

Regression

Goal: Fit a model to predict a continuous target value:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p$$

Simple linear regression:

$$y = \beta_0 + \beta_1 x_1$$

$$y = ax + b$$

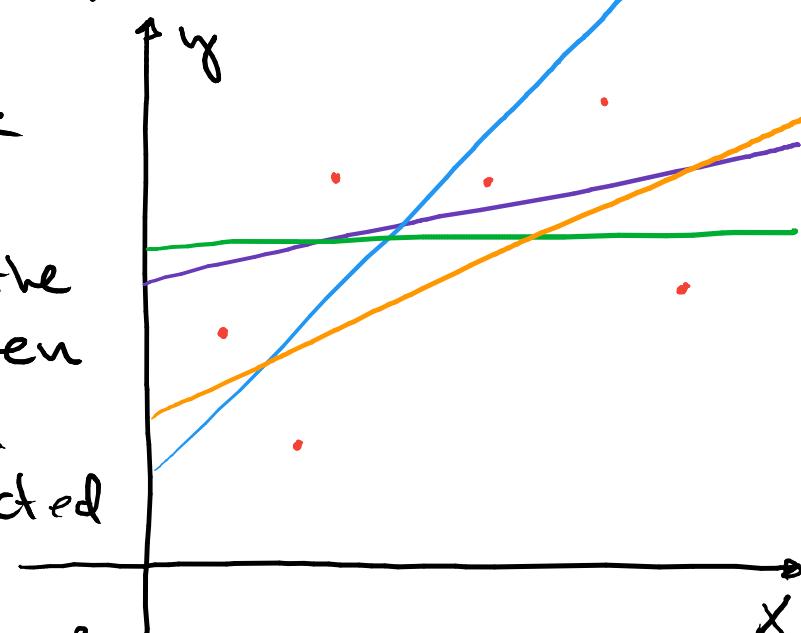
Determine β_0, β_1 , s.t. the "best" line is fitted:

By "best" we mean the one that reduces the error between the observed and the predicted value:

$$SSE = \sum (y_i - \hat{y}_i)^2$$

Observed

Predicted



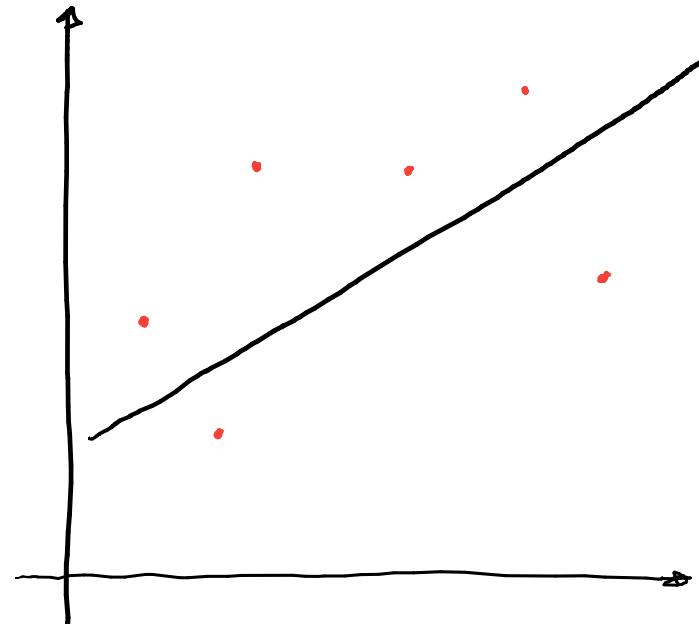
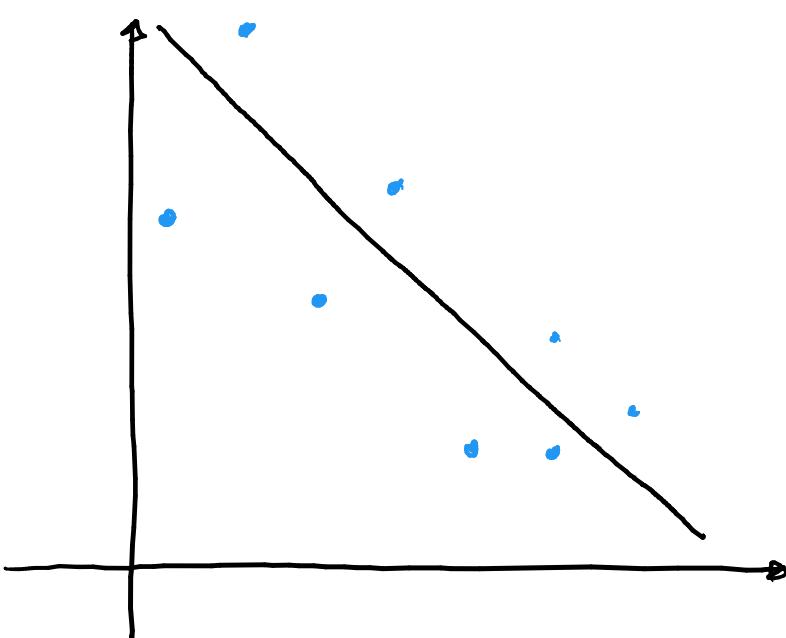
The association between x and y is called the **Correlation** and is denoted r : $-1 \leq r \leq 1$

