Machine Learning, LTH 2021

Assignment 2: Classification Using Support Vector Machines and K-means Clustering

Name: Isabelle Frodé
Date: April 29 2021

Assignment Overview

The assignment was given in the course Machine Learning FMAN45 at LTH spring 2021. The goal with the assignment was to study supervised and unsupervised classifiers. In the theory part of the assignment a non-linear kernel Support Vector Machine (SVM) algorithm for classification with both hard and soft constraint was studied. The Lagrange multiplier α for a SVM with hard constraints was calculated numerically using given data. The Lagrange dual problem was also derived from the primal formulation for the SVM with soft constraints. In the experimental Matlab part of the assignment classification of unsupervised data was performed using K-means clustering and classification. Classification was also performed using both supervised linear- and non-linear Gaussian kernel SVM classifiers.

Theory

Support Vector Machine (SVM) is an algorithm commonly used for classification. It is a supervised machine learning algorithm that use a kernel, set of mathematical functions, to transform the data in order to find the optimal boundary between potential outputs.

Linear hard margin SVM

T1. Calculating the Kernel Matrix

The linear hard margin SVM is defined as the solution to the optimization problem

$$\begin{aligned} &\underset{\mathbf{w},b}{\text{minimize}} \ \ \frac{1}{2}||\mathbf{w}||^2 \\ &\text{subject to} \ \ y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \ \ \forall i \end{aligned} \tag{1}$$

where $\mathbf{w} \in \mathbb{R}^d$ is the parameters that we want to optimize together with the bias $b \in \mathbb{R}$. y_i is the class which the data point $\mathbf{x}_i$ belongs to. When linear SVM is used for binary classification the parameters should be chosen so that $\mathbf{w}^T\mathbf{x}_i + b \geq 1$ if $\mathbf{x}_i$ belongs to the class $y_i = +1$ and $\mathbf{w}^T\mathbf{x}_i + b \leq -1$ if $\mathbf{x}_i$ belongs to the class $y_i = -1$.

The *margin* refers to the distance between the decision surface/line and the closest data point. The desirable largest margin is obtained as equation 1 is optimized.

A one-dimensional binary classification problem that is not linearly separable was given in the assignment description [1]. The dataset is presented in table 1.

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

Table 1: Dataset containing data points x_i and their class beloning y_i

To solve the classification problem a kernel $k(x, y) = \phi(x)^T\phi(y)$ and feature map $\phi(x) = (x \ x^2)^T$ is introduced. The kernel matrix $\mathbf{K}$ can then be calculated

$$k(x_1, x_1) = \phi(x_1)^T\phi(x_1) = x_1 \cdot x_1 + x_1^2 \cdot x_1^2 = 20, etc. \tag{2}$$

$$\mathbf{K} = [k(x_i, x_j)]_{1\leq i,j\leq 4} = \begin{bmatrix} k(x_1, x_1) & .. & k(x_1, x_4) \\ : & \ddots & \\ k(x_4, x_1) & & k(x_4, x_4) \end{bmatrix} = \begin{bmatrix} 20 & 6 & 2 & 12 \\ 6 & 2 & 0 & 2 \\ 2 & 0 & 2 & 6 \\ 12 & 2 & 6 & 20 \end{bmatrix} \tag{3}$$

T2. Solving the Langrangian Dual Problem

The Langrangian dual problem for the hard margin SVM was given in the assignment description [1] and follows:

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

For the data given in Table 1 the solution satisfy that $\alpha = \alpha_1 = \alpha_2 = \alpha_3 = \alpha_4$. Equation 3 could be solved numerically by differentiating with respect to α and then setting equal to 0 to maximize:

$$\begin{aligned} &\frac{d}{d\alpha}(\alpha - \frac{1}{2} \sum_{i,j=1}^{4} \alpha^2 y_i y_j k(x_i, x_j)) = 4 - \alpha \sum_{i,j=1}^{4} y_i y_j k(x_i, x_j) \\ &\implies \alpha = \frac{4}{\sum_{i,j=1}^{4} y_i y_j k(x_i, x_j)} = \frac{1}{9} \end{aligned} \tag{5}$$

T3. Reduce the Classifiers Function

The following classifier equation is the solution to the optimization problem in equation 1:

