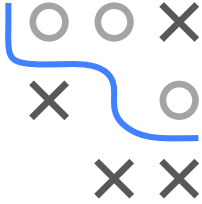
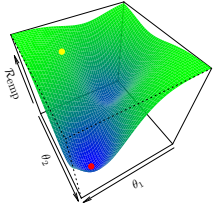
Introduction to Machine Learning



ML-Basics
Optimization

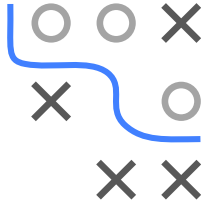


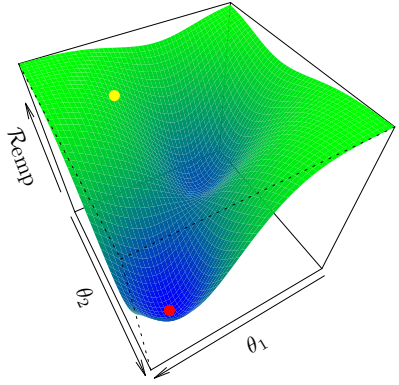
Learning goals

- Understand how the risk function is optimized to learn the optimal parameters of a model
- Understand the idea of gradient descent as a basic risk optimizer

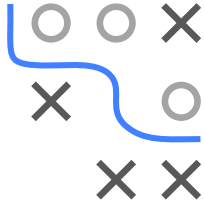
LEARNING AS PARAMETER OPTIMIZATION

- Operationalize search for model f that matches training data best by looking for parametrization $\boldsymbol{\theta} \in \Theta$ with lowest risk $\mathcal{R}_{\text{emp}}(\boldsymbol{\theta})$.
- Traverse error surface downwards; often local search from some start point to minimum (hopefully).





LEARNING AS PARAMETER OPTIMIZATION / 2



ERM optimization problem:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta} \in \Theta} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta})$$

For **(global) minimum** $\hat{\boldsymbol{\theta}}$:

$$\forall \boldsymbol{\theta} \in \Theta: \quad \mathcal{R}_{\text{emp}}(\hat{\boldsymbol{\theta}}) \leq \mathcal{R}_{\text{emp}}(\boldsymbol{\theta})$$

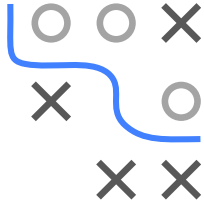
Does not imply that $\hat{\boldsymbol{\theta}}$ is unique.

- Best numerical optimizer depends on problem structure
- Continuous params? Uni-modal $\mathcal{R}_{\text{emp}}(\boldsymbol{\theta})$?
- Numerical optimization not our focus here, now

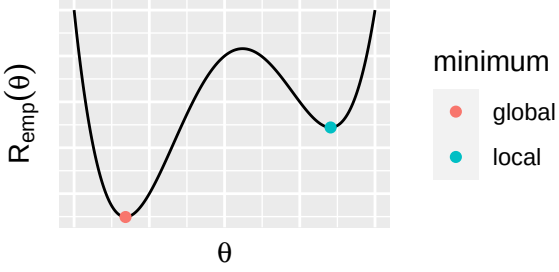
LOCAL MINIMA

Definition of **local minimum** $\hat{\boldsymbol{\theta}}$:

$$\exists \epsilon > 0 \; \forall \boldsymbol{\theta} \text{ with } \left\| \hat{\boldsymbol{\theta}} - \boldsymbol{\theta} \right\| < \epsilon : \quad \mathcal{R}_{\text{emp}}(\hat{\boldsymbol{\theta}}) \leq \mathcal{R}_{\text{emp}}(\boldsymbol{\theta})$$



Clearly every global minimum is also a local minimum.
Finding local minimum is easier than global one.

