

Midterm Exam 2 Review

Linear Regression (Closed form)

$$\mathcal{X} = \mathbb{R}^{d+1}, \mathcal{Y} = \mathbb{R} \text{ regression}$$

$$\mathcal{F} \subset \{f \mid f: \mathbb{R}^{d+1} \rightarrow \mathbb{R}\}$$

$$\mathcal{D} = ((\vec{x}_1, y_1), \dots, (\vec{x}_n, y_n))$$

$$\hat{L}(f) = \frac{1}{n} \sum_{i=1}^n \ell(f(\vec{x}_i), y_i) \quad \begin{array}{l} \text{an estimate of } L(f) \\ \text{for all } f \in \mathcal{F} \end{array}$$

we want

$$\hat{f} = \operatorname{argmin}_{f \in \mathcal{F}} \hat{L}(f)$$

pick $\mathcal{F} = \{f \mid f: \mathbb{R}^{d+1} \rightarrow \mathbb{R} \text{ and } f(\vec{x}) = \vec{x}^T \vec{w} \text{ where } \vec{w} \in \mathbb{R}^{d+1}\}$

Learner

$$\mathcal{A}(\mathcal{D}) = \hat{f} \in \mathcal{F}$$

$$\text{where } \hat{f}(\vec{x}) = \vec{x}^T \hat{\vec{w}} \quad \text{and} \quad \hat{\vec{w}} = \underbrace{A^{-1}}_{\substack{\text{Depends on } \mathcal{D} \\ \downarrow}} \vec{b}$$

Algorithm: Closed form linear regression Learner

input: $D = ((\vec{x}_1, y_1), \dots, (\vec{x}_n, y_n))$

$$A \leftarrow \sum_{i=1}^n \vec{x}_i \vec{x}_i^T$$

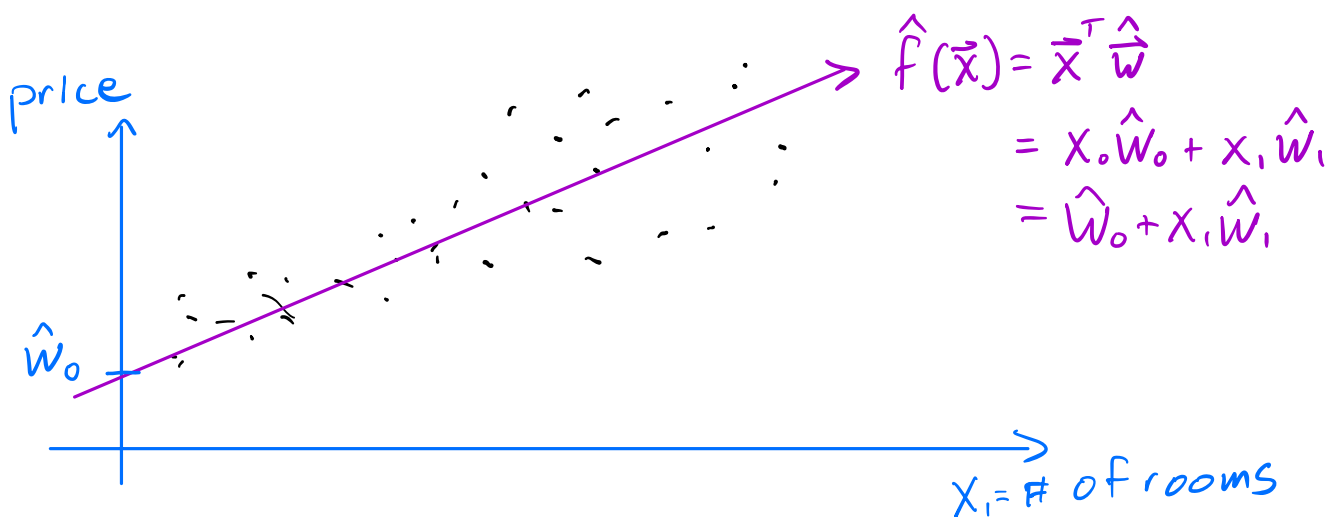
$$b \leftarrow \sum_{i=1}^n \vec{x}_i y_i$$

$$\hat{\vec{w}} \leftarrow A^{-1} b$$

return $\hat{f}(\vec{x}) = \vec{x}^T \hat{\vec{w}}$

Ex: $d=1$, $\mathcal{X} = \mathbb{R}^{1+1} = \mathbb{R}^2$, $\mathcal{Y} = \mathbb{R}$, $\hat{\vec{w}} = (\hat{w}_0, \hat{w}_1)^T$

$$D = ((\vec{x}_1, y_1), \dots, (\vec{x}_n, y_n))$$



Gradient Descent

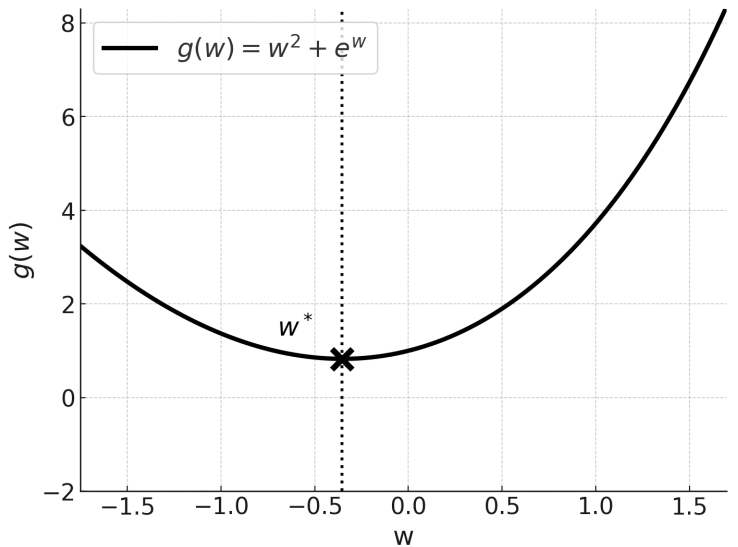
Ex: $g(w) = w^2 + e^w$, $g'(w) = 2w + e^w$, $g''(w) = 2 + e^w \geq 0$
Convex

