Deep Learning - 2019

Machine Learning Basics

Prof. Avishek Anand

Why do we use Machine Learning

- Essence of Machine Learning
  - A pattern exists
  - We cannot pin it down mathematically
  - We have data on it…
  - Can we (hope) learn the underlying function that generates this data ?

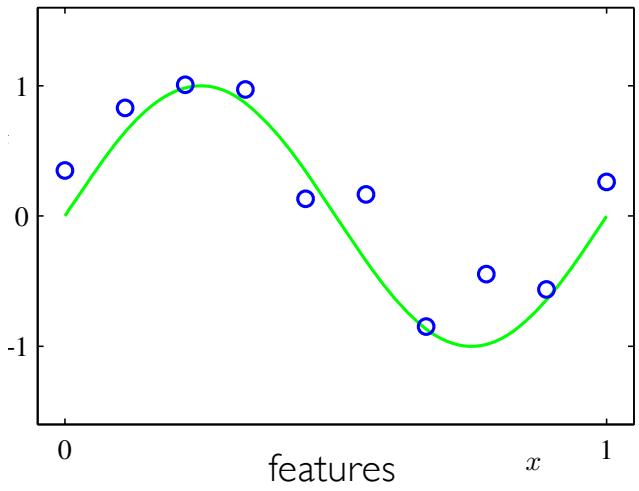
- **The credit rating problem** – Decide whether someone should get credit or not based on a rating.
  - What credit rating should we give this customer in a scale of [1-100] ? Regression Problem
  - Should we accept or reject this customer for a loan ? Classification Problem

age	23 years
gender	male
annual salary	$30,000
years in residence	1 year
years in job	1 year
current debt	$15,000

Training data: Unknown distribution samples

- There is some data generating distribution g(x) -- Unseen
- We have some samples from this distribution – Seen/observed

$$g(x) = \sin(2\pi x) + \epsilon$$



- We want to build a model on these observed samples so as to approximate the True underlying data generating distribution ?
  - Train on the observed data – Observed data is called the training data
- Success of the model is how good we generalize
  - How close we get to the Actual distribution ?

Training data and Hypothesis space

- Input / Query : Customer Application $\mathbf{x}$

age	23 years
gender	male
annual salary	$30,000
years in residence	1 year
years in job	1 year
current debt	$15,000

- Output : good / bad customer or credit score ? y

- Target Function: **ideal** credit approval formula (unknown and something we want to approximate)

$$g : \mathcal{X} \rightarrow \mathcal{Y}$$

- Training data / Data: historical records (x = features and y = labels)

$$\left(\mathbf{x}_1, y_1\right), \left(\mathbf{x}_2, y_2\right), \cdots, \left(\mathbf{x}_N, y_N\right)$$

- Hypothesis: Formula to be used (polynomial family, probabilistic, neural networks) $f : \mathcal{X} \rightarrow \mathcal{Y}$

Given training data, a hypothesis class, find a hypothesis that describes the data the best…

How do we measure performance ?

Given training data, a hypothesis class, find a hypothesis that describes the data the best…

- Makes good prediction on unseen data
  - Assumption: seen and unseen data follow the same distribution
  - Test data: Sampled from the training data

- How do we measure good predictions ?
  - What you cannot measure, you cannot optimize…

- Measures: Minimize some notion of error in the unseen / test data
  - Error in our credit risk example – how much does our prediction deviate from the actual response ?
  - Absolute loss, squared loss

Setting up a ML Problem

$\{(x_i, y_i) : 1 \leq i \leq n\}$ $\quad\quad$ $y = f(x) \in \mathcal{H}$

- Given **training data** i.i.d from a **distribution D**, find a **hypothesis** (from a hypothesis set you consider) s.t. the **expected loss** is small

$$\mathbb{E}_{(x,y)\sim D}\left[\, l(f, x, y) \,\right]$$

- In practice we minimize the **Empirical Loss (Training error)**-- the loss on the training data

$$\mathcal{L}(f) = \frac{1}{n}\sum_{i=1}^{n} l\left(f, x_i, y_i\right)$$

- Expected Loss vs Empirical Loss: Expected loss (Test error) can be very different from empirical loss (Training error)?
  - **Expected loss:** True risk (loss a random test point) or test error

Setting up a ML Problem

- ML 1-2-3
  - Collect data and extract features
  - Build Model: choose hypothesis class and loss function
  - Optimization: Minimize the empirical loss

- Design decisions
  - Choose good hypothesis class – polynomials, neural networks, graphical models
  - Choose good loss function – linear loss, quadratic loss, cross-entropy
  - Choose good optimization and regularization techniques

