CSE 446: Machine Learning
Winter 2018

Assignment 2

from
Lukas Nies
University of Washington

02/01/18

Contents

0 Policies

0.1 List of Collaborators

My collaborators were Edith Heiter (discussed Problem 2 and 4) and Julia Hess (discussed ?). The development of the answers though was completely independent and individually.

0.2 List of Acknowledgments

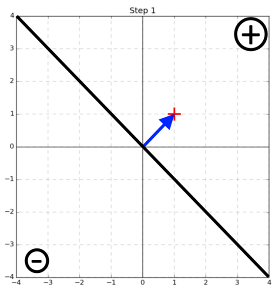
None.

0.3 Policies

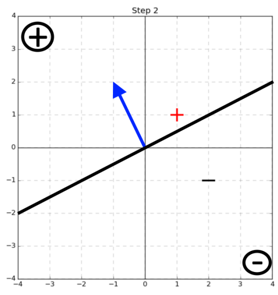
I have read and understood these policies.

1 Perceptron Exercise

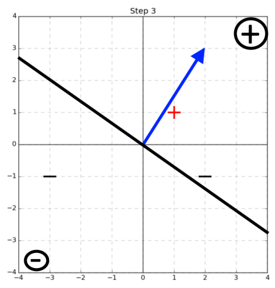
The evolution of the weight vector w are shown in figure 1 after the data was observed so the weights were already updated. In the same figure the maximum margin is shown since the perceptron didn't maximize the margin after the training set.



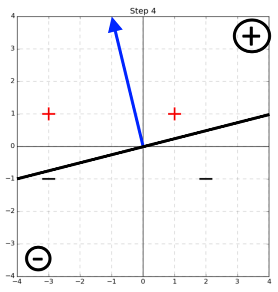
(a) Step One



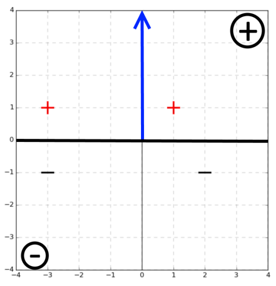
(b) Step Two



(c) Step Three



(d) Step Four



(e) Maximum margin

Figure 1: Visualization of the perceptron's decision boundary. The plots show the weight vector w in blue and the corresponding decision boundary (black) after they were updated. Plot (e) shows the maximum decision margin which is has the value 1 in this case.

To solve Problem 1.3 I will follow the proof according to [1].

Theorem 1.1. *If data D is linear separable with the geometric margin $\gamma^* = 0.5$ and has the maximum norm $\|x\|^2 \leq 5 \forall x \in D$ then the algorithm will convert after at most $\frac{5}{\gamma^2}$ updates.*

Proof. Suppose x^* realizes $\gamma^* > 0$ and let $w^{(k)}$ be the k^{th} weight vector after k updates. We show in the following that the angle between w and w^* decreases such that the algorithm converges. That is when $w^* \cdot w$ increases faster than $\|w\|^2$.
After the k^{th} update we find

$$w^* \cdot w^{(k)} = w^* \left(w^{(k-1)} + yx\right) = w * \cdot w^{(k-1)} + yw^* \cdot x \geq w * \cdot w^{(k-1)} + k\gamma$$

and

$$\|w^{(k)}\|^2 = \|w^{(k-1)} + yx\|^2 = \|w^{(k-1)}\|^2 + 2yw^{(k-1)} \cdot x + y^2 \cdot \|x\|^2 \geq \|w^{(k-1)}\|^2 + 5 + 0$$

The last line shows that $\|w^{(k)}\|$ increases by at least 5 every update. Therefore $\|w^{(k)}\|^2 \leq 5k$. So

$$\sqrt{5k} \geq \|w^{(k)}\| \geq w^* \cdot w^{(k)} \geq 5\gamma \Leftrightarrow k \leq \frac{5}{\gamma^2}$$

∎

The maximum number of mistakes (updates) the perceptron has to make until it converges is given by $\frac{5}{\gamma^2} = \frac{5}{0.25} = 20$.

