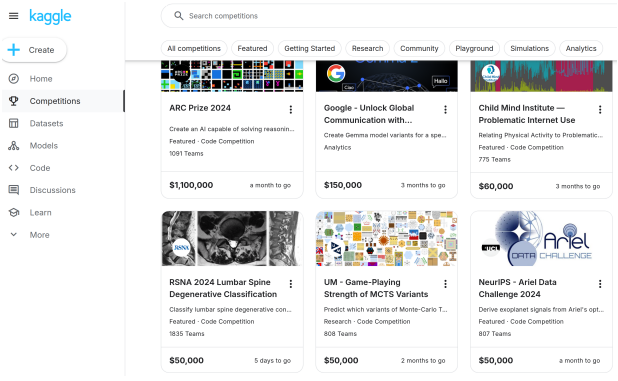
Model Assessment, Hyperparameter Tuning, Feature Engineering

Johanni Brea

Introduction to Machine Learning

What Would You Do to Win a Kaggle Competition?



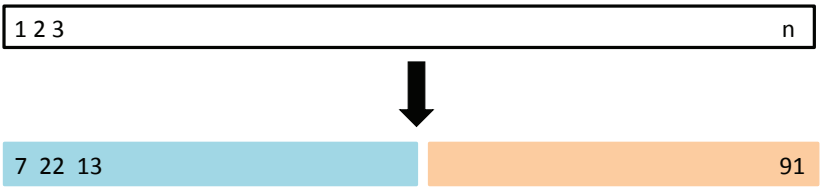
- You are given training set $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ and test inputs $\{x_{n+1}, \ldots, x_{n+m}\}$.
- You test different machine learning methods; each method makes another prediction $\{\hat{y}_{n+1}, \ldots, \hat{y}_{n+m}\}$ on the test data.
- The prediction from which method would you submit to kaggle?

Table of Contents

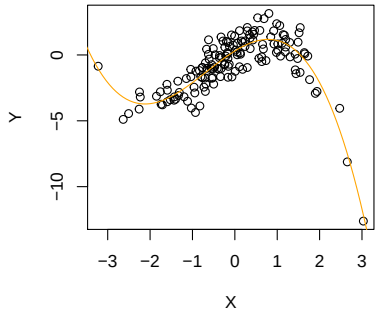
1. Training, Validation and Test Set
2. Cross-Validation
3. Tuning Models
4. Data Cleaning
5. Feature Engineering
6. Transformations of the Output
7. A Recipe for Supervised Learning

The Validation Set Approach

shuffle the data points (each number indicates one row in a data frame) and split into two parts.

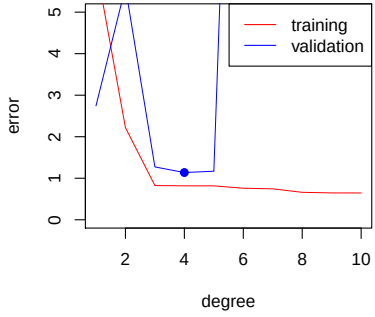


Validation Set Approach Applied to Artificial Data



data generator

$Y = 0.3 + 2X - 0.8X^2 - 0.4X^3 + \epsilon$



polynomial fits with different degrees d

"optimal" $d = 4$ (lowest validation error)

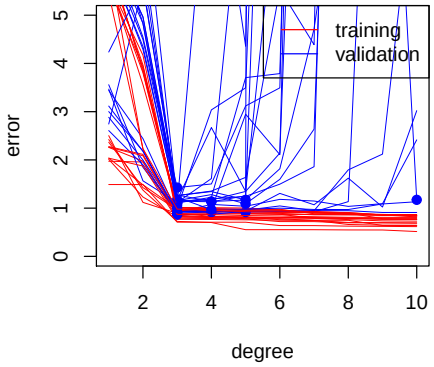
Training, Validation and Test Set

- **Training Set**: Subset of the full data used to find the parameters.
- **Validation Set**: Held-out subset of the full data used for model selection, i.e. finding the hyper-parameters.
- **Test Set**: Held-out subset of the data to estimate the test error of the best model.

Machine Learning Competitions e.g. on kaggle.com

1. Start of the competition: Participants obtain a data set, but not the test set.
2. Participants split the data set into training and validation sets as they want to fit the parameters and tune the hyper-parameters.
3. End of the competition: Organizers evaluate all submitted solutions on the test set.

