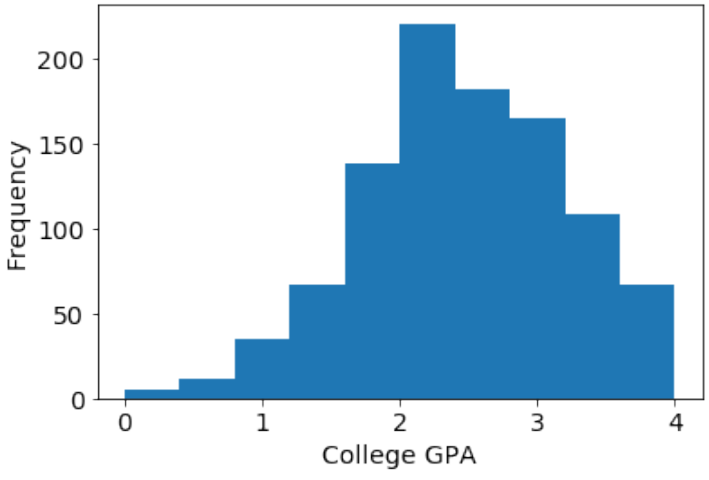
Linear regression

Linear regression

Fitting a line to a bunch of points.

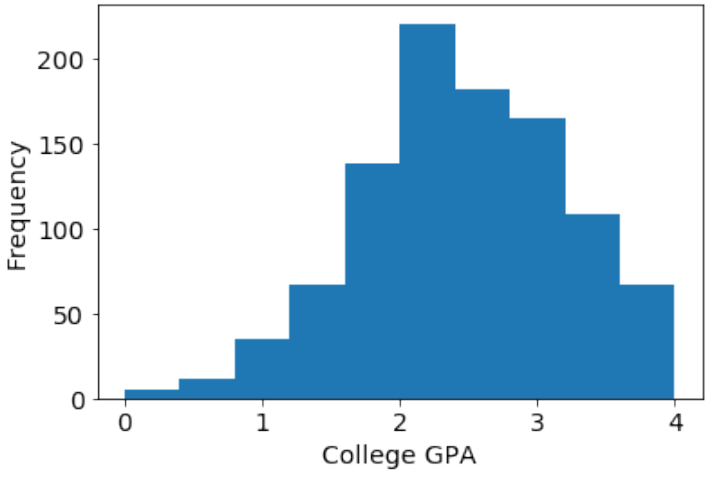
Example: college GPAs

Distribution of GPAs of students at a certain Ivy League university.



Example: college GPAs

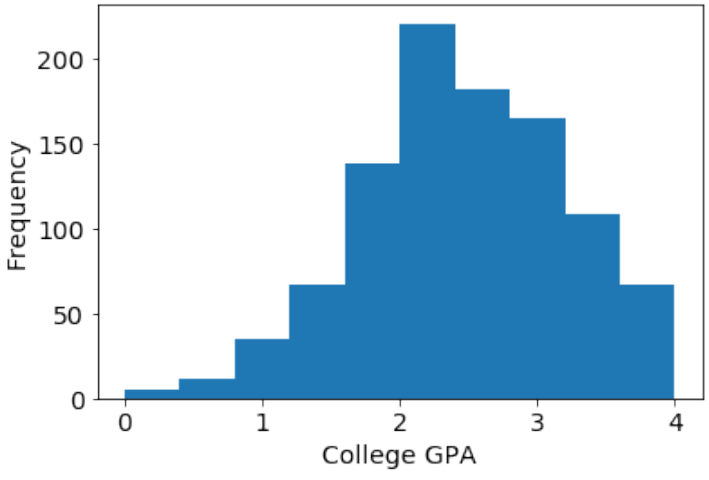
Distribution of GPAs of students at a certain Ivy League university.



What GPA to predict for a random student from this group?

Example: college GPAs

Distribution of GPAs of students at a certain Ivy League university.

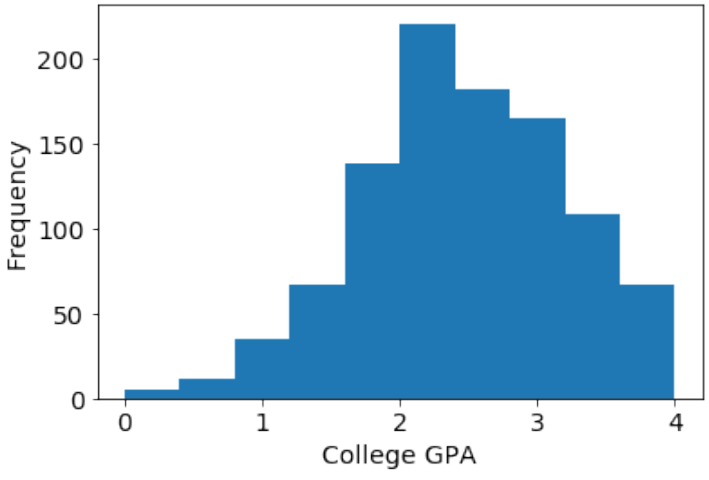


What GPA to predict for a random student from this group?

