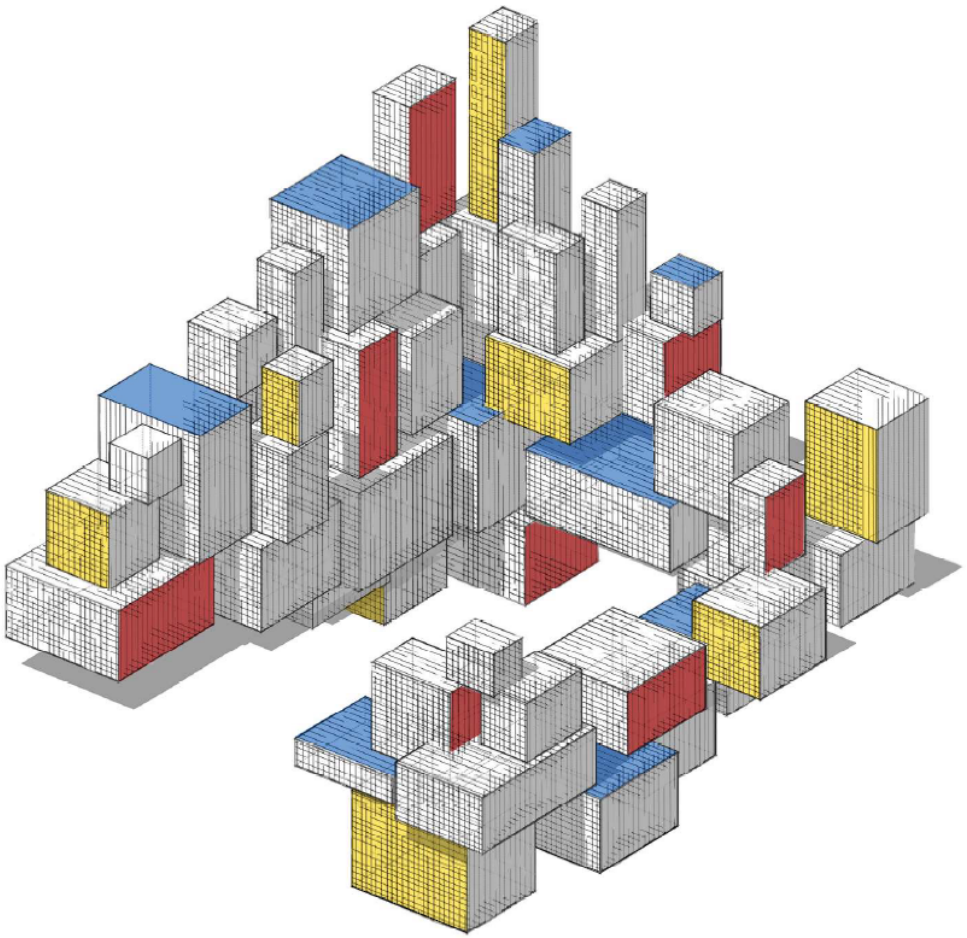
Statistical Learning



Purpose

We introduce some common concepts and themes in statistical learning.

- Supervised learning
- Unsupervised learning
- Assessing the predictive performance of supervised learning

Statistical Learning

A main challenge in data science is the mathematical analysis of the data.

When the goal is to interpret the model and quantify the uncertainty in the data, this analysis is usually referred to as statistical learning.

There are two major goals for modeling data:

- To accurately predict some future quantity of interest, given some observed data.
- To discover unusual or interesting patterns in the data.

Statistical Learning

Given an input or feature vector $\boldsymbol{x}$, one of the main goals of machine learning is to predict an output or response variable y.

For example, $\boldsymbol{x}$ could be a digitized signature and y a binary variable that indicates whether the signature is genuine or false.

Another example is where $\boldsymbol{x}$ represents the weight and smoking habits of an expecting mother and y the birth weight of the baby.
This is formalized via a prediction function, which takes as an input $\boldsymbol{x}$ and outputs a **g**uess $g(\boldsymbol{x})$ for y (denoted by $\widehat{y}$, for example).

In a sense, g encompasses all the information about the relationship between the variables $\boldsymbol{x}$ and y, excluding the effects of chance and randomness in nature.

Regression and Classification

In regression problems, the response variable y can take any real value.

In contrast, when y can only lie in a finite set, say $y \in \{0, \ldots, c-1\}$, then predicting y is conceptually the same as classifying the input $\boldsymbol{x}$ into one of c categories, and so prediction becomes a classification problem.

Loss

We can measure the accuracy of a prediction $\widehat{y}$ with respect to a given response y by using some loss function $\text{Loss}(y, \widehat{y})$.

In a regression setting the usual choice is the squared-error loss $(y - \widehat{y})^2$.

In the case of classification, the zero–one (also written 0–1) loss function $\text{Loss}(y, \widehat{y}) = \mathbb{I}\{y \neq \widehat{y}\}$ is often used, which incurs a loss of 1 whenever the predicted class $\widehat{y}$ is not equal to the class y.

Later on we will encounter various other useful loss functions, such as the cross-entropy and hinge loss functions.

Error

The word error is often used as a measure of distance between a "true" object y and some approximation $\widehat{y}$ thereof.

