Machine Learning, Neural Networks and Deep Learning

Dr. B. Wilbertz[1]

[1]Trendiction S.A. / Talkwalker

24th September, 2021

Who I am

- Benedikt Wilbertz
- Head of Data Science and Machine Learning at Talkwalker
- Former Post-Doc from Paris 6
- `benedikt.wilbertz@gmail.com`

Organization

Lecture has 2 parts

Part I: Introduction and classic machine learning by B. Wilbertz

- 2021/09/24 - 9h00/12h00
- 2021/10/01 - 9h00/12h00
- 2021/10/08 - 9h00/12h00
- 2021/10/15 - 9h00/12h00

Part II: Deep Learning by P. Gallinari

- Starting 2021/10/22.

Google Meet session for first part will stay at
`https://meet.google.com/pmv-htva-zcg`

Organization / InClass Competition

Slides and other material will be available at

`https://drive.google.com/drive/folders/1OG6vL_y8oxKwQOx6d_6W-ZtLQ81pgyRj`

Kaggle InClass Competition

`https://www.kaggle.com/c/m2-proba-finance-2021`

- The ranking on this competition will contribute to your final grade for this course
- Invitation link for joining the competition:
  `https://www.kaggle.com/t/f1958ec06f7e440e983a69fd75a5ec52`
- **Important:** Choose your *name* or *student number* as team name! (Account name whatever you like).
- In case of problems:
  `https://www.kaggle.com/about/inclass/faqs`

References and Disclaimer

Literature for part I

- *Applied Predictive Modeling*, Kuhn, M. and Johnson, K. (Springer, 2013)
- *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Hastie, T. and Tibshirani, R. and Friedman, J. (Springer, 2013)
- *An Introduction to Statistical Learning: with Applications in R*, James, G. and Witten, D. and Hastie, T. and Tibshirani, R. (Springer, 2013)

Disclaimer: Many of the figures and slides in this lecture are taken from "An Introduction to Statistical Learning: with applications in R" (Springer, 2013)

Three pillars of Machine Learning

Three pillars of Machine Learning

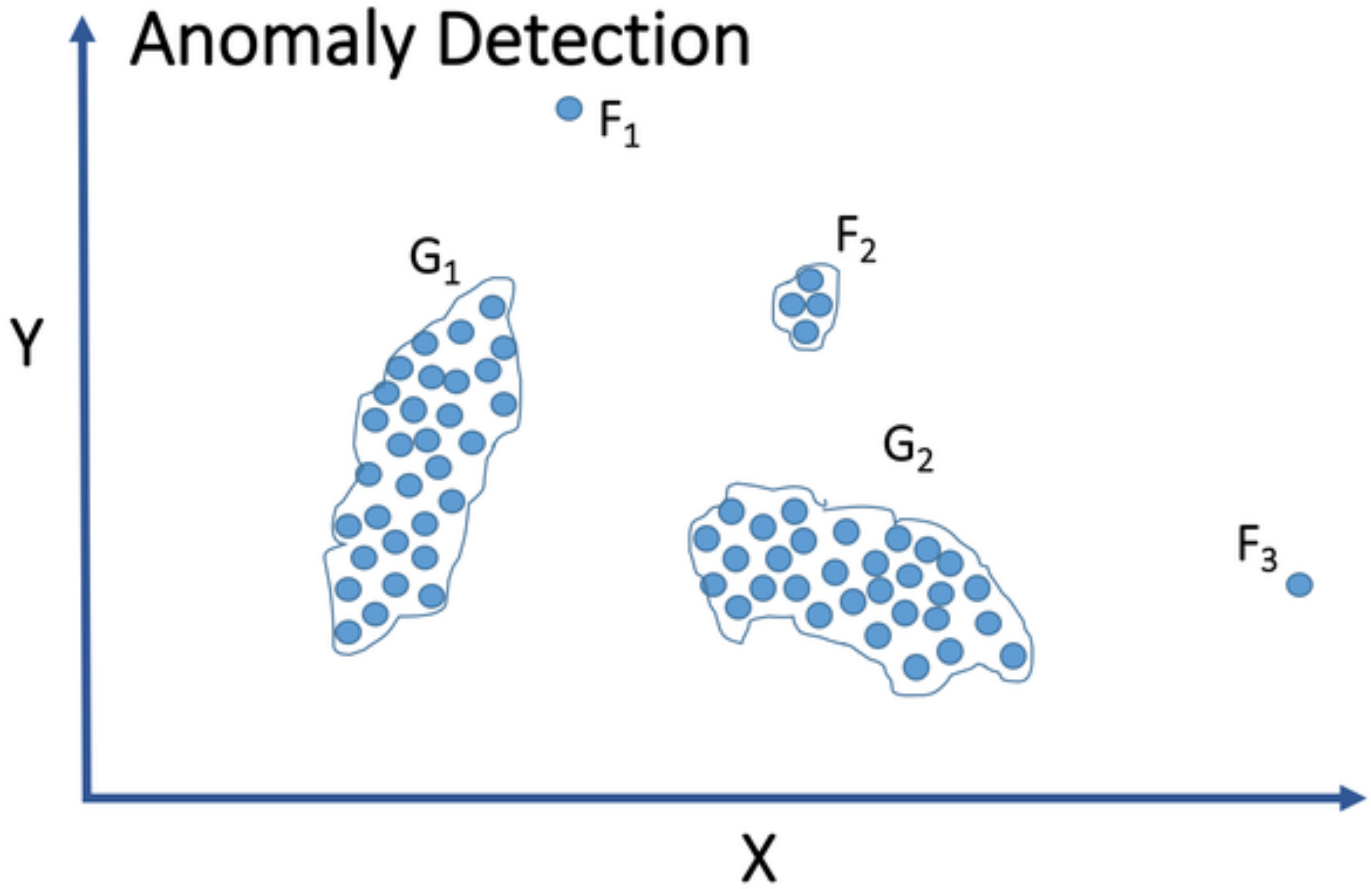
1. Unsupervised Learning
2. Supervised Learning
3. Reinforcement Learning

Unsupervised Learning

Task of inferring a function to describe hidden structure from unlabeled data (e.g. Clustering).

Unsupervised Learning

Task of inferring a function to describe hidden structure from unlabeled data (e.g. Clustering).



Supervised Learning

Task of inferring a function from labeled training data (e.g. Classification, Regression).

Supervised Learning

Task of inferring a function from labeled training data (e.g. Classification, Regression).

Achieving Human Parity on Automatic Chinese to English News Translation

Hany Hassan,* Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William Lewis, Mu Li, Shujie Liu, Tie-Yan Liu, Renqian Luo, Arul Menezes, Tao Qin, Frank Seide, Xu Tan, Fei Tian, Lijun Wu, Shuangzhi Wu, Yingce Xia, Dongdong Zhang, Zhirui Zhang, and Ming Zhou

Microsoft AI & Research

Abstract

Machine translation has made rapid advances in recent years. Millions of people are using it today in online translation systems and mobile applications in order to communicate across language barriers. The question naturally arises whether such systems can approach or achieve parity with human translations. In this paper, we first address the problem of how to define and accurately measure human parity in translation. We then describe Microsoft's machine translation system and measure the quality of its translations on the widely used WMT 2017 news translation task from Chinese to English. We find that our latest neural machine translation system has reached a new state-of-the-art, and that the translation quality is at human parity when compared to professional human translations. We also find that it significantly exceeds the quality of crowd-sourced non-professional translations.

Reinforcement Learning

Learning how software agents should take actions in an environment as to maximize some notion of cumulative reward (e.g. playing games).

The Supervised Learning Problem

Starting point

- Outcome measurement Y (also called dependent variable, response, target).
- Vector of p predictor measurements X (also called inputs, regressors, covariates, features, independent variables).
- In the regression problem, Y is quantitative (e.g price, blood pressure).
- In the classification problem, Y takes values in a finite, unordered set (survived/died, digit 0-9, cancer class of tissue sample).
- We have training data $(x_1, y_1), \ldots, (x_N, y_N)$. These are observations (examples, instances) of these measurements.

Objectives

Objectives

On the basis of the training data we would like to:

- Accurately predict unseen test cases.
- Understand which inputs affect the outcome, and how.
- Assess the quality of our predictions and inferences.

Philosophy

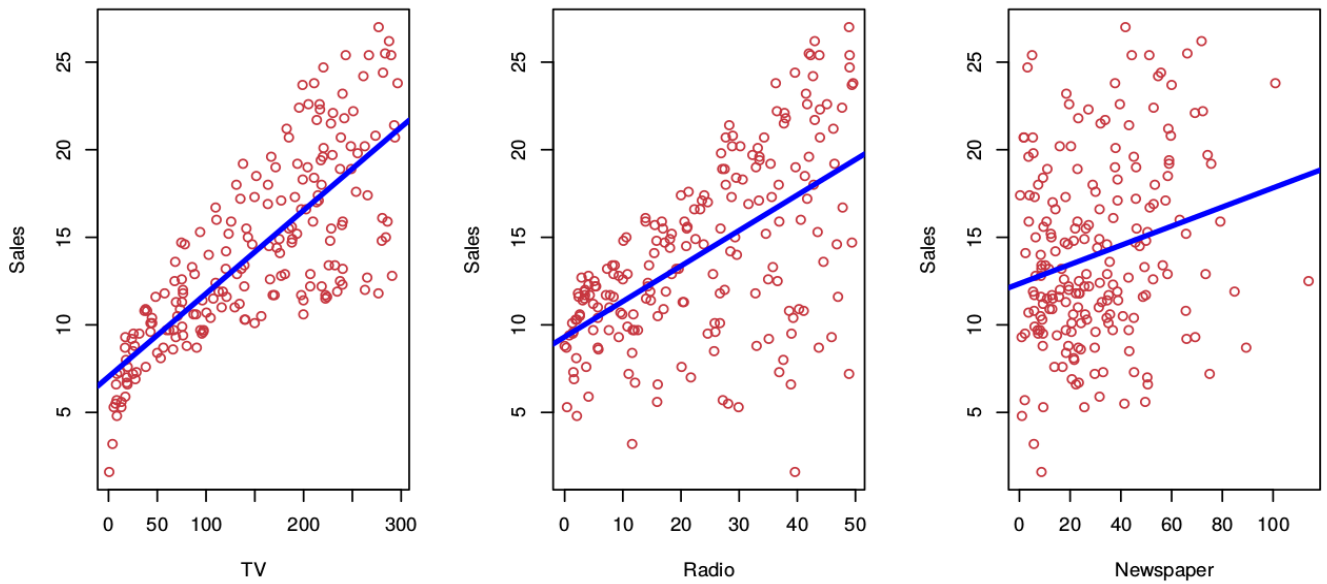
- It is important to understand the ideas behind the various techniques, in order to know how and when to use them.
- One has to understand the simpler methods first, in order to grasp the more sophisticated ones.
- It is important to accurately assess the performance of a method, to know how well or how badly it is working [simpler methods often perform as well as fancier ones!]
- This is an exciting research area, having important applications in science, industry and finance.
- Statistical learning is a fundamental ingredient in the training of a modern *data scientist*.

Unsupervised learning

Unsupervised learning

- No outcome variable, just a set of predictors (features) measured on a set of samples.
- objective is more fuzzy – find groups of samples that behave similarly, find features that behave similarly, find linear combinations of features with the most variation.
- difficult to know how well your are doing.
- different from supervised learning, but can be useful as a pre-processing step for supervised learning.

What is Machine Learning (or Statistical Learning)?



Shown are Sales vs `TV`, `Radio` and `Newspaper`, with a blue linear-regression line fit separately to each. Can we predict `Sales` using these three? Perhaps we can do better using a model

$$\texttt{Sales} \approx f(\texttt{TV}, \texttt{Radio}, \texttt{Newspaper})$$

Notation

Here `Sales` is a *response* or *target* that we wish to predict. We generically refer to the response as Y. `TV` is a *feature*, or *input*, or *predictor*; we name it X_1. Likewise name `Radio` as X_2, and so on. We can refer to the *input vector* collectively as

$$X = \begin{pmatrix} X_1 \\ X_2 \\ X_3 \end{pmatrix}$$

Now we write our model as

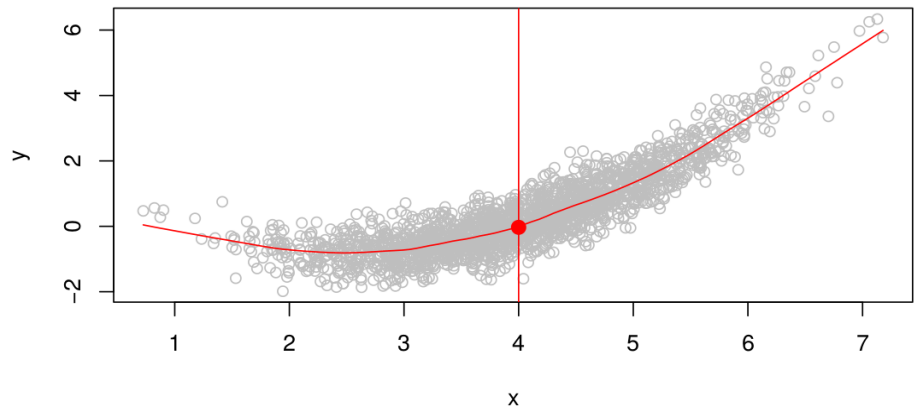
$$Y = f(X) + \varepsilon$$

where ε captures measurement errors and other discrepancies.