$r < 0$:

Negative correlation

$r > 0$:

Positive correlation



When $r = 0$: No correlation

$$SSE = \sum (y_i - \hat{y}_i)^2$$

The goal of least squares regression is to minimize the sum of these squares, s.t.

$$\hat{y} = \beta_0 + \beta_1 x$$

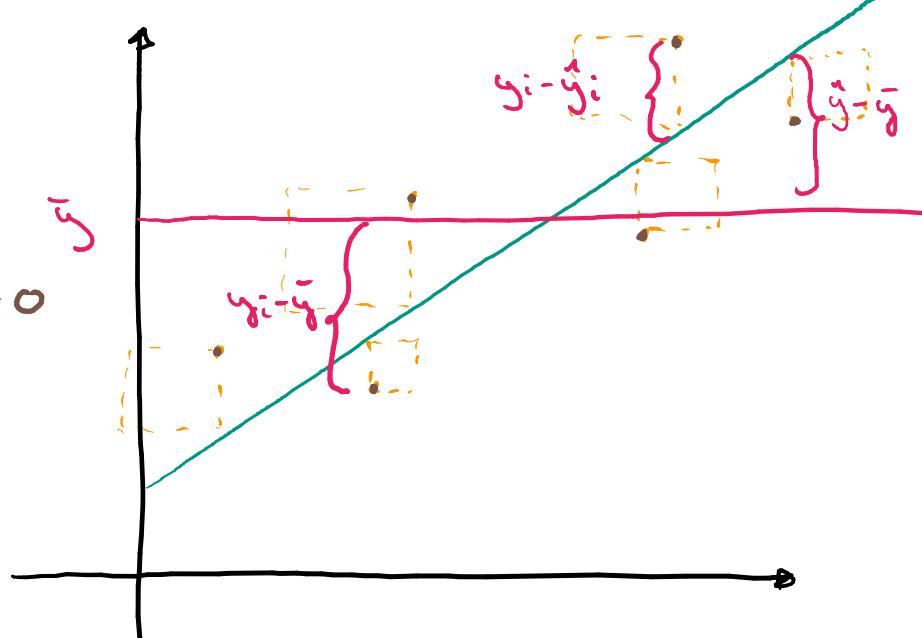
the smallest possible error

Errors:

$$SST = \sum (y_i - \bar{y})^2 \quad (\text{Total sum of squares})$$

$$SSE = \sum (y_i - \hat{y})^2 \quad (\text{Sum of squared errors})$$

$$SSR =$$



Deviations:

Total Deviation:

Explained :

Unexplained :

L

$$r^2 = \frac{SSR}{SST} = \frac{\text{Explained}}{\text{Total}}$$

} r^2 tells us the amount of variance my model is able to explain

$$0 \leq r^2 \leq 1$$

r^2 tells us something about our model's predictive ability.

r tells us something about how well X and y co-vary.

How to determine Parameters $\hat{\beta}_i$:

1) Least squares

↳ statistics

↳ linear algebra

2) Gradient Descent (Today)

↳ Often used to explain regression in M.L. since it is an easy entry to gradient descent

3) Singular Value Decomposition (Next week)

↳ Efficient, the one that is implemented

Simple linear Regression

↳ Supervised Learning Problem

Training Data: $\{(x_i, y_i)\}_{i=1}^n$

Model: $\hat{y} = f(x) = \beta_0 + \beta_1 x$

We define a loss function

$$\mathcal{L} = \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2 = \frac{1}{n} \sum_{i=1}^n (y_i - \beta_0 - \beta_1 x_i)^2$$

