CSED538/AIGS538 Deep Learning

Part 5. Machine Learning Basics

Won Hwa Kim

Slides courtesy of Minsu Cho and Namhoon Lee

Machine Learning

- A form of applied statistics with extensive use of computers to estimate complicated functions
- Two central approaches
  - Frequentist estimators and Bayesian inference
- Two categories of machine learning
  - Supervised learning and unsupervised learning
- Most machine learning algorithms based on stochastic gradient descent optimization
- Focus on building machine learning algorithms

Definitions

Machine Learning?

"the field of study that gives computers the ability to learn without being explicitly programmed." - Arthur Samuel

Definitions

Machine Learning?

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E." - T. Michell 1997

T: Task

- Machine learning tasks are typically difficult to solve with fixed programs designed by humans.
- Machine learning tasks are usually described in terms of how the system should process an example where an example is a collection of features measured from an object or event.
- Many kinds of tasks
  - Classification, regression, transcription, translation, anomaly detection, synthesis & sampling, imputation of missing values, denoising, density estimation, etc.

P: Performance Measure

- Performance *P* is usually specific to the task *T.*
  - For classification tasks, *P* is usually accuracy or error rate; the proportion of examples correctly classified.
  - For regression tasks, *P* is a different performance metric that gives the model a continuous-valued score for each example.
  - In some cases, it is difficult to decide what should be measured, or we know what quantity we would ideally like to measure but measuring it is impractical.
- Interested in the performance on data not seen before
  - Use a test set of data that is separate from the data used for training the machine learning system.

E: Experience

Unsupervised learning algorithms

Learn useful structural properties of the dataset (learn entire probability distribution that generated a dataset, whether explicitly or implicitly)

Supervised learning algorithms

Experience a dataset containing features with each example associated with a **label** or **target**

- Most learning algorithms in our course are allowed to experience an entire training **dataset**.
- But nor very formally defined - blur between supervised and unsupervised *algorithms*

Example: Linear Regression

- Linear regression takes a vector *x* as input and predicts the value of a scalar *y* as its output

$$\hat{y} = \boldsymbol{w}^\top \boldsymbol{x}, \tag{5.3}$$

  - Task: predict scalar *y* from vector *x*
  - Experience: set of vectors *X* and vector of targets *y*
  - Performance: mean squared error => normal equations

$$\mathrm{MSE}_{\mathrm{test}} = \frac{1}{m} ||\hat{\boldsymbol{y}}^{(\mathrm{test})} - \boldsymbol{y}^{(\mathrm{test})}||_2^2, \tag{5.5}$$

$$\nabla_{\boldsymbol{w}} \frac{1}{m} ||\hat{\boldsymbol{y}}^{(\mathrm{train})} - \boldsymbol{y}^{(\mathrm{train})}||_2^2 = 0 \tag{5.7}$$

  - Solution of the normal equations is

$$\boldsymbol{w} = \left(\boldsymbol{X}^{(\mathrm{train})\top} \boldsymbol{X}^{(\mathrm{train})} \right)^{-1} \boldsymbol{X}^{(\mathrm{train})\top} \boldsymbol{y}^{(\mathrm{train})} \tag{5.12}$$

Example: Linear Regression

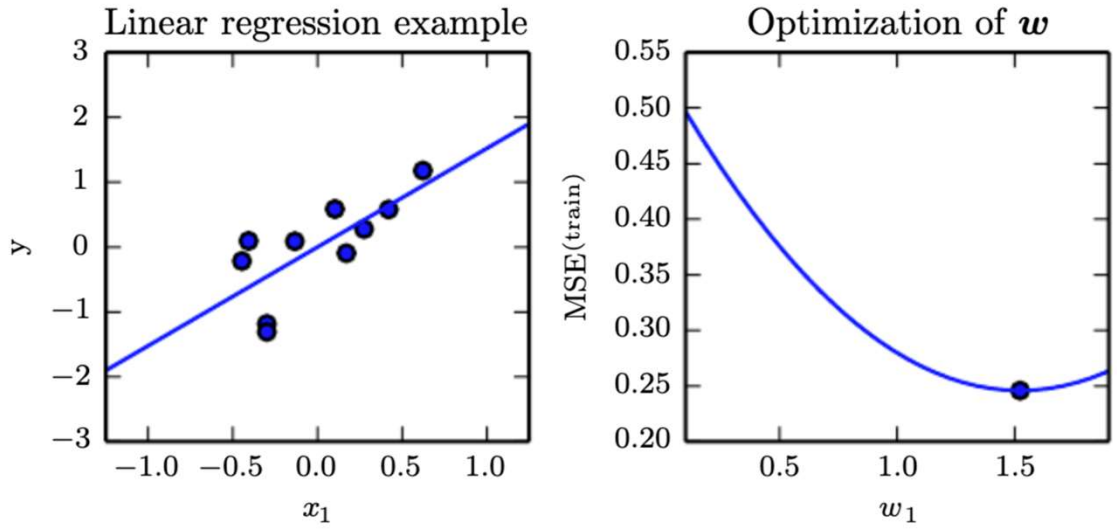


Figure 5.1: A linear regression problem, with a training set consisting of ten data points, each containing one feature. Because there is only one feature, the weight vector $\boldsymbol{w}$ contains only a single parameter to learn, w_1. *(Left)*Observe that linear regression learns to set w_1 such that the line $y = w_1 x$ comes as close as possible to passing through all the training points. *(Right)*The plotted point indicates the value of w_1 found by the normal equations, which we can see minimizes the mean squared error on the training set.

Generalization

Generalization issue

- The central challenge in machine learning is to perform well on new, previously unseen inputs, and not just on the training inputs.
- We want the generalization error (test error) to be low as well as the training error.

- This is what separates machine learning from optimization.
- How can we affect performance on the test set when we get to observe only the training set?
  - Statistical learning theory provides answers.
  - We assume train and test sets are generated independent and identically distributed – i.i.d.
  - Thus, expected train and test errors are equal.