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

and for any support vector x_s it holds that

$$y_s \left(\sum_{j=1}^{4} \alpha_j y_j k(x_j, x) + b \right) = 1 \tag{7}$$

The equation for the classifier for the dataset in Table 1 can be reduced by inserting the kernel and using that $\alpha_j = \alpha$:

$$\begin{aligned} &g(x) = \sum_{j=1}^{4} \alpha_j y_j k(x_j, x) + b = \sum_{j=1}^{4} \alpha_j y_j \phi(x_j)^T \phi(x) + b \mid_{\alpha_j = \alpha} = \\ &= \alpha \sum_{j=1}^{4} y_j (x_j \cdot x + x_j^2 \cdot x^2) + b = \alpha \left(x \sum_{j=1}^{4} y_j x_j + x^2 \sum_{j=1}^{4} y_j x_j^2 \right) + b \end{aligned} \tag{8}$$

b can be calculated using the support vector equation 7 and inserting arbitrary data pair as support vector from Table 1, for example $(x_s, y_s) = (x_1, y_1)$:

$$\begin{aligned} b &= \frac{1}{y_s} - \sum_{j=1}^{4} \alpha_j y_j k(x_j, x_s) = \frac{1}{y_1} - \frac{1}{9} \sum_{j=1}^{4} y_j k(x_j, x_1) \\ &= \frac{1}{1} - \frac{1}{9} \left(1 \cdot 20 - 1 \cdot 6 - 1 \cdot 2 + 1 \cdot 12\right) = -\frac{5}{3} \end{aligned} \tag{9}$$

After inserting the data from Table 1 and the calculated α and b the solution g(x) on the simplest possible form could be written:

$$g(x) = \frac{1}{9}(0 \cdot x + 6 \cdot x^2) + b = \frac{2}{3}x^2 - \frac{5}{3} \tag{10}$$

T4. Classifiers Function Similar Dataset

Another one-dimensional binary classification problem, not linearly separable was given in the assignment description [1]. The data is presented in Table 2.

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: Dataset containing data points x_i and their class beloning y_i

The dataset contains the same data as the dataset in Table 1, with 3 extra pairs: $(x_1, y_1) = (-3, +1)$, $(x_4, y_4) = (0, -1)$ and $(x_7, y_7) = (4, +1)$. The solution g(x) calculated in the previous section is assumed to hold for this data aswell, keeping the same kernel. Testing the hypothesis for the new points gives:

$$\begin{aligned} g(x_1 = -3) = \frac{2}{3}(-3)^2 - \frac{5}{3} = \frac{13}{3} > 1 &\implies y_i = +1 \\ g(x_4 = 0) = \frac{2}{3}0^2 - \frac{5}{3} = -\frac{5}{3} < 1 &\implies y_i = -1 \\ g(x_7 = 4) = \frac{2}{3}(4)^2 - \frac{5}{3} = 9 > 1 &\implies y_i = +1 \end{aligned} \tag{11}$$

Equation 11 show that the solution g(x) calculated for the small dataset also holds for the larger dataset in Table 2 even though the new pairs are not support vectors.

Langrangian Dual of the Soft Margin SVM

The difference between hard margin and soft margin SVM is that soft margin SVM allow errors $\boldsymbol{\xi}$ in the classification. Each data point $\mathbf{x}_i$ has a corresponding classification error ξ_i. The linear soft margin SVM is defined as the solution to the optimization problem

$$\begin{aligned} \underset{\mathbf{w},b,\boldsymbol{\xi}}{\text{minimize}} \;\; & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} \;\; & y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i \\ & \xi_i \geq 0 \end{aligned} \tag{12}$$

T5. Derivation of the Langrangian Dual Problem

Langrange is a useful tool for solving constrained optimization problems. The primal optimization problem is then converted into a primal Lagrange optimization problem and thereafter into a dual Lagrange to easier apply the kernel. In soft margin SVM there are two constrains, hence we apply two Lagrange multipliers α and β to construct a new optimization problem from equation 13.

