University of Colorado **Boulder**

CSCI 5622: Machine Learning

Lecture 11

Overview

- Motivation for dimensionality reduction
- Feature selection
  - Wrappers
  - Filters
  - Embedded methods
- Feature transformation
  - Principal Component Analysis (PCA)
  - Autoencoders

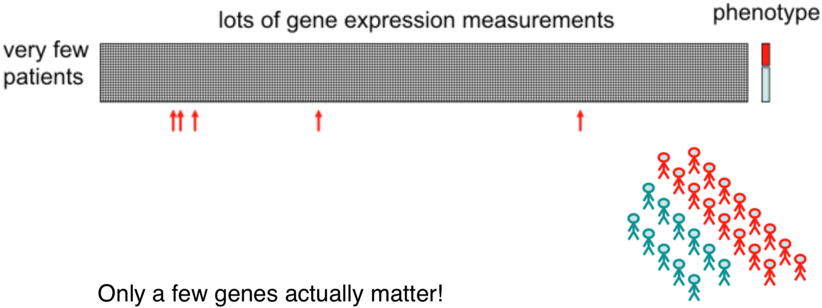
Overview

- Motivation for dimensionality reduction
- Feature selection
  - Wrappers
  - Filters
  - Embedded methods
- Feature transformation
  - Principal Component Analysis (PCA)
  - Autoencoders

Motivation for dimensionality reduction

Examples of large feature spaces

Predicting recurrence of lung cancer



Only a few genes actually matter!

Need small, interpretable subset to help doctors!

Motivation for dimensionality reduction

Examples of large feature spaces

Text classification


```
<REUTERS TOPICS=''YES'' LEWISSPLIT=''TRAIN''
CGISPLIT=''TRAINING-SET'' OLDID=''12981'' NEWID=''798''>
<DATE> 2-MAR-1987 16:51:43.42</DATE>
<TOPICS><D>livestock</D><D>hog</D></TOPICS>
<TITLE>AMERICAN PORK CONGRESS KICKS OFF TOMORROW</TITLE>
<DATELINE> CHICAGO, March 2 - </DATELINE><BODY>The American Por
Congress kicks off tomorrow, March 3, in Indianapolis with 160
of the nations pork producers from 44 member states determining
industry positions on a number of issues, according to the
National Pork Producers Council, NPPC.
Delegates to the three day Congress will be considering 26
resolutions concerning various issues, including the future
direction of farm policy and the tax law as it applies to the
agriculture sector. The delegates will also debate whether to
endorse concepts of a national PRV (pseudorabies virus) control
and eradication program, the NPPC said. A large
trade show, in conjunction with the congress, will feature
the latest in technology in all areas of the industry, the NPPC
added. Reuter
\&\#3;</BODY></TEXT></REUTERS>
```

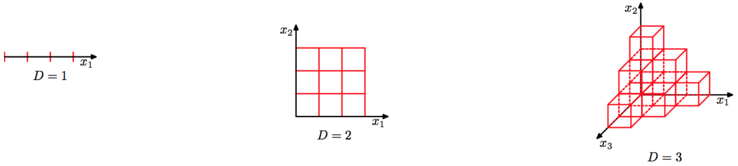

"Bag-of-Words" representation:

$x = \{0, 3, 0, 0, 1, ..., 2, 3, 0, 0, 0, 1\}$ one entry per word!

Easily 50,000 words! Very sparse - easy to overfit!

What is curse of dimensionality

Number of cells grows exponentially as dimensionality increases



# of cells r^D

r: number of divisions in each dimension

- Large number of cells, even if D is moderately large
- So to cover the whole space reasonably well, you need exponentially number of training data points

Dimensionality Reduction

Broader question

- How can we detect low dimensional structure in high dimensional data?