If y is real-valued, the absolute error $|y - \widehat{y}|$ and the squared error $(y - \widehat{y})^2$ are both well-established error concepts

For vectors, we have instead the norm $\|\boldsymbol{y} - \widehat{\boldsymbol{y}}\|$ and squared norm $\|\boldsymbol{y} - \widehat{\boldsymbol{y}}\|^2$ for.

The squared error $(y - \widehat{y})^2$ is just one example of a loss function.

Risk

It is unlikely that any mathematical function g will be able to make accurate predictions for all possible pairs $(\boldsymbol{x}, y)$.

Even with the same input $\boldsymbol{x}$, the output y may be different, depending on chance circumstances or randomness.

For this reason, we adopt a probabilistic approach and assume that each pair $(\boldsymbol{x}, y)$ is the outcome of a random pair $(\boldsymbol{X}, Y)$ that has some joint probability density $f(\boldsymbol{x}, y)$.

We then assess the predictive performance via the expected loss, usually called the risk, for g:

$$\ell(g) = \mathbb{E}\,\mathrm{Loss}(Y, g(\boldsymbol{X})). \tag{1}$$

Optimal Prediction Function

Given the distribution of $(\boldsymbol{X}, Y)$ and any loss function, the goal is to find the optimal prediction function that yields the smallest risk:

$$g^* = \operatorname*{argmin}_g \underbrace{\mathbb{E}\,\mathrm{Loss}(Y, g(\boldsymbol{X}))}_{\ell(g)}.$$

In the classification case with zero–one loss function the risk is equal to the probability of incorrect classification: $\ell(g) = \mathbb{P}[Y \neq g(\boldsymbol{X})]$.

In this context, the prediction function g is called a classifier.

Optimal Prediction Function

For classification with 0-1 loss, we have:

$$g^*(\boldsymbol{x}) = \underset{y \in \{0,\ldots,c-1\}}{\operatorname{argmax}} \; f(y \mid \boldsymbol{x}),$$

where $f(y \mid \boldsymbol{x}) = \mathbb{P}[Y = y \mid \boldsymbol{X} = \boldsymbol{x}]$ is the conditional probability of $Y = y$ given $\boldsymbol{X} = \boldsymbol{x}$.

For regression with the squared-error loss, we have:

Theorem: Optimal Prediction Function

For the squared-error loss $\text{Loss}(y, \widehat{y}) = (y - \widehat{y})^2$, the optimal prediction function g^* is equal to the conditional expectation of Y given $\boldsymbol{X} = \boldsymbol{x}$:

$$g^*(\boldsymbol{x}) = \mathbb{E}[Y \mid \boldsymbol{X} = \boldsymbol{x}].$$

Proof.

Let $g^*(\boldsymbol{x}) = \mathbb{E}[Y \mid \boldsymbol{X} = \boldsymbol{x}]$. For any function g, the squared-error risk satisfies

$$\begin{aligned}
\mathbb{E}(Y - g(\boldsymbol{X}))^2 &= \mathbb{E}[(Y - g^*(\boldsymbol{X}) + g^*(\boldsymbol{X}) - g(\boldsymbol{X}))^2] \\
&= \mathbb{E}(Y - g^*(\boldsymbol{X}))^2 + 2\mathbb{E}[(Y - g^*(\boldsymbol{X}))(g^*(\boldsymbol{X}) - g(\boldsymbol{X}))] + \mathbb{E}(g^*(\boldsymbol{X}) - g(\boldsymbol{X}))^2 \\
&\geqslant \mathbb{E}(Y - g^*(\boldsymbol{X}))^2 + 2\mathbb{E}[(Y - g^*(\boldsymbol{X}))(g^*(\boldsymbol{X}) - g(\boldsymbol{X}))] \\
&= \mathbb{E}(Y - g^*(\boldsymbol{X}))^2 + 2\mathbb{E}\left\{(g^*(\boldsymbol{X}) - g(\boldsymbol{X}))\mathbb{E}[Y - g^*(\boldsymbol{X}) \mid \boldsymbol{X}]\right\}.
\end{aligned}$$

In the last equation we used the tower property. By the definition of the conditional expectation, we have $\mathbb{E}[Y - g^*(\boldsymbol{X}) \mid \boldsymbol{X}] = 0$. It follows that $\mathbb{E}(Y - g(\boldsymbol{X}))^2 \geqslant \mathbb{E}(Y - g^*(\boldsymbol{X}))^2$, showing that g^* yields the smallest squared-error risk. □

Training Set

Since, the optimal prediction function g^* depends on the typically unknown joint distribution of $(\boldsymbol{X}, Y)$, it is not available in practice.

Instead, all that we have available is a finite number of (usually) independent realizations from the joint density $f(\boldsymbol{x}, y)$.

We denote this sample by $\mathcal{T} := \{(\boldsymbol{X}_1, Y_1), \ldots, (\boldsymbol{X}_n, Y_n)\}$ and call it the training set ($\mathcal{T}$ is a mnemonic for **t**raining) with n examples.

It will be important to distinguish between a random training set $\mathcal{T}$ and its (deterministic) outcome $\tau := \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$.

Supervised Learning

In supervised learning we try to learn the functional relationship g^* between the feature vector (explanatory variable) $\boldsymbol{x}$ and response y in the presence of a teacher who provides a training set $\mathcal{T}$ with n examples.

Denote by $g_{\mathcal{T}}$ the best (by some criterion) approximation for g^* that we can construct from $\mathcal{T}$.

