Machine Learning Homework Sheet 10

Dimensionality Reduction & Matrix Factorization

1 PCA & SVD

Problem 1: Consider the latent space distribution

$$p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{0}, \boldsymbol{I})$$

and a conditional distribution for the observed variable $\boldsymbol{x} \in \mathbb{R}^d$,

$$p(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{W}\boldsymbol{z} + \boldsymbol{\mu}, \boldsymbol{\Phi})$$

where $\boldsymbol{\Phi}$ is an arbitrary symmetric, positive-definite noise covariance variable. Now suppose that we make a nonsingular linear transformation of the data variables $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ where $\boldsymbol{A}$ is a non-singular $d \times d$ matrix. If $\boldsymbol{\mu}_{ML}$, $\boldsymbol{W}_{ML}$, and $\boldsymbol{\Phi}_{ML}$ represent the maximum likelihood solution corresponding to the original untransformed data, show that $\boldsymbol{A}\boldsymbol{\mu}_{ML}$, $\boldsymbol{A}\boldsymbol{W}_{ML}$, and $\boldsymbol{A}\boldsymbol{\Phi}_{ML}\boldsymbol{A}^T$ will represent the corresponding maximum likelihood solution for the transformed data set. Finally, show that the form of the model is preserved if $\boldsymbol{A}$ is orthogonal and $\boldsymbol{\Phi}$ is proportional to the unit matrix so $\boldsymbol{\Phi} = \sigma^2\boldsymbol{I}$ (i.e. probabilistic PCA). The transformed $\boldsymbol{\Phi}$ matrix remains proportional to the unit matrix, and hence probabilistic PCA is covariant under a rotation of the axes of data space, as is the case for conventional PCA.

The model for $\boldsymbol{y}$ is a *noiseless* linear transformation. Given that the distribution of $\boldsymbol{x}$ is known, we therefore know the distribution of $\boldsymbol{y}$. Because of the definitions for $\boldsymbol{z}$ and $\boldsymbol{x}|\boldsymbol{z}$ we know that $\boldsymbol{x}$ is a Gaussian with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{W}\boldsymbol{W}^T + \boldsymbol{\Phi}$. And thus, $\boldsymbol{y}$ is also Gaussian with mean $\boldsymbol{A}\boldsymbol{\mu}$ and covariance $\boldsymbol{A}\boldsymbol{W}\boldsymbol{W}^T\boldsymbol{A}^T + \boldsymbol{A}\boldsymbol{\Phi}\boldsymbol{A}^T$. Now, assuming that the maximum likelihood solutions for the conditional model for $\boldsymbol{x}$ are $\boldsymbol{\mu}_x$, $\boldsymbol{W}_x$ and $\boldsymbol{\Phi}_x$, by simple *matching patterns* the MLE solutions for $\boldsymbol{y}$ are $\boldsymbol{A}\boldsymbol{\mu}_x$, $\boldsymbol{A}\boldsymbol{W}_x$ and $\boldsymbol{A}\boldsymbol{\Phi}_x\boldsymbol{A}^T$.

Now, if $\boldsymbol{A}$ is orthogonal and $\boldsymbol{\Phi}$ a scaled indentity matrix, the model characteristics are also preserved since $\boldsymbol{A}\boldsymbol{\Phi}_x\boldsymbol{A}^T = \sigma^2\boldsymbol{I}\boldsymbol{A}\boldsymbol{A}^T = \sigma^2\boldsymbol{I}^2 = \sigma^2\boldsymbol{I}$.

Problem 2: Use the SVD shown below. Suppose a new user Leslie assigns rating 3 to Alien and rating 4 to Titanic, giving us a representation of Leslie in the 'original space' of $[0, 3, 0, 0, 4]$. Find the representation of Leslie in concept space. What does that representation predict about how well Leslie would like the other movies appearing in our example data?

