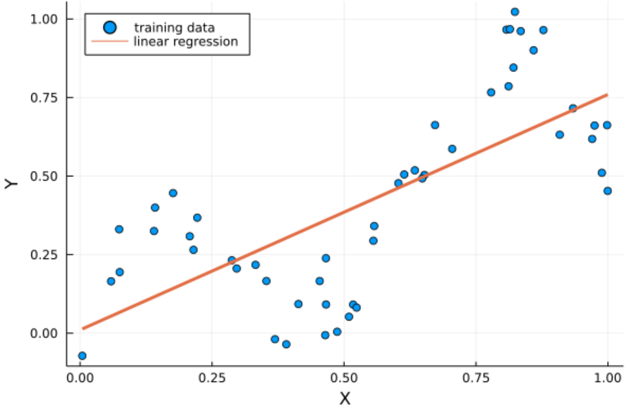
The Flexibility of Models and the Bias-Variance Trade-Off

Johanni Brea

Introduction to Machine Learning

When Linear Methods are Not Flexible Enough



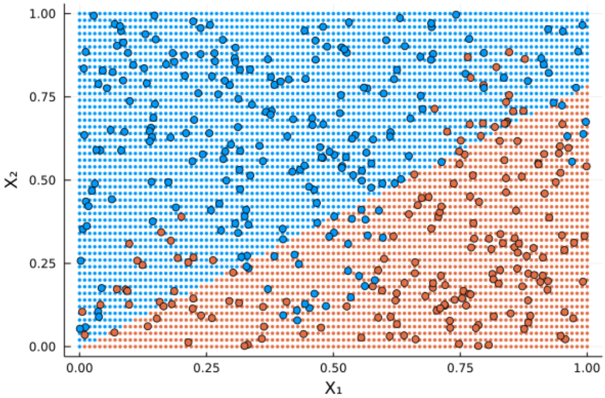
Data Generating Process

$$Y = 0.3\sin(10X) + 0.7X + \epsilon$$

There is structure in the data that linear regression fails to capture.

Linear regression has a high **reducible error**.

When Linear Methods are Not Flexible Enough



large dots: training data
small dots: classification with logistic regression at decision threshold 0.5

Data Generating Process
Y = red if
$$\sigma(20(0.3\sin(10X_1) + 0.7X_1 - X_2)) > \epsilon$$

There is structure in the data that logistic regression fails to capture.

Logistic regression has a high **reducible error**.

Table of Contents

1. **Error Decomposition for Regression**
2. Polynomial Regression
3. K Nearest-Neighbors
4. Underfitting and Overfitting
5. Bias-Variance Decomposition

Error Decomposition for Regression

Assume the true conditional data generating process has the form

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

This is also sometimes written as $Y = f(X) + \epsilon$.

We call f the **systematic** information that X provides about Y and ϵ the **noise**.

Let us assume with some supervised machine learning method we find

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

Let us define the **predicted average response** $\hat{Y} = \hat{f}(X)$.

We would like to know how much the predicted average response $\hat{Y}$ differs in expectation from test data.

Blackboard: Error Decomposition for Regression

$$E_{Y|X}(Y - \hat{Y})^2 = \underbrace{\left(f(X) - \hat{f}(X)\right)^2}_{\text{Reducible}} + \underbrace{\mathrm{Var}(\epsilon)}_{\text{Irreducible}}$$

Quiz

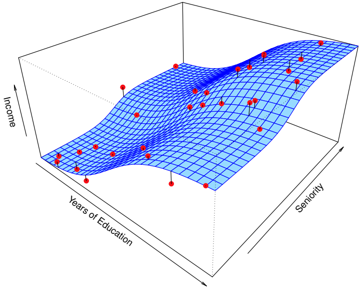
- Let us assume the red data points in this figure were generated with the help of a function whose graph is shown in blue. What is correct?

  A. A linear fit has a non-zero reducible error.
  B. The irreducible error is zero.
  C. A method with zero prediction error on the red data has a reducible error of zero.

