FMAN45 Machine Learning -Assignment 2.

Nils Romanus

May 2022

1 Solving a nonlinear kernel SVM with hard constraints

T1

Using the non-linear feature map of $\phi(x) = [x^2 x]^T$ one may define the kernel $k(x, y) = \phi(x)^T \phi(y)$. Using the data provided in table 1 one may compute the kernel matrix $[k(x_i), k(x_j)]_{1 \leq i,j \leq 4}$ for the aforementioned data by computing the element wise product for all indices (i, j), arriving at the kernel matrix presented in 1.

$$\begin{pmatrix} 20 & 6 & 2 & 12 \\ 6 & 2 & 0 & 2 \\ 2 & 0 & 2 & 6 \\ 12 & 2 & 6 & 20 \end{pmatrix} \tag{1}$$

i	1	2	3	4
x_i	-2	-1	+1	+2
y_i	+1	-1	-1	+1

Table 1: First data set from assignement instructions

T2

The Lagrangian dual problem for the SVM is given by equation 2

$$\begin{aligned} \underset{\alpha_1, \ldots, \alpha_4}{\text{maximize}} \quad & \sum_{i=1}^{4} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{4} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \\ \text{subject to} \quad & \alpha_i \geq 0, \\ & \sum_{i=1}^{4} y_i \alpha_i = 0 \quad \forall i \end{aligned} \tag{2}$$

Under the assumption of $\alpha = \alpha_i \forall i$, the maximization problem can be simplified to expression 3.

$$\begin{aligned} \underset{\alpha}{\text{maximize}} \quad & 4\alpha - \frac{1}{2} \alpha^2 \sum_{i,j=1}^{4} y_i y_j k(x_i, x_j) \\ \text{subject to} \quad & \alpha \geq 0, \\ & \sum_{i=1}^{4} y_i \alpha = 0 \quad \forall i \end{aligned} \tag{3}$$

To solve 3 for the data listed in table 1 one may re-use the results stored in the kernel matrix 1 to compute the summation term. By using the entries of the kernel matrix coupled with the values of y_i, y_j expression 3 reduces to expression 4.

$$\begin{aligned} \underset{\alpha}{\text{maximize}} \quad & 4\alpha - \frac{1}{2}\alpha^2((20-6-2+12)+(6-2-0+2)+...) := \underset{\alpha}{\text{maximize}} \quad \mathcal{L} \\ \text{subject to} \quad & \alpha \geq 0, \\ & \sum_{i=1}^{4} y_i \alpha = 0 \quad \forall i \end{aligned} \tag{4}$$

By differentiating the lagrangian $\mathcal{L}$ with respect to α and equaling it to zero one arrives at the solution expressed in 5 yielding us an α that satisfies both conditions.

$$\frac{\partial \mathcal{L}}{\partial \alpha} = 0 \iff 4 + 36\alpha = 0 \iff \alpha = \frac{1}{9} \tag{5}$$

T3

The SVM classifier function is given by expression 6.

$$g(x) = \sum_{j=1}^{4} \alpha_j y_j k(x_j, x) + b \tag{6}$$

With the α computed in task T2 the classifier equation reduces to the simple polynomial in expression 7.

$$\alpha \sum_{j=1}^{4} k(x_j, x) + b = \frac{1}{9}(-2x + 4x^2 + x - x^2 - x - x^2 + 2x + 4x^2) + b = \frac{2x^2}{3} + b \tag{7}$$

For any support vector expression 8 holds.

$$y_s\big(\sum_{j=1}^{4} \alpha_j y_j k(x_j, x_s) + b\big) = 1 \tag{8}$$

By using any data pair from table 1 coupled with expression 8 and 7 one may compute the bias term b as shown in equation 9.

$$y_1\big(\sum_{j=1}^{4} \alpha_j y_j k(x_j, x_1) + b\big) = 1 \iff -1(\frac{2 \times (-2)^2}{3} + b) = 1 \iff b = -\frac{5}{3} \tag{9}$$