- Do not fool yourself (practical advice)
  - Setup good measures, good representative training and test data, no neural nets until they are necessary…

Linear Regression

- Given training data i.i.d from a distribution D, find a hypothesis s.t. the empirical loss is small
- Hypothesis $f(x) = w^T x$

$$X = \begin{bmatrix} -\mathbf{x}_1^\top - \\ -\mathbf{x}_2^\top - \\ -\mathbf{x}_N^\top - \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_N \end{bmatrix}$$

- Loss function L2 Loss

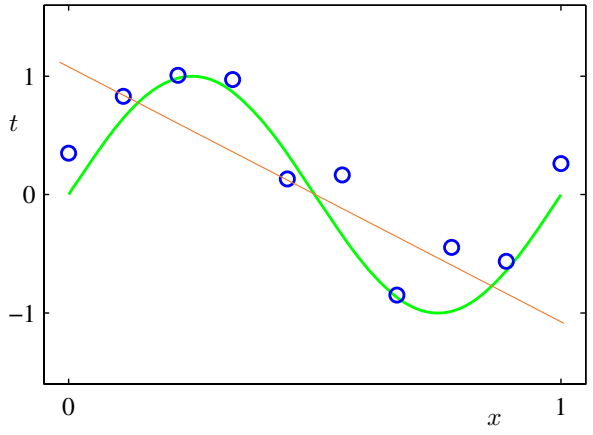
$$\mathcal{L}(f_w) = \frac{1}{n} \sum_{j=1}^{n} \left(w^T x_i - y_i \right)^2 = \frac{1}{n} \|Xw - y\|_2^2$$

- The best hypothesis finding problem = find the best parameters w

Linear Regression

- The best hypothesis is the one that minimizes the **loss function on training data**
  - Loss function is also called in-sample error

$$\mathcal{L}(f) = \frac{1}{n}\sum_{i=1}^{n}\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_I - y_i\right)^2$$
$$= \frac{1}{n}\|X\mathbf{w} - \mathbf{y}\|^2$$



- Minimizing the loss is easier for continuous functions (use basic calculus)

$$\nabla_w \mathcal{L}(f) = \frac{2}{n}\mathrm{X}^{\top}\left(\mathrm{X}\mathrm{w} - \mathbf{y}\right) = \mathbf{0}$$

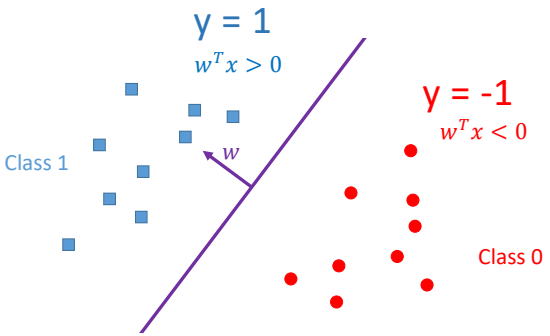
$$\mathbf{w} = X^{\dagger}\mathbf{y} \text{ where } \mathbf{X}^{\dagger} = \left(\mathbf{X}^{\top}\mathbf{X}\right)^{-1}\mathbf{X}^{\top}$$

Idea 1: Simple Linear Model

- Attributes or features of the customer $\mathbf{x} = (x_1, \cdots, x_d)$
- Hypothesis class: $f(x) = w^T x$

$$X = \begin{bmatrix} -\mathbf{x}_1^\top - \\ -\mathbf{x}_2^\top - \\ -\mathbf{x}_N^\top - \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_N \end{bmatrix}$$

- Approve Credit $\sum_{i=1}^{d} w_i x_i > \text{threshold} \quad w^T x > \text{threshold}$
- Deny Credit $\sum_{i=1}^{d} w_i x_i < \text{threshold} \quad w^T x < \text{threshold}$



- A simple Loss function can we expressed as $\mathcal{L}(f) = \frac{1}{n} \sum_{i=1}^{n} 1_{f(x_i) \neq y_i}$

$$\mathcal{L}(f) = |\,\text{sign}\,(w^T x - \text{threshold}\,) - y|$$

What is the problem with this loss function ? *Hard to optimize (non-differentiable)*