- Assume we perform linear regression on the red data points and compute the residuals $\hat{\epsilon}_i = y_i - \hat{y}_i$ and the empirical variance $\frac{1}{n-1}\sum_{i=1}^{n}\hat{\epsilon}^2$. The irreducible error is

  A. larger B. smaller

  than this empirical variance.



- What is the irreducible error for the following data generating process?

$$p(y|x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(y-2x)^2}{2}}$$

A. 1 B. 2 C. 4

Summary

- The true systematic information f that X provides about Y is usually unknown.
- Our goal: find the function $\hat{f}$ that minimizes the reducible error.
- The test error of $\hat{f}$ is never lower than the irreducible error.

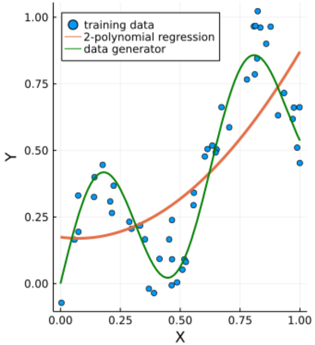
Table of Contents

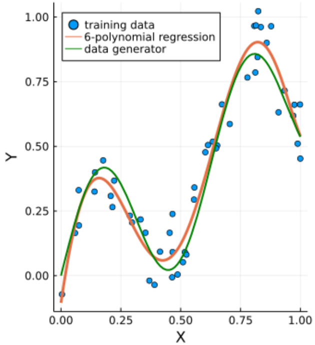
1. Error Decomposition for Regression
2. **Polynomial Regression**
3. K Nearest-Neighbors
4. Underfitting and Overfitting
5. Bias-Variance Decomposition

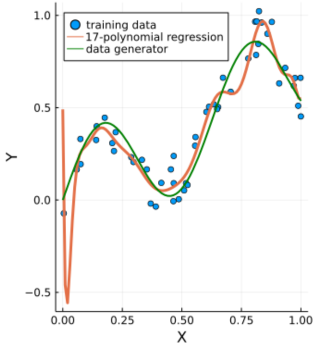
Polynomial Regression

Finding the parameters of a polynomial $f(X) = \theta_0 + \theta_1 X + \theta_2 X^2 + \cdots \theta_d X^d$
is equivalent to finding the parameters in linear regression with polynomial features

$$X_1 = X, X_2 = X^2, \ldots, X_d = X^d$$







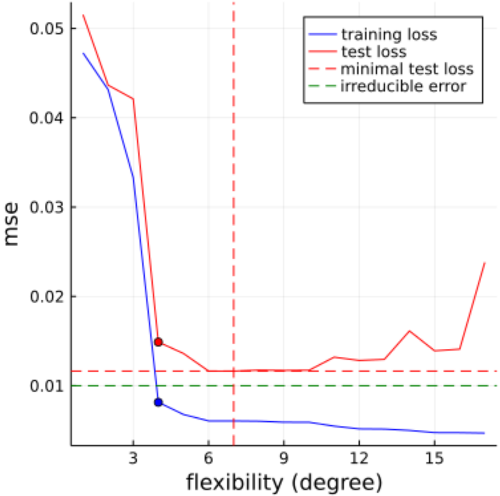
Polynomial Regression with MLJ

```
# In two steps
H = MLJ.transform(Polynomial(degree = 3), X)  # transform to polynomial features
model = LinearRegressor()                     # we will do linear regression
mach = machine(model, H, y)                   # fit polynomial features H

# In one step with a pipeline
model = Polynomial(degree = 3) |> LinearRegressor()
# this model is the same as
model = Pipeline(Polynomial(degree = 3), LinearRegressor())
mach = machine(model, X, y)

# For classification:
model = Polynomial(degree = 3) |> LogisticClassifier()
```

