COMPUTER SCIENCE
INDIANA UNIVERSITY
School of Informatics and Computing
Bloomington

Representation learning

Reminders/Comments

- Assignment 2 due today
- Thank you for the feedback!
- Motivation for math: math is fun!
  - Real reason: machine learning is heavily math-based
  - even understanding new techniques that you might implement require a basic understanding of optimization and probabilities
  - understanding the algorithm is difficult without understanding the derivation; understanding the algorithm means being better equipped to implement it and better equipped to modify it for your purposes
- Pace of derivations:
  - for the coming neural networks derivation, will have a chance to apply similar gradient descent techniques from before; this derivation I will do very slowly as a chance to clear up any issues with matrices and gradients

Pop quiz question 1

- Briefly describe the difference difference between MAP estimation and ML estimation.

Question 2

- Briefly describe the difference between batch and stochastic gradient descent for optimization.

Questions 3 and 4

- What is the definition of E[X]?
- What is the difference between estimating p(y l x) and E[Y l x]?

Question 5: types of data

- Imagine you have a dataset of 5 points, with d-dimensional features.
  - (a) If the corresponding targets are {-3.0, 2.2, -5.3, -1.0, 4.3}, then what estimation technique might you use?
  - (b) If the corresponding targets are {1.0, 6.0, 3.0, 2.0, 2.0} and you know y is always a positive integer, then what estimation technique might you use?
  - (c) If the corresponding targets are {1, 2, 3, 2, 1} and you know y is always in {1, 2, 3}, then what estimation technique might you use?

Topics so far

- Basics of probabilities, including PMFs (discrete values) and PDFs (continuous values)
- Basics of parameter estimation: MAP and ML
- Generalized linear models
  - linear regression
  - Poisson regression
  - logistic regression
  - multinomial regression
- Generative and discriminative
  - e.g. naive Bayes vs logistic regression

Topics so far

- Solving an optimization
  - write down the (negative) log-likelihood
  - taking the gradient and trying to find the point where it is zero
  - either we have a closed form solution (formula w = function(x, y))
  - or we have to do gradient descent to reach a stationary point where the gradient is zero
  - we can use the Hessian to check properties of the stationary point and for second-order gradient descent to speed convergence
- Practical issues
  - collinear features making the closed form solution unstable
  - regularization to improve stability and avoid overfitting
  - huge datasets, making stochastic gradient descent more viable

Algorithms and techniques

- The algorithms themselves are fundamental algorithms to ML, but also constitute simple examples of the general parameter estimation and optimization techniques

- More advanced algorithms build off of these basic techniques

- In many ways, the problem specification techniques and optimization techniques are the most important topic
  - rather than the algorithms themselves

- By now, you hopefully understand
  - the fundamental problems (e.g., regression, classification)
  - how to formally specify these problems as optimization (based on probabilities to model the uncertainty)
  - how to solve those optimizations, for the simple cases we’ve done

Representation learning

- Generalized linear models enabled many p(y I x) distributions
  - Underneath, still learning a linear representation for E[y I x], which may not have enough representation capacity
- Approach we discussed earlier: augment current features x using polynomials
- There are many strategies to augmenting x
  - fixed representations, like polynomials, wavelets
  - learned representations, like neural networks and matrix factorization