The first step in the derivation was to rewrite the optimization problem to a generalized Lagrange formulation $\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta})$. In addition to the former formulation in equation 13 with the objective function $f(\mathbf{w})$, it will also contain the first inequality constraint, i.e. the KKT conditions $g_i(\mathbf{w})$ with Lagrange multiplier $\boldsymbol{\alpha}$ and the second inequality constraint $h_i(\mathbf{w})$ with Lagrange multiplier $\boldsymbol{\beta}$:

$$f(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}\xi_i \tag{13}$$

$$g_i(\mathbf{w}) = 1 - \xi_i - y_i(\mathbf{w}^T\mathbf{x}_i + b) \leq 0 \tag{14}$$

$$h_i(\mathbf{w}) = -\xi_i \leq 0 \tag{15}$$

The generalized Lagrange was then be formulated:

$$\begin{aligned} \mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = f(\mathbf{w}) + \sum_{i=1}^{n}\alpha_i g_i(\mathbf{w}) + \sum_{i=1}^{n}\beta_i l_i(\mathbf{w}) = \\ = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}\xi_i + \sum_{i=1}^{n}\alpha_i[1 - \xi_i - y_i(\mathbf{w}^T\mathbf{x}_i + b)] + \sum_{i=1}^{n}\beta_i[-\xi_i] \end{aligned} \tag{16}$$

The primal part $\Theta_p(\mathbf{w})$ could also be formulated:

$$\Theta_p(\mathbf{w}) = \max_{\boldsymbol{\alpha},\boldsymbol{\beta},\alpha_i \geq 0} \mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \begin{cases} f(\mathbf{w}), & \text{if } \mathbf{w} \text{ satisfy constraints} \\ \infty, & \text{otherwise} \end{cases} \tag{17}$$

It holds for the primal part Θ_p that it has the same value as the objective in the original problem $f(\mathbf{w})$ for all values satisfying the primal constraints. If the constraints are violated the value is positive infinity. Next step in the derivation was to minimize Θ_p as in the

original problem in equation 13. Instead of maximizing with respect to $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ first I changed order and minimized with respect to $\mathbf{w}$. This allows for solving the dual part of the problem first instead of the primal.

$$\begin{aligned} \min_{\mathbf{w}} \ \Theta_p(\mathbf{w}) = & \min_{\mathbf{w}} \max_{\boldsymbol{\alpha},\boldsymbol{\beta},\alpha_i \geq 0} \ \mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \\ = & \max_{\boldsymbol{\alpha},\boldsymbol{\beta},\alpha_i \geq 0} \min_{\mathbf{w}} \ \mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \end{aligned} \tag{18}$$

The dual part $\Theta_d(\boldsymbol{\alpha}, \boldsymbol{\beta})$ could then be formulated:

$$\Theta_d(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \min_{\mathbf{w}} \ \mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) \tag{19}$$

Θ_d was solved by first differentiating with respect to $\mathbf{w}$ and also b, ξ_i for constraints. Setting equal to 0 to minimize gave the following

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^{n} \alpha_i y_i x_i = 0 \implies \mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i x_i \tag{20}$$

$$\frac{\partial \mathcal{L}}{\partial b} = -\sum_{i=1}^{n} \alpha_i y_i = 0 \implies \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{21}$$

$$\frac{\partial \mathcal{L}}{\partial \xi_i} = C - \alpha_i - \beta_i = 0 \implies C = \alpha_i + \beta_i \tag{22}$$

Inserting it in the generalized Lagrange formulation in equation 17 gives the dual form formulation. Inserting the optimal $\mathbf{w}$ in the equation gives the term

$$\begin{aligned} \frac{1}{2}\mathbf{w}^T\mathbf{w} &= \frac{1}{2}\left(\sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i\right)^T \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \sum_{j=1}^{n} \alpha_j y_j \mathbf{x}_j^T \\ &= \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_j^T \mathbf{x}_j = \frac{1}{2}K \end{aligned} \tag{23}$$

letting $K = \sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_j^T \mathbf{x}_j$ and calculating the second term:

$$C\sum_{i=1}^{n} \xi_i = \sum_{i=1}^{n} \alpha_i \xi_i + \sum_{i=1}^{n} \beta_i \xi_i \tag{24}$$