Polynomial Regression

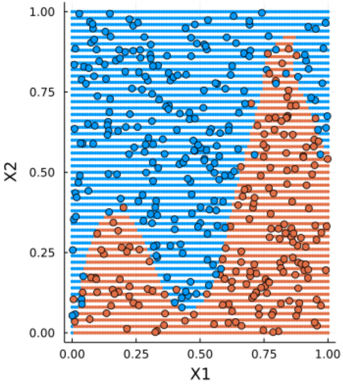


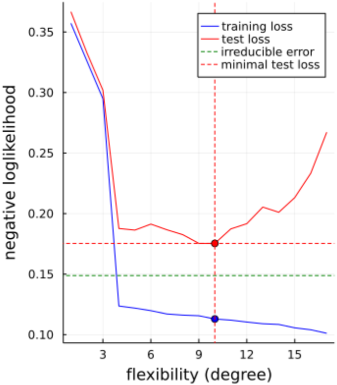
- Polynomial regression with degree 1 is linear regression.
- For small degrees (<6) training loss and test loss are high.
- Minimal test loss is reached for intermediate degrees; it is only slightly higher than the irreducible error.
- For high degrees (>10) the method is too flexible; it can fit peculiarities of the given training set. Therefore the training loss is lowest there, but the test loss is higher.

Multiple Polynomial Classification

$$f(X) = \theta_0 + \theta_1 X_1 + \theta_2 X_2 + \theta_3 X_1 X_2 + \theta_4 X_1^2 + \theta_5 X_2^2 + \cdots + \theta_{(d+1)(d+2)/2} X_2^d$$

(all powers up to *d*'th order)





Summary

- Polynomial regression is like multiple regression with the additional features being higher orders of the original feature(s).
- For p predictors and d degrees there are $\binom{p+d}{p}$ parameters [1]. This number grows quickly in p and d, making polynomial regression unpractical for high dimensional input data. For example a fourth order polynomial in 20 dimensions has $\binom{24}{20} = 10625$ parameters.
- The degree is a **hyper-parameter** that controls the flexibility of the method.
- The training loss decreases with increasing flexibility (degree).
- The test loss is U-shaped as a function of the flexibility.
- The lowest test loss is not lower than the irreducible error.

[1] Distribute d balls into $p+1$ urns: all terms of the form $1^{d_0} x_1^{d_1} \cdots x_p^{d_p}$ s.t. $\sum_{i=0}^{p} d_i = d$

Table of Contents

1. Error Decomposition for Regression

2. Polynomial Regression

3. **K Nearest-Neighbors**

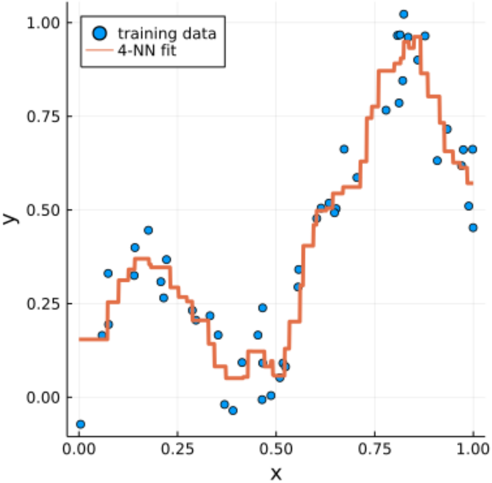
4. Underfitting and Overfitting

5. Bias-Variance Decomposition

kNN Regression

$$\hat{y} = \frac{1}{K} \sum_{i \in \mathcal{N}_0} y_i$$

where $\mathcal{N}_0$ is the set of k nearest neighbours of x_0.

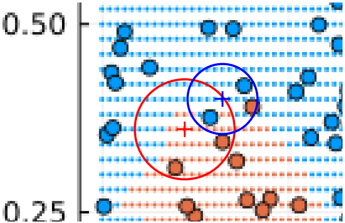


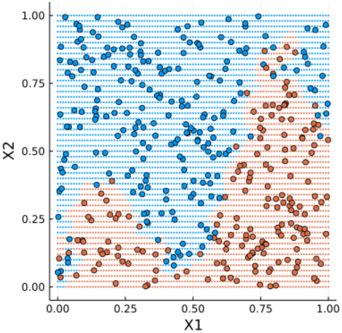
kNN Classification

$$\widehat{\Pr}(Y = j | X = x_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} I(y_i = j)$$

where $\mathcal{N}_0$ is the set of k nearest neighbours of x_0.

