Statistics and Data Science 365 / 565

Data Mining and Machine Learning

February 6

Yale

Outline

- Recap: supervised learning goal
- How do we assess if we've learned well?
- Training vs testing
- Linear Regression
- Quadratic Regression
- Kernel Regression
- KNN regression
- Notebook

Recap: Supervised learning goals

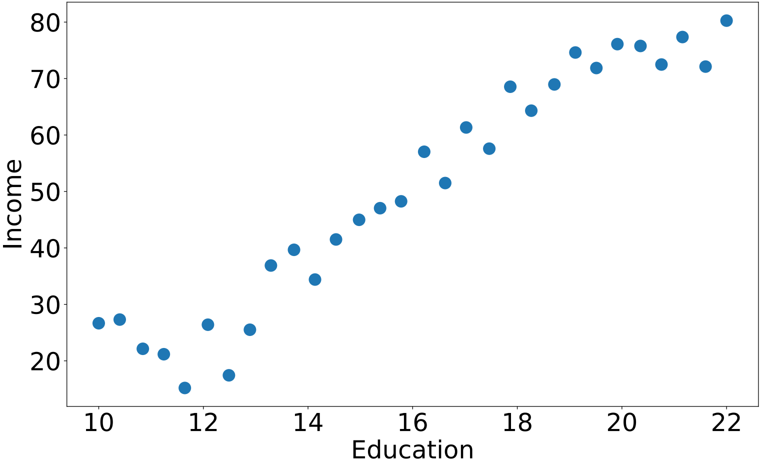
Create a function $\widehat{f} : \mathcal{X} \mapsto \mathcal{Y}$ by “learning” from data $z_i = (x_i, y_i)$.

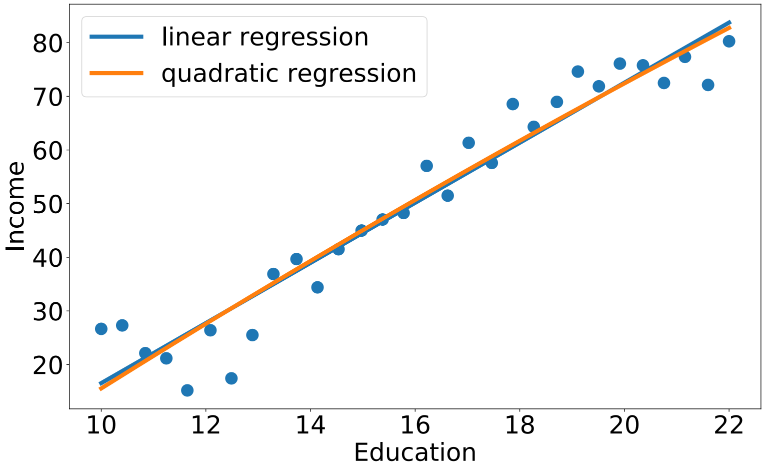
Probably want $\widehat{f}(x_i) \approx y_i$ (Notation: $\approx$ means approximately)

Regression: $\mathcal{Y}$ is continuous/ordered (prices)

Classification: $\mathcal{Y}$ is discrete/unordered (labels: hot dog v not hot dog)