$$w \in \mathbb{R} = \mathcal{W}$$

$$g'(w) = 2w + e^w = 0$$

$$\underbrace{2w = -e^w}$$

No closed form
solution



Second-order Gradient Descent:

$$g(w) \approx g_{w^{(0)}}(w) = g(w^{(0)}) + g'(w^{(0)})(w - w^{(0)}) + \frac{g''(w^{(0)})}{2} (w - w^{(0)})^2$$

$$g'_{w^{(0)}}(w) = g'(w^{(0)}) + g''(w^{(0)})(w - w^{(0)}) = 0$$

$$\Rightarrow w = w^{(0)} - \frac{g'(w^{(0)})}{g''(w^{(0)})}$$

General: $w^{(t+1)} = w^{(t)} - \frac{g'(w^{(t)})}{g''(w^{(t)})}$

$w^{(t+1)}$ approaches $w^* = \arg \min_{w \in \mathcal{W}} g(w)$
as $t \rightarrow \infty$

First-order Gradient Descent

$$w^{(t+1)} = w^{(t)} - \eta^{(t)} g'(w^{(t)})$$

Multivariate Gradient Descent

Objective:

$$\vec{w}^* = (w_1^*, \dots, w_d^*)^T = \arg \min_{\vec{w} \in \mathcal{W}} g(\vec{w})$$

gradient descent update rule:

$$\vec{w}^{(t+1)} = \vec{w}^{(t)} - \eta^{(t)} \nabla g(\vec{w}^{(t)})$$

$$\text{where } \nabla g(\vec{w}^{(t)}) = \left(\frac{\partial g}{\partial w_1}(\vec{w}^{(t)}), \dots, \frac{\partial g}{\partial w_d}(\vec{w}^{(t)}) \right)^T \in \mathbb{R}^d$$

Selecting the step size

1. constant: $\eta^{(t)} = \eta \in (0, \infty)$

2. Inverse decaying: $\eta^{(t)} = \frac{\eta}{1+\lambda t}$ where $\eta, \lambda \in (0, \infty)$

3. Exponential decaying: $\eta^{(t)} = \eta \frac{1}{e^{\lambda t}}$ where $\eta, \lambda \in (0, \infty)$

4. Normalized gradient: $\eta^{(t)} = \frac{\eta}{\epsilon + \|\nabla g(\vec{w}^{(t)})\|}$ where $\eta, \epsilon \in (0, \infty)$
 ϵ is small
(ex: $\epsilon = 10^{-8}$)

$$\|\nabla g(\vec{w}^{(t)})\| = \sqrt{\sum_{j=1}^d \left(\frac{\partial g}{\partial w_j}(\vec{w}^{(t)}) \right)^2}$$

(Batch) Gradient Descent (BGD)

$$\vec{w}^{(t+1)} = \vec{w}^{(t)} - \eta^{(t)} \nabla \hat{L}(\vec{w}^{(t)})$$

$$\nabla \hat{L}(\vec{w}^{(t)}) = \frac{2}{n} \sum_{i=1}^n (\vec{x}_i^T \vec{w} - y_i) \vec{x}_i$$

Algorithm: BGD Linear Regression Learner
(with a constant size)

input: $D = ((\vec{x}_1, y_1), \dots, (\vec{x}_n, y_n))$, η , T number of
"epochs"