What is $f(X)$ good for?

What is $f(X)$ good for?

- With a good f we can make predictions of Y at new points $X = x$.
- We can understand which components of $X = (X_1, X_2, \ldots, X_p)$ are important in explaining Y, and which are irrelevant. e.g. `Seniority` and `Years of Education` have a big impact on `Income`, but `Marital Status` typically does not.
- Depending on the complexity of f, we may be able to understand how each component X_j of X affects Y.



Is there an ideal $f(X)$? In particular, what is a good value for $f(X)$ at any selected value of X, say $X = 4$? There can be many Y values at $X = 4$. A good value is

$$f(4) = \mathbb{E}(Y|X = 4)$$

This ideal $f(x) = \mathbb{E}(Y|X = x)$ is called the *regression function*.

The regression function $f(x)$

The regression function $f(x)$

- Is also defined for vector X; e.g.

$$f(x) = f(x_1, x_2, x_3) = \mathbb{E}(Y|X_1 = x_1, X_2 = x_2, X_3 = x_3)$$

- Is the *ideal* or *optimal* predictor of Y with regard to mean-squared prediction error: $f(x) = \mathbb{E}(Y|X = x)$ is the function that minimizes $\mathbb{E}[(Y - g(X))^2|X = x]$ over all measurable functions g at any point $X = x$.
- $\varepsilon = Y - f(x)$ is the irreducible error – i.e. even if we knew $f(x)$, we would still make errors in prediction, since at each $X = x$ there is typically a distribution of possible Y values.
- For any estimate $\hat{f}(x)$ of $f(x)$, we have

$$\mathbb{E}[(Y - \hat{f}(X))^2|X = x] = [f(x) - \hat{f}(x)]^2 + \mathrm{Var}(\varepsilon)$$

How to estimate f?

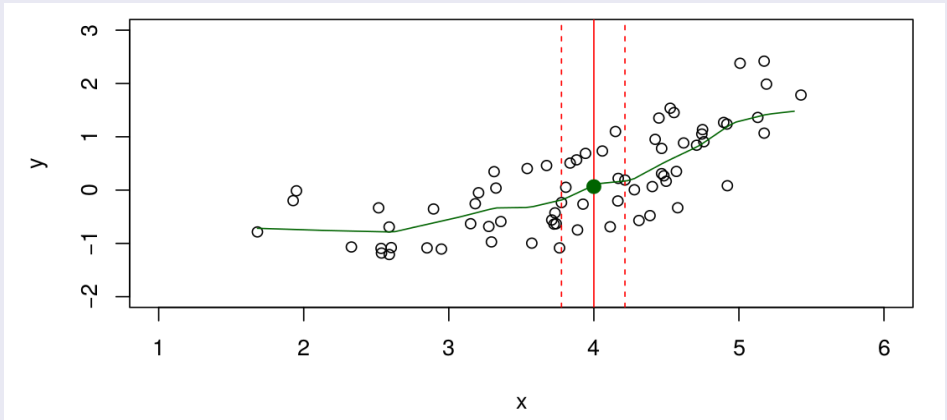
- Typically we have few if any data points with $X = 4$ exactly.
- So we cannot compute $\mathbb{E}(Y|X = x)$!

Nearest Neighbor Averaging

- Relax the definition and let

$$\hat{f}(x) = \text{Ave}(Y|X \in \mathcal{N}(x))$$

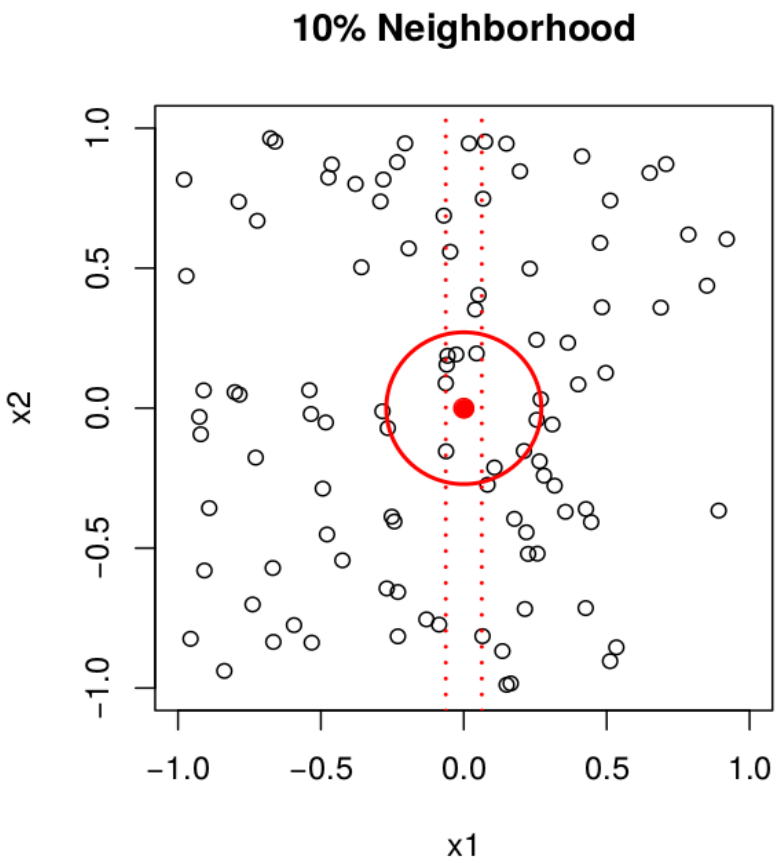
where $\mathcal{N}(x)$ is some *neighborhood* of x.

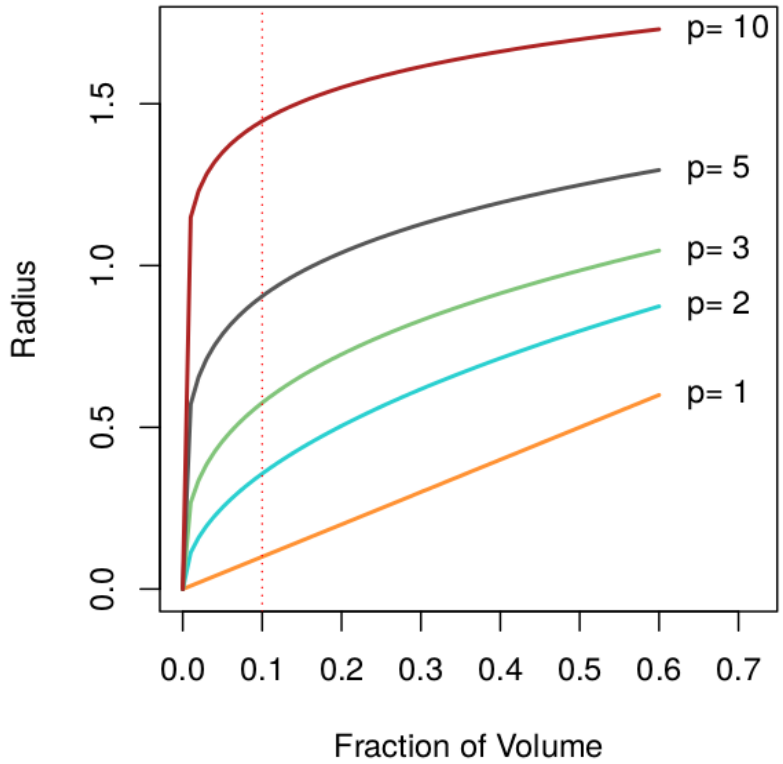


Nearest Neighbor Averaging

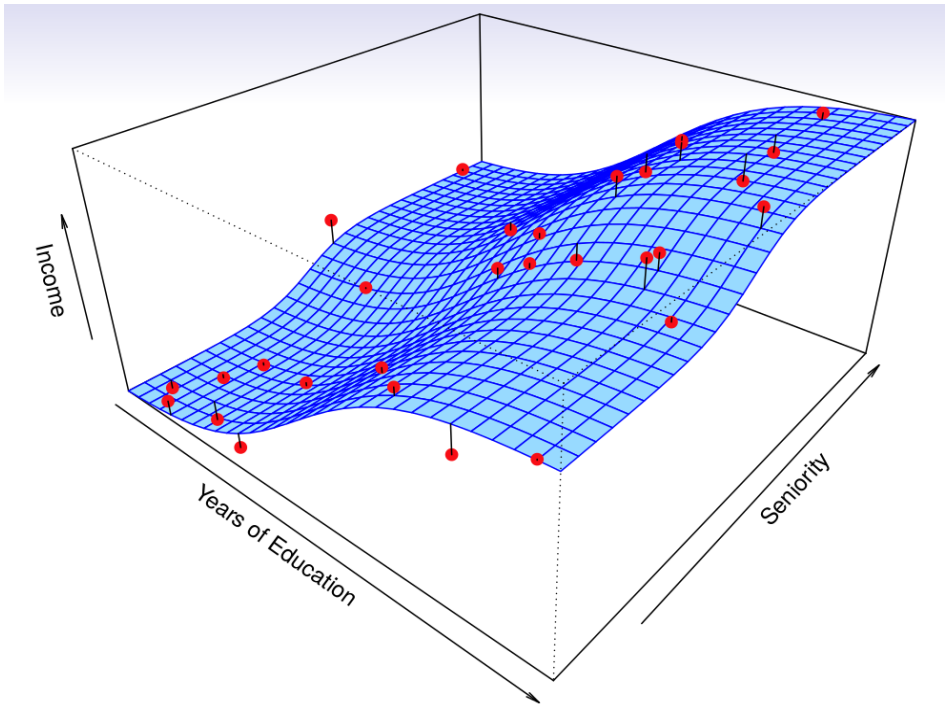
- Nearest neighbor averaging can be pretty good for small p - i.e. $p \leq 4$ and large-ish N.
- Nearest neighbor methods can be lousy when p is large. Reason: the curse of dimensionality. Nearest neighbors tend to be far away in high dimensions.
  - We need to get a reasonable fraction of the N values of y_i to average to bring the variance down - e.g. 10%.
  - A 10% neighborhood in high dimensions need no longer be local, so we lose the spirit of estimating $\mathbb{E}(Y|X = x)$ by local averaging.

The curse of dimensionality





More Structured Models



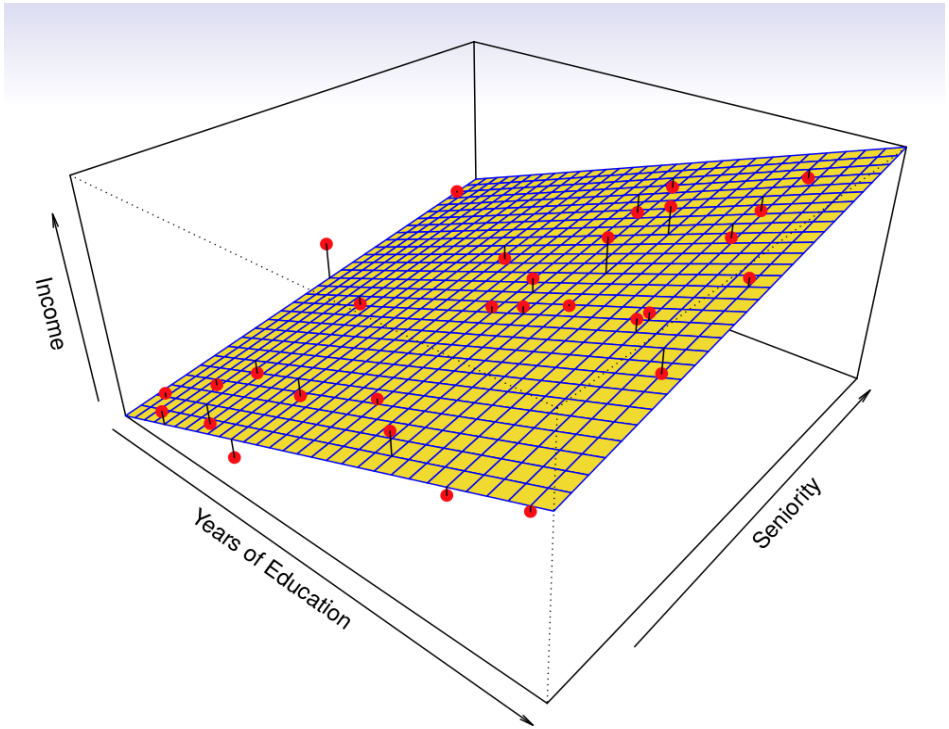
Simulated example. Red points are simulated values for `income` from the model

$$\texttt{income} = \texttt{f}(\texttt{education}, \texttt{seniority}) + \varepsilon$$

f is the blue surface.