$k = 3$



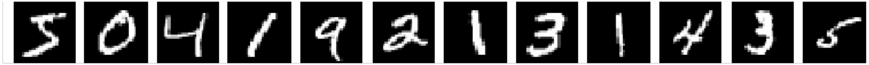


Properties of kNN

- KNN is a non-linear method.
- kNN is an example of a **non-parametric method**, where the raw training data is used to make predictions.

  This is in contrast to **parametric methods** (like linear regression), where all information about the training data is stored in the parameters θ.
- The number of neighbors k is a **hyper-parameter**; it controls the assumed smoothness of the function family and does not depend on the training data.
- If the hyper-parameter k is small, the function family is **flexible**, which allows to follow the training data accurately. If k is large the function family is **inflexible** and follows only the main trend of the data.
- kNN is very fast to fit (no parameters are learnt).
- kNN is slow to make predictions, because the neighbors of the test point need to be searched in the training set.

kNN Classification on MNIST



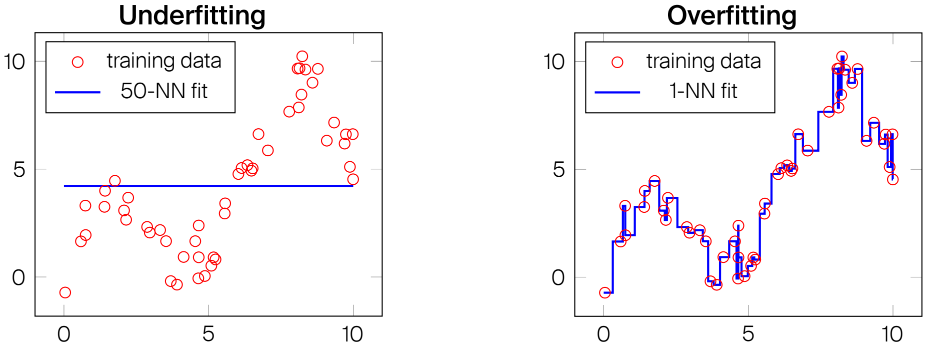
First nearest neighbors classification on MNIST reaches a misclassification rate of approximately 3%.

This is better than linear classification.

Table of Contents

1. Error Decomposition for Regression
2. Polynomial Regression
3. K Nearest-Neighbors
4. **Underfitting and Overfitting**
5. Bias-Variance Decomposition

Underfitting and Overfitting



the function family is not flexible enough.
training and test error are high.

the function family is too flexible.
training error is low, test error is high.

Hyper-parameters control the flexibility (k for kNN, degree for polynomial regression).

Table of Contents

1. Error Decomposition for Regression

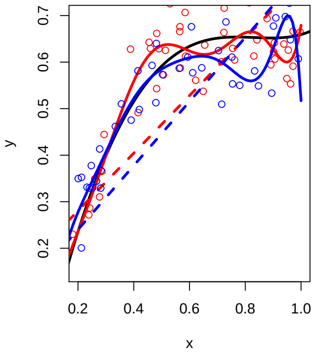
2. Polynomial Regression

3. K Nearest-Neighbors

4. Underfitting and Overfitting

5. **Bias-Variance Decomposition**

Fitting Multiple Training Sets



- Red and blue data points generated with
  $Y = f(X) + \epsilon$ with $\text{Var}(\epsilon) = 0.06^2$
  and $f(X) = \sin(2X) + 2(X - 0.5)^3 - 0.5X$ (black line)
- Red/blue lines: fits on red/blue training set
- The linear fits (dashed lines) are close to each other (small variance) but far away from f (large bias).
- The polynomial fits (with $d = 10$) are far from each other (large variance) but close to f (small bias).

