COMPUTER SCIENCE
INDIANA UNIVERSITY
School of Informatics and Computing
Bloomington

B555 Course Review

Reminders/comments

- Assignment 4 due this Wednesday
- Project due this Friday
- I do not have office hours today
- Please complete the online questionnaire: https://ocqbest.indiana.edu

Probability review

- Quantify uncertainty using probability theory
- Discussed sigma-algebras and probability measures
- Discussed random variables as functions of event-space
- Discussed relationships between random variables, including (in)dependence and conditional independence (belief network)
- Discussed operations, like expected value, marginalization, Bayes rule, chain rule

Exercise

- Suppose that we have created a machine learning algorithms that predicts whether a link will be clicked with 99% sensitivity (TPR) and 99% specificity (FPR). The rate the link is actually clicked is 1/1000 visits to a website. If we predict the link will be clicked on a specific visit, what is the probability it will actually be clicked?

- Let C be binary RV, with C = 1 indicating predict click

- p(C = 1 | y = 1) = TPR

- p(C = 1 | y = 0) = 1-FPR

ML and MAP

- MAP: formalize problem using the posterior, select model

$$M_{MAP} = \underset{M \in \mathcal{M}}{\arg\max} \{p(M|\mathcal{D})\}$$

- In discrete spaces: p(M | D) is the PMF

- In continuous spaces: p(M | D) is the PDF

- ML: formalize problem using the likelihood, select model

$$M_{ML} = \underset{M \in \mathcal{M}}{\arg\max} \{p(\mathcal{D}|M)\} .$$

Exercise questions

- For w in a constrained space, say w in [-10,10], what is the relationship between MAP and ML?

- What assumptions underly ML and MAP?

- We typically assume iid data. What happens if we no longer assume iid data?

Linear regression

- Assume p(y l x) is Gaussian distributed with fixed variance for noise term epsilon

$$\nabla E(\mathbf{w}) = \mathbf{0} \text{ we find that}$$

$$\mathbf{w}^* = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}.$$

Exercise

- Among the $k = 20$ features in a regression data set D, the 3rd feature can be expressed as a linear combination of the first two. You attempt to use OLS linear regression on such data.

- What will happen?

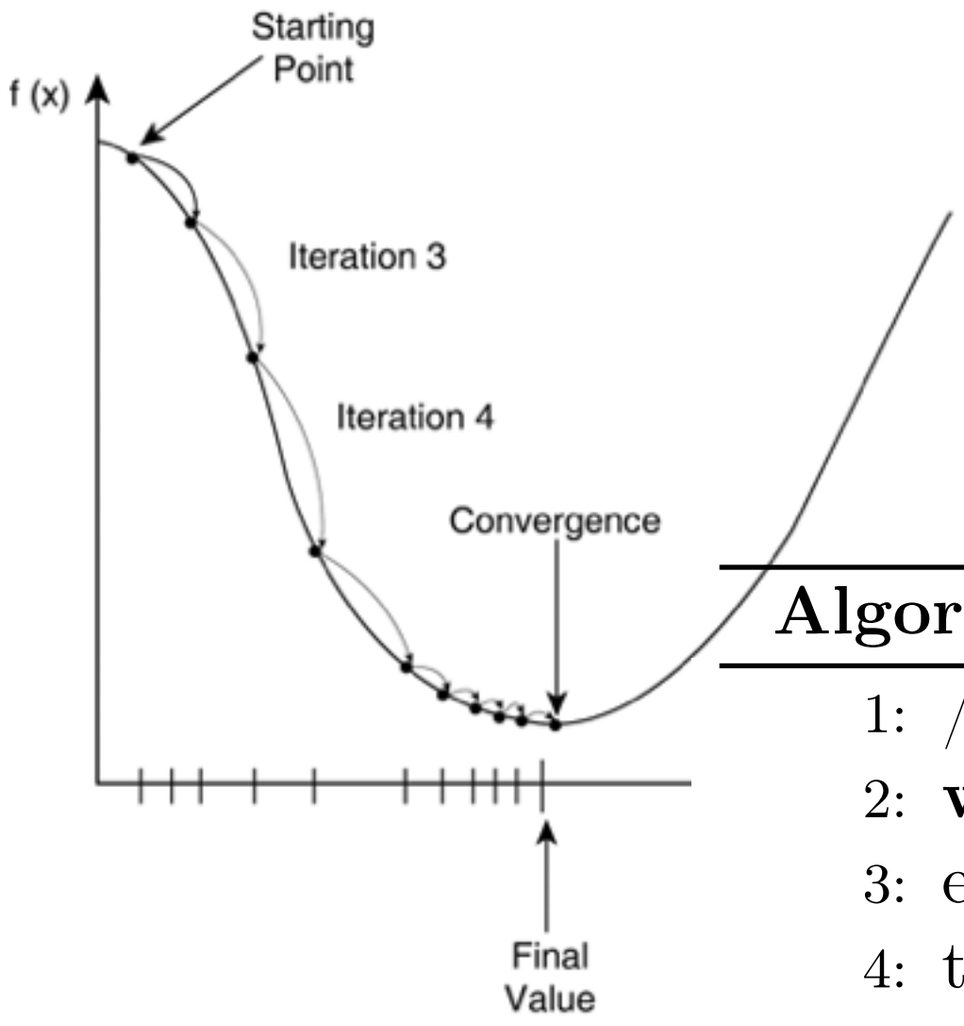
- What strategies can be used to remedy this issue?

