Regression

CMPT 882

Mar. 11

Outline

- Probability Overview
- Regression
- Classification

Regression

- Given $x \in \mathbb{R}^n$
  - “features”, “covariate”, “predictors”

- Predict $y \in \mathbb{R}^m$
  - “response”, “outputs”

- Learn the function $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$ such that $y \approx f(x)$
  - f is the model for regression
  - Use data: $\{x_i, y_i\}_{i=1}^{N}$

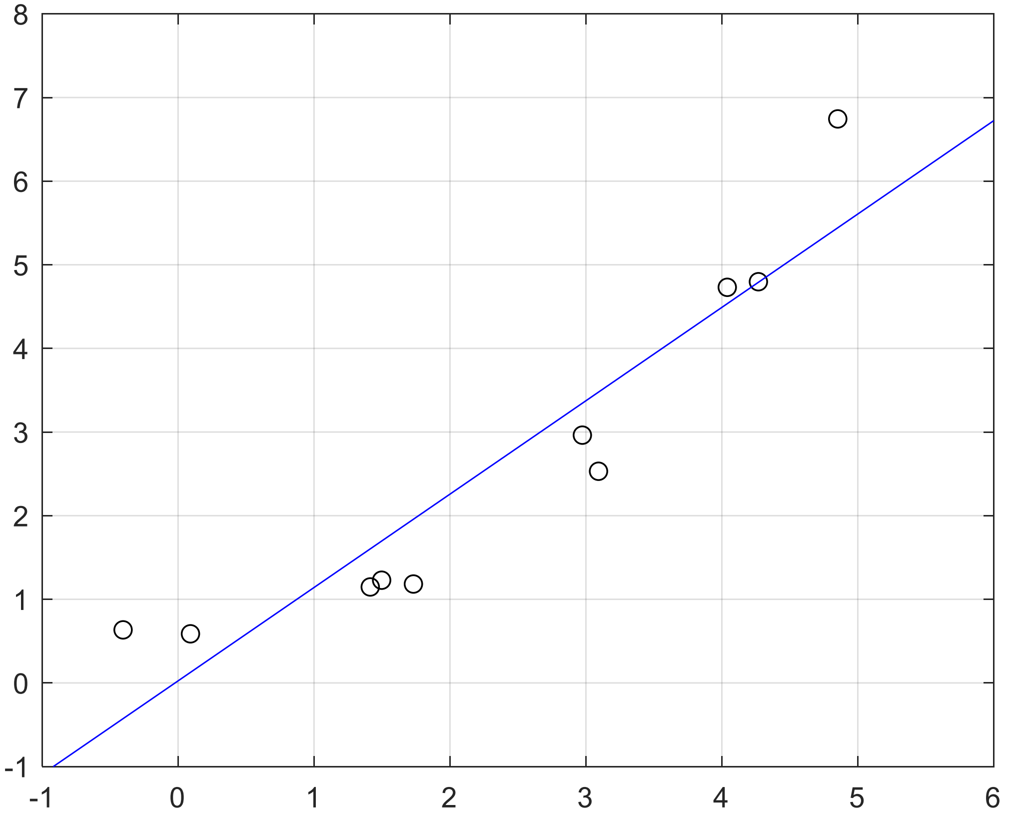
- Parametrize the function f using the parameters θ → $y \approx f_\theta(x)$
  - θ and the form of f determines the class of functions in your model
  - Learning f → learning parameters θ

Regression

- Supervised learning is regression
  - f_θ is determined through “supervision” by data $\{x_i, y_i\}$
- Deep learning is regression using a neural network
  - Neural network (for now): complex f_θ with many components in θ
- Neural networks are hard to analyze, but analyzing regression with simple(r) models provides good intuition

Models for Regression

- Simplest model: Linear
  - $y = \theta^\top x + \epsilon$, where ϵ is noise
- Put data into matrix vector form: (scalar y)
  - $X = \begin{pmatrix} -x_1^\top - \\ \vdots \\ -x_N^\top - \end{pmatrix} \in \mathbb{R}^{N\times n}, Y = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix} \in \mathbb{R}^N$
  - Minimize loss function: $l(\theta) = \|Y - X\theta\|_2^2$
    - Seems to make sense if noise is zero-mean

