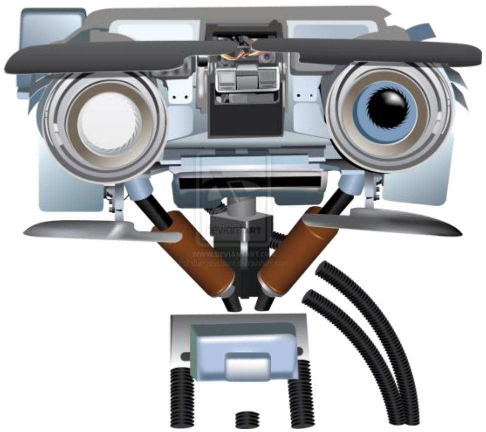


Supervised Learning I: Regression

Today

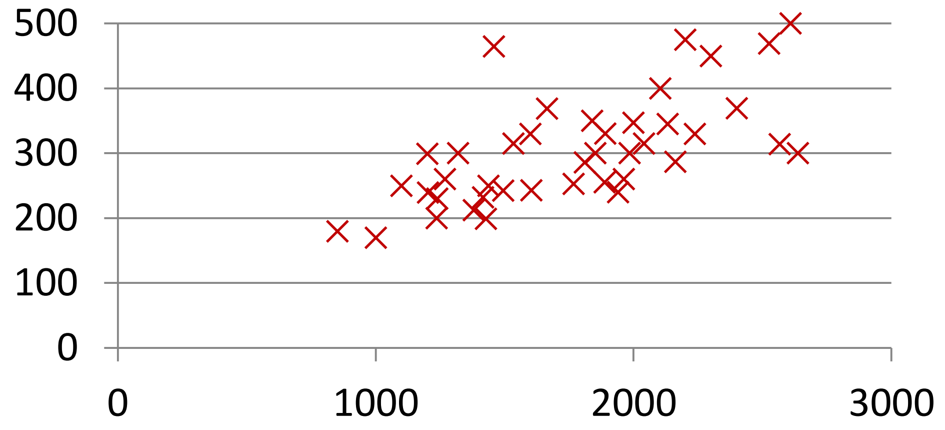
- Multivariate linear regression
- Solution for SSD cost
  - Indirect
  - Direct
- Maximum likelihood cost

Linear Regression

Hypothesis:

$$h_\theta(x) = \theta_0 + \theta_1 x$$

θ_i's: Parameters



Cost Function:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)^2$$

Multidimensional inputs

Size (feet^2)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
…	…	…	…	…

Notation:

n = number of features

$x^{(i)}$ = input (features) of i^{th} training example.

$x_j^{(i)}$ = value of feature j in i^{th} training example.

Multivariate Linear Regression

Hypothesis:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

For convenience of notation, define $x_0 = 1$.

θ_i 's: Parameters

Cost Function:

$$J(\theta_0, \theta_1, \ldots, \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Goal: $\underset{\theta_0, \theta_1, \ldots, \theta_n}{\text{minimize}} \; J(\theta_0, \theta_1, \ldots, \theta_n)$ **How??**

Two potential solutions

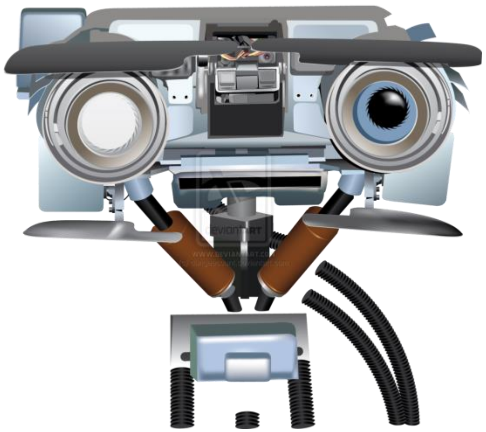
$$\min_{\theta} J(\theta; x^{(1)}, y^{(1)}, \dots, x^{(m)}, y^{(m)})$$

Gradient descent (or other iterative algorithm)

- Start with a guess for θ
- Change θ to decrease $J(\theta)$
- Until reach minimum