Generalization

- Because the model is developed from the training set, the expected test error is usually greater than the expected training error.
- Factors determining machine performance are
  - Make the training error small
  - Make the gap between train and test error small

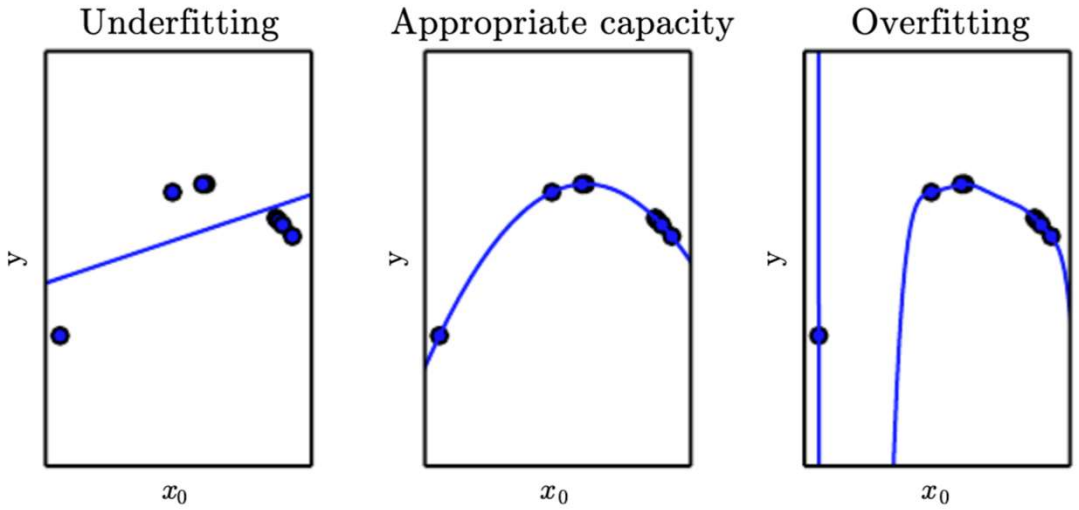
Underfitting and Overfitting

Underfitting

It occurs when the model is not able to obtain a sufficiently low error value on the training set

Overfitting

It occurs when the gap between the training error and test error is too large



Capacity

Capacity

Model's ability to fit a variety of functions

- Low-capacity models struggle to fit the training data
- High-capacity models can overfit the training data

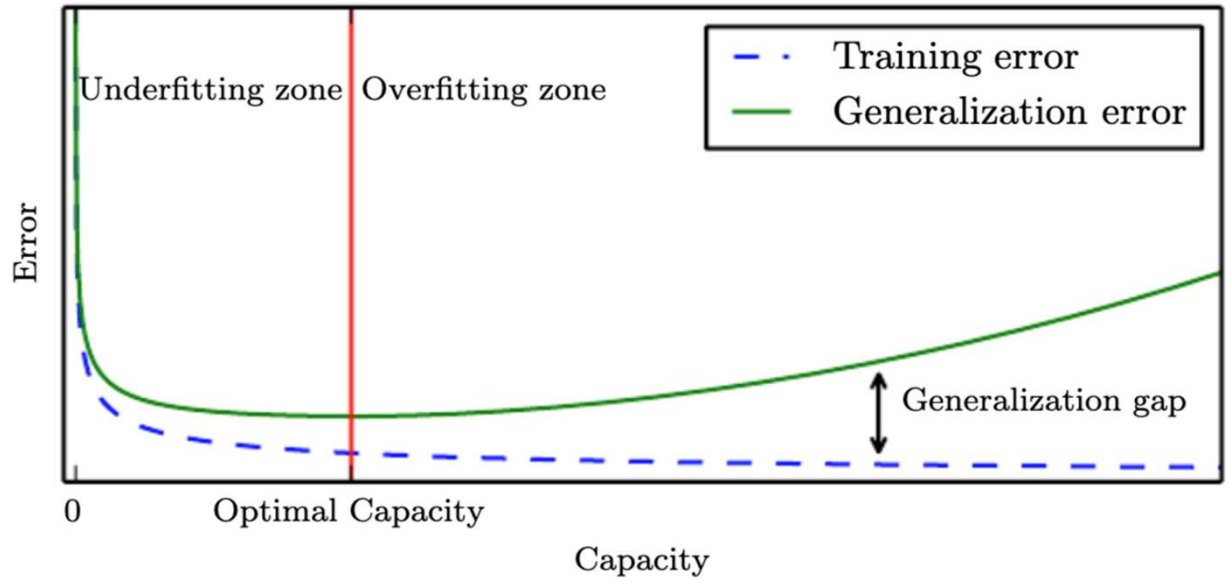
- Underfitting and overfitting can be controlled somewhat by altering the model's capacity
  - One way to control the capacity of a model is by choosing its hypothesis space, e.g., simple linear vs. polynomial linear regression.
  - We can also change the family of functions –called the model's **representational capacity**.

Capacity

- Statistical learning theory provides various means of quantifying model capacity.
  - The most well-known is the **Vapnik-Chervonenkis dimension**.
  - Old non-statistical **Occam's razor method** takes simplest of competing hypotheses.
  - The discrepancy between training error and generalization error is bounded from above by a quantity that grows as the model capacity grows but shrinks as the number of training examples increases.