With the bias term computed one may arrive at the full expression for the classifier equation as shown in 10

$$g(x) = \frac{2x^2}{3} - \frac{5}{3} \tag{10}$$

T4

Given the data set listed in table 2 and the same kernel $k(x, y)$ one can derive the classifier equation $g(x)$ by the same methodology as before.

i	1	2	3	4	5	6	7
x_i	-3	-2	-1	0	+1	+2	+4
y_i	+1	+1	-1	-1	-1	+1	+1

Table 2: Second data set from assignment instructions

By leveraging the hint in the assignment instructions and more closely inspecting table 2 one may however note that 2 bears a striking similarity to the data set in 1. One may note that the "class shift" in table 2 occurs for the exact same x values as in table 2, i.e $x = \pm 1$. We will hence get the same "decision boundary" as before - the support vectors are the same. Since we are using the same kernel the classifier becomes the exact same. The solution to the classifier equation is therefore found in expression 10.

2 The Lagrangian dual of the soft margin SVM

T5

The primal formulation of the linear soft margin classifier is given by expression 11 where ξ denotes the collection of classification errors ξ_i

$$\begin{aligned} \underset{\mathbf{w}, b, \xi}{\text{minimize}} \quad & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0 \quad \forall i \end{aligned} \tag{11}$$

By using the original definition of the linear hard margin SVM Lagrangian, presented in 2, for an n-dimensional parameter space, and adding the new error term one arrives at equation 12 describing the soft margin Lagrangian. The first two terms in equation 12 stem from the primal forumulation, the third from the first set of constraints and the fourth from the second set of constraints.

$$\mathcal{L}(\mathbf{w}, b, \xi) = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i - \sum_{i=1}^{n}\alpha_i\big(y_i(\mathbf{w}^T\mathbf{x}_i + b + \xi - 1) - \sum_{i=1}^{n}\lambda_i\xi_i \tag{12}$$

Subject to $\alpha_i, \xi_i \geq 0$. One may now formulate the optimization problem as 13

$$\underset{\alpha_1,\ldots,\alpha_n}{\text{maximize}} \quad \underset{\mathbf{w},b,\xi}{\text{minimize}}\mathcal{L} \tag{13}$$

As with any optimization problem KKT conditions are derived by equaling the components of the gradient of the lagrangian to 0, yielding us equations 14, 15 and 16.

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i = 0 \iff \mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \tag{14}$$

$$\frac{\partial \mathcal{L}}{\partial b} = -\sum_{i=1}^{n} \alpha_i y_i = 0 \iff \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{15}$$

$$\frac{\partial \mathcal{L}}{\partial \xi_i} = C - \lambda_i - \alpha_i \iff C = \lambda_i + \alpha_i \tag{16}$$

With the constraint $0 \leq \alpha_i \leq C$. By using expression 14, describing the weights, in the original expression 12 for the Lagrangian, one arrives at equation 18.

$$\underset{\mathbf{w},b,\xi}{\text{minimize}} \mathcal{L} = \frac{1}{2} || \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i ||^2 + C \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{n} \alpha_i \big(y_i (\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i^T \mathbf{x}_i + b + \xi - 1) - \sum_{i=1}^{n} \lambda_i \xi_i \tag{17}$$

$$= -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j + C \sum_{i=1}^{n} \xi_i (C - \alpha_i - \lambda_i) + \sum_{i=1}^{n} \alpha_i (1 - y_i b) \tag{18}$$

Using conditions 15 and 16 equation 18 simplifies to expression 19

$$\underset{\mathbf{w},b,\xi}{\text{minimize}} \mathcal{L} = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \tag{19}$$

Finally resulting in expression 20 when using 19 in 13, which is an n-dimensional copy of expression 2 with a further constraint on α_i.

$$\begin{aligned} \underset{\alpha_1 \ldots \alpha_n}{\text{maximize}} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C, \\ & \sum_{i=1}^{n} y_i \alpha_i = 0 \quad \forall i \end{aligned} \tag{20}$$

T6

Using the complementary slackness of the KKT conditions one may show that support vectors with $y_i(\mathbf{w}^T \mathbf{x}_i + b) < 1$ have coefficient $\alpha_i = C$. For these support vectors we know that $\xi_i > 0$ as given by the first condition in 11. Therefore the term containing $\sum_{i=1}^{n} \lambda_i \xi_i$, corresponding to the aforementioned condition, in the Lagrangian expression 12, vanishes. Or in other words $\lambda_i = 0$. Using the KKT condition derived in 16 one can see that the these support vectors must have $\alpha_i = C$ under this condition.

