ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

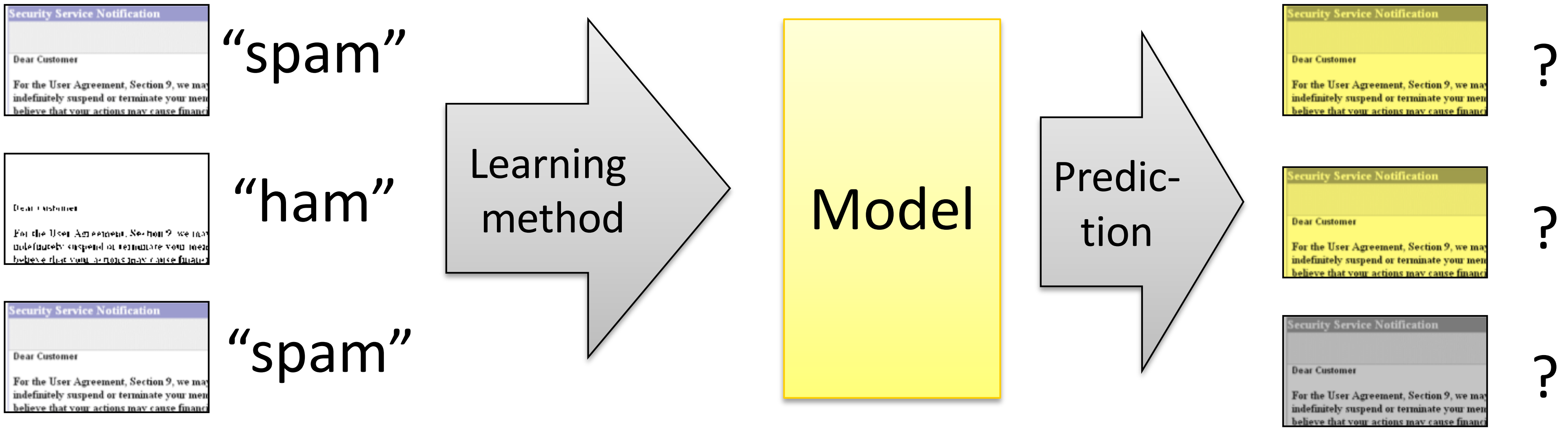
inf | Informatik Computer Science

Introduction to Machine Learning

Linear Regression

Prof. Andreas Krause
Learning and Adaptive Systems (las.ethz.ch)

Basic Supervised Learning Pipeline



Training Data

"spam"

"ham"

"spam"

Learning method

Model

Predic-
tion

Test Data

?

?

?

$\mathcal{X}$ $\mathcal{Y}$

$f : \mathcal{X} \to \mathcal{Y}$

Model fitting

Prediction/
Generalization

Regression

- Instance of supervised learning
- **Goal**: Predict real valued labels (possibly vectors)
- Examples:

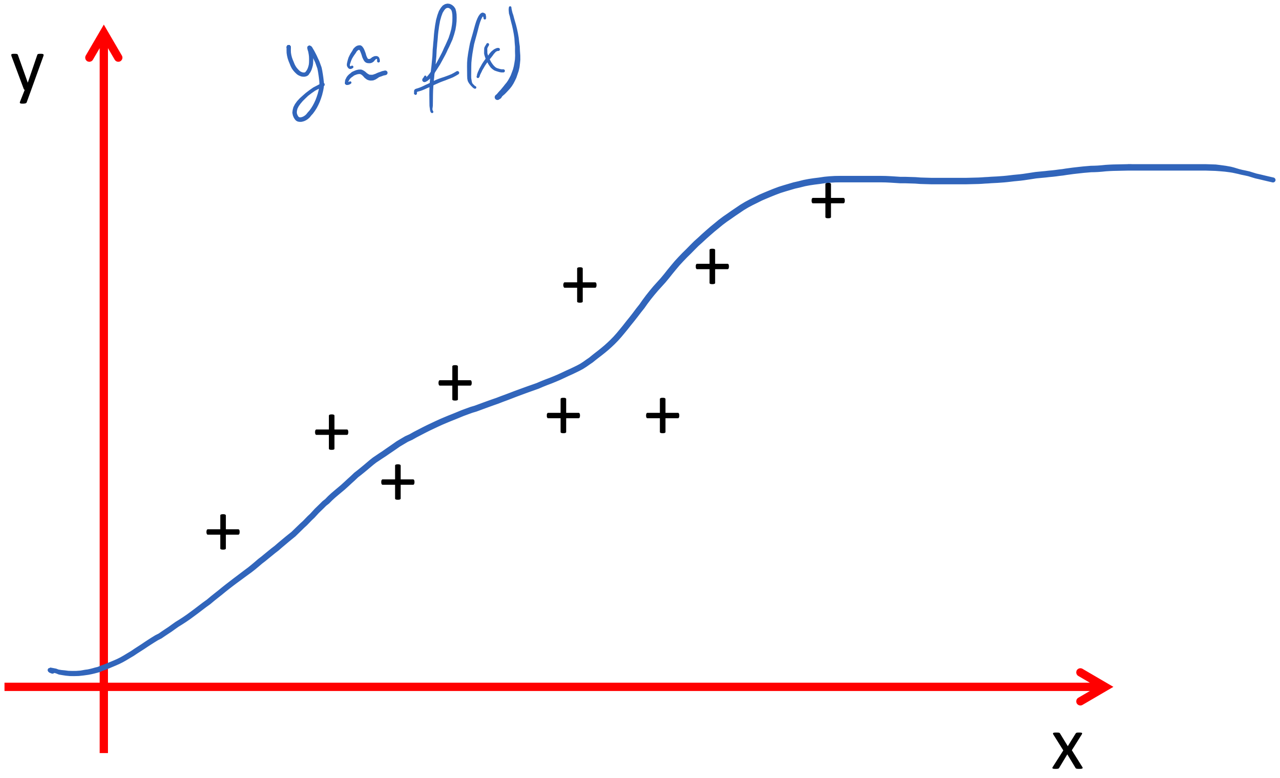
X	**Y**
Flight route	Delay (minutes)
Real estate objects	Price
Customer & ad features	Click-through probability