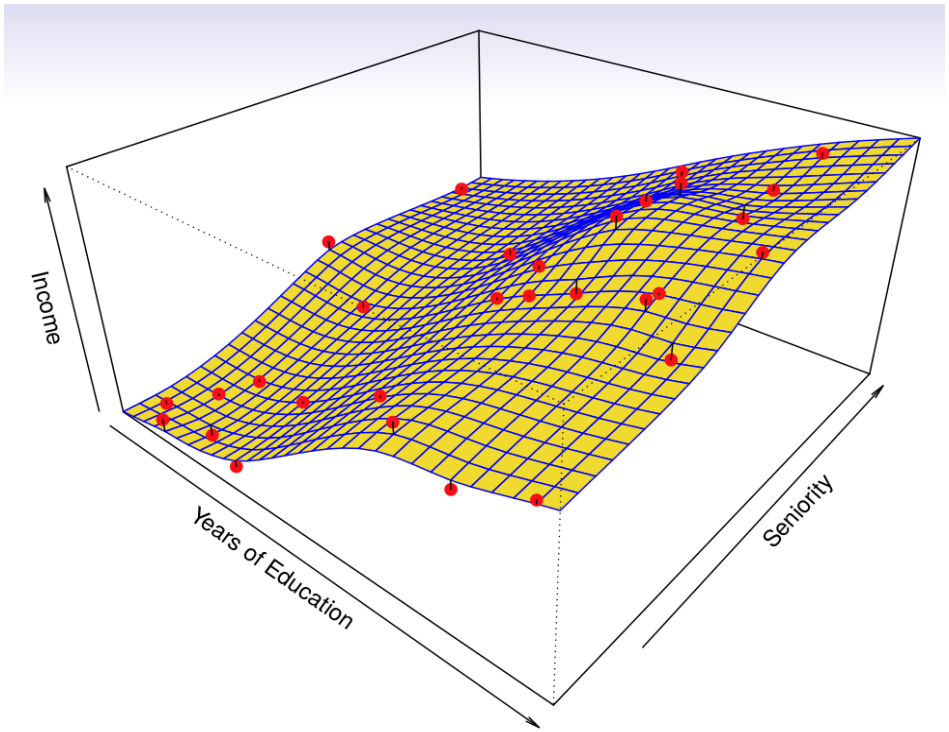
Linear Regression

Linear regression model fit to the simulated data.



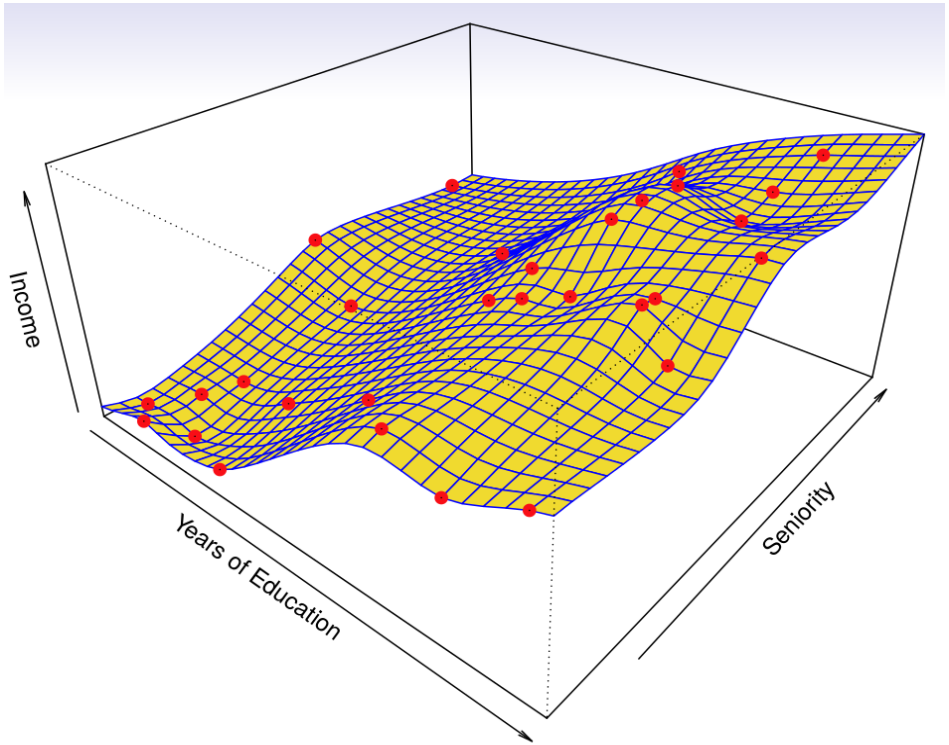
$$\hat{f}_L(\texttt{education}, \texttt{seniority}) = \hat{\beta}_0 + \hat{\beta}_1 \times \texttt{education} + \hat{\beta}_2 \times \texttt{seniority}$$

Spline Regression



More flexible regression model $\hat{f}_S(\texttt{education}, \texttt{seniority})$ fit to the simulated data. Here we use a technique called a *thin-plate spline* to fit a flexible surface. We control the roughness of the fit.

Spline Regression



Even more flexible spline regression model $\hat{f}_S(\texttt{education}, \texttt{seniority})$ fit to the simulated data. Here the fitted model makes no errors on the training data! Also known as *overfitting*.

Some trade-offs

Trade-Offs

- Prediction accuracy versus interpretability. – Linear models are easy to interpret; thin-plate splines are not.
- Good fit versus over-fit or under-fit. – How do we know when the fit is just right?
- Parsimony versus black-box. – We often prefer a simpler model involving fewer variables over a black-box predictor involving them all.

Assessing Model Accuracy

Suppose we fit a model $\hat{f}(x)$ to some training data $\texttt{Tr} = \{x_i, y_i\}_1^N$, and we wish to see how well it performs.

- We could compute the average squared prediction error over $\texttt{Tr}$:

$$\text{MSE}_{\texttt{Tr}} = \text{Ave}_{i \in \texttt{Tr}}[y_i - \hat{f}(x_i)]^2$$

This may be biased toward more overfit models.

- Instead we should, if possible, compute it using fresh test data $\texttt{Te} = \{x_i, y_i\}_1^M$:

$$\text{MSE}_{\texttt{Te}} = \text{Ave}_{i \in \texttt{Te}}[y_i - \hat{f}(x_i)]^2$$

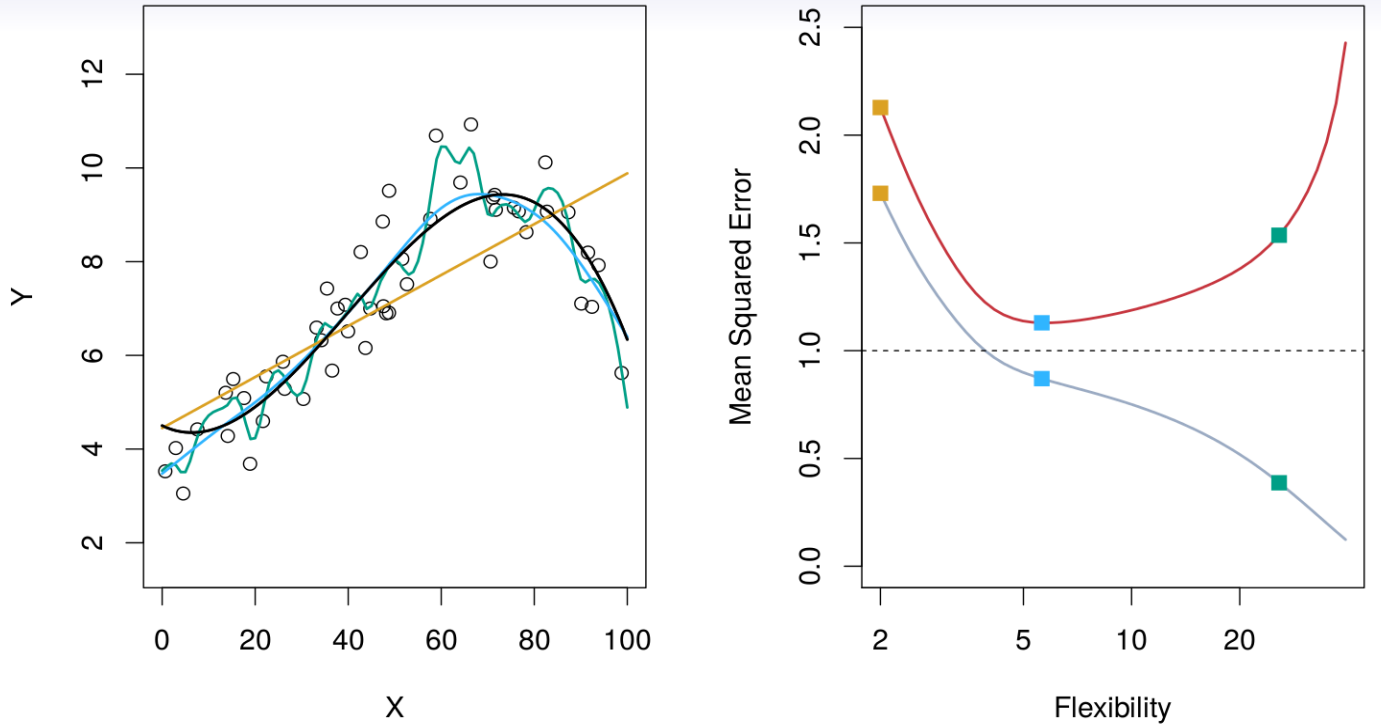


Figure 1: Left: Data simulated from f, shown in black. Three estimates of f are shown: the linear regression line (orange curve), and two smoothing spline

Black curve is truth. Red curve on right is $\text{MSE}_{\texttt{Te}}$, grey curve is $\text{MSE}_{\texttt{Tr}}$. Orange, blue and green curves/squares correspond to fits of different flexibility.

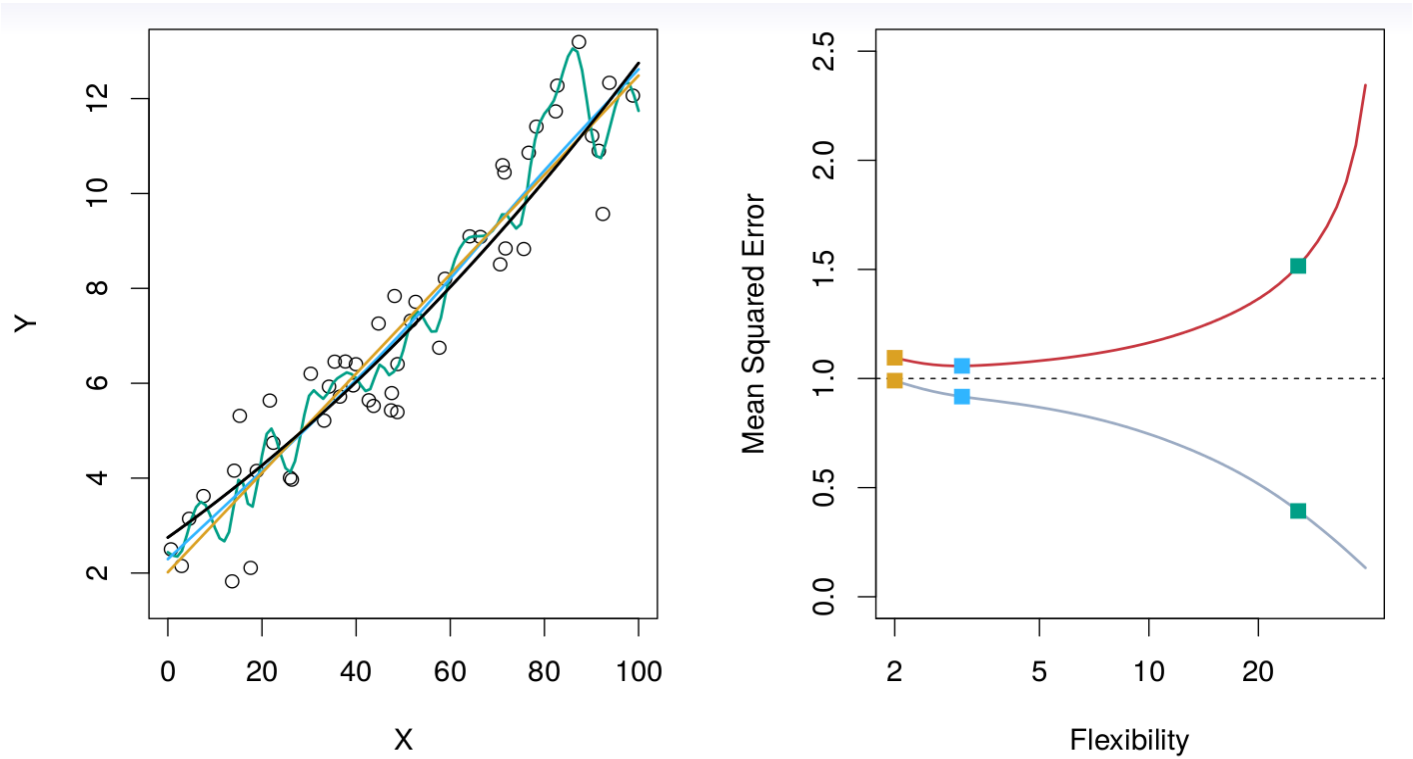


Figure 2: Details are as in Figure 1, using a different true f that is much closer to linear. In this setting, linear regression provides a very good fit to the data.