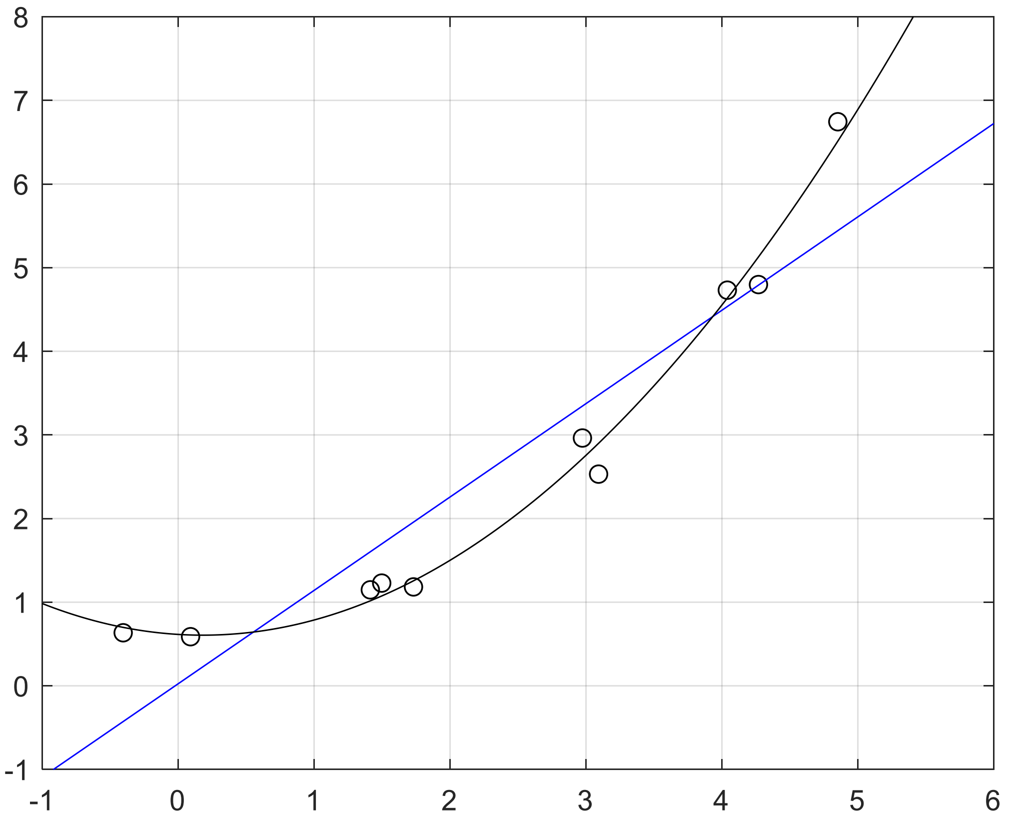


$$\theta^* = \arg\min_\theta \|Y - X\theta\|_2^2$$

$$\theta^* = (X^T X)^{-1} X^T Y$$

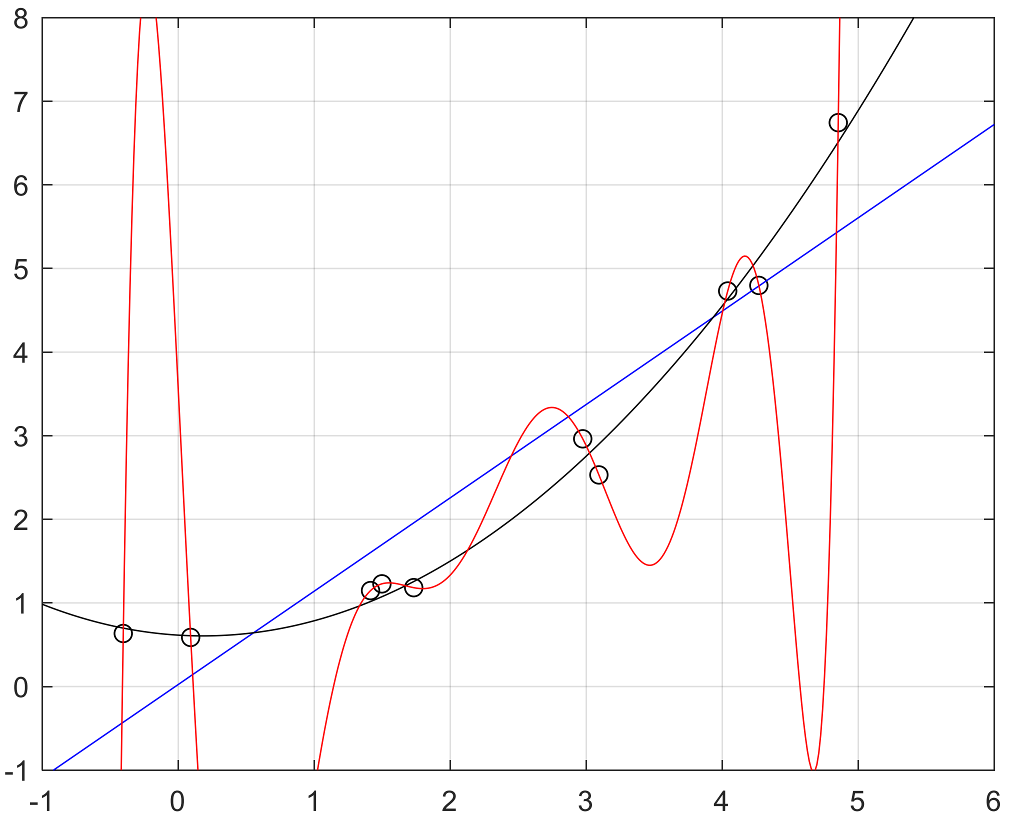
Feature Augmentation

- Raw data: $\{x_i, y_i\}_{i=1}^N$
  - But perhaps $y = f(x)$ is nonlinear
  - Augment data: $\bar{x}_i = (1, x_i, x_i^2), \bar{y}_i = y_i$
- Use linear model between $\bar{x}$ and $\bar{y}$
  - $\bar{y} = \theta^\top \bar{x} + \epsilon = \theta_0 + \theta_1 x + \theta_2 x^2 + \epsilon$
  - Effectively a quadratic model



Feature Augmentation

- Raw data: $\{x_i, y_i\}_{i=1}^{N}$
  - But perhaps $y = f(x)$ is nonlinear
  - Augment data: $\bar{x}_i = (1, x_i, x_i^2), \bar{y}_i = y_i$
- Use linear model between $\bar{x}$ and $\bar{y}$
  - $\bar{y} = \theta^\top \bar{x} + \epsilon = \theta_0 + \theta_1 x + \theta_2 x^2 + \epsilon$
  - Effectively a quadratic model

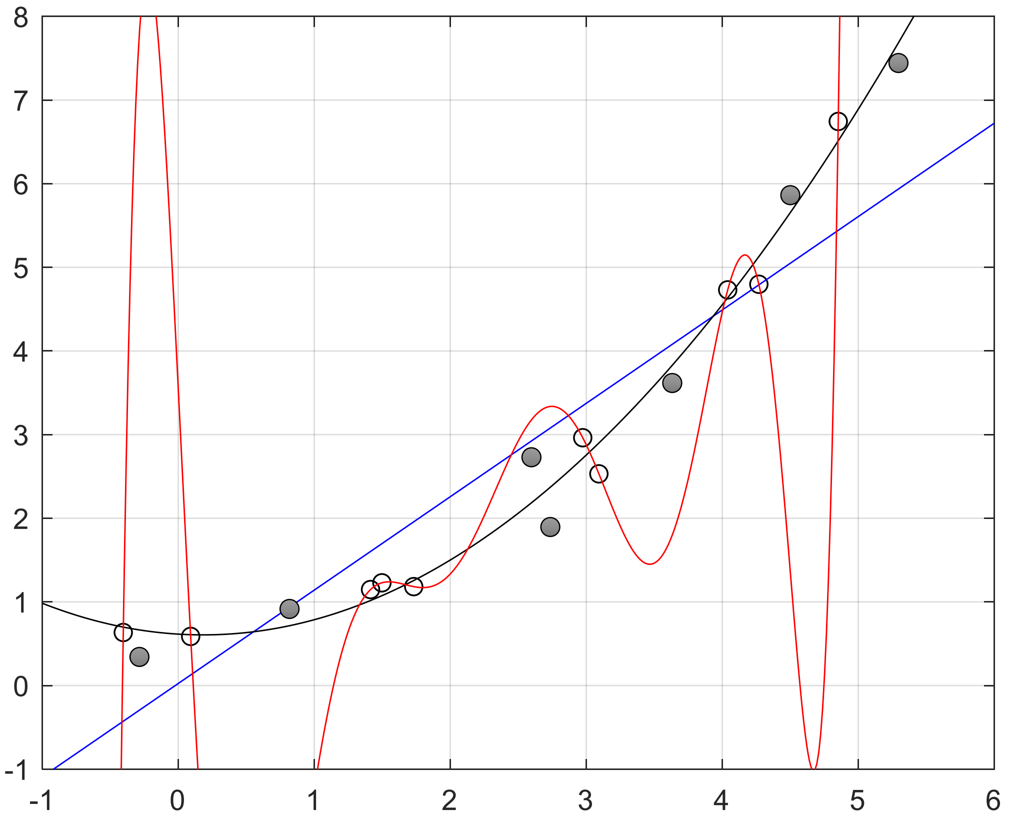


- In general, $\bar{x}_i = (1, x_i, x_i^2, \dots x_i^N)$ → degree N polynomial
  - Correspondingly, more parameters are required

Observations

- More parameters → less training error, but potentially more test error
  - **Training error**: error when fitting model f_θ to data
  - **Test error**: error when using model to do prediction

- In our example, the true model is quadratic
  - High order polynomial would have very large test errors → overfitting
  - In general, the true model is unknown



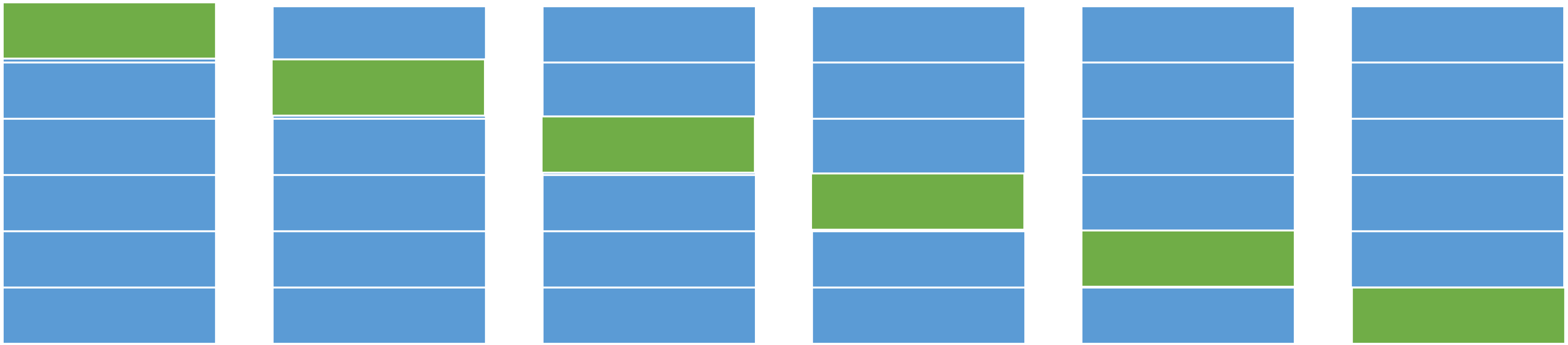
Addressing Overfitting

- Validation of Trained Models (hold-out data)
  - Divide data up into training and validation (hold-out) data
  - Do training on the training data → minimize training error
  - Validate the model on validation data → obtain **validation error**

