Home Assignment 2

Arvid Gramer

FMAN45 Machine Learning
Lunds Tekniska Högskola, Lunds Universitet
May 2023

1 Solving a nonlinear kernel SVM with hard constraints

T1: *Compute the kernel matrix* $\mathbf{K}$ *using the data from the table.*

The kernel matrix is a matrix where at each position (i, j) is the value of $k(x_i, x_j) = (x_i, x_i^2)(x_i, x_j^2)^\intercal$. The matrix from our data becomes:

$$\mathbf{K} = \begin{bmatrix} 20 & 6 & 2 & 12 \\ 6 & 2 & 0 & 2 \\ 2 & 0 & 2 & 6 \\ 12 & 2 & 6 & 20 \end{bmatrix} \tag{1}$$

T2: *Use the data to solve, by hand, the maximization problem for* α*:* The maximisation problem is:

$$\underset{\alpha_1 \ldots \alpha_4}{\text{maximize}} \quad \sum_{i=1}^{4} -\frac{1}{2} \sum_{i,j=1}^{4} \alpha_i \alpha_j y_i y_j k(x_i, x_j) \tag{2}$$

$$\text{Subject to} \quad \alpha_i \geq 0 \text{ and } \sum_{i=1}^{4} y_i \alpha_i = 0, \forall i$$

Using that $\alpha_i = \alpha \ \forall i$ we may express the problem as:

$$\underset{\alpha}{\text{maximize}} \quad 4\alpha - \frac{1}{2}\alpha^2 \sum_{i=1}^{4} \sum_{j=1}^{4} y_i y_j k(x_i, x_j) \tag{3}$$

We derive the expression w.r.t α and find that the expression is maximized when:

$$4 - \alpha \sum_i \sum_j y_j k(x_i, x_j) y_i = 4 - \alpha \begin{bmatrix} y_1 & y_2 & y_3 & y_4 \end{bmatrix} \mathbf{K} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = 0 \tag{4}$$

$$\implies 4 - 36\alpha = 0 \implies \alpha = 1/9$$

T3 *For the data-target pairs, use the value of* α *that you found in task T2 to reduce the classifier function to the simplest possible form. This gives you a simple polynomial in* x. We know that for any support vector x_s we have:

$$y_s \left(\sum_{j=1}^{4} \alpha_j y_j k\left(x_j, x_s\right) + b \right) = 1 \tag{5}$$

and that the classifier equation is given by:

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

Using equation 6 and inputting our kernel function, $\alpha = 1/9$ and known values x_j we find:

$$g(x)=\alpha\sum_{j=1}^{4} y_j(x_j x+x_j^2x^2)+b=\frac{3}{2}x^2+b. \tag{7}$$

To find b we use equation (5) for any of the values of x: $-2/3+b=1 \implies b=-5/3$. This gives: $g(x)=3/2x^2-5/3$.

T4: *With the same kernel k(x, y) as above, what is the solution g(x) of the nonlinear kernel SVM with hard constraint on the given, extended dataset? You must show all the steps towards obtaining your solution.*

When thinking a bit about the hint, and studying the new data points $(x_i, y_i) = (-3, 1),\ (0, -1)(4, +1)$, one might notice that they are well inside the old decision boundaries. A plot of the old and new data points, as well as the classifier function is found in figure 1. The first set of data points are marked with blue and the new set are marked with red. The classifier function is marked as well. Since all the new data points end up within the decision boundaries, the classifier function $g(x)$ will not change and remain as $g(x) = 3/2x^2 - 5/3$.