Here the truth is smoother, so the smoother fit and linear model do really well.

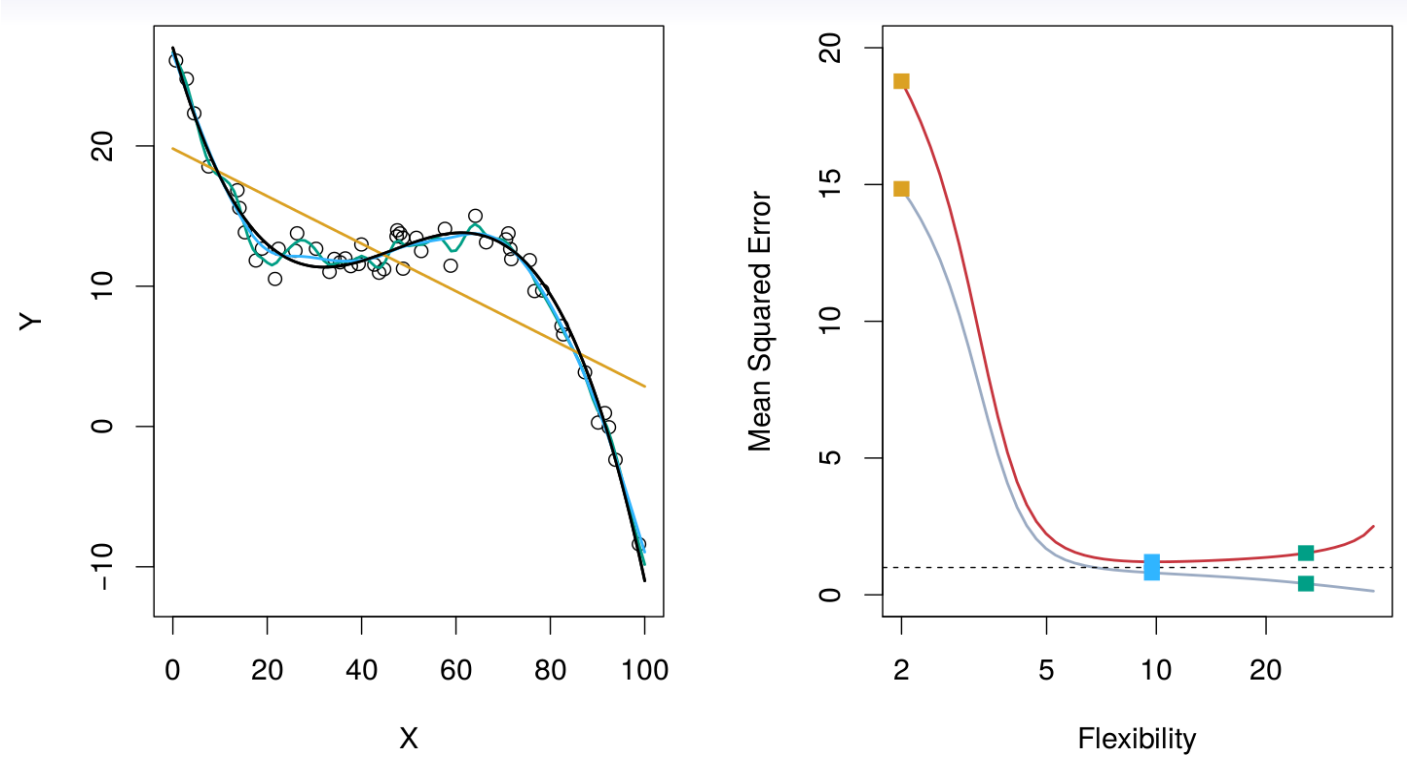


Figure 3: Details are as in Figure 1, using a different f that is far from linear. In this setting, linear regression provides a very poor fit to the data.

Here the truth is wiggly and the noise is low, so the more flexible fits do the best.

Bias-Variance Trade-off

Suppose we have fit a model $\hat{f}(x)$ to some training data $\texttt{Tr}$, and let (x_0, y_0) be a test observation drawn from the population. If the true model is $Y = f(X) + \varepsilon$ (with $f(x) = E(Y|X = x)$), then

$$\mathbb{E}\left(y_0 - \hat{f}(x_0)\right)^2 = \text{Var}(\hat{f}(x_0)) + [\text{Bias}(\hat{f}(x_0))]^2 + \text{Var}(\varepsilon).$$

The expectation averages over the variability of y_0 as well as the variability in $\texttt{Tr}$. Note that $\text{Bias}(\hat{f}(x_0)) = \mathbb{E}[\hat{f}(x_0)] - f(x_0)$. Typically as the flexibility of $\hat{f}$ increases, its variance increases, and its bias decreases. So choosing the flexibility based on average test error amounts to a bias-variance trade-off.

Full Probabilistic Formulation

For a complete probabilistic formulation, we also need to incorporate the selections for the training sample and define:

- $\mathcal{Y}$: $\mathcal{Y} = \mathbb{R}^d$ for regression problems or $\mathcal{Y} = \{c_1, \ldots, c_K\}$ for classification into K classes.
- Y: $\mathcal{Y}$-valued random variable modeling the distribution of the target.
- $\mathcal{X}$: typically $\mathcal{X} = \mathbb{R}^p$, space of predictors
- X: $\mathcal{X}$-valued random variable modeling the distribution of the predictors x_i.
- $\mathcal{N}$: $\mathcal{N} = (\mathcal{Y} \times \mathcal{X})^N$, space containing all training sample of size N
- $\mathbf{N}$: random variable representing all training samples of size N. Independent of X and Y.

All random variables $X, Y, \mathbf{N}$ are defined on a joint probability space $(\Omega, \mathcal{S}, \mathbb{P})$.

Error decomposition

To assess the performance of a prediction $\hat{f}_{\mathbf{N}}(X)$ which was trained by a random sample of N observation from $\mathcal{Y} \times \mathcal{X}$, we fix a random predictor $x_0 := X(\omega)$ and derive for the mean squared error:

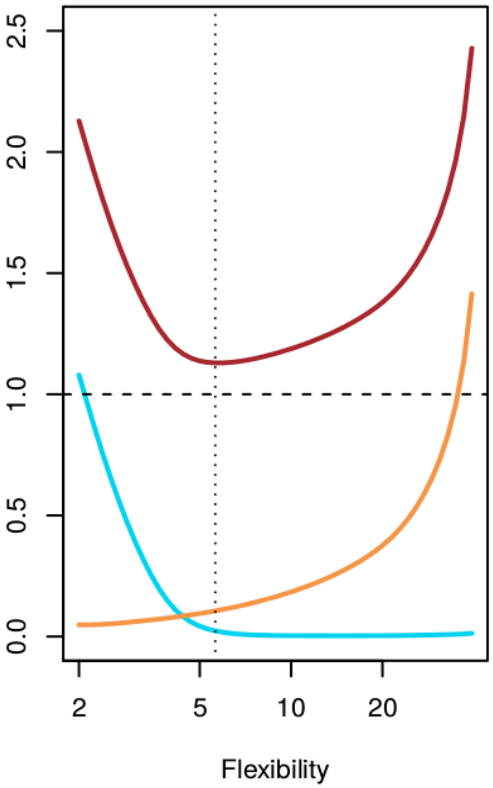
$$\begin{aligned}
\mathrm{MSE}(x_0) &:= \mathbb{E}\left(\left[Y - \hat{f}_{\mathbf{N}}(X)\right]^2 | X = x_0\right) \\
&= \mathbb{E}\left(\left[Y - \mathbb{E}(Y|X=x_0)\right]^2 | X = x_0\right) \\
&\quad + \left[\mathbb{E}(Y|X=x_0) - \mathbb{E}(\hat{f}_{\mathbf{N}}(X)|X=x_0)\right]^2 \\
&\quad + \mathbb{E}\left(\left[\hat{f}_{\mathbf{N}}(X) - \mathbb{E}(\hat{f}_{\mathbf{N}}(X)|X=x_0)\right]^2 | X = x_0\right)
\end{aligned}$$

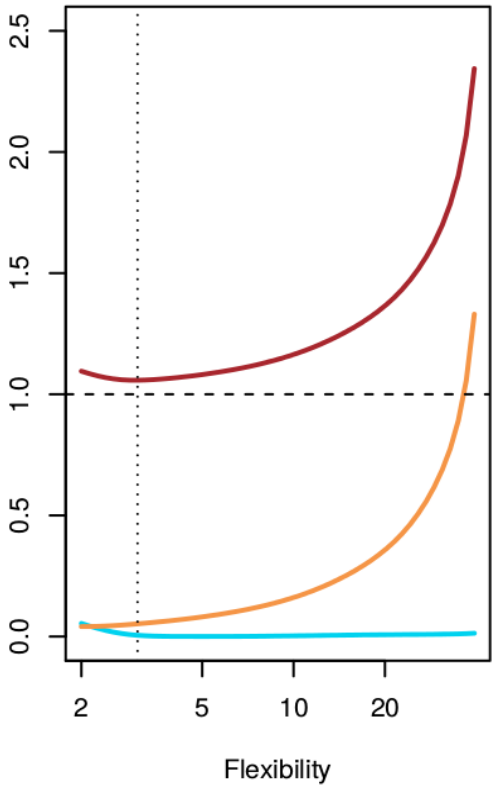
$\sigma^2(Y|X=x_0)$ *irreducible error*

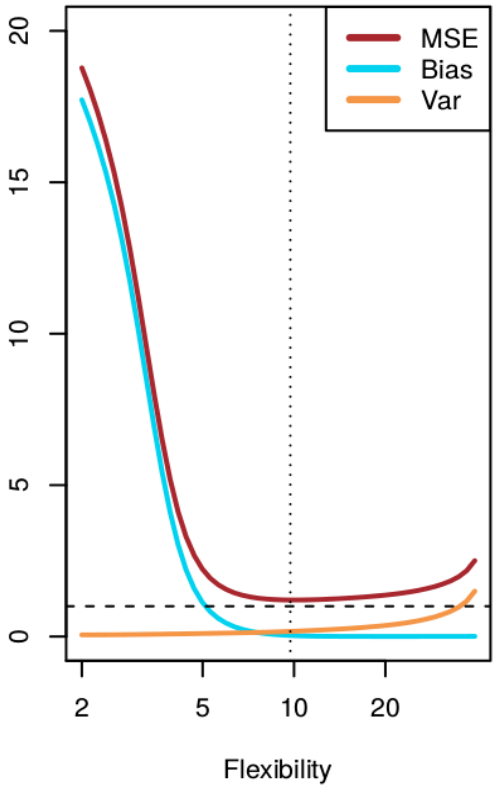
$\left(\mathbb{E}(Y|X=x_0) - \mathbb{E}(\hat{f}_{\mathbf{N}}(X)|X=x_0)\right)^2$ model bias