- Regularization
  - Add penalty to size of parameters

N-Fold Cross Validation

- Divide data into N (roughly) equal parts
- Go through each part
  - Do training on the other $N - 1$ parts (so one part is hold-out)
  - Evaluate model on the hold-out data to get → validation error
- Validation error is the average of all validation errors from above
  - Approximates performance during **test**, where new data is generated



Regularization

- Previously: $l(\theta) = \|y - X\theta\|_2^2$
  - Example: $l(\theta) = \sum\left(y_i - \theta_0 - \theta_1 x_i - \theta_2 x_i^2 - \theta_3 x_i^3 - \theta_4 x_i^4\right)^2$

- L2 regularization:
  - Heuristic: the underlying ground truth model does not have large θ
  - $l(\theta) = \|Y - X\theta\|_2^2 + \lambda\|\theta\|_2^2$
  - "**Tikhonov regularization**"
  - Statistics: "**ridge regression**"
  - Machine learning: "**weight decay**"

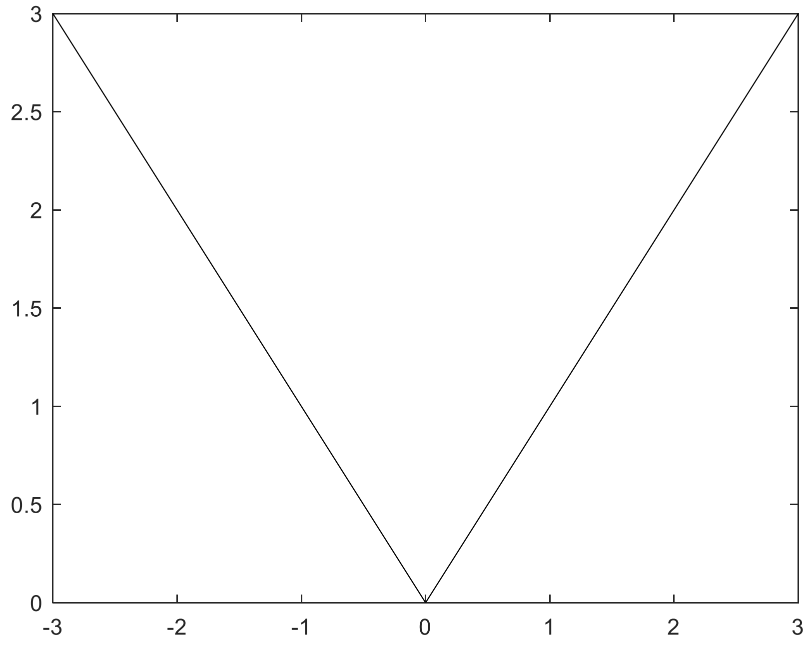
- L1 regularization:
  - Heuristic: many parameters in the underlying ground truth model are 0
  - $l(\theta) = \|Y - X\theta\|_2^2 + \lambda\|\theta\|_1$
  - Statistics: "**LASSO**"
  - Signal processing: "**basis pursuit**"

- "**Elastic net regularization**": combination of both
  - $l(\theta) = \|Y - X\theta\|_2^2 + \lambda\left((1-\alpha)\|\theta\|_2^2 + \alpha\|\theta\|_1\right)$

Regularization

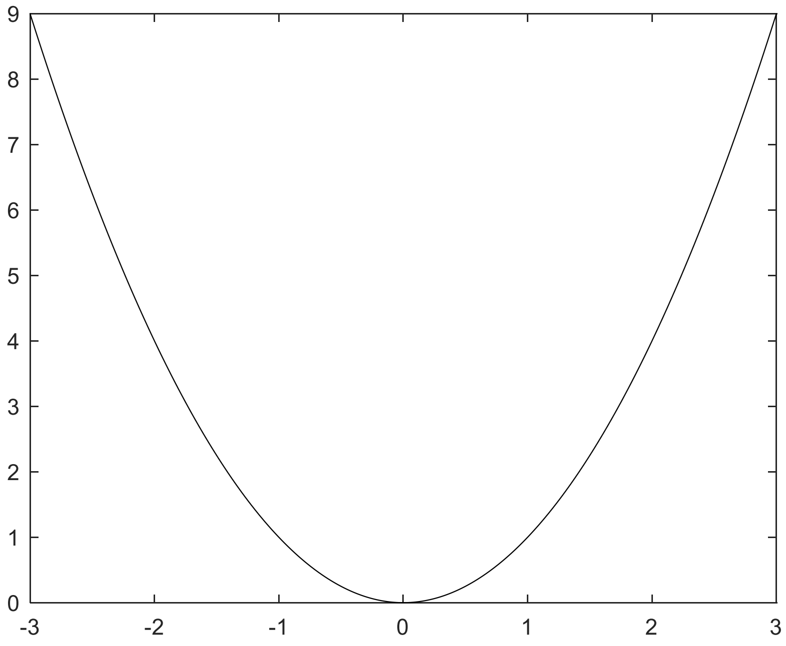
L1: $\|\boldsymbol{\theta}\|_{\mathbf{1}} = \sum_{i} |\boldsymbol{\theta}_{i}|$

- Does not prioritize reduction of any component of θ
- Encourages sparsity



L2: $\|\boldsymbol{\theta}\|_{2} = \sum_{i} \boldsymbol{\theta}_{i}^{2}$

- Prioritizes reduction of large components of θ



Outline

- Probability Overview
- Regression
- Classification

Probability Review

- Sensor measurements and robot state are modeled as random variables
  - Random variables denoted with upper case: X
  - Realized, specific value denoted with lower case: x

- Discrete random variable
  - Probability mass function (pmf)

    $$p(x) \coloneqq P(X = x)$$
  - $\sum_x p(x) = 1$

- Continuous random variable
  - Probability density function (pdf)

    $$P(X \in [a, b]) = \int_a^b p(x)dx$$
  - $\int_{-\infty}^{\infty} p(x)dx = 1$

Basic Properties of Random Variables

- Joint distribution

  $$p(x, y) \coloneqq P(X = x \text{ and } Y = y)$$

  - If X and Y are independent, then

    $$p(x, y) = p(x)p(y)$$

- Condition probability
  - The probability that $X = x$ given that we know $Y = y$, denoted $p(x|y)$

    $$p(x|y) \coloneqq \frac{p(x, y)}{p(y)}$$

  - If X and Y are independent, then

    $$p(x|y) = p(x)$$

  - Useful re-arrangement

    $$p(x, y) = p(x|y)p(y)$$

Theorem of total probability and Bayes' rule

Discrete random variables

- $p(y) = \sum_x p(x, y) = \sum_x p(y|x)p(x)$

Continuous random variables

- $p(y) = \int p(x, y)dx = \int p(y|x)p(x)dx$

- $p(x, y) = p(x|y)p(y) = p(y|x)p(x)$
- Isolate $p(x|y)$ to obtain Bayes' rule

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}$$

- Using law of total probability

$$p(x|y) = \frac{p(y|x)p(x)}{\sum_x p(y|x)p(x)}$$

$$p(x|y) = \frac{p(y|x)p(x)}{\int p(y|x)p(x)dx}$$

- Notational simplification: $p(x|y) = \eta p(y|x)p(x)$

Expectation and variance

Discrete random variables

- $E[X] = \sum_x xp(x)$

Continuous random variables

- $E[X] = \int xp(x)dx$

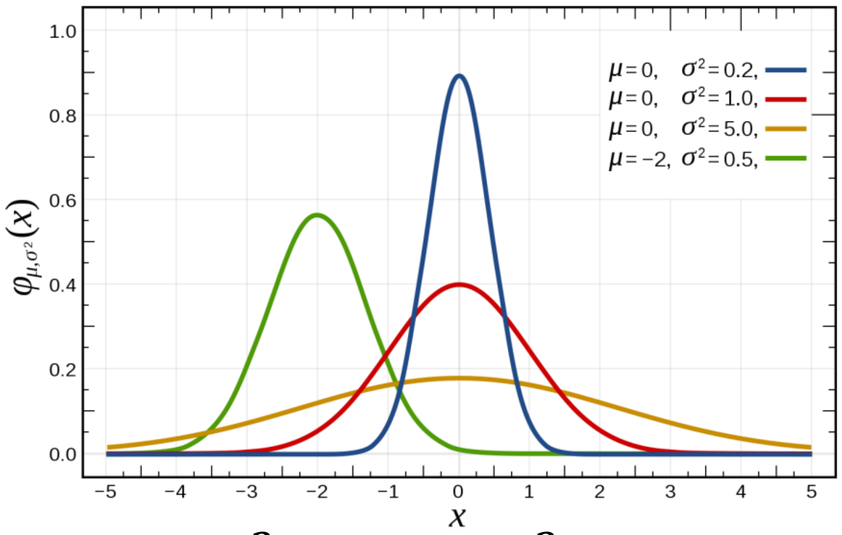
- Expectation is a linear operator

$$E[aX + b] = aE[X] + b$$

- Covariance:

$$\mathrm{cov}(X, Y) = E[(X - E[X])(Y - E[Y])] = E[XY^\top] - E[X]E[Y]^\top$$

Maximum Likelihood



- Simplest model: Linear
  - $y = \theta^\top x + \epsilon$, where ϵ is noise
  - Assume noise is normally distributed with zero mean and variance σ^2: $\epsilon \sim N(0, \sigma^2)$
  - $P_\theta(y|x) \sim N(\theta^\top x, \sigma^2)$

- Data consists of $\{x_i, y_i\}_{i=1}^N$
  - Pick the most likely θ
  - $\theta^* = \arg\max_\theta P_\theta(y_1, y_2, \dots, y_N | x_1, x_2, \dots, x_N)$
  - If we assume y_i are independent and identically distributed (i.i.d.), then

$$P_\theta(y_1, y_2, \dots, y_N | x_1, x_2, \dots, x_N) = \prod_{i=1}^N P_\theta(y_i | x_i)$$

Maximum Likelihood

$$\theta^* = \arg\max_\theta P_\theta(y_1, y_2, \dots, y_N | x_1, x_2, \dots, x_N)$$

$$= \arg\max_\theta \prod_{i=1}^{N} P_\theta(y_i | x_i)$$

- Now, use the fact that $P_\theta(y|x) \sim N(\theta^\top x, \sigma^2)$

$$= \arg\max_\theta \left\{ \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y_i - \theta^\top x_i)^2}{2\sigma^2}} \right\}$$

$$= \arg\max_\theta \left\{ e^{-\sum_{i=1}^{N} \frac{(y_i - \theta^\top x_i)^2}{2\sigma^2}} \right\}$$

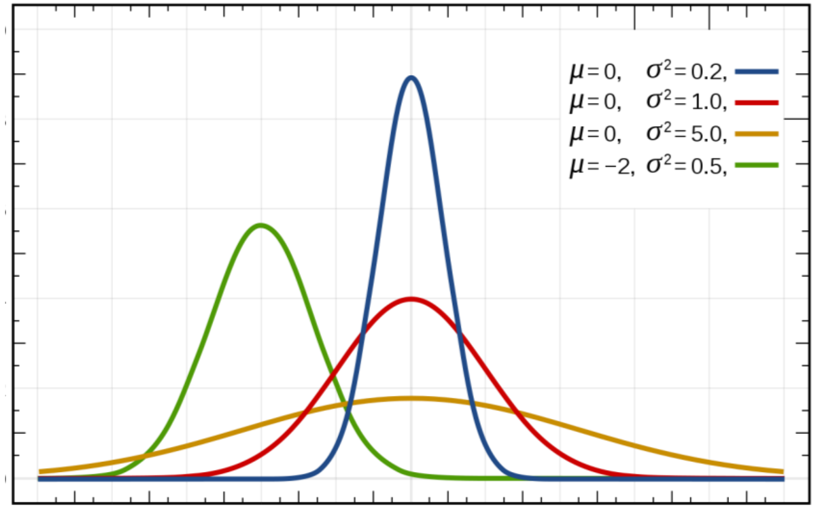
$$= \arg\min_\theta \left\{ \sum_{i=1}^{N} (y_i - \theta^\top x_i)^2 \right\}$$

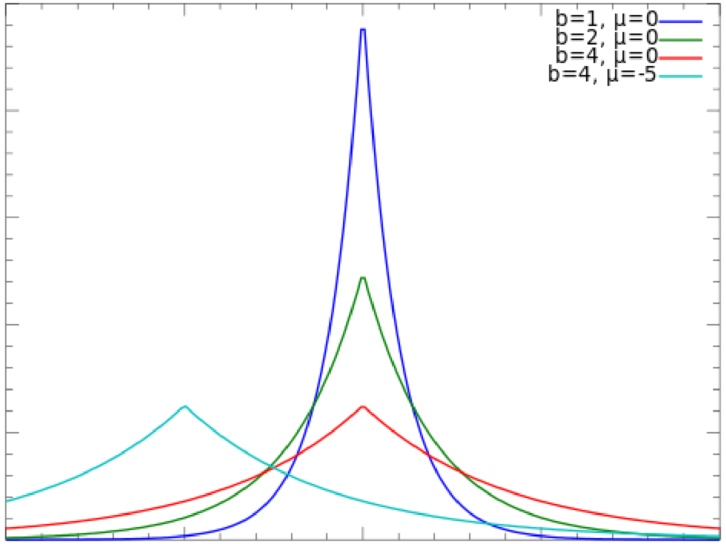
$$= \arg\min_\theta \|y - X\theta\|_2^2$$

Maximum Likelihood vs. 2-Norm Minimization

- Assume noise is normally distributed, then the following are equivalent:
  - θ obtained from maximum likelihood
  - θ obtained from minimizing 2-norm of error
- In general, different loss functions correspond different assumptions about noise and parameter distributions
  - L2 regularization: ϵ normally distributed, θ is normally distributed
  - L1 regularization: ϵ normally distributed, θ Laplacian distributed

Normal distribution





Laplace distribution

Classification

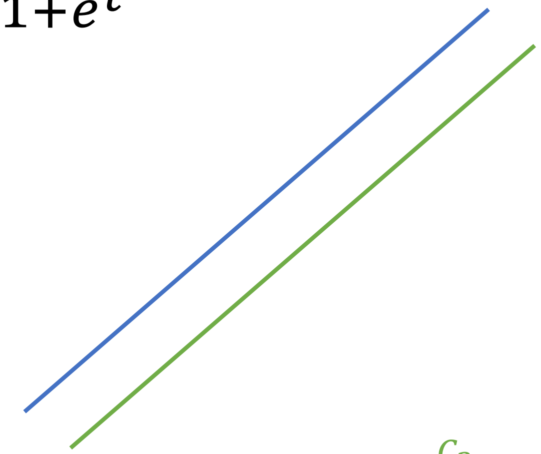
- Given $x \in \mathbb{R}^n$
  - "features", "covariate", "predictors"

- Predict $\boldsymbol{y} \in \{\mathbf{0}, \mathbf{1}\}^{\boldsymbol{m}}$
  - "response", "outputs"
  - Sometimes there may be many values for each component of y
  - For example, in optical character recognition (numbers only), $y \in \{0,1, \ldots, 9\}$

- Learn the function $f\colon \mathbb{R}^n \to \mathbb{R}^m$ such that $y \approx f(x)$
  - Use data: $\{x_i, y_i\}_{i=1}^N$

Logistic Regression

- Common model for binary classification, $y \in \{0,1\}$
- Assume $P_\theta(y = 1|x) = f\left(\sum_{i=1}^{N} \theta_i x^i\right)$ where $f(t) = \frac{e^t}{1+e^t}$
- Interpretation: Suppose $\sum_{i=1}^{N} \theta_i x^i = C$ is fixed
  - Then $P_\theta(y = 1\,|x)$ is fixed, and equal to $\frac{e^C}{1+e^C}$
    - 2D example: $\theta_1 x^1 + \theta_2 x^2 = C$ is a line
  - In addition,
    - $f(t) \to 0$ as $t \to -\infty$
    - $f(t) \to 1$ as $t \to \infty$