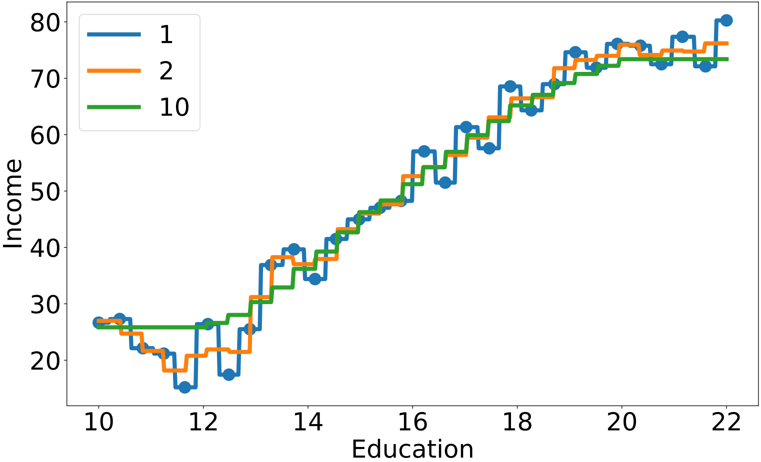
Lots of approaches

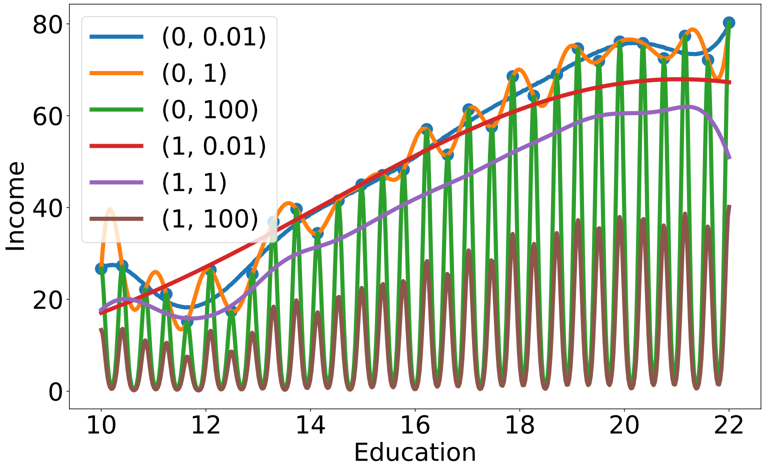

80
70
60
50
40
30
20
Income
10
12
14
16
18
20
22
Education


linear regression
quadratic regression
Income
Education
80
70
60
50
40
30
20
10
12
14
16
18
20
22

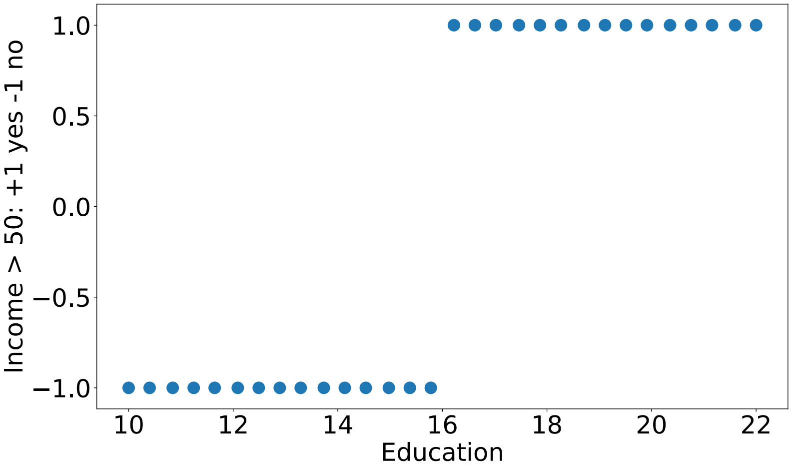
Nearest neighbor

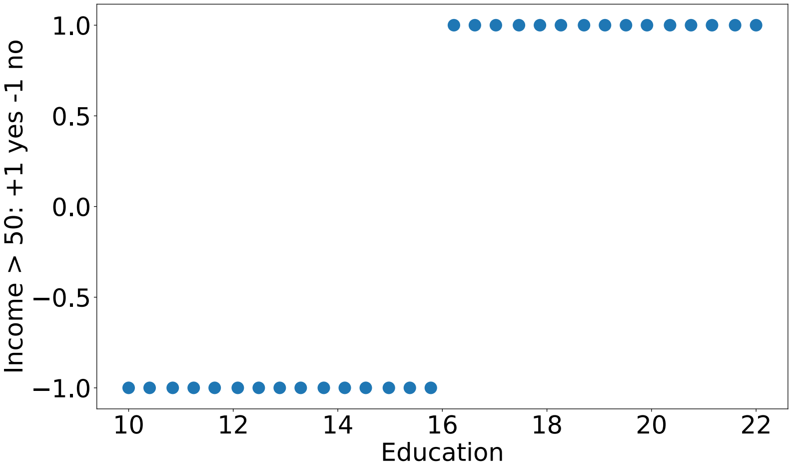
Estimate answer of unseen point by averaging neighbors