2 Implementation: Perceptron

3 Beat the Perceptron

4 PCA on Digits

4.1 Convariance matrix construction

Let $X \in \mathbb{R}^{n \times d}$ be the data matrix, $\vec{x}_i \in \mathbb{R}^{d \times 1}$ be the i^{th} element of X, $\vec{\mu} \in \mathbb{R}^{d \times 1}$ be the mean vector of all $\vec{x}_i$, $\vec{1}$ be a vector in $\mathbb{R}^{n \times 1}$ consisting of 1's, and $X_c = X - \vec{1}\vec{\mu}^T$ be the centered data matrix. Note that all vectors are considered as column vectors.

In figure 2 ten digits from the MNIST database are plotted, in figure 3 the mean image ($\vec{\mu}$) is plotted.

There are two methods to write down the covariance matrix:

- Vector based method:

$$\Sigma = \frac{1}{n} \sum_{i=1}^{n} (\vec{x}_i - \vec{\mu}) \cdot (\vec{x}_i - \vec{\mu})^T \tag{1}$$

Table 1: Add caption

dataset	question number 2	3	4	5
2	Is linear separable because error rate doesn't change after $\approx$ 380 epochs	Does not overfit because test-set error doesn't rise	No features	/
3	Is linear separable because error rate doesn't change after $\approx$ 2000 epochs	Does not overfit because test-set error doesn't rise	No noise	
4				
5				
6				
7				
8				
9				

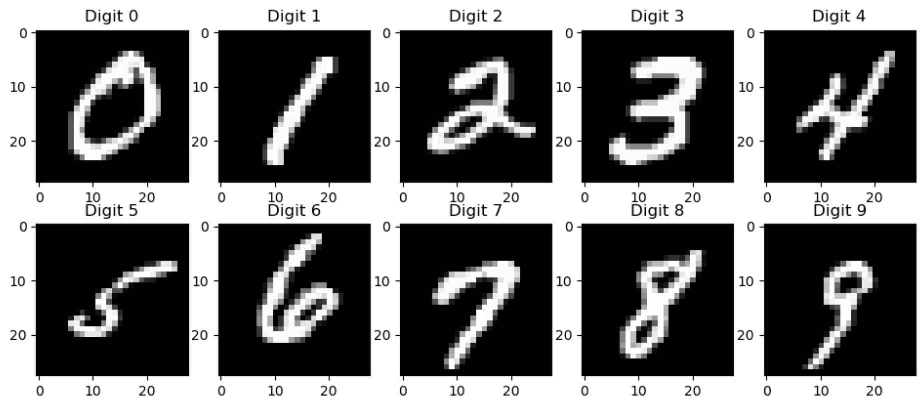


Figure 2: Grey scale plot of ten digits of the MNIST database

- Matrix based method:

$$\Sigma = \frac{1}{n}(X_c^T \cdot X_c) \tag{2}$$

By considering the dimensions of the centered data matrix $X_c \in \mathbb{R}^{n \times d}$ we find for sigma $\Sigma \in \mathbb{R}^{d \times d}$.

By coding both methods (code attached to .tar.gz, as usual) we found for the vector method a runtime of roughly 191s and $<$ 1s for the matrix based method.

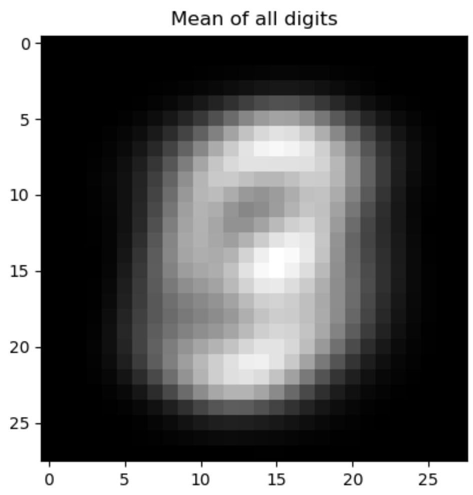


Figure 3: Grey scale plot of the mean image of the MNIST database

This is in increase of over 19100% in runtime. By testing the average absolute difference between the two methods we found a very small value (9.78×10^{-16}), in the range of machine precision so the two results are equal. NUMBER 4.1.6 IS MISSING!