2 The Lagrangian dual of the soft margin SVM

T5: *Show that the Lagrangian dual problem for the linear soft margin classifier*

$$\underset{\mathbf{w},b,\boldsymbol{\xi}}{\text{minimize}}\frac{1}{2}\|\mathbf{w}\|^2+C\sum_{i=1}^{n}\xi_i \tag{8}$$

$$\begin{aligned}\text{subject to } & y_i\left(\mathbf{w}^\top\mathbf{x}_i+b\right)\geq 1-\xi_i\\ & \xi_i\geq 0\end{aligned}$$

is given by:

$$\underset{\alpha_1,\ldots,\alpha_n}{\text{maximize}}\sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^\top\mathbf{x}_j \tag{9}$$

$$\text{subject to } 0\leq\alpha_i\leq C \tag{10}$$

$$\sum_{i=1}^{n}\alpha_i y_i=0 \tag{11}$$

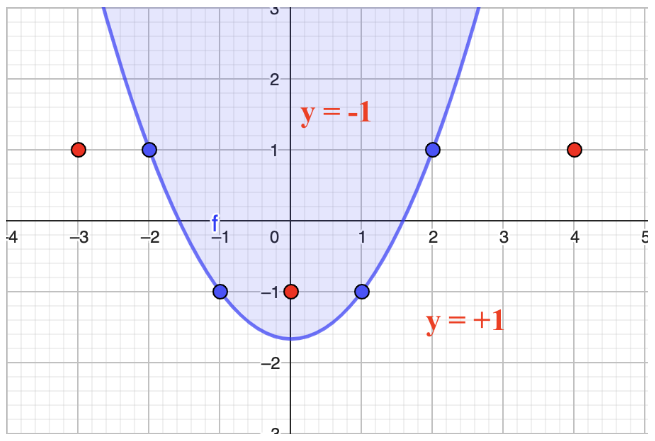

Figure 1 A plot of the first set of data points (in blue, the support vectors), as well as the new data points (in red) and the old classifier function. One may notice that the data points lie within the old decision boundaries and the classifier function will thus remain the same even after adding the new data points.

We begin by identifying the standard form. This is expressed as:

$$\begin{aligned} \text{minimize} \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq 0,\ i = 1...m \\ & h_j(x) = 0,\ \ j = 1...q \end{aligned}$$

from this we find the Lagrangian $\mathcal{L}$ as:

$$\mathcal{L} = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{j=1}^{q} \alpha_j h_j(x).$$

Identifying terms in our problem we find our Lagrangian to be:

$$\mathcal{L} = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i - \sum \alpha_i (y_i \left(\mathbf{w}^\top \mathbf{x}_i + b\right) + \xi_i - 1) - \sum \lambda_i \xi_i \tag{12}$$

The primal problem is to minimize this function, and then for the dual problem we want to maximize that minima w.r.t α and λ. From the KKT-conditions we want the gradient $\nabla\mathcal{L} = 0$. This gives the following conditions:

$$\frac{\partial \mathcal{L}}{\partial w} = w - \sum_i \alpha_i y_i \mathbf{x_i} \implies \qquad w = \sum \alpha_i y_i \mathbf{x_i} \tag{13}$$

$$\frac{\partial \mathcal{L}}{\partial b} = \sum_i \alpha_i y_i \implies \qquad \sum_i \alpha_i y_i = 0 \tag{14}$$

$$\frac{\partial \mathcal{L}}{\partial \xi_i} = C - \alpha_i - \lambda_i \implies \qquad C = \lambda_i + \alpha_i \tag{15}$$

Inserting condition (13) into the Lagrangian we get:

$$\begin{aligned}\mathcal{L} &= \frac{1}{2}\|\Sigma_i\alpha_i y_i \mathbf{x_i}\|^2 + C\Sigma_i\xi_i \\ &\quad - \Sigma_i\alpha_i((y_i\left((\Sigma_j\alpha_j y_j\mathbf{x_j})^\top \mathbf{x}_i + b\right) + \xi_i - 1) - \Sigma\lambda_i\xi_i \\ &= \frac{1}{2}(\Sigma_j\alpha_j y_j\mathbf{x_j})^\top(\Sigma_i\alpha_i y_i\mathbf{x_i}) - \Sigma_i\alpha_i\left(y_i\left((\Sigma_j\alpha_j y_j\mathbf{x_j})^\top\mathbf{x_i} + b\right) + \xi_i - 1\right) \\ &\quad + C\Sigma_i\xi_i + \Sigma_i\lambda_i\xi_i \\ &= \frac{1}{2}(\Sigma_j\alpha_j y_j\mathbf{x_j})^\top(\Sigma_i\alpha_i y_i\mathbf{x_i}) - (\Sigma_j\alpha_j y_j\mathbf{x_j})^\top(\Sigma_i\alpha_i y_i\mathbf{x_i}) - \Sigma_i\alpha_i\xi_i + \Sigma_i\alpha_i \\ &\quad + C\Sigma_i xi_i + \Sigma_i\lambda_i\xi_i - \Sigma_i\alpha_i y_i b \\ &= -\frac{1}{2}\Sigma_j\Sigma_i\alpha_i\alpha_j y_i y_j \mathbf{x_i}^\top\mathbf{x_j} - b\Sigma_i\alpha_i y_i + \Sigma_i\alpha_i + \Sigma_i\xi_i(C - \alpha_i - \lambda_i). \end{aligned} \tag{16}$$