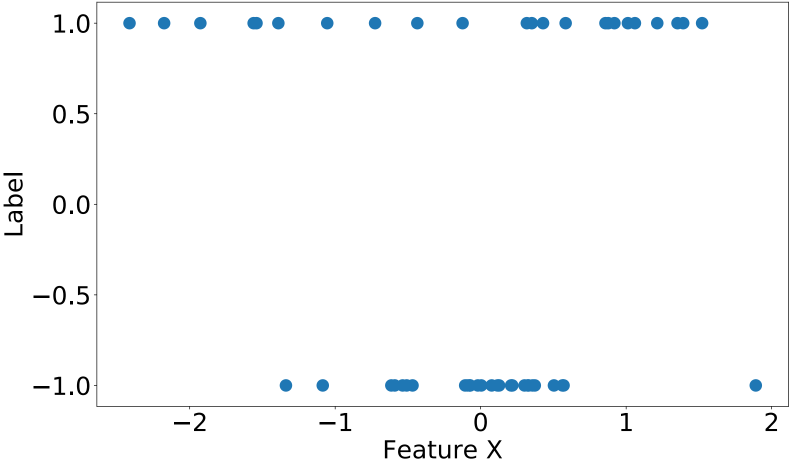


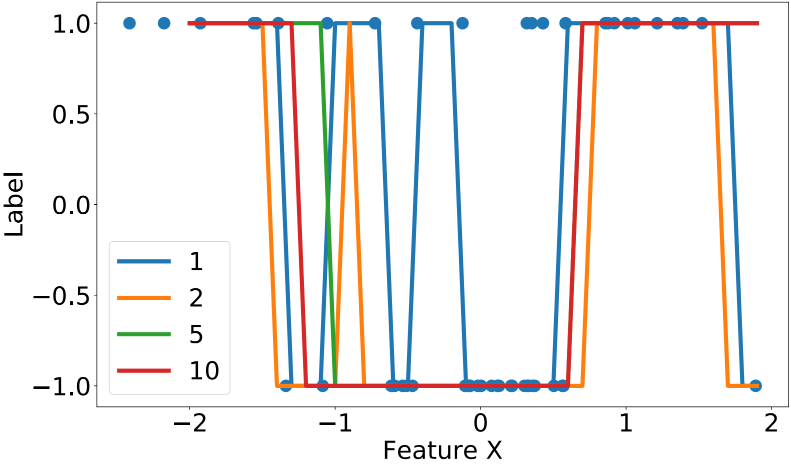
(0, 0.01)
(0, 1)
(0, 100)
(1, 0.01)
(1, 1)
(1, 100)
Income
0
20
40
60
80
Education
10
12
14
16
18
20
22

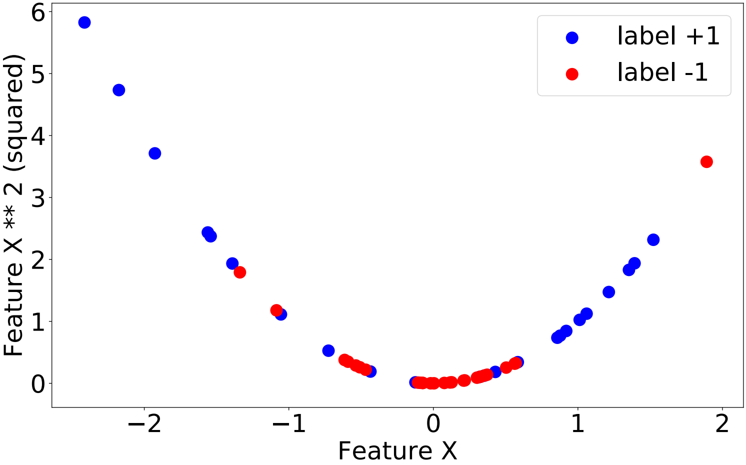

1.0
0.5
0.0
−0.5
−1.0
Income > 50: +1 yes -1 no
10
12
14
16
18
20
22
Education



Income > 50: +1 yes -1 no
1.0
0.5
0.0
−0.5
−1.0
10
12
14
16
18
20
22
Education


Label
Feature X
1.0
0.5
0.0
−0.5
−1.0
−2
−1
0
1
2


1.0
0.5
0.0
−0.5
−1.0
Label
−2
−1
0
1
2
Feature X
1
2
5
10



label +1
label -1
Feature X ** 2 (squared)
Feature X
0
1
2
3
4
5
6
−2
−1
0
1
2

Loss

How do we check how well $f(x) \approx y$ for a specific example?

One way (of many reasonable approaches): Write loss as $\ell(y', y)$: y' prediction, y truth for example features x. This is more "Frequentist" will see "Bayesian" way later

Classification:

Hamming $0/1$ loss: $\mathbb{1}(y' \neq y)$

Notation:

$$\mathbb{1}(A) = \begin{cases} 1 & \text{if } A \text{ is true} \\ 0 & \text{if } A \text{ is false} \end{cases}$$

Loss

How do we check how well $f(x) \approx y$ for a specific example?