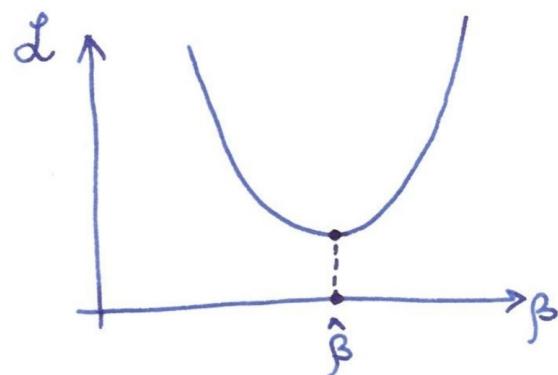
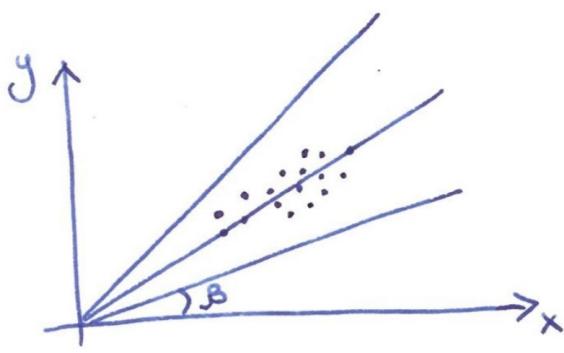
This is called $MSE = \text{Mean Squared Error}$

$$MSE =$$

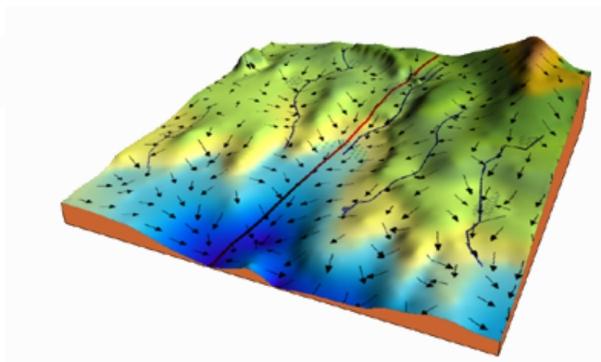
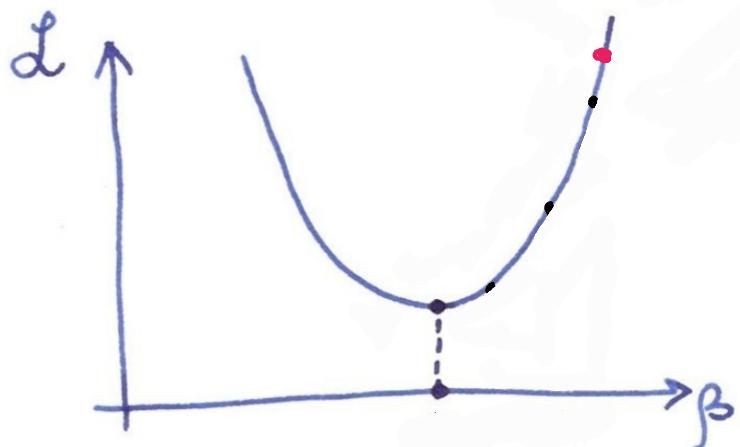
Baby linear Regression:

$$f(x) = \beta x$$

$$L(\beta) = \frac{1}{n} \sum (y_i - \hat{\beta} x_i)^2$$



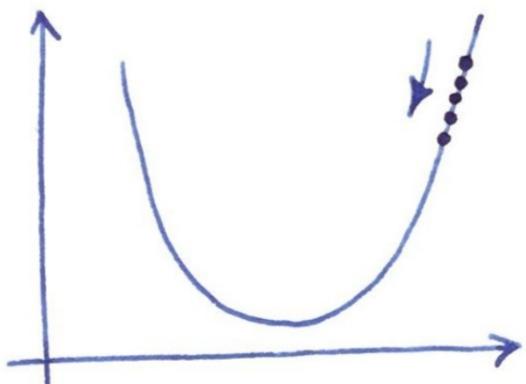
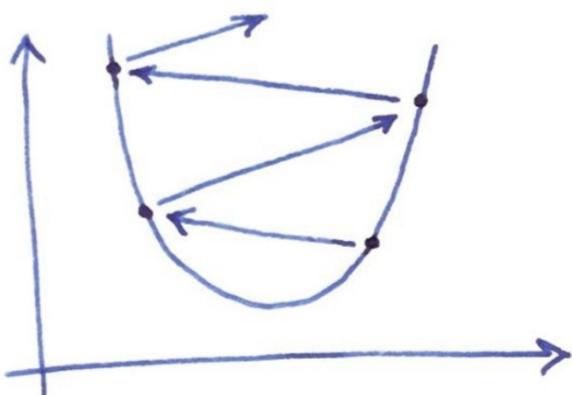
Gradient Descent



