registermate

small pony

Linear Regression

Dr Jamie A Ward

Lecture 5

Linear Regression Refresher Quiz

1. Given the hypothesis function $h_\theta(x_i) = \theta_0 + \theta_1 x_i$, where $\theta = [\theta_0, \theta_1]^T = [0,1]^T$, and data samples $(\mathbf{x}, \mathbf{y}) = \{(1,1), (2,2), (3,3)\}$, what is the value of the loss function $J(\theta)$?

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h\left(x^{(i)};\theta\right) - y^{(i)}\right)^2$$

2. Plot the function $J(\theta)$ w.r.t. (with respect to) parameter θ.

3. How would you apply Gradient Descent to $J(\theta)$?
    - clue: GD uses the derivative of the function being optimised (in this case, J)

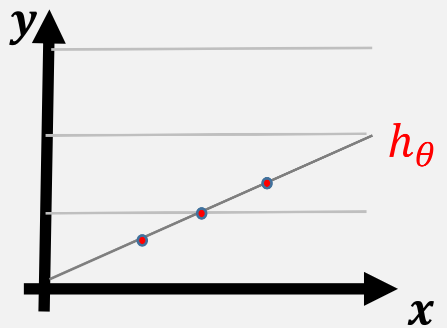
*Answers to these in last week's slides.

Lecture 5: Linear Regression

Multivariate Linear Regression

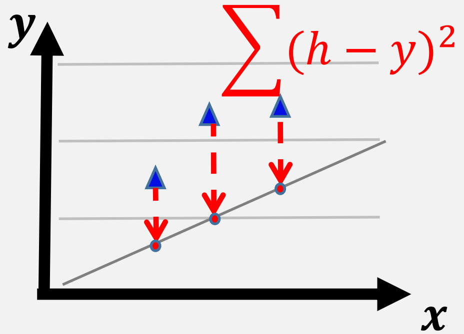
- ▸ N-dimensional regression
- ▸ Feature scaling
- ▸ The Normal Equation
- ▸ Polynomial regression
- ▸ A bit about Noise

Summary



Linear Regression Hypothesis

$$h_\theta\left(x^{(i)}\right) = \theta_0 + \theta_1 x^{(i)}$$



Linear Regression Loss

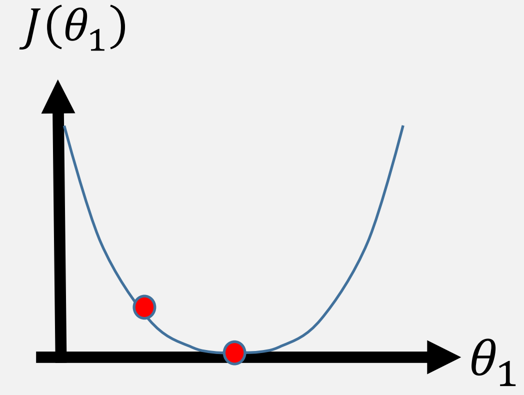
$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h\left(x^{(i)};\theta\right) - y^{(i)}\right)^2$$

Gradient descent algorithm

while not converged:

$$\theta_j^{new} = \theta_j^{old} - \alpha\frac{\partial}{\partial\theta_j}J\left(\theta_0^{old},\theta_1^{old}\right)$$

for $j = 0,1$

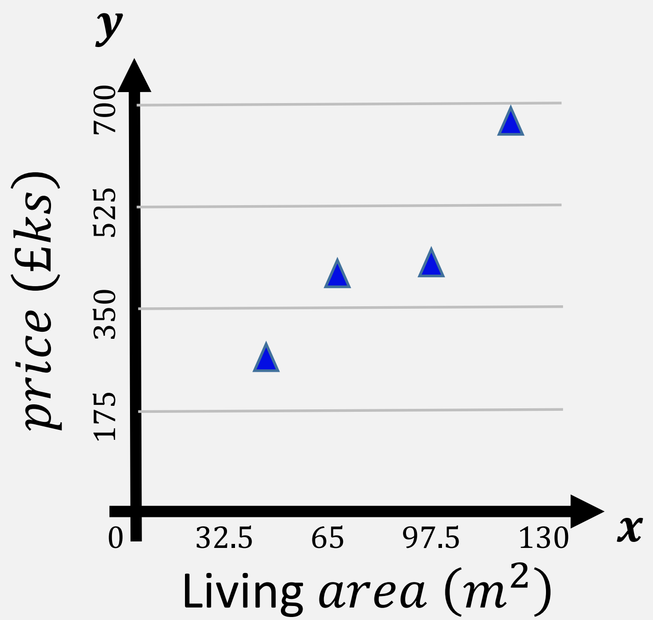


Univariate Linear Regression

Find relationships between a dependent variable (y) and independent variable (x).

$$h_\theta\left(x^{(i)}\right) = \theta_0 + \theta_1 x^{(i)}$$

Area (m^2)	Price (£ks)
100	400
72	389
50	250
122	689
x	y



Multivariate Linear Regression

But if we have several independent, or explanatory, variables, e.g.