Generalization Error

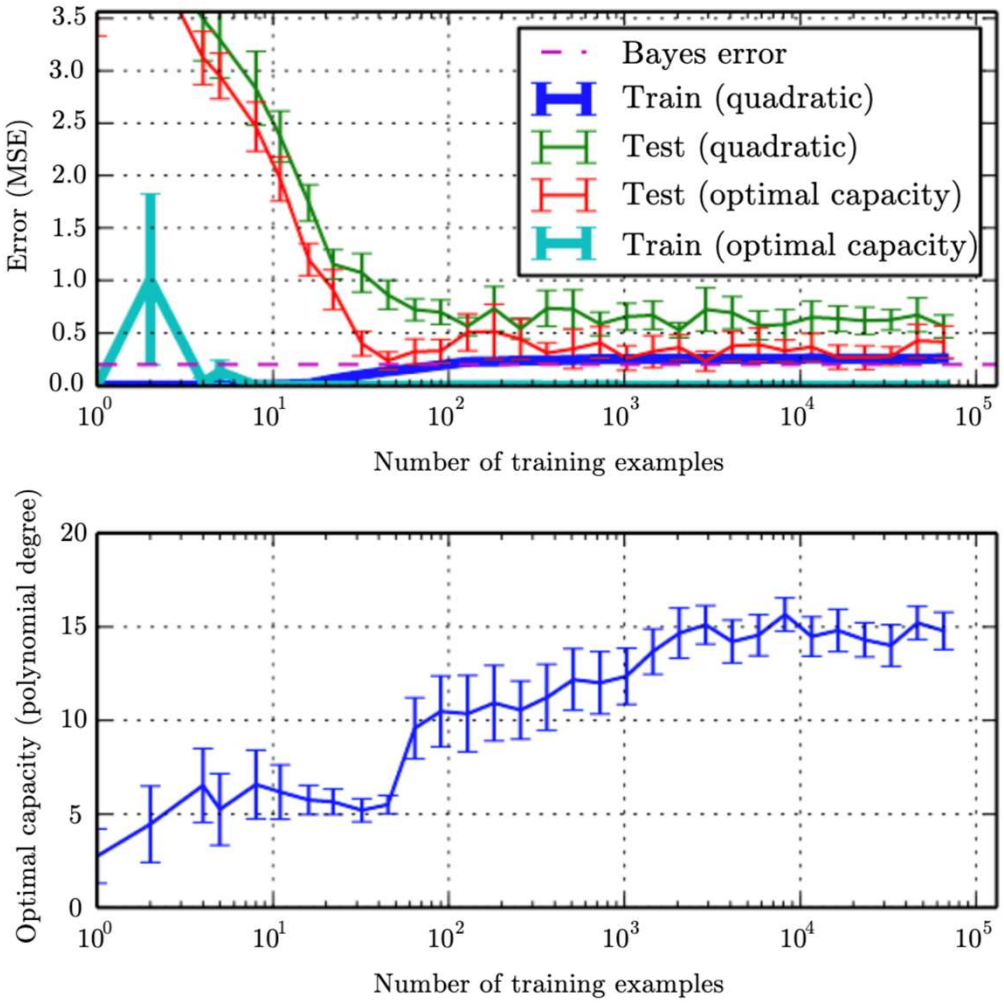
- Training error usually decreases until it asymptotes to the minimum possible error.
- Generalization error usually has a U-shaped curve as a function of model capacity.



Generalization Error

- Parametric models learn a function described by a parameter vector whose size is finite and fixed before any data is observed.
- Non-parametric models can reach arbitrarily high capacities
  - Ex) nearest neighbor regression: the complexity of the nearest neighbor algorithm is a function of the training set size.
- Training and generalization error vary as the size of the training set varies.
  - Test error decreases with increased training set size.

The Effect of the Training Dataset

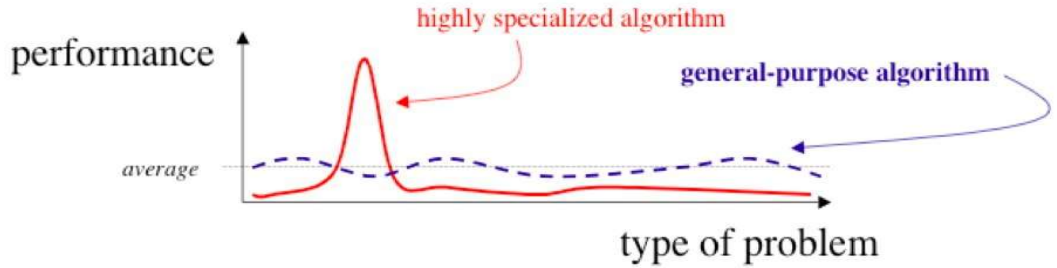


No Free Lunch Theorem

No free lunch theorem

Averaged over **all** possible data generating distributions, every classification algorithm has the same error rate when classifying previously unseen points. (Wolpert, 1996)

- In other words, no machine learning algorithm is universally any better than any other.



- However. By making good assumptions about the kind of probability distributions we encounter in real-world applications, we can design learning algorithms that perform well on these distributions.
  - This is the goal of machine learning research.

Regularization

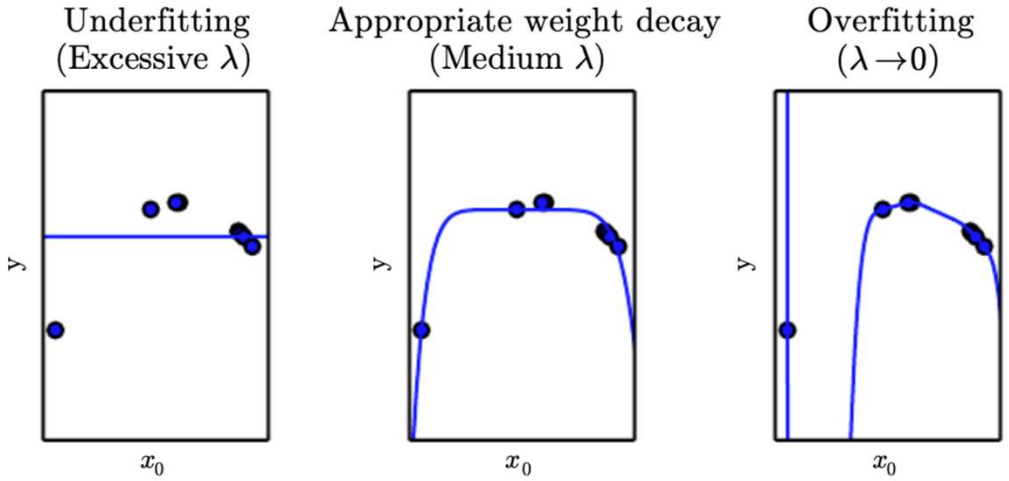
- The no free lunch theorem implies that we design our learning algorithm on a specific task.
- Thus, we build preferences into the algorithm.
- Regularization is a way to do this and weight decay is a method of regularization.

Regularization

- Applying weight decay to linear regression

$$J(\boldsymbol{w}) = \mathrm{MSE}_{\mathrm{train}} + \lambda \boldsymbol{w}^\top \boldsymbol{w}, \tag{5.18}$$

- This expresses a preference for smaller weights
  - λ controls the preference strength
  - $\lambda = 0$ imposes no preference strength.