Update Rule:

$$\beta \leftarrow \beta - \eta \cdot \frac{d L(\beta)}{d \beta}$$

Here η^* (eta) is the learning rate



Too large $\eta \rightarrow$ Divergence

Too small $\eta \rightarrow$ slow convergence

Finding the derivative: Gradient Descent

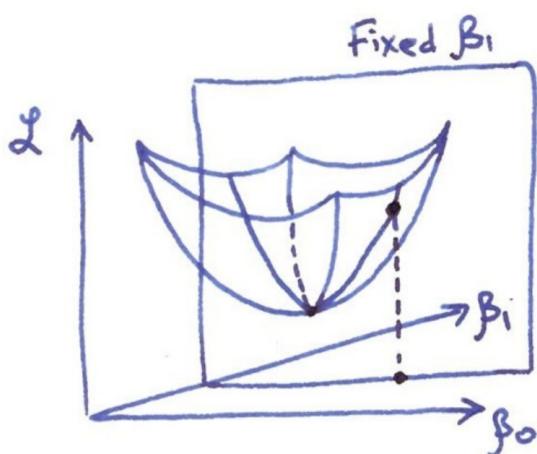
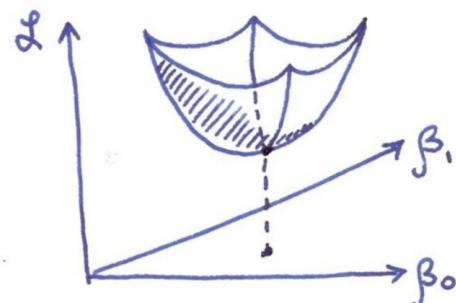
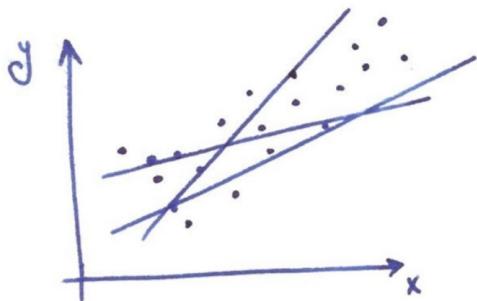
$$L(\beta) = \frac{1}{n} \sum (y_i - \beta x_i)^2$$

$$L'(\beta) =$$

=

Linear Regression: Gradient Descent:

$$\mathcal{L}(\beta_0, \beta_1) = \frac{1}{n} \sum_{i=1}^n (y_i - \beta_0 - \beta_1 x_i)^2$$



$$\beta_0 \leftarrow \beta_0 - \eta \frac{\partial \mathcal{L}}{\partial \beta_0}$$

$$\beta_1 \leftarrow \beta_1 - \eta \frac{\partial \mathcal{L}}{\partial \beta_1}$$

In vector form:

$$\bar{\beta} \leftarrow$$

Computing the gradient:

$$\mathcal{L} = \frac{1}{n} \sum_{i=1}^n (y_i - \beta_0 - \beta_1 x_i)^2$$

We need partial derivatives:

$$\frac{\partial \mathcal{L}}{\partial \beta_0} =$$

$$\frac{\partial \mathcal{L}}{\partial \beta_1} =$$

Why MSE?

Assume SSE:

n	L
100	200
500	800
1000	1200

OK if we want to compare models!

As n increases, SSE gets larger!

MSE	n	L	L/n
	100	200	2
	500	800	1.6
	1000	1200	1.2

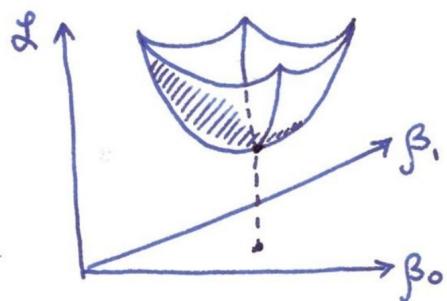
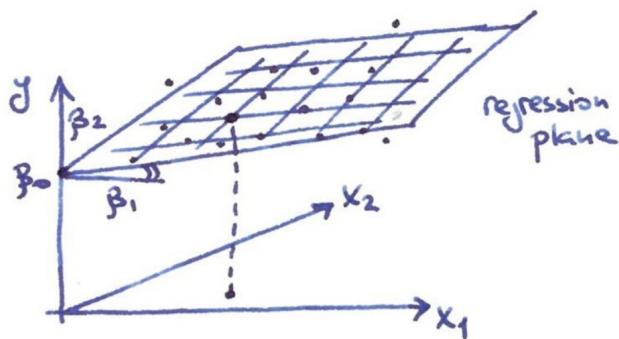
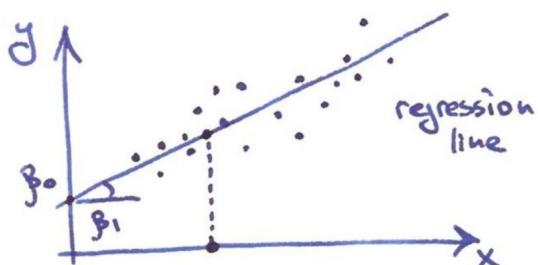
As n increases, MSE decreases

↳ error decreases as algorithm is gaining more experience.

