183.605
Machine Learning for Visual Computing
Assignment 1

Group 12:
Hanna Huber (0925230)
Lena Trautmann (1526567)
Elisabeth Wetzer ()

November 16, 2016

- Upload a zip-file with the required programs. You can choose the programming language.
- Add a PDF document with answers to all of the questions of the assignment (particularly all required plots) and description and discussion of results.

1 Assignment 1

1.1 Part 1: Binary classification and the perceptron

1.1.1 Reading data

Tasks:

- Read the data using functions of your programming language resp. simulation software.

First, we wrote a new .csv-file with our .py-script to delete all extra blanks. Afterwards we could read in the data using `dlmread()`.

- Plot the input vectors in $\mathbb{R}^2$ and visualize corresponding target values (e.g. by using color).

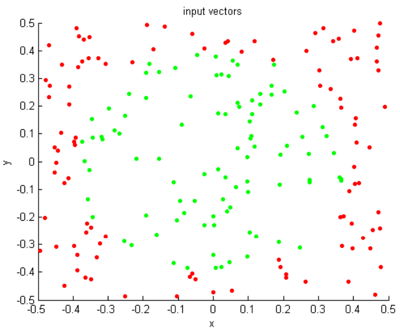

Figure 1: Plot of the input vectors with the target value visualized by colour.

Fig. 1 shows the input vectors.

- Use the feature transformation $(x_1, x_2) \to (x_1^2, x_2^2)$ and plot the data in the new feature space. The data should now be linearly separable.

Fig. 2 shows the transformed input vectors.

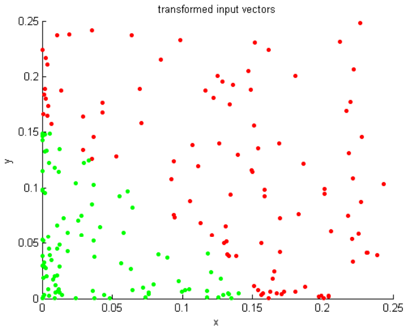

Figure 2: Plot of the transformed input vectors with the target value visualized by colour.

1.1.2 Perceptron training algorithm

The function

$$\texttt{y = perc(w,X).}$$

simulates a perceptron. The first argument is the weight vector $\mathbf{w}$ and the second argument is a matrix with input vectors in its columns $\mathbf{X}$. The output $\mathbf{y}$ is a binary vector with class labels 1 or -1.

The function

$$\texttt{w = percTrain(X,t,maxIts,online).}$$

returns a weight vector $\mathbf{w}$ corresponding to the decision boundary separating the input vectors in $\mathbf{X}$ according to their target values $\mathbf{t}$.

The argument `maxIts` determines an upper limit for iterations of the gradient based optimization procedure. If this upper limit is reached before a solution vector is found, the function returns the current $\mathbf{w}$, otherwise it returns the solution weight vector. `online` is *true* if the *online*-version of the optimization procedure is to be used or *false* for the *batch*-version.

Tasks:

- Implement both functions. Use homogeneous coordinates and the corresponding augmented weight vector $\mathbf{w} \in \mathbb{R}^3$.
- Plot the data and decision boundary in $\mathbb{R}^2$, both in the original data space (see e.g. Figure 3) and in the feature space of basis functions, each at three different stages of the training: after the first iteration, after approximately half of the required iterations, and after convergence.
- Initialize $\mathbf{w} = \mathbf{0}$. What is the influence of the learning rate?

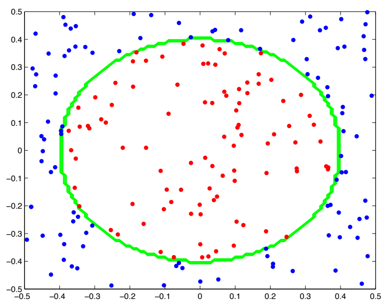

Figure 3: Plot of the decision boundary in the original data space found by the perceptron (green curve) together with labelled data points.

Table 1: Weight vector derived for the LMS-rule and of closed form

	ϕ_0	ϕ_1	ϕ_2
wLMS	-0.1288	-16.6005	3.7258
wClosed			

1.2 Part 2: Linear basis function models for regression

1.2.1 Experimental setup

At the end of the setup we have:

- xtrain and ttrain: the training data
- phi: the transformation function
- xtrain_phi: the transformation of xtrain

1.2.2 Optimization: LMS-learning rule vs. closed form

Tasks:

- What is the resulting weight vector when using the LMS-rule?

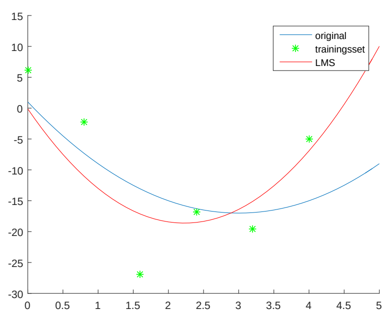

Figure 4: Optimization using the LMS-rule

The values of the resulting weight vector are summarized in Tab. 1 and the corresponding curve plotted in Fig. 4.

- How can you determine the optimal $\mathbf{w}^*$ in closed form? Compare $\mathbf{w}^*$ with the outcome of the LMS-rule training.

The closed form is calculated by updating the weight vector, determined by all N points, in one step.

- What is the influence of γ? Which value for γ represents a good tradeoff between number of iterations and convergence?

1.2.3 Model-complexity and model-selection

Determine $\mathbf{w}^*$ in closed form for 2000 different training sets, in which only the t_i are varyied according to $\mathcal{N}(\mu = y_i, \sigma = 16)$, while the x_i remain unchanged.

Tasks[1]:

- Select a fixed x', which is not an observation of the training set, but lies between two observations (e.g. $x' = 2$)
- Estimate the *mean squared error*

$$\mathit{mse} = \mathcal{E}(f(x') - f_{\mathbf{w}^*}(x'))^2, \tag{1}$$

i.e., the mean of the squared residuals of the models prediction $f_{\mathbf{w}^*}(x')$ from the true function value $f(x')$ for all $0 \leq d \leq 8$ ($d = 0$ corresponds to a constant function) using at least 2000 trials.

[1] In all tasks $\mathcal{E}$ refers to the expected value with respect to the random variable $\mathbf{w}^*$, i.e. $\mathcal{E} \equiv \mathcal{E}_{\mathbf{w}^*}$

- Estimate by the same way the quantities *bias*$^2 = (f(x') - \mathcal{E} f_{\mathbf{w}^*}(x'))^2$ and *var*$= (f_{\mathbf{w}^*}(x') - \mathcal{E} f_{\mathbf{w}^*}(x'))^2$.
- Plot *mse*, *bias*^{2} and *var* against d together in one plot. What is the relation of the quantities?
- (optional) Generate the above plots only for $d = 8$, but minimize instead of $E(\mathbf{w})$ the regularized error function

$$E_\lambda(\mathbf{w}) = \sum_{i=1}^{N} (t_i - \mathbf{w}^T \mathbf{\Phi}(x_i))^2 + \lambda \|\mathbf{w}\|^2, \tag{2}$$

i.e. $\mathbf{w}^* = \arg\min^*_{\mathbf{w}} E_\lambda(\mathbf{w})$. Plot the quantities against λ instead of d. Hint: The minimum can be obtained in closed form (see lecture slides).