3 Dimensionality reduction on MNIST using PCA

3.1 E1

By transforming the provided subset of the MNIST training data to make it zero-mean and performing a single value decomposition one may compute a 2 dimensional PCA. The result from this PCA is vizualised in figure 1. The figure shows two distinct clusters with some overlap amongst the groups.

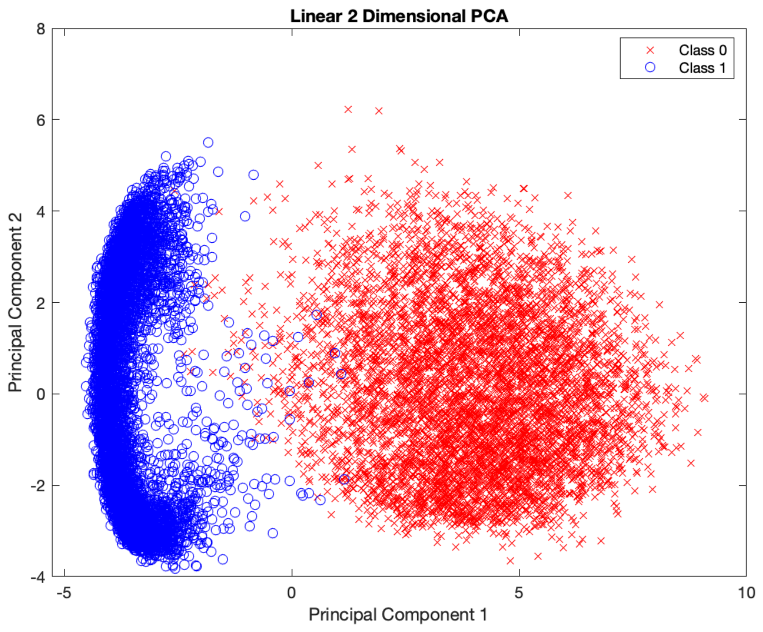

Figure 1: traindata01 projected on its two principal components using a 2 dimensional linear PCA.

4 Clustering of unsupervised data using K-means

By using unsupervised K means one may cluster the provided subset of the MNIST training data. Using the same pre-trained PCA from task E1, one may visualise the results from this clustering in two dimensions. Figure 2 displays K-means clustering using $K = 2$ clusters with centroids marked in black. The results in figure 2 are very similar to the visualisation in figure 1 indicating that the clustering results in a similar grouping as the ground truth classes. However we can see some difference for the samples in between that have been assigned to the 'wrong' cluster, indicating that we will have some missclassification if we use K means for classification. Figure 3 shows the results from a similar clustering run but with $K = 5$ clusters. We here see a greater overlap between clusters.This is due to the fact that we are projecting the clusters on the two principal components. In the original dimension there is no overlap between groups however since we are projecting this grouping on to a lower dimensional space we see an overlap in this view. Had we performed the PCA before the $K = 5$-means clustering there would be no overlap, however we would loose information that might have been useful in the clustering.