Motivations

- Exploratory data analysis & visualization: you can plot data now
- Compact representation: small memory/computational footprint, lossy data compression
- Robust statistical modeling: curse of dimensionality

Dimensionality Reduction

General rules of dimensionality reduction

- Relevant features: the features that we need to perform well
- Irrelevant features: the features that are unnecessary
- Redundant features: the features that that become irrelevant in the presence of others

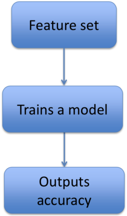
Overview

- Motivation for dimensionality reduction
- Feature selection
    - Wrappers
    - Filters
    - Embedded methods
- Feature transformation
    - Principal Component Analysis (PCA)
    - Autoencoders

Feature selection

Wrappers

- Rely on a feature search strategy to find an "optimal" subset of features based on the performance of the classifier
- Pros
  - High accuracy
  - Specific to the classifier of interest
- Cons
  - Computationally expensive



Feature selection

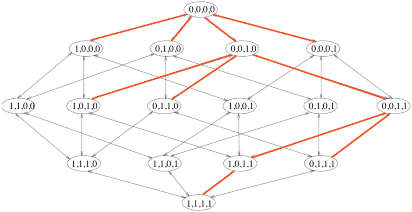
Wrappers - Possible feature search strategies

- Exhaustive search
  - Try all possible feature combinations
  - M features $\rightarrow 2^M$ possible subsets
- Sequential forward selection
  - Greedy incremental selection of best performing features
- Recursive backward elimination
  - Starting from the full feature set, greedy selection of features which hurt performance
- Genetic algorithms
  - Random selection of features
  - Update of feature selection probabilities based on performance metrics

Feature selection

Wrappers: Sequential Feature Selection

- Cost is $M + (M-1) + \ldots + 1 = \frac{M(M+1)}{2}$, instead of 2^M



Overview

- Motivation for dimensionality reduction
- Feature selection
  - Wrappers
  - Filters
  - Embedded methods
- Feature transformation
  - Principal Component Analysis (PCA)
  - Autoencoders

Feature selection

Filters

- We only pick the most informative features for the outcome
- We do not run a machine learning model
- We rank features according to their information and choose a cut-off point
- Pros
  - Computationally cheap
- Cons
  - No feature interaction is taken into account
  - Machine learning model is not taken into account

k	$J(X_k)$
35	0.846
42	0.811
10	0.810
654	0.611
22	0.443
59	0.388
...	...
212	0.09
39	0.05

Feature selection

Filters - Possible feature evaluation metrics

- Correlation of feature x_k with target variable y

$$\rho(x_k, y) = \frac{Cov(x_k, y)}{Var(x_k)Var(y)}$$

Measures linear dependencies

- Mutual information of feature x_k with target variable y

$$I(x_k, y) = \sum_i \sum_j P(x_k = i, y = j) \frac{P(x_k = i, y = j)}{P(x_k = i)P(y = j)}$$

where P is the probability estimate from the data
Assumes known probability distribution of the data.

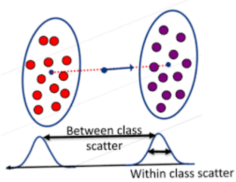
Feature selection

Filters - Possible feature evaluation metrics

- Fisher's criterion of feature d of sample $\mathbf{x_n}$ from class k

$$F(d) = \frac{\text{Within-class scatter}}{\text{Between-class scatter}} = \frac{\sum_k \sum_{\mathbf{x_n} \in C_k} (x_{nd} - \mu_{kd})^2}{\sum_k (\mu_d - \mu_{kd})^2}$$

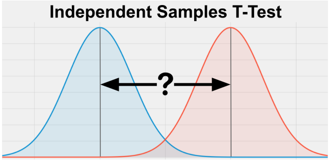
where μ_{kd} is the mean of feature d from class k, and μ_d is the mean of feature d from all samples
Measures within-class scatter in relation to between-class scatter

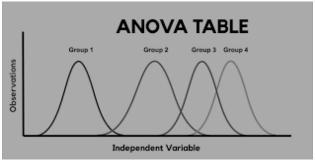


Feature selection

Filters - Possible feature evaluation metrics

- Statisitical tests (t-test, ANOVA)
  - T-test can be used for binary classification problems, ANOVA can be used for more than two classes
  - Assess whether the considered classes depict significantly different values of a feature

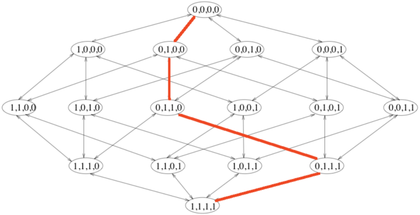




Feature selection

Filters

- A lot less expensive



Overview

- Motivation for dimensionality reduction
- Feature selection
    - Wrappers
    - Filters
    - Embedded methods
- Feature transformation
    - Principal Component Analysis (PCA)
    - Autoencoders