Drawback of Validation Set Approach

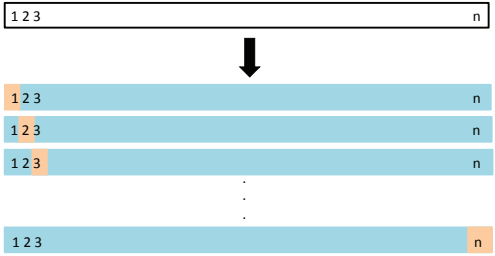


The validation estimate of the test error can be highly variable, depending on precisely which observations are included in the training set and which observations are included in the validation set $\Rightarrow$ high variance in model selection.

Table of Contents

1. Training, Validation and Test Set
2. Cross-Validation
3. Tuning Models
4. Data Cleaning
5. Feature Engineering
6. Transformations of the Output
7. A Recipe for Supervised Learning

Leave-One-Out Cross-Validation (LOOCV)



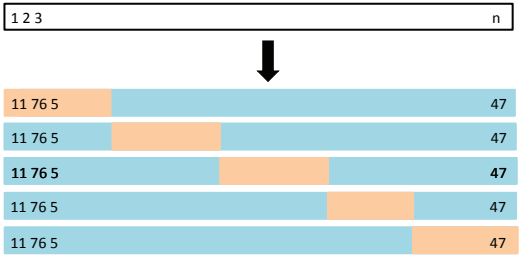
$$\text{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n} \text{MSE}_i \qquad \text{MSE}_i = (y_i - \hat{y}_i)^2$$

where $\hat{y}_i$ is the prediction obtained by fitting without (x_i, y_i).

Disadvantage: computational cost of n fits

(except for linear regression, see section 5.1.2 of textbook).

K-Fold Cross-Validation

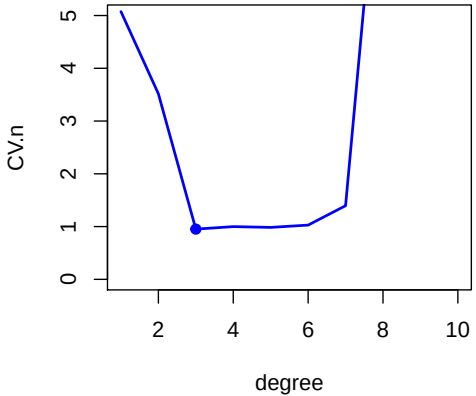


$$\mathrm{CV}_{(K)} = \sum_{k=1}^{K} \frac{n_k}{n} \mathrm{MSE}_k \qquad \mathrm{MSE}_k = \frac{1}{n_k} \sum_{i \in C_k} (y_i - \hat{y}_i)^2$$

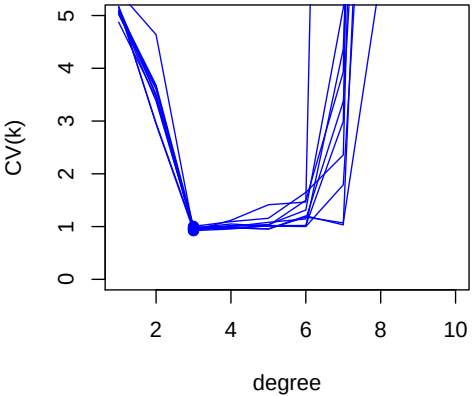
where $\hat{y}_i$ are predictions obtained by fitting without the data in part C_k.

Cross-Validation Applied to the Artificial Data

Leave-one-out Cross Validation



5-Fold Cross Validation



How to choose K

- The choice of the number of folds K is somewhat arbitrary.
  Typical choices are $K = 5$ or $K = 10$.
- LOOCV has higher computational costs, since n fits are made instead of K
  (Except for least squares linear or polynomial regression.)

Summary: How to Split the Available Data

Assumption: Hyperparameters fixed
Goal: Find best parameters

Use all data to estimate the parameters.

Assumption: Hyperparameters fixed
Goal: Estimate test error

Split data into training and test set(s).
Estimate parameters on the training set(s).
Estimate test error on the test set(s).