Direct minimization

- Take derivative, set to zero
- Sufficient condition for minima
- Not possible for most "interesting" cost functions



Solving Linear Regression

Gradient Descent

Gradient Descent Algorithm

Set $\theta = 0$

Repeat {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

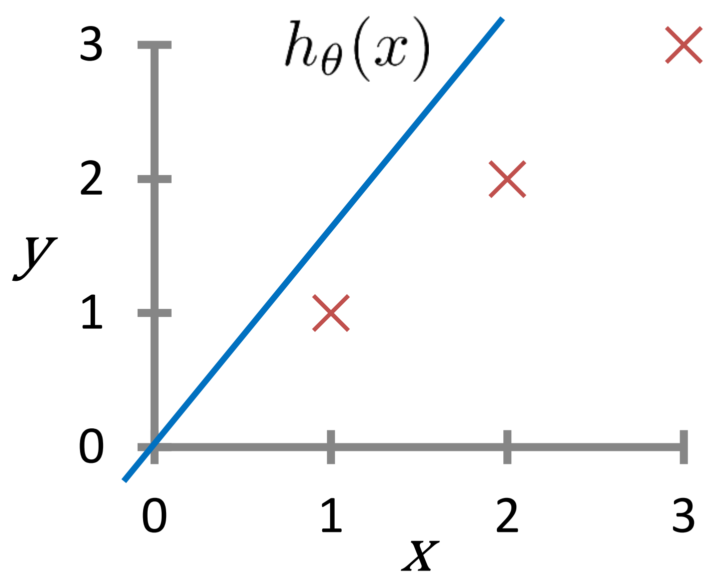
simultaneously for all
$j = 0, \ldots, n$

} until convergence

Gradient Descent: Intuition

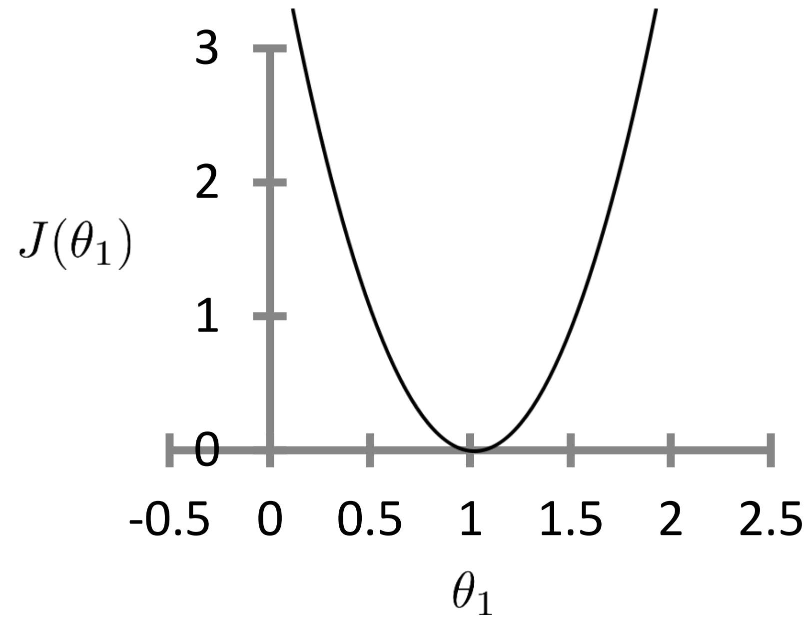
$h_\theta(x)$

(for fixed θ_1, this is a function of x)