$\vec{w} \leftarrow$ random vector in $\mathbb{R}^{d+1}$

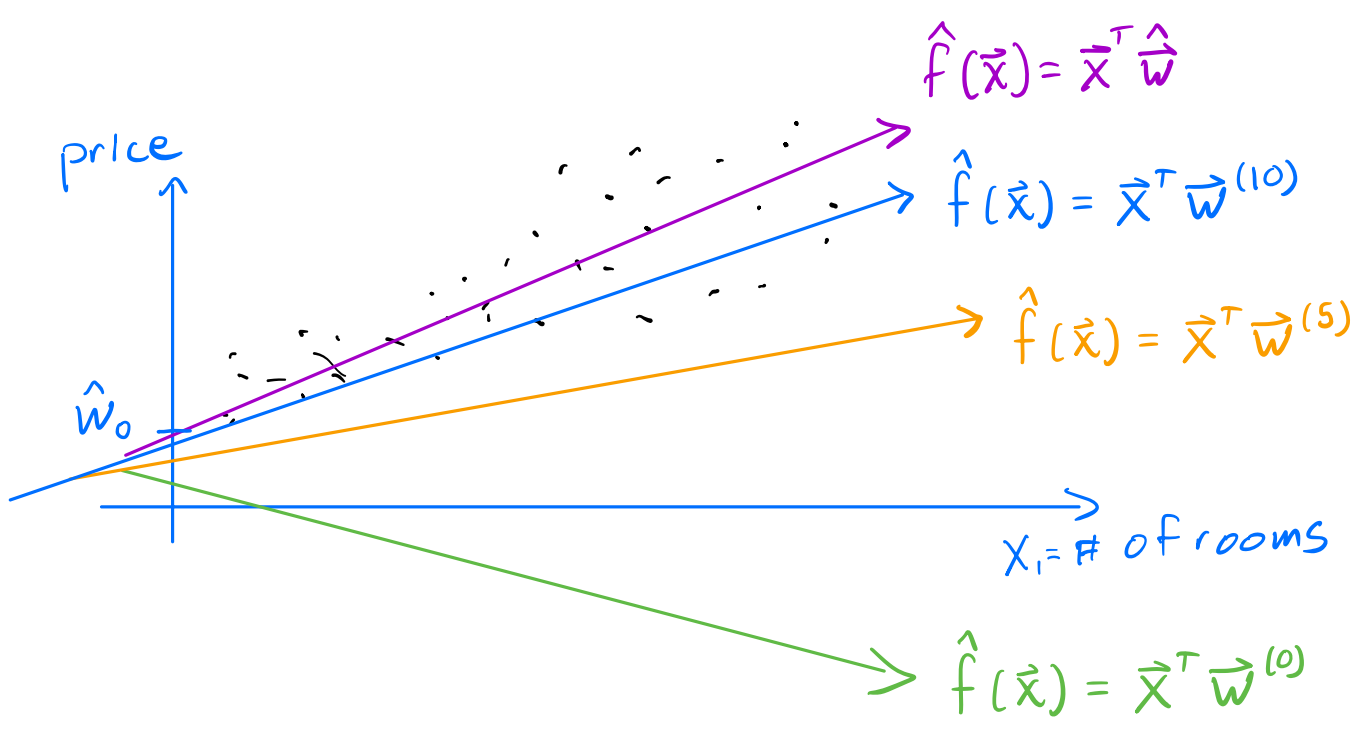
for $t = 1, \dots, T$

$$\nabla \hat{L}(\vec{w}) \leftarrow \frac{2}{n} \sum_{i=1}^n (\vec{x}_i^T \vec{w} - y_i) \vec{x}_i$$

$$\vec{w} = \vec{w} - \eta \nabla \hat{L}(\vec{w})$$

return $\hat{f}(\vec{x}) = \vec{x}^T \vec{w}$

$\hat{w}_0$



Computation

Closed form: $O(nd^2 + d^3)$

BGD: $O(ndT)$

if $T < d$: $O(ndT) < O(nd^2) \leq O(nd^2 + d^3)$

BGD is more computationally efficient
if $T < d$

Mini-Batch Gradient Descent (MBGD)

$$D = \left((\vec{x}_1, y_1), \dots, (\vec{x}_b, y_b), \right. \\ (\vec{x}_{b+1}, y_{b+1}), \dots, (\vec{x}_{2b}, y_{2b}), \\ \vdots \\ \left. (\vec{x}_{(M-1)b+1}, y_{(M-1)b+1}), \dots, (\vec{x}_{Mb}, y_{Mb}) \right)$$

b : mini-batch size

$M = \text{floor}\left(\frac{n}{b}\right)$: number of mini batches

we don't use $n - Mb \leq b$ datapoints

$$\hat{L}_m(\vec{w}) = \frac{1}{b} \sum_{i=(m-1)b+1}^{mb} (\vec{x}_i^T \vec{w} - y_i)^2 \quad m \in \{1, \dots, M\}$$

Algorithm: MBbD Linear Regression Learner
(with a constant size)

input: $D = ((\vec{x}_1, y_1), \dots, (\vec{x}_n, y_n))$, η , T , b

$\vec{w} \leftarrow$ random vector in $\mathbb{R}^{d+1}$

$M \leftarrow \text{floor}(\frac{n}{b})$

for $t = 1, \dots, T$

Randomly shuffle D

for $m = 1, \dots, M$

$$\nabla \hat{L}(\vec{w}) \leftarrow \frac{2}{b} \sum_{i=(m-1)b+1}^{mb} (\vec{x}_i^T \vec{w} - y_i) \vec{x}_i$$

$$\vec{w} = \vec{w} - \eta \nabla \hat{L}(\vec{w})$$

return $\hat{f}(\vec{x}) = \vec{x}^T \vec{w}$

Advantage is MT is now the number of gradient steps

Setting $b=n \Rightarrow M=1$ gives back BGD

$b=1 \Rightarrow M=n$ gives

"Stochastic GD (SGD)"

Polynomial Regression

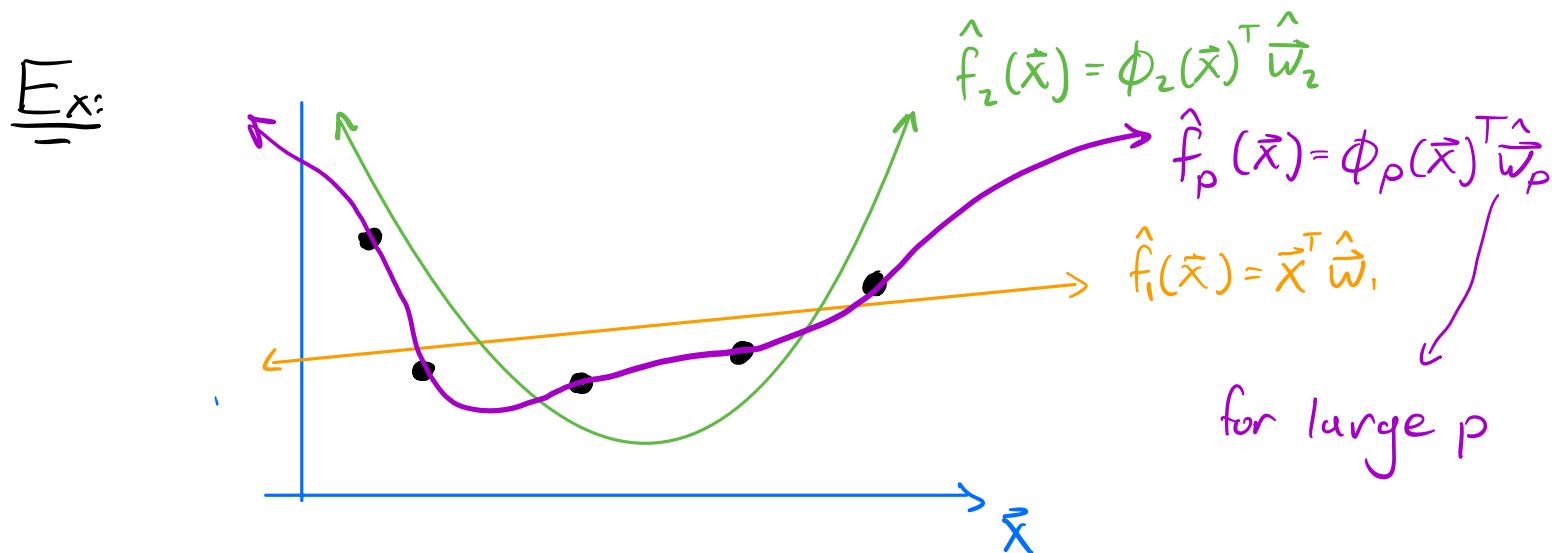
$\phi_p: \mathcal{X} \rightarrow \mathbb{Z}$ is a degree p polynomial "feature map"