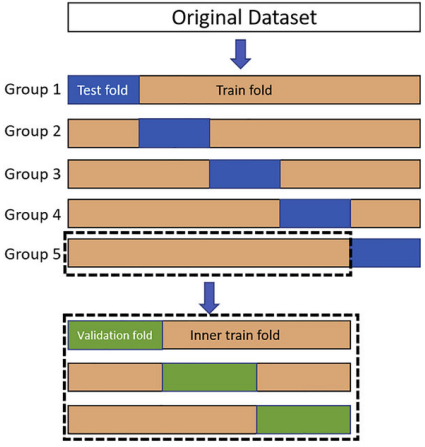
Assumption: Hyperparameters unknown
Goal: Find best hyperparameters

Split data into training and validation set(s).
Estimate parameters on the training set(s).
Estimate test errors for all hyperparameter choices on the validation set(s).
Select hyperparameters with lowest test error.

Assumption: Hyperparameters unknown
Goal: Estimate test error

Split data into training, validation and test set(s). Estimate parameters on the training set(s). Select hyperparameters with lowest test error estimated with the validations set(s). Estimate test error on the test set(s).

Nested Cross-Validation



- Outer loop: for each group keep test set for final evaluation.
- Inner loop: find optimal hyper-parameters with standard cross-validation. Note: for each group a different hyper-parameter setting may be optimal.
- Estimate the test error, by computing the average error on the test sets of the outer loop.

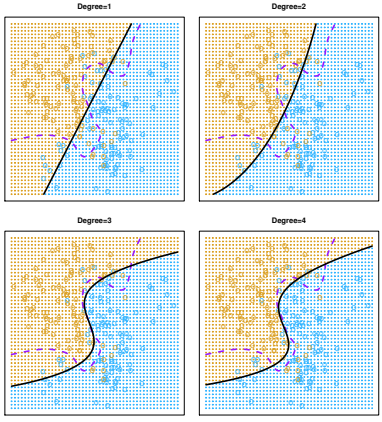
figure from `https://doi.org/10.1016/j.patter.2021.100329`

Cross-Validation on Classification Problems

Instead of the log-likelihood, one can also use e.g. the average misclassification rate on the held-out sample for cross-validation in classification problems.

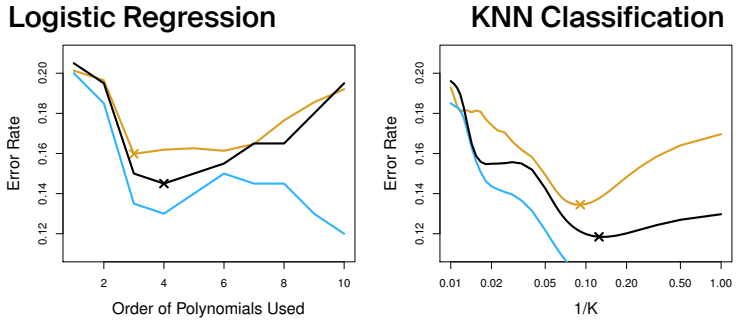
$$\text{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$$

Optimal decision boundary (purple)
Estimated decision boundary (black)
for polynomial degrees 1-4.



misclassification rates:
$\text{degree} = 1 : 0.201, \text{degree} = 2 : 0.197$
$\text{degree} = 3 : 0.160, \text{degree} = 4 : 0.162$

Cross-Validation on Classification Problems



training error true test error 10-fold CV

Note: the true test error is known, because the data generator is known in this case.

Side remark: How can it be that the training error goes up with increasing flexibility of the method (order of the polynomial used)? Logistic regression maximizes the likelihood of the parameters β_l given the data $\Rightarrow$ training likelihood is monotonically increasing with order of polynomials, but the training error (misclassification rate) is not necessarily decreasing, because the misclassification rate is a different measure than the log-likelihood.

Quiz

Which of the following statements are correct?

- After finding in a model comparison the best performing model on the validation set, we compute the error on the validation set and the error on the test set.
  1. The test error is usually larger than the validation error.
  2. Test error and validation error are roughly equal.
  3. The test error is usually smaller than the validation error.
- The error on unseen data tends to be lower for a model trained on all available data compared to a model trained on a training set with 80% of all available data.
- In a model comparison (e.g. to select hyper-parameters) it is acceptable to fit each model on all available data and compare them on a validation set consisting of 50% of the data.
- Estimates of the test error with the validation set approach have lower variance than those with LOOCV.
- In a binary classification task we could use the AUC instead of the error rate to perform cross-validation.