The third term with the optimal $\mathbf{w}$:

$$\sum_{i=1}^{n} \alpha_i [1 - \xi_i - y_i(\mathbf{w}^T \mathbf{x}_i + b)] =$$
$$= \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \alpha_i \xi_i - \sum_{i=1}^{n} \alpha_i y_i \left(\sum_{j=1}^{n} \alpha_j y_j \mathbf{x}_j \right)^T \mathbf{x}_i - \sum_{i=1}^{n} \alpha_i y_i b \tag{25}$$
$$= \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \alpha_i \xi_i - K - b \sum_{i=1}^{n} \alpha_i y_i$$

Using that $\sum_{i=1}^{n} \alpha_i y_i = 0$ we get $b \sum_{i=1}^{n} \alpha_i y_i = 0$. Inserting in the Lagrangian formulation gives the dual part:

$$\Theta_d(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \min_{\mathbf{w}} \ \mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) =$$
$$= \frac{1}{2} K + \sum_{i=1}^{n} \alpha_i \xi_i + \sum_{i=1}^{n} \beta_i \xi_i + \sum_{i=1}^{n} \alpha_i - \sum_{i=1}^{n} \alpha_i \xi_i - K - b \sum_{i=1}^{n} \alpha_i y_i - \sum_{i=1}^{n} \beta_i \xi_i \tag{26}$$
$$= \sum_{i=1}^{n} \alpha_i - \frac{1}{2} K = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

The dual Lagrange problem could then be formulated:

$$\max_{\boldsymbol{\alpha}, \boldsymbol{\beta}, \alpha_i \geq 0} \Theta_d(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \max_{\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_n} \sum_{j=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$
$$\text{subject to } \ 0 \leq \alpha_i \leq C, \tag{27}$$
$$\sum_{i=1}^{n} \alpha_i y_i = 0$$

The constraints come from equation 22 and 23. That $\alpha_i \leq C$ follows from that β_i is allowed to be 0.

T6. Coefficient α_i of the support vectors

The constraints $g(\mathbf{w})$ and $h(\mathbf{w})$ form the KKT conditions of the problem. Complementary slackness of the KKT conditions means that

$$\sum_{i=1}^{n} \alpha_i g_i(\mathbf{w}^*) = 0 \implies \alpha_i(-y_i(\mathbf{w}^T \mathbf{x}_i + b) + 1 - \xi_i) = 0 \tag{28}$$

$$\sum_{i=1}^{n} \beta_i h_i(\mathbf{w}^*) = 0 \implies \beta_i \xi_i = 0 \tag{29}$$

where $\mathbf{w}^*$ is the optimal $\mathbf{w}$. Equation 29 show that for $\alpha_i > 0$ it holds that

$$-y_i(\mathbf{w}^T\mathbf{x}_i + b) + 1 - \xi_i = 0 \implies y_i(\mathbf{w}^T\mathbf{x}_i + b) = 1 - \xi_i \tag{30}$$

From equation 23 we have

$$C - \alpha_i - \beta_i = 0 \implies \alpha_i = C - \beta_i \tag{31}$$

From equation 31 and 32 we could see that the support vector $y_i(\mathbf{w}^T\mathbf{x}_i + b) < 1$ must have $\xi_i > 0$. In order to satisfy the KKT condition in equation 30, β_i must be equal to 0. Inserting $\beta_i = 0$ in equation 32 gives $\alpha_i = C$.

Experimental Part

In this section the MNIST image dataset was studied. The dataset contains of handwritten digits divided in 12665 data-target pairs. The data is images $\mathbf{X}_i \in [0,1]^{28\text{x}28}$ and the targets is $t_i \in 0, 1, ..., 9$ but in this assignment only the images with targets $t_i \in 0, 1$ are used. The images have a dimension of 28x28 = 784 which is not a high resolution for an image.