One way (of many reasonable approaches): Write loss as $\ell(y', y)$: y' prediction, y truth for example features x. This is more “Frequentist” will see “Bayesian” way later

Regression:

Squared loss: $(y - y')^2$

Absolute loss: $|y - y'|$

Choice of loss has implications. Explore in next pset.

How to combine over all data? Empirical risk of function f

$$\widehat{R}(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i), y_i)$$

Doing so carries a lot of implications (everything weighted the same, bias towards majority group of data, ...), but it's a good place to start.

$\ell(y', y)$	$\widehat{R}$ name
$(y' - y)^2$	mean-squared error (MSE)
$\lvert y - y' \rvert$	mean absolute deviation (MAD)
$\mathbb{1}(y \neq y')$	Hamming error/0-1 Loss

Empirical Risk Minimization

$$\widehat{f} = \arg\min_{f \in \mathcal{F}} \widehat{R}(f) = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i)$$

$\mathcal{F}$: function/model class–how do we model data?

lots of choices. But first: concepts notation.

Training Risk vs. Test Risk

Learning with empirical risk is based on **training risk**: computed on data used in fitting/learning/training/estimating the model.

We are more interested in **test risk** computed on *unseen data*.

Training Risk vs. Test Risk

Learning with empirical risk is based on **training risk**: computed on data used in fitting/learning/training/estimating the model.

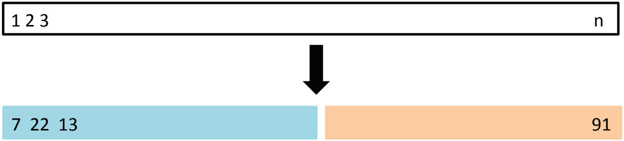
We are more interested in **test risk** computed on *unseen data*.What if we don't have other data?

Training Risk vs. Test Risk

Learning with empirical risk is based on **training risk**: computed on data used in fitting/learning/training/estimating the model.

We are more interested in **test risk** computed on *unseen data*. What if we don't have other data?

We can randomly split our data into a test set and a training set.



Help avoid over-fitting to training data min: Let S be a set of ordered values (e.g. numbers)

$$v = \min S$$

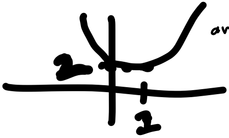
is the largest number such that no number in S is smaller than v

Let $h : \mathcal{X} \mapsto S$ and $X \subset \mathcal{X}$

$$\min_{x \in X} h(x) = \min\{y \mid \exists x \in X \text{ s.t. } y = h(x)\}$$

For $X = \mathcal{X}$

$$\min_{x} h(x) = \min_{x \in \mathcal{X}} h(x)$$



argmin $= \{1\}$
min $= 2$

$$\mathcal{V} = \arg\min_{x} h(x)$$

is the set of all possible values $g \in \mathcal{X}$ such that $h(g) = \min_x h(x)$.

$$g = \arg\min_{x} h(x)$$

for g to be anything in $\mathcal{V}$

Back to learning

Example 0 Suppose $x_i = 1 \ \forall i$ (for all i)

y_i height of person i.

$\mathcal{F} = \{f : \mathbb{R} \mapsto \mathbb{R} \mid \ \forall x \ f(x) = \mu\}$

MSE of $f \equiv \mu$

$$\widehat{R}(\mu) = \frac{1}{n} \sum_{i=1}^{n} (\mu - y_i)^2$$

Solution: $\widehat{\mu} = \frac{1}{n} \sum_i y_i$

Let $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$

$$
\begin{aligned}
\widehat{R}(\mu) &= \frac{1}{n}\sum_{i=1}^{n}(\mu - y_i)^2 \\
&= \frac{1}{n}\sum_{i=1}^{n}(\mu - \bar{y})^2 + \sum_{i=1}^{n}\frac{1}{n}(\bar{y} - y_i)^2 \quad \text{Let's go through the algebra}
\end{aligned}
$$