- Without further information, predict the **mean**, 2.47.

Example: college GPAs

Distribution of GPAs of students at a certain Ivy League university.

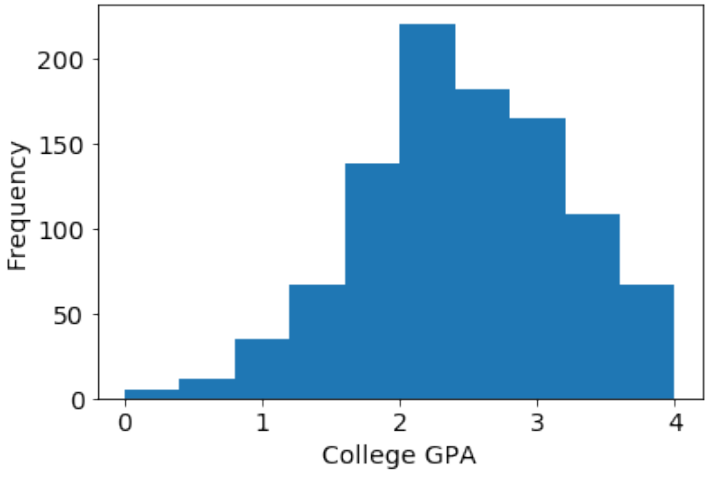


What GPA to predict for a random student from this group?

- Without further information, predict the **mean**, 2.47.
- What is the average squared error of this prediction? That is, $\mathbb{E}[((\text{student's GPA}) - (\text{predicted GPA}))^2]$?

Example: college GPAs

Distribution of GPAs of students at a certain Ivy League university.

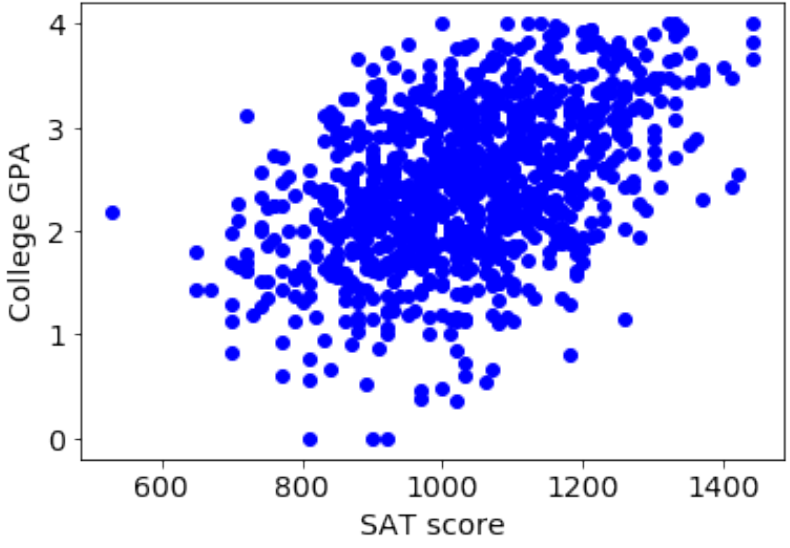


What GPA to predict for a random student from this group?

- Without further information, predict the **mean**, 2.47.
- What is the average squared error of this prediction? That is, $\mathbb{E}[((\text{student's GPA}) - (\text{predicted GPA}))^2]$? The **variance** of the distribution, 0.55.