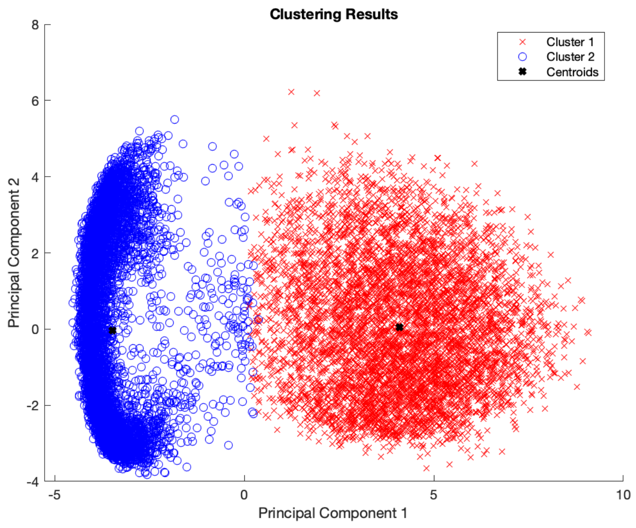

Figure 2: 2 K means clusters projected on the principal components of traindata01 with cluster centroids marked in black

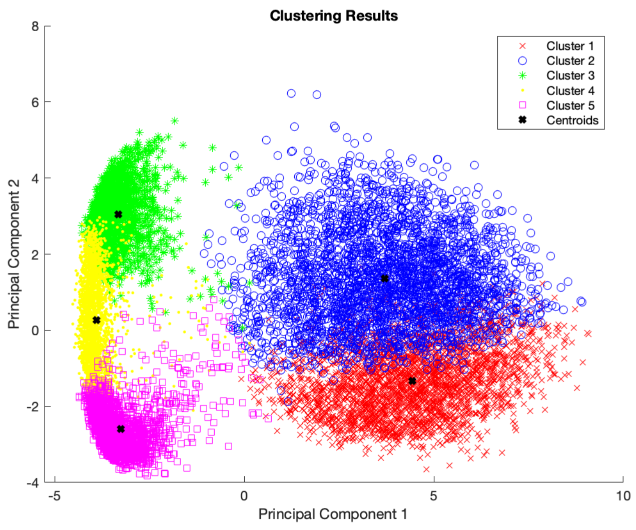

Figure 3: 5 K means clusters projected on the principal components of train-data01 with cluster centroids marked in black

E3

Instead of using the PCA vizualisations one might instead display the centroids as images. Figure 4 shows $K = 2$ subplots of the centroids from each cluster as an image. The clustering displayed in figure 4 is congruent with the human intuition for which cluster the image should belong to. Figure 5 shows a similar visualisation with $K = 5$ clusters. To the human eye this clustering is slightly counter-intuitive as one would think that all the ones should be in one cluster and all the zeros in another. This displays the limitations of the K means method. As shown in figure 5, not knowing how many clusters are appropriate before applying the algorithm can produce unintended results. However, if one is to use K-means for classification, multiple different clusters can still be given the same label, as shown in task E4.

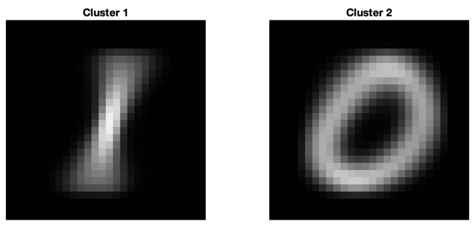

Figure 4: 2 K means centroids visualised as images

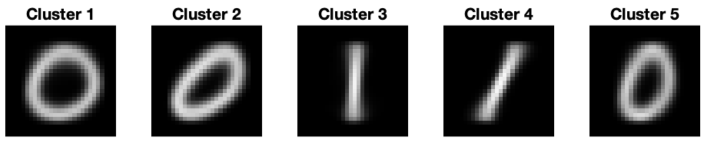

Figure 5: 5 K means centroids visualised as images

E4

By assigning each cluster(centroid) a class according the the most common class in the *training data* of the associated cluster we can convert our clustering machine to a classifier. One may then apply the classifier to the subset of the MNIST test data testdata01. Classification results from training and testing, using $K = 2$ clusters, are presented in table 3. Table 3 also contains the missclassification rate which is defined as the share of samples that have been missclassified compared to the total number of samples.