- Is this likely to happen in practice?

Optimization

- Once problem is formalized, need to find the solution to that problem
- How can we find a solution?
- Is there a general strategy to always find a solution?
- What strategies could you use to find a solution?

Gradient descent



Algorithm 1: Batch Gradient Descent$(E, \mathbf{X}, \mathbf{y})$

// A non-optimized, basic implementation of batch g
$\mathbf{w} \leftarrow$ random vector in $\mathbb{R}^d$
err $\leftarrow \infty$
tolerance $\leftarrow 10e^{-4}$
$\alpha \leftarrow 0.1$
while $|E(\mathbf{w}) - \text{err}| > \text{tolerance}$ **do**
  // The step-size α should be chosen by line-search
  $\mathbf{w} \leftarrow \mathbf{w} - \alpha \nabla E(\mathbf{w}) = \mathbf{w} - \alpha \mathbf{X}^\top (\mathbf{X}\mathbf{w} - \mathbf{y})$
end while
return $\mathbf{w}$

First-order and second-order

- First-order gradient descent: first-order Taylor approximation to the function at that point

$$f(x + \alpha d) \approx f(x) + ((x + \alpha d) - x)f'(x) = f(x) + \alpha d f'(x)$$

- Second-order gradient descent: use a second-order Taylor approximation to the function at that point

$$f(x) \approx f(x_0) + (x - x_0)f'(x_0) + \frac{1}{2}(x - x_0)^2 f''(x_0).$$

$$f'(x) \approx f'(x_0) + (x - x_0)f''(x_0) = 0.$$

$$x = x_0 - \frac{f'(x_0)}{f''(x_0)}.$$

$$\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} - \left(H_{f(\mathbf{x}^{(i)})}\right)^{-1} \cdot \nabla f(\mathbf{x}^{(i)}),$$

Stochastic gradient descent

Algorithm 2: Stochastic Gradient Descent$(E, \mathbf{X}, \mathbf{y})$

1: $\mathbf{w} \leftarrow$ random vector in $\mathbb{R}^d$
2: **for** $t = 1, \ldots, n$ **do**
3: // For some settings, we need the step-size α_t to decrease with time
4: $\mathbf{w} \leftarrow \mathbf{w} - \alpha_t \nabla E_t(\mathbf{w}) = \mathbf{w} - \alpha_t(\mathbf{x}_t^\top \mathbf{w} - y_t)\mathbf{x}_t$
5: **end for**
6: **return** $\mathbf{w}$

For batch error: $\hat{E}(\mathbf{w}) = \sum_{t=1}^n E_t(\mathbf{w})$
e.g., $E_t(\mathbf{w}) = (\mathbf{x}_t^\top \mathbf{w} - y_t)^2$
$\hat{E}(\mathbf{w}) = \sum_{t=1}^n E_t(\mathbf{w}) = \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2$
$\nabla \hat{E}(\mathbf{w}) = \sum_{t=1}^n \nabla E_t(\mathbf{w})$
$E(\mathbf{w}) = \int_{\mathcal{X}} \int_{\mathcal{Y}} f(\mathbf{x}, y)(\mathbf{x}^\top \mathbf{w} - y)^2 dy d\mathbf{x}$

- Stochastic gradient descent (stochastic approximation) minimizes with an unbiased sample of the gradient $\mathbb{E}[\nabla E_t(\mathbf{w})] = \nabla E(\mathbf{w})$

- Question: can you have first & second-order stochastic gradient descent?