Regularization

Regularization

Regularization is any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error.

- More generally, we can add a penalty called a regularizer to the cost function.
- Expressing preferences for one function over another is a more general way of controlling a model's capacity, rather than including or excluding members from the hypothesis space.

Hyperparameters

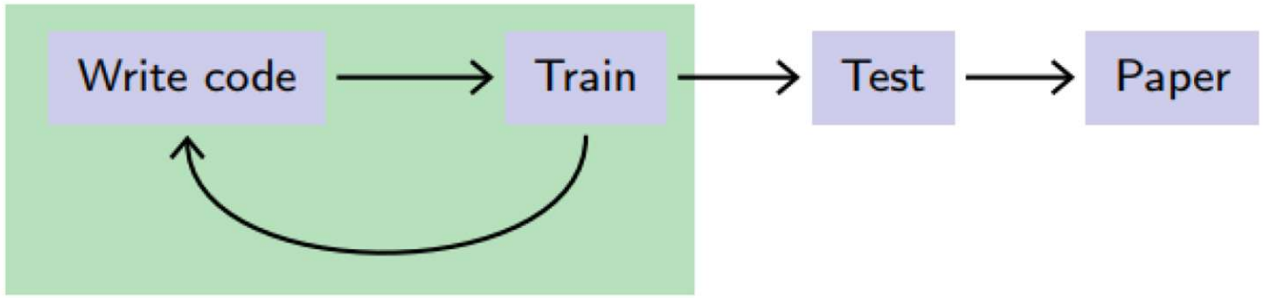
Hyperparameters

- Most machine learning algorithms have several settings, called **hyperparameters**, to control the behavior of the algorithm.

- Examples
  - In polynomial regression, the degree of the polynomial is a capacity hyperparameter.
  - In weight decay, λ is a hyperparameter.

Evaluation Protocol

- Simple training and evaluation

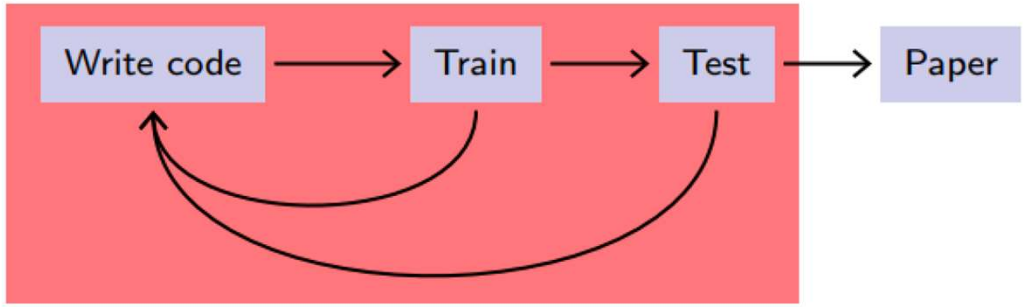


Warning

- The trained model may overfit and fail to generalize.

Evaluation Protocol

- Improper training and evaluation



Warning

- You are cheating. Never try this!

Training & validation Sets

Training and Validation sets

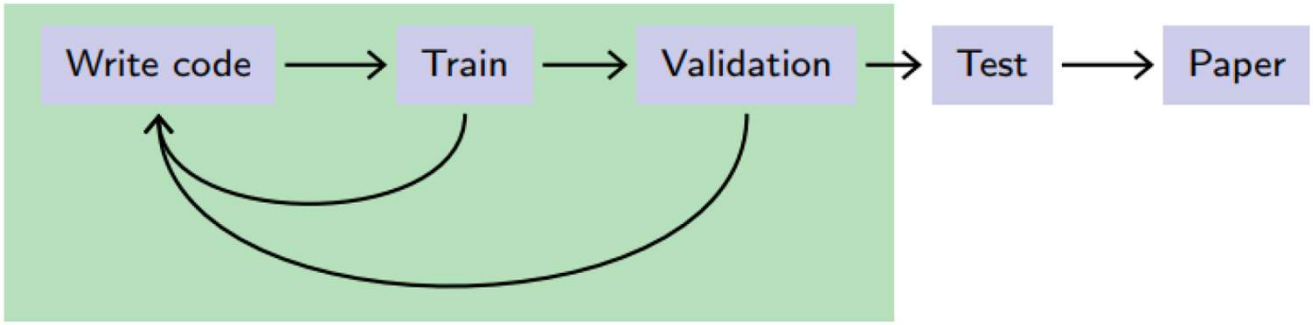
- To avoid overfitting during training, **the training set** is often split into two sets.
- Typically, 80% of the training data is used for training and 20% for validation.

Warning

- Note that repeated evaluation and tuning even without observing the actual data may incur overfitting at the end.

Evaluation Protocol

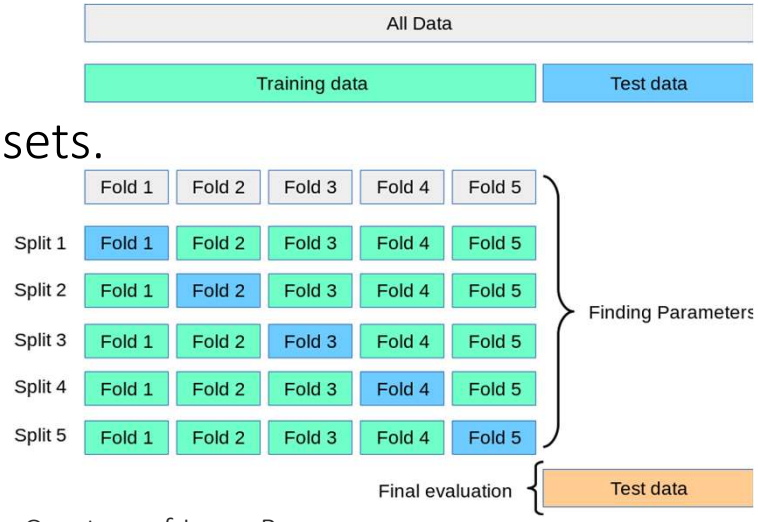
- Proper training and evaluation
  - keep a separate validation set for hyper-parameters tuning and early stop.