Logistic Regression

- Assume $P_\theta(y = 1|x) = f\left(\sum_{i=1}^{N} \theta_i x^i\right)$ where $f(t) = \frac{e^t}{1+e^t}$
  - Observe that $P_\theta(y|x) = \frac{e^{y\theta^\top x}}{1+e^{\theta^\top x}}$

- Maximize the probability by choosing θ

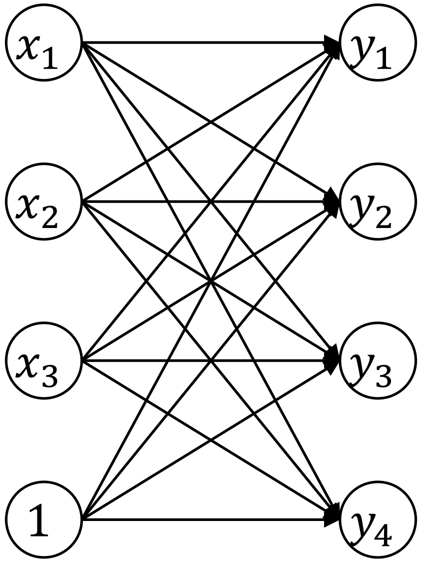
$$\theta^* = \arg\max_\theta \prod_{i=1}^{n} P_\theta(y_i|x_i)$$

Logistic Regression

$$
\begin{aligned}
\theta^* &= \arg\max_{\theta} \prod_{i=1}^{n} P_\theta(y_i|x_i) \\
&= \arg\max_{\theta} \prod_{i=1}^{n} \frac{e^{y_i \theta^\top x_i}}{1 + e^{\theta^\top x_i}} \\
&= \arg\max_{\theta} \log\left(\prod_{i=1}^{n} \frac{e^{y_i \theta^\top x_i}}{1 + e^{\theta^\top x_i}} \right) \\
&= \arg\max_{\theta} \sum_{i=1}^{n} \left(y_i \theta^\top x_i - \log\left(1 + e^{\bar{\theta}^{T} x_i}\right)\right) \\
&= \arg\min_{\theta} \left\{ \sum_{i=1}^{n} \underbrace{\log\left(1 + e^{\theta^\top x_i}\right)}_{g(t) = \log(1+e^t) \text{ is convex}} - \theta^\top \left(\sum_{i=1}^{n} y_i x_i \right) \right\}
\end{aligned}
$$

Neural Networks

- A specific form of $f_\theta(x)$



$$y_1 = f(x^\top w_1 + b_1)$$

$$y_2 = f(x^\top w_2 + b_2)$$

$$y_3 = f(x^\top w_3 + b_3)$$

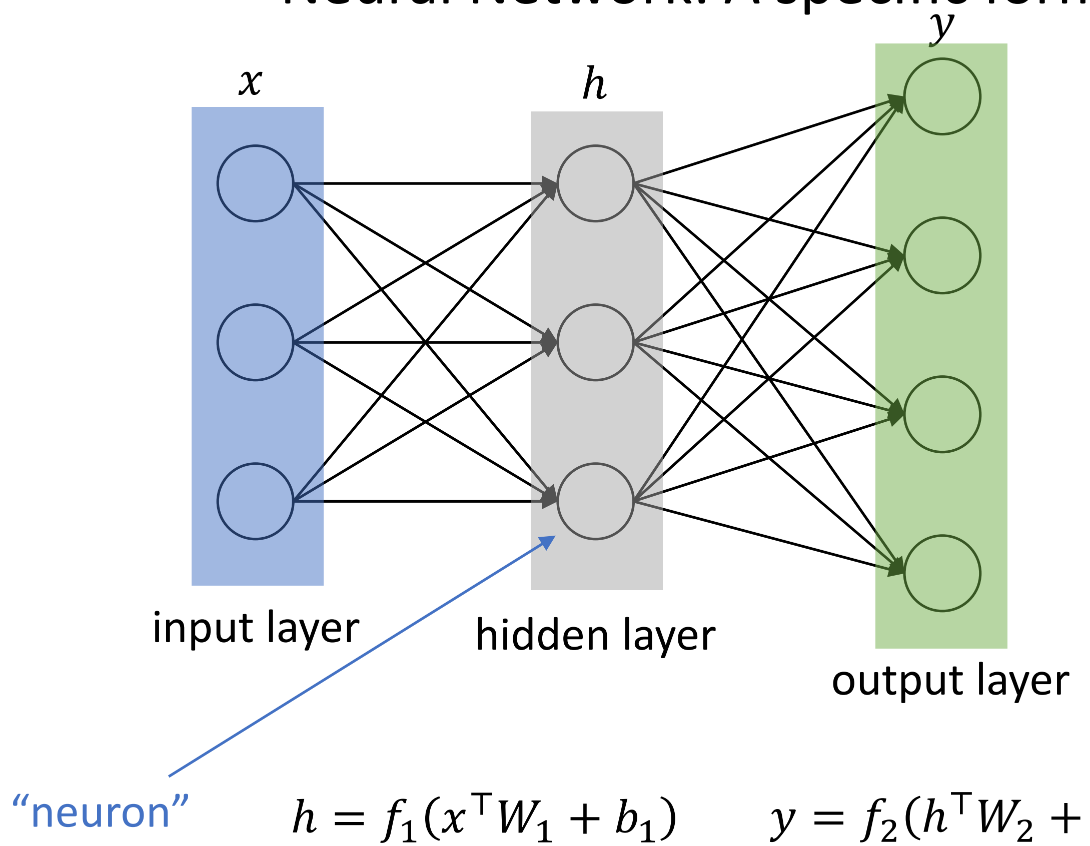
$$y_4 = f(x^\top w_4 + b_4)$$

$$y = f(x^\top W + b)$$

- Parameters θ are W and b
- "**Weights**"

Neural Networks

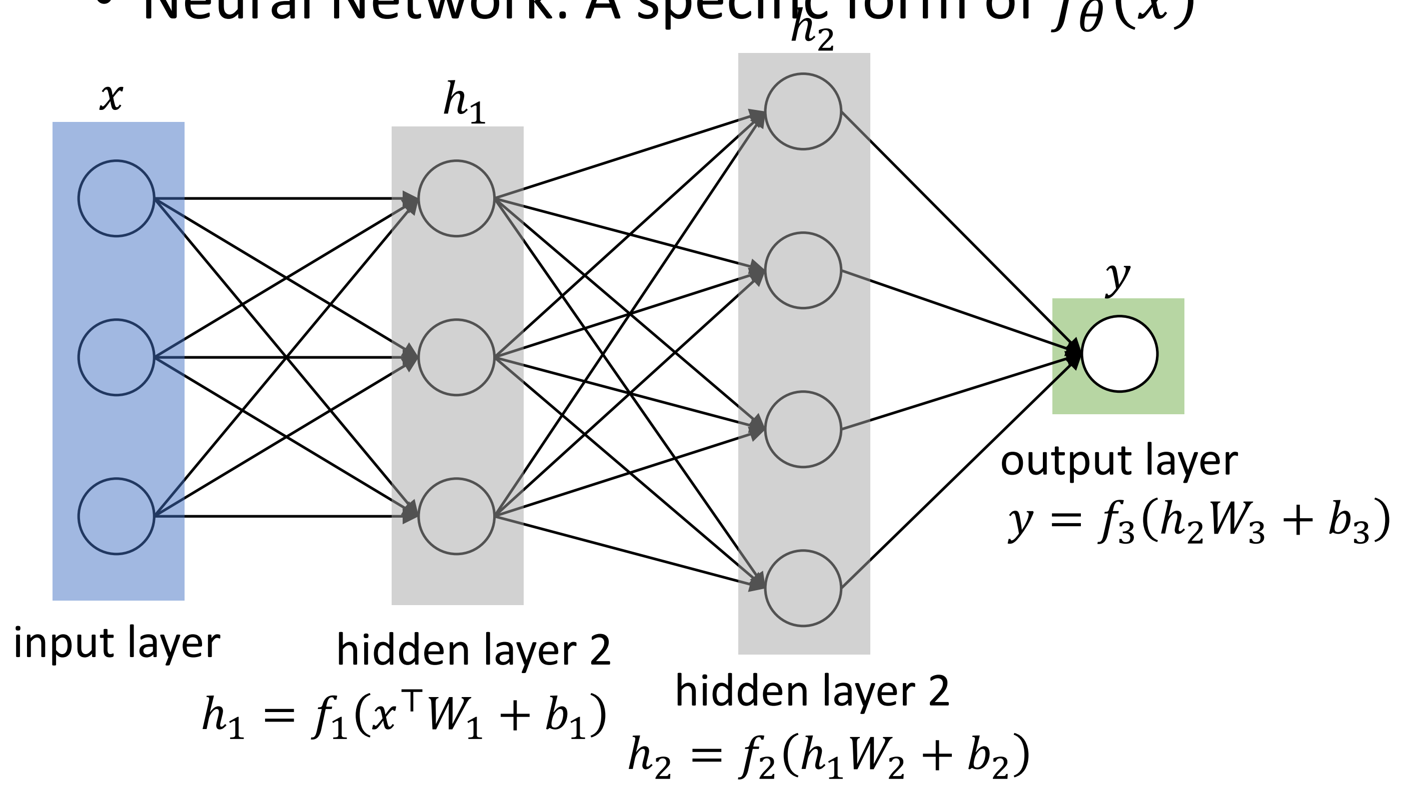
- Regression: Choose θ such that $y \approx f_\theta(x)$
  - Neural Network: A specific form of $f_\theta(x)$