Note that this so-called learner $g_{\mathcal{T}}$ is a random function. A particular outcome is denoted by g_τ.

The main supervised learning methodologies are regression and classification.

More advanced supervised learning techniques, including reproducing kernel Hilbert spaces, tree methods, and deep learning.

Unsupervised Learning

In unsupervised learning we make no distinction between response and explanatory variables, and the objective is simply to learn the structure of the unknown distribution of the data.

In other words, we need to learn $f(\boldsymbol{x})$.

In this case the guess $g(\boldsymbol{x})$ is an approximation of $f(\boldsymbol{x})$ and the risk is of the form

$$\ell(g) = \mathbb{E}\,\mathrm{Loss}(f(\boldsymbol{X}), g(\boldsymbol{X})).$$

The main methodologies for unsupervised learning include clustering, principal component analysis, and kernel density estimation.

Training Loss

Given an arbitrary prediction function g, it is typically not possible to compute its risk $\ell(g) = \mathbb{E}\,\text{Loss}(Y, g(\boldsymbol{X}))$.

However, using the training sample $\mathcal{T}$, we can approximate $\ell(g)$ via the empirical (sample average) risk

$$\ell_{\mathcal{T}}(g) = \frac{1}{n}\sum_{i=1}^{n} \text{Loss}(Y_i, g(\boldsymbol{X}_i)), \tag{2}$$

which we call the training loss.

The training loss is thus an unbiased estimator of the risk (the expected loss) for a prediction function g, based on the training data.

Learner

To approximate the optimal prediction function g^* (the minimizer of the risk $\ell(g)$) we

1. first select a suitable collection of approximating functions $\mathcal{G}$ and then
2. take our learner to be the function in $\mathcal{G}$ that minimizes the training loss; that is,

$$g_{\mathcal{T}}^{\mathcal{G}} = \underset{g \in \mathcal{G}}{\operatorname{argmin}}\, \ell_{\mathcal{T}}(g). \tag{3}$$

For example, the simplest and most useful $\mathcal{G}$ is the set of linear functions of $\boldsymbol{x}$; that is, the set of all functions $g : \boldsymbol{x} \mapsto \boldsymbol{\beta}^\top \boldsymbol{x}$ for some real-valued vector $\boldsymbol{\beta}$.

Overfitting

Note that minimizing the training loss over *all* possible functions g (rather than over all $g \in \mathcal{G}$) does not lead to a meaningful optimization problem, as any function g for which $g(\boldsymbol{X}_i) = Y_i$ for all i gives minimal training loss.

In particular, for a squared-error loss, the training loss will be 0. Unfortunately, such functions have a poor ability to predict new (that is, independent from $\mathcal{T}$) pairs of data. This poor generalization performance is called overfitting.

> By choosing g a function that predicts the training data exactly (and is, for example, 0 otherwise), the squared-error training loss is zero. Minimizing the training loss is not the ultimate goal!

Generalization Risk

The prediction accuracy of new pairs of data is measured by the generalization risk of the learner. For a fixed training set τ it is defined as

$$\ell(g_\tau^{\mathcal{G}}) = \mathbb{E}\,\mathrm{Loss}(Y, g_\tau^{\mathcal{G}}(\boldsymbol{X})), \tag{4}$$

where $(\boldsymbol{X}, Y)$ is distributed according to $f(\boldsymbol{x}, y)$.

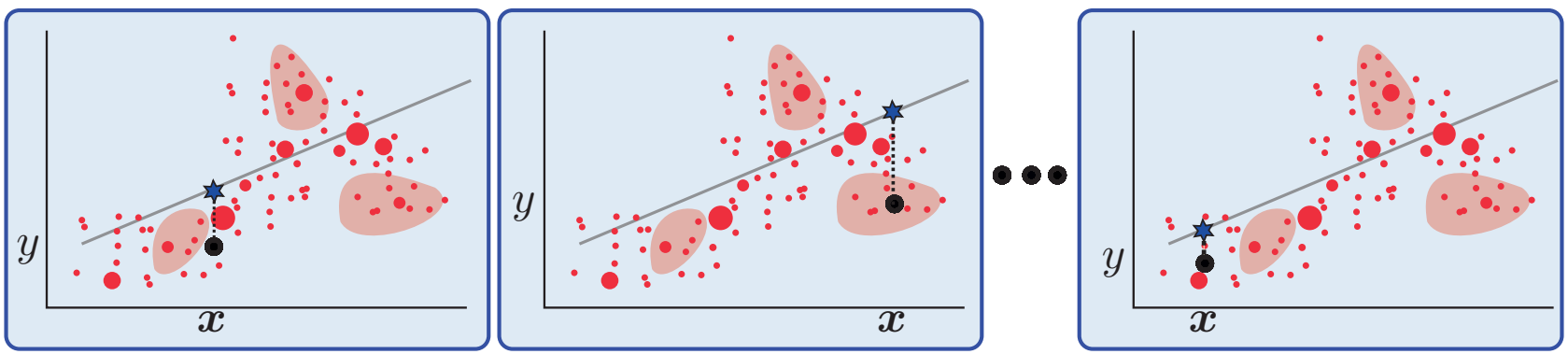


Figure: The generalization risk for a fixed training set is the weighted-average loss over all possible pairs $(\boldsymbol{x}, y)$.