- $x_2^{(i)}$: number of rooms
- $x_3^{(i)}$: number of floors
- $x_4^{(i)}$: age of house

Area (m^2)	# rooms	# floors	Age (years)	Price (£ks)
100	3	2	50	400
72	2	1	25	389
50	1	1	10	250
122	4	2	92	689
300	5	3	65	900
x_1	x_2	x_3	x_4	y

How might we model this?

$$h_\theta(x^{(i)}) = \theta_0 + \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)} + \theta_3 x_3^{(i)} + \theta_4 x_4^{(i)}$$

$$= \theta_0 + \sum_{j=1}^{n} \theta_j x_j^{(i)}$$

$$\boxed{h_\theta(x^{(i)}) = \sum_{j=0}^{n} \theta_j x_j^{(i)} \qquad \text{(where } x_0 = 1 \text{ and } n = 4)}$$

Multivariate Linear Regression

$$h_\theta\left(x^{(i)}\right) = \sum_{j=0}^{n} \theta_j x_j^{(i)}$$

	Area (m^2)	# rooms	# floors	Age (years)	Price (£ks)
(1)	100	3	2	50	400
(2)	72	2	1	25	389
(3)	50	1	1	10	250
(4)	122	4	2	92	689
(5)	300	5	3	65	900
	$x_1^{(i)}$	$x_2^{(i)}$	$x_3^{(i)}$	$x_4^{(i)}$	$y^{(i)}$

Notation refresher

m = number of samples

n = number of features

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

$x^{(3)}$ = feature vector at i^{th} training example

e.g. For this dataset:

$m =$

$n =$

$x_4^{(3)} =$

$x^{(3)} =$

Multivariate Linear Regression

$$h_\theta\left(x^{(i)}\right) = \sum_{j=0}^{n} \theta_j x_j^{(i)}$$

	Area (m^2)	# rooms	# floors	Age (years)	Price (£ks)
(1)	100	3	2	50	400
(2)	72	2	1	25	389
(3)	50	1	1	10	250
(4)	122	4	2	92	689
(5)	300	5	3	65	900
	$x_1^{(i)}$	$x_2^{(i)}$	$x_3^{(i)}$	$x_4^{(i)}$	$y^{(i)}$

Notation refresher

$m =$ number of samples

$n =$ number of features

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

$x^{(3)} =$ feature vector at i^{th} training example

e.g. For this dataset:

$m = 5$

$n = 4$

$x_4^{(3)} = 10$

$$x^{(3)} = \begin{bmatrix} 50 \\ 1 \\ 1 \\ 10 \end{bmatrix}$$

Multivariate Linear Regression

$$h_\theta\left(x^{(i)}\right) = \sum_{j=0}^{n} \theta_j x_j^{(i)} \quad = \theta_0 \cancelto{1}{x_0^{(i)}} + \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)} + \theta_3 x_3^{(i)} + \theta_4 x_4^{(i)}$$

(where $x_0^{(1)} = 1$ and $n = 4$)

What does this look like in vector / matrix form?

(Try with $m = 1$ for simplicity)

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \in \mathbb{R}^{n+1} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \end{bmatrix} \in \mathbb{R}^{n+1}$$

(where $x_0 = 1$)

$n = 4$

$$h_\theta(x) = \sum_{j=0}^{4} \theta_j x_j = \begin{bmatrix} \theta_0 & \theta_1 & \theta_2 & \theta_3 & \theta_4 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \theta^T x$$

Multivariate Gradient Descent

1. Multivariate Linear Regression Hypothesis

$$h_\theta\left(x^{(i)}\right) = \sum_{j=0}^{n} \theta_j x_j^{(i)} \qquad \text{(where } x_0^{(i)} = 1\text{)}$$

2. Linear Regression Loss (mean squared error / least squares)

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}\left(h\left(x^{(i)};\theta\right) - y^{(i)}\right)^2$$

3. Using 1 & 2, we get the following gradient descent updates

while not converged:

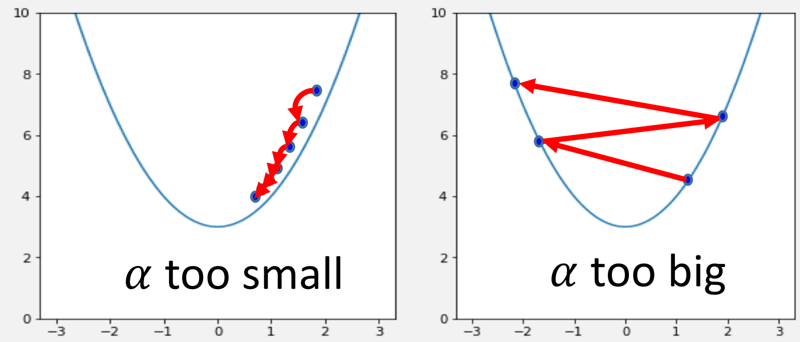
$$\theta_j^{new} = \theta_j^{old} - \alpha\frac{1}{m}\sum_{i=1}^{m}\left(h\left(x^{(i)};\theta^{old}\right) - y^{(i)}\right) x_j^{(i)}$$

for $j = 0,1,\dots,n$

(where $x_0 = 1$)