RMSE = Root mean squared error:

$$L = \sqrt{\frac{1}{n} \sum (y_i - \hat{y}_i)^2}$$

Multiple linear Regression has > 1 predictors



The Model:

$$f(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p$$

It is convenient to define $x_0 \equiv 1$:

$$f(x) = \bar{\beta} \cdot \bar{x} =$$

The Loss and Gradient:

$$L(\beta) =$$

Partial Derivatives:

$$\frac{\partial L}{\partial \beta_k} = -\frac{2}{n} \sum_{i=1}^n (y^{(i)} - \beta^\top \mathbf{x}^{(i)}) x_k^{(i)}$$

Gradient:

$$\nabla L = -\frac{2}{n} \sum_{i=1}^n (y^{(i)} - \beta^\top \mathbf{x}^{(i)}) \mathbf{x}^{(i)}$$

Design Matrix:

$$\mathbf{X} = \begin{pmatrix} \mathbf{x}^{(1)} \\ \mathbf{x}^{(2)} \\ \vdots \\ \mathbf{x}^{(n)} \end{pmatrix} = \begin{pmatrix} x_0^{(1)} & x_1^{(1)} & \cdots & x_p^{(1)} \\ x_0^{(2)} & x_1^{(2)} & \cdots & x_p^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_0^{(n)} & x_1^{(n)} & \cdots & x_p^{(n)} \end{pmatrix} = (\mathbf{x}_0 \quad \mathbf{x}_1 \quad \cdots \quad \mathbf{x}_p)$$

Response/Observation vector:

$$\mathbf{y} = \begin{pmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(n)} \end{pmatrix}$$

Given \mathbf{X} and β :

$$\hat{\mathbf{y}} = \mathbf{X}\beta$$

We can state the loss function as:

$$\begin{aligned} L &= \frac{1}{n} \sum_{i=1}^n (y^{(i)} - \hat{y}_i)^2 = \frac{1}{n} \|\mathbf{y} - \mathbf{X}\hat{\beta}\|^2 \\ &= \frac{1}{n} (\mathbf{y} - \mathbf{X}\hat{\beta})^\top (\mathbf{y} - \mathbf{X}\hat{\beta}) \end{aligned}$$

Gradient:

$$\nabla L = -\frac{2}{n} \mathbf{X}^\top (\mathbf{y} - \mathbf{X}\hat{\beta}) = 0$$

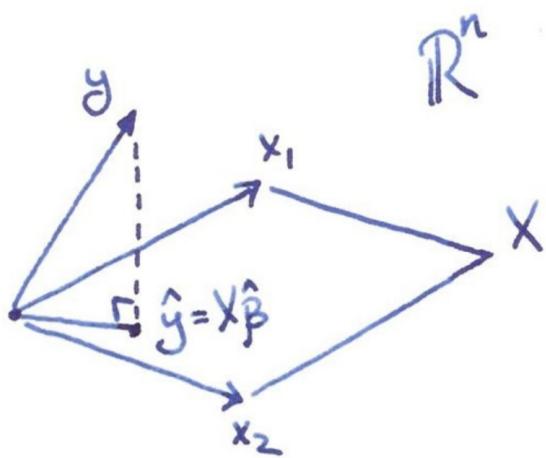
Matrix Algebra:

$$\begin{aligned} \textcircled{1} \quad \mathbf{A}\bar{x} &= \bar{b} \\ \bar{x} &= \mathbf{A}^{-1}\bar{b} \end{aligned}$$

$$\textcircled{2} \quad [\mathbf{A} \quad \mathbf{b}] \xrightarrow{\text{ref}} [\mathbf{I} \quad \bar{x}]$$

$\hat{y} = y\text{-hat}$:

$$\hat{y} = X \hat{\beta} =$$



The orthogonal projection of y onto the feature-space spanned by the features (x_1, x_2, \dots, x_p) in the n -dimensional space.

$$\|y - X \hat{\beta}\|^2 \text{ minimal} \Rightarrow \perp$$

Linear Regression boils down to orthogonal projections!