Running example: Diabetes

[Efron et al '04]

- Features X:
  - Age
  - Sex
  - Body mass index
  - Average blood pressure
  - Six blood serum measurements (S1-S6)
- Label (target) Y
  - quantitative measure of disease progression

Regression

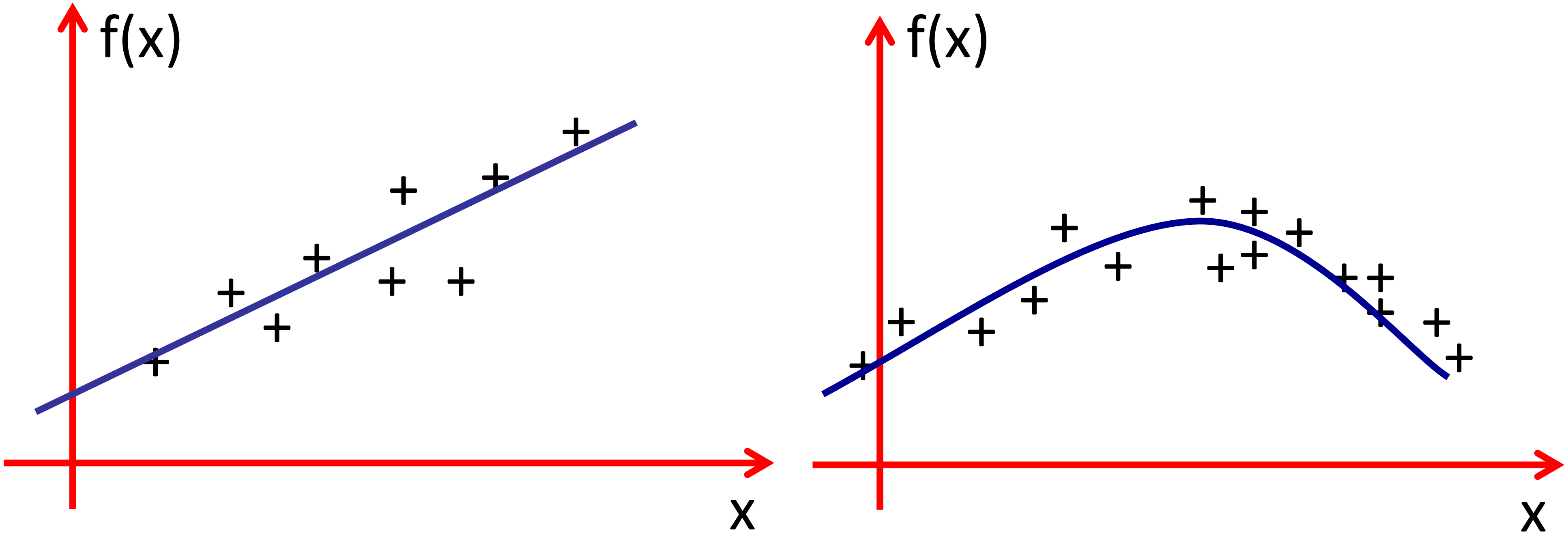


$y \approx f(x)$

- **Goal**: learn real valued mapping $f : \mathbb{R}^d \rightarrow \mathbb{R}$

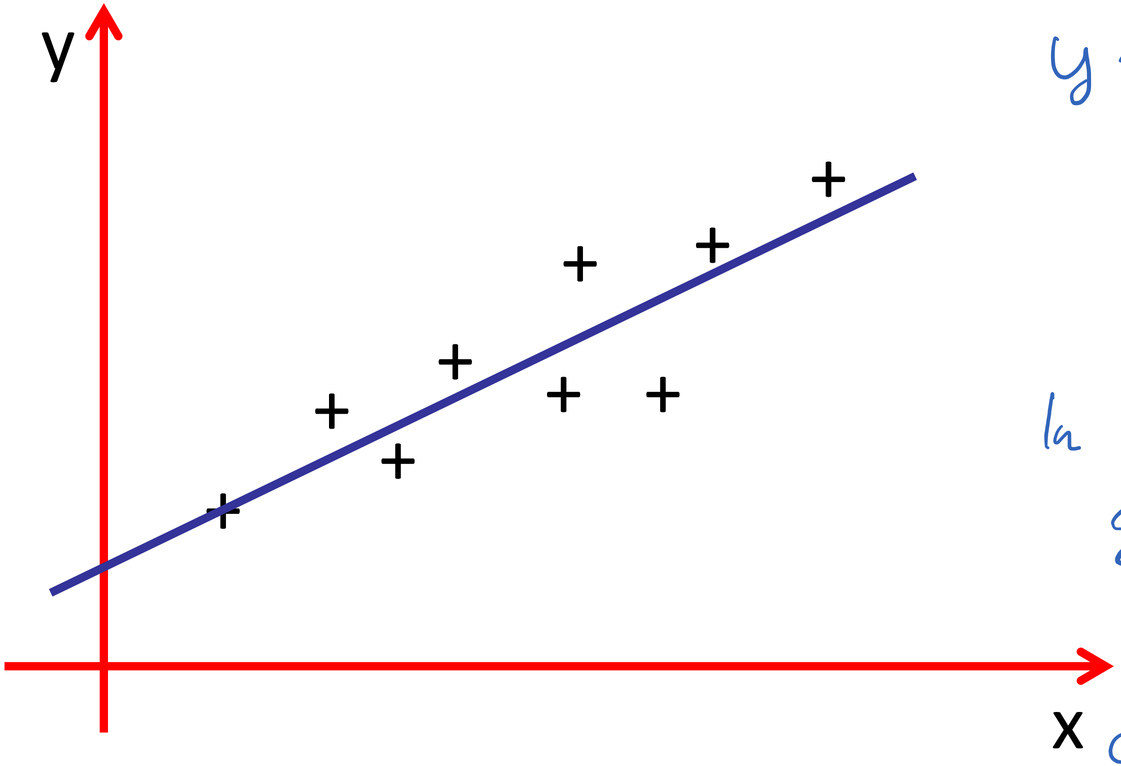
Important choices in regression

- What types of functions f should we consider? Examples



- How should we measure goodness of fit?

Example: linear regression



$y \approx f(x)$

$\uparrow$

f is linear (affine)

In 1-dim: $f(x) = ax + b$

2-dim: $f(x_1, x_2) = a x_1 + b x_2 + c$

d-dim: $f(x) = w_1 x_1 + \dots w_d \cdot x_d + w_0$