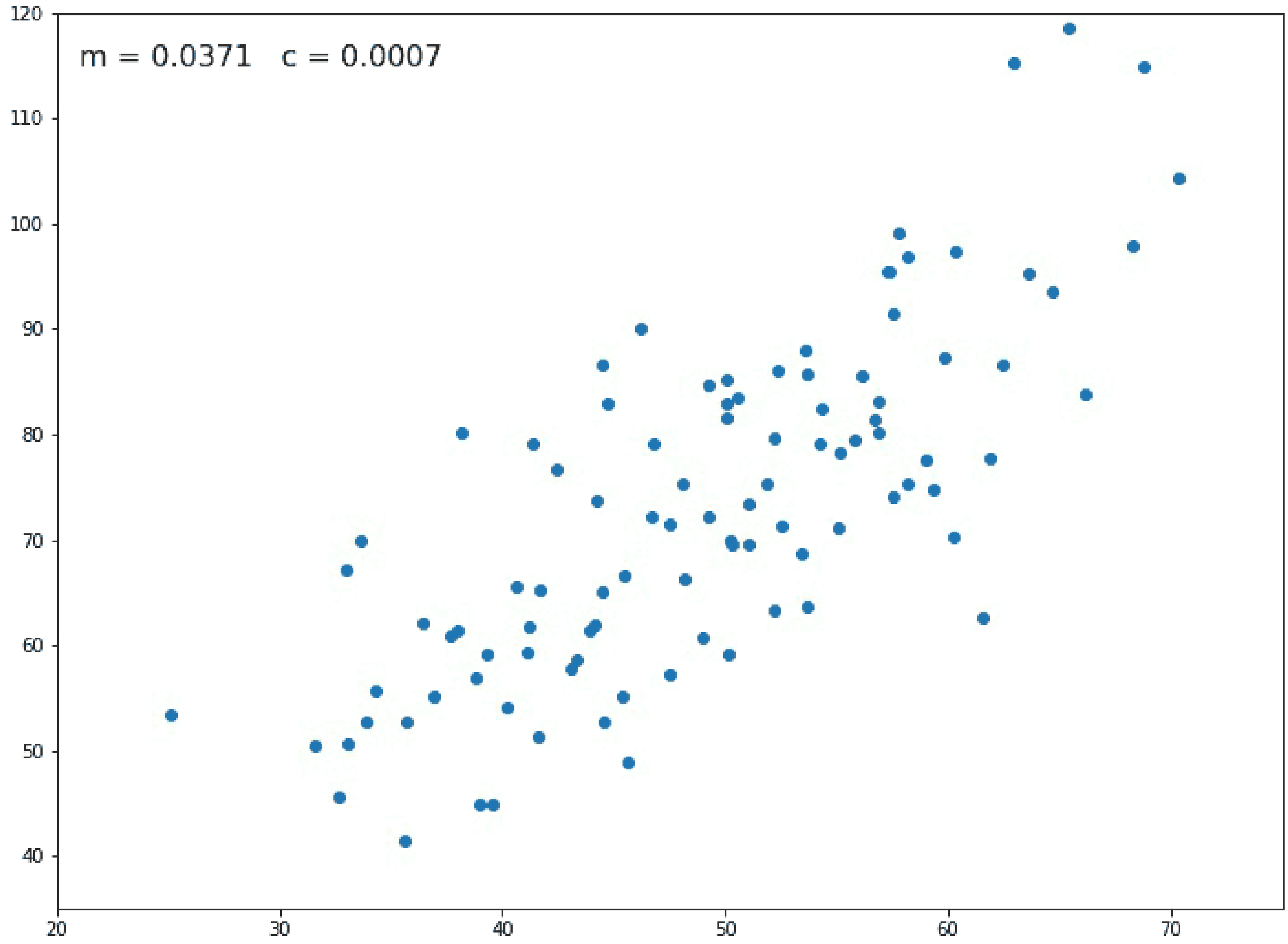


$J(\theta_1)$

(function of the parameter θ_1)



$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$



Gradient descent illustration (credit: https://towardsdatascience.com/

Gradient for Least Squares Cost

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

$$\frac{\partial}{\partial \theta_j} J(\theta) \quad =$$

Gradient for Least Squares Cost

For one example

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{\partial}{\partial \theta_j} \frac{1}{2} \left(h_\theta(x) - y\right)^2$$

Gradient for Least Squares Cost

For one example

$$
\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2} \left(h_\theta(x) - y \right)^2 \\
&= 2 \cdot \frac{1}{2} \left(h_\theta(x) - y \right) \cdot \frac{\partial}{\partial \theta_j} \left(h_\theta(x) - y \right)
\end{aligned}
$$