Performance of regression:

- 1) MSE
- 2) r^2
- 3) RMSE

Base Case:
- In classification: random guess
- In regression: guess the mean \bar{y}

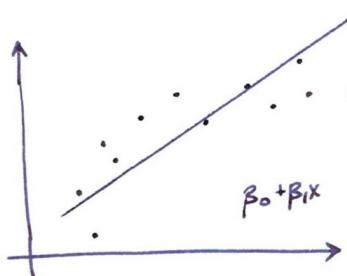
Polynomial regression:

Assume we model y and x and include x^2, x^3 etc \rightarrow Still linear model:

$$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 \dots$$

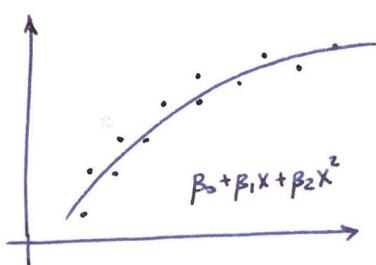
$$X = \begin{bmatrix} 1 & x_1^2 & x_1^3 & \dots \\ 1 & x_2^2 & x_2^3 & \dots \\ 1 & x_3^2 & x_3^3 & \dots \\ \vdots & \vdots & \vdots & \ddots \\ 1 & \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix}$$

Underfitting/Overfitting:



Under fitting
High Bias
Low Variance

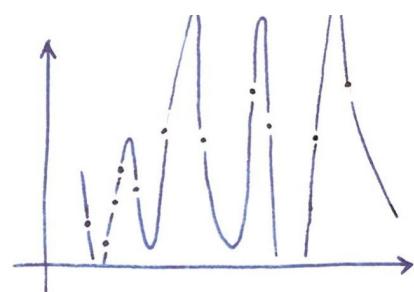
Bias: Inability to learn from data



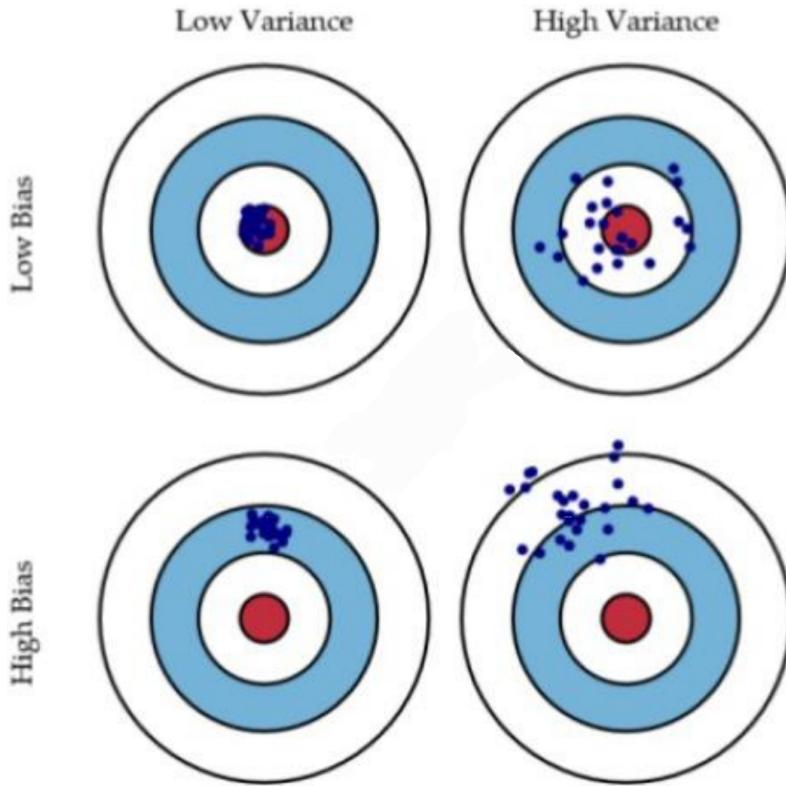
Overfitting
High Variance
Low Bias

Variance: rely too much on data

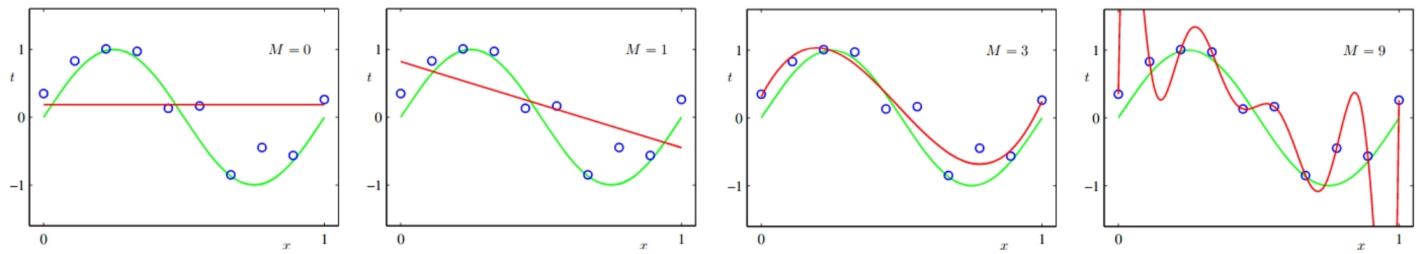
\rightarrow difference between data sets



