial \omega} = 0 \Rightarrow \omega + \sum_{i=1}^{n}\alpha_i(-y_i)x_i = \omega - \sum_{i=1}^{n}\alpha_i y_i x_i \Rightarrow \omega = \sum_{i=1}^{n}\alpha_i y_i x_i \tag{18}$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow -\sum_{i=1}^{n}\alpha_i y_i = 0 \Rightarrow \sum_{i=1}^{n}\alpha_i y_i = 0 \tag{19}$$

$$\frac{\partial L}{\partial \xi} = 0 \Rightarrow C - \alpha_i - \lambda_i = 0 \Rightarrow \lambda_i = C - \alpha_i \tag{20}$$

Now that we have some conditions we can use them to simplify the Lagrange function. First we use the condition in (18) and substitute all ω in (16) and get:

$$\max_{\alpha_i} L = \frac{1}{2}\left\|\sum_{i=1}^{n}\alpha_i y_i x_i\right\|^2 + C\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\alpha_i\left(1 - \xi_i + y_i\left(\left(\sum_{i=1}^{n}\alpha_i y_i x_i\right)^T x_i + b\right)\right) - \sum_{i=1}^{n}\lambda_i\xi_i \tag{21}$$

This equation we can rearrange and then rewrite it as:

$$\frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\alpha_j\alpha_i y_j y_i x_j^T x_i + C\sum_{i=1}^{n}\xi_i(C - \alpha_i - \lambda_i) + \sum_{i=1}^{n}\alpha_i - \sum_{i=1}^{n}\alpha_i y_i b - \sum_{j=1}^{n}\sum_{i=1}^{n}\alpha_j\alpha_i y_j y_i x_j^T x_i \tag{22}$$

From (20) we can remove the second term in (22) due to the expression in parenthesis equals to zero for all i. We can also remove the forth term using the condition found in (19). The first and last terms in (22) are the same which means we can simplify to:

$$\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{j=1}^{n} \sum_{i=1}^{n} \alpha_j \alpha_i y_j y_i x_j^T x_i \tag{23}$$

Where we use:

$$\alpha_i, \lambda_i \geq 0, \quad \forall i, \tag{24}$$
$$\lambda_i = C - \alpha_i, \quad \forall i \tag{25}$$
$$0 \leq \alpha_i \leq C, \quad \forall i \tag{26}$$

6 Task 6

For the last theoretical task we want to show that for support vectors according to (27) have coefficients $\alpha_i = C$

$$y_i(\omega^\top x_i + b) > 1 \tag{27}$$

The complementary slackness is express as:

$$\alpha_i(y_i(\omega^\top x_i + b) + \xi_i - 1) = 0 \tag{28}$$

$$\lambda_i \xi_i = 0 \tag{29}$$

Again we use the condition from (20) we can change our condition in (29) into:

$$(C - \alpha_i)\xi_i = 0 \tag{30}$$

If we know that $\xi_i > 0$ this only holds if $C = \alpha_i$ what also follows is that when we rewrite (28) we get:

$$y_i(\omega^\top x_i + b) + \xi_i - 1 = 0 \tag{31}$$
$$\Rightarrow \xi_i = 1 - y_i(\omega^\top x_i + b) \tag{32}$$

Since $\xi_i > 0$ we get the following inequality:

$$1 - y_i(\omega^\top x_i + b) > 0 \tag{33}$$
$$\Rightarrow y_i(\omega^\top x_i + b) < 1 \tag{34}$$

This proves what we wanted to prove and wraps up the theoretical part of the lab.

7 Task 7

For task 7 we were supposed to project the MNIST dataset, images showing zeros and ones, onto its first and second principal component. In order to to this computation we first altered the dataset into being zero mean. Then we performed singular value decomposition in order to find the first and second principal components which made it possible for us to make the dimensionality reduction. Figure 1 below shows the results when plotting.