  - E.g., given a train set, use 80% of it as a (sub)train split and 20% of it as a validation split.
  - A given test set should be held out for a single (or only a few) final evalution.

Cross-Validation

- *k*-fold cross validation
  - Divide the whole data into *k* non-overlapping subsets.
  - Run *k* round-robin trials:
    - train on *(k-1)/k* of the data
    - test on *1/k.*
  - Estimate the test error by averaging the errors of *k* trails.



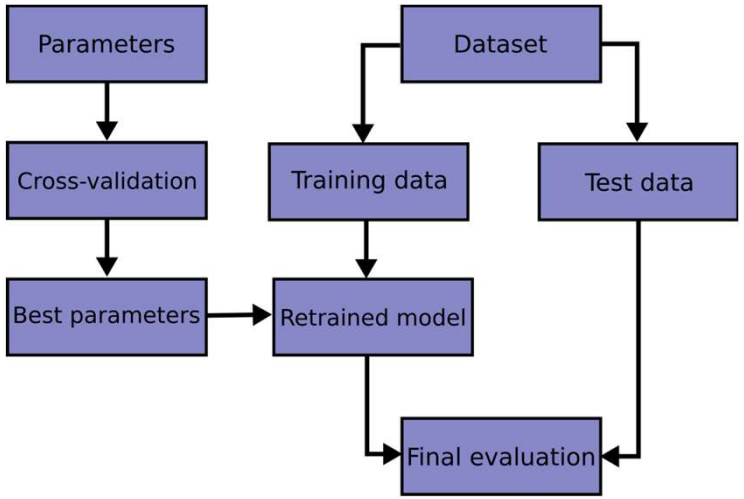
Courtesy of Jonas Benner

Note

- Used when we have a **small training/validation set**, which incurs statistical uncertainty around the estimated error.

Cross-Validation

- Training procedure
  - Given a (small) training dataset, devide it into multiple folds.
  - For each combination of hyperparameters, do cross-validation and measure the average error.
  - Select the best combination in terms of the average error.
  - Retrain the model with it on the whole training set.



Estimators, Bias and Variance

- Field of statistics gives us many tools
  - Solving a task not only on the training set but also to generalize
- Foundational concepts
  - Parameter estimation, bias, variance are useful to formally characterize notions of **generalization**, **underfitting** and **overfitting**

Estimators

Point estimation (Statistic)

Given a i.i.d. data points $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$
A point estimator or statistic is any function of the data:

$$\hat{\boldsymbol{\theta}}_m = g(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}). \tag{5.19}$$

- The single "best" prediction of some quantity of interest
  - Can be a scalar or a vector of parameters, or even a whole function
  - Denote a point estimate of a parameter θ by $\hat{\theta}$
- Frequentist perspective on statistics
  - A good estimator is a function whose output is close to the true underlying θ that generated the training data
  - Since the data is drawn from a random process, $\hat{\theta}$ is a **random variable**

Bias and Variance

Bias of an estimator

Expectation $\text{bias}(\hat{\boldsymbol{\theta}}_m) = \mathbb{E}(\hat{\boldsymbol{\theta}}_m) - \boldsymbol{\theta}$ (5.20)

Unbiased estimator if the bias = 0 and biased estimator otherwise.

Variance and Standard Error of an estimator

$$\text{Var}(\hat{\theta}) = \text{E}[\hat{\theta} - \text{E}(\hat{\theta})] = \text{E}(\hat{\theta}^2) - \text{E}^2(\hat{\theta}).$$

- A measure of how the estimate varies as we independently resample the dataset from the underlying data-generating process.
- The square root of the variance is called the standard error.
- E.g., the standard error of the sample mean is given by

$$\text{SE}(\hat{\mu}_m) = \sqrt{\text{Var}[\frac{1}{m}\sum_{i=1}^{m} x^{(i)}]} = \frac{\sigma}{\sqrt{m}}, \quad (5.46)$$

Estimators of the variance of a Gaussian Distribution

- An estimators of the variance parameter σ^2 of a Gaussian distribution
- Is the sample variance is unbiased estimator?

$$\hat{\sigma}_m^2 = \frac{1}{m}\sum_{i=1}^{m}\left(x^{(i)} - \hat{\mu}_m\right)^2, \tag{5.36}$$

Answer – biased estimator!

$$\text{bias}(\hat{\sigma}_m^2) = \mathbb{E}[\hat{\sigma}_m^2] - \sigma^2 \tag{5.37}$$

$$\mathbb{E}[\hat{\sigma}_m^2] = \mathbb{E}\left[\frac{1}{m}\sum_{i=1}^{m}\left(x^{(i)} - \hat{\mu}_m\right)^2\right] \tag{5.38}$$

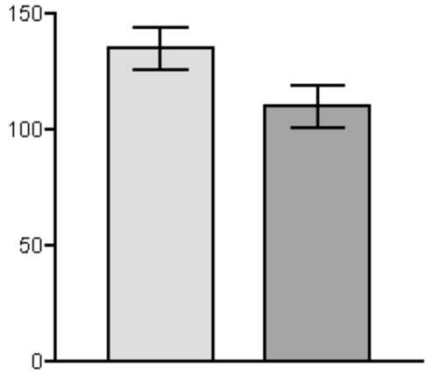
$$= \frac{m-1}{m}\sigma^2 \tag{5.39}$$

Standard Error of the Mean

Usefulness of the *standard error of the mean* in machine learning experiments

- The generalization error often estimated by computing the **sample mean** of the error on the test set.
- Based on the central limit theorem, we can use **standard error** to compute the probability that the true expectation falls in any chosen interval, e.g., the 95 percent confidence interval

$$(\hat{\mu}_m - 1.96\mathrm{SE}(\hat{\mu}_m), \hat{\mu}_m + 1.96\mathrm{SE}(\hat{\mu}_m)), \qquad (5.47)$$



* It is common to say that one algorithm is better than the other if the 95 percent confidence intervals do not overlap.