Polynomial representations

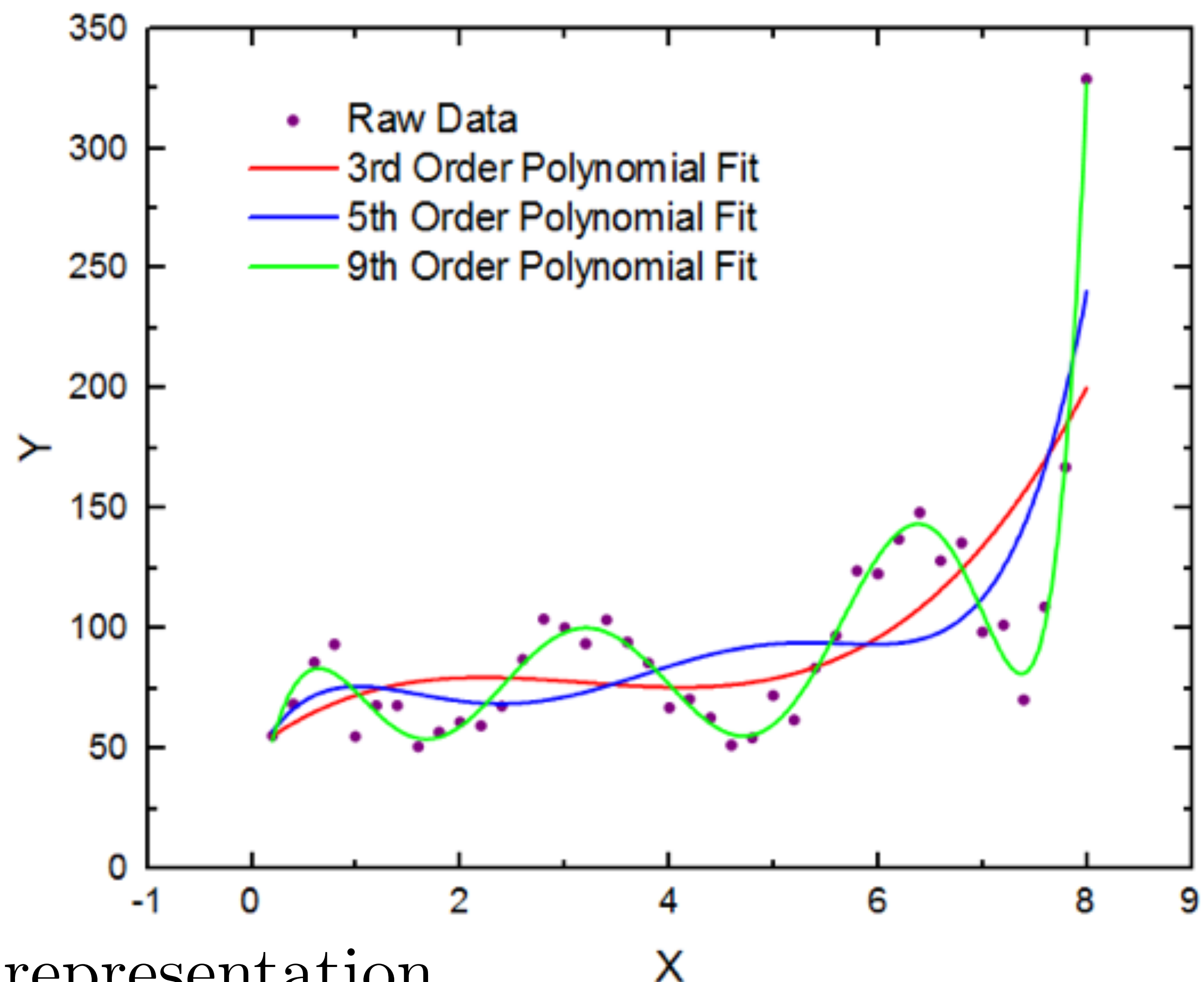
- Using Taylor series, many functions (any function we care about mostly) can be represented as a (high-order) polynomial