$h = f_1(x^\top W_1 + b_1)$ $\qquad$ $y = f_2(h^\top W_2 + b_2)$

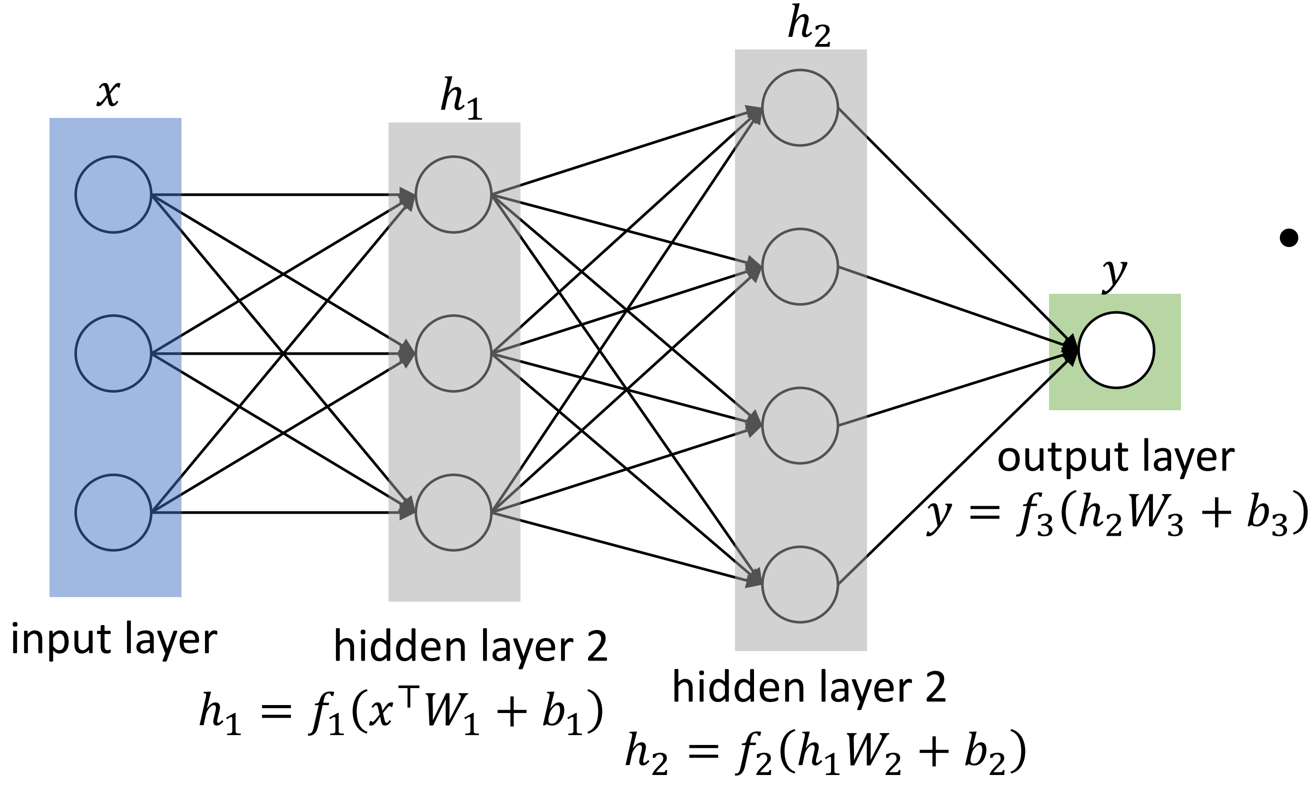
Neural Networks

- Regression: Choose θ such that $y \approx f_\theta(x)$
  - Neural Network: A specific form of $f_\theta(x)$



Neural Networks

- Regression: Choose θ such that $y \approx f_\theta(x)$
  - Neural Network: A specific form of $f_\theta(x)$



- Parameters θ are the weights W_i and b_i

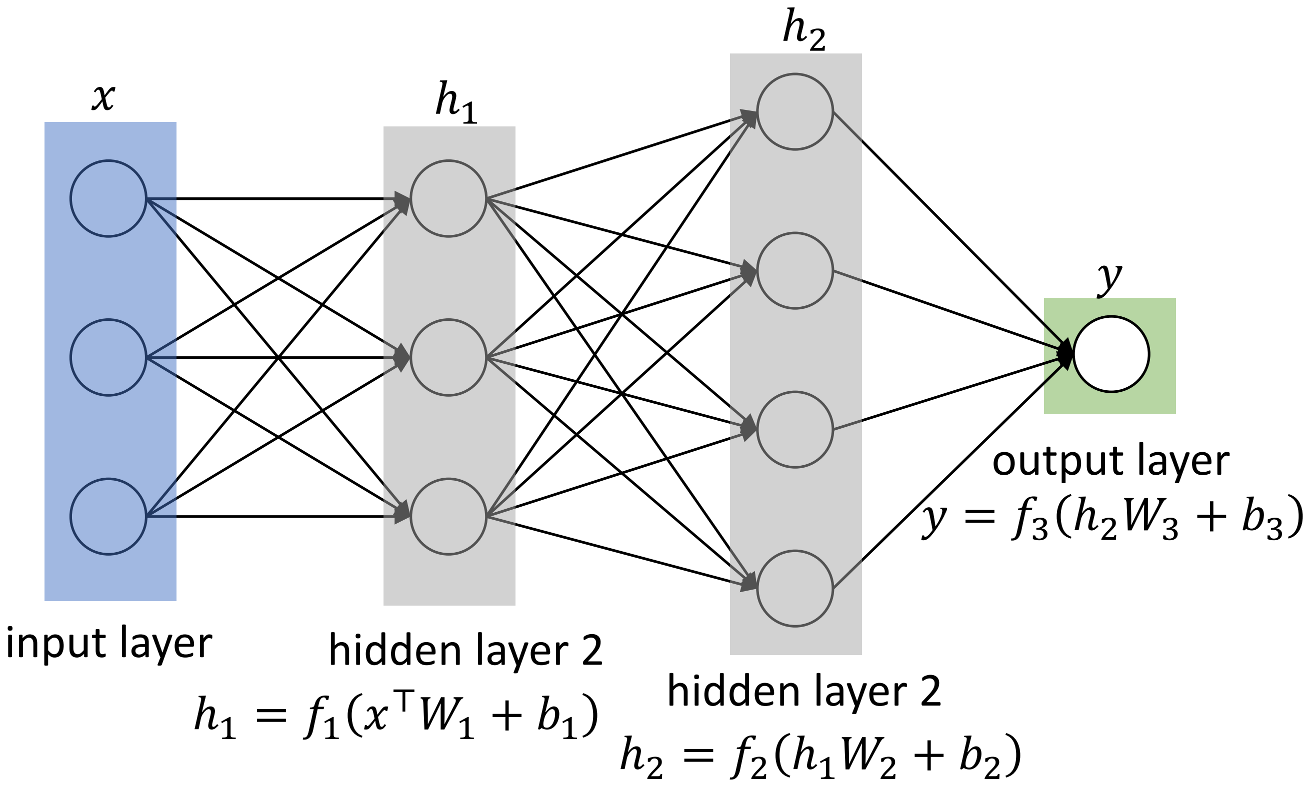
- f_1, f_2, f_3 are nonlinear
  - Otherwise f would just be a single linear function:

$$\begin{aligned} y &= \big((x^\top W_1 + b_1)W_2 + b_2\big)W_3 + b_3 \\ &= x^\top W_1 W_2 W_3 + b_1 W_2 W_3 + b_2 W_3 + b_3 \end{aligned}$$