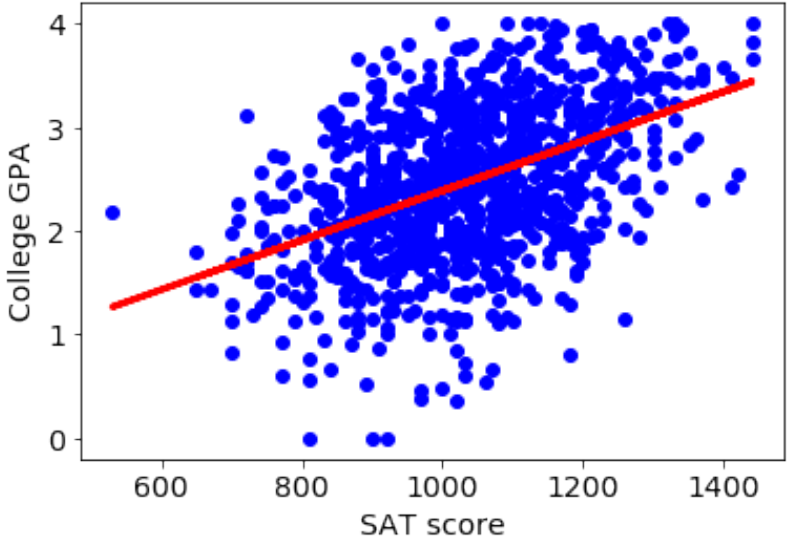
Better predictions with more information

We also have SAT scores of all students.



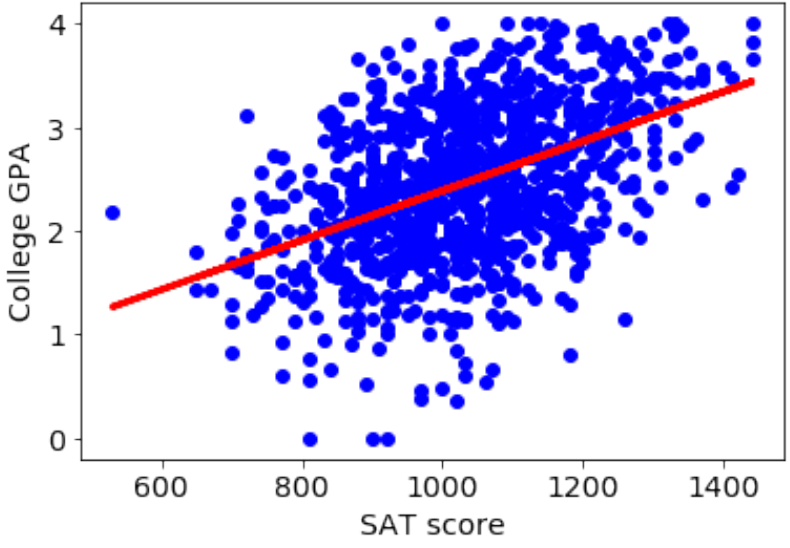
Better predictions with more information

We also have SAT scores of all students.



Better predictions with more information

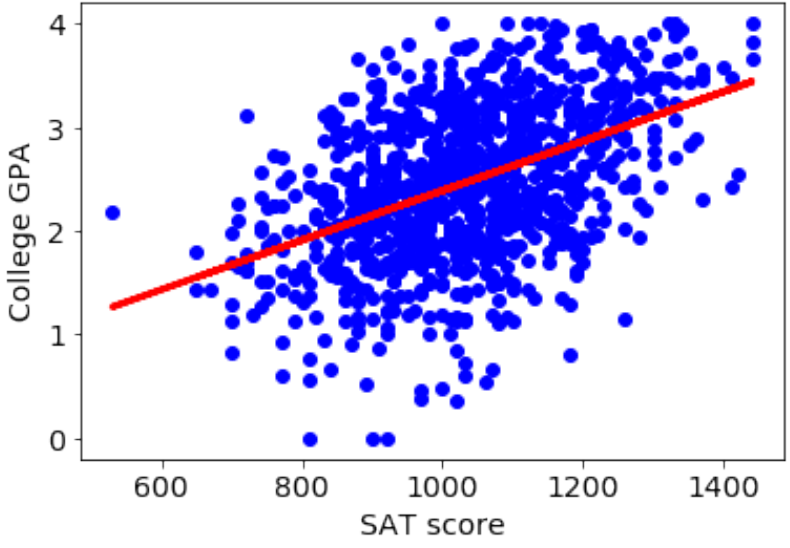
We also have SAT scores of all students.



Mean squared error (MSE) drops to 0.43.

Better predictions with more information

We also have SAT scores of all students.



Mean squared error (MSE) drops to 0.43.

This is a **regression** problem with:

- **Predictor variable**: SAT score
- **Response variable**: College GPA

Parametrizing a line

A line can be parameterized as $y = ax + b$ (a: slope, b: intercept).