Let $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$

$$\widehat{R}(\mu) = \frac{1}{n}\sum_{i=1}^{n}(\mu - y_i)^2$$

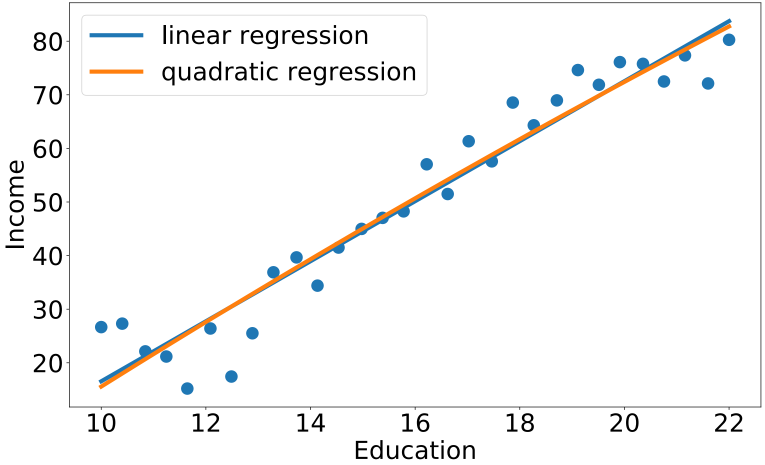
$$= \frac{1}{n}\sum_{i=1}^{n}(\mu - \bar{y})^2 + \frac{1}{n}(\bar{y} - y_i)^2$$ Let's go through the algebra

$$\frac{1}{n}\sum_{i=1}^{n}(\mu - y_i)^2 = \frac{1}{n}\sum_{i=1}^{n}(\mu - \bar{y} + \bar{y} - y_i)^2$$

$$= \frac{1}{n}\sum_{i=1}^{n}(\mu - \bar{y})^2 + \frac{1}{n}\sum_{i=1}^{n}(\bar{y} - y_i)^2 + \frac{2}{n}\sum_{i=1}^{n}(\mu - \bar{y})(\bar{y} - y_i)$$

$$= \frac{1}{n}\sum_{i=1}^{n}(\mu - \bar{y})^2 + \frac{1}{n}\sum_{i=1}^{n}(\bar{y} - y_i)^2$$

Example 1: Linear Regression



Example 1: Linear Regression $\mathcal{X} = \mathbb{R}$ and $\mathcal{Y} = \mathbb{R}$ that is $x_i \in \mathbb{R}$ $y_i \in \mathbb{R}$

Model: $\mathcal{F} = \{f(x) \mid f(x) = \theta_0 + \theta_1 x\} \rightarrow$ linear function

Model: $\mathcal{F} = \{f(x) \mid f(x) = \theta_0 + \theta_1 x + \theta_2 x^2\} \rightarrow$ quadratic function

Note: Generally the first model is considered linear regression, but later we will see why the second model is also linear regression but generally called quadratic regression.

Example 1: Linear Regression

Learn with Ordinary Least Squares. Minimize MSE

$\widehat{f}(x) = \widehat{\theta}_0 + \widehat{\theta}_1 x$

$$\widehat{\theta}_0, \widehat{\theta}_1 \in \arg\min_{\theta_0, \theta_1} \frac{1}{n} \sum_{i=1}^{n} (\theta_0 + \theta_1 x_i - y_i)^2$$

$\widehat{\theta}_0$ is the intercept

$$\widehat{\theta}_1 = \frac{\frac{1}{n}\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\frac{1}{n}\sum_i (x_i - \bar{x})^2}$$

$$\widehat{\theta}_0 = \bar{y} - \widehat{\theta}_1 \bar{x}$$

Can solve this with algebra. We will solve later using derivatives.

Example 2: Linear Regression

Full generality linear regression.

Learn with Ordinary Least Squares. Minimize MSE

$x_i \in \mathbb{R}^d$ (assume that $(x_i)_{(1)} = 1$ for every example)

Take $\theta \in \mathbb{R}^d \to f_\theta(x) = \sum_{j=1}^{d} \theta_{(j)} x_{(j)}$: linear/weighted combination

$$\widehat{\theta} \in \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} (f_\theta(x_i) - y_i)^2$$

Examples

Linear regression in 1-D: Data is $\widetilde{x}_i$ let $x \in \mathbb{R}^2$ $x = (1, \widetilde{x})$

$f(x) = \theta_{(1)} + \theta_{(2)} x_{(2)}$

Exactly the same as Example 1.