$\sigma^2(\hat{f}_{\mathbf{N}}(X)|X=x_0)$ model variance

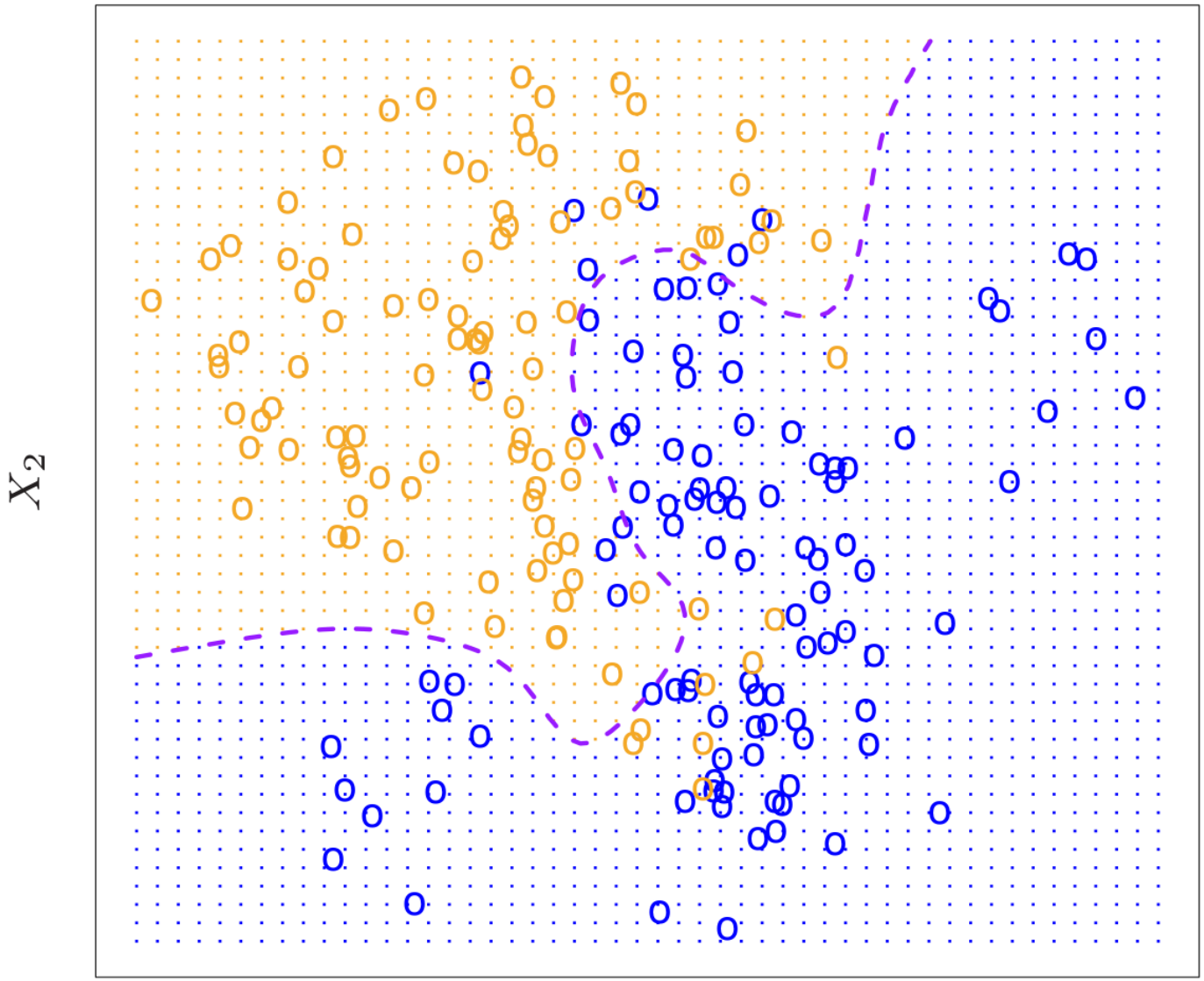
Bias-variance trade-off for the three examples