Gradient Descent

Learning rate, $\alpha > 0$, controls step size

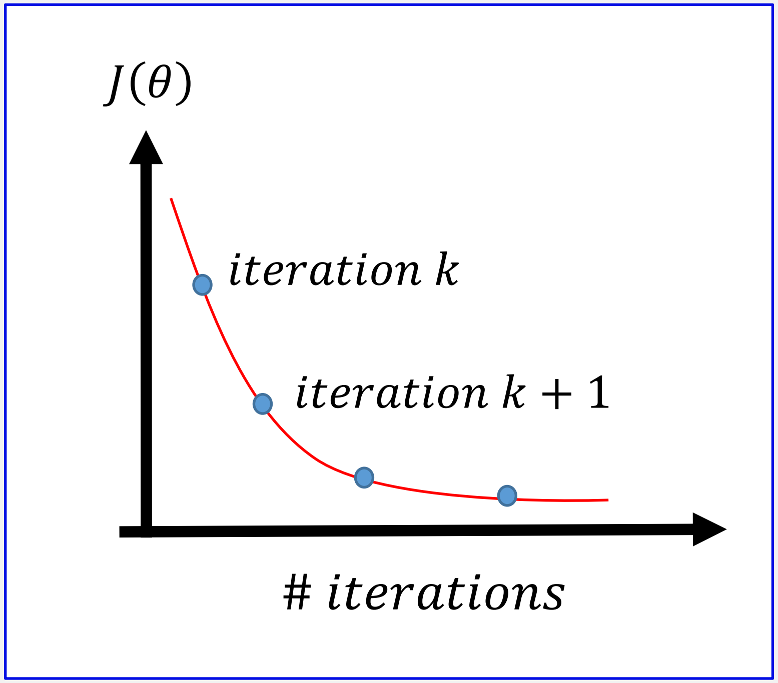


A note on Convergence

- What is an ideal value of α?
- usually $0.01 < \alpha < 10$

When has the algorithm converged?

1. When loss $J(\theta)$ decreases by less than some tolerance ε, e.g. $\varepsilon = 10^{-3}$
   (it helps to make a plot)
2. **Or** when the parameters θ stop changing.



Feature scaling

Multiple explanatory variables

- All on different scales, e.g.
  - area range(x_1) = (0-1000 m^2)
  - # rooms range(x_2) = (1-5)
- Scale the features
  - Avoids unnecessarily large or small θ
  - Gradient descent converges faster

Area (m^2)	# rooms	# floors	Age (years)	Price (£ks)
100	3	2	50	400
60	2	1	25	389
40	1	1	10	250
150	4	2	95	689
300	5	3	65	900
x_1	x_2	x_3	x_4	y

mean, $\bar{x}_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)}$

- Range normalization

Centre data around mean:

$$\mathrm{x}_\mathrm{j}^s = \frac{x_j - \bar{x}_j}{max(x_j) - \min(x_j)}$$

Min-max normalization (0 and 1):

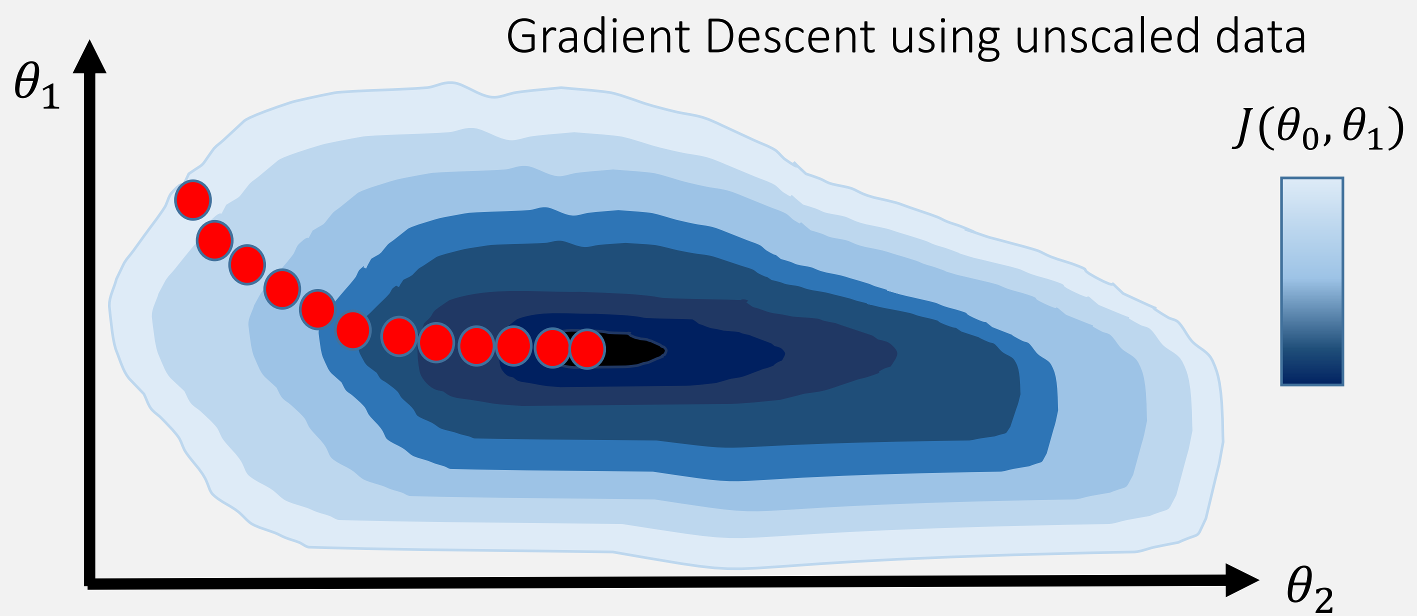
$$\mathrm{x}_\mathrm{j}^s = \frac{x_j - \min(x_j)}{max(x_j) - \min(x_j)}$$

- Standardisation (z-score)