Using the final two conditions (14) and (15) we all terms but two and we have minimized the primal problem. We now wish to maximize the last part w.r.t α and we therefore get a dual problem:

$$\underset{\alpha_1,\dots,\alpha_n}{\text{maximize}} \quad \Sigma_i\alpha_i - \frac{1}{2}\Sigma_j\Sigma_i\alpha_i\alpha_j y_i y_j\mathbf{x_i}^\top\mathbf{x_j} \tag{17}$$

T6: Complementary slackness in the KKT-conditions imply that $\lambda_i f_i = 0,\ i = 1, ..., m$. In our case $f_i = -\xi_i$, which yields $\lambda_i\xi_i = 0$. If this is inserted into the last term in equation (16), we get

$$\sum_i \xi_i(C - \alpha_i - \lambda_i) = \sum_i \xi_i(C - \alpha_i) \tag{18}$$

Since $y_i\left(\mathbf{w}^\top\mathbf{x}_i + b\right) < 1$ then $\xi_i > 1$. If this is the case the only way that term may vanish from the equation is if $C = \alpha_i$, which then needs to be fulfilled.

3 Dimensionality reduction on MNIST using PCA

E1: *Compute the linear PCA and visualize the training data in two dimensions.*

A PCA-function is implemented using Matlabs function `SVD`. The resulting projection of the data is displayed in figure 2.

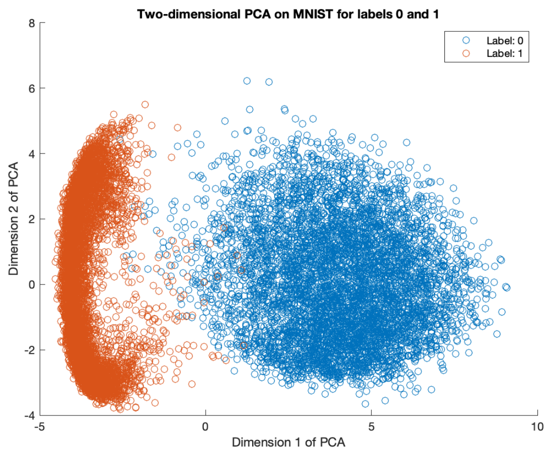

Figure 2 The training data from `MNIST_01` dataset projected on two principal components.

4 Clustering of unsupervised data using K-means

E2: *Implement K-means clustering using the provided skeleton. Display the resulting clusters and centroids for $K = 2$ and $K = 5$.*

The provided skeleton is filled in. As a distance metric between two points I used the standard L2-norm. The clusters overlap due to the dimensionality reduction. We are projecting $28 \times 28 = 784$ dimensions onto 2, which of coarse create some overlap. If you stand on the ground and an airplane is passing over you in the sky, it is not crashing into you despite it's shadow passing through yours. In the full dimensional 784 there is only overlap if it is exactly the same picture. For $K = 5$ we are also trying to cluster two classes into 5 clusters, which creates more overlap since we are trying to make a distinction between more similar images. The plot of the 2- and 5-means clusters are found in figure 3 and 4, respectively.