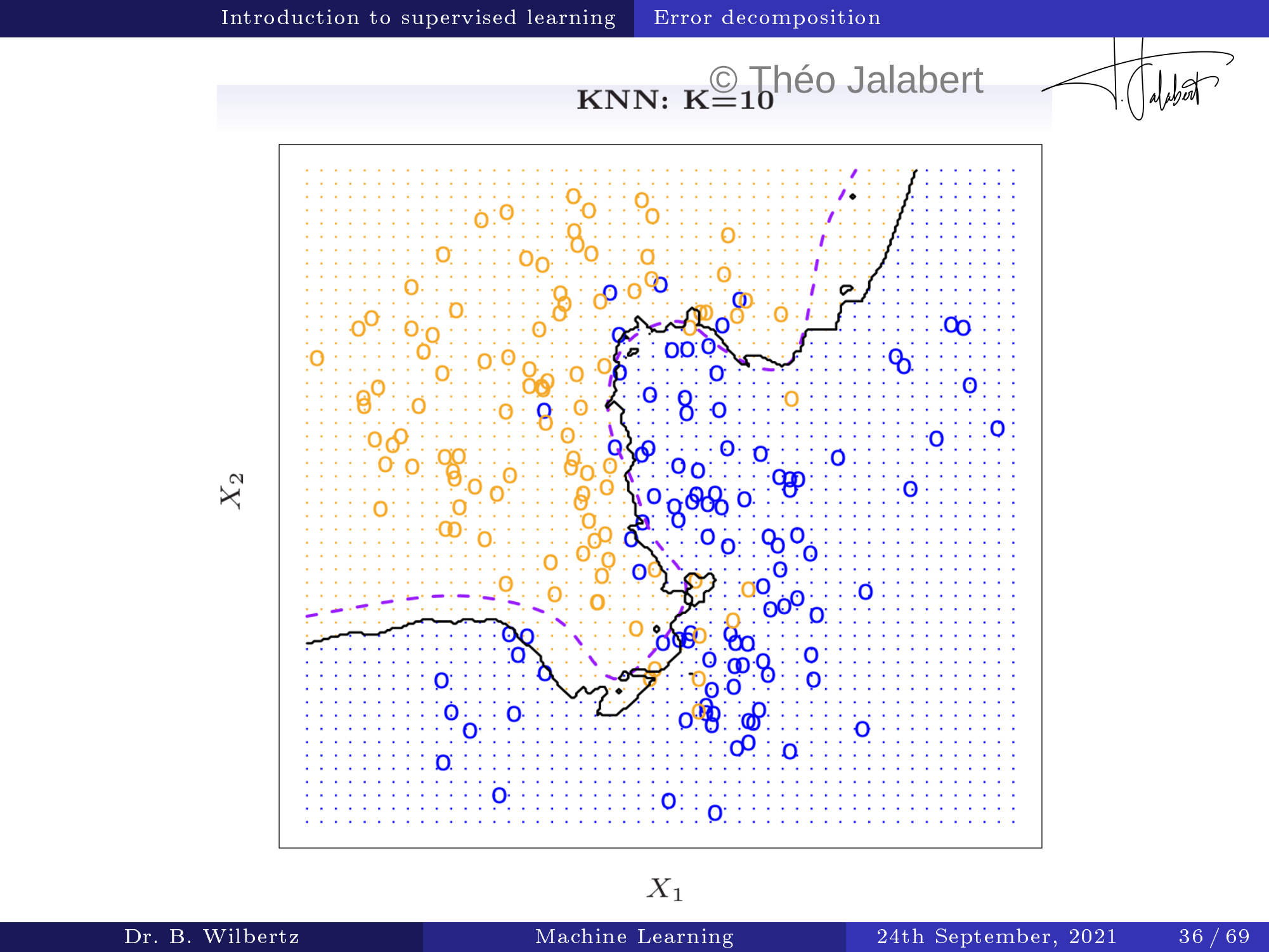






Example: K-nearest neighbors in two dimensions

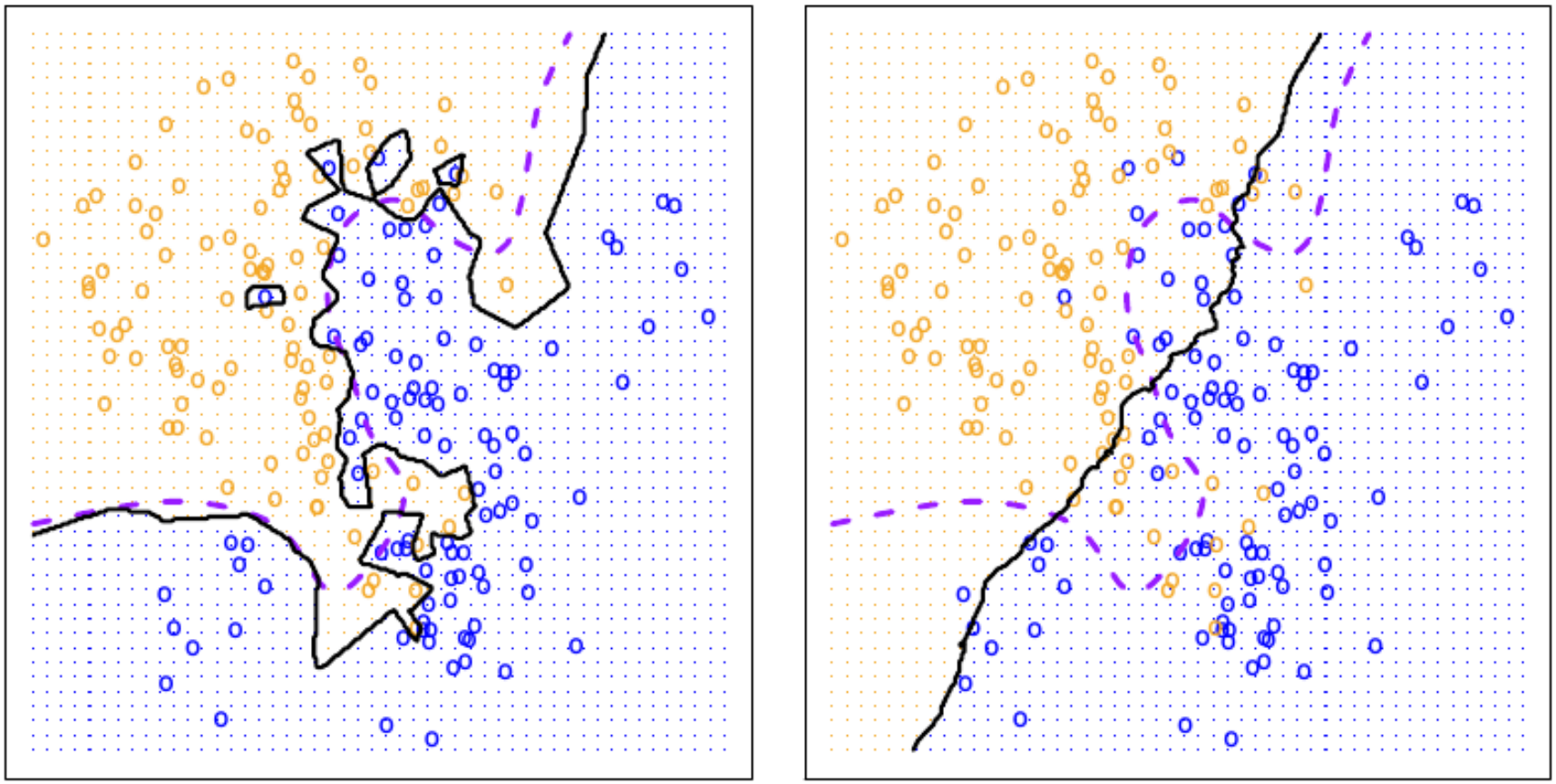


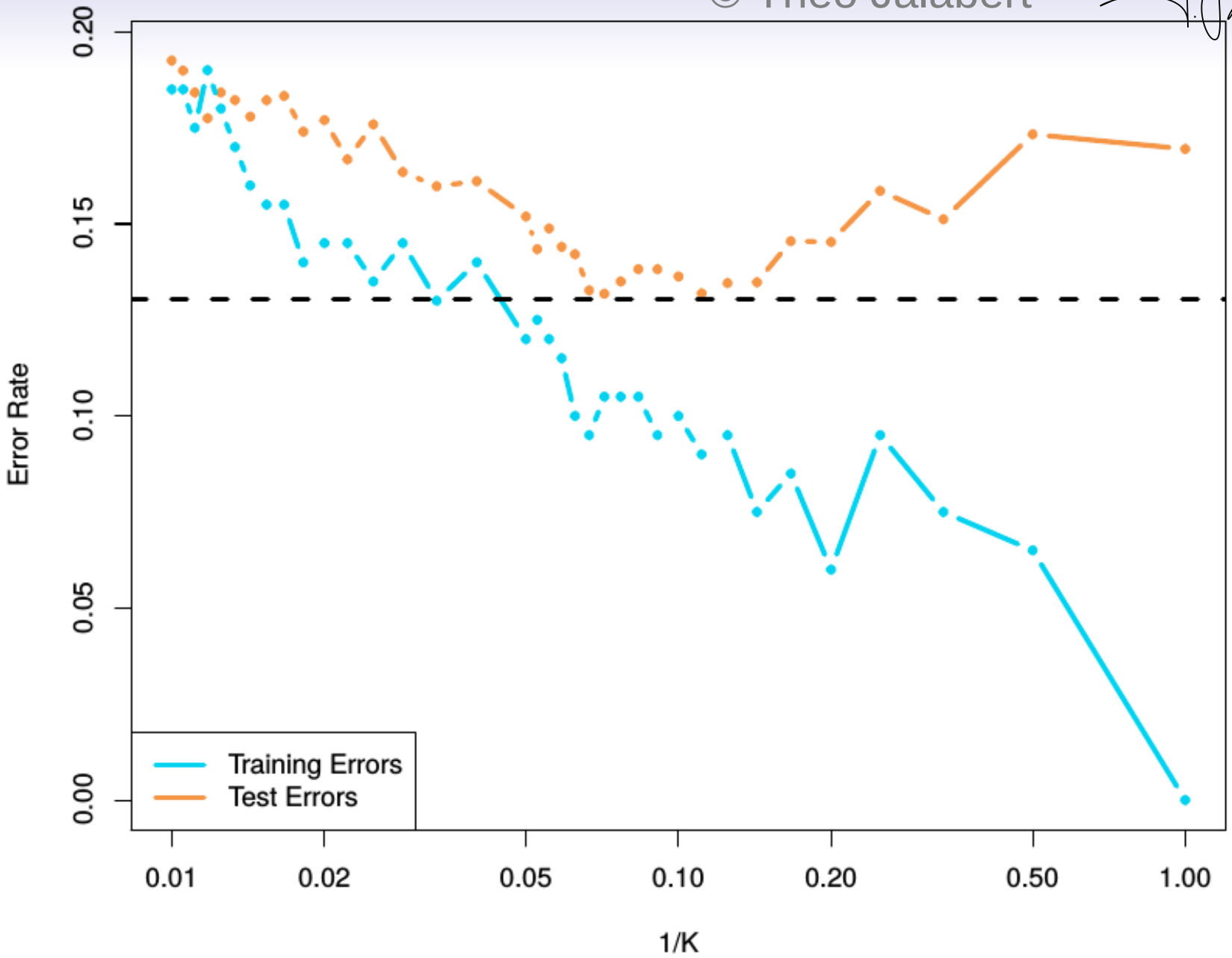


KNN: K=10

X_2

X_1





Cross-validation and the Bootstrap

- Recall the distinction between *the test error* and the *training error*:
- The *test error* is the average error that results from using a statistical learning method to predict the response on a new observation, one that was not used in training the method.
- In contrast, the *training error* can be easily calculated by applying the statistical learning method to the observations used in its training.
- But the training error rate often is quite different from the test error rate, and in particular the former can *dramatically underestimate* the latter.

More on prediction-error estimates

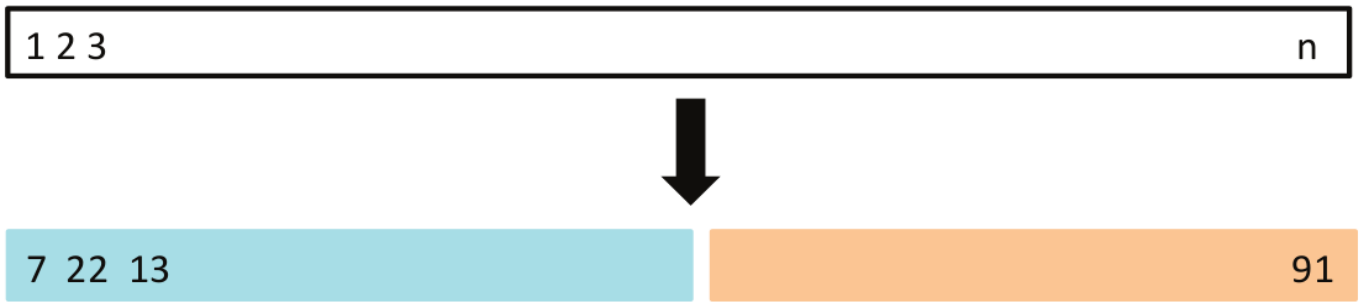
- Best solution: a large designated test set. Often not available
- Some methods make a *mathematical adjustment* to the training error rate in order to estimate the test error rate. These include the *Cp statistic*, *AIC* and *BIC*. But this adjustement is only valid for linear methods.
- Here we instead consider a class of methods that estimate the test error by *holding out* a subset of the training observations from the fitting process, and then applying the statistical learning method to those held out observations

Validation-set approach

Validation-set approach

- Here we randomly divide the available set of samples into two parts: a *training set* and a *validation* or *hold-out* set.
- The model is fit on the training set, and the fitted model is used to predict the responses for the observations in the validation set.
- The resulting validation-set error provides an estimate of the test error. This is typically assessed using MSE in the case of a quantitative response and misclassification rate in the case of a qualitative (discrete) response.

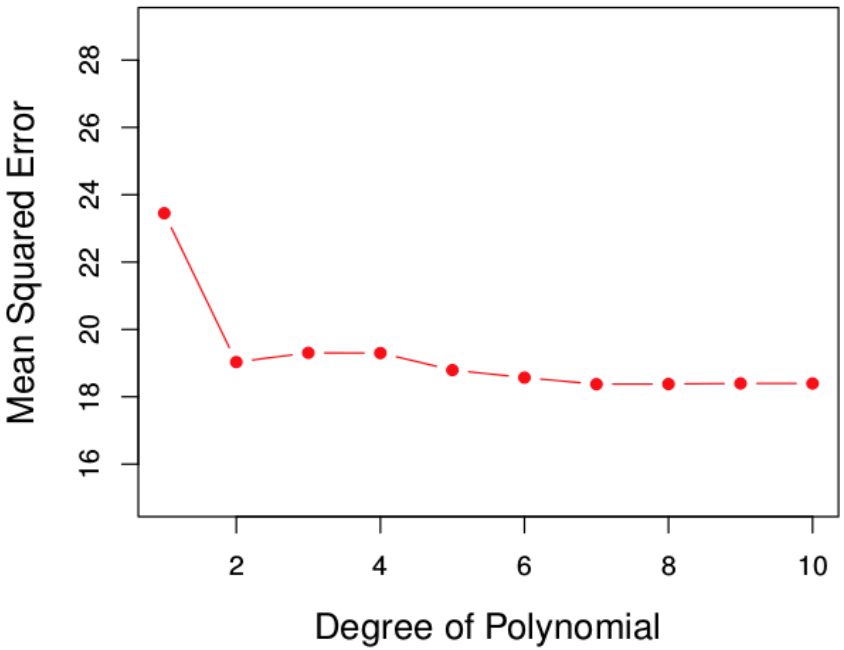
The Validation process

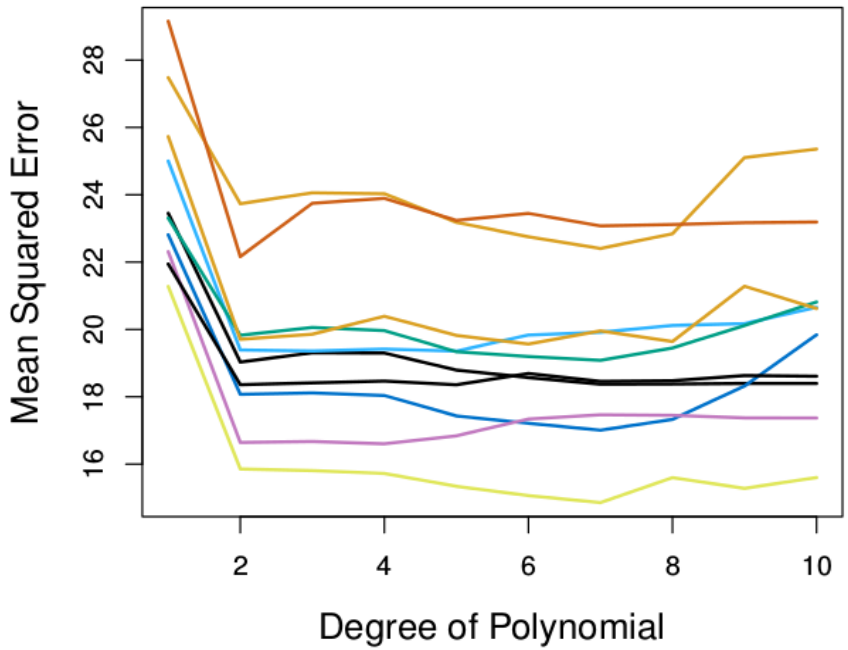


A random splitting into two halves: left part is training set, right part is validation set

Example: Automobile Data

- Want to compare linear vs higher-order polynomial terms in a linear regression
- We randomly split the 392 observations into two sets, a training set containing 196 of the data points, and a validation set containing the remaining 196 observations.





Drawbacks of validation set approach

Drawbacks 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.
- In the validation approach, only a subset of the observations – those that are included in the training set rather than in the validation set – are used to fit the model.
- This suggests that the validation set error may tend to *overestimate* the test error for the model fit on the entire data set.

K-fold Cross-validation

K-fold Cross-validation

- Widely used approach for estimating test error.
- Estimates can be used to select best model, and to give an idea of the test error of the final chosen model.
- Idea is to randomly divide the data into K equal-sized parts. We leave out part k, fit the model to the other $K-1$ parts (combined), and then obtain predictions for the left-out kth part.
- This is done in turn for each part $k = 1, 2, ...K$, and then the results are combined.

K-fold Cross-validation in detail

Divide data into K roughly equal-sized parts ($K = 5$ here)

1	2	3	4	5
Validation	Train	Train	Train	Train

The details

- Let the K parts be $C_1, C_2, \ldots, C_K$, where C_k denotes the indices of the observations in part k. There are n_k observations in part k: if N is a multiple of K, then $n_k = n/K$.
- Compute

$$\mathrm{CV}_{(K)} = \sum_{k=1}^{K} \frac{n_k}{n} \mathrm{MSE}_k$$

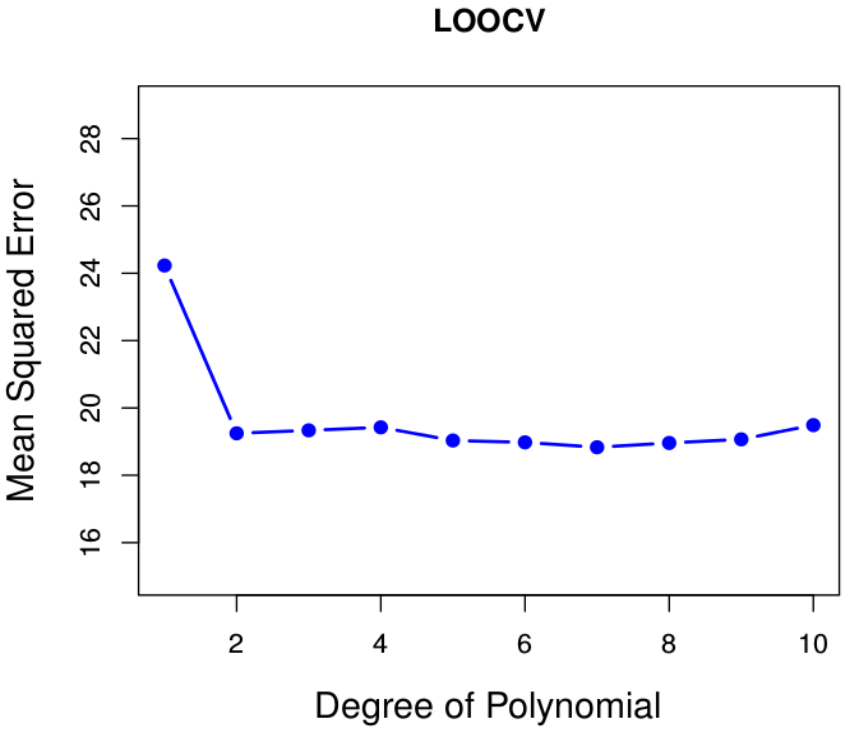
  where $\mathrm{MSE}_k = \sum_{i \in C_k} (y_i - \hat{y}_i)^2 / n_k$, and $\hat{y}_i$ is the fit for observation i, obtained from the data with part k removed.
- Setting $K = n$ yields n-fold or *leave-one out cross-validation* (LOOCV).