Mean Squared Error

- One estimator with more bias and one estimator with more variance
  - Which estimator is better?
- Trading off Bias and Variance
  - Most common way to negotiate this trade-off is to use cross-validation.
  - Can also use mean squared error (MSE) of the estimates

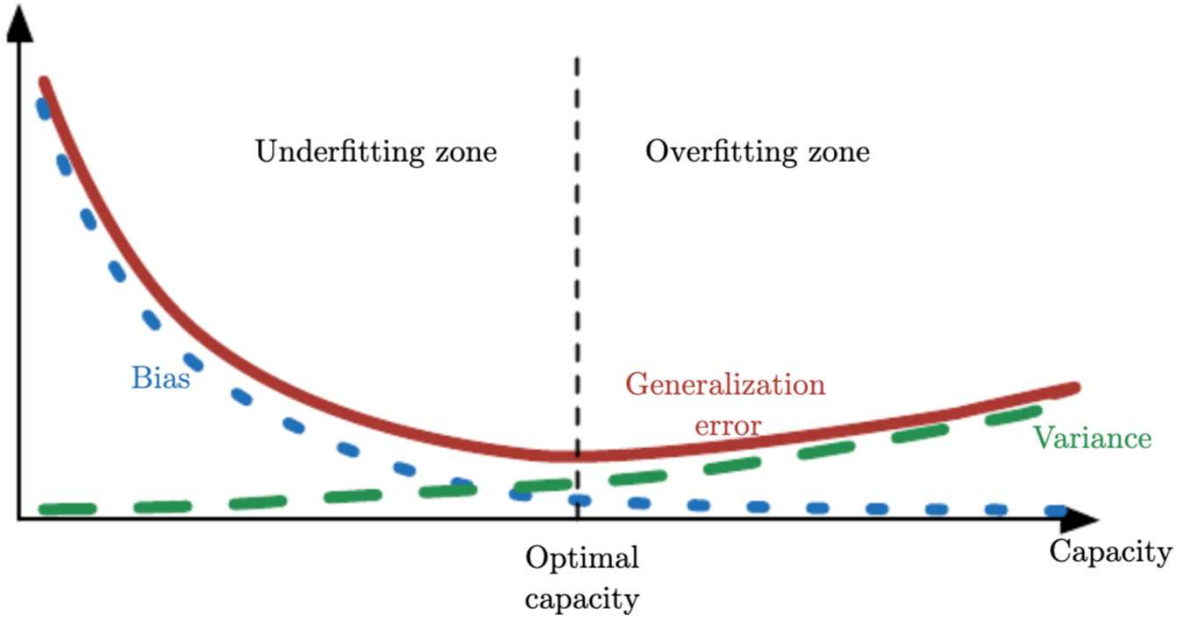
$$\text{MSE} = \mathbb{E}[(\hat{\theta}_m - \theta)^2] \tag{5.53}$$

$$= \text{Bias}(\hat{\theta}_m)^2 + \text{Var}(\hat{\theta}_m) \tag{5.54}$$

Note

- The relationship between **bias** and **variance** is tightly linked to **capacity**, **underfitting**, and **overfitting**.

Trade-off between bias and variance



Note

- When generalization is measured by MSE
- As capacity increases, **bias** tends to **decrease** and **variance** tends to **increase**, yielding U-shaped curve for generalization error.

Consistency

Consistency

- A point estimator is consistent if as the number of data points m in our dataset increases, our point estimates converge in probability to the true value of the corresponding parameters

$$\text{plim}_{m \to \infty} \hat{\theta}_m = \theta. \tag{5.55}$$

  - plim indicates convergence in probablity
  - Consistency ensures that the bias induced by the estimator diminishes as the number of data examples grows
  - However, the reverse is not true – think about this

Maximum Likelihood Principle

- We have seen some definitions of common estimators
- Where did these estimators come from?
- Having some principle from which we can derive specific functions that are good estimators for different models
  - **Maximum likelihood principle** is common

Maximum Likelihood Estimation

- The most common principle to estimation
  - Given a parametric family of probability distributions over the same space indexed by θ,

$$\boldsymbol{\theta}_{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta}} p_{\mathrm{model}}(\mathbb{X}; \boldsymbol{\theta}), \tag{5.56}$$

$$= \arg\max_{\boldsymbol{\theta}} \prod_{i=1}^{m} p_{\mathrm{model}}(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}). \tag{5.57}$$

  - Equivalent to

$$\boldsymbol{\theta}_{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{m} \log p_{\mathrm{model}}(\boldsymbol{x}^{(i)}; \boldsymbol{\theta}). \tag{5.58}$$

$$\boldsymbol{\theta}_{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x} \sim \hat{p}_{\mathrm{data}}} \log p_{\mathrm{model}}(\boldsymbol{x}; \boldsymbol{\theta}). \tag{5.59}$$

MLE and KL divergence

- Interpretation

$$\boldsymbol{\theta}_{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta}} \mathbb{E}_{\mathbf{x}\sim\hat{p}_{\mathrm{data}}} \log p_{\mathrm{model}}(\boldsymbol{x};\boldsymbol{\theta}). \tag{5.59}$$

  - Minimizing the dissimilarity between the empirical distribution and the model distribution.

$$D_{\mathrm{KL}}\left(\hat{p}_{\mathrm{data}} \| p_{\mathrm{model}}\right) = \mathbb{E}_{\mathbf{x}\sim\hat{p}_{\mathrm{data}}}\left[\log \hat{p}_{\mathrm{data}}(\boldsymbol{x}) - \log p_{\mathrm{model}}(\boldsymbol{x})\right]. \tag{5.60}$$

  - The term on the left is a function only of the data-generation process, not the model. Then, we need only minimize the right term → same with (5.59)

Why MLE?