For $\mathcal{X} = \mathbb{R}^{d+1}$, $\mathbb{Z} = \mathbb{R}^{\bar{p}}$ where $\bar{p} = \binom{(d+1)+p-1}{p} = \binom{d+p}{p}$

$$\tilde{\mathcal{F}}_p = \left\{ f \mid f: \mathbb{R}^{d+1} \rightarrow \mathbb{R} \text{ and } f(\bar{x}) = \phi_p(\bar{x})^T \bar{w}, \bar{w} \in \mathbb{R}^{\bar{p}} \right\}$$

$$\tilde{\mathcal{F}}_1 \subsetneq \tilde{\mathcal{F}}_2 \subsetneq \dots \subsetneq \tilde{\mathcal{F}}_p$$

$$\hat{L}(\hat{f}_1) \geq \hat{L}(\hat{f}_2) \geq \dots \geq \hat{L}(\hat{f}_p) \approx 0$$



Evaluating Predictors/Models

Objective (formal):

Define a Learner $\mathcal{A}: (\mathcal{X} \times \mathcal{Y})^n \rightarrow \{f \mid f: \mathcal{X} \rightarrow \mathcal{Y}\}$
such that $\mathbb{E}[L(\mathcal{A}(D))]$ is small

Defining $\mathcal{A}(D)$: Empirical Risk Minimization (ERM)

Estimation:

Use D to estimate $L(f)$ for all $f \in \mathcal{F} \subset \{f \mid f: \mathcal{X} \rightarrow \mathcal{Y}\}$
call the estimate $\hat{L}(f)$

Optimization:

pick $\hat{f}$ to be the $f \in \mathcal{F}$ that minimizes $\hat{L}(f)$

Function class

When should we expect ERM to work well?

- When $\mathcal{F}$ contains an f that can make $L(f)$ small
- When $\hat{L}(\hat{f})$ is a good estimate of $L(\hat{f})$

If $A(D) = f_0$ depends on D then

$$\mathbb{E}[\hat{L}(\hat{f}_0)] \neq \mathbb{E}[L(\hat{f}_0)]$$

$l(\hat{f}_0(\vec{X}_i), Y_i)$ are not i.i.d.

$\hat{f}_0$ depends on $(\vec{X}_1, Y_1), \dots, (\vec{X}_n, Y_n)$!

It can be shown that:

$$\mathbb{E}[L(\hat{f}_0)] - \mathbb{E}[\hat{L}(\hat{f}_0)]$$

- increases as $\mathcal{F}$ gets more complex
- decreases as n increases

Decomposing $E[L(A(D))]$

$$\text{Let } A(D) = \hat{f}_D$$

$\mathcal{F}$ any function class

$$f_{\text{Bayes}} = \argmin_{f \in \{f | f: x \rightarrow y\}} L(f)$$

$$f^* = \argmin_{f \in \mathcal{F}} L(f)$$

$$\hat{f}_D = \argmin_{f \in \mathcal{F}} \hat{L}(f)$$

$$E[L(\hat{f}_D)] = \underbrace{E[L(\hat{f}_D)] - L(f^*)}_{\text{Estimation Error (EE)}} + \underbrace{L(f^*) - L(f_{\text{Bayes}})}_{\text{Approximation Error (AE)}} + \underbrace{L(f_{\text{Bayes}})}_{\text{Irreducible Error (IE)}}$$

Irreducible Error: Due to inherent noise in labels

- Decreases if you gather more/better feature info
- Usually not possible to do "irreducible"