$$= \sum_{i=1}^{d} w_i \cdot x_i + w_0$$

$$= w^T x + w_0$$

$x = [x_1 \dots x_d]$

$w = [w_1 \dots w_d]$

Homogeneous representation

$$\underbrace{w^T x + w_0}_{\in \mathbb{R}} = \underbrace{\tilde{w}^T \tilde{x}}_{\in \mathbb{R}}$$

$\in \mathbb{R}^d$ $\qquad\qquad\qquad\qquad$ $\in \mathbb{R}^{d+1}$

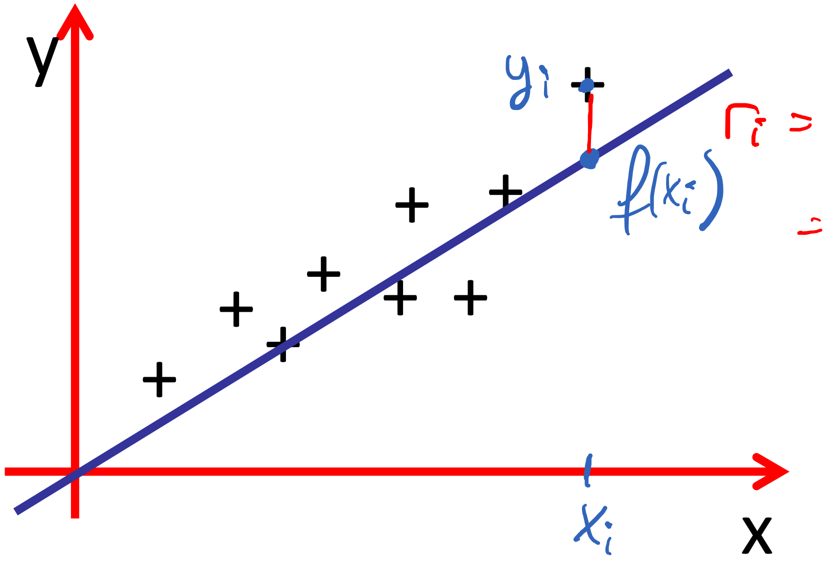
$$w = [w_1 \,..\, w_d] \qquad \curvearrowright \qquad \tilde{w} = [w_1 \dots v_d \;\; w_0]$$

$$x = [x_1 \,..\, x_d] \qquad\qquad \tilde{x} = [x_1 \,..\, x_d \;\; 1]$$

$$\Rightarrow \text{w.l.o.g.}: \quad f(x) = v^T x$$

Quantifying goodness of fit

$$D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \qquad \mathbf{x}_i \in \mathbb{R}^d \qquad y_i \in \mathbb{R}$$