Constrained optimization

- Do not expect you to know how to do this for the final

- Idea: introduce Lagrange multipliers to bring up constraints into the objective function (like regularizers)

- Solve for Lagrange multipliers as well

Exercise

- How do we select the starting point?
- How does the procedure differ if we are doing maximization or minimization?
- How do we pick the step-size?
- When should we use batch and stochastic? (see “The tradeoffs of large scale learning”)
- What about mini-batch? (see “Efficient Mini-batch Training for Stochastic Optimization”)

Exercise

- Understand the expected properties of your estimator
- Example: give NN the same training data, in the same order, with the same starting point —> should it produce the same final set of weights?
- Example: with enough samples, should I get close to the true parameters? for any small random subset, how different might my learned parameters be?

Generalized linear models

- Generalize distribution p(y | x) to any exponential family model
- Result: learning parameters w such that

$$1.\ \ f(E[y|\mathbf{x}]) = \boldsymbol{\omega}^T\mathbf{x}$$

$$2.\ \ p(y|\mathbf{x}) \in \text{Exponential Family}$$

- Is linear regression with p(y | x) a generalized linear model?

Classification

- Logistic regression
- Multinomial logistic regression
- Support vector machine (SVMs)
- Naive Bayes

Exercise

- What model might you use if
  - we have binary features and targets?
  - binary targets and continuous features?
  - positive targets?
  - categorical features with a large number of categories?
  - multi-class targets, with continuous features?
- When might logistic regression do better than linear regression?
- When might Poisson regression do better than linear regression?

Logistic regression vs naive Bayes

- When to chose one or the other?
- Theoretical results indicating that naive Bayes converges more quickly, but that asymptotically logistic regression has a lower error. See “On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes”
- Bias and variance: which one might have higher/lower bias and higher/lower variance?