E3: *Display the $K = 2$ and $K = 5$ centroids as images.*

The values of all 784 coordinates of the centroids are displayed using Matlab's `imshow()`. The resulting images for $K = 2, 5$ are found in figure 5 and figure 6, respectively. We see that when clustering the data into more clusters than there are labels, as for $K = 5$ we get several centroids that we as humans would label as the same number. These correspond to the two clusters that overlap on the right hand side of figure 4, as zeros, and the three on the left hand side of figure 4 as ones.

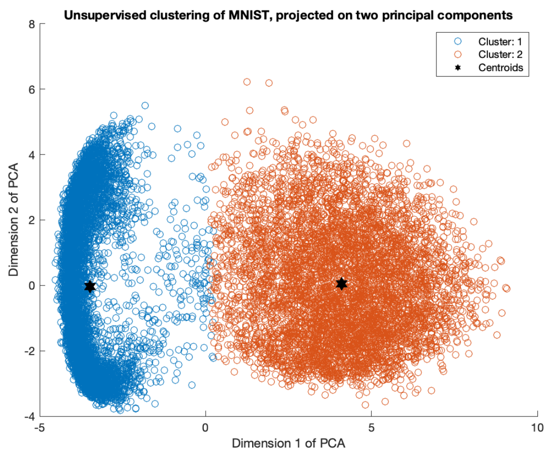

Figure 3 K = 2-means clustering on `MNIST_01` dataset and then projected onto 2 principal components. The method manages to make a good distinction between the two classes.

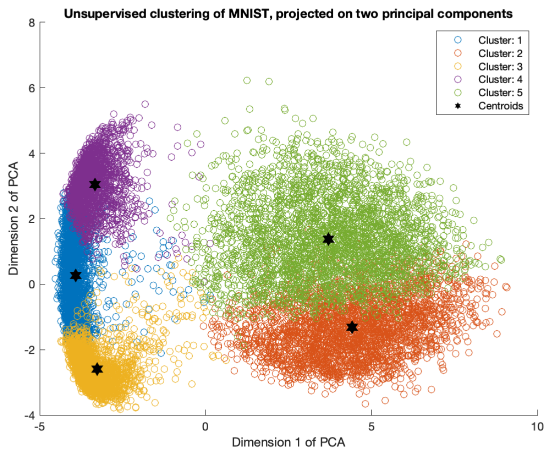

Figure 4 A display of K = 5-means clustering on `MNIST_01` dataset and then projected onto 2 principal components. The method manages to make a fairly good distinction between the two classes, but the overlap is more severe than for when only two clusters are created. This has to do with the fact that we actually just have two classes, so making it divide into more clusters is a bit forced. We also have to do with 784-dimensional data being projected onto two dimensions.

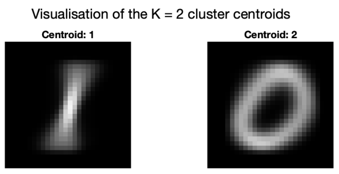

Figure 5 Displaying the two K = 2 centroids as images. It makes a clear distinction between the two classes

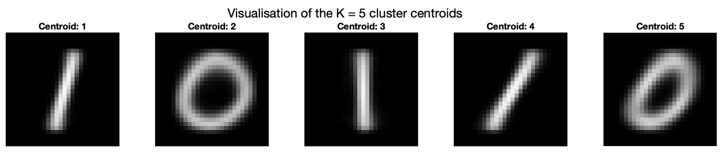

Figure 6 Displaying the two K = 5 centroids as images. We see that it is forced to create 5 clusters from the two classes, since the three clusters on the left side of figure 4 all are depicting ones.

E4: *Implement K-means clustering for classification.*

A function `K_means_classifier` is implemented. It returns the label of the centroid closest to each sample. The classifier takes as input the sample, the coordinates of the centroids and their respective label. One can therefore view the classifier as being trained outside the function. The performance of this classifier can be found in table 1.