Residuals

$r_i = y_i - f(x_i)$

$= y_i - w^T x_i$

$\rightarrow$ Cost $\hat{R}(w) = \sum_{i=1}^{n} r_i^2$

$= \sum_{i=1}^{n} (y_i - w^T x_i)^2$

Note: We've made 2 decisions

1) quantify error for 1 point via squared residual

2) we sum over all points

Least-squares linear regression optimization

[Legendre 1805, Gauss 1809]

- Given data set $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$

- How do we find the optimal weight vector?

$$\boxed{\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \sum_{i=1}^{n} (y_i - \mathbf{w}^T \mathbf{x}_i)^2}$$

Method 1: Closed form solution

- The problem $$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \sum_{i=1}^{n} (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

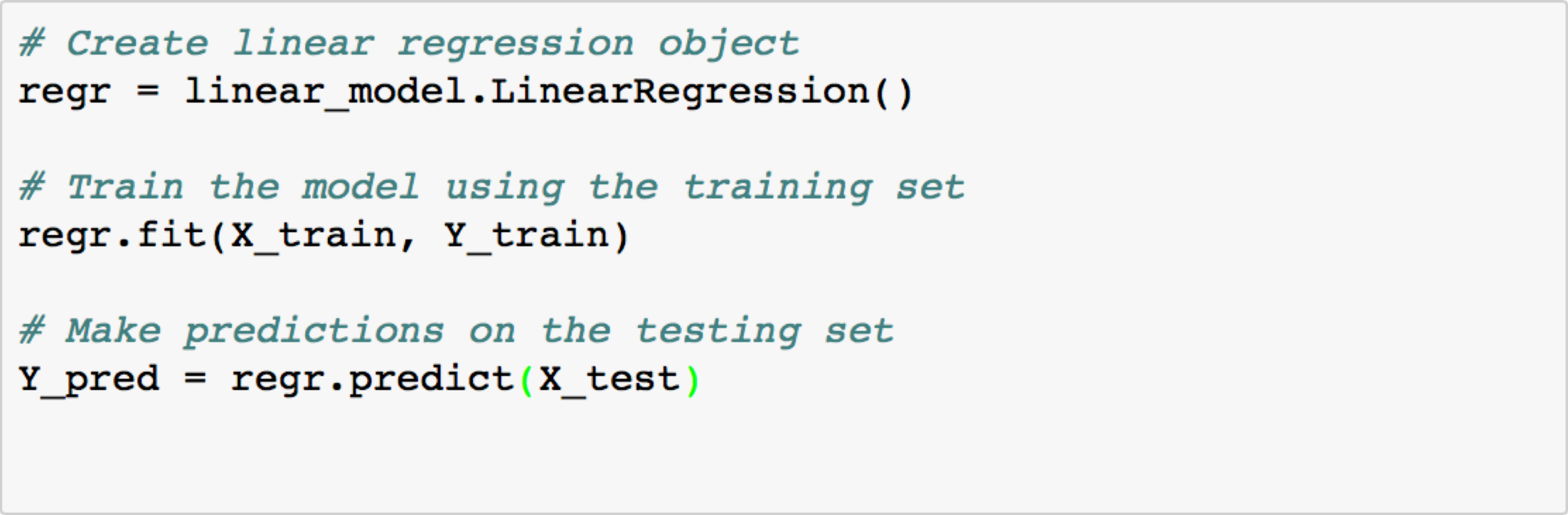
  can be solved in closed form:

$$\hat{\mathbf{w}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

- Hereby:

$$X = \left(\begin{array}{ccc} x_{1,1} & \cdots & x_{1,d} \\ \hline & \vdots & \\ \hline x_{n,1} & \cdots & x_{n,d} \end{array} \right) \qquad y \in \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