Table of Contents

1. Training, Validation and Test Set
2. Cross-Validation
3. Tuning Models
4. Data Cleaning
5. Feature Engineering
6. Transformations of the Output
7. A Recipe for Supervised Learning

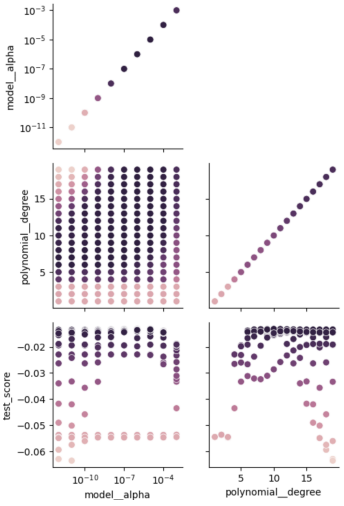
Tuning Models

Hyper-parameter tuning: finding the best model for the given data.

Common recipes:

- **Grid Search**: Perform cross-validation on a grid of hyper-parameter values. E.g. pick 20 different degrees in polynomial regression and 10 different regularization strengths, compute an estimate of the test loss on these 200 points with cross-validation, and pick the best degree and regularization strength. (`GridSearchCV`, `RandomizedSearchCV`)
- Trick: start search with a small subset of the data (because this is faster), repeat the search with a larger subset on the 50% best points, etc. (`HalvingGridSearchCV`, `HalvingRandomSearchCV`)
- Use more sophisticated sampling methods for the hyper-parameters to be evaluated, see e.g. `https://www.automl.org/`

Tuning Models: Example



- Regularization strength usually on a log-scale
- Initially, choose boundaries of the grid arbitrarily; if the best values are at the boundary, extend them.
- Maybe refine the grid in the region where the best values are.

Table of Contents

1. Training, Validation and Test Set
2. Cross-Validation
3. Tuning Models
4. Data Cleaning
5. Feature Engineering
6. Transformations of the Output
7. A Recipe for Supervised Learning

Dealing with Missing Data

We can either

- drop all data points that contain missing data.
  Disadvantage: fewer data points.
- impute missing data with e.g. the mean or the median of that predictor.
  Disadvantage: “wrong” data points.
- impute missing data with unsupervised learning tools, like matrix completion (see later in the course).

Removing Predictors

- Constant predictors should be removed.
- If multiple predictors are perfectly correlated, keep only one of them.
- One can also try to remove almost constant or almost perfectly correlated predictors, but – WATCH OUT – this may introduce errors.
- If the response Y is known to be independent of a predictor X, i.e. $P(Y, X) = P(Y)P(X)$, it should be removed. In praxis, it is usually not known if the response is really independent of a given predictor.

Standardization

Standardization is a transformation that shifts the data such that its mean is 0 and scales it such that its standard deviation is 1.

Formally: for data $x_1, \ldots, x_n$ with mean $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ and standard deviation $\sigma = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2}$ the standardized data is given by

$$\tilde{x}_i = \frac{x_i - \bar{x}}{\sigma}$$

Table of Contents

1. Training, Validation and Test Set
2. Cross-Validation
3. Tuning Models
4. Data Cleaning
5. Feature Engineering
6. Transformations of the Output
7. A Recipe for Supervised Learning

Feature Representation

Idea: Instead of fitting linear regression on p predictors,
fit linear regression on q features of the original predicators.

$$\hat{Y} = \theta_0 + \theta_1 H_1 + \theta_2 H_2 + \cdots + \theta_q H_q$$

with $H_i = f_i(X)$.

Previous Examples: Polynomial Regression & Bag-of-Words

Make a method more flexible by adding features.

With one-dimensional input X ($p = 1$), Polynomial Regression can be written as

$$\hat{Y} = \theta_0 + \theta_1 H_1 + \theta_2 H_2 + \cdots + \theta_q H_q$$

$$\text{where } H_i = f_i(X) = X^i$$

Bag-of-Words for the spam dataset can be seen as another example of feature engineering, where H_i = normalized count of word i in email X.

Categorical Predictors: Dummy Variables/One-Hot-Coding

Chicken weight as a function of time and diet.