4.2 Eigen-Analysis

By using the SVD on the covariance matrix we find following eigenvalues:

- $\lambda_1 = 5.10819064381$
- $\lambda_2 = 3.70090586283$
- $\lambda_{10} = 1.24026412182$
- $\lambda_{30} = 0.362081056419$
- $\lambda_{50} = 0.168298737356$
- $\sum_{i=1}^{d} \lambda_i = 52.4218857752$

The fractional reconstruction error for the first $k = 100$ eigenvalues is plotted in figure 4.

The first 11 eigenvalues make up 50% of the total variance and the first 44 eigenvalues make up 80% of the eigenvalues.

The first ten eigenvectors are shown in figure 5. They should represent the dimensions with the highest variance (biggest information) and therefore should carry essential information about the ten possible digits, zero to nine.

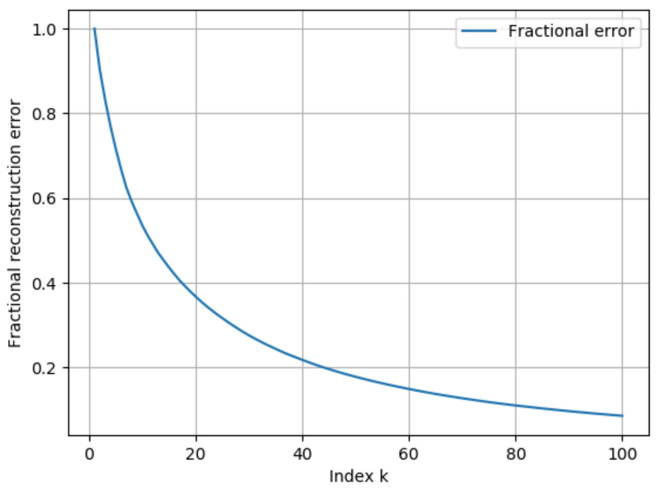


Figure 4: Fractional Reconstruction error for the first $k = 100$ eigenvalues

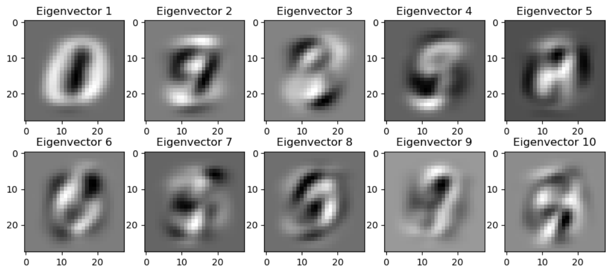


Figure 5: Plot of the first ten eigenvectors

4.3 Pseudocode for Reconstruction

Goal: best reconstruction (squared error should be small) for dimensionality reduction from d to k dimensions.

1. Use the top k eigenvectors since they carry the larges variance (information) for the reconstruction. The actual number k is a hyperparameter which has to be optimized.

2. By using SVD on the covariance matrix Σ we get the k eigenvalues. The top k eigenvalues make up the matrix $\hat{U} \in \mathbb{R}^{d\times k}$. The reduced data matrix is then given by

$$\hat{X} = \left(X - \vec{1} \cdot \vec{\mu}^T\right) \cdot \hat{U} \tag{3}$$

3. The d-dimensional reconstruction is then given by

$$X_{\text{rec}} = \hat{X} \cdot \hat{U}^T + \vec{1} \cdot \vec{\mu} \tag{4}$$

4.4 Visualization and Reconstruction

For $k = 30$ it takes roughly 2s to reconstruct the data. The plots are shown for different k in figures 6 and 7. For low values of k ($k = 1, 3, 5$) we can see that the reconstruction is composed of one, three and five eigenvalues. The reconstruction itself is not useful for determining the original digit. For medium k ($k = 10, 25, 50$) the pictures get more and more blurry until for higher k ($k = 200, 500+$) one finally can estimate the digit.