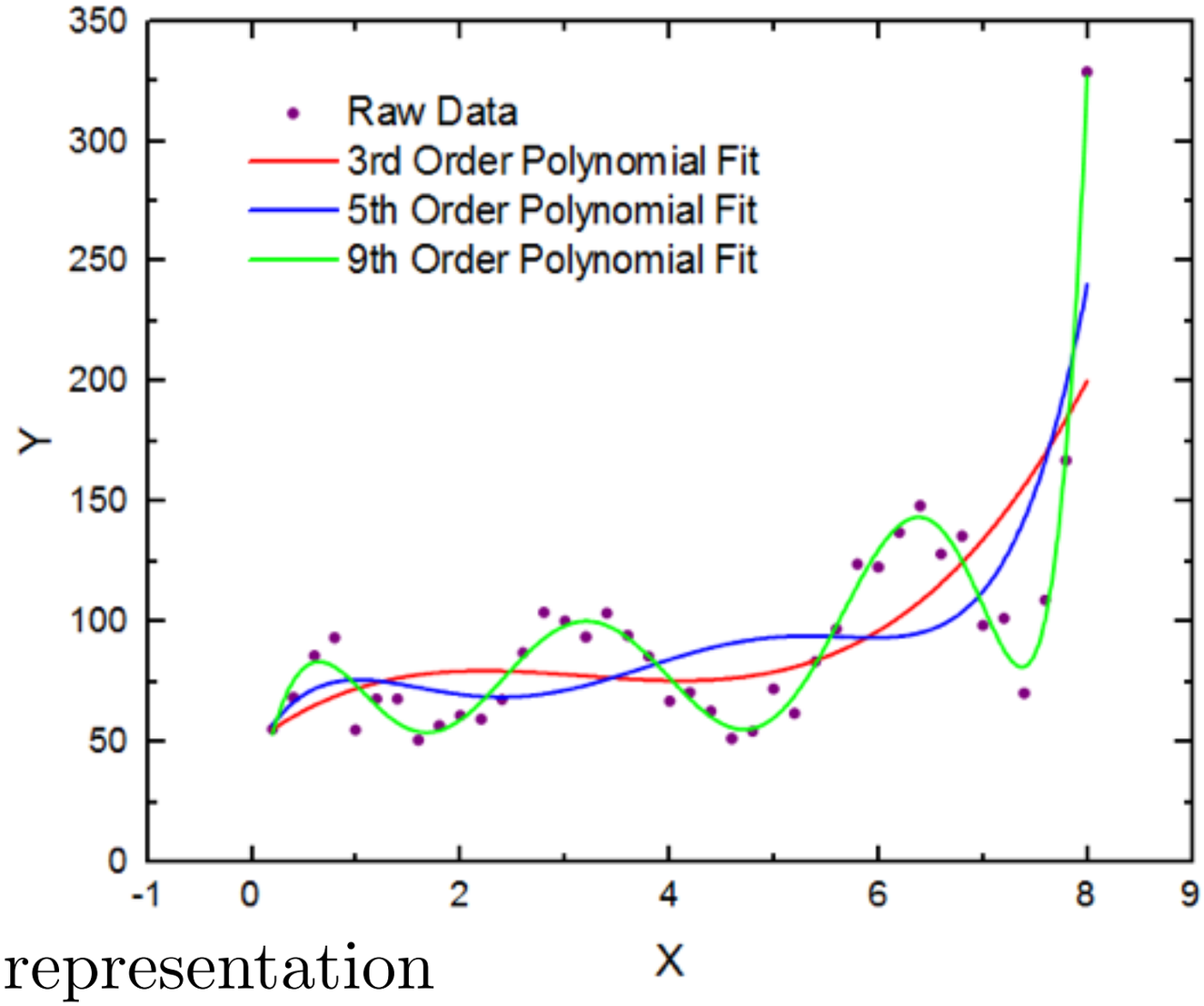
Representation learning

- Convert linear predictors into non-linear predictors, e.g.