Diet	Time	Weight
diet1	3	134
diet1	6	145
diet2	3	124

$H_i = 1$ if diet $X_1 =$ diet$i$, otherwise $H_i = 0$.

For example, as $x_{31} =$ diet2

$$(h_{31}, h_{32}, h_{33}, h_{34}) = (0, 1, 0, 0)$$

Diet1	Diet2	Diet3	Diet4	Time	Weight
1	0	0	0	3	134
1	0	0	0	6	145
0	1	0	0	3	124

Why not an integer code $X_1 \in \{1, 2, 3, 4\}$ or a binary code $X_1 \in \{(0, 0), (1, 0), (0, 1), (1, 1)\}$ for diet?

- The pairwise Euclidean distances are not the same (diet 1 is closer to diet 2 than to diet 4), which may be nonsensical for the given data.
- $\hat{Y} = f(X)$ may look more complicated (non-linear) for the integer or binary code than for the one-hot code.

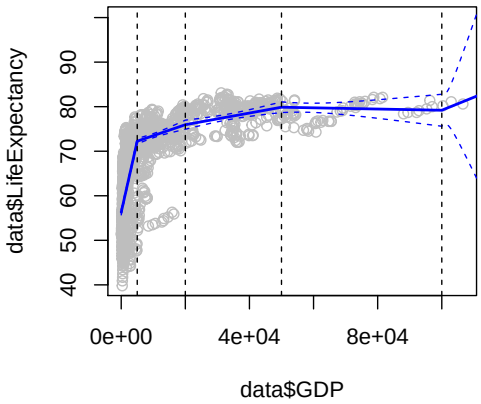
Categorical Predictors: Dummy Variables/One-Hot-Coding

When fitting a linear model with intercept, one level (an arbitrarily selected “standard” level) should be dropped for each predictor; the coefficients are interpreted as change relative to the standard level.

E.g. gender (female or male), treatment (1, 2 or 3)

Intercept	Male	Treat2	Treat3
1	1	0	0
1	1	0	1
1	1	0	0
1	0	1	0
1	0	0	0

Splines



A **degree-d spline** is a piecewise degree-d polynomial, with continuity in derivatives up to degree $d-1$.

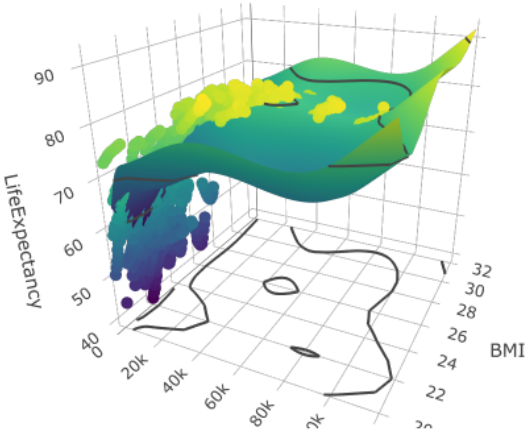
$$H_1 = X, H_2 = X^2, \ldots, H_d = X^d$$
$$H_{1+d} = h(X, c_1), \ldots, H_{K+d} = h(X, c_K)$$

with knots $c_1, \ldots, c_K$ and truncated power basis function:

$$h(x, c) = \begin{cases} (x-c)^d & x > c \\ 0 & \text{otherwise} \end{cases}$$

There are also other possibilities for the basis of a degree-d spline. E.g. the B-spline basis (not discussed here) has better numerical properties.

Generalized Additive Model (GAM)



$$\hat{Y} = s_1(X_1) + s_2(X_2) + \ldots + s_p(X_p)$$

with splines $s_i(X_i) = \sum_j \beta_{ij} H_{ij}$.

Respecting Neighbourhood Relationships

Suppose some predictor X_1 is an angle between 0° and 360°.

If the values are taken as such, 2° looks more different from 359° than from 90° in the sense that |2 - 359| > |2 - 90|.

Alternative: $H_1 = \sin(X_1), H_2 = \cos(X_1)$

In this representation 2° is much closer to 359° than to 90° in the sense that $\|(\sin(2), \cos(2)) - (\sin(359), \cos(359))\| < \|(\sin(2), \cos(2)) - (\sin(90), \cos(90))\|$.