Expected Generalization Risk

For a random training set $\mathcal{T}$, the generalization risk is thus a random variable that depends on $\mathcal{T}$ (and $\mathcal{G}$). If we average the generalization risk over all possible instances of $\mathcal{T}$, we obtain the expected generalization risk:

$$\mathbb{E}\,\ell(g_{\mathcal{T}}^{\mathcal{G}}) = \mathbb{E}\,\mathrm{Loss}(Y, g_{\mathcal{T}}^{\mathcal{G}}(\boldsymbol{X})), \tag{5}$$

where $(\boldsymbol{X}, Y)$ in the expectation above is independent of $\mathcal{T}$.

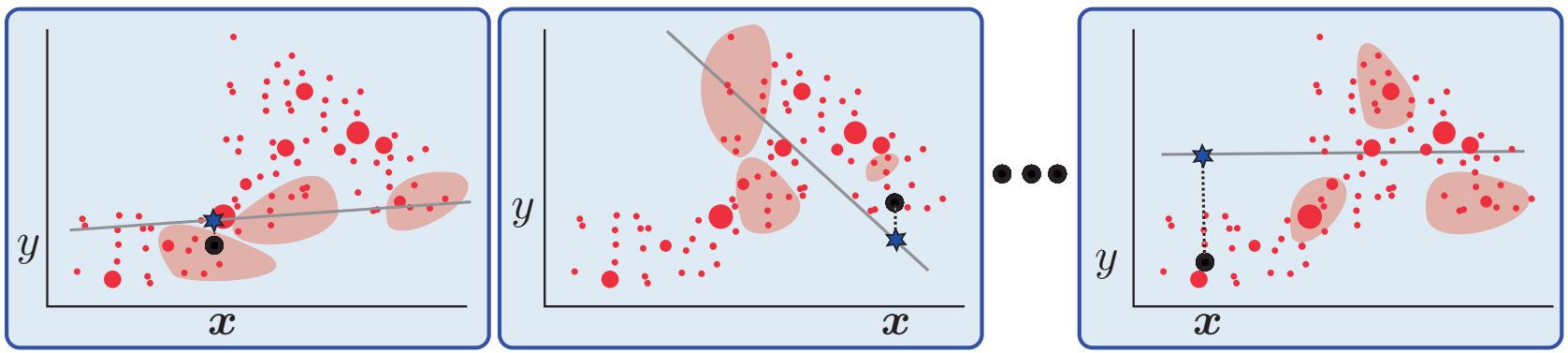


Figure: The expected generalization risk is the weighted-average loss over all possible pairs $(\boldsymbol{x}, y)$ and over all training sets.

Test Loss

For any outcome τ of the training data, we can estimate the generalization risk without bias by taking the sample average

$$\ell_{\mathcal{T}'}(g_\tau^{\mathcal{G}}) := \frac{1}{n'} \sum_{i=1}^{n'} \text{Loss}(Y_i', g_\tau^{\mathcal{G}}(\boldsymbol{X}_i')), \tag{6}$$

where $\{(\boldsymbol{X}_1', Y_1'), \ldots, (\boldsymbol{X}_{n'}', Y_{n'}')\} =: \mathcal{T}'$ is a so-called test sample. The test sample is completely separate from $\mathcal{T}$, but is drawn in the same way as $\mathcal{T}$; that is, via independent draws from $f(\boldsymbol{x}, y)$, for some sample size n'. We call the estimator (6) the test loss.

For a random training set $\mathcal{T}$ we can define $\ell_{\mathcal{T}'}(g_{\mathcal{T}}^{\mathcal{G}})$ similarly. It is then crucial to assume that $\mathcal{T}$ is independent of $\mathcal{T}'$.

Test and Validation Sets

To compare the predictive performance of different learners (as measured by the test loss), we can use the same fixed training set τ and test set τ' for all learners.

When there is an abundance of data, the "overall" data set is usually (randomly) divided into a training and test set. If the latter is used for model selection, a third set is needed for testing the performance of the selected model.

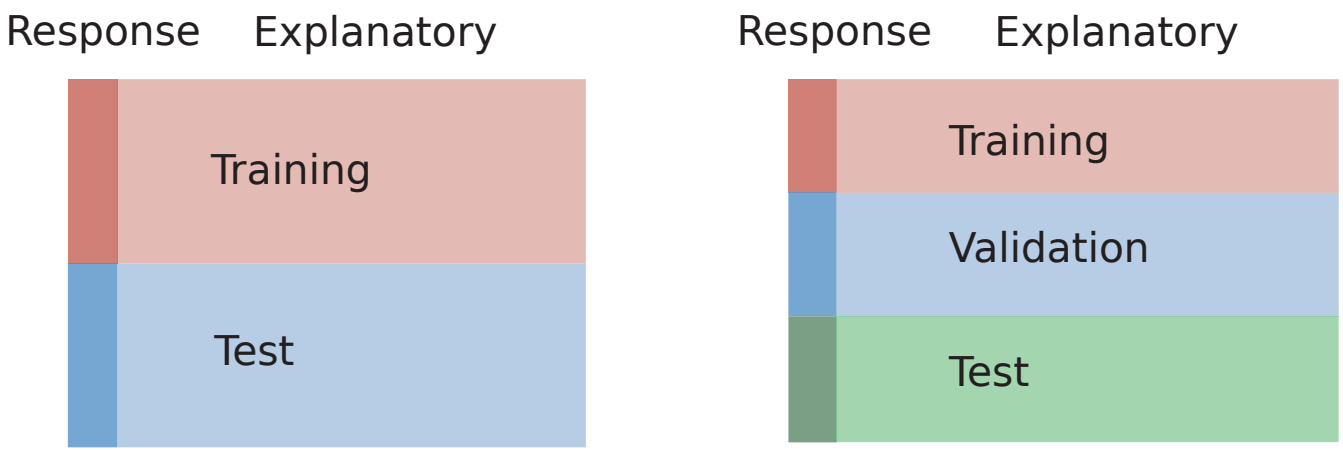


Figure: Statistical learning algorithms often require the data to be divided into various sets.