$$\mathbf{x} \rightarrow$$

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

$$w_6 x_1^2 + w_5 x_2^2 + w_4 x_1 x_2$$

$$+ w_2 x_2 + w_1 x_1 + w_0$$



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

$$\text{transformed to more powerful representation}$$

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

Radial basis function network

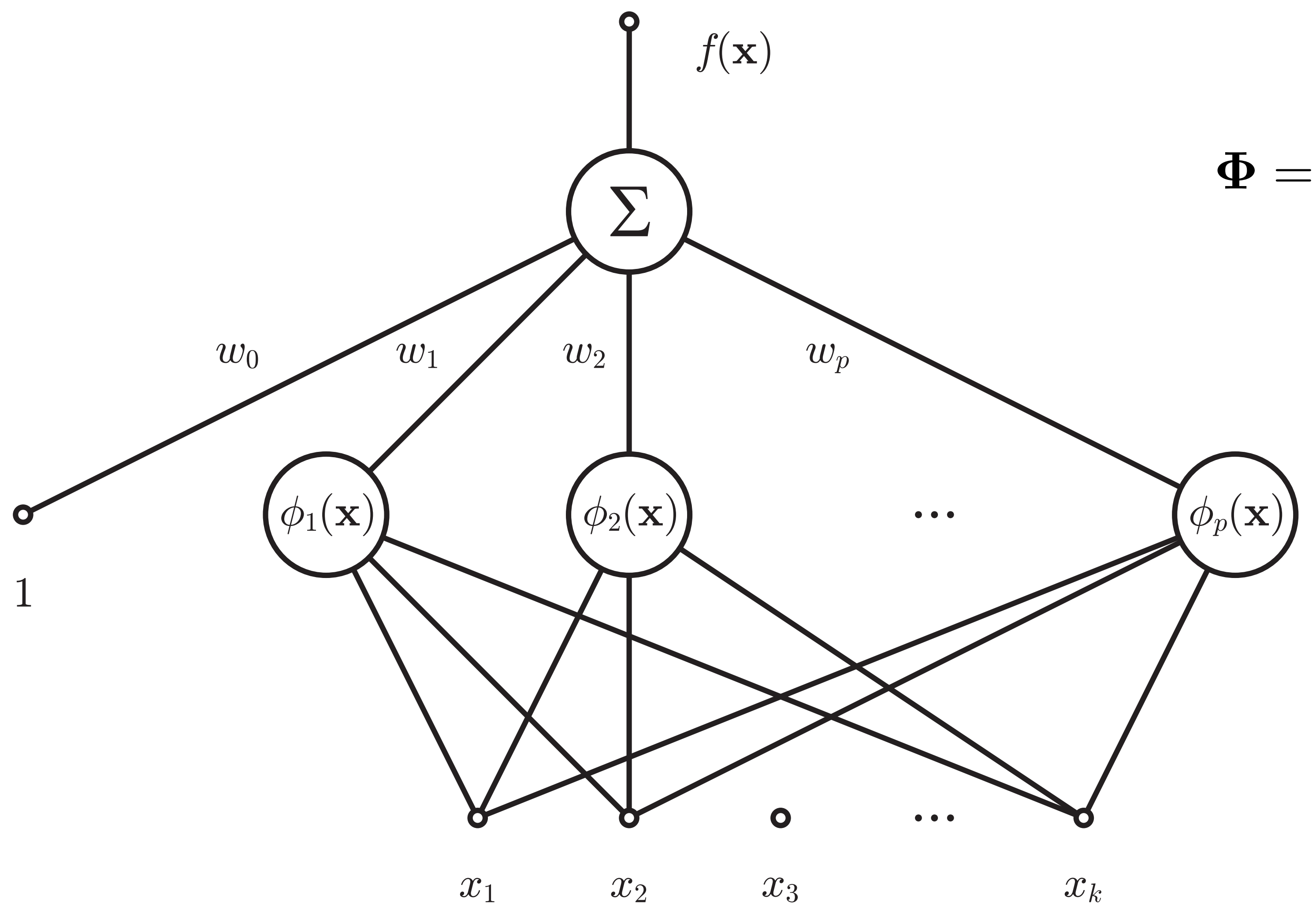


Figure 7.1: Radial basis function network.

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

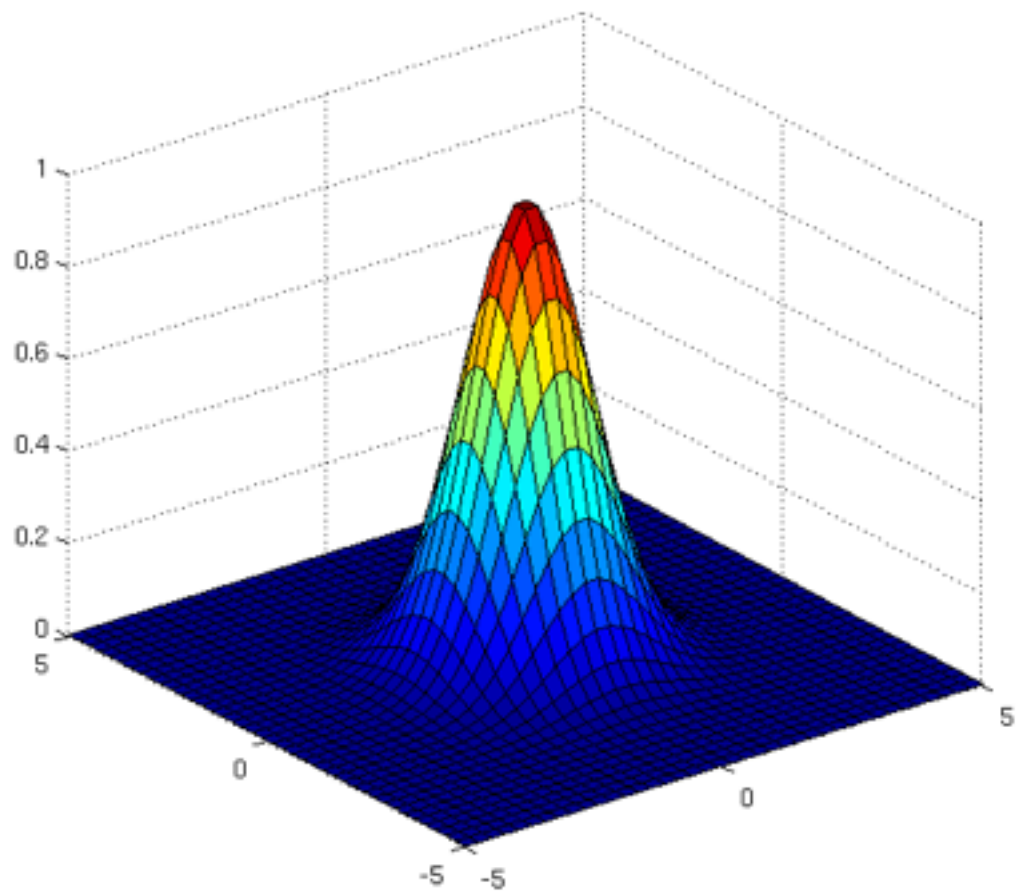
e.g., $\phi_j(\mathbf{x}) = e^{-\frac{\left\|\mathbf{x}-\mathbf{c}_j\right\|^2}{2\sigma_j^2}}$,