$$\mathbf{x} \rightarrow$$

$$\text{2nd-order polynomial}(\mathbf{x}) =$$

$$w_6 x_1^2 + w_5 x_2^2 + w_4 x_1 x_2 + w_2 x_2 + w_1 x_1 + w_0$$



$$\mathbf{x}^\top \mathbf{w} = g(E[y|\mathbf{x}])$$

transformed to more powerful representation

$$\text{polynomial}(\mathbf{x})^\top \mathbf{w} = g(E[y|\mathbf{x}])$$

Radial basis function network

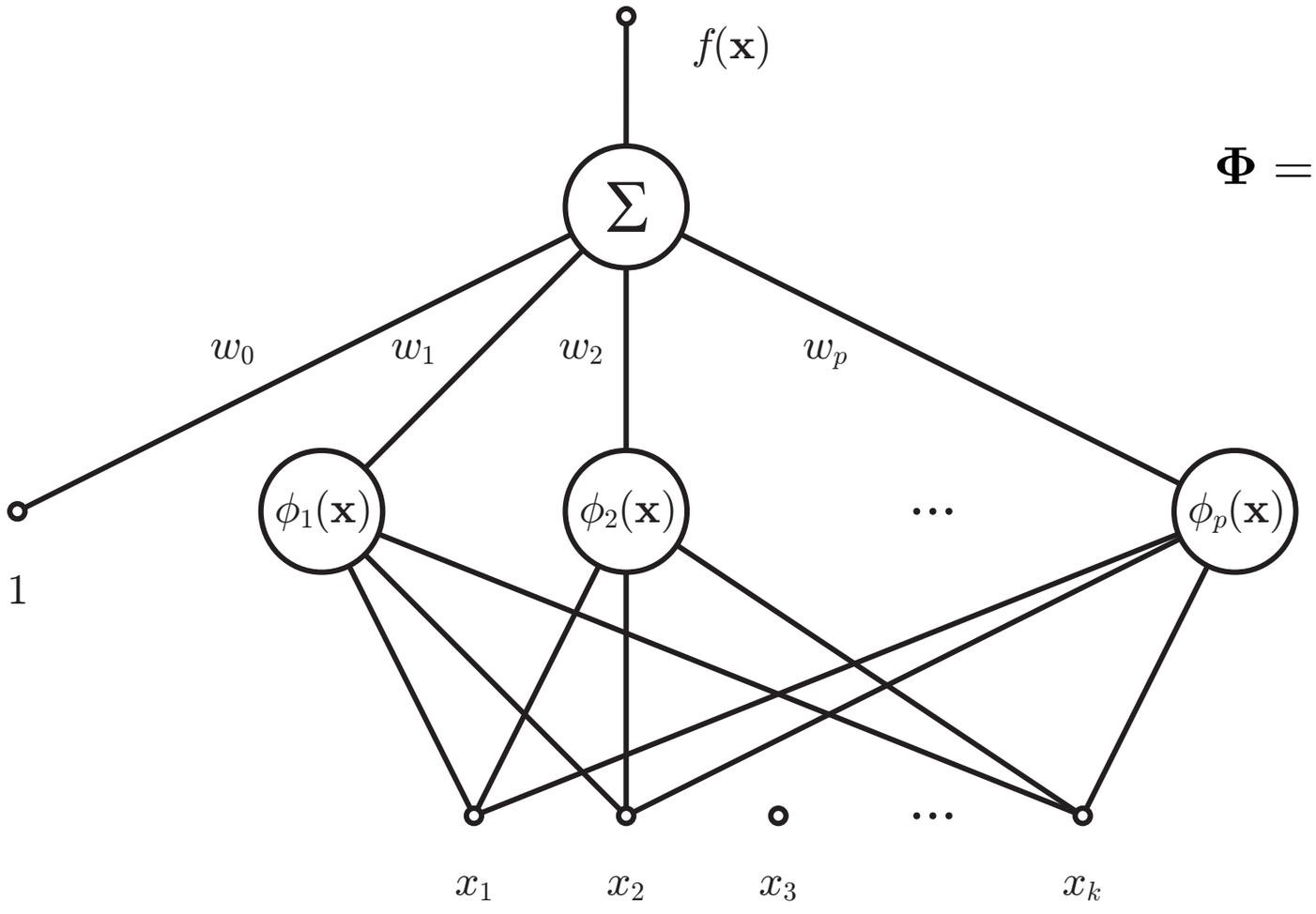


Figure 7.1: Radial basis function network.

$$\mathbf{\Phi} = \begin{bmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_p(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & & \\ \vdots & & \ddots & \\ & & & \\ \phi_0(\mathbf{x}_n) & & & \phi_p(\mathbf{x}_n) \end{bmatrix}$$

e.g., $\phi_j(\mathbf{x}) = e^{-\frac{\|\mathbf{x}-\mathbf{c}_j\|^2}{2\sigma_j^2}}$,

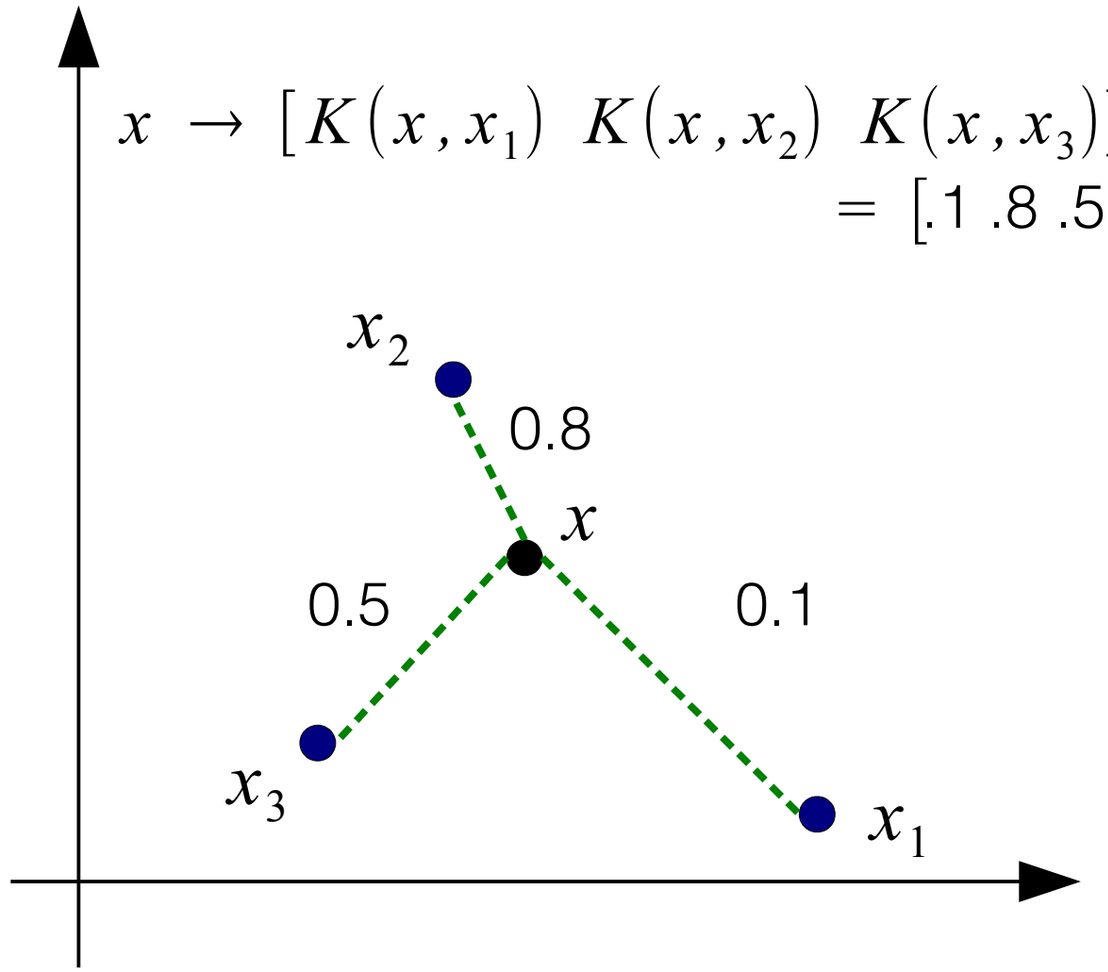
$$\begin{aligned} f(\mathbf{x}) &= w_0 + \sum_{j=1}^{p} w_j \phi_j(\mathbf{x}) \\ &= \sum_{j=0}^{p} w_j \phi_j(\mathbf{x}) \end{aligned}$$