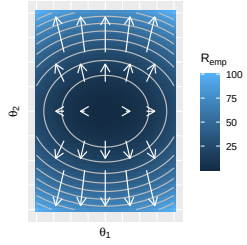


LOCAL MINIMA AND STATIONARY POINTS

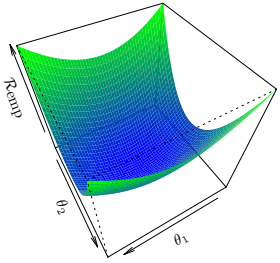
If $\mathcal{R}_{\text{emp}}$ continuously differentiable, **sufficient condition** for local minimum:
$\hat{\boldsymbol{\theta}}$ is **stationary**, so 0 gradient, so no local improvement possible:

$$\frac{d\mathcal{R}_{\text{emp}}}{d\boldsymbol{\theta}}(\hat{\boldsymbol{\theta}}) = 0$$

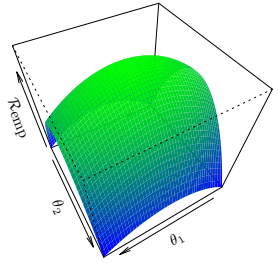
and Hessian at $\hat{\boldsymbol{\theta}}$ is positive definite.
Neg. gradient points into direction of fastest local decrease;
Hessian measures local curvature.



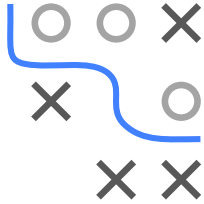
$\frac{d\mathcal{R}_{\text{emp}}}{d\boldsymbol{\theta}}(\boldsymbol{\theta})$



const. pos. def. Hessian

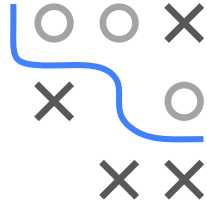


const. neg. def. Hessian

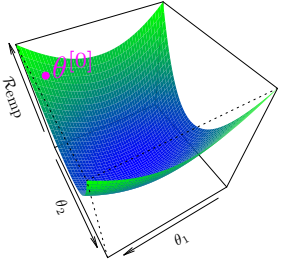


GRADIENT DESCENT

- Iteratively improve current candidate $\boldsymbol{\theta}^{[t]}$
- Move in direction of neg. gradient, so direction of steepest descent
- Use step size / learning rate α

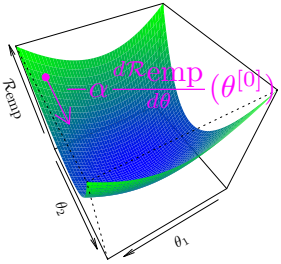


$$\boldsymbol{\theta}^{[t+1]} = \boldsymbol{\theta}^{[t]} - \alpha \frac{d\mathcal{R}_{\text{emp}}}{d\boldsymbol{\theta}}(\boldsymbol{\theta}^{[t]})$$

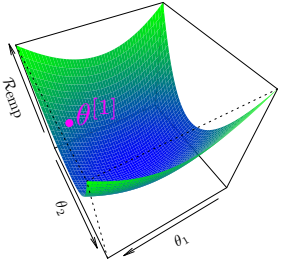


Random start $\boldsymbol{\theta}^{[0]}$ with $\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}^{[0]}) = 76.25$.

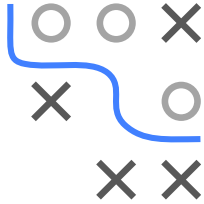
GRADIENT DESCENT - EXAMPLE



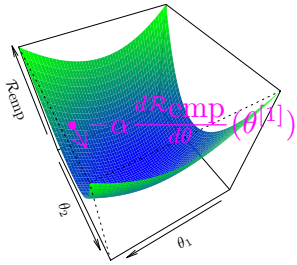
Direction of the neg. gradient at $\boldsymbol{\theta}^{[0]}$