Notation

$\boldsymbol{x}$ and $\boldsymbol{X}$	Fixed and random explanatory (feature) vector.
y and Y	Fixed and random response.
$f(\boldsymbol{x}, y)$	Joint pdf of $\boldsymbol{X}$ and Y.
τ and $\mathcal{T}$	Fixed and random training data $\{(\boldsymbol{X}_i, Y_i)\}$.
$\mathbf{X}$	Matrix of explanatory variables, with rows $\{\boldsymbol{x}_i^\top\}$.
g	Prediction (guess) function.
$\text{Loss}(y, \widehat{y})$	Loss incurred when predicting response y with $\widehat{y}$.
$\ell(g)$	Risk for g, i.e., $\mathbb{E}\,\text{Loss}(Y, g(\boldsymbol{X}))$.
g^*	Optimal prediction function; i.e., $\text{argmin}_g\, \ell(g)$.
$\ell_\tau(g)$	Training loss for g, i.e., the sample average estimate of $\ell(g)$ based on τ.
$g_\tau^{\mathcal{G}}$	The learner: $\text{argmin}_{g \in \mathcal{G}}\, \ell_\tau(g)$.

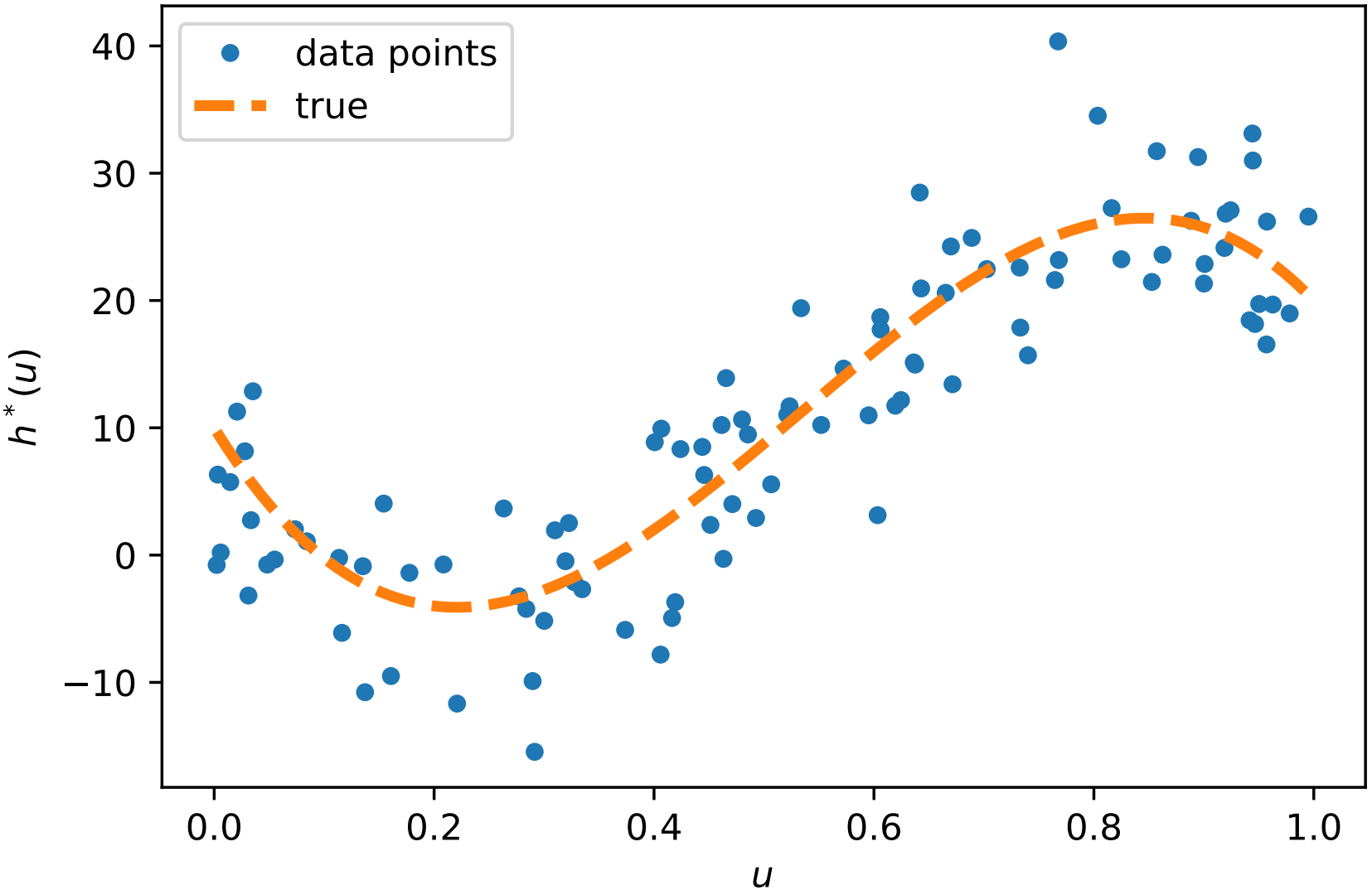
Finding Good Learners

To find good learners, one must rely on knowledge from three important pillars of the mathematical sciences.