RBF and Kernel representation

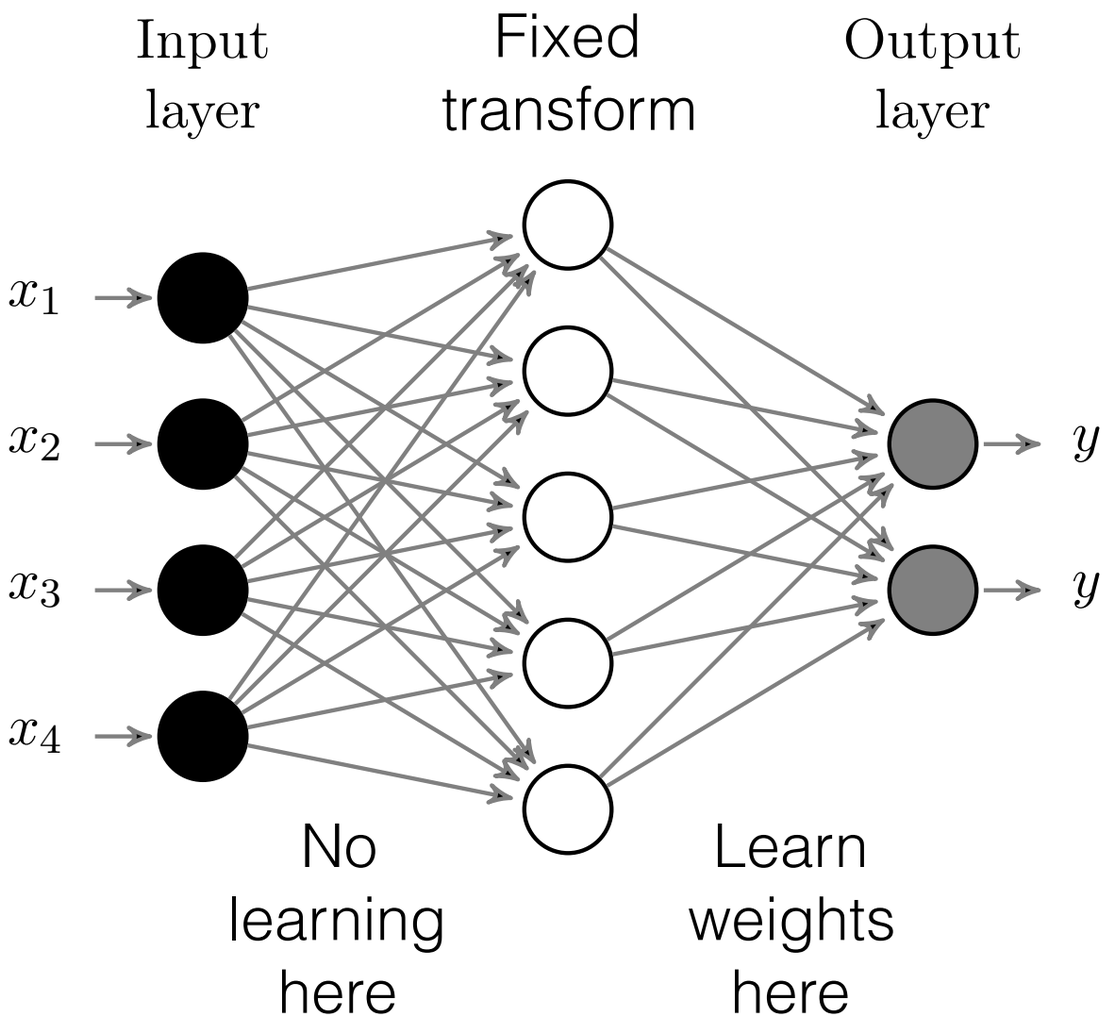
$$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-\|\mathbf{x} - \mathbf{x}'\|_2^2}{\sigma^2}\right)$$

