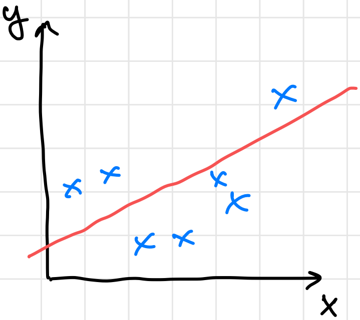
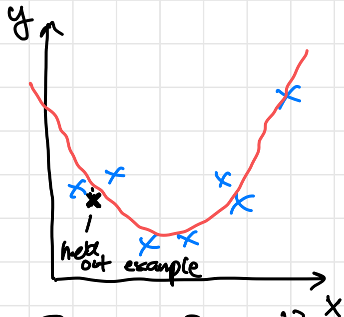
9/15/2023 : Overfitting, Regularization



Features: $[1, x]$

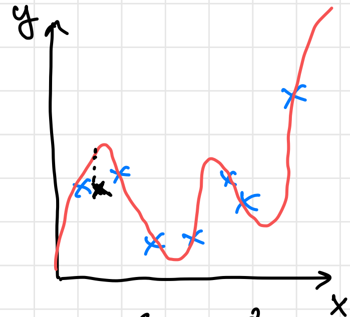
too simple
"underfitting"



Features: $[1, x, x^2]$

good balance
between underfitting & overfitting

WANT THIS



Features: $[1, x, x^2, x^3, x^4, x^5, x^6, x^7]$

zero training loss
but still not good
"overfitting" – too complex, unlikely to generalize to new x's

AVOID THIS

Separating training & testing

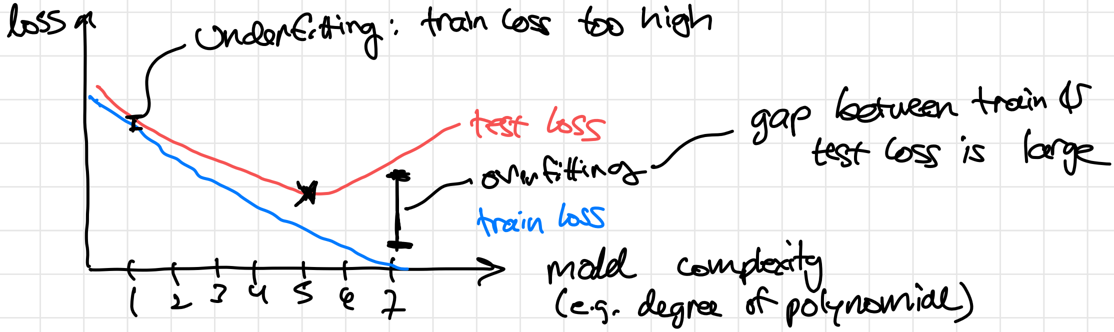
Dataset

split (randomly) into 3 subsets
separate

$\approx 70\%$ → Training data
what we use to fit parameters of model

$\approx 10\%$ → Development ("dev") dataset
Use to choose hyperparameters

$\approx 20\%$ → Test data
Evaluate how well model works
"Held out"-data b/c hidden from model during training



Big question: How do we choose right level of model complexity

Term: hyperparameter: Any setting of learning algorithm
- Which features?
- Learning rate
- How long to run gradient descent

vs. parameter: directly learned by the ML algorithm

To choose hyperparameters:

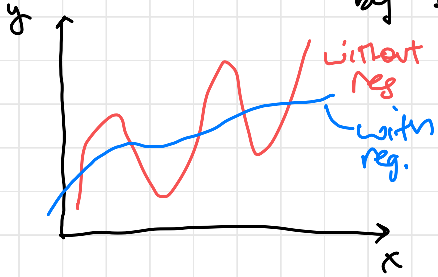
1. Train with different hyperparameter values to get 1 model per each choice
2. Evaluate each model on dev set
3. Choose model with lowest dev set loss
4. Evaluate this model only on test set

Why not use test set? Still a form of cheating

Models should only get one chance to take real test

dev set is a "practice test"

Regularization: A technique to reduce overfitting by encouraging "simpler" models / function



L_2 Regularization: Encourage $\boxed{L_2 \text{ norm}}$ of parameters to be small by adding additional term to loss

$= \|w\|$

e.g. linear regression

$$L(w) = \underbrace{\frac{1}{n}\sum_{i=1}^{n}\left(w^T x^{(i)} - y^{(i)}\right)^2}_{\text{original loss}} + \underbrace{\lambda \boxed{\|w\|^2}}_{\text{regularization term}}$$

λ: constant ≥ 0, $\lambda = 0$ is no regularization

$\|w\|^2 = \sum_{j=1}^{d} w_j^2$

How does this affect gradient?

$$\nabla_w L(w) = [\text{old gradient}] + \lambda \cdot 2w$$

During G.D. $\quad w \leftarrow w - \eta\left([\text{old gradient}] + \lambda 2 \cdot w\right)$

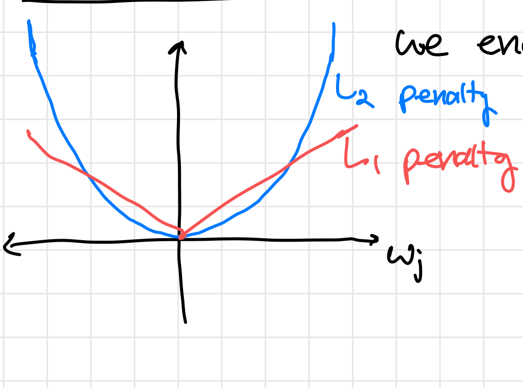
subtracting multiple of w ie step towards origin

"weight decay"

L_1 Regularization: Similar to L_2 but

we encourage $\|w\|_1 = \sum_{j=1}^{d} |w_j|$ to be small

by adding $\lambda \|w\|_1$ to loss



Gradient for L_1 loss:

$$\frac{d}{dw_j} \|w\|_1 = \text{sign}(w_j)$$

$$\text{sign}(z) = \begin{cases} 1 & \text{if } z > 0 \\ -1 & \text{if } z < 0 \\ 0 & \text{if } z = 0 \end{cases}$$

so $$\nabla_w \|w\|_1 = \begin{bmatrix} \text{sign}(w_1) \\ \vdots \\ \text{sign}(w_d) \end{bmatrix} = \text{sign}(w)$$

vs. $$\nabla_w \|w\|^2 = 2w$$

L_1: Always take constant-sized step — Sparsifying effect: encourages some w_j to $= 0$ exactly

L_2: Take small step for small w, large step for large w — Avoid really big w_j's