The line fitting problem

Pick a line (a, b) based on $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)}) \in \mathbb{R} \times \mathbb{R}$

- $x^{(i)}, y^{(i)}$ are predictor and response variables.
  E.g. SAT score, GPA of ith student.
- Minimize the mean squared error,

$$\text{MSE}(a, b) = \frac{1}{n} \sum_{i=1}^{n} (y^{(i)} - (ax^{(i)} + b))^2.$$

This is the **loss function**.

Minimizing the loss function

Given $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})$, minimize

$$L(a, b) = \sum_{i=1}^{n} (y^{(i)} - (ax^{(i)} + b))^2.$$

Multivariate regression: diabetes study

Data from $n = 442$ diabetes patients.

For each patient:

- 10 features $x = (x_1, \ldots, x_{10})$

  age, sex, body mass index, average blood pressure, and six blood serum measurements.

- A real value y: the progression of the disease a year later.

Regression problem:

- **response** $y \in \mathbb{R}$
- **predictor variables** $x \in \mathbb{R}^{10}$

Least-squares regression

Linear function of 10 variables: for $x \in \mathbb{R}^{10}$,

$$f(x) \;=\; w_1 x_1 + w_2 x_2 + \cdots + w_{10} x_{10} + b \;=\; w \cdot x + b$$

where $w = (w_1, w_2, \ldots, w_{10})$.

Least-squares regression

Linear function of 10 variables: for $x \in \mathbb{R}^{10}$,

$$f(x) \;=\; w_1 x_1 + w_2 x_2 + \cdots + w_{10} x_{10} + b \;=\; w \cdot x + b$$

where $w = (w_1, w_2, \ldots, w_{10})$.

Penalize error using **squared loss** $(y - (w \cdot x + b))^2$.

Least-squares regression

Linear function of 10 variables: for $x \in \mathbb{R}^{10}$,

$$f(x) = w_1 x_1 + w_2 x_2 + \cdots + w_{10} x_{10} + b = w \cdot x + b$$

where $w = (w_1, w_2, \ldots, w_{10})$.

Penalize error using **squared loss** $(y - (w \cdot x + b))^2$.

Least-squares regression:

- *Given:* data $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)}) \in \mathbb{R}^d \times \mathbb{R}$
- *Return:* linear function given by $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$
- *Goal:* minimize the **loss function**

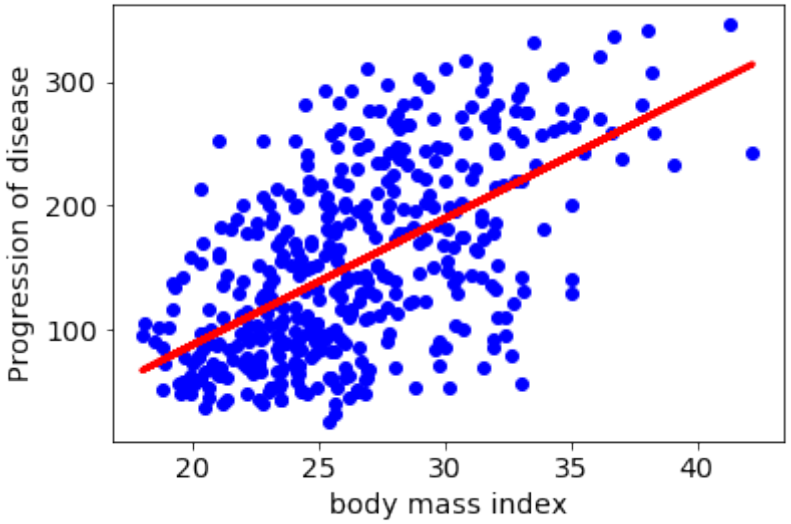
$$L(w, b) = \sum_{i=1}^{n} (y^{(i)} - (w \cdot x^{(i)} + b))^2.$$