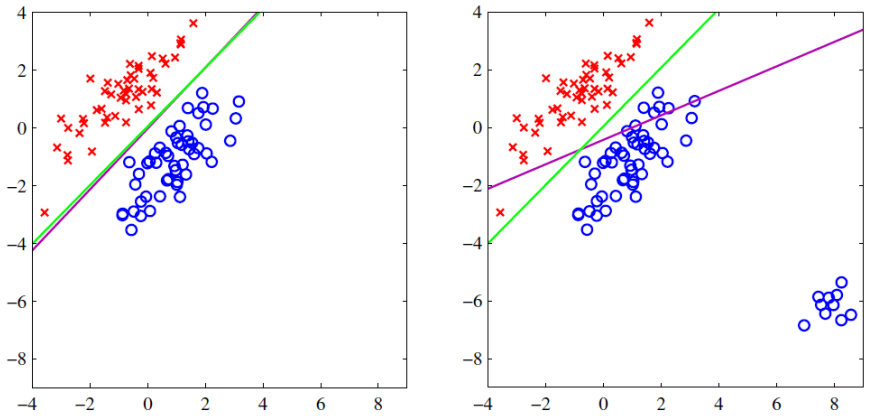
Idea 2: Linear Regression for Classification

$$X = \begin{bmatrix} -\mathbf{x}_1^\top - \\ -\mathbf{x}_2^\top - \\ -\mathbf{x}_N^\top - \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_N \end{bmatrix}$$

- Hypothesis class: $f(x) = w^T x$
- Pretend that you are doing linear regression: the labels are not only {1, -1} but can take any value

$$\mathcal{L}(f_w) = \frac{1}{n}\sum_{j=1}^{n}\left(w^T x_i - y_i\right)^2 = \frac{1}{n}\|Xw - y\|_2^2$$

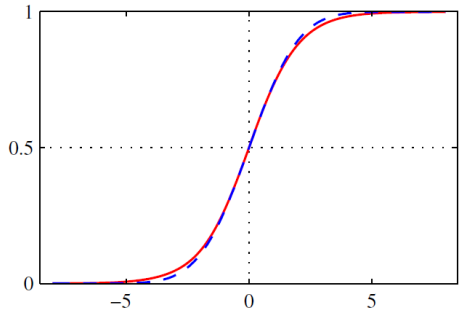
What is the problem with this ?



Not Robust to outliers

Logistic Regression

- Sigmoid function
  - **Continuous function:** optimization is easier (gradients can be computed)
  - **Squashes the value [0,1]:** Outliers are get the same extreme value 0 or 1



$$\sigma\left(w^T x\right) = \frac{1}{1+\exp\left(-w^T x\right)}$$

$$\mathcal{L}\left(f\right) = \frac{1}{n}\sum_{i=1}^{n}\left(\sigma\left(w^T x_i\right) - y_i\right)^2$$

Squared Loss for regression

$$\mathcal{L}\left(f\right) = \frac{1}{n}\sum_{j=1}^{n}\left(w^T x_i - y_i\right)^2$$

- We rely on the optimization for linear regression

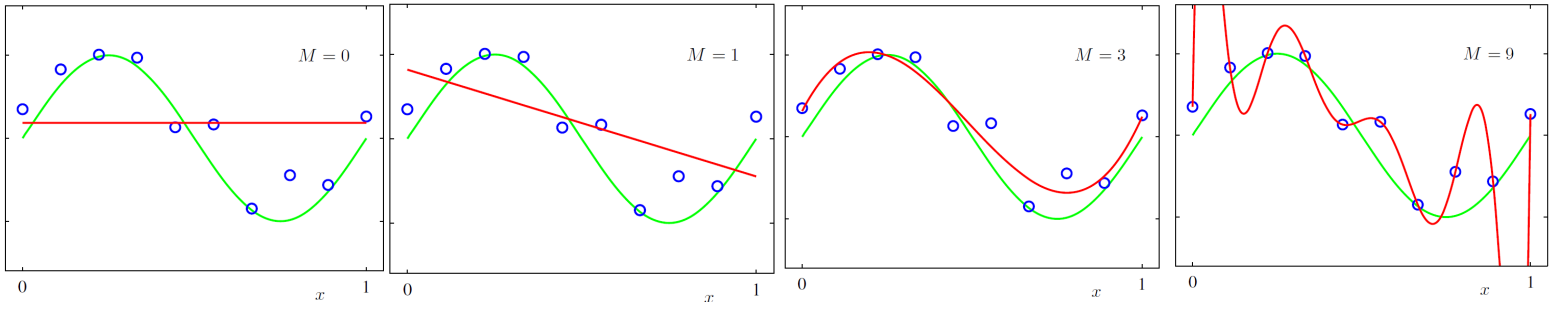
- Essentially we have been focusing on optimization. what about generalization ?