Feature selection

Embedded methods

- The classifier performs feature selection as part of the learning procedure
- Regularization is a great example

$$J(\mathbf{w}) = C(\mathbf{w}) + \lambda \|\mathbf{w}\|_1$$

  where C is the loss/cost function
- Pros
  - Feature selection is part of learning the procedure
- Cons
  - Computationally demanding

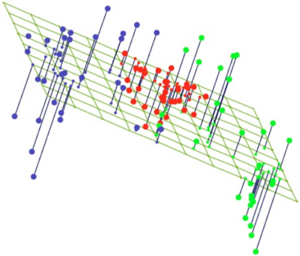
Overview

- Motivation for dimensionality reduction
- Feature selection
  - Wrappers
  - Filters
  - Embedded methods
- Feature transformation
  - Principal Component Analysis (PCA)
  - Autoencoders

Feature transformation

Linear feature transformation

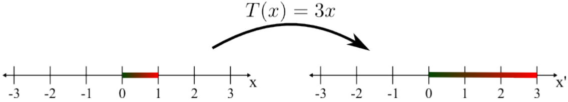
- $\mathbf{x} \in \mathbb{R}^D \rightarrow \mathbf{y} \in \mathbb{R}^M$, $D \gg M$
- linear transformation of original space: $\mathbf{y} = \mathbf{U}^T \mathbf{x}$, $\mathbf{U} \in \mathbb{R}^{D \times M}$



Feature transformation

Linear feature transformation

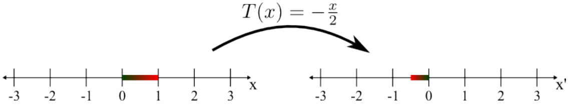
$T(x) = 3x$ maps the interval $[0, 1]$ to the interval $[0, 3]$



Feature transformation

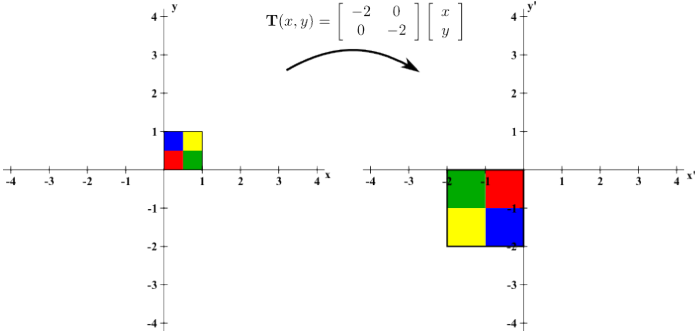
Linear feature transformation

$T(x) = -\frac{1}{2}x$ maps the interval $[0, 1]$ to the interval $[-\frac{1}{2}, 0]$



Linear feature transformation

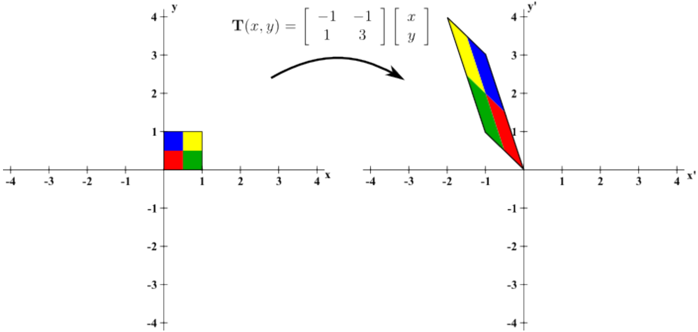
$T(x, y) = \begin{bmatrix} -2 & 0 \\ 0 & -2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$ maps the unit square $[0, 1] \times [0, 1]$ to the square $[-2, 0] \times [-2, 0]$ rotating the square by 180 degrees around the origin and stretching each side by a factor of 2.



(The quarters of the square are shown in different colors to help visualize how points within the square were mapped)

Linear feature transformation

$T(x,y) = \begin{bmatrix} -1 & -1 \\ 1 & 3 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$ maps the unit square $[0,1] \times [0,1]$ to the parallelogram shown below.