Back to the diabetes data

- No predictor variables: mean squared error (MSE) = 5930

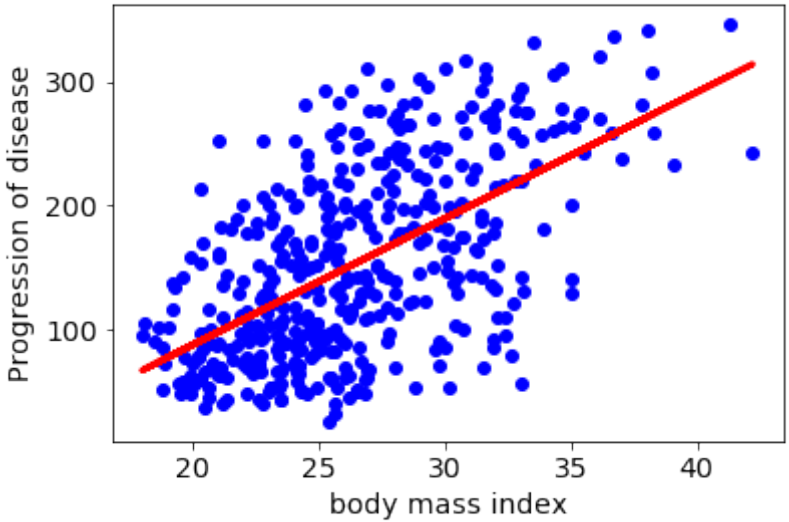
Back to the diabetes data

- No predictor variables: mean squared error (MSE) = 5930
- One predictor ('bmi'): MSE = 3890



Back to the diabetes data

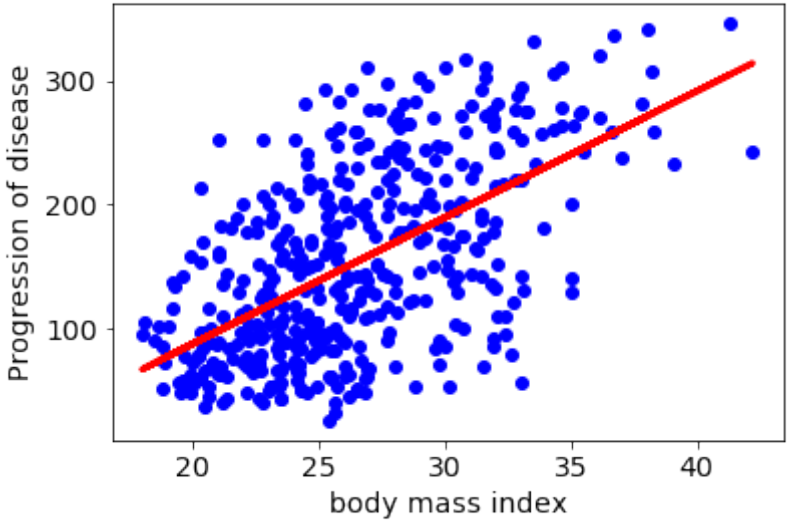
- No predictor variables: mean squared error (MSE) = 5930
- One predictor ('bmi'): MSE = 3890



- Two predictors ('bmi', 'serum5'): MSE = 3205

Back to the diabetes data

- No predictor variables: mean squared error (MSE) = 5930
- One predictor ('bmi'): MSE = 3890



- Two predictors ('bmi', 'serum5'): MSE = 3205
- All ten predictors: MSE = 2860

Least-squares solution 1

Linear function of d variables given by $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$:

$$f(x) = w_1 x_1 + w_2 x_2 + \cdots + w_d x_d + b = w \cdot x + b$$

$$\begin{bmatrix} w_1 \\ \vdots \\ w_d \end{bmatrix} \quad \begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix} \rightarrow d \text{ dim}$$

$$\tilde{w} = \begin{bmatrix} b \\ w_1 \\ \vdots \\ w_d \end{bmatrix} \qquad \tilde{x} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix}$$

$(d+1)$ - dim.

$$\langle \tilde{w}, \tilde{x} \rangle = \sum_{i=1}^{d+1} \tilde{w}_i \tilde{x}_i$$

$$= b + \sum_{i=1}^{d} w_i x_i$$

$$= b + \langle w, x \rangle$$

Least-squares solution 1

Linear function of d variables given by $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$:

$$f(x) = w_1 x_1 + w_2 x_2 + \cdots + w_d x_d + b = w \cdot x + b$$

Assimilate the intercept b into w:

- Add a new feature that is identically 1: let $\widetilde{x} = (1, x) \in \mathbb{R}^{d+1}$

$$\begin{pmatrix} 4 & 0 & 2 & \cdots & 3 \end{pmatrix} \Longrightarrow \begin{pmatrix} 1 & 4 & 0 & 2 & \cdots & 3 \end{pmatrix}$$

- Set $\widetilde{w} = (b, w) \in \mathbb{R}^{d+1}$
- Then $f(x) = w \cdot x + b = \widetilde{w} \cdot \widetilde{x}$