Overfitting

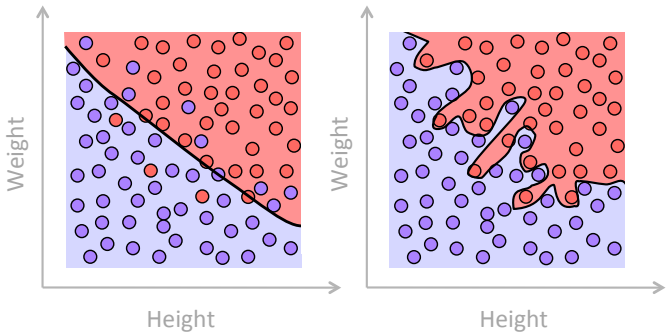
$$\mathbb{E}_{(x,y)\sim D}\left[\, l(f,x,y) \,\right]$$

- **Generalization:** The ability to fit to unseen data
- **Underfitting:** High Training loss, model not complex enough, low model capacity

$$g(x) = \sin(2\pi x) + \epsilon$$



- **Overfitting:** High test error although training error is low
  - High model capacity

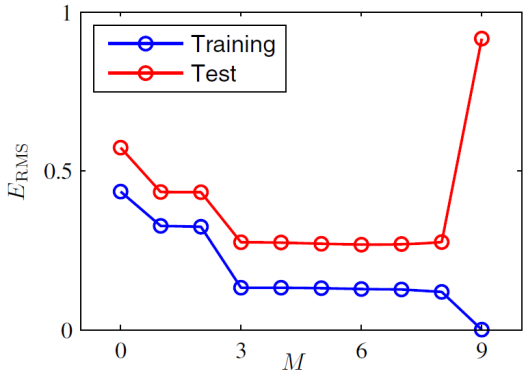


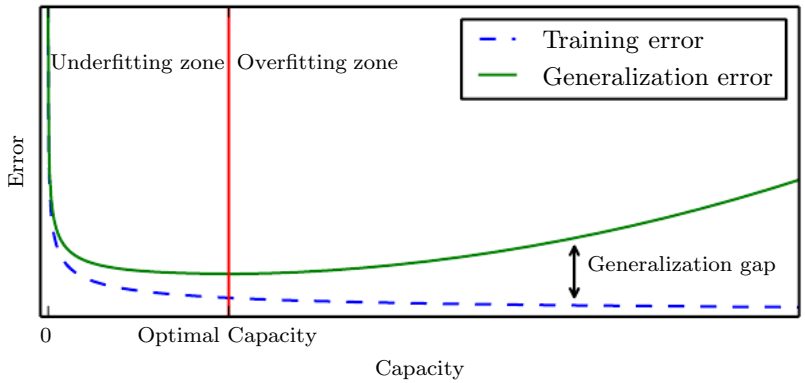
Preventing Overfitting

"among competing hypotheses that explain known observations equally well, one should choose the simplest one."

- Use more training data
- Constrain the models
  - Larger the hypothesis class, easier to find a hypothesis that has a good trade-off
  - Throw away or penalize useless hypotheses

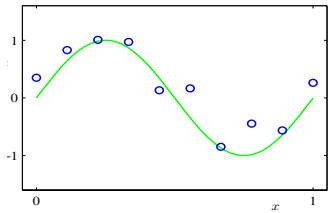
- Regularization: Limiting the hypothesis in the hypothesis space (reduces the gen. gap)





Bias-Variance Tradeoff

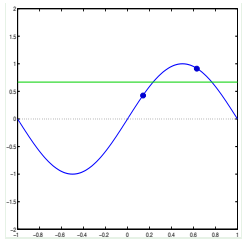
- Consider a data generating function that is a since curve
- Our training samples are always 2 points on this curve