  - “Activation functions”

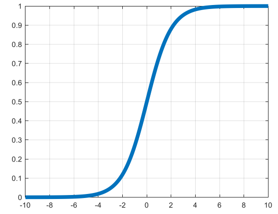
Neural Networks

- Regression: Choose θ such that $y \approx f_\theta(x)$
  - Neural Network: A specific form of $f_\theta(x)$



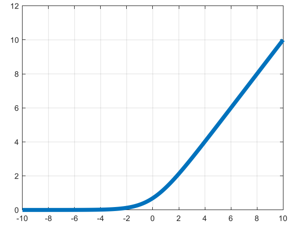
- Common choices of activation functions
  - Sigmoid:

$$\frac{1}{1 + e^{-x}}$$



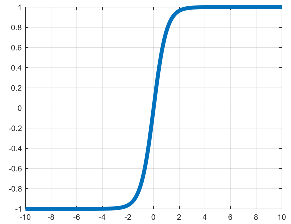
  - Softplus:

$$\log(1 + e^x)$$



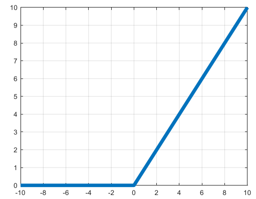
  - Hyperbolic tangent:

$$\tanh x$$



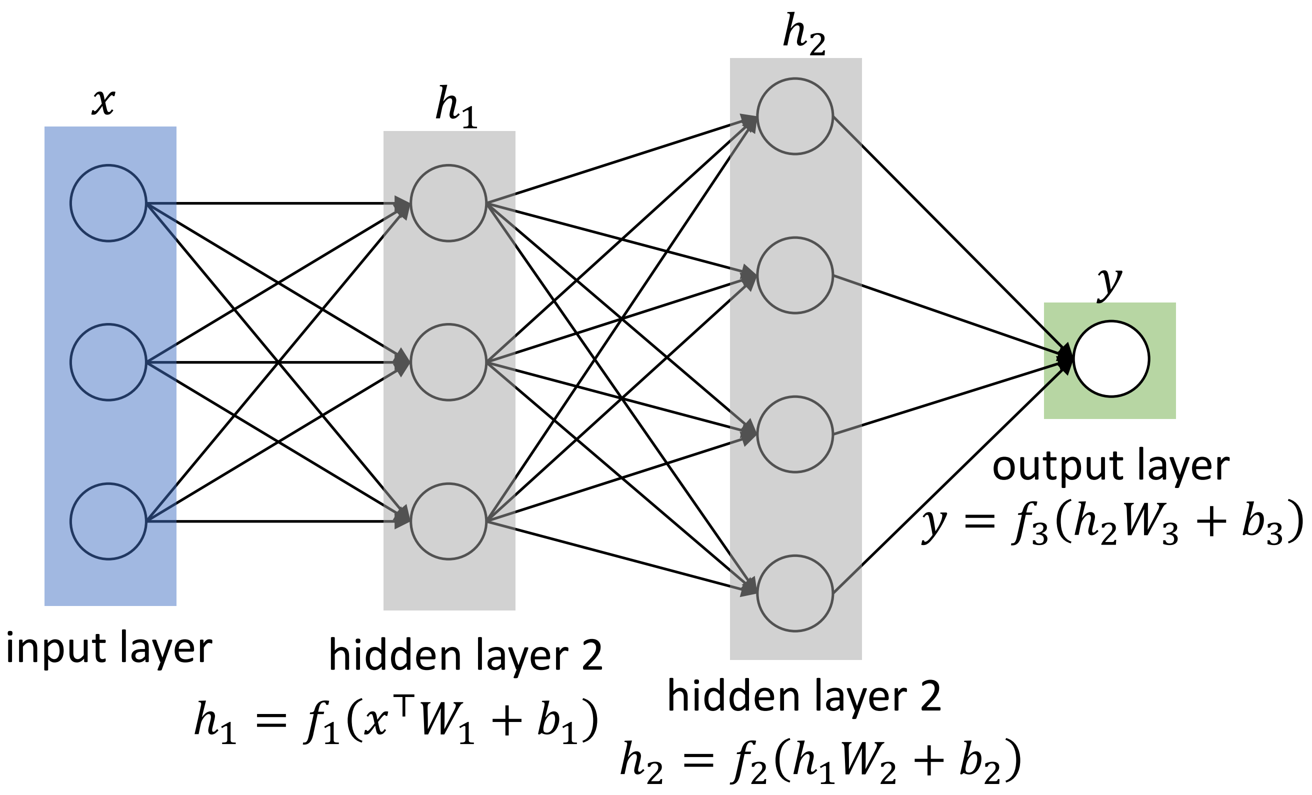
  - Rectified linear unit (ReLU):

$$\max(0, x)$$



Training Neural Networks

- Regression: Choose θ such that $y \approx f_\theta(x)$
  - Neural Network: A specific form of $f_\theta(x)$



- Given current θ, X, Y, compute $l(\theta; X, Y)$
  - Compares $f_\theta(X)$ with ground truth Y
  - Evaluation of f: "**Forward propagation**"

- Minimize $l(\theta; X, Y)$
  - Stochastic gradient descent

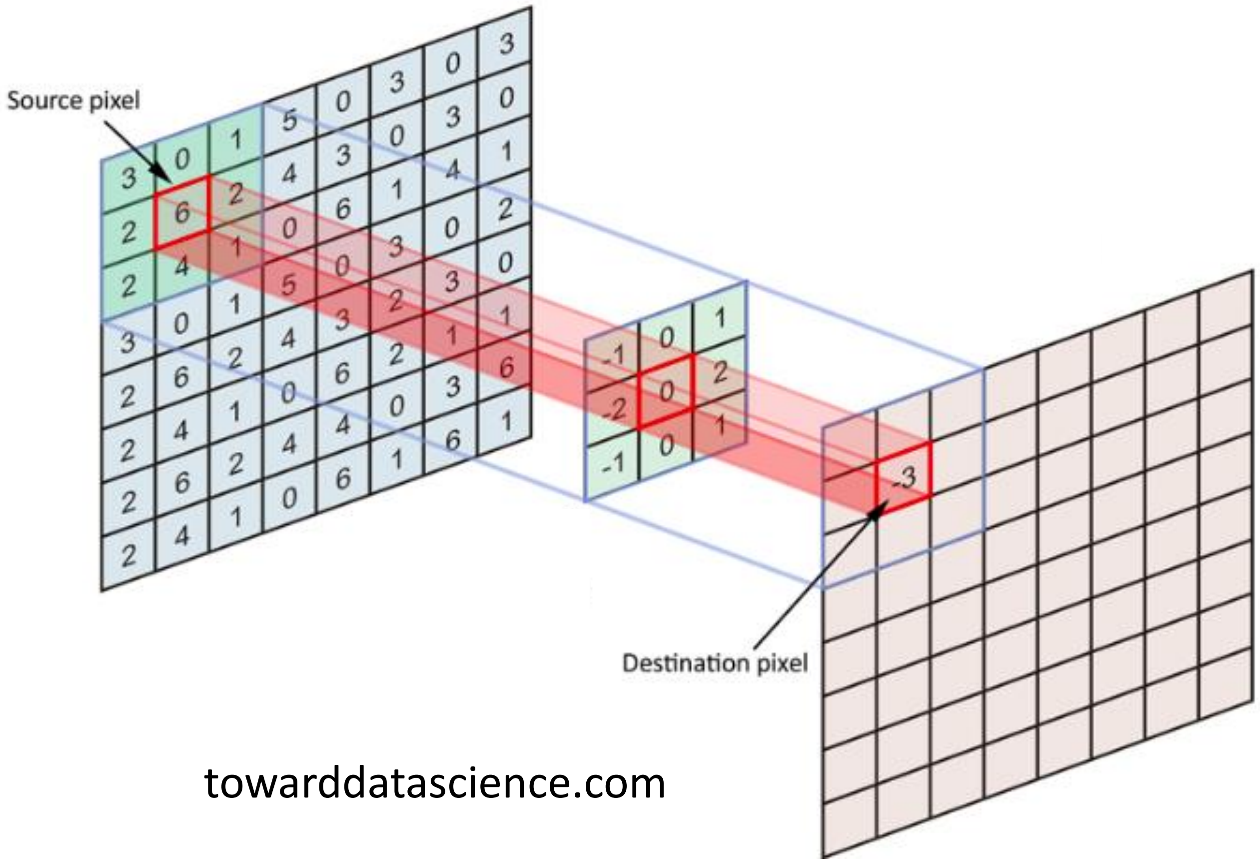
  - Computing the gradient
    - Chain rule: $\frac{dy}{dx} = \frac{dy}{h_2} \times \frac{dh_2}{dh_1} \times \frac{dh_1}{dx}$