$$f(\mathbf{x}) = \sum_{i=1}^{k} w_i k(\mathbf{x}, \mathbf{x}_i)$$

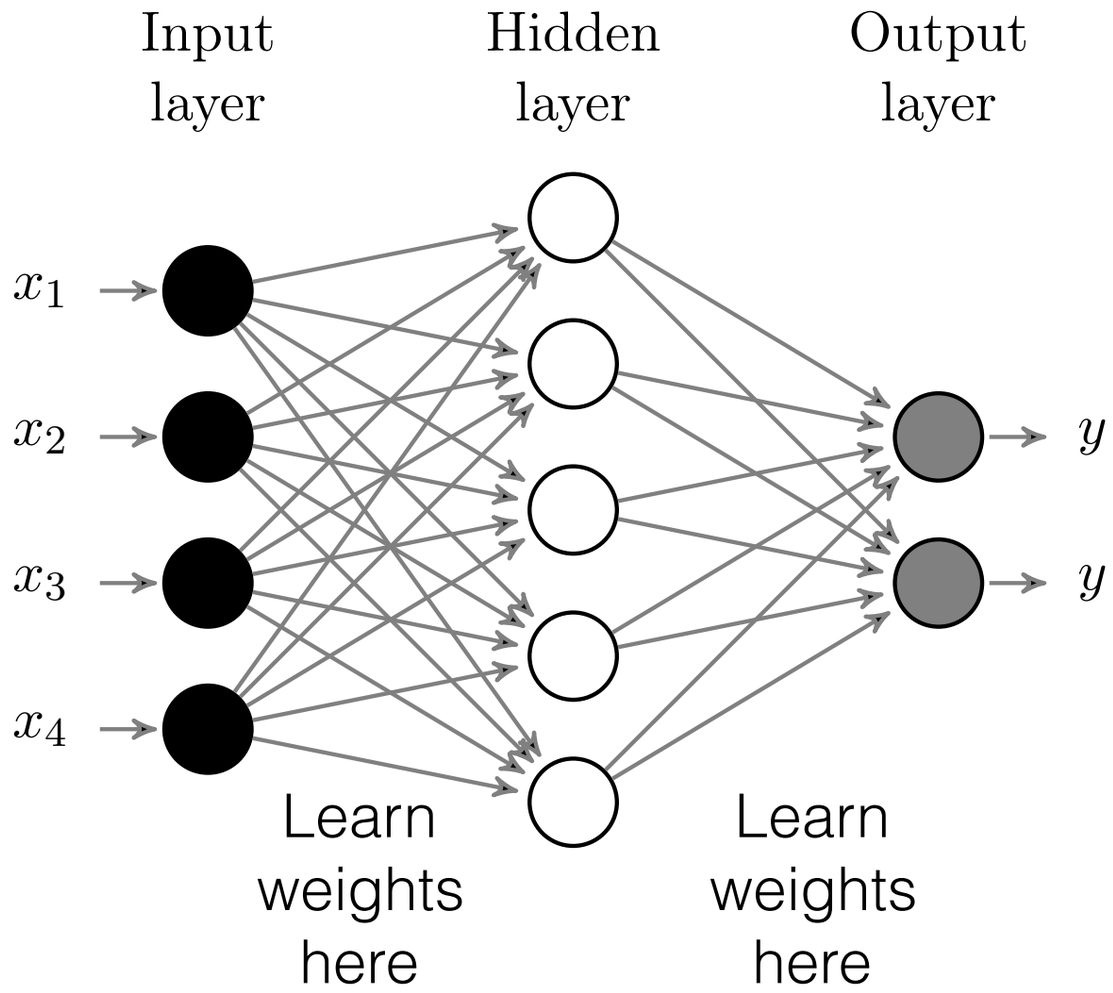
$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \rightarrow \begin{bmatrix} k(\mathbf{x}, \mathbf{x}_1) \\ \vdots \\ k(\mathbf{x}, \mathbf{x}_k) \end{bmatrix}$$