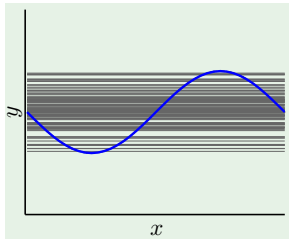


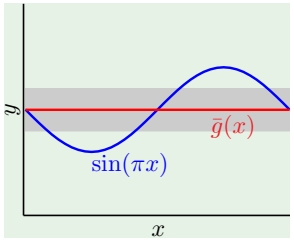
features

$$g(x) = \sin(2\pi x) + \epsilon$$
$$\epsilon \sim \mathcal{N}(0, \sigma^2)$$

$\mathcal{H}_0$

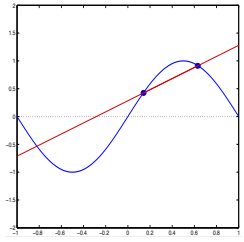


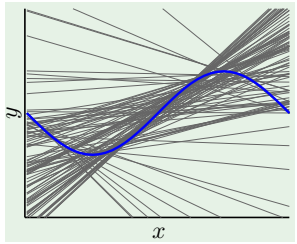


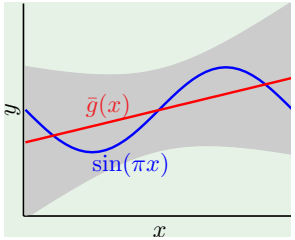


$$\text{Bias}[f(x)] = \mathbb{E}[f(x)] - g(x)$$

$\mathcal{H}_1$







$$\text{Var}[f(x)] = \mathbb{E}\left[f(x)^2\right] - \mathbb{E}[f(x)]^2$$

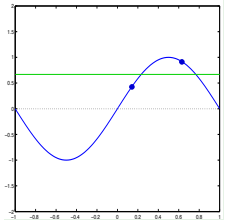
$$\mathbb{E}\left[(y - f(x))^2\right] = (\text{Bias}[f(x)])^2 + \text{Var}[f(x)] + \sigma^2$$

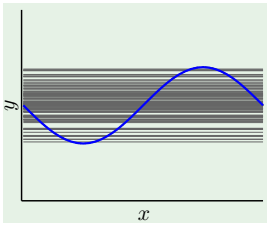
→ Irreducible error (Bayes error rate)

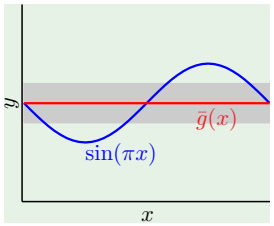
Bias-Variance Tradeoff

$$\mathbb{E}\left[(y - f(x))^2\right] = (\text{Bias}[f(x)])^2 + \text{Var}[f(x)] + \sigma^2$$

$\mathcal{H}_0$



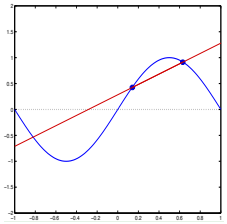


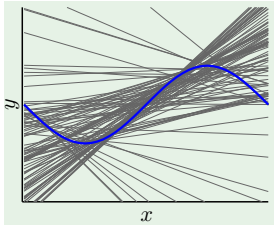


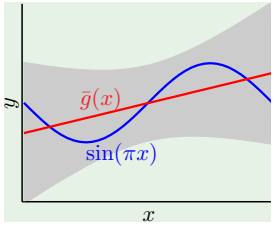
Bias = 0.50

Variance = 0.25

$\mathcal{H}_1$

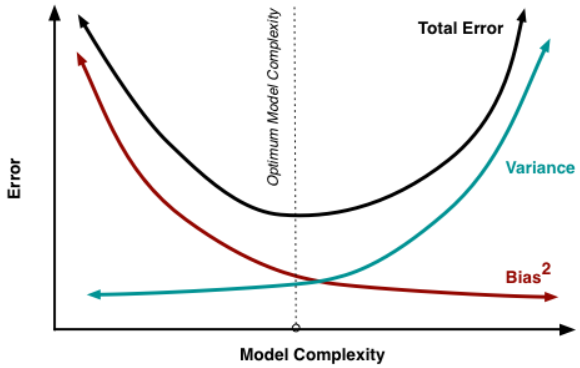






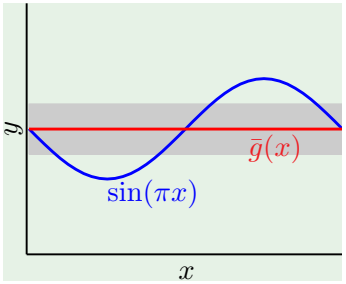
Bias = 0.21

Variance = 1.69



Bias-Variance Tradeoff

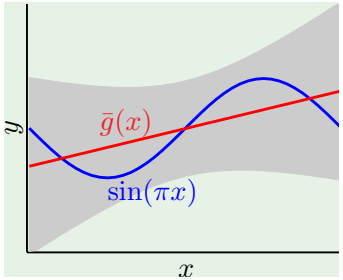
- The total expected error is $\mathbf{Bias}^2 + \mathbf{Variance}$
- Because of the bias-variance trade-off, we want to balance these two contributions.



Complex models (many parameters) usually have lower bias, but higher variance.

Simple models (few parameters) have higher bias, but lower variance.