E1. Dimensionality Reduction

A linear principal component anlysis (PCA) was computed in Matlab to reduce the dimensionality d of the MNIST dataset to $d = 2$. PCA uses singular value decomposition (SVD) which is the factorization:

$$\mathbf{X} = \mathbf{USV}^T \tag{32}$$

for a matrix $\mathbf{X} \in \mathbb{R}^{D\text{x}N}$. For $P = \min(D, N)$ it holds that $\mathbf{U} \in \mathbb{R}^{D\text{x}P}$ is the left singular vectors and $\mathbf{S} = \text{diag}(\mathbf{s}), \mathbf{s} \in \mathbb{R}^P$ the diagonal matrix of P many singular values. $\mathbf{V} \in \mathbb{R}^{N\text{x}P}$ is further the right singular vectors. The dimensionality of $\mathbf{X}$ is reduced through a projection of $\mathbf{X}$ onto the *d=2* left singular vectors with the largest absolute singular values. The linear PCA is visualized in Figure 1.

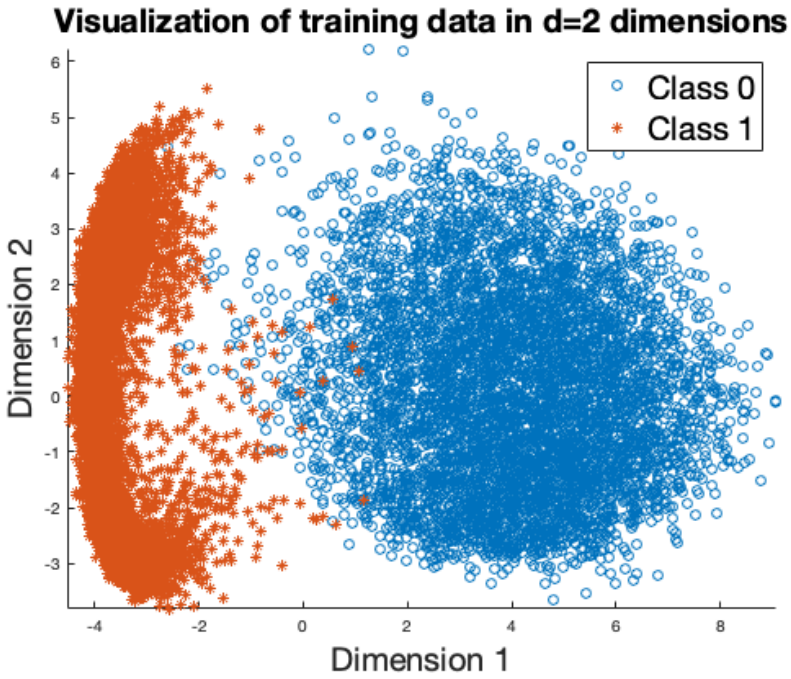

Figure 1: Visualization of training data in d=2 dimensions using PCA

K-means Clustering of Unsupervised Data

E2. Implementation of K-means Clustering Algorithm

In this section a K-means clustering approach was used to cluster the images without the targets. The algorithm was run on the training data for K = 2 clusters and K = 5 clusters.

The results are shown in Figure 2-3

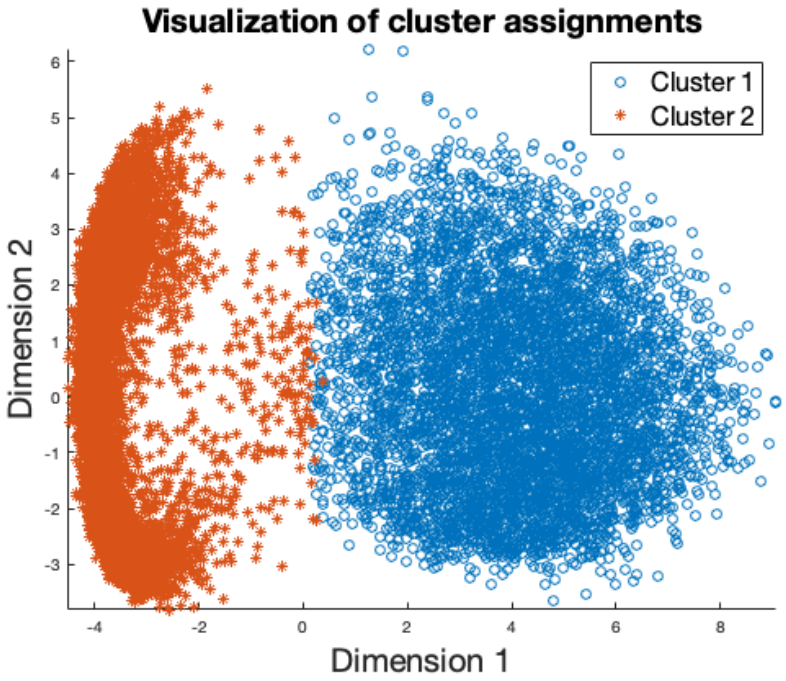

Figure 2: Visualization of cluster assignments of training data for K=2 clusters in d=2 dimensions