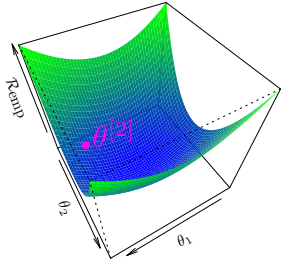
Arrive at $\boldsymbol{\theta}^{[1]}$ with $\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}^{[1]}) \approx 42.73$.
We improved:
$\mathcal{R}_{\text{emp}}(\boldsymbol{\theta}^{[1]}) < \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}^{[0]})$



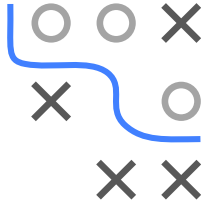
GRADIENT DESCENT - EXAMPLE



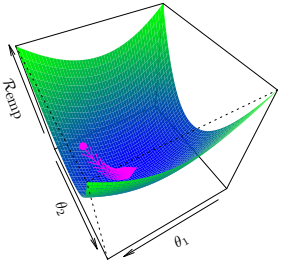
Now iterate, do the same at $\theta^{[1]}$



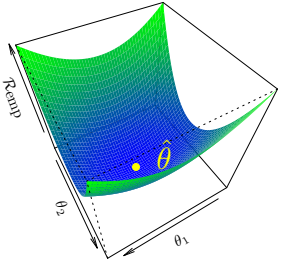
Now $\theta^{[2]}$ has risk $\mathcal{R}_{\text{emp}}(\theta^{[2]}) \approx 25.08$



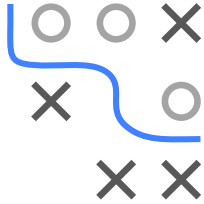
GRADIENT DESCENT - EXAMPLE



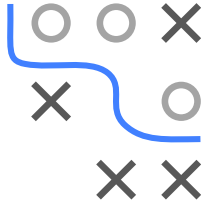
We iterate this until some form of convergence or termination



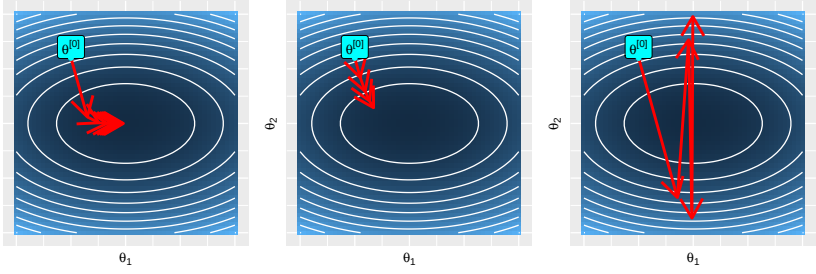
We arrive close to a stationary $\hat{\boldsymbol{\theta}}$ which is hopefully at least a local minimum



GRADIENT DESCENT - LEARNING RATE



- Neg. gradient is direction that looks locally promising to reduce $\mathcal{R}_{\text{emp}}$.
- Hence weights components higher in which $\mathcal{R}_{\text{emp}}$ decreases more.
- However, the length of $-\frac{d}{d\boldsymbol{\theta}}\mathcal{R}_{\text{emp}}$ measures only the local decrease rate, i.e., there are no guarantees that we will not go "too far".
- We use a learning rate α to scale the step length in each iteration. Too much can lead to overstepping and no convergence, too low leads to slow convergence.
- Usually, a simple constant rate or rate-decrease mechanisms to enforce local convergence are used.



good convergence for α_1 slow convergence for $\alpha_2 (< \alpha_1)$ no convergence for $\alpha_3 (> \alpha_1)$

FURTHER TOPICS

- Few models, e.g. linear regression, can be optimized analytically.
- GD is a so-called first-order method. Second-order methods (like Newton-Raphson) use the Hessian to refine the search direction for faster convergence.
- There exist many improvements of GD (momentum, ADAM), e.g., to smartly control the learn rate, to escape saddle points, to mimic second order behavior without computing the Hessian.
- If the gradient is not computed on the complete data, but instead on small, random batches, this is **stochastic gradient descent** (SGD). For large-scale problems, this is usually more efficient.