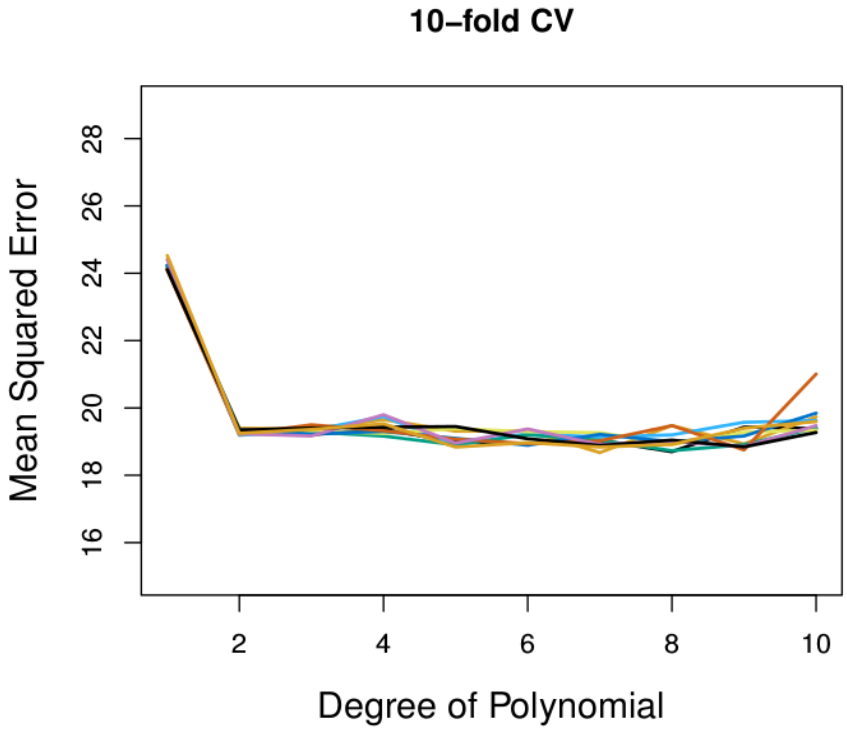
A nice special case!

- With least-squares linear or polynomial regression, an amazing shortcut makes the cost of LOOCV the same as that of a single model fit! The following formula holds:
- $$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2,$$

  where $\hat{y}_i$ is the ith fitted value from the original least squares fit, and h_i is the leverage (diagonal of the "hat" matrix). This is like the ordinary MSE, except the ith residual is divided by $1 - h_i$.
- LOOCV sometimes useful, but typically doesn't *shake up* the data enough. The estimates from each fold are highly correlated and hence their average can have high variance.
- a better choice is $K = 5$ or 10.

Auto data revisited





True and estimated test MSE for the simulated data

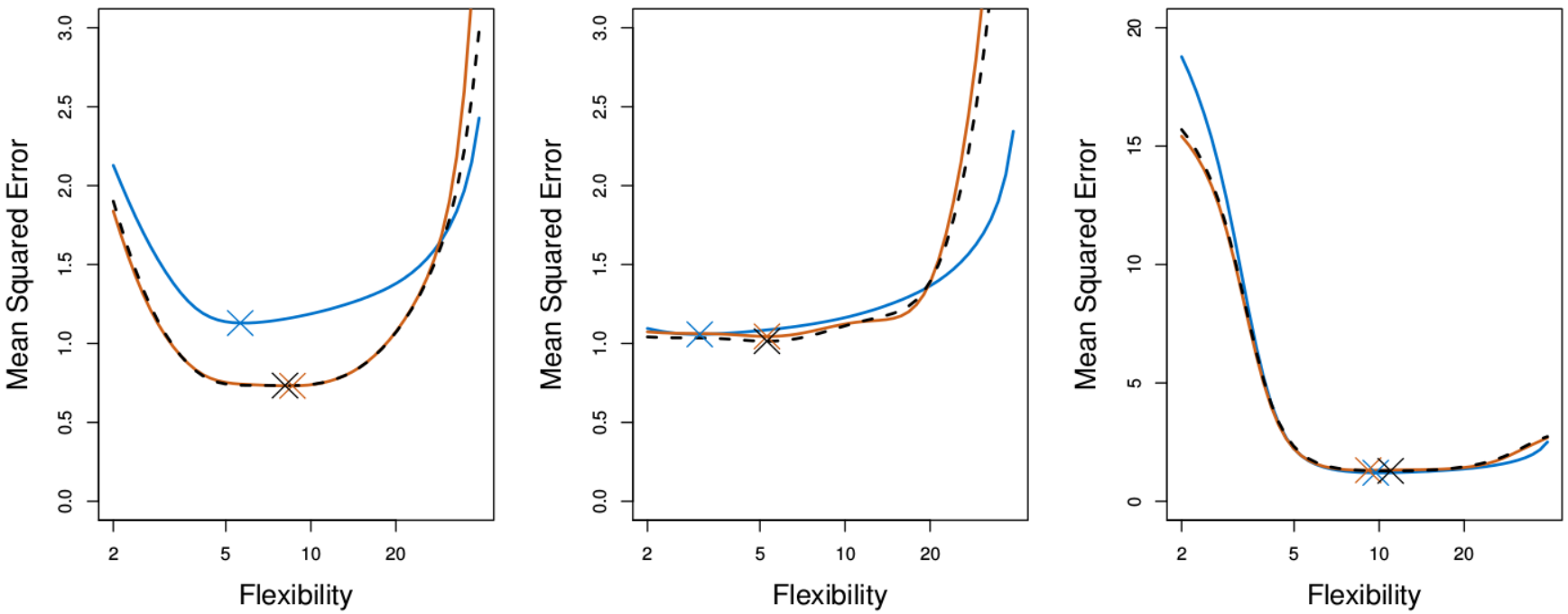


Figure 4: Blue curve: true error (from simulated data), dashed black line: LOOCV, orange line: 10-CV

Other issues with Cross-validation

Other issues with Cross-validation

- Since each training set is only $(K-1)/K$ as big as the original training set, the estimates of prediction error will typically be biased upward.
- This bias is minimized when $K = n$ (LOOCV), but this estimate has high variance, as noted earlier.
- $K = 5$ or 10 provides a good compromise for this bias-variance tradeoff.

The Bootstrap

The Bootstrap

- The bootstrap is a flexible and powerful statistical tool that can be used to quantify the uncertainty associated with a given estimator or statistical learning method.
- For example, it can provide an estimate of the standard error of a coefficient, or a confidence interval for that coefficient.

Where does the name came from?

Where does the name came from?

- The use of the term bootstrap derives from the phrase *to pull oneself up by one's bootstraps*, widely thought to be based on one of the eighteenth century "The Surprising Adventures of Baron Münchhausen" by Rudolph Erich Raspe: *The Baron had fallen to the bottom of a deep lake. Just when it looked like all was lost, he thought to pick himself up by his own bootstraps.*
- It is not the same as the term "bootstrap" used in computer science meaning to "boot" a computer from a set of core instructions, though the derivation is similar.

A simple example

- Suppose that we wish to invest a fixed sum of money in two financial assets that yield returns of X and Y, respectively, where X and Y are random quantities.
- We will invest a fraction α of our money in X, and will invest the remaining $1-\alpha$ in Y .
- We wish to choose α to minimize the total risk, or variance, of our investment. In other words, we want to minimize $\mathrm{Var}(\alpha X + (1-\alpha)Y)$.
- One can show that the value that minimizes the risk is given by

$$\alpha = \frac{\sigma_Y^2 - \sigma_{XY}}{\sigma_X^2 + \sigma_Y^2 - 2\sigma_{XY}}$$

where $\sigma_X^2 = \mathrm{Var}(X), \sigma_Y^2 = \mathrm{Var}(Y)$ and $\sigma_{XY} = \mathrm{Cov}(X, Y)$.

Example continued

- But the values of σ_X^2, σ_Y^2 and σ_{XY} are unknown.
- We can compute estimates for these quantities, $\hat{\sigma}_X^2, \hat{\sigma}_Y^2$ and $\hat{\sigma}_{XY}$, using a data set that contains measurements for X and Y.
- We can then estimate the value of α that minimizes the variance of our investment using

$$\hat{\alpha} = \frac{\hat{\sigma}_Y^2 - \hat{\sigma}_{XY}}{\hat{\sigma}_X^2 + \hat{\sigma}_Y^2 - 2\hat{\sigma}_{XY}}$$

Example continued

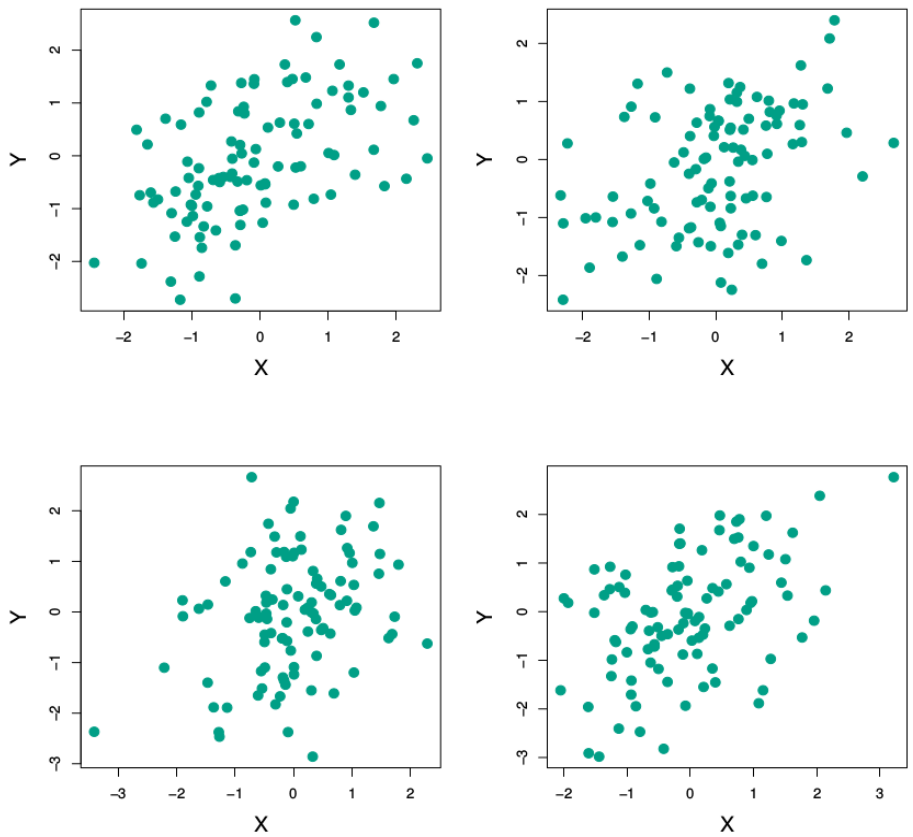


Figure 5: Each panel displays 100 simulated returns for investments X and Y . From left to right and top to bottom, the resulting estimates for α are 0.576, 0.532, 0.657, and 0.651.

Example continued

- To estimate the standard deviation of $\hat{\alpha}$, we repeated the process of simulating 100 paired observations of X and Y , and estimating α 1,000 times.
- We thereby obtained 1,000 estimates for α, which we can call $\hat{\alpha}_1, \hat{\alpha}_2, \ldots, \hat{\alpha}_{1000}$.
- The left-hand panel of the Figure on the next slides displays a histogram of the resulting estimates.
- For these simulations the parameters were set to $\sigma_X^2 = 1, \sigma_Y^2 = 1.25$, and $\sigma_{XY} = 0.5$, and so we know the true value of α is 0.6 (indicated by the red line).

Example continued

- The mean over all 1,000 estimates for α is

$$\bar{\alpha} = \frac{1}{1000} \sum_{r=1}^{1000} \hat{\alpha}_r = 0.5996$$

very close to $\alpha = 0.6$, and the standard deviation of the estimates is

$$\sqrt{\frac{1}{1000 - 1} \sum_{r=1}^{1000} (\hat{\alpha}_r - \bar{\alpha})^2} = 0.083.$$

- This gives us a very good idea of the accuracy of $\hat{\alpha}$: $\mathrm{SE}(\hat{\alpha}) \approx 0.083$.
- So roughly speaking, for a random sample from the population, we would expect $\hat{\alpha}$ to differ from α by approximately 0.08, on average.