Approximation Error: Due to a small $\mathcal{F}$

- Decreases if you make $\mathcal{F}$ larger

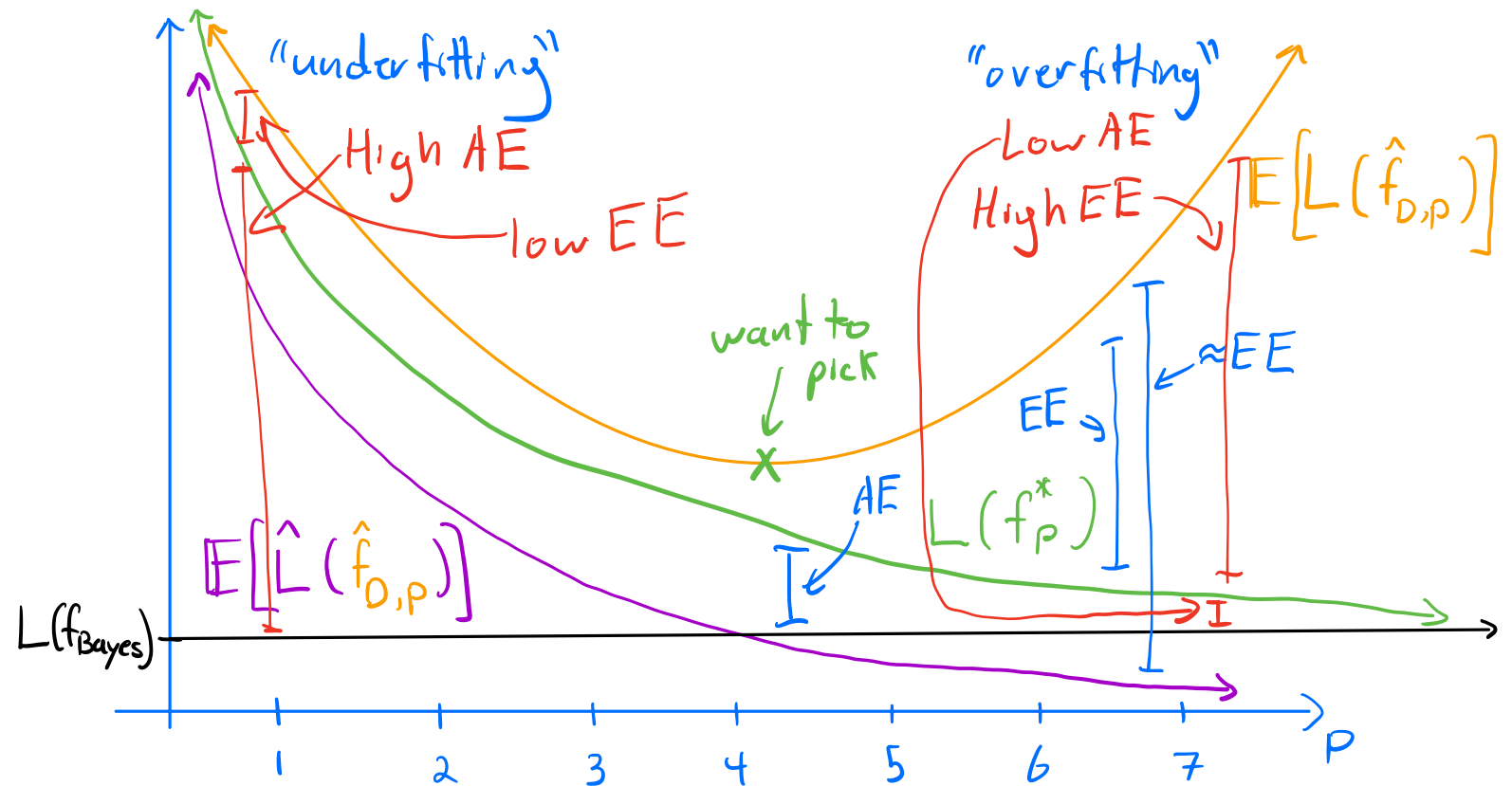
Estimation Error: Due to random dataset D

- Decreases if you increase n
- Increases if you increase $\mathcal{F}$

$$\mathcal{A}(D) = \hat{f}_{D,p} = \arg\min_{f \in \tilde{F}_p} \hat{L}(f) \quad \tilde{F}_1 \subset \dots \subset \tilde{F}_p$$

$$E[L(\hat{f}_0)] - E[\hat{L}(\hat{f}_0)]$$

$$E[L(\hat{f}_0)] = \underbrace{E[L(\hat{f}_0)] - L(f^*)}_{\text{Estimation Error (EE)}} + \underbrace{L(f^*) - L(f_{\text{Bayes}})}_{\text{Approximation Error (AE)}} + \underbrace{L(f_{\text{Bayes}})}_{\text{Irreducible Error (IE)}}$$



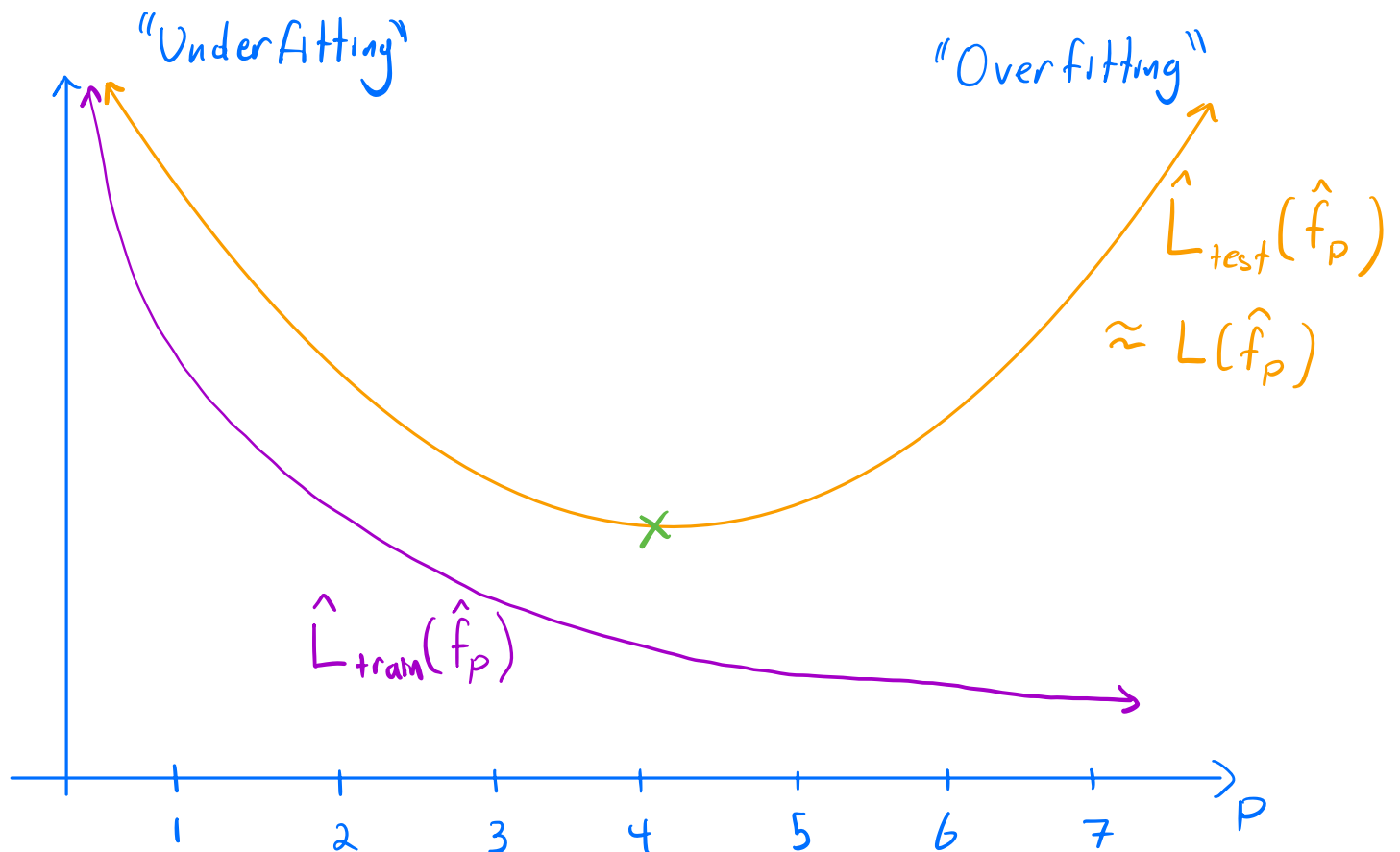
Can't calculate L , so use train, test split