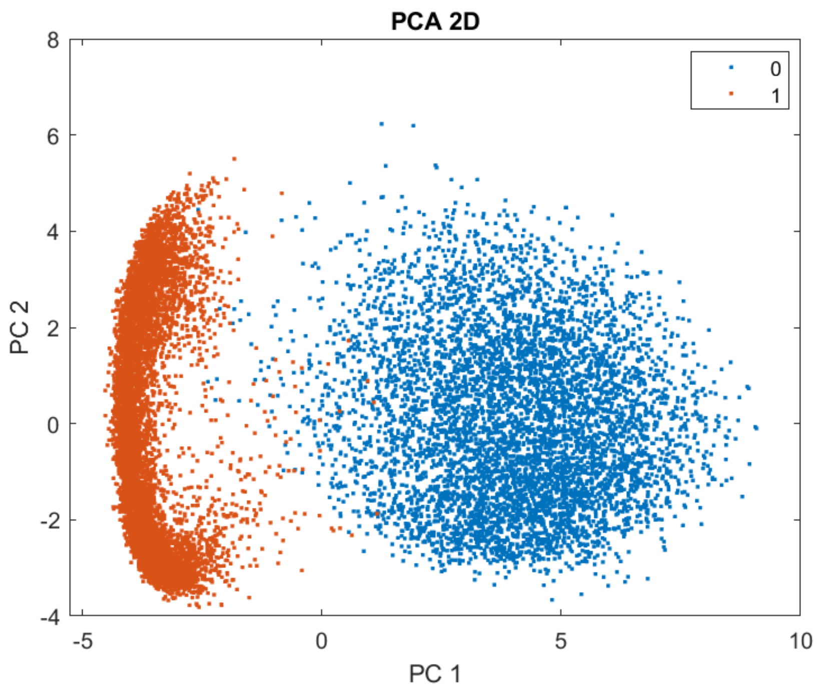
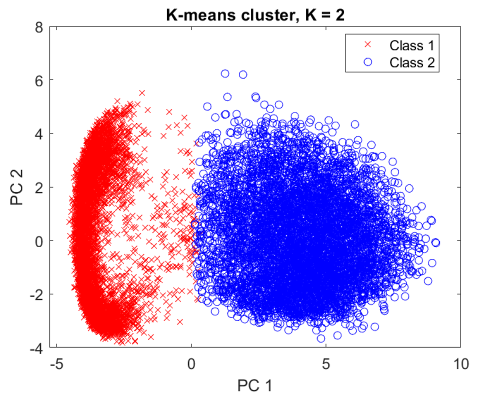


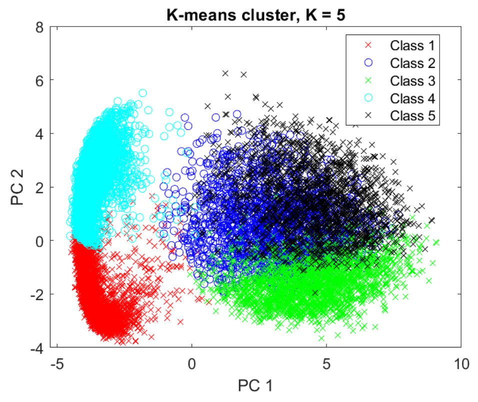
Figure 1: PCA in two dimensions for the MNIST training dataset

8 Task 8

For this task we were supposed to split up the dataset into different clusters using K-means clustering. This was done dividing the data into K=2 and K=5 clusters. Since the data set is just images of ones and zeros, dividing it into 5 clusters is a bad idea, due to the fact 5 clusters representing two classes might cause some confusion. As we can see in the two figures below showing how the clusters group the datapoints using two and five clusters.