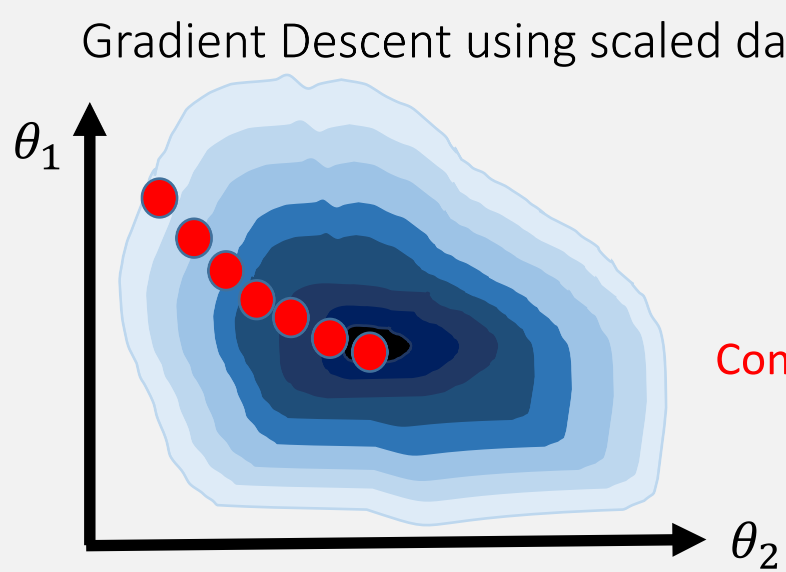
Ensure data has mean = 0 and standard deviation = 1:

$$\mathrm{x}_\mathrm{j}^s = \frac{x_j - \bar{x}_j}{std(x_j)}$$

Feature scaling: effect on regression parameters



e.g. if feature $x_1 >> x_2$
Then θ_2 will be larger
than θ_1 to compensate.



Converges faster!

Feature scaling: example

Given a possible area (x_1) range of $0 \ to \ 1000m^2$, normalise this feature to within the range $[-1, 1]$

$$\overline{x_1} = \frac{1}{5}(100 + 60 + 40 + 150 + 300)$$

$$= \frac{650}{5} = 130$$

$$\text{range}(\mathrm{x}_1) = \max(x_1) - \min(x_1) = 1000$$

$$\boldsymbol{x_1^s} = \frac{x_1 - \overline{x_1}}{range(x_1)}$$

$$= \begin{bmatrix} 100 - 130 \\ 60 - 130 \\ 40 - 130 \\ 150 - 130 \\ 300 - 130 \end{bmatrix} \Big/ 1000 = \begin{bmatrix} -0.03 \\ -0.07 \\ -0.09 \\ 0.2 \\ 0.17 \end{bmatrix}$$

Area (m^2)	# rooms	# floors	Age (years)	Price (£ks)
100	3	2	50	400
60	2	1	25	389
40	1	1	10	250
150	4	2	95	689
300	5	3	65	900
x_1	x_2	x_3	x_4	y

-0.03
-0.07
-0.09
0.2
0.17
x_1^s

Feature scaling: example

Range normalisation applied to all the features, given the following stats:

$$x_1\text{: range } [0, 1000],\ \bar{x}_1 = 130$$
$$x_2\text{: range } [1, 10],\ \bar{x}_2 = 3$$
$$x_3\text{: range } [1, 10],\ \bar{x}_3 = 2$$
$$x_4\text{: range } [0, 100],\ \bar{x}_4 = 25$$

$$\boldsymbol{x}_j^s = \frac{\boldsymbol{x}_j - \bar{x}_j}{range(x_j)}$$

$$x_1^s\text{: range } [-1, 1],\ \bar{x}^s{}_1 = 0$$
$$x_2^s\text{: range } [-1, 1],\ \bar{x}^s{}_2 = 0$$
$$x_3^s\text{: range } [-1, 1],\ \bar{x}^s{}_3 = 0$$
$$x_4^s\text{: range } [-1, 1],\ \bar{x}^s{}_4 = 0$$

Area (m^2)	# rooms	# floors	Age (years)	Price (£ks)
100	3	2	50	400
60	2	1	25	389
40	1	1	10	250
150	4	2	95	689
300	5	3	65	900
x_1	x_2	x_3	x_4	y

-0.03	0	0	0.25
-0.07	-0.1	-0.1	0
-0.09	-0.2	-0.1	-0.15
0.2	0.1	0	0.7
0.17	0.2	0.1	0.4
x_1^s	x_2^s	x_3^s	x_4^s

Alternative Solutions for Linear Regression

Iterative optimisation

- (Batch) Gradient Descent
  - Slow for lots of data, m
  - Fast for lots of features, n

while not converged: $(x_0 = 1)$

$$\theta_j^{new} = \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)} \right) x_j^{(i)}$$

for $j = 0, 1, \dots, n$

Alternative Solutions for Linear Regression

Iterative optimisation

- (Batch) Gradient Descent
  - Slow for lots of data, m
  - Fast for lots of features, n