$$D_{\text{train}} = ((\vec{x}_1, y_1), \dots, (\vec{x}_{n-m}, y_{n-m}))$$

$$D_{\text{test}} = ((\vec{x}_{n-m+1}, y_{n-m+1}), \dots, (\vec{x}_n, y_n))$$

$$|D_{\text{train}}| = n-m, |D_{\text{test}}| = m$$

$$\mathcal{A}(D_{\text{train}}) = \hat{f}_p = \arg \min_{f \in \tilde{F}_p} \hat{L}_{\text{train}}(f) \quad \tilde{F}_1 \subset \dots \subset \tilde{F}_p$$



Bias-Variance Tradeoff

$$\mathbb{E}[L(\hat{f}_D)]$$

Let l be squared loss

$$\text{if } \bar{f} = f^*$$

$= \mathbb{E}\mathbb{E}$

$= \mathbb{A}\mathbb{E}$

$$= \underbrace{\mathbb{E}\left[\underbrace{\mathbb{E}\left[(\hat{f}_D(\vec{x}) - \bar{f}(\vec{x}))^2 \mid \vec{x}\right]}_{\text{Variance} = \text{Var}[\hat{f}_D(\vec{x}) \mid \vec{x}]} \right]}_{= \mathbb{E}\mathbb{E}} + \underbrace{\mathbb{E}\left[\underbrace{(\bar{f}(\vec{x}) - f_{\text{Bayes}}(\vec{x}))^2}_{\text{Bias}} \right]}_{= \mathbb{A}\mathbb{E}} + \underbrace{L(f_{\text{Bayes}})}_{\text{IE}}$$

where $\bar{f}(\vec{x}) = \mathbb{E}[\hat{f}_D(\vec{x}) \mid \vec{x}]$ "expected predictor"

Effects of changing F, n on Bias, Variance

Bias $\downarrow$ if $F \uparrow$

Variance $\downarrow$ if $n \uparrow$

$\uparrow$ if $F \uparrow$

Regularization

$$\text{let } \vec{w} = (w_0, w_1, \dots, w_{\bar{p}-1})^T \in \mathbb{R}^{\bar{p}}$$

Observation: large values of $|w_0|, |w_1|, \dots, |w_{\bar{p}-1}|$ leads to more complex $f_p(\vec{x}) = \phi_p(\vec{x})^T \vec{w}$

Regularization: penalize large weights

If $f_p \in \mathcal{F}_p$:

$$= \hat{L}(f_p)$$

$$\hat{L}_\lambda(f_p) = \frac{1}{n} \sum_{i=1}^n \ell(f_p(\vec{x}_i), y_i) + \frac{\lambda}{n} \sum_{j=1}^{\bar{p}-1} w_j^2$$

Regularization parameter
 $\lambda \in [0, \infty)$

Regularized
estimated
loss

$j=0$ not included "Regularizer"