Neural networks



GLM with augmented fix representation



Two-layer neural network

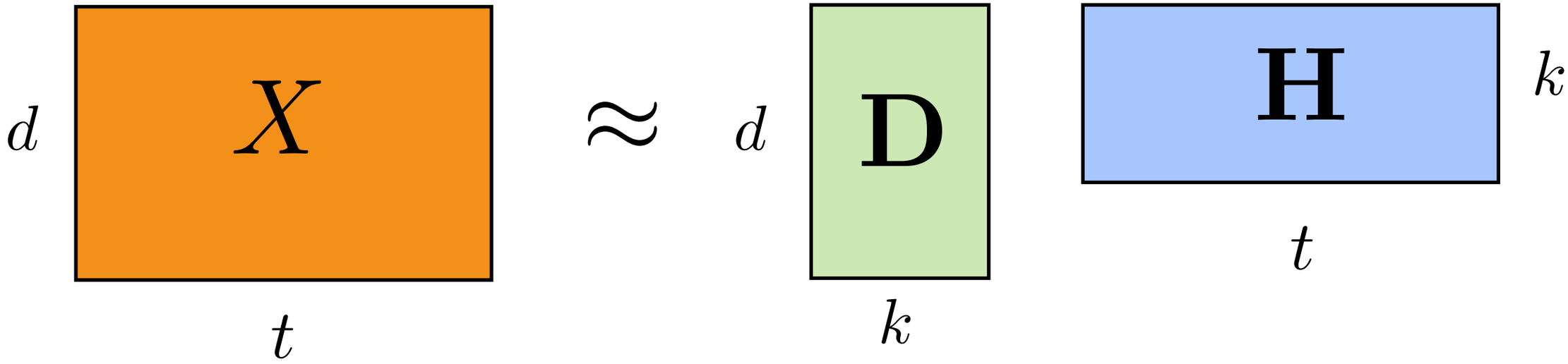
Matrix factorization

- Only expect you to know about unsupervised matrix factorization
- We have the objective:

$$\min_{D \in \mathbb{R}^{d \times k}, H \in \mathbb{R}^{k \times t}} \sum_{i=1}^{t} L(\mathbf{D}\mathbf{H}_{:i}, \mathbf{X}_{:i}) + \lambda R_D(\mathbf{D}) + \lambda R_H(\mathbf{H})$$

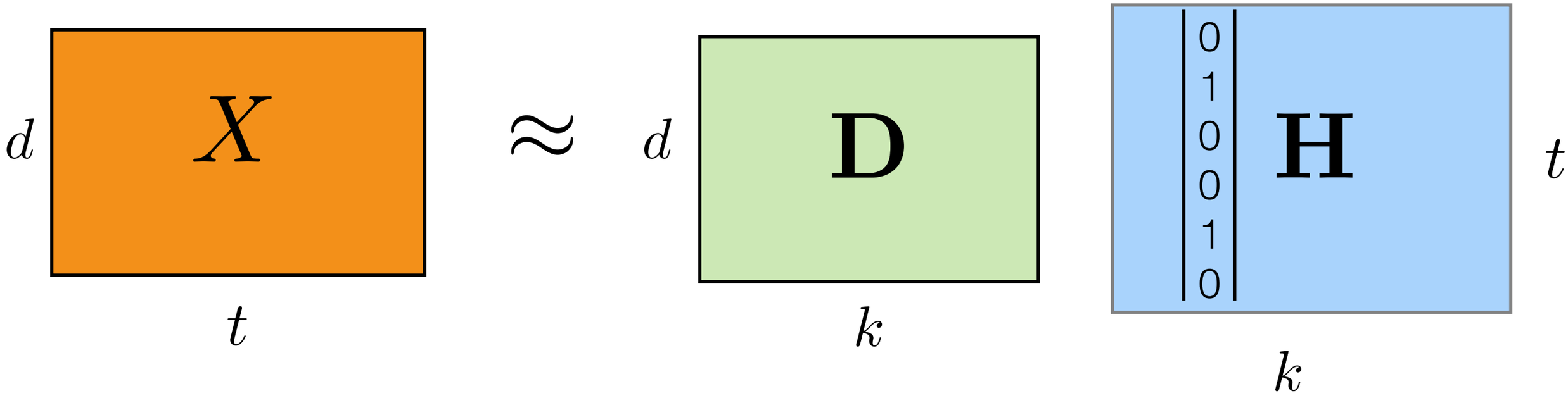
- For several settings, we have closed form solutions
  - PCA, CCA, ISOMAP, …
- For others, we do not
  - sparse coding, exponential family PCA, …

PCA



If k < d, then we obtain dimensionality reduction (PCA)

Sparse coding



- For sparse representation, usually k > d
- Many entries in new representation are zero

Exercise

- PCA linearly decomposes the data matrix X = DH
- Another view of this is that X is projected, to get PX
- Based on your knowledge of the solution of PCA, what does this projection look like?
- Does that mean learning with a PCA representation still gives a linear predictor, in terms of the original features?