- Under appropriate conditions, the maximum likelihood estimator has the property of consistency. (see 5.5.2)

Conditional Log-Likelihood

- MLE can be generalized to estimate conditional probablity

$$\boldsymbol{\theta}_{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta}} P(\boldsymbol{Y} \mid \boldsymbol{X}; \boldsymbol{\theta}). \tag{5.62}$$

  - If the samples are i.i.d.,

$$\boldsymbol{\theta}_{\mathrm{ML}} = \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{m} \log P(\boldsymbol{y}^{(i)} \mid \boldsymbol{x}^{(i)}; \boldsymbol{\theta}). \tag{5.63}$$

MLE and MSE

- E.g., linear regression as MLE with Gaussian condition

$$\sum_{i=1}^{m} \log p(y^{(i)} \mid \boldsymbol{x}^{(i)}; \boldsymbol{\theta}) \tag{5.64}$$

$$= -m \log \sigma - \frac{m}{2} \log(2\pi) - \sum_{i=1}^{m} \frac{\left\| \hat{y}^{(i)} - y^{(i)} \right\|^2}{2\sigma^2}, \tag{5.65}$$

$$\mathrm{MSE}_{\mathrm{train}} = \frac{1}{m} \sum_{i=1}^{m} ||\hat{y}^{(i)} - y^{(i)}||^2 , \tag{5.66}$$

Note

- (5.65) and (5.66) are actually solving for the same θ!
- MSE can be derived from MLE perspective

Bayesian Statistics

- Frequentist (sec 5.4.1) perspective is that θ is fixed but unknown
- Bayesian makes a prediction considering all possible values of $\boldsymbol{\theta}$ with uncertainty

Bayesian Statistics

- Using probability to reflect degrees of certainty in states of knowledge

$$p(\boldsymbol{\theta} \mid x^{(1)}, \ldots, x^{(m)}) = \frac{p(x^{(1)}, \ldots, x^{(m)} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(x^{(1)}, \ldots, x^{(m)})} \tag{5.67}$$

  - Making predictions using a full distribution over θ.

$$p(x^{(m+1)} \mid x^{(1)}, \ldots, x^{(m)}) = \int p(x^{(m+1)} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid x^{(1)}, \ldots, x^{(m)}) \, d\boldsymbol{\theta}. \tag{5.68}$$

Maximum a Posteriori (MAP) Estimation

- The MAP estimate chooses the point of maximal posterior probability

$$\boldsymbol{\theta}_{\text{MAP}} = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta} \mid \boldsymbol{x}) = \arg\max_{\boldsymbol{\theta}} \log p(\boldsymbol{x} \mid \boldsymbol{\theta}) + \log p(\boldsymbol{\theta}). \qquad (5.79)$$

  - An approximation to a point estimate of the Bayesian
    - Standard log likelihood + **prior distribution**
  - The prior helps to reduce the variance in the MAP point estimate
    - However, it does so at the price of increased bias

Another perspective

- Many regularized estimation strategies, such as maximum likelihood learning regularized with weight decay, can be interpreted as making the MAP approximation to Bayesian inference

Learning Algorithm Examples

- Supervised learning algorithms (Sec 5.7)
  - Logistic regression
  - Support Vector Machines
  - *k*-nearest neighbor algorithm
  - Decision trees – breaks input space into regions
- Unsupervised learning algorithms (Sec 5.8)
  - Principal Component Analysis (PCA)
  - *k*-means Clustering

Stochastic Gradient Descent

- SGD is an extension of gradient descent (in Sec. 4.3)
  - Given the negative log-likelihood of the training data $L(\boldsymbol{x}, y, \boldsymbol{\theta}) = -\log p(y \mid \boldsymbol{x}; \boldsymbol{\theta})$

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{x},\mathrm{y}\sim\hat{p}_{\mathrm{data}}} L(\boldsymbol{x}, y, \boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m} L(\boldsymbol{x}^{(i)}, y^{(i)}, \boldsymbol{\theta}), \tag{5.96}$$

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{m}\sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}} L(\boldsymbol{x}^{(i)}, y^{(i)}, \boldsymbol{\theta}). \tag{5.97}$$

  - Mini-batch SGD $\mathbb{B} = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m')}\}$

$$\boldsymbol{g} = \frac{1}{m'}\nabla_{\boldsymbol{\theta}} \sum_{i=1}^{m'} L(\boldsymbol{x}^{(i)}, y^{(i)}, \boldsymbol{\theta}) \tag{5.98}$$

Note

- Nearly all of deep learning is powered by stochastic gradient descent (SGD)

Building an ML Algorithm

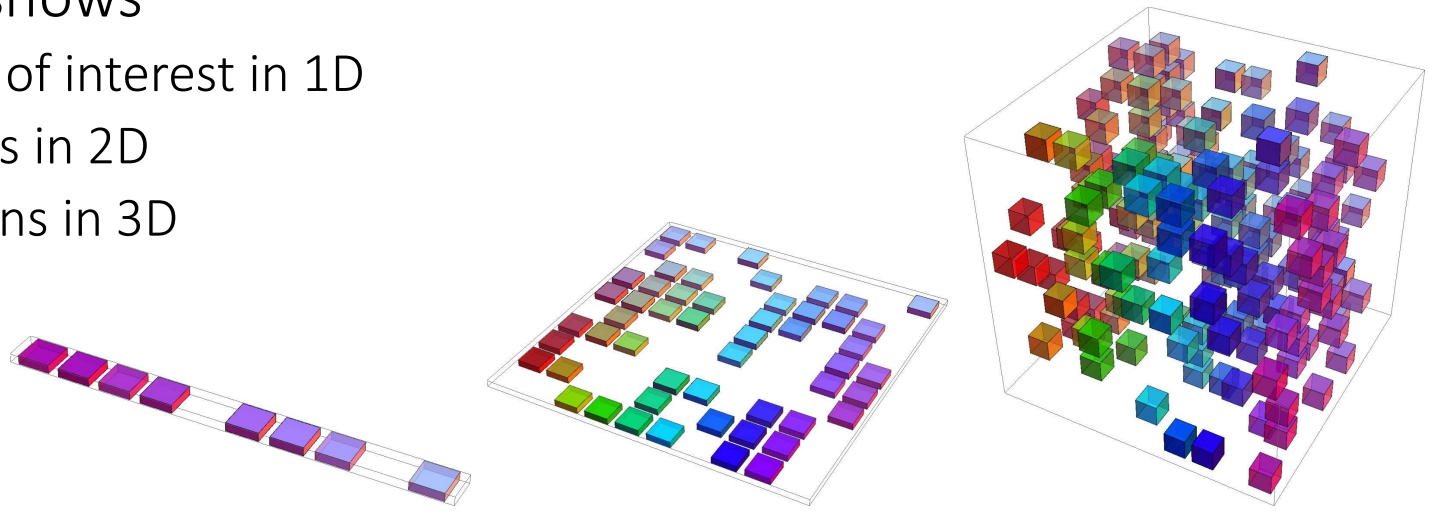
- Deep learning algorithms use a simple recipe.
  - Combine a specification of a **dataset**, a **cost function**, an **optimization procedure**, and a **model**.
- E.g., linear regression algorithm
  - Dataset: $\boldsymbol{X}$ and $\boldsymbol{y}$
  - Cost function: $-\mathbb{E}_{\mathbf{x},\mathrm{y}\sim\hat{p}_{\mathrm{data}}} \log p_{\mathrm{model}}(y \mid \boldsymbol{x}),$ (5.100)
  - Adding regularization: $\lambda||\boldsymbol{w}||_2^2 - \mathbb{E}_{\mathbf{x},\mathrm{y}\sim p_{\mathrm{data}}} \log p_{\mathrm{model}}(y \mid \boldsymbol{x}).$ (5.101)
  - Model: $p_{\mathrm{model}}(y \mid \boldsymbol{x}) = \mathcal{N}(y; \boldsymbol{x}^\top \boldsymbol{w} + b, 1)$
  - Optimization: SGD, the normal equation solver, etc.