1. **Function approximation.** This is a natural way to represent a relationship between variables. We usually assume that this mathematical function is not completely known, but can be approximated well given enough computing power and data.
2. **Optimization.** Given a class of approximating functions, we wish to find the best possible function in that class. This requires some kind of efficient search or optimization procedure.
3. **Probability and Statistics.** In general, the data used to fit the model is viewed as a realization of a random process or numerical vector, whose probability law determines the accuracy with which we can predict future observations.

Example

Data: $(U_i, Y_i), i = 1, \ldots, 100$, where the $\{U_i\} \sim_{\text{iid}} \mathcal{U}(0, 1)$ and, given $U_i = u_i$, $Y_i \sim \mathcal{N}(10 - 140u_i + 400u_i^2 - 250u_i^3,\ 25)$.



Polynomial Regression

This is an example of a polynomial regression model.

Using a squared-error loss, the optimal prediction function $h^*(u) = \mathbb{E}[Y \mid U = u]$ is thus

$$h^*(u) = 10 - 140u + 400u^2 - 250u^3.$$

To obtain a good estimate of $h^*(u)$ based on the training set $\tau = \{(u_i, y_i), i = 1, \ldots, n\}$, we minimize the outcome of the squared-error training loss:

$$\ell_\tau(h) = \frac{1}{n} \sum_{i=1}^{n} (y_i - h(u_i))^2,$$

over a suitable set $\mathcal{H}$ of candidate functions.

Polynomial Regression

Let us take the set $\mathcal{H}_p$ of polynomial functions in u of order $p-1$:

$$h(u) := \beta_1 + \beta_2 u + \beta_3 u^2 + \cdots + \beta_p u^{p-1} \tag{7}$$

for $p = 1, 2, \ldots$ and parameter vector $\boldsymbol{\beta} = [\beta_1, \beta_2, \ldots, \beta_p]^\top$.

Note that optimization over $\mathcal{H}_p$ is a parametric optimization problem, in that we need to find the best $\boldsymbol{\beta}$.

If we map each feature u to a feature vector $\boldsymbol{x} = [1, u, u^2, \ldots, u^{p-1}]^\top$, then the right-hand side of (7) can be written as the function

$$g(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{\beta},$$

which is linear in $\boldsymbol{x}$ (as well as $\boldsymbol{\beta}$).

Linear Prediction Function

Thus, instead of working with the set $\mathcal{H}_p$ of polynomial functions we may prefer to work with the set $\mathcal{G}_p$ of linear functions. This brings us to a very important idea in statistical learning:

> Expand the feature space to obtain a *linear* prediction function.

Let us now reformulate the learning problem in terms of the new explanatory (feature) variables $\boldsymbol{x}_i = [1, u_i, u_i^2, \ldots, u_i^{p-1}]^\top$, $i = 1, \ldots, n$. It will be convenient to arrange these feature vectors into a matrix $\mathbf{X}$ with rows $\boldsymbol{x}_1^\top, \ldots, \boldsymbol{x}_n^\top$ (the so-called model matrix):

$$\mathbf{X} = \begin{bmatrix} 1 & u_1 & u_1^2 & \cdots & u_1^{p-1} \\ 1 & u_2 & u_2^2 & \cdots & u_2^{p-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & u_n & u_n^2 & \cdots & u_n^{p-1} \end{bmatrix}.$$

Least Squares Solution

Collecting the responses $\{y_i\}$ into a column vector $\boldsymbol{y}$, the training loss can now be written compactly as