while not converged: $(x_0 = 1)$

$$\theta_j^{new} = \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$

for $j = 0,1,\ldots,n$

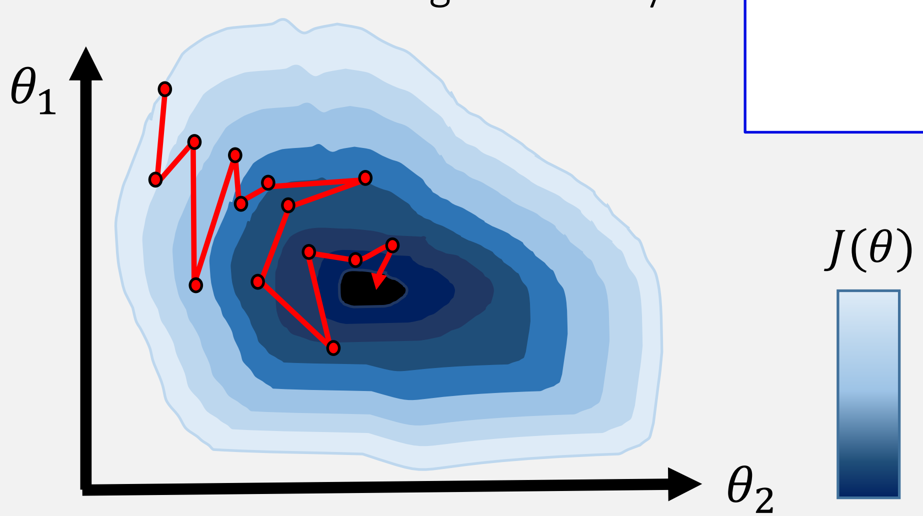
- Stochastic Gradient Descent
  - Fast for lots of data, m
  - Fast for lots of features, n
  - Doesn't converge smoothly

while not converged: $(x_0 = 1)$

for $i = random(1 \text{ to } m)$:

$$\theta_j^{new} = \theta_j - \alpha\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$

for $j = 0,1,\ldots,n$



Alternative Solutions for Linear Regression

Iterative optimisation

- (Batch) Gradient Descent
  - Slow for lots of data, m
  - Fast for lots of features, n

while not converged: $(x_0 = 1)$

$$\theta_j^{new} = \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$

for $j = 0,1, \dots, n$

- Stochastic Gradient Descent
  - Fast for lots of data, m
  - Fast for lots of features, n

while not converged: $(x_0 = 1)$

for $i = random(1 \text{ to } m)$:

$$\theta_j^{new} = \theta_j - \alpha\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$

for $j = 0,1, \dots, n$

Analytic solution

- Normal Equation

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

Normal Equation

Closed-form (analytic) solution

- Find appropriate θ where $\boldsymbol{X}\theta = \boldsymbol{y}$

  (Calculate the derivative for $J(\theta)$ wrt θ and set to zero)

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

x_0	Area x_1	# rooms x_2	# floors x_3	Age x_4	Price $\boldsymbol{y}$
1	100	3	2	50	400
1	60	2	1	25	389
1	40	1	1	10	250
1	150	4	2	95	689

$$\begin{bmatrix} 1 & 100 & 3 & 2 & 50 \\ 1 & 60 & 2 & 1 & 25 \\ 1 & 40 & 1 & 1 & 10 \\ 1 & 150 & 4 & 2 & 95 \end{bmatrix} \cdot \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \end{bmatrix} = \begin{bmatrix} 400 \\ 389 \\ 250 \\ 689 \end{bmatrix}$$

(X=Design matrix) $\boldsymbol{X} \in \mathbb{R}^{m\times n+1}$ $\quad \boldsymbol{\theta} \in \mathbb{R}^{n+1}$ $\quad \boldsymbol{y} \in \mathbb{R}^{m}$

```
import numpy as np
X=np.array([[1,100,3,2,50],[1,60,2,1,25],[1,40,1,1,10],[1,150,4,2,95]])
y=np.array([400,389,250,689])

theta = np.linalg.pinv(X.T.dot(X)).dot(X.T).dot(y)
print( X.dot(theta) )  # check that Xθ = y
```

Alternative Solutions for Linear Regression

Iterative optimisation

- (Batch) Gradient Descent
  - Slow for lots of data, m
  - Fast for lots of features, n

while not converged: $(x_0 = 1)$

$$\theta_j^{new} = \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$

for $j = 0,1, \ldots, n$

- Stochastic Gradient Descent
  - Fast for lots of data, m
  - Fast for lots of features, n

while not converged: $(x_0 = 1)$

for $i = random(1 \text{ to } m)$:

$$\theta_j^{new} = \theta_j - \alpha\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$

for $j = 0,1, \ldots, n$

Analytic solution

- Normal Equation
  - Fast for lots of data, m
  - Slow for big n (>10000)
  - Scaling not required

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

$\sim O(n^3)$

very useful equation!

(a.k.a. Ordinary Least Squares Regression)

Alternative Solutions for Linear Regression

Iterative optimisation

- ▸ (Batch) Gradient Descent
  - Slow for lots of data, m
  - Fast for lots of features, n

while not converged: $(x_0 = 1)$

$$\theta_j^{new} = \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$

for $j = 0, 1, \ldots, n$

- ▸ Stochastic Gradient Descent
  - Fast for lots of data, m
  - Fast for lots of features, n

while not converged: $(x_0 = 1)$

for $i = random(1 \text{ to } m)$:

$$\theta_j^{new} = \theta_j - \alpha\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) x_j^{(i)}$$

for $j = 0, 1, \ldots, n$

Analytic solution

- ▸ Normal Equation
  - Fast for lots of data, m
  - Slow for big n (>10000)
  - Scaling not required