Quiz

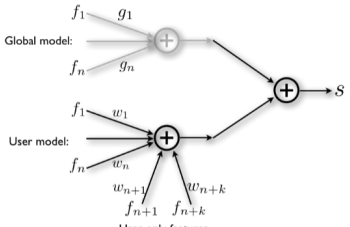
1. For every classification problem with (non-linear) decision boundary and zero irreducible noise there exists a feature representation such that logistic regression on the features solves the classification problem without errors.
2. To fit a degree-d spline with K knots we use a feature representation and linear regression with $d + K + 1$ parameters.
3. We want to predict the number of rented bicycles based on weather condition (sunny, cloudy, foggy, rainy), wind speed, week day (Monday, Tuesday, etc.). After one-hot coding relative to a standard level there are

   A 3 **B** 10 **C** 12 **D** 13 predictors

Gmail Priority Inbox in 2010

The Learning Behind Gmail Priority Inbox

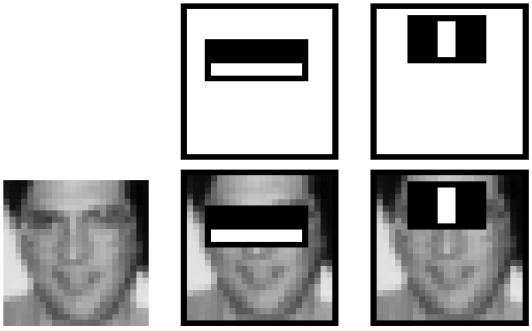
Douglas Aberdeen **Ondrej Pacovsky** **Andrew Slater**
Google Inc.
Zurich, Switzerland



2.1 Features

There are many hundred features falling into a few categories. *Social features* are based on the degree of interaction between sender and recipient, e.g. the percentage of a sender's mail that is read by the recipient. *Content features* attempt to identify headers and recent terms that are highly correlated with the recipient acting (or not) on the mail, e.g. the presence of a recent term in the subject. Recent user terms are discovered as a pre-processing step prior to learning. *Thread features* note the user's interaction with the thread so far, e.g. if a user began a thread. *Label features* examine the labels that the user applies to mail using filters. We calculate feature values during ranking and we temporarily store those values for later learning. Continuous features are automatically partitioned into binary features using a simple ID3 style algorithm on the histogram of the feature values.

Face Detection with Rectangle Features



The two most important features for face detection are shown. The first one is a 2-rectangles feature, the second one a 3-rectangles feature. The sum of the pixels which lie within the white rectangles are subtracted from the sum of pixels in the black rectangles.

Rapid object detection using a boosted cascade of simple features

http://dx.doi.org/10.1109/CVPR.2001.990517

Table of Contents

1. Training, Validation and Test Set
2. Cross-Validation
3. Tuning Models
4. Data Cleaning
5. Feature Engineering
6. Transformations of the Output
7. A Recipe for Supervised Learning

Transformations of the Output: Changing the Noise Model

Applying linear regression to g-transformed outputs is equivalent to assuming a "g-normal" distribution for the conditional data generator $Y|X$, i.e.

$$p(Y = y|X = x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(g(y)-f(x))^2}{2\sigma^2}} \tag{1}$$

For example: $g(y) = log(y) = Y$ is log-normally distributed.
Instead of thinking about suitable transformations of the output,
it may be preferable to think about which distribution is most reasonable
for the conditional data generator $Y|X$!

Table of Contents

1. Training, Validation and Test Set
2. Cross-Validation
3. Tuning Models
4. Data Cleaning
5. Feature Engineering
6. Transformations of the Output
7. A Recipe for Supervised Learning

A Recipe for Supervised Learning

1. Collect (a lot of) data.
2. Look at the raw data; clean it if necessary.
3. Select relevant features from the raw data,
   i.e. choose a suitable representation of the raw data.
4. Select a machine learning method.
5. Fit the data and tune hyperparameters, e.g. with cross-validation.
6.
   - training loss: high, test loss: high (underfitting?): select a more flexible method.
   - training loss: low, test loss: high (overfitting?): select a less flexible method.
7. Repeat 4-6 until the lowest test loss is found.
8. If unhappy with the lowest test loss, repeat 2-7 or collect more data.
9. Fit the best model on all available data for best performance on unseen data.

Suggested Reading

- 5.1. Cross-Validation
- 3.3.1 Qualitative Predictors
- 7.4 Regression Splines
- 7.7 Generalized Additive Models