(The quarters of the square are shown in different colors to help visualize how points within the square were mapped)

See a demo here: `https://mathinsight.org/applet/linear_transformation_2d`

Feature transformation

Linear feature transformation: Example

Assuming an input vector $\mathbf{x} = [x_1, x_2] \in \mathbb{R}^2$ and a transformation $\mathbf{Ux}$, what transformation do each of the following matrices perform?

If $\mathbf{U} = \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{bmatrix}$, then $\mathbf{Ux} = \begin{bmatrix} u_{11}x_1 + u_{12}x_2 \\ u_{21}x_1 + u_{22}x_2 \end{bmatrix}$.

If $\mathbf{U} = \begin{bmatrix} u_{11} & u_{12} \end{bmatrix}$, then $\mathbf{Ux} = \begin{bmatrix} u_{11}x_1 + u_{12}x_2 \end{bmatrix}$.

Please choose the type of transformation for each matrix U.

1. $\mathbf{U} = [1, 0; 0, 1]$
2. $\mathbf{U} = [1.5, 0; 0, 1.5]$
3. $\mathbf{U} = [0, 1; 1, 0]$
4. $\mathbf{U} = [1, 0]$
5. $\mathbf{U} = [1, 1]$
6. $\mathbf{U} = [1, 1; 0, 1]$

Linear feature transformation: Example

Feature transformation

Feature transformation

Linear feature transformation: Example

Assuming an input vector $\mathbf{x} = [x_1, x_2] \in \mathbb{R}^2$ and a transformation $\mathbf{Ux}$, what transformation do each of the following matrices perform?

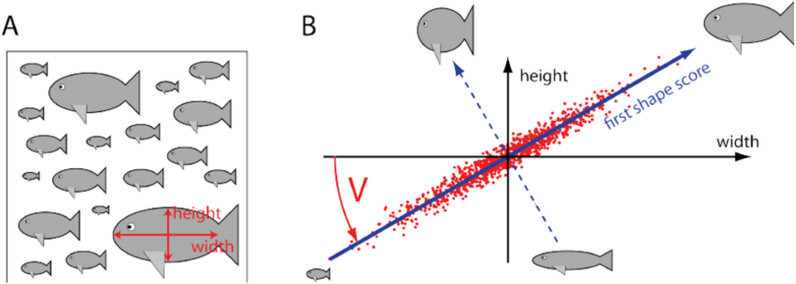
If $\mathbf{U} = \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \end{bmatrix}$, then $\mathbf{Ux} = \begin{bmatrix} u_{11}x_1 + u_{12}x_2 \\ u_{21}x_1 + u_{22}x_2 \end{bmatrix}$.

If $\mathbf{U} = \begin{bmatrix} u_{11} & u_{12} \end{bmatrix}$, then $\mathbf{Ux} = \begin{bmatrix} u_{11}x_1 + u_{12}x_2 \end{bmatrix}$.

1. $\mathbf{U} = [1, 0; 0, 1]$ Identity
2. $\mathbf{U} = [1.5, 0; 0, 1.5]$ Dilation
3. $\mathbf{U} = [0, 1; 1, 0]$ Flipping of axes
4. $\mathbf{U} = [1, 0]$ Preserving only first dimension
5. $\mathbf{U} = [1, 1]$ Substituting the first dimension by the sum of the two. Removing the second dimension.
6. $\mathbf{U} = [1, 1; 0, 1]$ Substituting the first dimension by the sum of the two. Preserving the second dimension.

Principal Component Analysis (PCA)

- A feature transformation method that finds new dimensions, or "principal components," that best capture the variance in the data
- Reduces the number of features while retaining the most important information

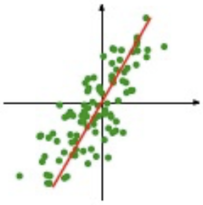


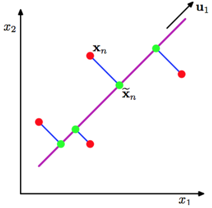
Principal Component Analysis (PCA): Representation

- Input: Data $\mathcal{D} = \{\mathbf{x_1}, \ldots, \mathbf{x_N}\}$, $\mathbf{x_n} \in \mathbb{R}^D$,centered inputs
- Output: Transformed/projected data $\{\mathbf{y_1}, \ldots, \mathbf{y_N}\}$, $\mathbf{y_n} \in \mathbb{R}^M$, $D \gg M$
- Transformation/Projection into subspace: $\mathbf{U} \in \mathbb{R}^{D \times M}$

$$\mathbf{y_n} = \mathbf{U}^T \mathbf{x_n} \ , \quad \mathbf{U}^T \mathbf{U} = \mathbf{I}$$