(a) K-means clustering with K=2 for MNIST training dataset



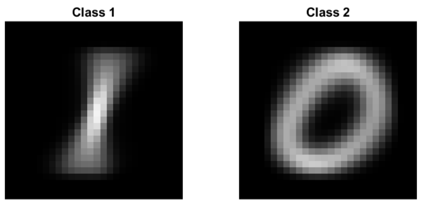
(b) K-means clustering with K=5 for MNIST training dataset

Figure 2: Plots for task 8

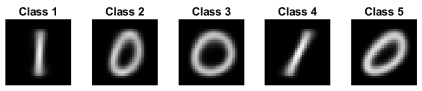
As wee can see, using two clusters works a lot better and shows strong resemblance to the dataset in figure 1. This is a drawback when using K-means clustering which is clearly seen in figure 2b, if we do not know how many classes we have the classification task becomes much harder since we need to initialize with the correct amount of clusters. An interesting notw is that the clusters in figure 2b overlap, which should not be possible since we assign a datapoint to a cluster based on its distance to the centerpoints of the clusters. The reason for the overlap is that we take distance before dimension reduction hence some points prior to reduction will be close to another cluster after reduction. When running the code for 5 clusters multiple times I also got different cluster representation, sometimes the left part of the figure was divided into three clusters and sometimes the right part, this due to the fact that K-means clustering is sensitive to the initialization of the center points of the clusters, which is also a set back to method.

9 Task 9

For this task we were just supposed to show the images of the two and five reconstructed images which is done below in figure 3.



(a) Reconstructed mages representing centers of the two clusters



(b) Reconstructed images representing centers of the five clusters

Figure 3: Plots for task 9

As we can see in figure 3, using five clusters will divide the two acutal classes into subclasses, for the ones we have one almost strictly vertical and one more tilted. For the zeros we have one vertical, one tilted and one which is more circular. Here we force the model to make 5 classes out of two and hence we get the result we get in figure 3b. One thing worth noting here is that the images in 3a are much more blurry than the ones in 3b, this is due to the fact the centers of the clusters for K=2 has to cover much more datapoints than for K = 5 and will therefor be more general.

10 Task 10

Now putting the K-means clustering to the test to see how well it performs on both the training and test data.

Table 1: K-means classification results

Training data	Cluster	# '0'	# '1'	Assigned to class	# misclassified
	1	114	6736	1	114
	2	5809	6	0	6
$N_{\text{train}} = 12665$				Sum misclassified:	120
				Misclassification rate (%):	0.95
Testing data	Cluster	# '0'	# '1'	Assigned to class	# misclassified
	1	11	1136	1	11
	2	968	0	0	0
$N_{\text{test}} = 2115$				Sum misclassified:	11
				Misclassification rate (%):	0.52

AS we can see in table 1 the classification with K = 2 works really well, for the training data we have around 99 percent accuracy and for the testdata it works even better at around 99.5 percent accuracy, this is usually not the case but since we are doing unsupervised training the performance on the test and training data will be roughly the same. Which will be shown in the next task.