- Overfitting : If **Variance** strongly dominates, it means there is too much variation between models
- Underfitting : If **Bias** strongly dominates, then the models are not fitting the data well enough



Regularization

- How do we tackle Under-fitting: Consider a higher model capacity (e.g. higher deg. poly)
- Answer to Overfitting – Regularization or restrict the models

Hard Constraints

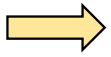
$$\min_\theta \mathcal{L}(\theta) = \frac{1}{n}\sum_{i=1}^{n} l\left(\theta, x_i, y_i\right)$$

$$\text{subject to: } R(\theta) \leq r$$

Example – L2 Regularization

$$\min_\theta \mathcal{L}(\theta) = \frac{1}{n}\sum_{i=1}^{n} l\left(\theta, x_i, y_i\right)$$

$$\text{subject to: } \|\theta\|_2^2 \leq r^2$$



Soft Constraints – L2 Reg.

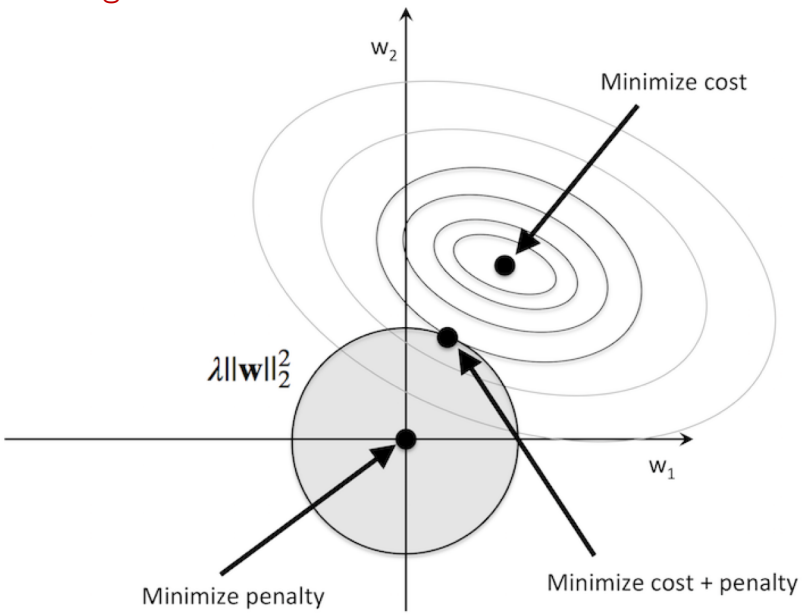
$$\min_\theta \mathcal{L}_{Reg}(\theta) = \frac{1}{n}\sum_{i=1}^{n} l\left(\theta, x_i, y_i\right) + \lambda\|\theta\|_2^2$$

- Regularization restricts the choice of parameters
- L2 Regularization is the most common regularization technique
  - Hard constraints can be translated to softer constraints that are easier to optimize

Regularization: Loss Surface

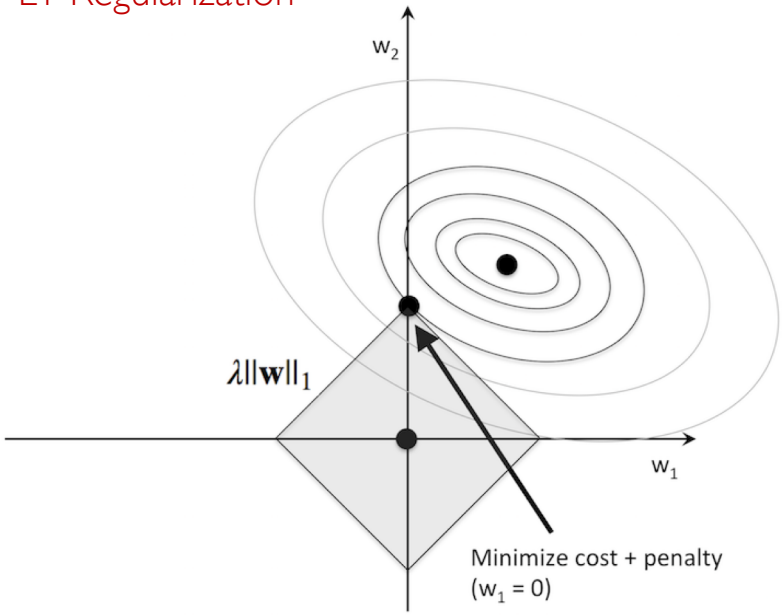
- How does the loss surface look like when there are 2 params for squared loss?