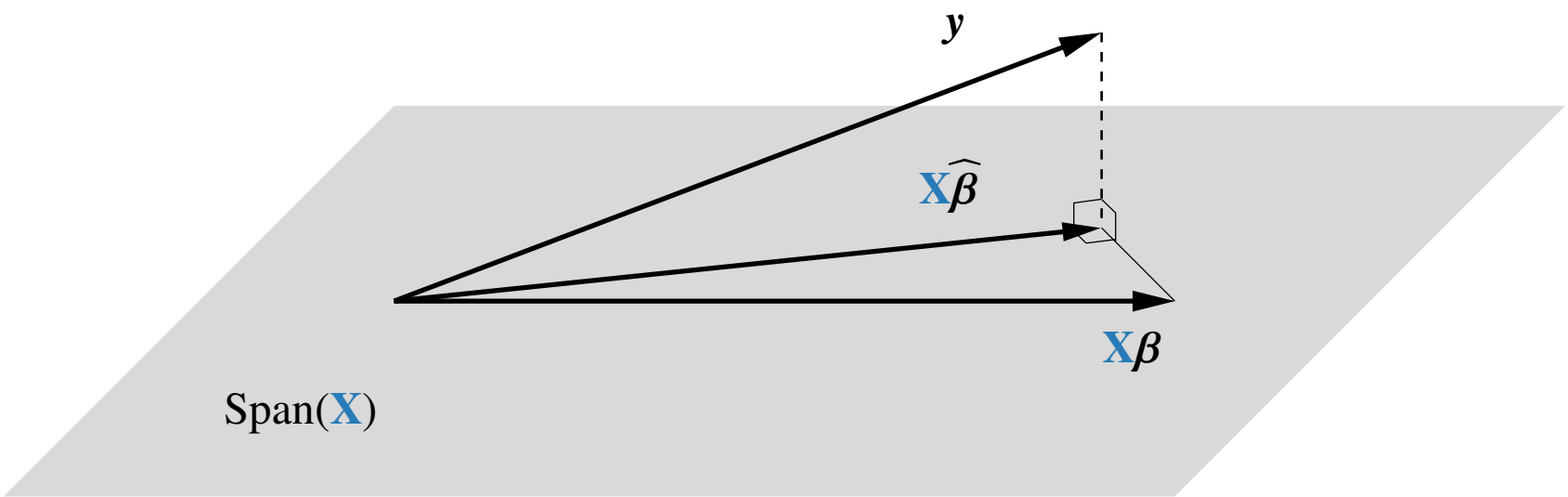
$$\frac{1}{n} \|\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta}\|^2. \tag{8}$$

To find the optimal learner in the class $\mathcal{G}_p$ we need to find the minimizer of (8):

$$\widehat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \|\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta}\|^2,$$

which is called the *ordinary least-squares* solution.

Projection Matrix



To find $\widehat{\boldsymbol{\beta}}$, we choose $\mathbf{X}\widehat{\boldsymbol{\beta}}$ to be equal to the orthogonal projection of $\boldsymbol{y}$ onto the linear space spanned by the columns of the matrix $\mathbf{X}$

That is, $\mathbf{X}\widehat{\boldsymbol{\beta}} = \mathbf{P}\boldsymbol{y}$, where $\mathbf{P}$ is the projection matrix.