Quadratic regression in 1-D: Data is $\widetilde{x}_i$, let $x \in \mathbb{R}^3$ be $x = (1, \widetilde{x}, \widetilde{x}^2)$. This is called a feature mapping.

$$
\begin{aligned}
f(x) &= \theta_1 + \theta_2 x_2 + \theta_3 x_3 \\
&= \theta_1 + \theta_2 \widetilde{x} + \theta_3 \widetilde{x}^2
\end{aligned}
$$

How to find optimal solutions? This is at the heart of ERM based "learning"

Notation

$v, w \in \mathbb{R}^d$, then $\|v\|_2^2 = \sum_{i=1}^d v_{(i)}^2$

$\langle v, w \rangle = \sum_{i=1}^d v_{(i)} w_{(i)}$

Given $y_i \in \mathbb{R}$ for $i \in [n]$, then $y = \mathrm{vec}(y_i) = \text{vector}[y_i] \in \mathbb{R}^n$ with $y_{(i)} = y_i$.

1-D linear regression solutions:

$$\widehat{\theta}_0 = \bar{y} - \widehat{\theta}_1 \bar{x}$$
$$\widehat{\theta}_1 = \frac{\langle \mathrm{vec}(x_i - \bar{x}), \mathrm{vec}(y_i - \bar{y}) \rangle}{\| \mathrm{vec}(x_i - \bar{x}) \|_2^2}$$

Notation

$A \in \mathbb{R}^{n \times d}$ and $X \in \mathbb{R}^{d \times k}$ then $C = AX \in \mathbb{R}^{n \times k}$

$$C_{(ij)} = \sum_{\ell=1}^{d} A_{(i\ell)} X_{(\ell j)}$$

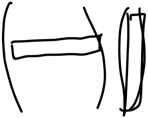
Transpose: $(A^T)_{(ij)} = A_{(ji)}$

Trace: $M \in \mathbb{R}^{d \times d}$ must be square $\mathrm{trace}(M) = \sum_{i=1}^{d} M_{(ii)}$

$C_{(ij)} = \langle A^T_{(i:)}, X_{(:j)} \rangle \quad = \quad A_{(i:)} \cdot X_{(:j)}$

$A_{(i:)}$ is the i^{th} row as a row vector. $X_{(:j)}$ is the j^{th} column as a column vector.

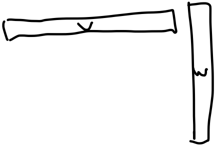
Recall vectors as $d \times 1$ or $1 \times d$ matrices. $v, w \in \mathbb{R}^d$



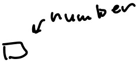
$$(Av)_{(j)} = \sum_{\ell} A_{(i\ell)} v_{(\ell)}$$
$$= \left(A_{(i:)} v\right) = \langle A^T_{(i:)}, v \rangle$$

$$v^T w = \overset{w^T v}{\cancel{w^T v\,\mathrm{quad}}} \quad \text{since transpose of a number is itself}$$
$$= \mathrm{trace}(w^T v) \quad \text{since trace of a number is itself}$$
$$= \mathrm{trace}(v w^T) \quad \text{HW}$$
$$= \langle v, w \rangle$$

$v^T w =$



$=$



Linear regression compactify

$x_i, \theta \in \mathbb{R}^d \to f_\theta(x) = \langle x, \theta \rangle = x^T\theta$

$X = \mathrm{matrix}[x_i]$, $y = \mathrm{vec}(y_i)$

$$X = \begin{pmatrix} - & x_1^T & - \\ - & x_2^T & - \\ & \vdots & \\ - & x_n^T & - \end{pmatrix} \qquad y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

Ordinary Least Squares

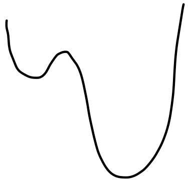
$$\widehat{\theta} \in \arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} (f_\theta(x_i) - y_i)^2$$

Can rewrite empirical risk as

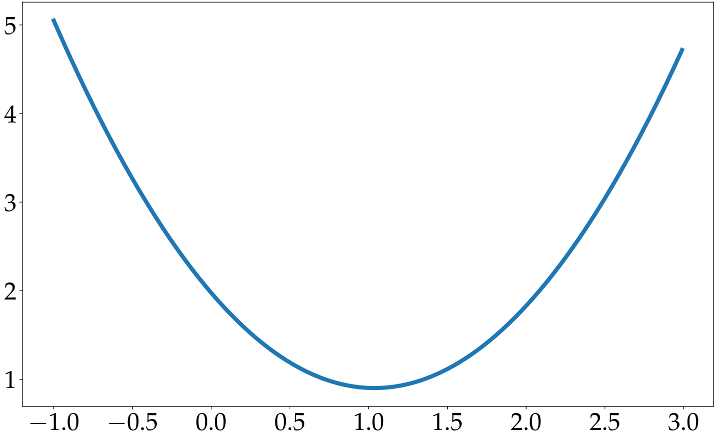
$$\widehat{R}(\theta) = \frac{1}{n} \|X\theta - y\|_2^2$$