L2 Regularization



$$\min_{\theta} \mathcal{L}_{Reg}(\theta) = \frac{1}{n}\sum_{i=1}^{n} l\left(\theta, x_i, y_i\right) + \lambda\|\theta\|_2^2$$

L1 Regularization

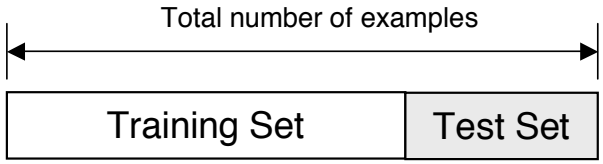


$$\min_{\theta} \mathcal{L}_{Reg}(\theta) = \frac{1}{n}\sum_{i=1}^{n} l\left(\theta, x_i, y_i\right) + \lambda\|\theta\|_1$$

Validation

How to choose the optimal capacity?

- e.g., choose the best degree for polynomial curve fitting



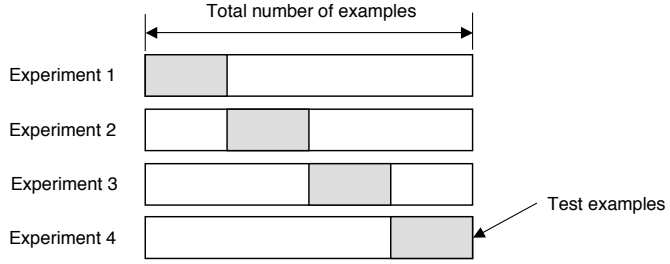
How to set best hyperparameters ?

- **Hyperparameters:** Settings that we can use to control the behavior of the learning algorithm
- Effect of regularization, learning rates (optimization spec.)

- Cannot be done by training data alone
- Create held-out data to approx. the test error -- validation data set

- Given a dataset D = (x,y)…
  - K Points are taken for validation → N-K are used for training

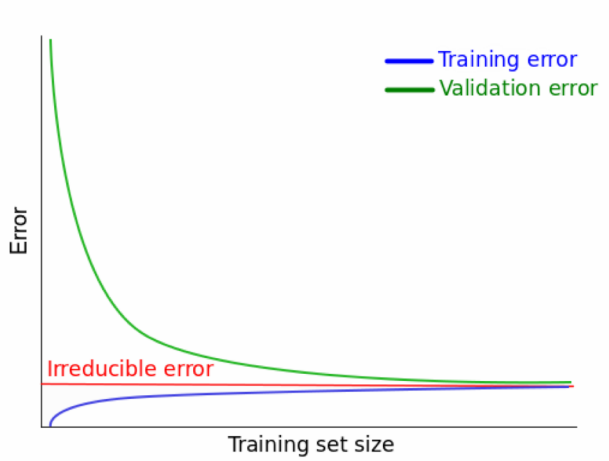
- Cross Validation: Use the entire dataset by binning the training data



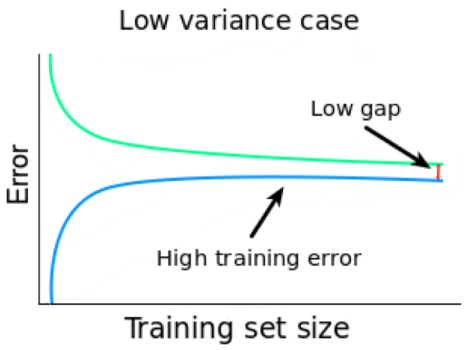
Validation – learning curves

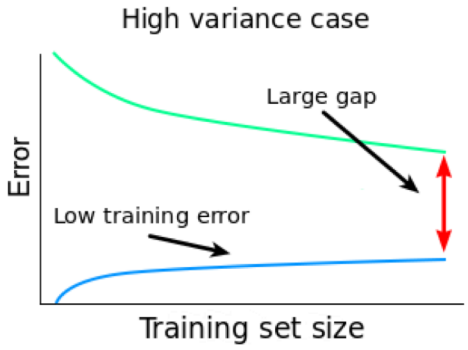
How to choose the optimal capacity?
How to set best hyperparameters ?