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

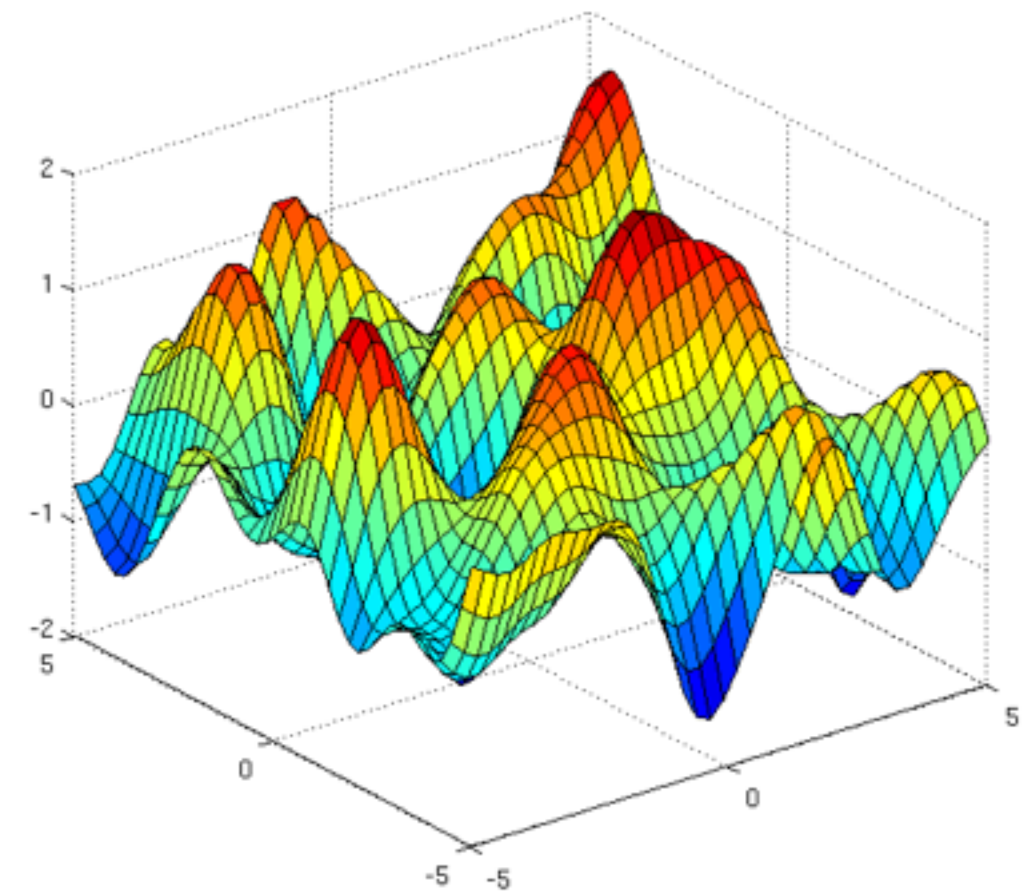
Gaussian kernel / Gaussian radial basis function

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

Kernel



Possible function f with several centers



Selecting centers

- Many different strategies to decide on centers
  - many ML algorithms use kernels as basis, including SVMs, Gaussian process regression
- For kernel representations, typical strategy is to select training data as center
- Clustering techniques to find centers
- A grid of values to best (exhaustively) cover the space
- Other strategies based on information gain