Back to optimization

Example 0: $\widehat{R}(\mu) = (\mu - \bar{y})^2 + \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2$

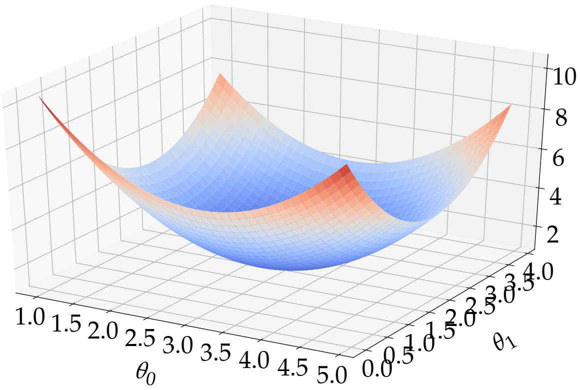


Generated: `y = np.random.randn(100) + 1`



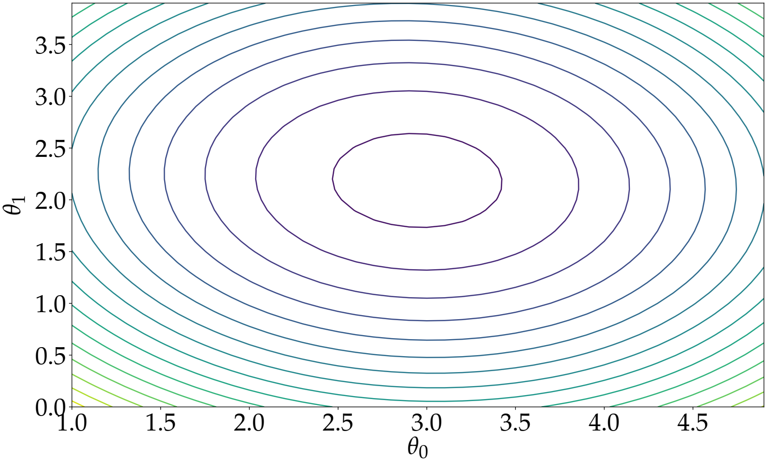
Example 1: $\widehat{R}(\theta_0, \theta_1) = \frac{1}{n}\sum_{i=1}^{n}(\theta_0 + \theta_1 x_i - y_i)^2$

```
n=100;x=np.random.randn(n);
y=x*2+3+np.random.randn(n)
```



Example 1: $\widehat{R}(\theta_0, \theta_1) = \sum_{i=1}^{n}(\theta_0 + \theta_1 x_i - y_i)^2$

```
n=100;x=np.random.randn(n);
y=x*2+3+np.random.randn(n)
```



Recap

Learning as optimization

$$\widehat{f} = \arg\min_{f \in \mathcal{F}} \widehat{R}(f)$$

$\mathcal{F}$ is a model. So far linear regression.

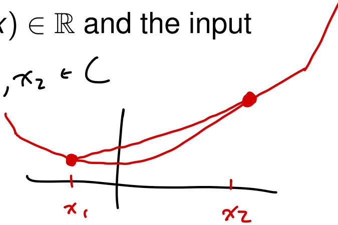
Let's get to solving.

Bowl shapes are special

Notation: Convex functions and gradients

Take $g : C \mapsto \mathbb{R}$ a function $C \subset \mathbb{R}^d$. That is $g(x) \in \mathbb{R}$ and the input space (domain) of g is C.

$x_1, x_2 \in C$



Convex: For $\lambda \in [0, 1]$ we have
$\lambda g(x_1) + (1 - \lambda) g(x_2) \geq g(\lambda x_1 + (1 - \lambda) x_2)$

Partial derivative: Define $e_i \in \mathbb{R}^d$ such that $(e_i)_{(j)} = \mathbb{1}(i = j)$

$$\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

$$\left.\frac{\partial g(x)}{\partial x_{(i)}}\right|_{x=z} = \lim_{h \to 0} \frac{g(z + h e_i) - g(z)}{h}$$

Partial derivative evaluated at point z.