How to solve? Example: Scikit Learn

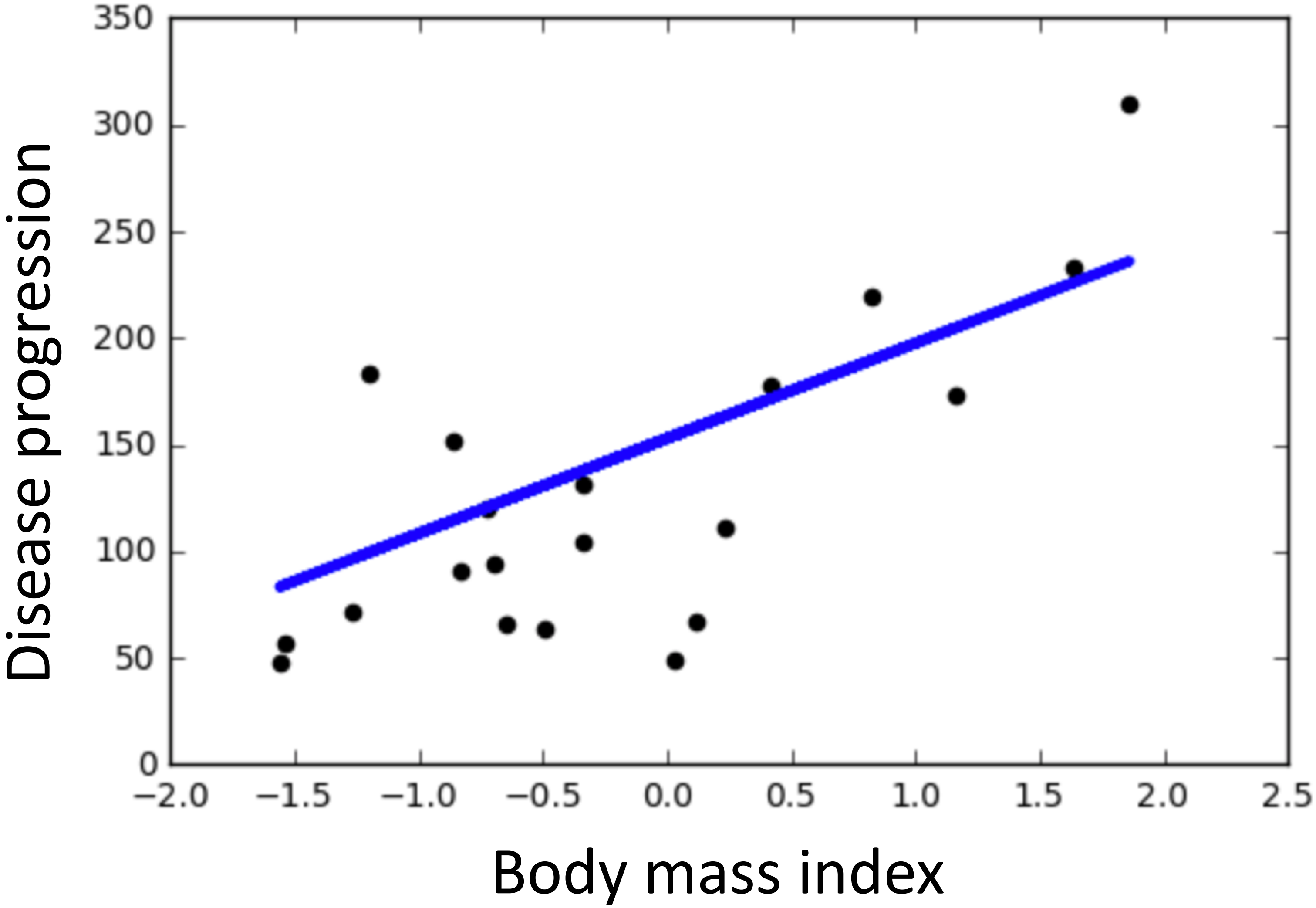


```
# Create linear regression object
regr = linear_model.LinearRegression()

# Train the model using the training set
regr.fit(X_train, Y_train)

# Make predictions on the testing set
Y_pred = regr.predict(X_test)
```