Gradient for Least Squares Cost

For one example

$$
\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2} \left(h_\theta(x) - y\right)^2 \\
&= 2 \cdot \frac{1}{2} \left(h_\theta(x) - y\right) \cdot \frac{\partial}{\partial \theta_j} \left(h_\theta(x) - y\right) \\
&= \left(h_\theta(x) - y\right) \cdot \frac{\partial}{\partial \theta_j} \left(\sum_{i=0}^{n} \theta_i x_i - y \right)
\end{aligned}
$$

Gradient for Least Squares Cost

For one example

$$
\begin{aligned}
\frac{\partial}{\partial\theta_j}J(\theta) &= \frac{\partial}{\partial\theta_j}\frac{1}{2}\left(h_\theta(x)-y\right)^2 \\
&= 2\cdot\frac{1}{2}\left(h_\theta(x)-y\right)\cdot\frac{\partial}{\partial\theta_j}\left(h_\theta(x)-y\right) \\
&= \left(h_\theta(x)-y\right)\cdot\frac{\partial}{\partial\theta_j}\left(\sum_{i=0}^{n}\theta_i x_i - y\right) \\
&= \left(h_\theta(x)-y\right)x_j
\end{aligned}
$$

What is this?

Gradient Descent Algorithm

Set $\boldsymbol{\theta} = 0$

Repeat {

$$\boldsymbol{\theta}_j := \boldsymbol{\theta}_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(\boldsymbol{x}^{(i)}\right) - y^{(i)} \right) \boldsymbol{x}_j^{(i)}$$

simultaneously for all $j = 0, \ldots, n$

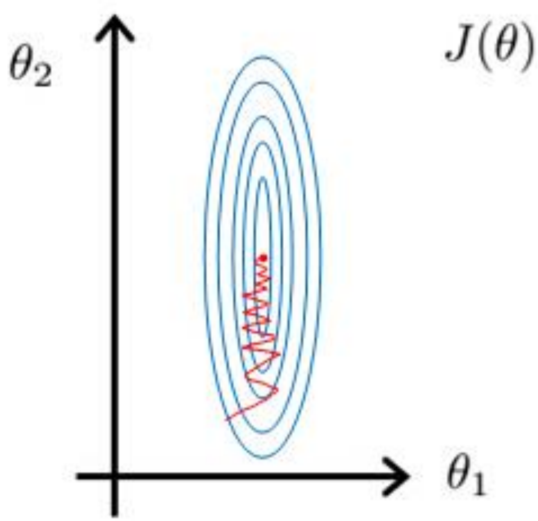
} until convergence

in vector form?

PS 1: vectorize the update and implement in Python (use matrix operations, not loops!)

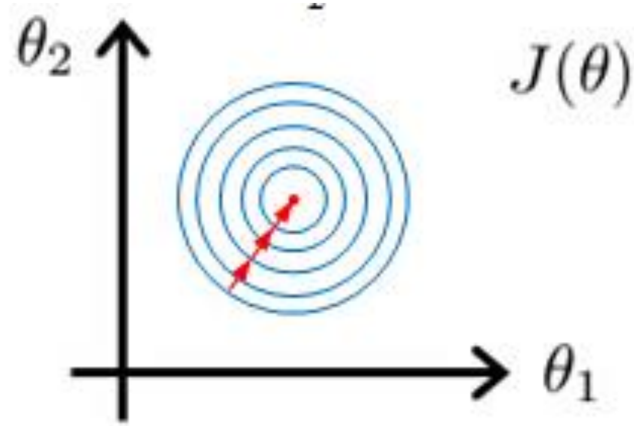
Feature normalization

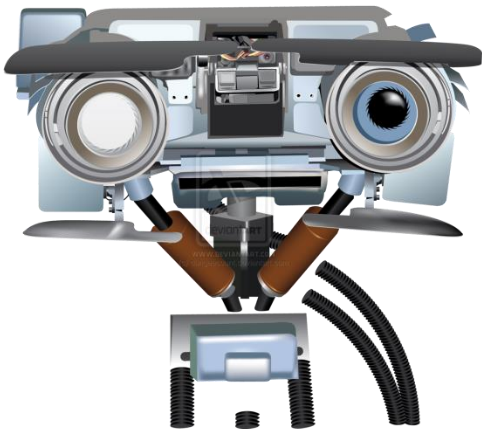
- If features have very different scale, GD can get "stuck" since x_j affects size of gradient in the direction of j[th] dimension
- Normalizing features to be zero-mean (μ) and same-variance (σ) helps gradient descent converge faster



normalize

$$x_j^{(i)} = \frac{x_j^{(i)} - \mu_j}{\sigma_j}$$





Solving Linear Regression

Direct Solution

Direct solution

Want to minimize SSD:

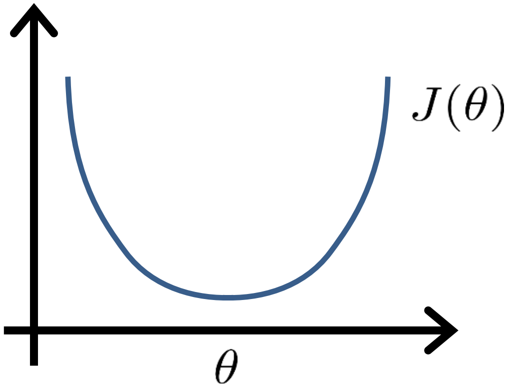
$$J(\theta_0, \theta_1, \ldots, \theta_m) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

Find minima of function:

$\theta \in \mathbb{R}^{n+1}$

$\frac{\partial}{\partial \theta_j} J(\theta) = \cdots = 0$ (for every j)

Solve for $\theta_0, \theta_1, \ldots, \theta_n$



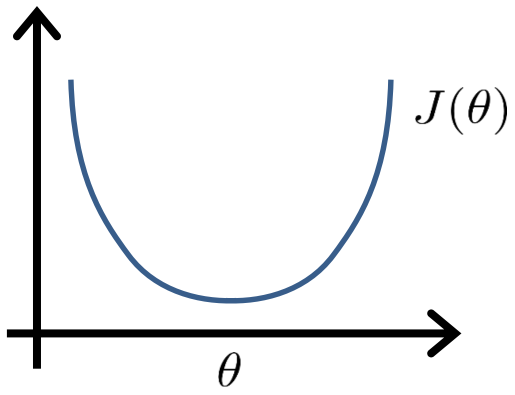
Direct solution

Re-write SSD using vector-matrix notation:

$$J(\theta) = \frac{1}{2m}(X\theta - y)^T(X\theta - y)$$

Where:

$$X = \begin{bmatrix} — (x^{(1)})^T — \\ — (x^{(2)})^T — \\ \vdots \\ — (x^{(m)})^T — \end{bmatrix} \quad \vec{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$



Solution: Normal Equation

$$\theta = (X^TX)^{-1}X^Ty$$

Derivation of Normal Equations

- SSE in matrix form:

$$\mathrm{J}(\theta) = \frac{1}{2m}(X\theta - y)^T(X\theta - y) =$$
$$= \frac{1}{2m}\{\theta^T\{X^T X\}\,\theta - 2\{X^T y\}^T\theta + const\}$$

- Take derivative with respect to θ (vector), set to 0

$$\frac{\partial J}{\partial \theta} \propto X^T X\,\theta - \ X^T y = 0$$ ignore constant multiplier

$$\theta \quad = (X^T X)^{-1}\ X^T y$$

- Also known as the least mean squares, or least squares solution

Example: $m = 4.$

x_0	Size (feet²) x_1	Number of bedrooms x_2	Number of floors x_3	Age of home (years) x_4	Price ($1000) y
1	2104	5	1	45	460
1	1416	3	2	40	232
1	1534	3	2	30	315
1	852	2	1	36	178

Design Matrix

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix}$$

$$y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}$$

Normal Equation

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

Trade-offs

m training examples, *n* features.

Gradient Descent

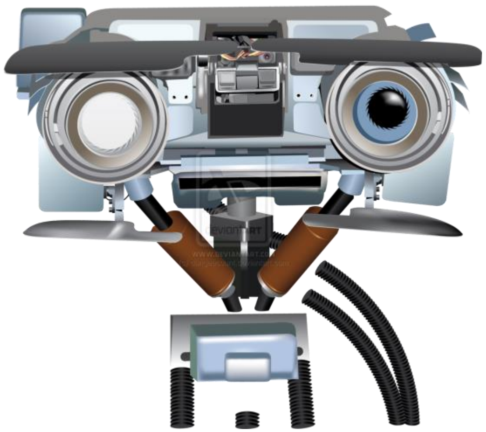
- Need to choose α.
- Needs many iterations.