11 Task 11

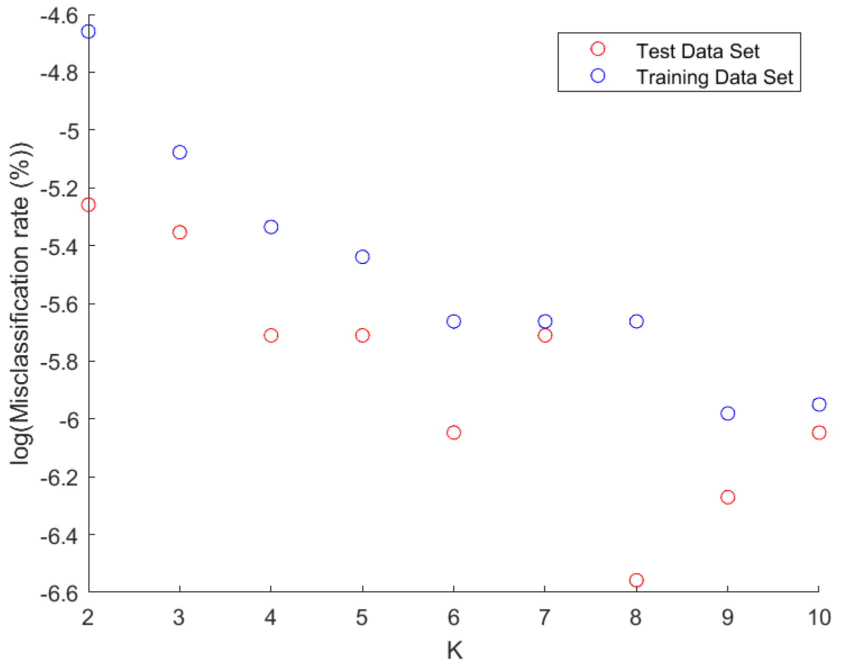


Figure 4: Log of misclassifcation rate for different K's ranging from 2 to 10

AS we can see in figure 4 the misclassifcation rate decreases as we increase the number of clusters. For the testdata we have the high misclassifcation rate at K=2 this gave around 99.5 percent accuracy which then increases to around 99.9 percent for K = 8. It does however look like the accuracy starts converging towards 99.8 percent accuracy at K=6 an onwards.

12 Task 12

Now we will train model using the MNIST dataset again but this time using the labels as a target vector during training, hence we are no longer doing unsupervised training but supervised. WE will train using a (SVM) i.e a linear support vector machine. In the table below the performance of the classifier is presented.

Table 2: Linear SVM classification results

Training data	Predicted class	True class: # ’0’	# ’1’
	’0’	5923	0
	’1’	0?	6742
$N_{\text{train}} = 12665$		Sum misclassified:	0
		Misclassification rate (%):	0
Testing data	**Predicted class**	**True class: # ’0’**	**# ’1’**
	’0’	979	1
	’1’	1	1134
$N_{\text{test}} = 2115$		Sum misclassified:	2
		Misclassification rate (%):	0.095

As we can see in table 2 the models performance is flawless on the training data, probably due to the fact that we are training on the dataset which makes it a less viable source to measure performance. However the performance on the test data is also very good with an classification accuracy above 99.9 percent.

13 Task 13

For this we trained a model using Gaussian Kernel support vector machine to do classification on the same dataset as before. TO find a good value for the hyperparamater β, since having β as 1 gave poor results on the testdata. I made a loop trying ranging the value of beta from 1 to 6 with a 0.2 interval and I stopped the loop if the model achieved a 100 percent accuracy on the testdata which it did with β being 4.8. Then I calibrated values between 4.6 and 4.8 which resulted that the optimal beta was around 4.7109. In table 3 below the results are presented in a similar table is task 12.

Table 3: Gaussian kernel SVM classification results $\beta = 4.7109$

Training data	Predicted class	True class: # ’0’	# ’1’
	’0’	5923	0
	’1’	0	6742
$N_{\text{train}} = 12665$		Sum misclassified:	0
		Misclassification rate (%):	0
Testing data	**Predicted class**	**True class: # ’0’**	**# ’1’**
	’0’	980	0
	’1’	0	1135
$N_{\text{test}} = 2115$		Sum misclassified:	0
		Misclassification rate (%):	0

14 Task 14

We can not expect the same result on new images, our hyperparamaters are tuned to fit our dataset or datasets concerning similar tasks such as the testdata set. There is a risk that we overfitted our model to become really good att recognizing zeros and ones but for new images or numbers it would perform very poorly due to the fact that we overfitted the model. We can look at it as the tradeoff between bias and variance during the training of our models. Since we are optimizing our hyperparamters with respect to the dataset we are using the model becomes bias in solving that specific task and that reduce variance and hence becomes less viable when solving similar but different task such as classifying other number for example. Hence we might have a lot worse generalization when doing new tasks from similar distributions.