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

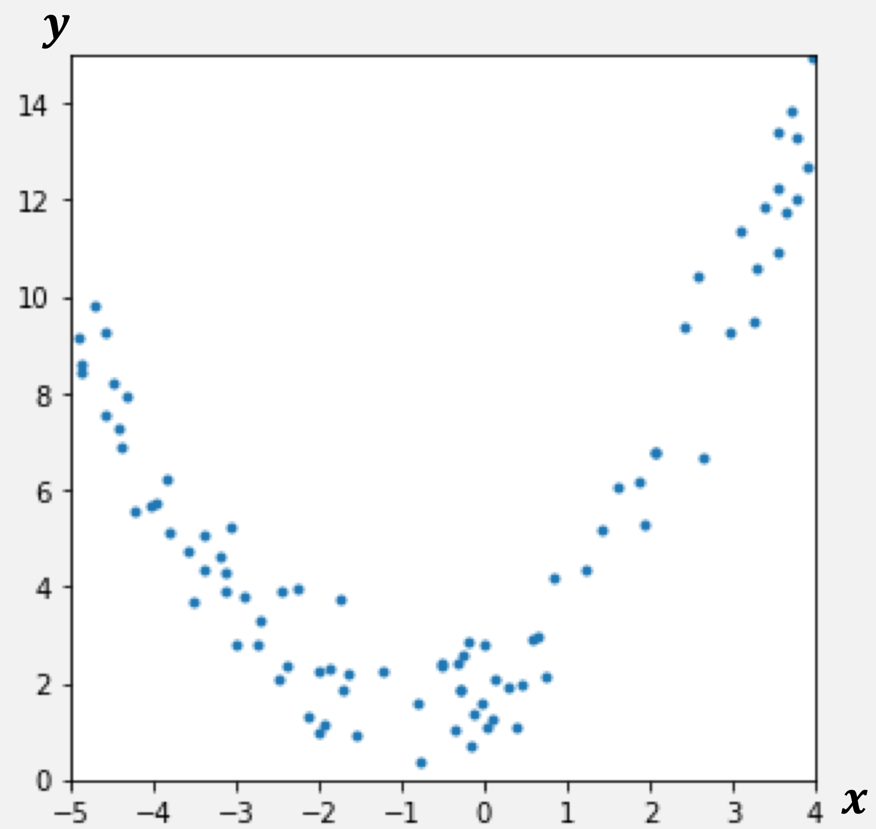
$\sim O(n^3)$

very useful equation!

Polynomial Regression

A special case of multivariate linear regression

$h_\theta(x) =$

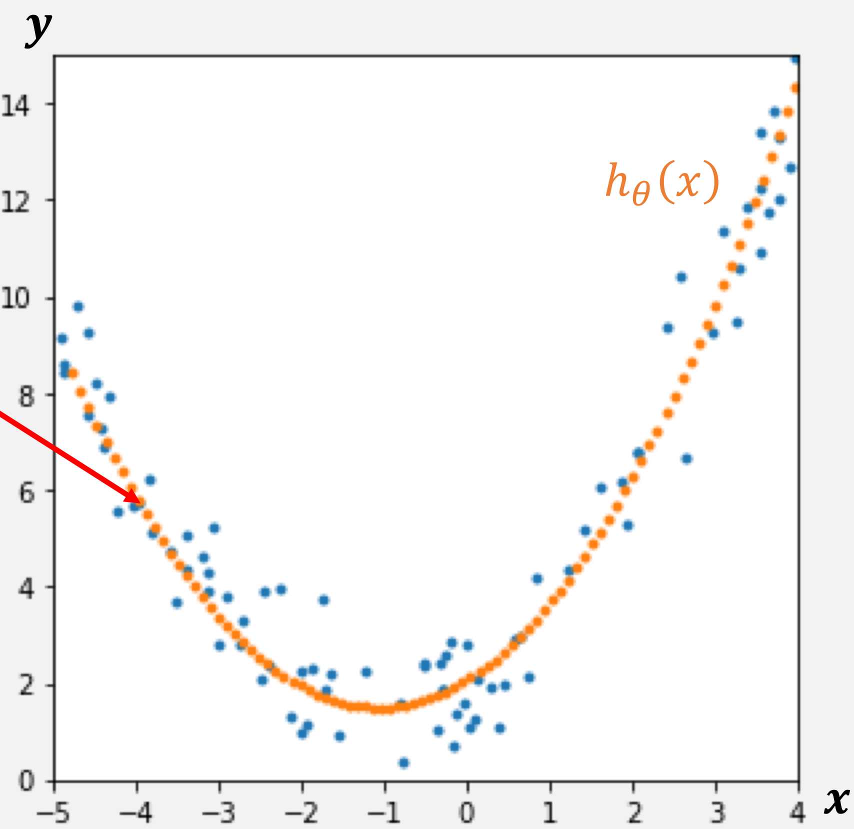


Polynomial Regression

A special case of multivariate linear regression

- Use a polynomial (non-linear) hypothesis

$$
\begin{aligned}
h_\theta(x) &= \theta_0 + \theta_1 x + \theta_2 x^2 \\
&= 2 + x + 0.5x^2
\end{aligned}
$$



This is non-linear in x...

- But it is still linear wrt θ
- Linear regression methods can still be used!