The Bias-Variance Trade-Off

Expected test MSE

$$E_{Y|X,\hat{f}}\left(Y-\hat{f}(X)\right)^2=[\mathrm{Bias}(\hat{f}(X))]^2+\mathrm{Var}(\hat{f}(X))+\mathrm{Var}(\epsilon)$$

- Here, $\mathrm{Var}(\hat{f}(X))$ refers to the variance of the estimator, if it would be fitted multiple times to each time another training set.
- Generally: flexible methods have higher variance but lower bias than less flexible methods.

Blackboard: Bias-Variance Decomposition

$$E_{Y|X,\hat f}\left(Y-\hat f(X)\right)^2 = E_{\varepsilon,\hat f}\left(f(X)+\varepsilon-\hat f(X)\right)^2$$

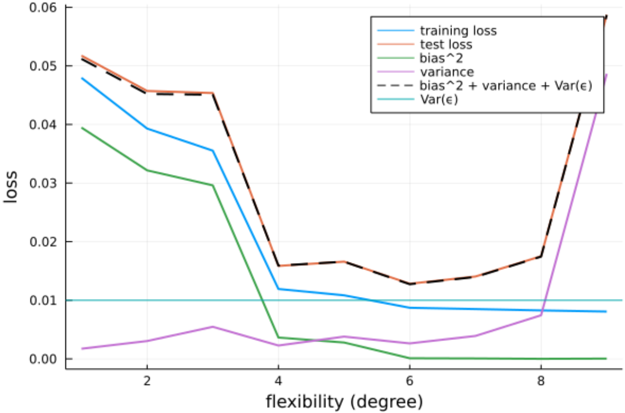
$$= \underbrace{E_{\hat f}\left(f(X)-\hat f(X)\right)^2}_{\text{expected reducible error}} + \underbrace{Var(\varepsilon)}_{\text{irreducible error}}$$

$$E_{\hat f}\left(f(X)-\hat f(X)\right)^2 = E_{\hat f}\left(f(X)-E_{\hat f}(\hat f(X))+E_{\hat f}(\hat f(X))-\hat f(X)\right)^2$$

$$= \underbrace{\left(f(X)-E_{\hat f}(\hat f(X))\right)^2}_{\text{squared bias}} + \underbrace{Var\,\hat f(X)}_{\text{variance}}$$

$$2\left(f(x)-E_{\hat f}(\hat f(x))\right)\left(E_{\hat f}(\hat f(x))-E_{\hat f}(\hat f(x))\right)=0$$

The Bias-Variance Trade-Off



For 1000 different training sets of size $n = 50$, polynomial fits of degrees 1 to 9 where performed. Each fit was evaluated on a large test set.

The plot shows the average training and test loss (over all 1000 training sets), and the average squared bias and variance (over all test points).

Note that the squared bias, the variance and the irreducible error sum up to the test loss.

Quiz

Which of the following statements is correct?

1. Flexible machine learning methods tend to have higher variance than rigid machine learning methods.
2. For large values of the hyper-parameter k, kNN has a higher risk of overfitting the training data than for small k.
3. If we have access to different training sets, we can estimate the variance of a machine learning method without having access to the variance of the irreducible error or the true data generating function f.
4. If we have access to different training sets, we can estimate the squared bias of a machine learning method without having access to the variance of the irreducible error or the true data generating function f.

Summary

- Rigid (inflexible) methods
  - may underfit the data.
  - may have a large bias.
  - tend to have little variance when fitted on different training sets.
- Flexible methods
  - may overfit the data.
  - can achieve a low bias.
  - tend to have high variance when fitted on different training sets.

In expectation over training sets and responses, the test MSE of a method is given by the sum of the method’s variance, its squared bias and the irreducible error.

Side remark: Thanks to their simplicity, polynomial regression and KNN are great for educational purposes, but they are rarely used in modern applications of machine learning. For successful choices of more advanced machine learning methods it is, however, crucial to understand the flexibility of a method, the bias-variance decomposition and overfitting and underfitting.

Suggested Reading

- 2.1 What is Statistical Learning
- 2.2 Assessing Model Accuracy
- 7.1 Polynomial Regression
- 3.5 Comparison of Linear Regression with K-Nearest Neighbors