5 Bayes Optimal Classifier

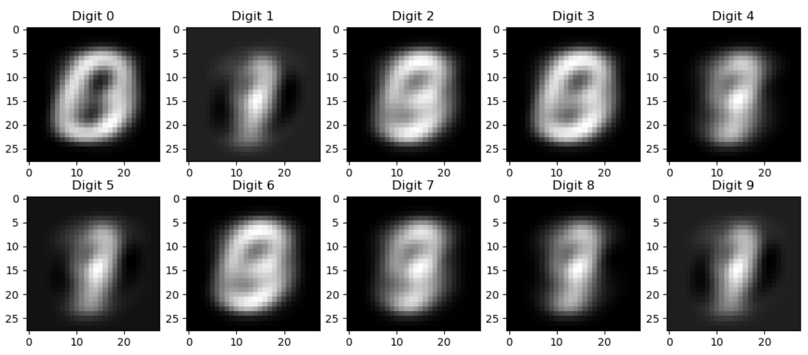
6 Bonus: Dimensionality Reduction

7 Bonus: The Bayes Optimal Hypothesis for Regression

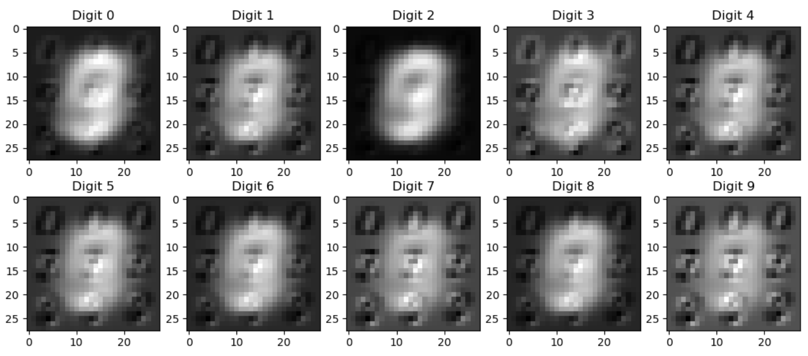
*Bibliography

References

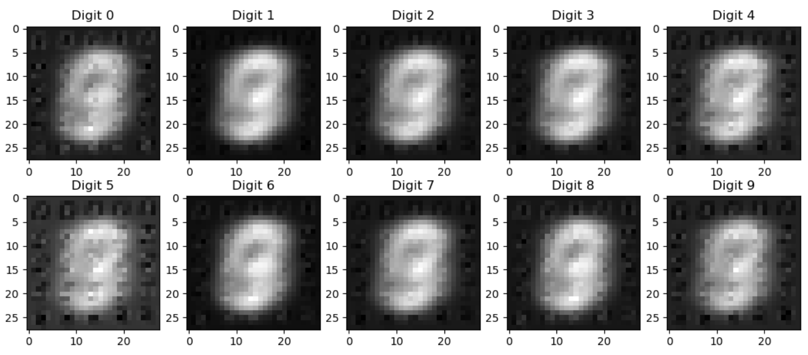
[1] Hal Daume III. *A Course in Machine Learning.* Secondary printing, January 2017.