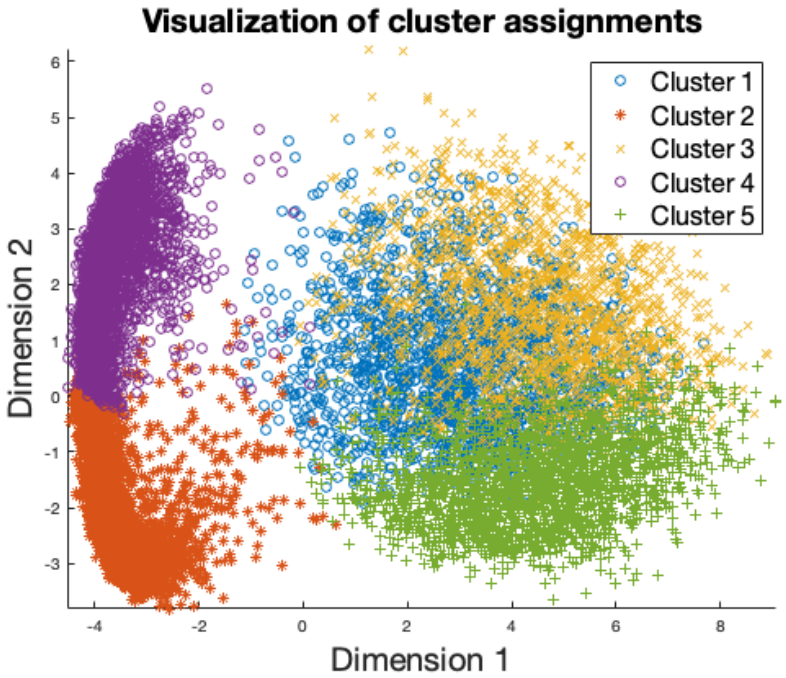

Figure 3: Visualization of cluster assignments of training data for K=5 clusters in d=2 dimensions

The clusters visualized in Figure 2-3 seem to overlap, which is a consequence of the dimensionality reduction. 2 dimensions are visualized but there are $28^2 = 784$ dimensions that influence how the algorithm cluster the data which means that there is no actual overlap in the clusters.

E3. Centroids as Images for Each Cluster

The centroids was then displayed for K=2 and K=5 clusters. The results is displayed in Figure 4 - 5.

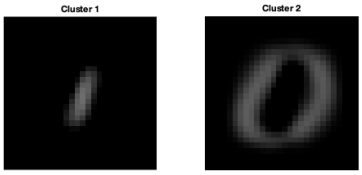

Figure 4: Centroids as images for K=2 clusters

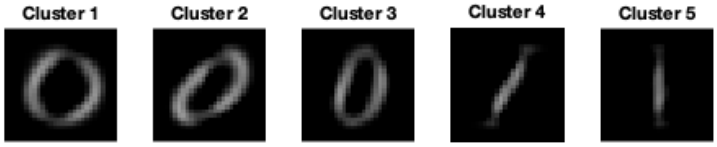

Figure 5: Centroids as images for K=5 clusters

E4. Implementation of K-means Classifier

A K-means classifier function was implemented. Each cluster centroid was assigned the label of which it had the most example in when comparing to the training data. The computed labels was compared to the training labels to count how many misclassifications there were, first for the training data and then for the test data. The results are presented in Table 3.

Table 3: 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	1135	1	11
	2	969	0	0	0
$N_{\text{test}} = 2115$				Sum misclassified:	11
				Misclassification rate (%):	0.52

E5. Misclassification Rate for Different Number of Clusters

In this section, the number of misclassifications for a few different K values was calculated. The results are presented in Figure 6.