Also works for, e.g.:

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 \sqrt{x}$$

$$h_\theta(\boldsymbol{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_4 x_2^2$$

A bit about noise

So far all our hypothesis functions mapped input to output in a deterministic way, e.g.

$$h_\theta(x) = 2 + x + 0.5x^2$$

But real world data has random, non deterministic changes, or noise

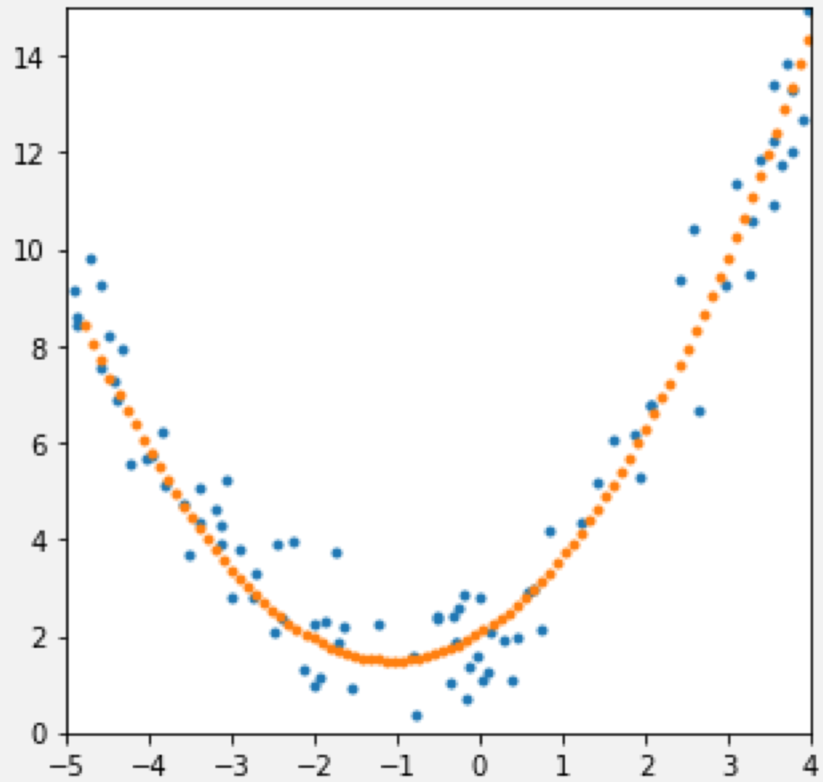
We can model this with a noise term, ϵ:

$$y(x) = h_\theta(x) + \epsilon$$

Or:

$$\epsilon = (h_\theta(x) - y(x))$$

(The residual from the Loss function)



A bit about noise

$$y = \sum_{j=0}^{n} \theta_j x_j^{(i)} + \epsilon^{(i)}$$

$$= \boldsymbol{X}\theta + \boldsymbol{\epsilon}$$

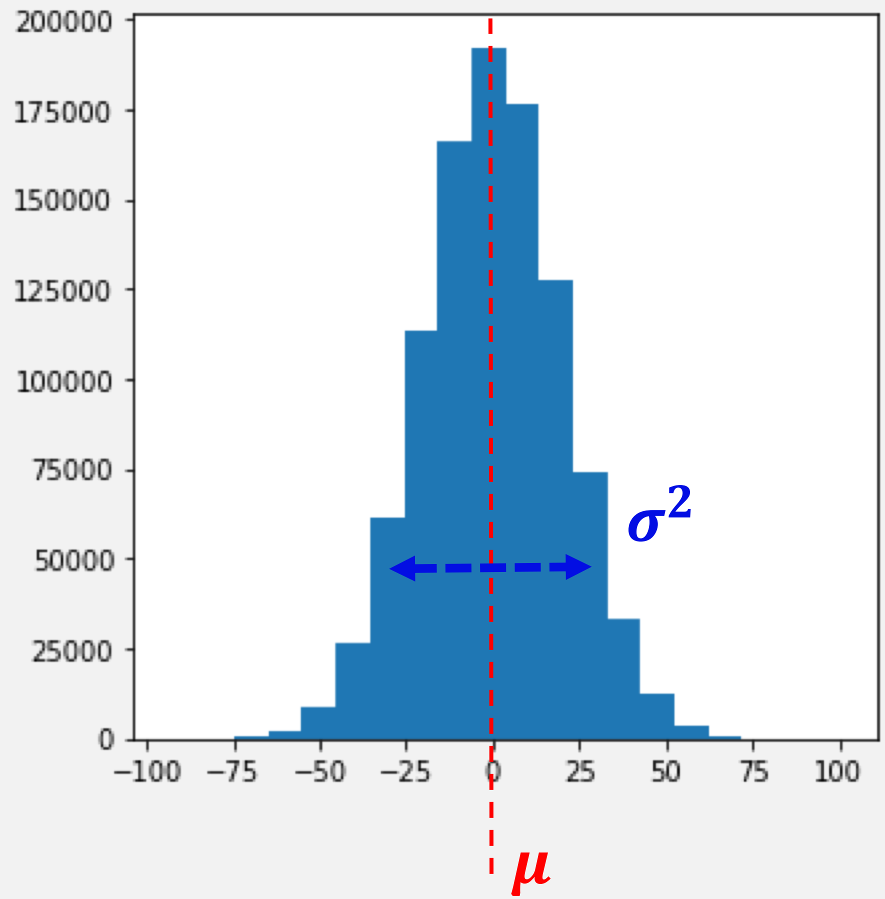
Additive noise ϵ is often modelled as a Gaussian random variable.

Gaussian, or normal distribution:

$$\epsilon = \mathcal{N}(\mu, \sigma^2)$$

where μ = mean, and
σ^2 = variance of the noise.