Goal: find $\widetilde{w} \in \mathbb{R}^{d+1}$ that minimizes

$$L(\widetilde{w}) = \sum_{i=1}^{n} \left(y^{(i)} - \widetilde{w} \cdot \widetilde{x}^{(i)} \right)^2$$

predicted value

actual response

Least-squares solution 2

Write

$$X = \begin{pmatrix} \longleftarrow & \widetilde{x}^{(1)} & \longrightarrow \\ \longleftarrow & \widetilde{x}^{(2)} & \longrightarrow \\ & \vdots & \\ \longleftarrow & \widetilde{x}^{(n)} & \longrightarrow \end{pmatrix}, \quad y = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{pmatrix}$$

X ↓ data matrix

$n \times (d+1)$

Then the loss function is

$$L(\widetilde{w}) = \sum_{i=1}^{n} (y^{(i)} - \widetilde{w} \cdot \widetilde{x}^{(i)})^2 = \|y - X\widetilde{w}\|^2$$

and it minimized at $\widetilde{w} = (X^T X)^{-1}(X^T y)$.

$(d+1) \times 1$.

$(d+1) \times (d+1)$ $(d+1) \times n$, $n \times 1$

$(d+1) \times 1$

Generalization behavior of least-squares regression

Given a **training set** $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)}) \in \mathbb{R}^d \times \mathbb{R}$, find a linear function, given by $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$, that minimizes the squared loss

$$L(w, b) = \sum_{i=1}^{n} (y^{(i)} - (w \cdot x^{(i)} + b))^2.$$

Is training loss a good estimate of **future** performance?

Generalization behavior of least-squares regression

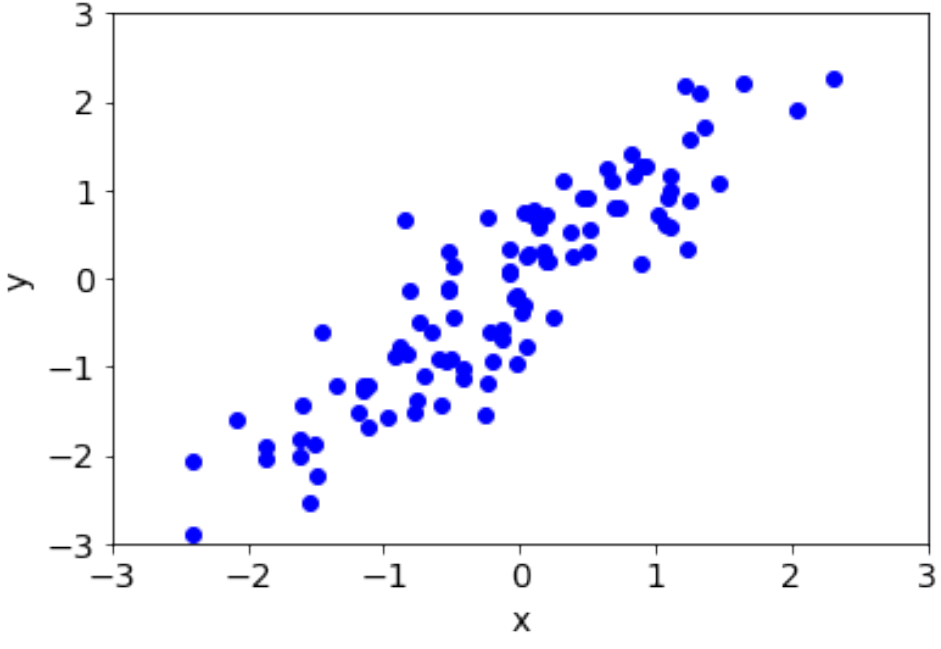
Given a **training set** $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)}) \in \mathbb{R}^d \times \mathbb{R}$, find a linear function, given by $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$, that minimizes the squared loss