Figure 6: Graph showing how misclassification rate depends on number of clusters

Figure 6 show that the number of misclassifications goes down when adding more clusters. The slope is steeper when we have less clusters, i.e. the difference in misclassification is larger between K=2 clusters and K=5 than between K=5 and K=10 clusters.

E6. Training a Linear SVM Classifier

In this section a linear SVM classifier was trained using the supervised training data. The built-in matlab function **fitcsvm** was used to fit a classification SVM. The function returns a model for the training data with the given training target responses. The results on the training and test data is presented in Table 4.

Table 4: 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. Training a Non-Linear Kernel SVM Classifier

In this section a non-linear SVM classifier with an Gaussian kernel was trained using the supervised training data. The built-in matlab function **fitcsvm** was used to fit a classification SVM with the Gaussian kernel and scaling parameter σ^2. The function returns a model for the training data with the given training target responses. The default scaling in the Matlab function is $\beta = \sqrt{1/\sigma^2}$. The SVM was howeve trained and evaluated for different β parameters to find the optimal value of β. The results are presented in Figure 7.

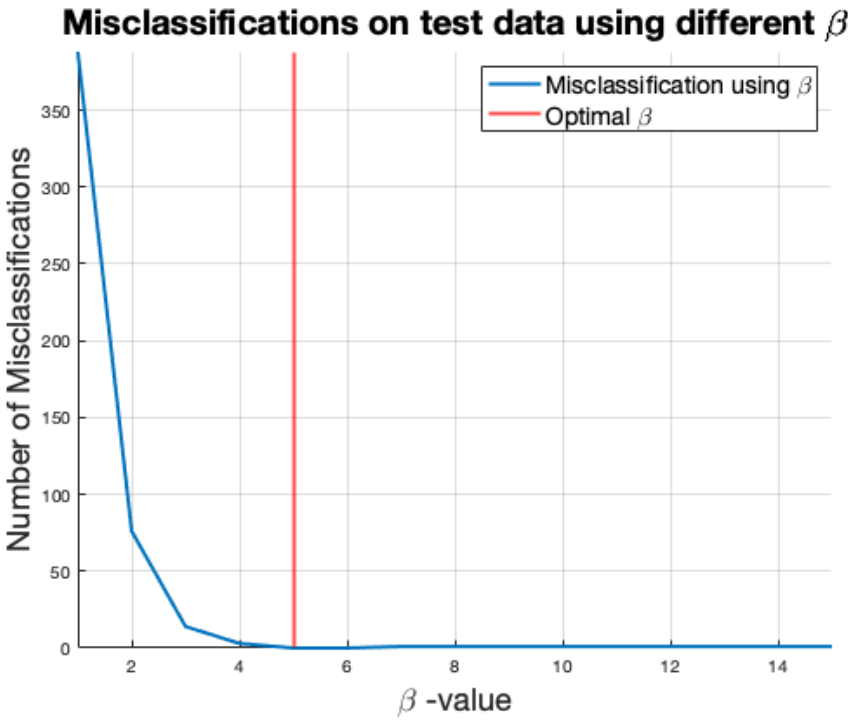

Figure 7: Graph showing how misclassification rate depends on the choice of β

Figure 7 show how the misclassification rate on the test data goes down for larger values of β. The β-values that gave the best result of no misclassifications at all was for $\beta = 5$ and $\beta = 6$, hence $\beta = 5$ was chosen as the optimal β-value. For values of $\beta > 6$ the number of misclassifications was small (1 or 2 missclassifications) but > 0. The results on the test

and training data using the optimal $\beta = 5$ is presented in Table 5.

Table 5: 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. Gaussian Kernel SVM on New Images

As the results in the previous section show, the Gaussian kernel SVM perform very well on the train and test data when using a decent choice of parameters. If we applied the Gaussian kernel SVM on new unseen images I would expect it to work quite well, but we could not expect the same results. The value of the β-parameter in the previous section was chosen to minimize the number of misclassifications for the specific test data set. That indicates an overfit of the model to the test data, hence the model could not be expected to work as good for new images as it did for the test data images.

References

[1] FMAN45 Machine Learning *Assignment Instructions: Assignment 2* Spring 2021. LTH. Downloaded: 2021-04-20. https://canvas.education.lu.se/courses/11000/files/1221084?wrap=1