- Evaluation metric: many possible metrics yielding the same solution
  - Derivation 1: Maximize captured variance
  - Derivation 2: Minimize projection error

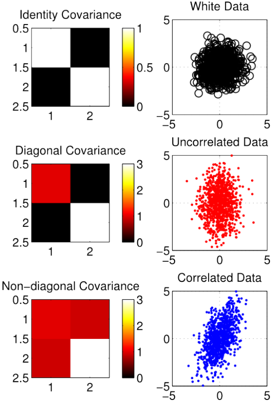




Covariance Matrix

For 2-dimensional samples $\mathbf{x_n} = [x_{n1}, x_{n2}]^T$, we assume means μ_1 and μ_2 for dimensions 1 and 2.

$$\boldsymbol{\Sigma} = \begin{bmatrix} \sum_{n=1}^{N} (x_{n1} - \mu_1)^2 & \sum_{n=1}^{N} (x_{n1} - \mu_1)(x_{n2} - \mu_2) \\ \sum_{n=1}^{N} (x_{n1} - \mu_1)(x_{n2} - \mu_2) & \sum_{n=1}^{N} (x_{n2} - \mu_2)^2 \end{bmatrix}$$

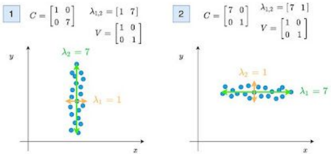


Covariance Matrix

- Assume D variables $f_1, f_2, \ldots, f_D$
- The covariance matrix S is a square and symmetric matrix used to describe the covariance values between pairwise combinations of the random variables
- The diagonal element i represent the variance of variable f_i: $var(f_i)$
- The non-diagonal element i, j represents the co-variance between variables f_i and f_j: $cov(f_i, f_j)$

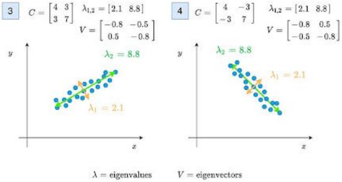
$$S = \begin{bmatrix} var(f_1) & cov(f_1, f_2) & \ldots & cov(f_1, f_D) \\ \vdots & & & \vdots \\ cov(f_d, f_1) & cov(f_d, f_2) & \ldots & var(f_D) \end{bmatrix} \in \mathbb{R}^{D \times D}$$

Eigenvalues and Eigenvectors of a Covariance Matrix



- When the covariance between variables is zero, eigenvalues will be directly equal to the variance values

Eigenvalues and Eigenvectors of a Covariance Matrix



- The eigenvectors (matrix **V**) show the direction of the covariance
- The eigenvalues($\lambda_{1,2}$) show the spread across each direction

Principal Component Analysis (PCA): Optimization

- Compute the $D \times D$ covariance matrix of samples $\mathbf{x_1}, \ldots, \mathbf{x_N}$ containing D features

$$\mathbf{S} = \frac{1}{N}\mathbf{X}^T\mathbf{X} \in \mathbb{R}^{D\times D}, \quad \mathbf{X} = \begin{bmatrix} - \tilde{\mathbf{x}}_\mathbf{1}^T - \\ \vdots \\ - \tilde{\mathbf{x}}_\mathbf{N}^T - \end{bmatrix} \in \mathbb{R}^{N\times D}$$

- Compute the eigenvalues $\lambda_1, \ldots, \lambda_D$ and eigenvectors $\mathbf{u_1}, \ldots, \mathbf{u_D}$ of $\mathbf{S}$
- Use the eigenvectors corresponding to the $M \leq D$ largest eigenvalues as the PCA transformation matrix

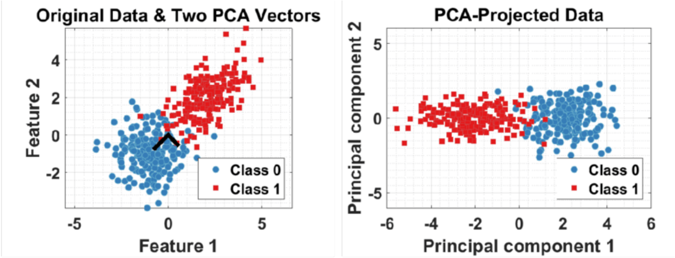
$$\mathbf{U} = \begin{bmatrix} | & & | \\ \mathbf{u_1} & \ldots & \mathbf{u_M} \\ | & & | \end{bmatrix} \in \mathbb{R}^{D\times M}$$