The projection is given by $\boldsymbol{P} = \boldsymbol{M} \cdot \boldsymbol{V}$, thus the representation of Leslie in concept space is given by $[0, 3, 0, 0, 4] \cdot \boldsymbol{V} = [1.74, 2.84]$. It seems that Leslie has a higher preference for "classic" movies (the score is 2.84) such as "Titanic" and "Casablanca" compared to the "sci-fi" movies (the score is 1.74). Thus, since she already saw "Titanic", "Casablanca" would be a reasonable recommendation.

	Matrix	Alien	Star Wars	Casablanca	Titanic
Joe	1	1	1	0	0
Jim	3	3	3	0	0
John	4	4	4	0	0
Jack	5	5	5	0	0
Jill	0	0	0	4	4
Jenny	0	0	0	5	5
Jane	0	0	0	2	2

Figure 11.6: Ratings of movies by users

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 0 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 0 & 0 & 2 & 2 \end{bmatrix}}_{M} = \underbrace{\begin{bmatrix} .14 & 0 \\ .42 & 0 \\ .56 & 0 \\ .70 & 0 \\ 0 & .60 \\ 0 & .75 \\ 0 & .30 \end{bmatrix}}_{U} \underbrace{\begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix}}_{\Sigma} \underbrace{\begin{bmatrix} .58 & .58 & .58 & 0 & 0 \\ 0 & 0 & 0 & .71 & .71 \end{bmatrix}}_{V^{\mathrm{T}}}$$

In general, if $\hat{\boldsymbol{U}}, \hat{\boldsymbol{\Sigma}}, \hat{\boldsymbol{V}}^T$ are the full singular values/vectors of $\boldsymbol{M}$ (obtained by performing full SVD on $\boldsymbol{M}$) and $\boldsymbol{U}, \boldsymbol{\Sigma}, \boldsymbol{V}^T$ are the respective truncated versions (i.e. by taking only the top K singular values/vectors) it holds that the projected data $\boldsymbol{P}$ can be obtain in two alternative and equivalent ways: $\boldsymbol{P} = \boldsymbol{U} \cdot \boldsymbol{\Sigma}$ or $\boldsymbol{P} = \boldsymbol{M} \cdot \boldsymbol{V}$. We usually prefer the second way since we only need to compute the top k singular vectors.

Problem 3: Load the notebook `homework_10_dim_reduction.ipynb` from Piazza. Fill in the missing code and run the notebook. Convert the evaluated notebook to pdf and add it to the printout of your homework.

Note: We suggest that you use Anaconda for installing Python and Jupyter, as well as for managing packages. We recommend that you use Python 3.

For more information on Jupyter notebooks consult the Jupyter documentation. Instructions for converting the Jupyter notebooks to PDF are provided within the notebook.

2 Matrix Factorization

Problem 4: Load the notebook `homework_10_matrix_factorization.ipynb` from Piazza. Fill in the missing code and run the notebook. Convert the evaluated notebook to pdf and add it to the printout of your homework.

Note: We suggest that you use Anaconda for installing Python and Jupyter, as well as for managing packages. We recommend that you use Python 3.

For more information on Jupyter notebooks consult the Jupyter documentation. Instructions for converting the Jupyter notebooks to PDF are provided within the notebook.

3 Autoencoders

Problem 5: We train a linear autoencoder to D-dimensional data. The autoencoder has a single K-dimensional hidden layer, there are no biases, and all activation functions are identity ($\sigma(x) = x$).

- Why is it usually impossible to get zero reconstruction error in this setting if $K < D$?
- Under which conditions is this possible?

We have $f(\boldsymbol{x}) = \boldsymbol{X}\boldsymbol{W}_1\boldsymbol{W}_2$ where $\boldsymbol{X}$ is the data matrix and the dimensions of the weight matrices are $D \times K$ for $\boldsymbol{W}_1$ and $K \times D$ for $\boldsymbol{W}_2$.

The final multiplication $\boldsymbol{W}_2$ brings points from K-dimensions up into D-dimensions but the points will still all be in a K-dimensional linear subspace. Unless the data happen to lie exactly in a K-dimensional linear subspace, they can't be exactly fitted.