Another way to think of kernels

- Kernel function k(xi, xj) need not be a radial basis function
- Every kernel function k(xi, xj) = < phi(xi), phi(xj) >, for some augmented feature representation phi(xi)
- So can think of kernel representation / RBFs as
  - using similarity features to prototypes or representative instances
  - OR using some higher-dimensional, implicit representation phi(x) that may be a highly non-linear useful representation —> for this interpretation, your algorithm has to simplify to only using dot product between phi vectors (such as kernel regression or SVMs)

Example: polynomial kernel

$$\boldsymbol{\phi}(\mathbf{x}) = \begin{bmatrix} \mathbf{x}_1^2 \\ \sqrt{2}\mathbf{x}_1\mathbf{x}_2 \\ \mathbf{x}_2^2 \end{bmatrix}$$

$$k(\mathbf{x}, \mathbf{x}') = \langle \boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{x}') \rangle = \langle \mathbf{x}, \mathbf{x}' \rangle^2$$

In general, for order d polynomials, $k(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle^d$

Example: Gaussian kernel

- For a Gaussian kernel, phi(x) is an infinite dimensional vector, even though we know the kernel is

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(\frac{-\|\mathbf{x} - \mathbf{x}'\|_2^2}{\sigma^2}\right) = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle = \phi(\mathbf{x})^\top \phi(\mathbf{x}') \in \mathbb{R}$$

- Implicitly, still learning phi(x) w = y

- How do we avoid using phi(x) explicitly, for both training and prediction?

Example: kernel linear regression

Recall $\mathbf{w} = (\mathbf{\Phi}^\top \mathbf{\Phi})^{-1} \mathbf{\Phi}^\top \mathbf{y}$

Exercise: $(\mathbf{\Phi}^\top \mathbf{\Phi})^{-1} \mathbf{\Phi}^\top = \mathbf{\Phi}^\top (\mathbf{\Phi} \mathbf{\Phi}^\top)^{-1}$

Therefore $\mathbf{w} = \mathbf{\Phi}^\top \mathbf{a}$

$$\begin{aligned} ||\mathbf{\Phi} \mathbf{w} - \mathbf{y}||_2^2 &= ||\mathbf{\Phi} \mathbf{\Phi}^\top \mathbf{a} - \mathbf{y}||_2^2 \\ &= ||\mathbf{K} \mathbf{a} - \mathbf{y}||_2^2 \end{aligned}$$

$$\text{where } K_{ij} = k(\mathbf{x}_i, \mathbf{x_j}) = < \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) >$$

$$\text{Prediction on } \mathbf{x} : \begin{bmatrix} k(\mathbf{x}, \mathbf{x}_1) \\ \vdots \\ k(\mathbf{x}, \mathbf{x}_n) \end{bmatrix}^\top \mathbf{a} \approx \mathbf{y}$$

where $\mathbf{a} \in \mathbb{R}^n$

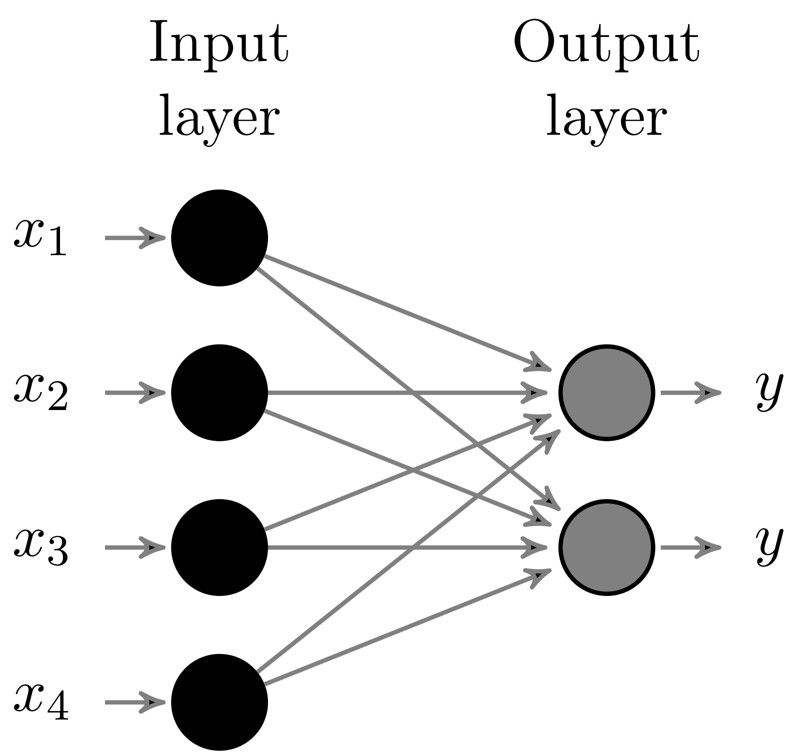
Other fixed representations

- Fourier basis
- Wavelets
- Tile coding (also called CMAC for cerebellar model articulation controller)