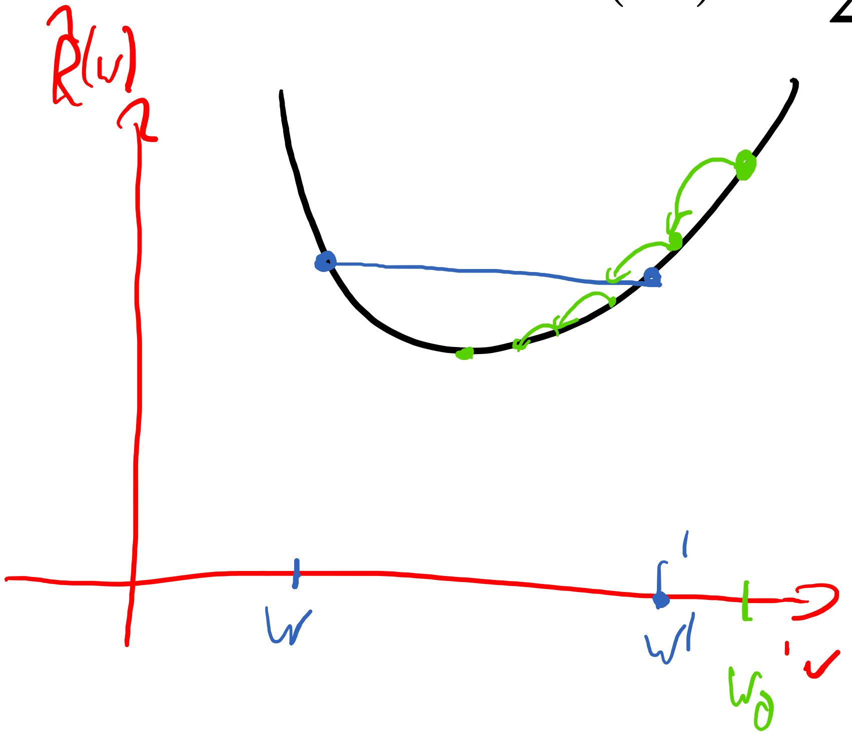
Demo

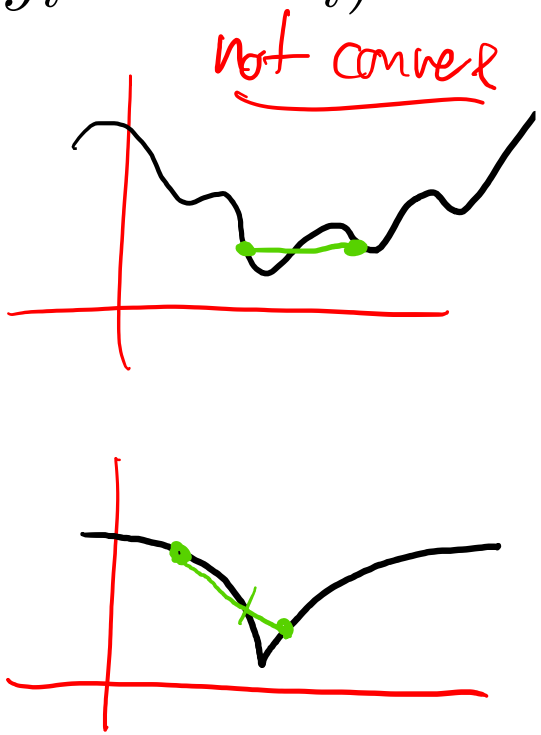


350
300
250
200
150
100
50
0
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5
Disease progression
Body mass index

Method 2: Optimization

- The objective function $\hat{R}(\mathbf{w}) = \sum_i (y_i - \mathbf{w}^T \mathbf{x}_i)^2$ is convex!





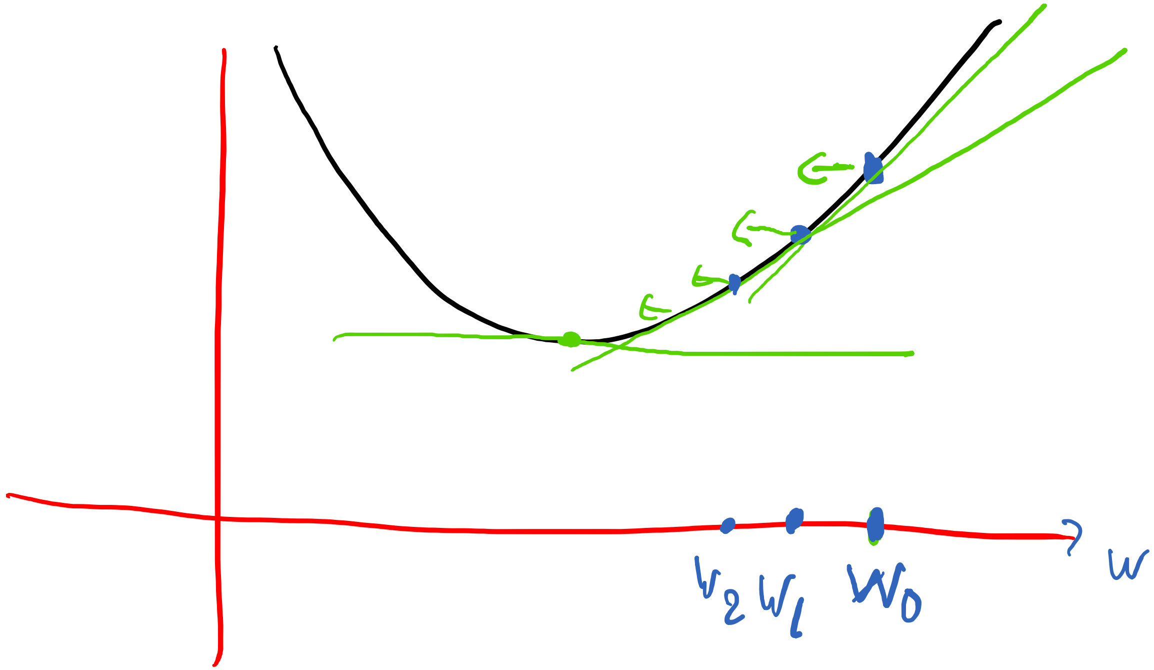
$f : \mathbb{R}^d \to \mathbb{R}$ convex iff $\forall x, x', \lambda \in [0,1]$ it holds that

$$f(\lambda x + (1-\lambda)x') \leq \lambda f(x) + (1-\lambda) f(x')$$

Gradient Descent

- Start at an arbitrary $\mathbf{w}_0 \in \mathbb{R}^d$
- For t=0,1,2,... do $\mathbf{w}_{t+1} = \mathbf{w}_t - \overset{\frac{1}{2}}{\eta_t} \nabla \hat{R}(\mathbf{w}_t)$