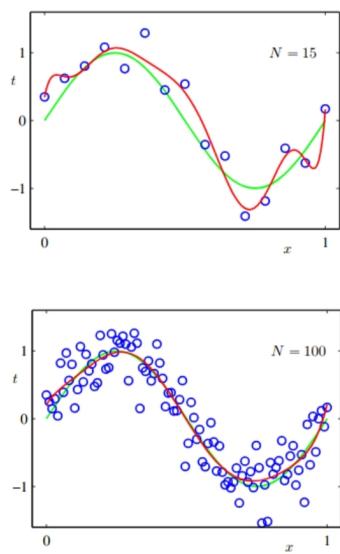
Bias-Variance Trade off



Overfitting and high Variance



	$M = 0$	$M = 1$	$M = 3$	$M = 9$
w_0^*	0.19	0.82	0.31	0.35
w_1^*		-1.27	7.99	232.37
w_2^*			-25.43	-5321.83
w_3^*			17.37	48568.31
w_4^*				-231639.30
w_5^*				640042.26
w_6^*				-1061800.52
w_7^*				1042400.18
w_8^*				-557682.99
w_9^*				125201.43



Bishop, *Pattern Recognition and Machine Learning*

How to avoid: Regularization

Idea: Penalize large coefficient

$$L = \frac{1}{n} \|y - X\beta\|^2$$

Here L is penalty term and
is called the regularisation Parameter

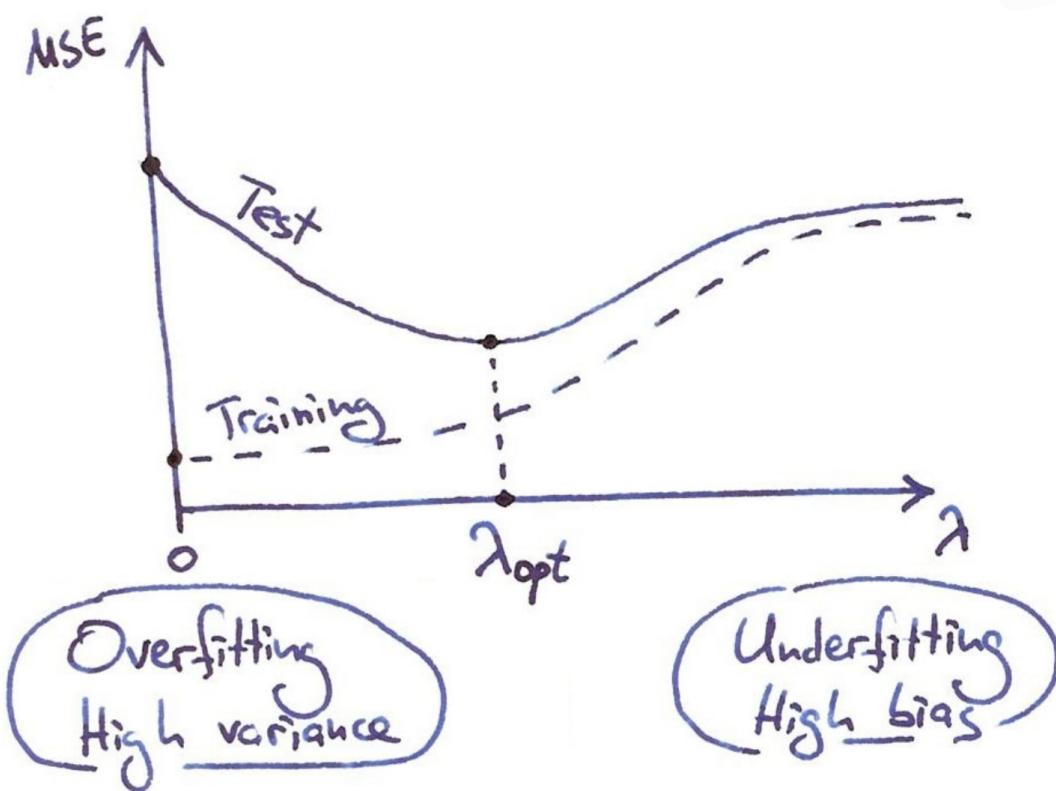
Common choices:

$$R(\beta) = \|\beta\|_2^2 = \sum \beta_i^2 \quad \text{ridge } L_2$$

$$R(\beta) = \|\beta\|_1 = \sum |\beta_i| \quad \text{lasso } L_1$$

$$R(\beta) = \lambda_1 \|\beta\|_1 + \lambda_2 \|\beta\|_2^2 = \lambda \cdot \sum (\alpha \cdot |\beta_i| + (1-\alpha) \beta_i^2) \quad \text{Elastic Net}$$

Loss function: $L = \frac{1}{n} \|y - X\beta\|^2 + \lambda R(\beta)$



Ridge Regression:

Loss function:

$$\mathcal{L} = \frac{1}{n} \|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda \|\beta\|^2.$$

Gradient:

$$\nabla \mathcal{L} = -\frac{2}{n} \mathbf{X}^\top (\mathbf{y} - \mathbf{X}\beta) + 2\lambda\beta.$$

Gradient descent:

$$\begin{aligned} \beta \leftarrow \beta - \eta \nabla \mathcal{L} &= \beta + \eta \frac{2}{n} \mathbf{X}^\top (\mathbf{y} - \mathbf{X}\beta) - 2\eta\lambda\beta = \\ &= \underbrace{(1 - 2\eta\lambda)}_{\text{"weight decay"}} \beta + \eta \frac{2}{n} \mathbf{X}^\top (\mathbf{y} - \mathbf{X}\beta). \end{aligned}$$

coefficients decrease

Analytic Solution:

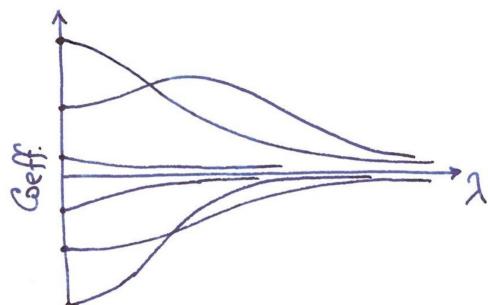
$$-\frac{2}{n} \mathbf{X}^\top (\mathbf{y} - \mathbf{X}\hat{\beta}) + 2\lambda\hat{\beta} = 0$$

$$\mathbf{X}^\top \mathbf{X}\hat{\beta} + n\lambda\hat{\beta} = \mathbf{X}^\top \mathbf{y}$$

$$(\mathbf{X}^\top \mathbf{X} + n\lambda\mathbf{I})\hat{\beta} = \mathbf{X}^\top \mathbf{y}$$

$$\hat{\beta} = (\mathbf{X}^\top \mathbf{X} + n\lambda\mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}$$

This is an example of a
Shrinkage estimator.



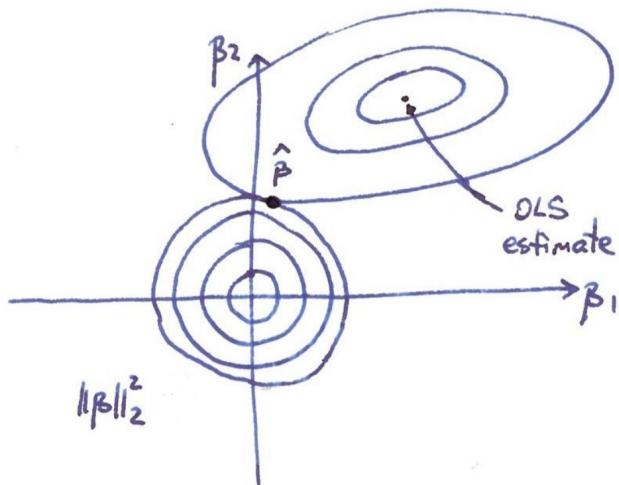
Lasso:

$$L = \frac{1}{n} \|y - X\beta\|^2 + \lambda \|\beta\|_1$$

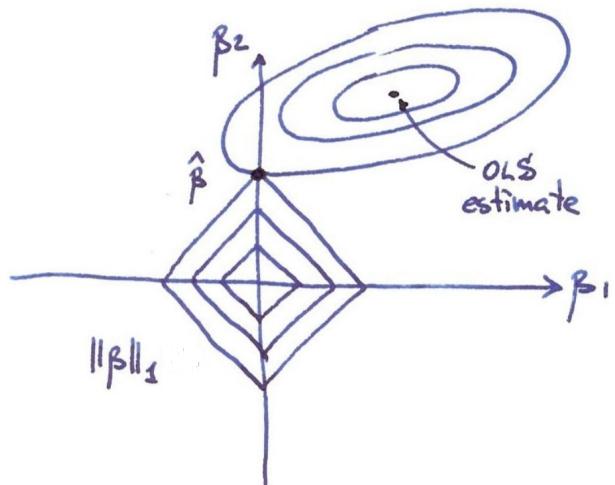
No analytic solution

→ Drive coefficient to zero (sparse)
↳ feature selection!

Ridge vs. Lasso



Ridge



Lasso

Elastic Net:

"Combines" ridge and lasso:

α : "how much" lasso vs. ridge

$$R(\beta) = \lambda_1 \|\beta\|_1 + \lambda_2 \|\beta\|_2^2 = \lambda \cdot \sum (\alpha \cdot \|\beta\|_1 + (1-\alpha) \|\beta\|^2)$$