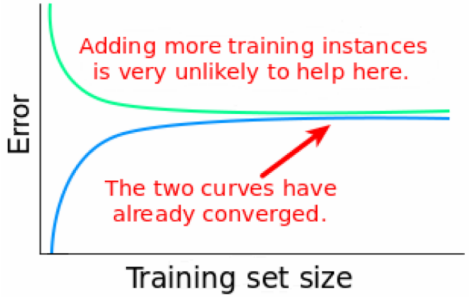
Learning Curves: Measure training and validation set performance with increasing training data size

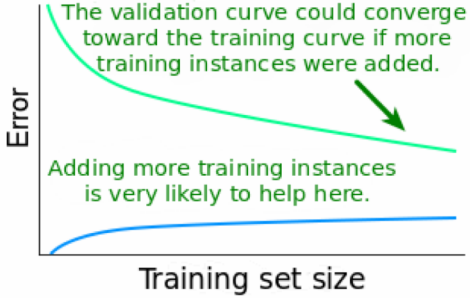


- What if the training error is small for large training set ?
- What if the training error is v. small but validation error is large ?









Density estimation : Maximum Likelihood Estimator

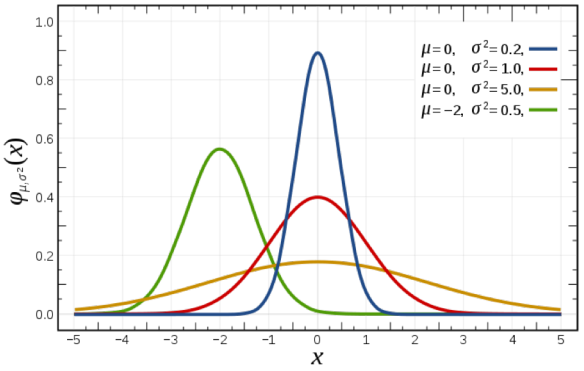
- An attempt to make the model distribution match the empirical distribution
  - Parameterization for the model that best fits the data
- Assume data is independent and identically distributed s(i.i.d) so that p factorizes

Parametric Family of distributions

$$\boldsymbol{\theta}_{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta}} p_{\mathrm{model}}(\mathbb{X}; \boldsymbol{\theta})$$

$$\boldsymbol{\theta}_{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x}\sim\hat{p}_{\mathrm{data}}} \log p_{\mathrm{model}}(\boldsymbol{x}; \boldsymbol{\theta})$$

- MLE Example: Gaussian distribution (on white board)
- Parameter space view



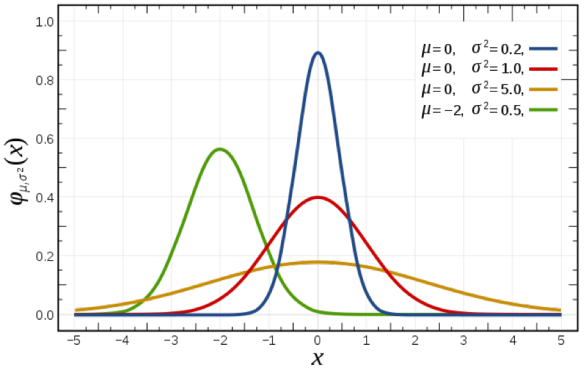
MLE Example

$$\boldsymbol{\theta}_{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x}\sim\hat{p}_{\mathrm{data}}} \log p_{\mathrm{model}}(\boldsymbol{x};\boldsymbol{\theta})$$

- Given a set of input data = $\{x_1, \ldots, x_n\}$ find the MLE if we assume it fits the Gaussian $N(\mu, \sigma^2)$

$$p(x_1,\ldots,x_N|\mu,\sigma^2) = \prod_{n=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{\frac{-(x_n-\mu)^2}{2\sigma^2}\right\}$$

$$\mathcal{L}(\mu,\sigma) = -\frac{1}{2}N\log(2\pi\sigma^2) - \sum_{n=1}^{N}\frac{(x_n-\mu)^2}{2\sigma^2}$$



- What are the partial derivatives w.r.t paramters ?
  - Set them to zero to analytically solve for the optimal parameters
  - Not always possible for all distributions of Likelihoods

Thank You

Email for Assignment Submissions: pir-assignments@l3s.de
Email for Lecture related queries : leonhardt@l3s.de, zzhang@l3s.de
Use [DL2019] in the subject line

References

- Bishop Book (PRML)
- Deep Learning Book (Chapter 3)

- Bias Variance Trade-off
  - Other lectures with slides: https://www.cs.cmu.edu/~wcohen/10-601/bias-variance.pdf

- MLE:
  - http://www.cs.cornell.edu/courses/cs4780/2017sp/lectures/lecturenote04.html

- Some Interesting blogs and popular sources:
  - http://scott.fortmann-roe.com/docs/BiasVariance.html
- Advanced stuff (not necessary and only for interested readers)
  - A modern take on bias-variance in Neural Nets: https://arxiv.org/abs/1810.08591