Table 1 K-means classification results

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:	118
				Misclassification rate (%):	0.93
Testing data	Cluster	# '0'	# '1'	Assigned to class	# misclassified
	1	12	1135	1	12
	2	968	0	0	0
$N_{\text{test}} = 2115$ 12				Sum misclassified:	12
				Misclassification rate (%):	0.57

E5: *Test other K values? Can you lower misclassification rate further?*

In order to investigate the matter we perform a grid search where we measure the

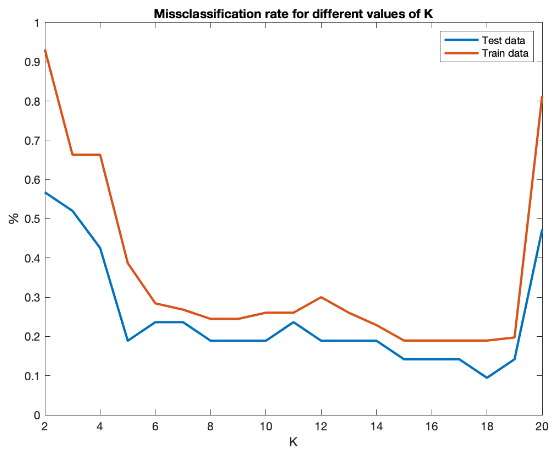

Figure 7 The missclassification rate for different values of hyperparameter K. One would expect the training and testing values to diverge for high levels of K since many clusters is a way of overfitting the model.

misclassification rate for different values of K. The misclassification rate, both on training and testing data, is then plotted against the value of $K = [1, 20]$. We expect the model to perform better for higher K, especially on training data, but that the test data would diverge for too high K since this would be overfitting the model. We see that this is the case for $K > 20$. What is interesting is that the misclassification is higher on train than on test data, which is uncommon. For large Ks the fitting and classification process is quite computationally expensive which prevents a more extensive grid search.

E6: *Train a linear SVM classifier.*

The result from testing the classifier on training and testing data is found in table 2. The classifier performs very well, with only two misclassifications on test data.

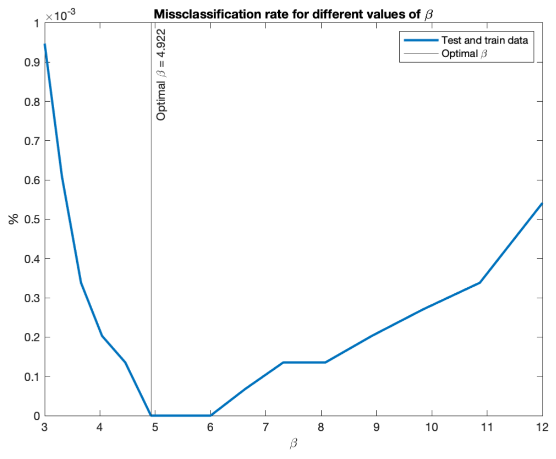

Figure 8 Missclassification rate for different values of hyperparameter β. The optimal β is clearly between 5 and 6.

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

E7: *Deploy a Gaussian kernel SVM and find an optimal value of hyperparameter β*

I begin by grid searching a wide range of values of $\beta = 1/\sqrt(\sigma^2)$, logaritmically distributed between 0.01 and 20. This grid search indicate that we should search between 3 and 12, so we narrow our search down to this range, still logaritmically distributed. This grid search show that there are values of β that all yield missclassification rates at 0%, around $\beta = 5$. A plot of this grid search is found in figure 8. The assessment of the Gaussian kernel SVM-model with $\beta = 4.922$ and its predictions is found in table 3.

Table 3 Gaussian kernel 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'	980	0
	'1'	0	1135
$N_{\text{test}} = 2115$		Sum misclassified:	0
		Misclassification rate (%):	0

E8: *Can we expect the same performance on new images?*

The performance on test data is often viewed as a proxy for how well the model will perform on unseen data. However, in this case, we have chosen hyperparameters using performance on test data, meaning that it is not unseen and we can not expect this. The model could therefore be overfitted. Furthermore, even if the test data was not subject to *data leakage*, that is using data that should be unseen, we would still not be able to expect the same performance. Things can happen so that new data no longer can be seen as samples from the same distribution as our training and testing data, such as a drift in peoples handwriting or new samples from a person that writes much smaller etc.