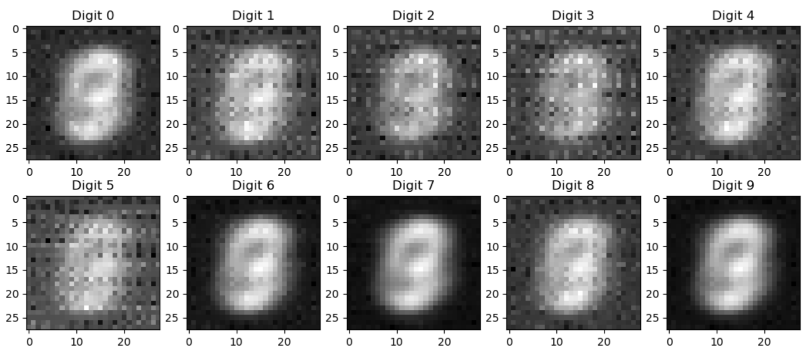
(a) Reconstruction for $k = 1$



(b) Reconstruction for $k = 3$

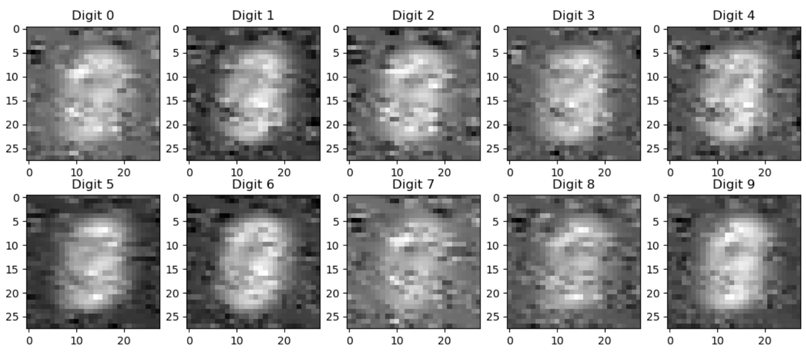


(c) Reconstruction for $k = 5$

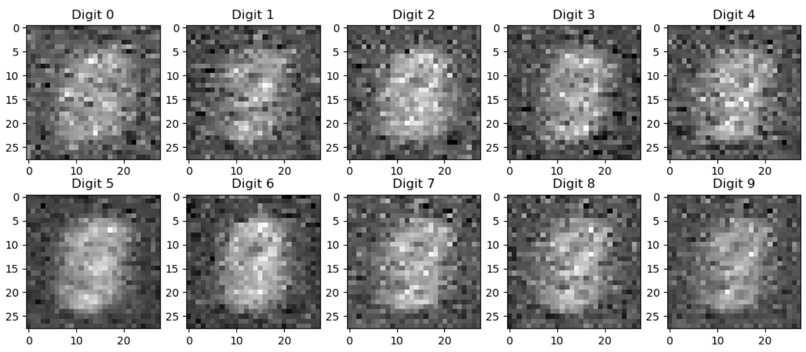


(d) Reconstruction for $k = 10$

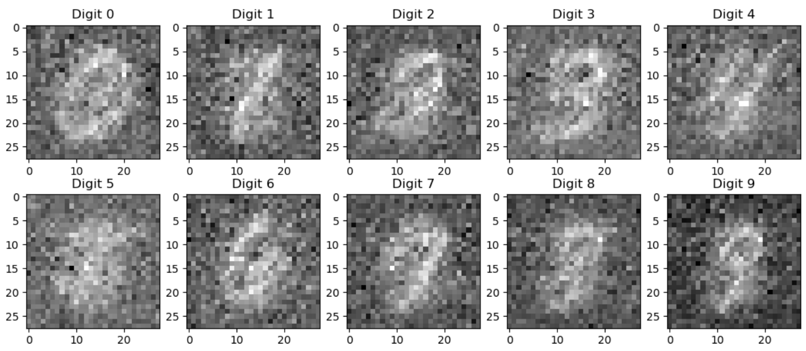
Figure 6: Reconstruction of the ten digits used in Problem part 4.1 for different k.



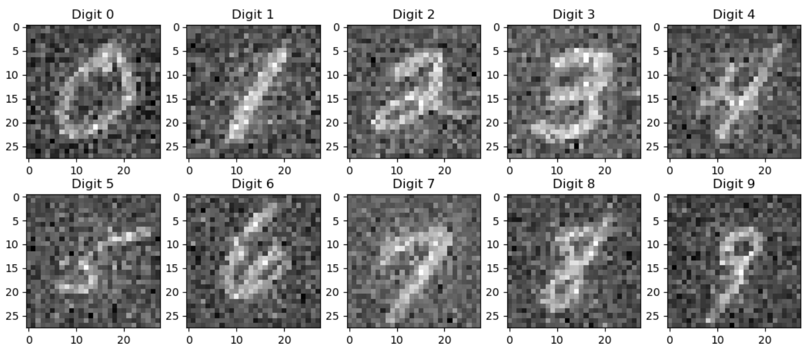
(a) Reconstruction for $k = 25$



(b) Reconstruction for $k = 50$



(c) Reconstruction for $k = 200$



(d) Reconstruction for $k = 500$

Figure 7: Reconstruction of the ten digits used in Problem part 4.1 for different k. Continuation from previous page.