Table 3: K-means classification results using $K = 2$

Training data	Cluster	# '0'	# '1'	Assigned to class	# misclassified
	1	112	6736	1	112
	2	5811	6	0	6
$N_{\text{train}} = 12665$				Sum misclassified:	120
				Misclassification rate (%):	0.95
Testing data	Cluster	# '0'	# '1'	Assigned to class	# misclassified
	1	969	0	0	0
	2	11	1135	1	11
$N_{\text{test}} = 2115$				Sum misclassified:	11
				Misclassification rate (%):	0.52

E5

Similar to how the grid search scheme was implemented in assignment 1, one may try a different number of clusters and see if classification performance improves. Figure 6 displays the misclassification rate during training and testing for different values of K. One may note that the missclassification rate during training decreases as the number of clusters increase. On the other hand the test missclassification rate does not show a similar trend. One can interpret these figures as the classifying model becoming more over fitted to the training data as K increases. This is similar to how a polynomial linear regressor becomes more over-fitted as we allow more higher order terms. More clusters allow for a more complex pattern to be captured, however with more clusters our classifier also becomes more sensitive to noise in the training set.

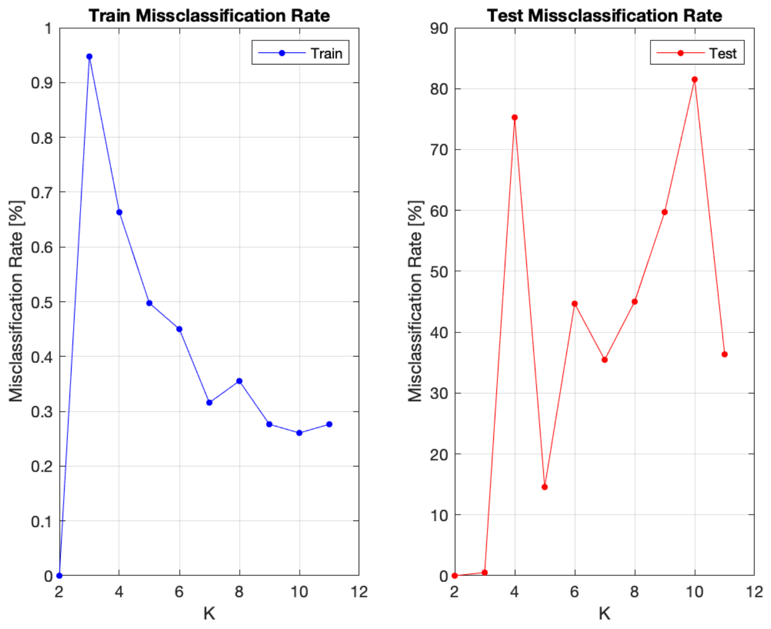

Figure 6: Misclasification rate during training and testing for different number of clusters

5 Classification of MNIST digits using SVM

E6

Using the supervised training data one may train and test a linear soft margin SVM classifier. The performance of the classifier, using default value of $C = 1$, during training and testing is shown in table 4. The SVM classifier out-performs the K-means classifier.

Table 4: Linear SVM classification results using $C = 1$

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 (%):	0.0946

E7

The created SVM can be further customized by changing the kernel from a linear one to a Gaussian one. The classification results using this new model can be viewed in table 5. One may tweak the scaling parameter σ^2 of the Gaussian Kernel by changing $\beta = \sqrt{1/\sigma^2}$ to reach optimal performance. After some trial and error, a missclassification rate of 0 % can be reached using $\beta = 5$.

Table 5: Gaussian kernel SVM classification results using $\beta = 5$

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'		5923	0
	'1'		0	6742
$N_{\text{test}} = 2115$			Sum misclassified:	0
			Misclassification rate (%):	0

E8

The Gaussian kernel with the specified β is tailored to fit the training data of our specified subset of the MNIST data. Despite performing well on these specific train and test sets this, result will not necessarily generalize to other subsets of MNIST. Despite performing well on this train-test pair the model could still be over-fitted to this specific pair. Even though we are not using the test set when setting the weights of the SVM we might have over fitted our kernel configuration to fit this specific pair of training and test data. For the SVM to perform well on other data sets one might have to construct a different kernel configuration.