- Hereby, η_t is called learning rate



Convergence of gradient descent

- Under mild assumptions, if step size sufficiently small, gradient descent converges to a stationary point (gradient = 0)
- For convex objectives, it therefore finds the optimal solution!
- In the case of the squared loss, constant stepsize ½ converges linearly

$$\Rightarrow \exists t_0 \ \forall t \geq t_0 \ \exists \alpha < 1 \text{ s.t. } \overset{\leq \varepsilon}{\underbrace{\left(\hat{R}(w_{t+1}) - \overset{\text{optimal value}}{\hat{R}(\hat{w})}\right)}_{\text{"gap" at } t+1}} \leq \alpha \underbrace{\left(\hat{R}(w_t) - \hat{R}(\hat{w})\right)}_{\text{"gap" at } t}$$

$\Rightarrow$ can find $\underline{\varepsilon\text{-optimal}}$ solution in $O(\ln \frac{1}{\varepsilon})$ iter.

Computing the gradient

$$\nabla \hat{R}(w) = \left[\frac{\partial}{\partial w_1} \hat{R}(w) \quad \cdots \quad \frac{\partial}{\partial w_d} \hat{R}(w) \right]$$

In 1-dim: $\nabla \hat{R}(w) = \frac{d}{dw} \hat{R}(w) = \frac{d}{dw} \sum_{i=1}^{n} (y_i - w \cdot x_i)^2$

$$= \sum_{i=1}^{n} \frac{d}{dw} (y_i - w \cdot x_i)^2$$

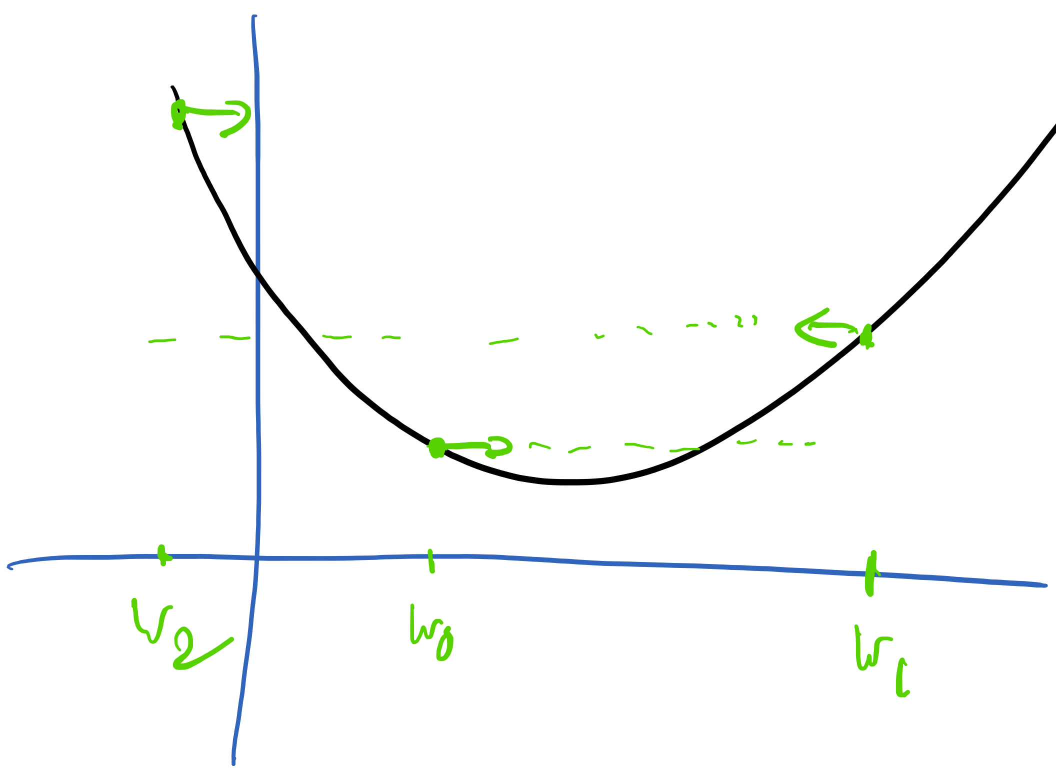
$$= \sum_{i=1}^{n} 2 \underbrace{(y_i - w \cdot x_i) \cdot}_{r_i} (-x_i) = -2 \sum_{i=1}^{n} r_i x_i$$

In d-dim: $\nabla \hat{R}(w) = \qquad = -2 \sum_{i=1}^{n} \overset{\in \mathbb{R}}{\overset{\downarrow}{r_i}} \underset{\in \mathbb{R}^d}{\underset{\uparrow}{x_i}}$

Demo: Gradient descent

Choosing a stepsize

- What happens if we choose a poor stepsize?