Bias vs. Variance

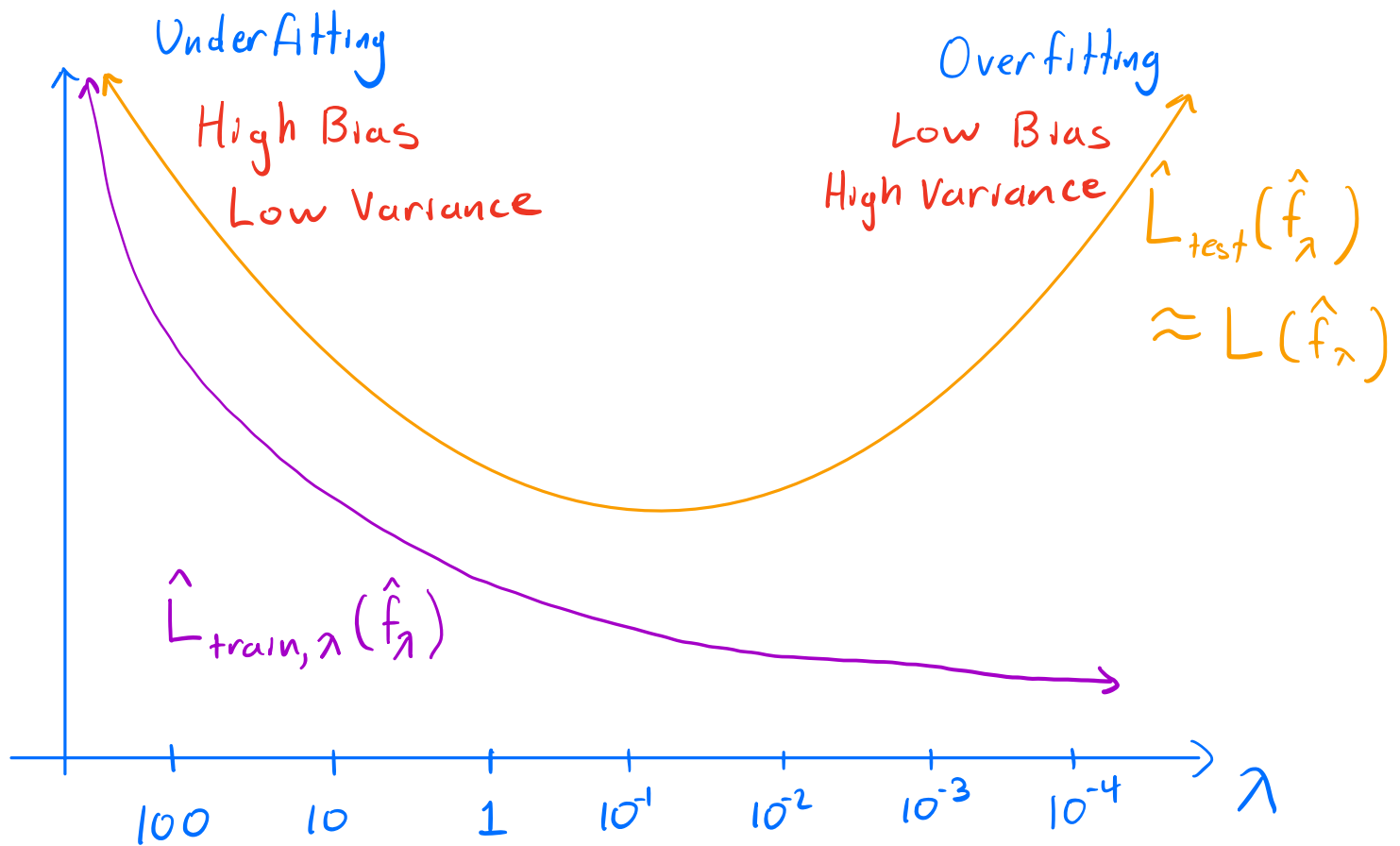
$$\text{Bias: } (\bar{f}_\lambda(\vec{x}) - f_{\text{Bayes}}(\vec{x}))^2$$

- Decreases if λ decreases

$$\text{Variance: } E[(\hat{f}_{0,\lambda}(\vec{x}) - \bar{f}_\lambda(\vec{x}))^2 | \vec{x}]$$

- Increases if λ decreases
- Decreases if n increases

$$\hat{f}_\lambda = \operatorname{argmin}_{f \in \mathcal{F}_0} \hat{L}_\lambda(f)$$



Minimizing $\hat{L}_\lambda(f)$

Objective:

$$\hat{\vec{w}}_\lambda = \operatorname{argmin}_{\vec{w} \in \mathbb{R}^{d+1}} \hat{L}_\lambda(\vec{w}) \quad \text{using squared loss, } \mathcal{F}_1$$

$$\text{where } \hat{L}_\lambda(\vec{w}) = \frac{1}{n} \sum_{i=1}^n (\vec{x}_i^T \vec{w} - y_i)^2 + \frac{\lambda}{n} \sum_{j=1}^d w_j^2$$

BGD:

$$\vec{w}^{(t+1)} = \vec{w}^{(t)} - \eta^{(t)} \nabla \hat{L}_\lambda(\vec{w}^{(t)})$$

MLE

$$f_{\text{Bayes}} = \underset{f \in \{f \mid f: \mathcal{X} \rightarrow \mathcal{Y}\}}{\operatorname{argmin}} L(f)$$

assume squared loss and Regression

$$\begin{aligned} f_{\text{Bayes}}(\vec{x}) &= \mathbb{E}[Y \mid \vec{X} = \vec{x}] \\ &= \int_{\mathcal{Y}} y \, p_{Y|\vec{X}}(y|\vec{x}) \, dy \end{aligned}$$

Use $\mathcal{D}$ to estimate $p_{Y|\vec{X}}$

MLE Basics

$\mathcal{D} = (Z_1, \dots, Z_n)$ and Z_i are i.i.d. with p_Z

Assume p_Z is based on some parameter w^*

You are given a fixed data $\mathcal{D} = (z_1, \dots, z_n)$

$$\begin{aligned} w_{\text{MLE}} &= \underset{w \in \mathcal{W}}{\operatorname{argmax}} \underbrace{p(\mathcal{D}|w)}_{\text{likelihood}} \\ &= \underset{w \in \mathcal{W}}{\operatorname{argmin}} \underbrace{-\log(p(\mathcal{D}|w))}_{\text{negative log-likelihood}} \end{aligned}$$

$$= \arg \min_{w \in \mathcal{H}} -\log \left(\prod_{i=1}^n p(z_i | w) \right)$$

$$= \arg \min_{w \in \mathcal{H}} -\sum_{i=1}^n \log(p(z_i | w))$$

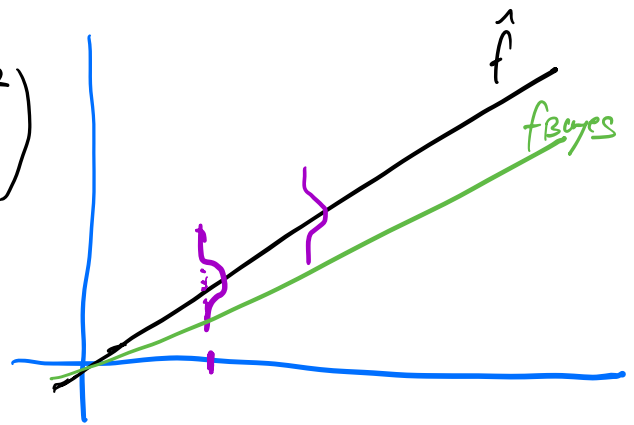
Estimating $P_{Y|\vec{X}}$

$$D = ((\vec{X}_1, Y_1), \dots, (\vec{X}_n, Y_n)) \in (\mathcal{X} \times \mathcal{Y})^n, P_D, p_D$$