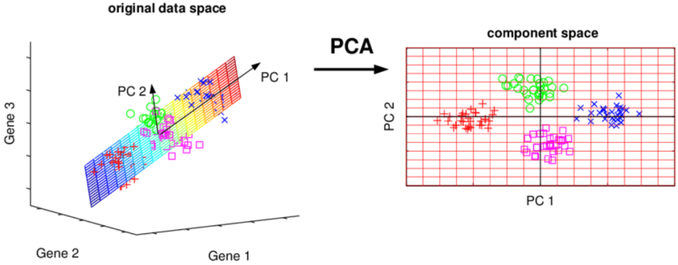
- Transform/project the original data: $\mathbf{y_n} = \mathbf{U}^T\mathbf{x_n} \in \mathbb{R}^{M\times 1}$, $n = 1, \ldots, N$

Principal Component Analysis (PCA): Algorithm

- Step 0: Mean normalize input features
- Step 1: Compute covariance matrix $\mathbf{S} = \frac{1}{N}\mathbf{X}^T\mathbf{X} = \frac{1}{N}\sum_n \mathbf{x_n}\mathbf{x_n}^T$
- Step 2: Diagonalize $\mathbf{S}$ and find eigenvector matrix $\mathbf{P}$ (where $\mathbf{SP} = \mathbf{P\Lambda}$, and $\mathbf{\Lambda}$ is a $D \times D$ diagonal matrix)
- Step 3: Take the first $M \leq D$ eigenvectors or *principal components* (corresponding to the M largest eigenvalues) and form reduced matrix $\mathbf{U}$
- Step 3: Project data into reduced space: $\tilde{\mathbf{x}}_\mathbf{n} = \mathbf{U}^T\mathbf{x_n}$

Principal Component Analysis (PCA): Example





Principal Component Analysis (PCA)

Original Images

Eigenvectors

they look like blurred original images

Mean | $\lambda_1 = 3.4 \cdot 10^5$ | $\lambda_2 = 2.8 \cdot 10^5$ | $\lambda_3 = 2.4 \cdot 10^5$ | $\lambda_4 = 1.6 \cdot 10^5$

Used to centralize inputs

Principal Component Analysis (PCA)

How to determine the number of principal components M?

Plot eigenspectrum

$$\frac{\sum_{d=1}^{M} \lambda_d}{\sum_{d=1}^{D} \lambda_d} \geq \text{threshold}, \text{ where common choices are } 95\%,\ 99\%$$

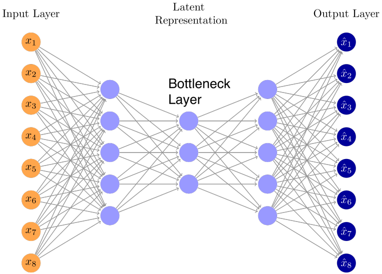
https://www.freecodecamp.org/news/an-overview-of-principal-component-analysis-6340e3bc4073/

Overview

- Motivation for dimensionality reduction
- Feature selection
  - Wrappers
  - Filters
  - Embedded methods
- Feature transformation
  - Principal Component Analysis (PCA)
  - Autoencoders

Autoencoders

- A subset of encoder-decoder architectures trained via unsupervised learning to reconstruct their own input data
- Encoder: layers that encode a compressed representation of the input, typically containing a progressively smaller number of nodes
- Bottleneck: the most compressed representation of the input
- Decoder: layers with a progressively larger number of nodes that decompress the encoded representation to its pre-encoded form
- Mean square error (MSE) loss: $L(\mathbf{W}, \mathbf{b}) = \sum_n (f_{\mathbf{W},\mathbf{b}}(\mathbf{x}) - \mathbf{x})^2$



What have a learned so far

- Dimensionality reduction for visualization, compression, avoid curse of dimensionality
- Feature selection to select the most informative features
- Feature transformation to transform the features into a reduced space
- **Readings:** Alpaydin 6.1-6.3