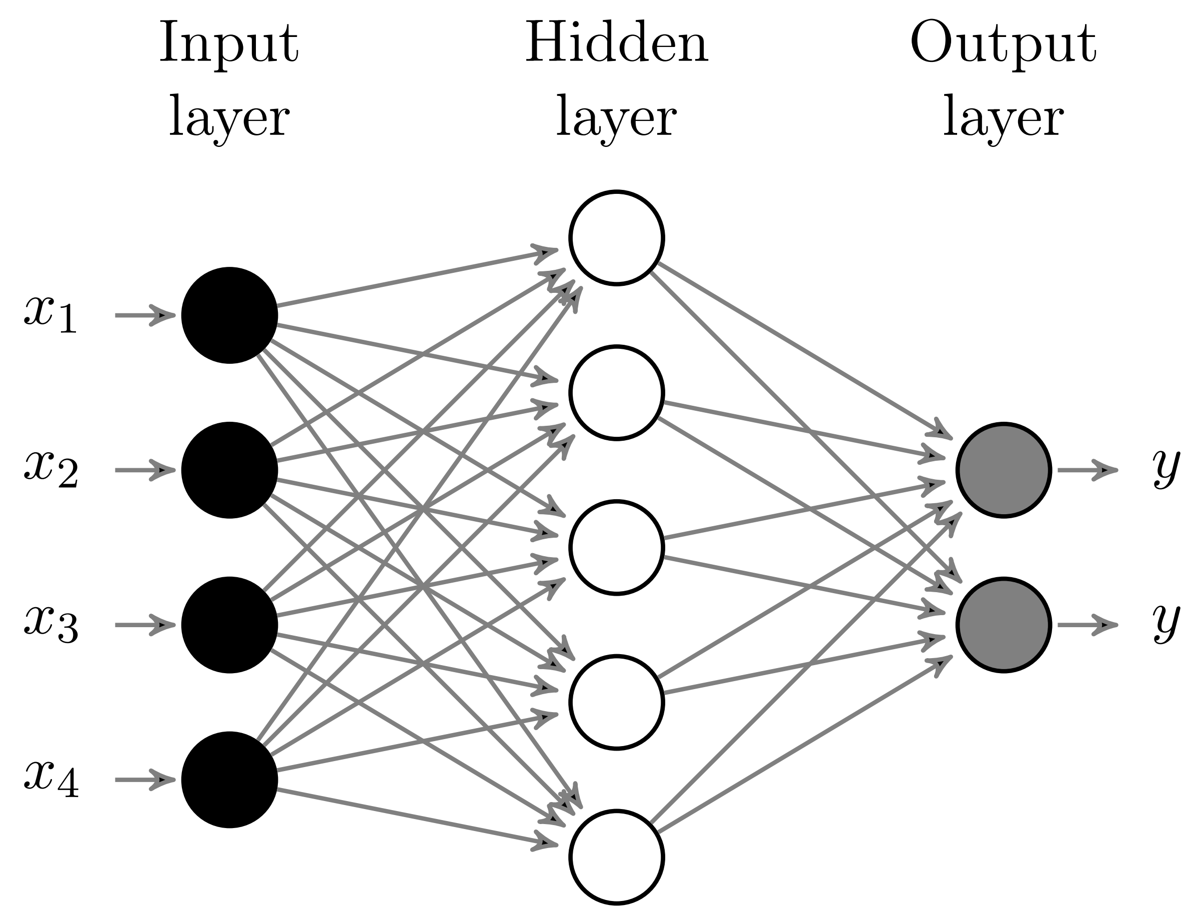
Learning representations

- One way to learn a representation is to learn the parameters to the previously mentioned fixed representations
    - e.g. could learn bandwidth sigma to Gaussian RBF
- There are, however, strategies for learning a representation more from scratch; we will focus on two main ones
    - Neural networks
    - Matrix factorization techniques (regularized factor models)

Generalized linear model vs. neural network



GLM
(e.g. logistic regression)



Two-layer neural network