$(\vec{X}_i, Y_i)$ are i.i.d with $P_{\vec{X}, Y}$ and $p_{\vec{X}, Y}$

Assume $Y_i | \vec{X}_i = \vec{x}_i \sim \mathcal{N}(\vec{x}_i^T \vec{w}^*, 1)$

$$P_{Y|\vec{X}=\vec{x}}(y|\vec{x}) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(y - \vec{x}^T \vec{w}^*)^2}{2}\right)$$



$$\vec{w}_{MLE} = \arg \min_{\vec{w} \in \mathbb{R}^{d_H}} \sum_{i=1}^n \frac{(Y_i - \vec{x}_i^T \vec{w})^2}{2}$$

$$= \arg \min_{\vec{w} \in \mathbb{R}^{d_H}} \hat{L}(\vec{w})$$

$$f_{Bayes}(\vec{x}) \approx \vec{x}^T \vec{w}_{MLE}$$

MAP

If P_{riX} is a function of some parameter $w \in W$ then we just need to estimate that parameter

$$\text{MLE: } \arg \max_{w \in W} \underbrace{p(D|w)}_{\text{Likelihood}}$$

$$\text{MAP: } \arg \max_{w \in W} \underbrace{p(w|D)}_{\text{Posterior}}$$

Bayes' Rule

$$\begin{aligned} w_{\text{MAP}} &= \arg \max_{w \in W} p(w|D) \\ &= \arg \max_{w \in W} \frac{p(w, D)}{P(D)} \\ &= \arg \max_{w \in W} \frac{p(D|w) p(w)}{P(D)} \\ &= \arg \max_{w \in W} \underbrace{p(D|w)}_{\text{likelihood}} \underbrace{p(w)}_{\text{prior}} \end{aligned}$$

$$= \arg \min_{w \in \mathcal{W}} -\log(p(D|w) p(w))$$

$$= \arg \min_{w \in \mathcal{W}} [-\log(p(D|w)) - \log(p(w))]$$

if n large then:

$$w_{\text{MAP}} \approx w_{\text{MLE}}$$

if $p(w) = c$ a constant then: $w_{\text{MAP}} \approx w_{\text{MLE}}$

if $\text{Var}(w)$ is large then: $w_{\text{MAP}} \approx w_{\text{MLE}}$

small then: $w_{\text{MAP}} \approx \mathbb{E}[w]$

Estimating $\vec{w}$ for $P_{Y|\vec{X}}$

$$D = ((\vec{X}_1, Y_1), \dots, (\vec{X}_n, Y_n)) \in (\mathcal{X} \times \mathcal{Y})^n, \mathbb{P}_D, \rho_D$$

$(\vec{X}_i, Y_i)$ are i.i.d with $\mathbb{P}_{\vec{X}, Y}$ and $\rho_{\vec{X}, Y}$

Assume $Y_i | \vec{X}_i = \vec{x}_i \sim \mathcal{N}(\vec{x}_i^T \vec{w}^*, 1)$

$$P_{Y|\vec{X}=\vec{x}}(y|\vec{x}) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(y - \vec{x}^T \vec{w}^*)^2}{2}\right)$$

Assume $w_j \sim \mathcal{N}(0, \frac{1}{\lambda})$ are i.i.d. for $j \in \{1, \dots, d\}$

and $w_0 \sim \mathcal{N}(0, \sigma^2)$ for very large σ

$\approx \text{Uniform}(-a, a)$ for large a

w_0 is independent of w_j for all $j \in \{1, \dots, d\}$

$$\vec{w}_{\text{MAP}} = \underset{\vec{w} \in \mathbb{R}^{d+1}}{\text{argmax}} \quad p(\vec{w} | D)$$

$$= \underset{\vec{w} \in \mathbb{R}^{d+1}}{\text{argmin}} \left[\underbrace{\sum_{i=1}^n \frac{(y_i - \vec{x}_i^T \vec{w})^2}{2}}_{= \frac{n}{2} \hat{L}} + \underbrace{\frac{\lambda}{2} \sum_{j=1}^d w_j^2}_{\text{almost regularizer}} \right]$$

$$= \underset{\vec{w} \in \mathbb{R}^{d+1}}{\text{argmin}} \left[\hat{L}_{\lambda}(\vec{w}) \right]$$

$$f_{\text{Bayes}}(\vec{x}) \approx \vec{x}^T \vec{w}_{\text{MAP}}$$

Lasso regression

Assume $W_j \sim \text{Laplace}(0, 1/\lambda)$ are i.i.d for $j \in \{1, \dots, d\}$

and $W_0 \sim \text{Laplace}(0, b)$ for very large b

$\approx \text{Uniform}(-a, a)$ for large a

W_0 is independent of W_j for all $j \in \{1, \dots, d\}$

$$\vec{w}_{\text{MAP}} = \arg \max_{\vec{w} \in \mathbb{R}^{d+1}} p(\vec{w} | D)$$

$$= \arg \min_{\vec{w} \in \mathbb{R}^{d+1}} \left[\sum_{i=1}^n \frac{(y_i - \vec{x}_i^T \vec{w})^2}{2} + \lambda \sum_{j=1}^d |w_j| \right]$$