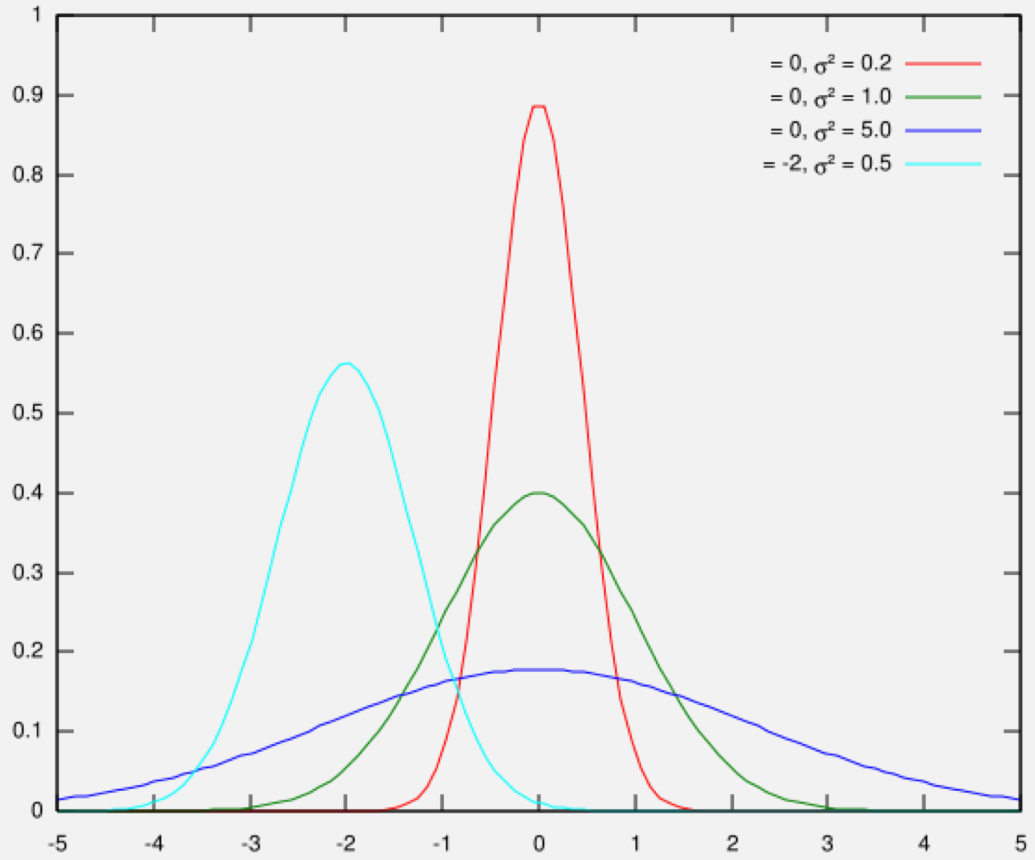
Normal distribution of 1 million samples



Normal distribution

Commonly used as a probability distribution

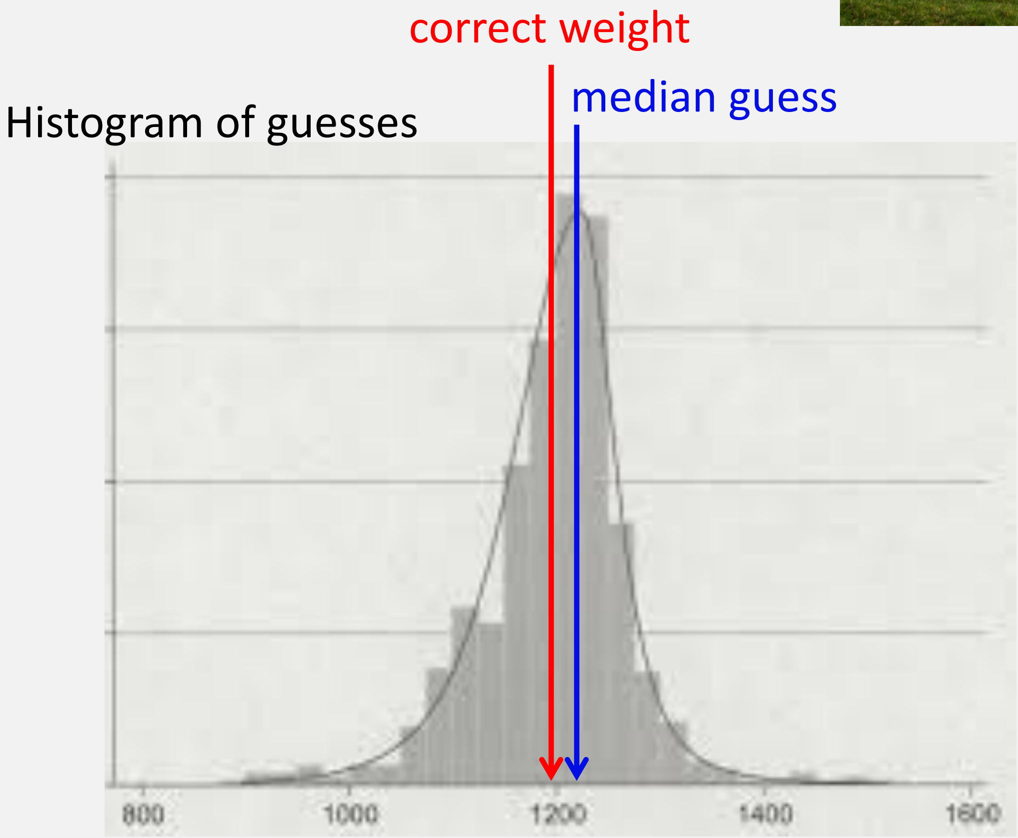
Defined by only 2 variables: mean (μ) and variance (σ^2)



How much do I weigh?

The wisdom of the crowds

In 1906 at the Plymouth Fair, Francis Galton asked a crowd of 800 people to guess the weight of an ox.



https://www.doc.gold.ac.uk/dept/Evaluation