Overload notation

$$\frac{\partial g(x)}{\partial x_{(i)}}$$

Is partial derivative evaluated at point x

Gradient (local direction of ascent):

$$\nabla_x g(x) = \begin{pmatrix} \frac{\partial g(x)}{\partial x_{(1)}} \\ \frac{\partial g(x)}{\partial x_{(2)}} \\ \vdots \\ \frac{\partial g(x)}{\partial x_{(d)}} \end{pmatrix}$$

Optimization: $\widehat{x} \in \arg\min_x g(x) \iff \nabla g(\widehat{x}) = 0$

For convex: local stationarity implies optimality. Not so for general functions.

Example 0

$$\widehat{R}(\mu) = \frac{1}{n}\sum_{i=1}^{n}(\mu - y_i)^2$$

Take derivative (since no need for partial derivative)

$$\widehat{R}'(\mu) = \frac{2}{n}\sum_{i=1}^{n}(\mu - y_i)$$

Set to zero and solve

$$\frac{2}{n}\sum_{i=1}^{n}(\mu - y_i) = 0$$

$$\widehat{\mu} = \frac{1}{n}\sum_{i=1}^{n} y_i$$

Example 2

$g(\theta) = (x\theta - y)^2$

$g'(\theta) = 2(x\theta - y)x$

$$\widehat{R}(\theta) = \frac{1}{n}\|X\theta - y\|_2^2$$

$e = X\theta - y$

Gradient

$$\nabla\widehat{R}(\theta) = \frac{2}{n}X^T(X\theta - y) \Rightarrow (\nabla\hat{R}(\theta))_{(j)} = (X_{(:,j)})^T e$$

Chain rule, analogous to taking derivative $(x\theta - y)^2$ for $x, y \in \mathbb{R}$. Set to zero.

$$\frac{2}{n}X^T(X\theta - y) = 0$$

Solve

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

Calculations

$$X = \begin{pmatrix} - x_1^T - \\ - x_2^T - \\ \vdots \\ - x_n^T - \end{pmatrix}$$

$\widehat{R}(\theta) = \frac{1}{n} \sum_{i=1}^{n} (x_i^T \theta - y_i)^2$

$X_{(ij)} = (x_i)_{(j)}$

$$\frac{\partial \widehat{R}}{\partial \theta_{(j)}} = \frac{1}{n} \sum_{i=1}^{n} \frac{\partial (x_i^T \theta - y_i)^2}{\partial \theta_{(j)}}$$

$$= \frac{2}{n} \sum_{i=1}^{n} (x_i)_{(j)} (x_i^T \theta - y_i) \qquad (*)$$

$$= \frac{2}{n} X_{(:j)}^T (X\theta - y)$$

$e = X\theta - y$

$v = X_{(:j)}$

Why?

Define $v \in \mathbb{R}^n$ with $v_{(i)} = (x_i)_{(j)}$

$e \in \mathbb{R}^n$ $\quad e_{(i)} = x_i^T \theta - y_i$

$(*) = \frac{2}{n} v^T e = \frac{2}{n} (X_{(:j)})^T (X\theta - y)$

Matrix Calculations

Notation: $e = X\theta - y$

$\uparrow$ error vector

$$\widehat{R}(\theta) = \|X\theta - y\|_2^2$$

Suppose we have a function $g : \mathbb{R}^n \mapsto \mathbb{R}$, let $h(v) = \nabla g(v)$ then

$$\nabla_\theta g(X\theta - y) = X^T h(v) \quad \text{chain rule}$$

$v = X\theta - y$

Often write

$$\frac{\partial g(X\theta - y)}{\partial(X\theta - y)} = \nabla g(v)|_{v = X\theta - y} = h(X\theta - y)$$

Example: $\cancel{h}\,g(v) = \|v\|_2^2$ can verify that $\nabla \|v\|_2^2 = 2v$, then

$$\nabla \|X\theta - y\|_2^2 = X^T(2(X\theta - y))$$

$$g(v) = \sum_{i=1}^{n} v_{(i)}^2$$

$$\frac{\partial g}{\partial v_{(j)}} = \sum_{i=1}^{n} \frac{\partial (v_{(i)}^2)}{\partial v_{(j)}}$$

$$= \sum_{i=1}^{n} 2 v_{(i)} \mathbb{1}(i=j)$$

$$= 2 v_{(j)}$$

$$\Rightarrow \nabla g = 2v$$

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

Some issues

- $(X^T X)$ invertibility
- $O(p^3)$ computation (O is like “order of”)
- can be sensitive to noise/outliers

Conclusion

Learning/training/estimating model as optimization

Training vs test set

Data is random, avoid over-fitting to training data

Gradients help to understand optimality