Challenges Motivating Deep Learning

- The simple machine learning algorithms work well on many important problems.
- But they don’t solve the central problems of AI
  - Recognizing objects, etc.
- Deep learning was motivated by this challenge.

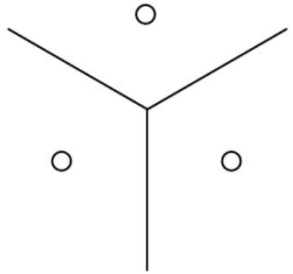
The Curse of Dimensionality

- Simple machine learning problems get difficult when the number of dimensions in the data is high.
- This phenomenon is known as the curse of dimensionality
- Next figure shows
  - 10 regions of interest in 1D
  - 100 regions in 2D
  - 1000 regions in 3D



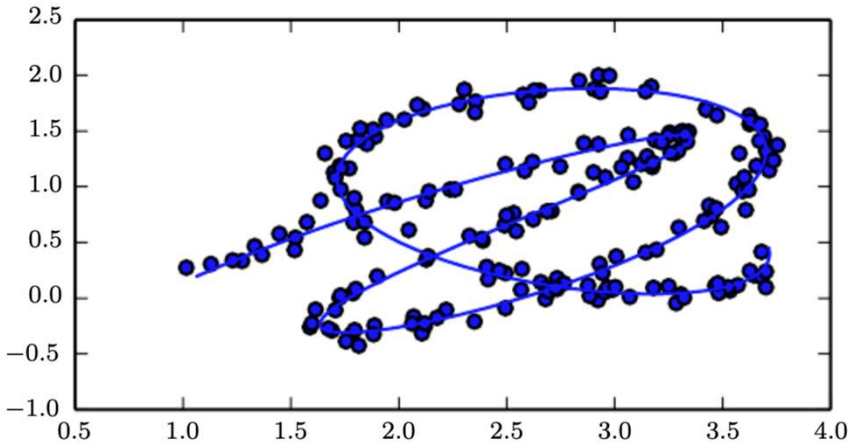
Local Constancy & Smoothness

- To generalize well, machine learning algorithms need to be guided by prior beliefs.
  - One of the most widely used "priors" is the **smoothness prior** or **local constancy prior**.
  - Function should change little within a small region
- Extreme case
  - To distinguish *O(k)* regions in input space requires *O(k)* samples
  - For the *kNN* algorithm, each training sample defines at most one region.



Manifold Learning

- A **manifold** is a connected region.
- Manifold algorithms reduce space of interest
  - Variation across *n*-dim Euclidean space can be reduced by assuming that most of the space consists of invalid input.
  - Probability distributions over images, text strings, and sounds that occur in real life are highly concentrated.



Can we see facial image from random samples?

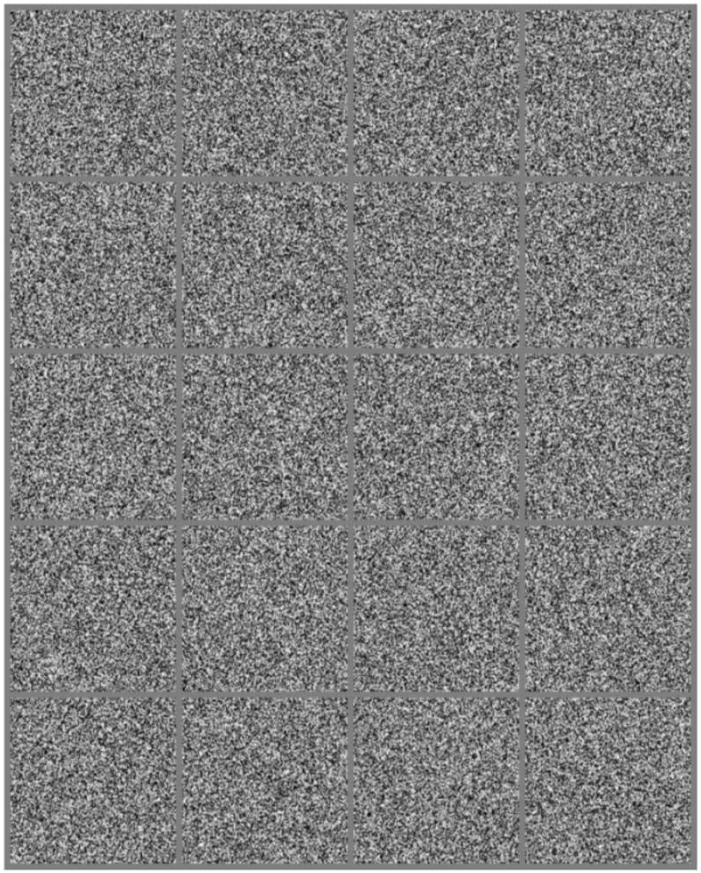


Figure 5.12

Nope.

Structure of face images?

Figure 5.13

Maybe real images are lay on the manifold space.

Overleaf users

Warning

You can ignore this slide if you're **not** working with Overleaf.

Warning

You can ignore this slide if you're **not** working with Overleaf.