Pseudo-Inverse

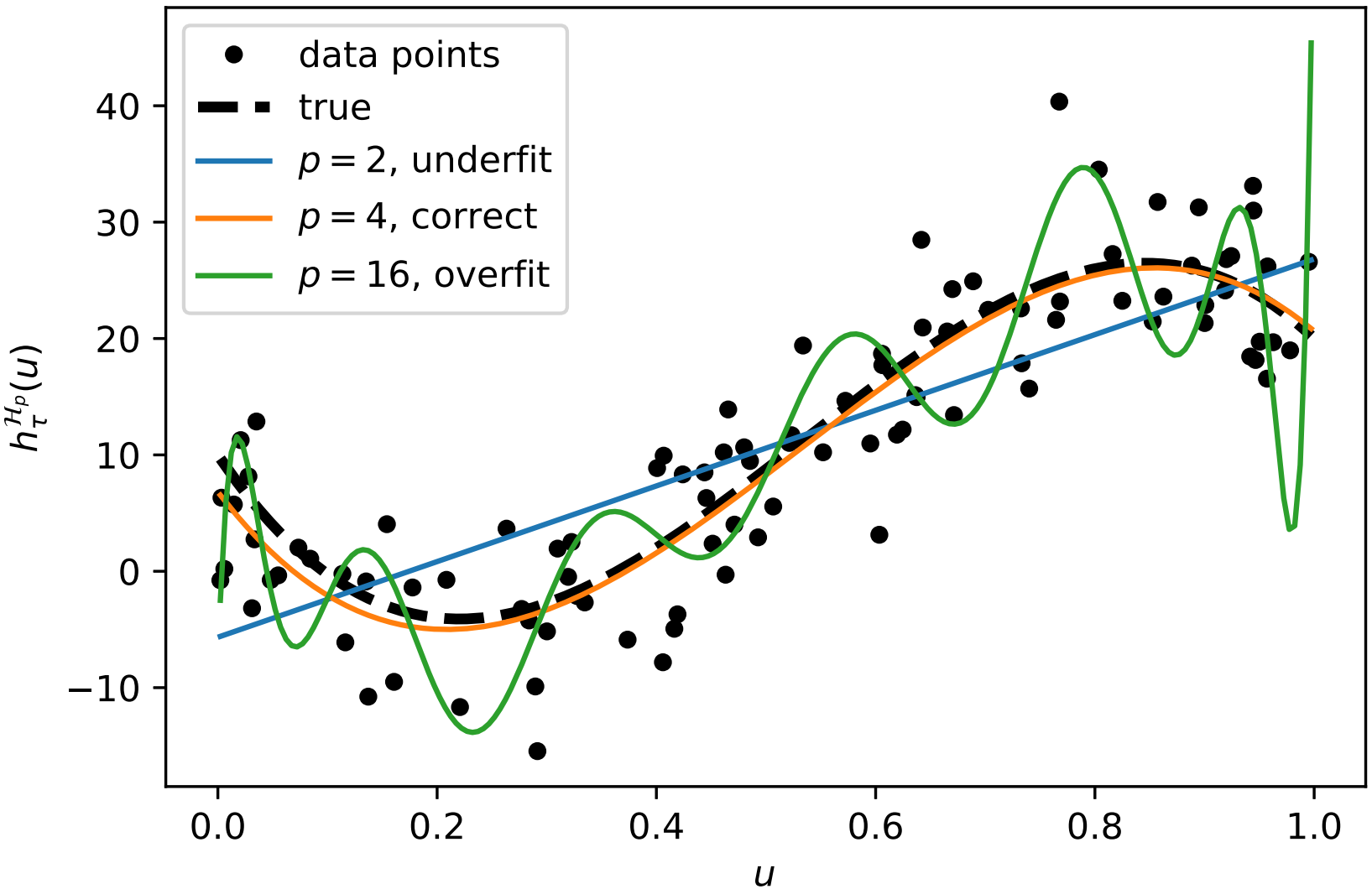
The projection matrix is given by

$$\mathbf{P} = \mathbf{X}\,\mathbf{X}^{+},$$

where the $p \times n$ matrix $\mathbf{X}^{+}$ is the pseudo-inverse of $\mathbf{X}$.

If $\mathbf{X}$ happens to be of *full column rank* (so that none of the columns can be expressed as a linear combination of the other columns), then $\mathbf{X}^{+} = (\mathbf{X}^{\top}\mathbf{X})^{-1}\mathbf{X}^{\top}$.

Fitted Curves



Model Selection

We see that for $p = 16$ the fitted curve lies closer to the data points, but is further away from the dashed true polynomial curve, indicating that we overfit. The choice $p = 4$ (the true cubic polynomial) is much better than $p = 16$, or indeed $p = 2$ (straight line).

Each function class $\mathcal{G}_p$ gives a different learner $g_\tau^{\mathcal{G}_p}$, $p = 1, 2, \ldots$. To assess which is better, we assess the predictive performance of the learners using the test loss, computed from a test data set.

If we collect all n' test feature vectors in a matrix $\mathbf{X}'$ and the corresponding test responses in a vector $\boldsymbol{y}'$, then, similar to (8), the test loss can be written compactly as

$$\ell_{\tau'}(g_\tau^{\mathcal{G}_p}) = \frac{1}{n'} \|\boldsymbol{y}' - \mathbf{X}'\widehat{\boldsymbol{\beta}}\|^2,$$

where $\widehat{\boldsymbol{\beta}}$ is found from the training data.

Test Loss

Note the characteristic "bath-tub" shape of the test loss (estimate of the generalization risk), being lowest for $p = 4$ (corresponding to the true model).

Here, the test loss becomes numerically unreliable after $p = 16$.

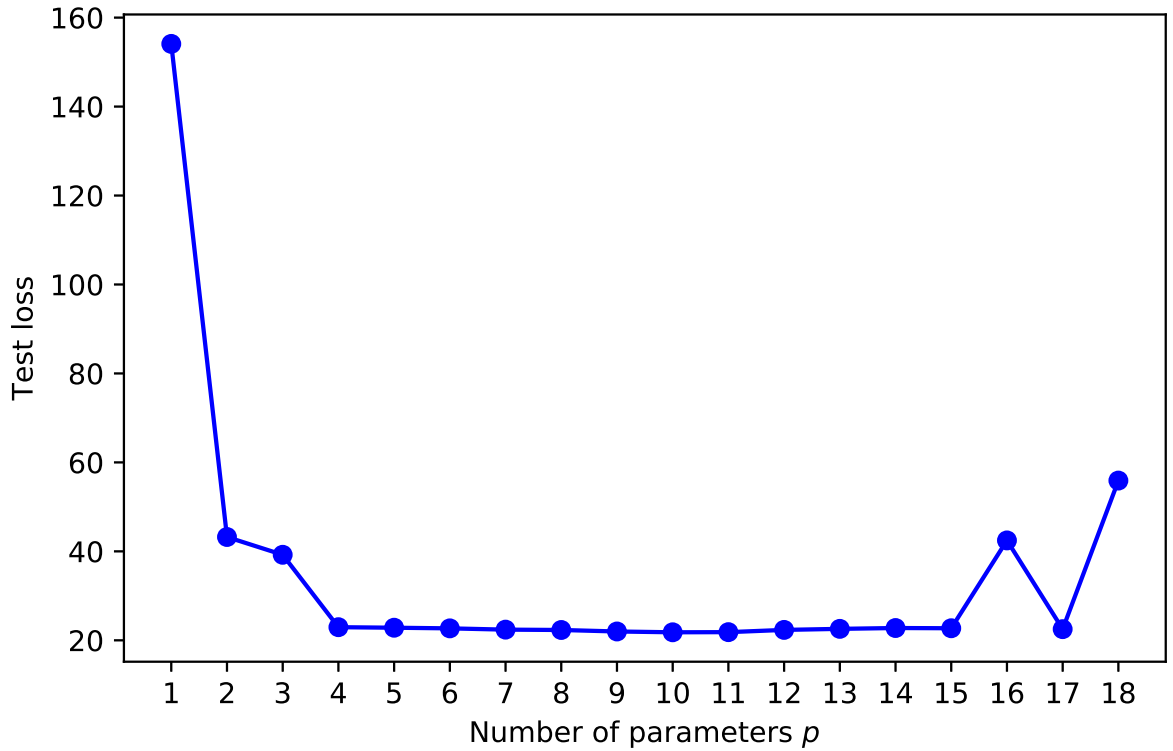


Figure: Test loss as function of the number of parameters p of the model.