Matrices

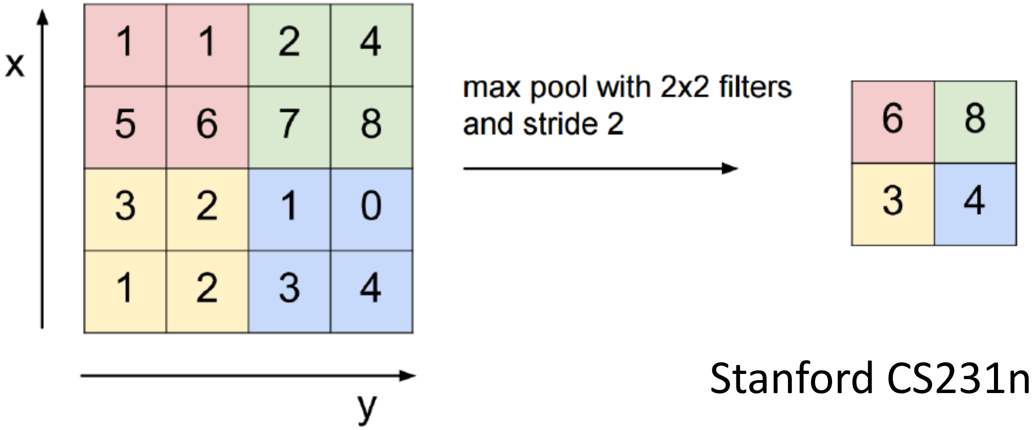
    - Evaluation of $\frac{dy}{dx}$: "**Back propagation**"

Common Operations

- Fully connected (dot product)
- Convolution
  - Translationally invariant
  - Controls overfitting
- Pooling (fixed function)
  - Down-sampling
  - Controls overfitting
- Nonlinearity layer (fixed function)
  - Activation functions, e.g. ReLU

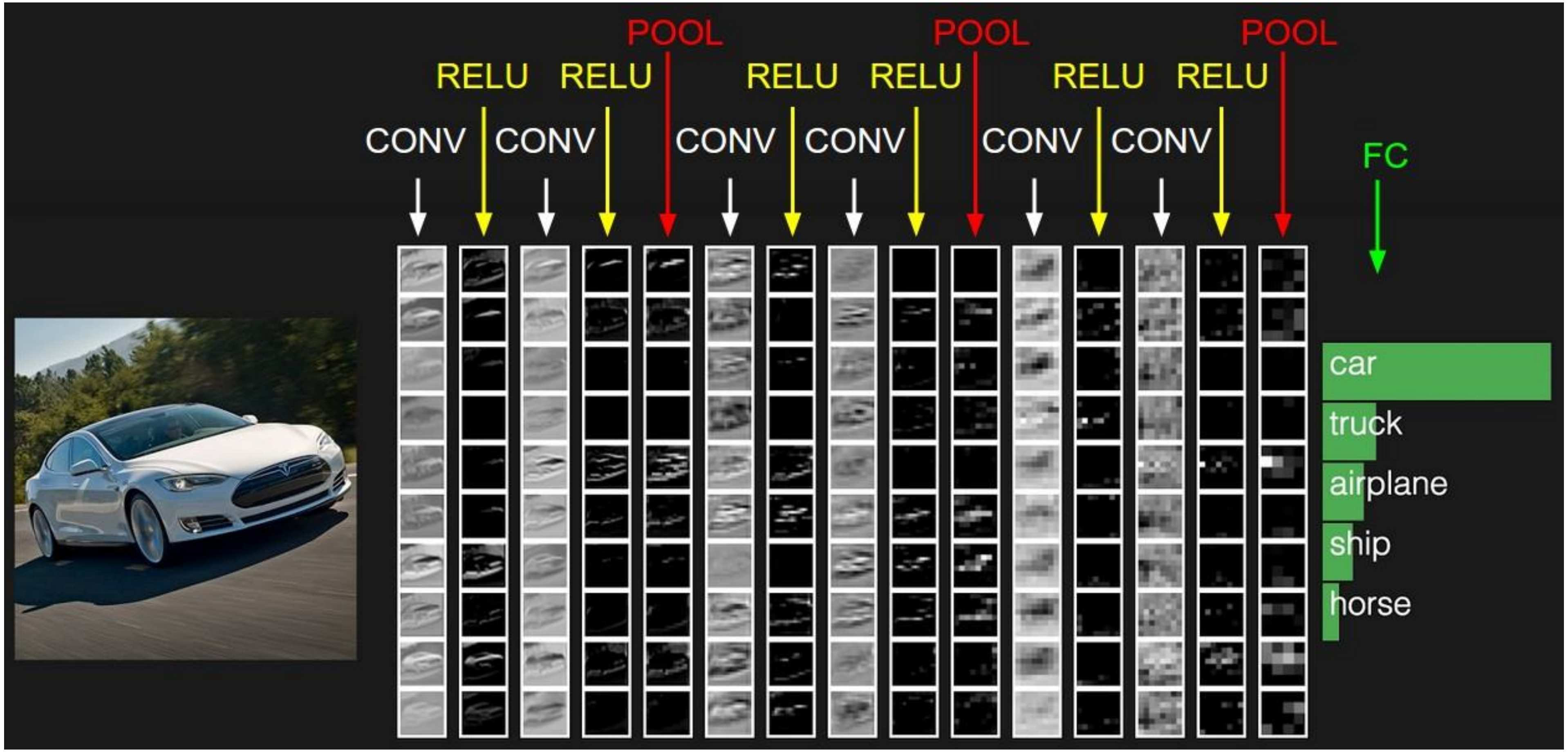


towarddatascience.com



Stanford CS231n

Example: Small VGG Net From Stanford CS231n



Neural Network Architectures

- Convolutional neural network (CNN)
  - Has translational invariance properties from convolution
  - Common used for computer vision
- Recurrent neural network RNN
  - Has feedback loops to capture temporal or sequential information
  - Useful for handwriting recognition, speech recognition, reinforcement learning
  - Long short-term memory (LSTM): special type of RNN with advantages in numerical properties
- Others
  - General feedforward networks, variational autoencoders (VAEs), conditional VAEs,

Training Neural Networks

- Training process (optimization algorithm)
  - Standard L1 and L2 regularization
  - Dropout: randomly set neurons to zero in each training iteration
  - Transform input data (e.g. rotating, stretching, adding noise)
  - Learning rate (step size) and other hyperparameter tuning

- Software packages: Efficient gradient computation
  - Caffe, Torch, Theano, Tensor Flow