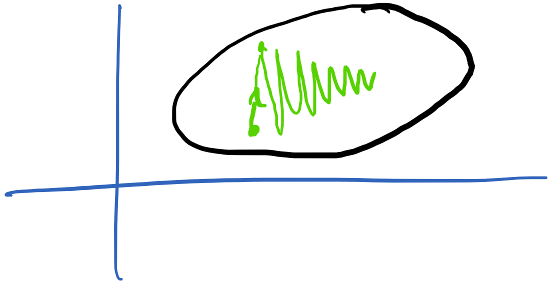


Adaptive step size

- Can update the step size adaptively. For example:
- 1) Via line search (optimizing step size every step)

Sps at iter t, have w_t, $g_t = \nabla \hat{R}(w_t)$

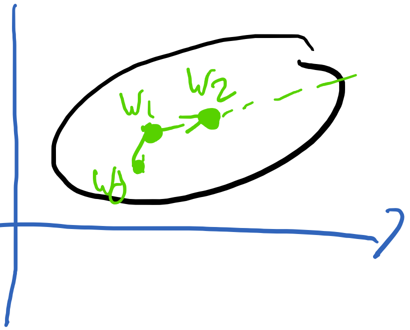
Define $\eta_t^* = \arg\min_{\eta \in [0,\infty)} \hat{R}(w_t - \eta g_t)$



- 2) „Bold driver“ heuristic
  - If function decreases, increase step size:

e.g. 1.1

If $\hat{R}(w_{t+1}) < \hat{R}(w_t)$: $\eta_{t+1} \leftarrow \eta_t \cdot c_{acc}$



  - If function increases, decrease step size:

e.g. 0.5

If $\hat{R}(w_{t+1}) > \hat{R}(w_t)$: $\eta_{t+1} \leftarrow \eta_t \cdot c_{dec}$

Demo: Gradient Descent for Linear Regression

Gradient descent vs closed form

- Why would one ever consider performing gradient descent, when it is possible to find closed form solution?

Closed form: $\hat{w} = (X^T X)^{-1} (X^T y)$

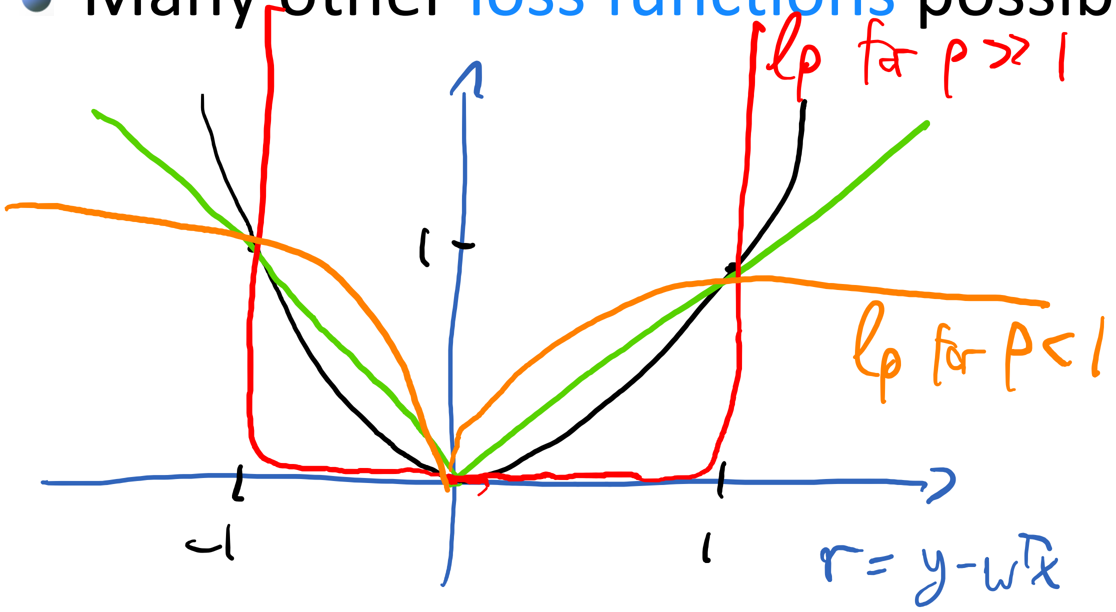
$O(nd^2)$ solve lin sys. $O(d^3)$

Gradient descent: Calc. $\nabla \hat{R}(w) = \sum_i \underbrace{(y_i - w^T x_i)}_{r_i} x_i \Rightarrow O(n \cdot d) \cdot \underbrace{\ln(\frac{1}{\varepsilon})}_{\#\text{iterations}}$

- Computational complexity
- May not need an optimal solution
- Many problems don‘t admit closed form solution

Other loss functions

- So far: Measure goodness of fit via squared error
- Many other loss functions possible (and sensible!)



least-squares

$$\ell_2(r) = r^2$$

Alternatives:

$$\ell_1(r) = |r|$$

$$\ell_p(r) = |r|^p$$

still convex for $p \geq 1$