$$L(w, b) = \sum_{i=1}^{n} (y^{(i)} - (w \cdot x^{(i)} + b))^2.$$

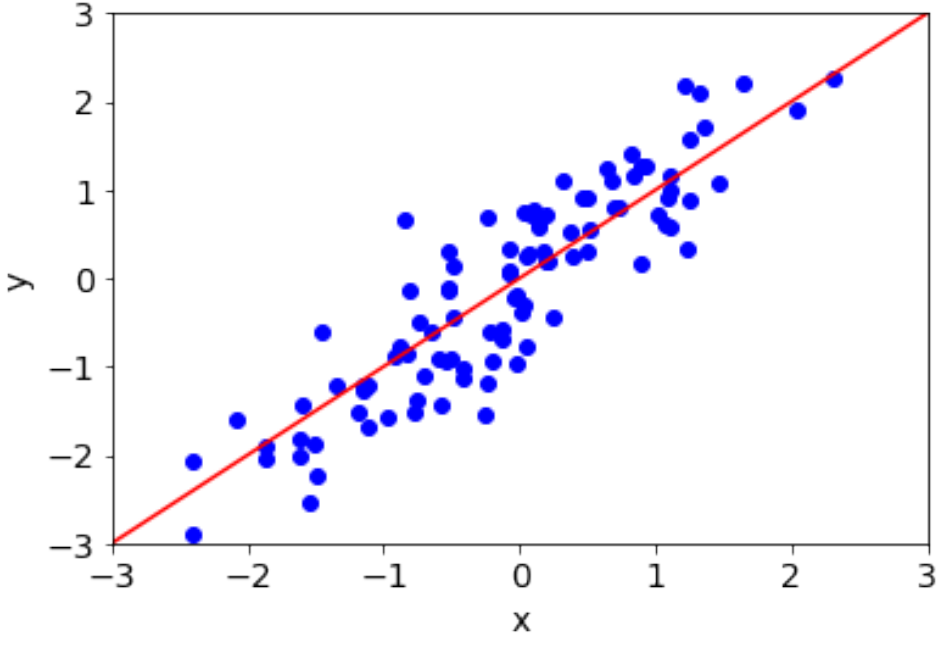
Is training loss a good estimate of **future** performance?

- If n is large enough: maybe.
- Otherwise: probably an underestimate.