- Works well even when *n* is large.

Normal Equations

- No need to choose α.
- Don't need to iterate.

- Need to compute $(X^T X)^{-1}$
- Slow if *n* is very large.



Maximum Likelihood for Linear Regression

So far, we have treated outputs as noiseless

- Defined cost function as “distance to true output”
- An alternate view:
  - data (x,y) are generated by unknown process
  - however, we only observe a noisy version
  - how can we model this uncertainty?

- Alternative cost function?

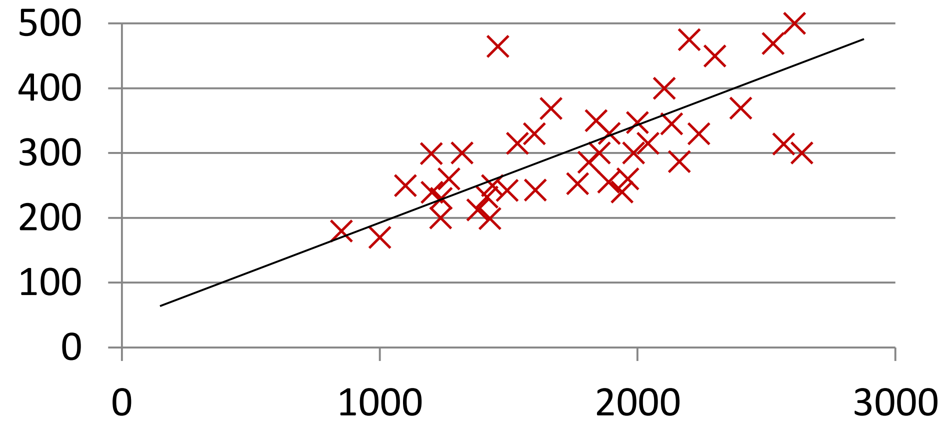
How to model uncertainty in data?

Hypothesis:

$$h_\theta(x) = \theta^T x$$

θ: parameters

$D = \left(x^{(i)}, y^{(i)}\right)$: data

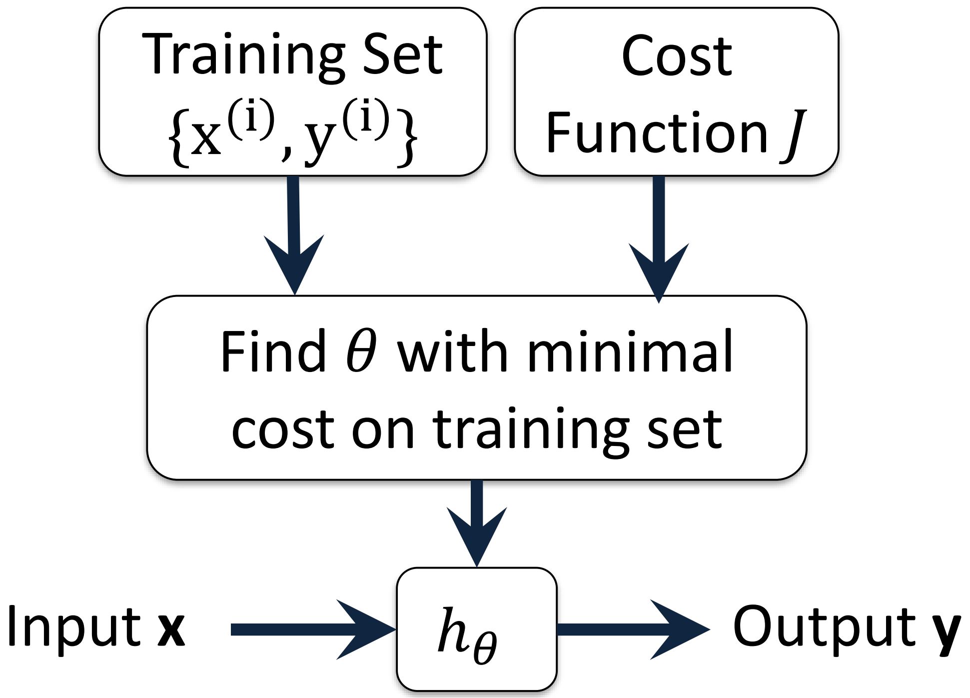


New cost function:

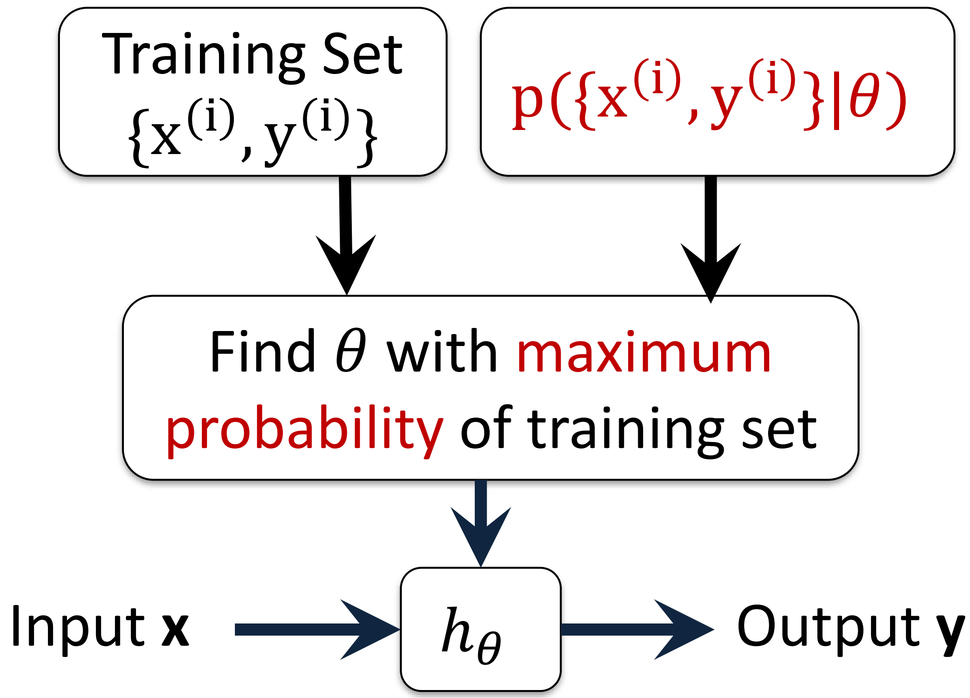
maximize probability of data given model:

$$p((x^{(i)}, y^{(i)})|\theta)$$

Recall: Cost Function



Alternative View:
“Maximum Likelihood”



Maximum Likelihood: Example

- Intuitive example: Estimate a coin toss

  I have seen 3 flips of heads, 2 flips of tails, what is the chance of head (or tail) of my next flip?

- Model:

  Each flip is a Bernoulli random variable X

  X can take only two values: 1 (head), 0 (tail)

  $$p(X = 1) = \theta, \quad p(X = 0) = 1 - \theta$$

- θ is a parameter to be identified from data

Maximum Likelihood: Example

- 5 (independent) trials

$X_1 = 1$ $X_2 = 0$ $X_3 = 1$ $X_4 = 1$ $X_5 = 0$

- Likelihood of all 5 observations:

$$p(X_1, \dots, X_5 | \theta) = \theta^3 (1 - \theta)^2$$

- Intuition

ML chooses θ such that likelihood is maximized

Maximum Likelihood: Example

- 5 (independent) trials

$X_1 = 1$ $X_2 = 0$ $X_3 = 1$ $X_4 = 1$ $X_5 = 0$

- Likelihood of all 5 observations:

$$p(X_1, \dots, X_5 | \theta) = \theta^3 (1 - \theta)^2$$

- Solution (left as exercise)

$$\theta_{ML} = \frac{3}{(3 + 2)}$$

i.e. fraction of heads in total number of trials