Results

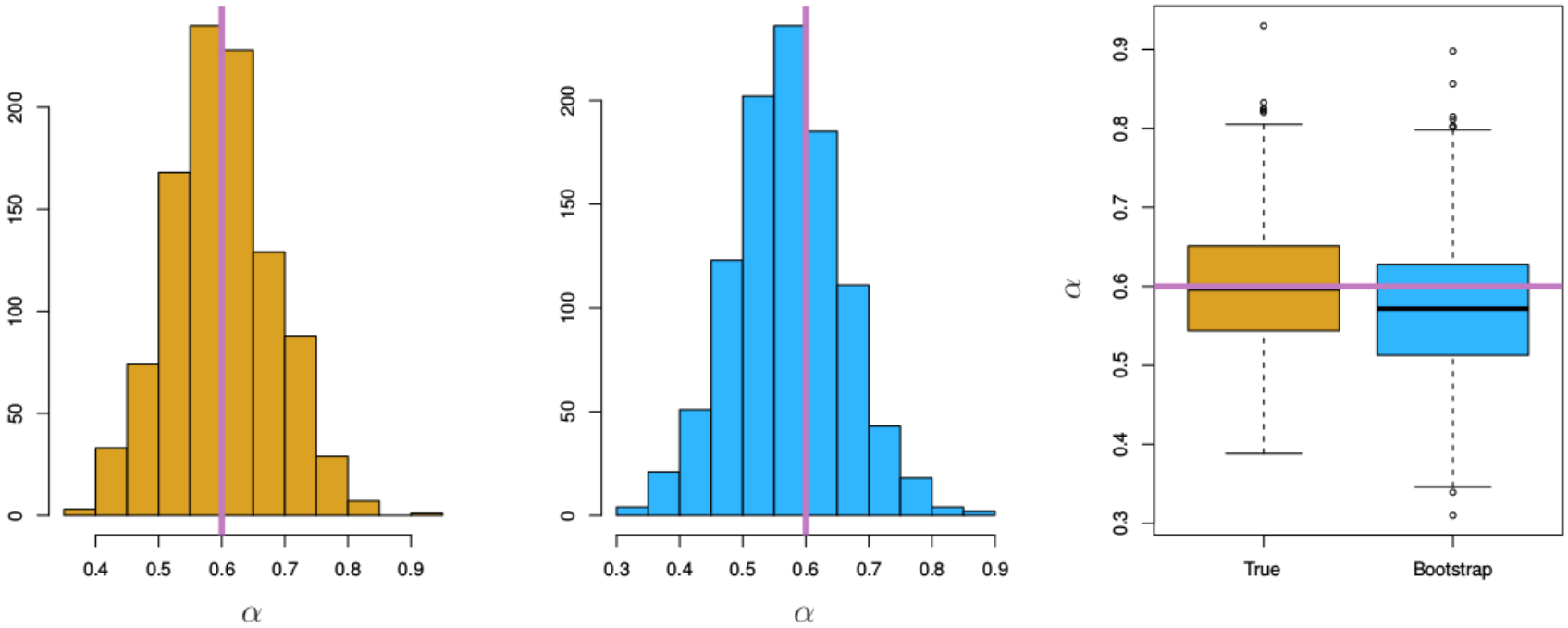
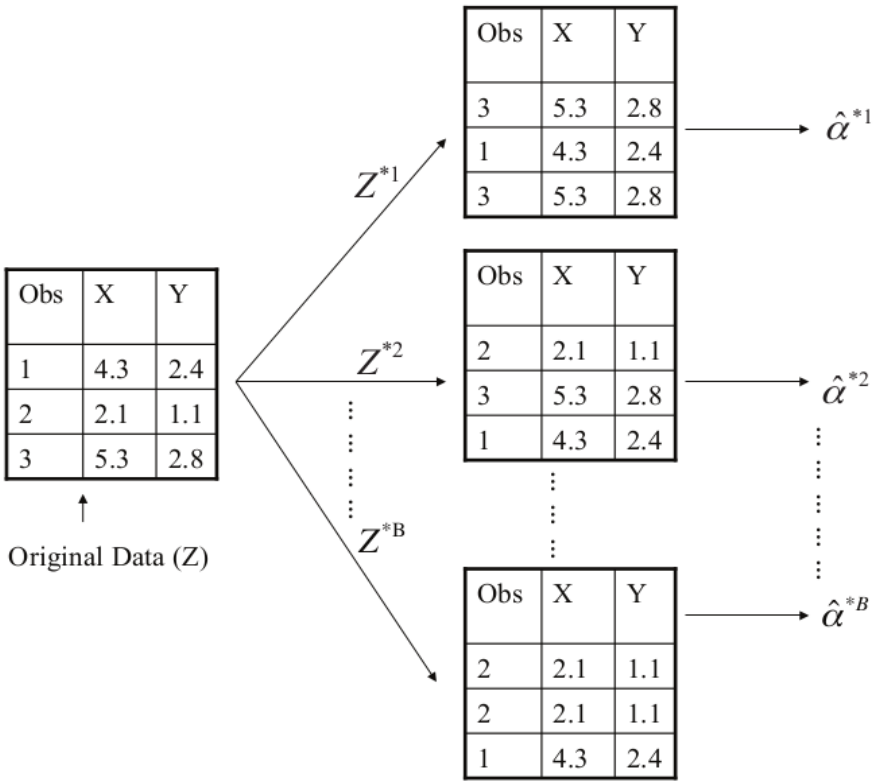


Figure 6: Left: A histogram of the estimates of α obtained by generating 1,000 simulated data sets from the true population. Center: A histogram of the estimates of α obtained from 1,000 bootstrap samples from a single data set. Right: The estimates of α displayed in the left and center panels are shown as boxplots. In each panel, the pink line indicates the true value of α.

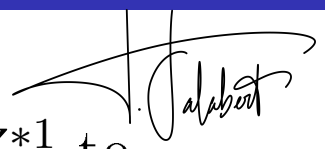
Now back to the real world

- The procedure outlined above cannot be applied, because for real data we cannot generate new samples from the original population.
- However, the bootstrap approach allows us to use a computer to mimic the process of obtaining new data sets, so that we can estimate the variability of our estimate without generating additional samples.
- Rather than repeatedly obtaining independent data sets from the population, we instead obtain distinct data sets by repeatedly sampling observations from the original data set *with replacement*.
- Each of these "bootstrap data sets" is created by sampling *with replacement*, and is the *same size* as our original dataset. As a result some observations may appear more than once in a given bootstrap data set and some not at all.

Example with just 3 observations



A graphical illustration of the bootstrap approach on a small sample containing $n = 3$ observations. Each bootstrap data set contains n observations, sampled with replacement from the original data set. Each bootstrap data set is used to obtain an estimate of α.



- Denoting the first bootstrap data set by Z^{*1} , we use Z^{*1} to produce a new bootstrap estimate for α, which we call $\hat{\alpha}^{*1}$
- This procedure is repeated B times for some large value of B (say 100 or 1000), in order to produce B different bootstrap data sets, $Z^{*1}, Z^{*2}, \ldots, Z^{*B}$, and B corresponding α estimates, $\hat{\alpha}^{*1}, \hat{\alpha}^{*2}, \ldots, \hat{\alpha}^{*B}$
- We estimate the standard error of these bootstrap estimates using the formula

$$\mathrm{SE}_B(\hat{\alpha}) = \sqrt{\frac{1}{B-1}\sum_{r=1}^{B}(\hat{\alpha}^{*r} - \bar{\hat{\alpha}}^{*})^2}$$

- This serves as an estimate of the standard error of $\hat{\alpha}$ estimated from the original data set. See center and right panels of Figure on previous slides. Bootstrap results are in blue. For this example $\mathrm{SE}_B(\hat{\alpha}) = 0.087$.

Other uses of the bootstrap

Other uses of the bootstrap

- Primarily used to obtain standard errors of an estimate.
- Also provides approximate confidence intervals for a population parameter. For example, looking at the histogram in the middle panel of the Figure 6, the 5% and 95% quantiles of the 1000 values is (.43, .72).
- This represents an approximate 90% confidence interval for the true α.
- The above interval is called a *Bootstrap Percentile* confidence interval. It is the simplest method for obtaining a confidence interval from the bootstrap.

Can the bootstrap estimate prediction error?

Can the bootstrap estimate prediction error?

- In cross-validation, each of the K validation folds is distinct from the other $K-1$ folds used for training: there is no overlap. This is crucial for its success.
- To estimate prediction error using the bootstrap, we could think about using each bootstrap dataset as our training sample, and the original sample as our validation sample.
- But each bootstrap sample has significant overlap with the original data. About two-thirds of the original data points appear in each bootstrap sample.
- This will cause the bootstrap to seriously underestimate the true prediction error.

Removing the overlap

Removing the overlap

- Can fix this problem by only using predictions for those observations that did not (by chance) occur in the current bootstrap sample.
- That way we get an alternative way of estimating prediction error, though it uses often more iterations than cross-validation

Performance Measures for Regression Models

Case $\mathcal{Y} = \mathbb{R}$

RMSE: Root-Mean-Squared Error

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

MAE: Mean-Absolute Error

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

R^2: R-Squared

$$\text{R}^2 = \left(\frac{\text{Cov}(y, \hat{y})}{\sigma(y)\sigma(\hat{y})}\right)^2$$

Proportion of Variance in y which is explained by $\hat{y}$.

R^2 is a measure of correlation, not accuracy. Hence even high R^2 values might lead to large deviations of $\hat{y} = \hat{f}(x)$ from y in some regions of x.

Performance Measures for Classification Models

Case $\mathcal{Y} = \{c_1, \ldots, c_k\}$

		True condition			
	Total population	Condition positive	Condition negative	Prevalence = $\frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$
Predicted condition	Predicted condition positive	**True positive**, Power	**False positive**, Type I error	Positive predictive value (PPV), Precision = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$
	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$
		True positive rate (TPR), Recall, Sensitivity, probability of detection = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR−}}$
		False negative rate (FNR), Miss rate = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) = $\frac{\text{FNR}}{\text{TNR}}$	F_1 score = $\frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$

Figure 7: $k = 2$, `https://en.wikipedia.org/wiki/Sensitivity_and_specificity`

Performance Measures for Classification Models

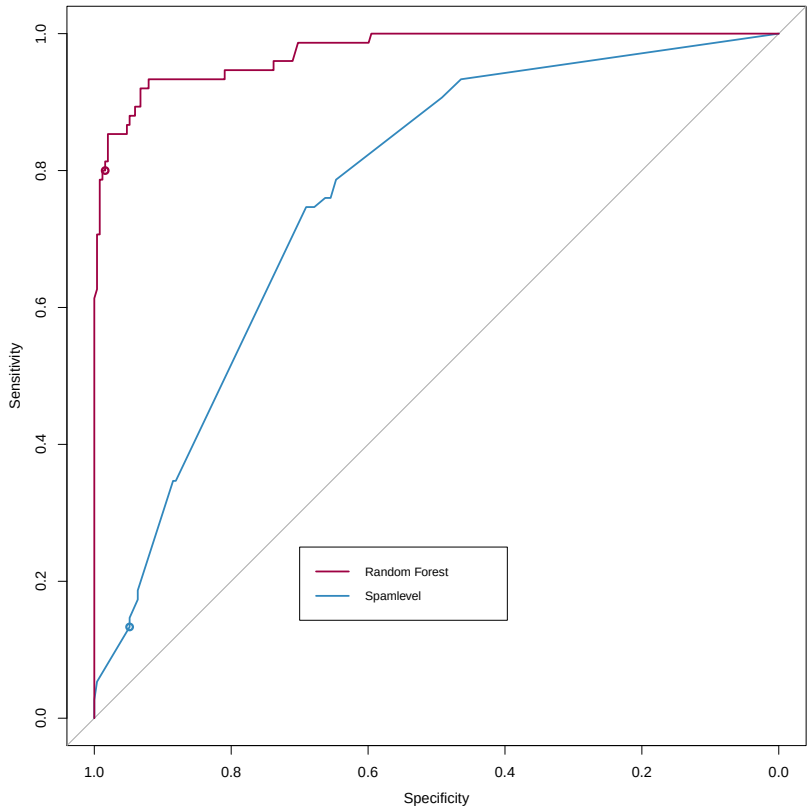
Most important measures

- Accuracy (TP + TN / all)
- Positive predictive value or Precision (TP / Predicted Positive)
- Sensitivity or Recall (TP / Ground Truth Positive)
- Specificity (TN / Ground Truth Negative)
- F_1 score (harmonic mean between Precision and Recall)
- ROC / AUC (Receiver-operator-curve / Area-under-the-curve)

Those measures should always be taken in relation to the No-Information-Rate (NIR) of dataset

Performance Measures for Classification Models

The ROC curve is created by plotting Sensitivity versus Specificity for different threshold of the classifier's confidence for the positive.



AUC is defined as the area under the ROC-Curve. Any uninformed classifier can always achive AUC $= 0.5$