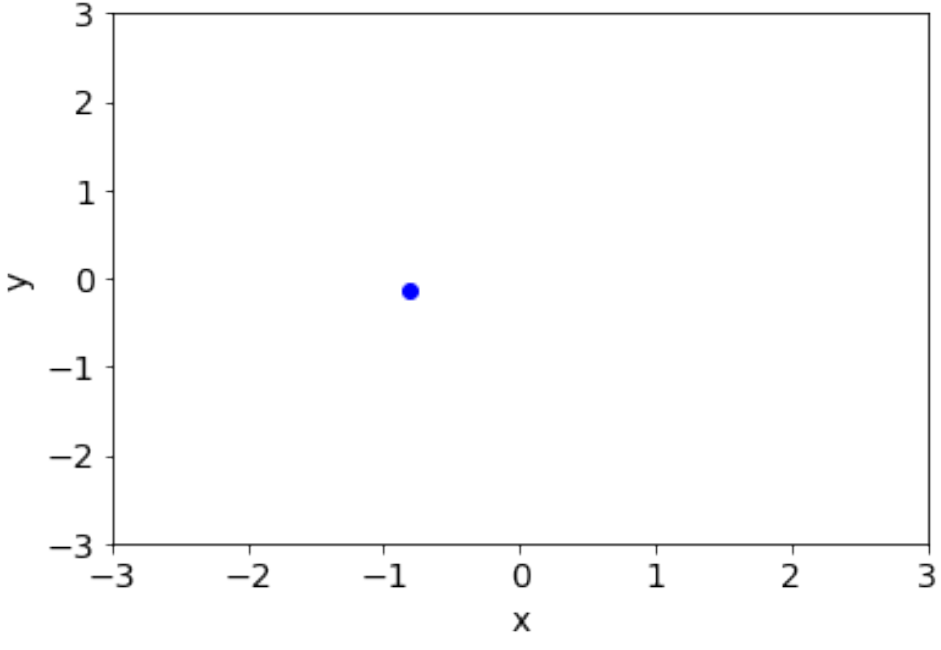
Example



Example



Example



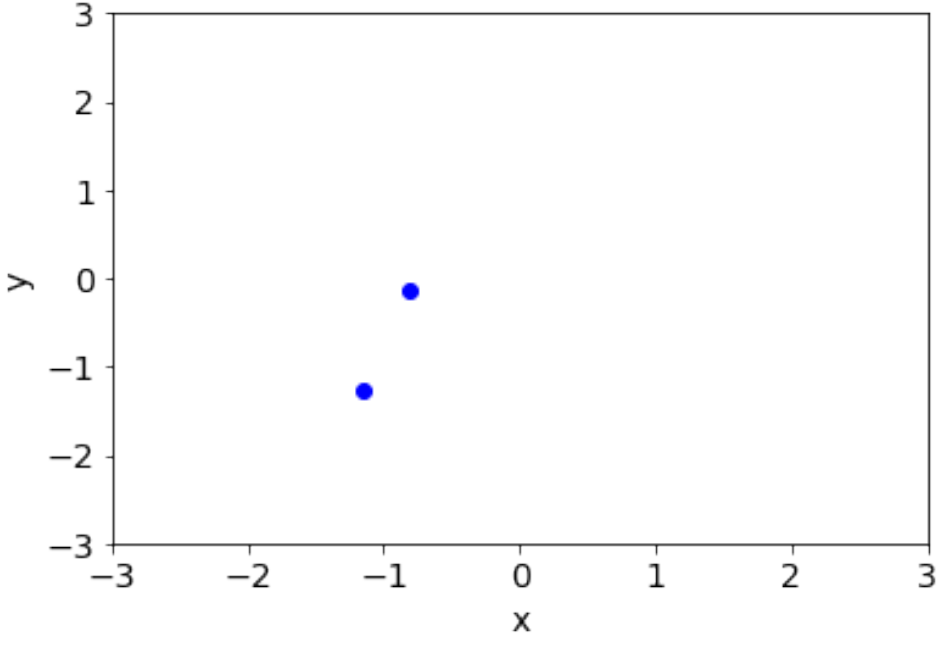
Example

$y^{(i)} = \langle w, x^{(i)} \rangle + b$

So $\text{Loss}(w, b) = \sum_i \left(y^{(i)} - (\langle w, x^{(i)} \rangle + b) \right)^2$

$y^{(i)} - (\langle w, x^{(i)} \rangle + b) = 0$

$= 0$



Better error estimates

Recall: k-**fold cross-validation**

- Divide the data set into k equal-sized groups $S_1, \ldots, S_k$
- For $i = 1$ to k:
  - Train a regressor on all data except S_i
  - Let E_i be its error on S_i
- Error estimate: average of $E_1, \ldots, E_k$

Better error estimates

Recall: k-**fold cross-validation**

- Divide the data set into k equal-sized groups $S_1, \ldots, S_k$
- For $i = 1$ to k:
  - Train a regressor on all data except S_i
  - Let E_i be its error on S_i
- Error estimate: average of $E_1, \ldots, E_k$

A nagging question:
When n is small, should we be minimizing the squared loss?

$$L(w, b) = \sum_{i=1}^{n} (y^{(i)} - (w \cdot x^{(i)} + b))^2$$

Ridge regression

Minimize squared loss **plus** a term that penalizes "complex" w:

$$L(w, b) = \sum_{i=1}^{n} (y^{(i)} - (w \cdot x^{(i)} + b))^2 + \lambda \|w\|^2$$

Adding a penalty term like this is called **regularization**.

Ridge regression

Minimize squared loss **plus** a term that penalizes "complex" w:

$$L(w, b) = \sum_{i=1}^{n} (y^{(i)} - (w \cdot x^{(i)} + b))^2 + \lambda \|w\|^2$$

Adding a penalty term like this is called **regularization**.

Put predictor vectors in matrix X and responses in vector y:

$$w = (X^T X + \lambda I)^{-1} (X^T y)$$

Toy example

Training, test sets of 100 points

- $x \in \mathbb{R}^{100}$, each feature x_i is Gaussian $N(0, 1)$
- $y = x_1 + \cdots + x_{10} + N(0, 1)$

Toy example

Training, test sets of 100 points

- $x \in \mathbb{R}^{100}$, each feature x_i is Gaussian $N(0, 1)$
- $y = x_1 + \cdots + x_{10} + N(0, 1)$

λ	training MSE	test MSE
0.00001	0.00	585.81
0.0001	0.00	564.28
0.001	0.00	404.08
0.01	0.01	83.48
0.1	0.03	19.26
1.0	0.07	7.02
10.0	0.35	2.84
100.0	2.40	5.79
1000.0	8.19	10.97
10000.0	10.83	12.63

The lasso

Popular "shrinkage" estimators:

- **Ridge regression**

$$L(w, b) = \sum_{i=1}^{n} (y^{(i)} - (w \cdot x^{(i)} + b))^2 + \lambda \|w\|_2^2$$

- **Lasso**: tends to produce sparse w

$$L(w, b) = \sum_{i=1}^{n} (y^{(i)} - (w \cdot x^{(i)} + b))^2 + \lambda \|w\|_1$$

The lasso

Popular "shrinkage" estimators:

- **Ridge regression**

$$L(w, b) = \sum_{i=1}^{n} (y^{(i)} - (w \cdot x^{(i)} + b))^2 + \lambda \|w\|_2^2$$

- **Lasso**: tends to produce sparse w

$$L(w, b) = \sum_{i=1}^{n} (y^{(i)} - (w \cdot x^{(i)} + b))^2 + \lambda \|w\|_1$$

Toy example:
Lasso recovers 10 relevant features plus a few more.