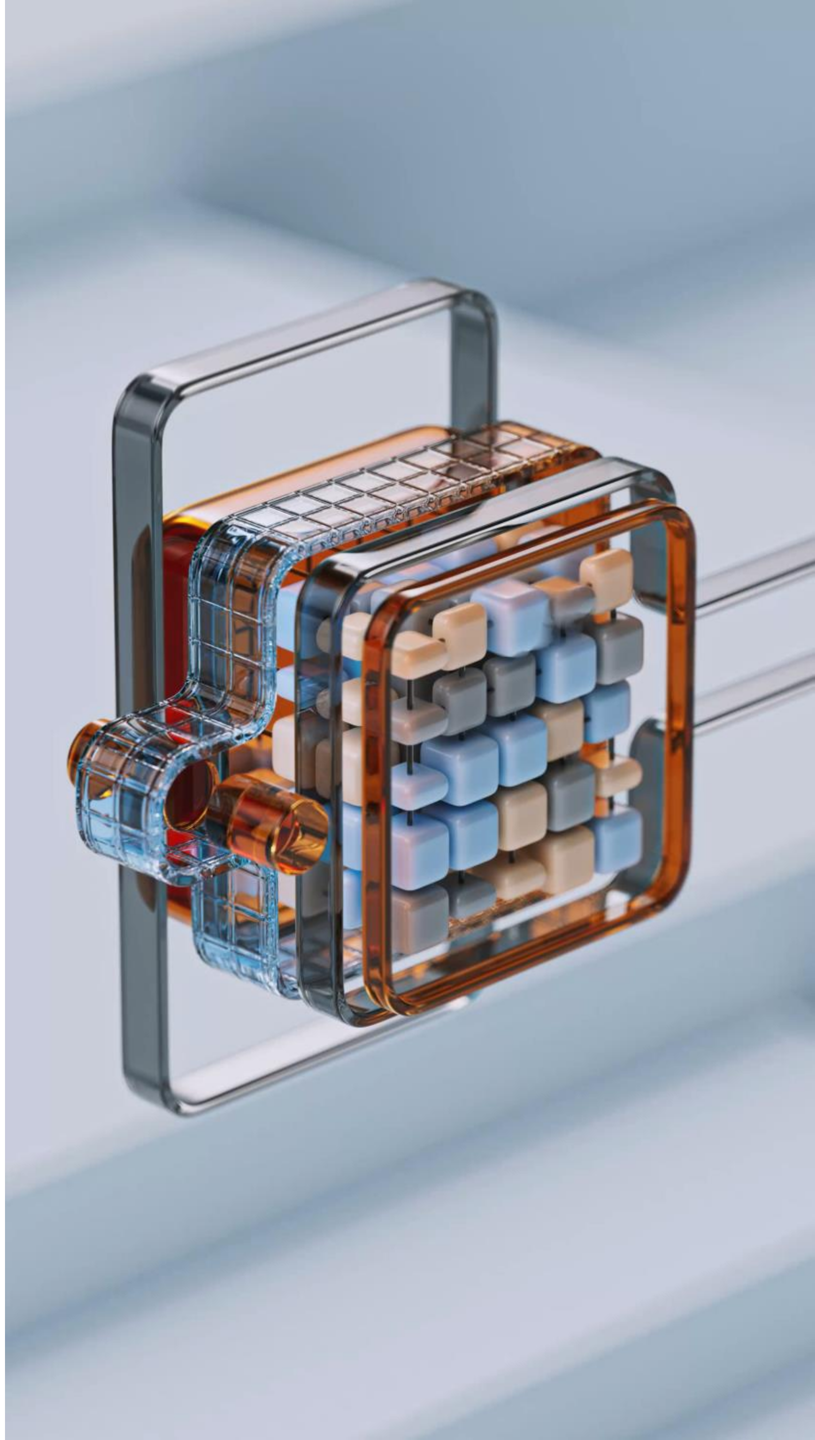
Battling the curse of dimensionality

Erik-Jan van Kesteren



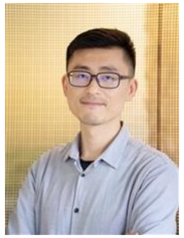
Today

- Introduction
- High-dimensional data / high dimensionality
- The curse of dimensionality
- Regularization / penalization
- Generalizations of the LASSO penalty
  - Elastic net
  - Group LASSO
- Implementation

Introduction

dr. Erik-Jan van Kesteren
e.vankesteren1@uu.nl
Course coordinator

dr. Dave Hessen
d.j.hessen@uu.nl
Lecturer – dimension reduction



dr. Qixiang Fang
q.fang@uu.nl
Lecturer – text mining

Thom Volker, MSc
t.b.volker@uu.nl
Lab teacher

Course website

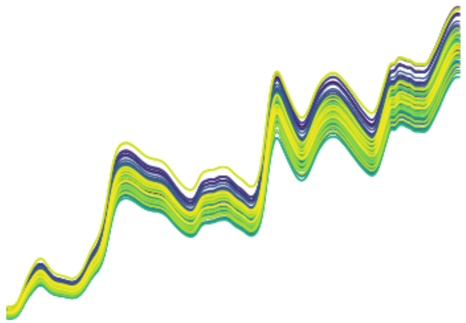
https://infomda2.nl/

Battling the Curse of Dimensionality

https://infomda2.nl

Requests - Service Desk NN SVG squash

INFOMDA2



Materials for Applied Data Science profile course INFOMDA2 *Battling the curse of dimensionality*.

Hosted on GitHub Pages — Theme by orderedlist

Battling the Curse of Dimensionality

This webpage contains all materials required for the Applied Data Science profile course INFOMDA2 ‘Battling the Curse of Dimensionality’.

Syllabus

You can find the course syllabus as a web page here or as a pdf here.

Lectures

Number	Title	Slides
01	High-dimensional data	Slides
02	Dimension reduction I	Slides
03	Dimension reduction II	Slides
04	Deep learning	Slides

Course overview

Week	Topic
1	Introduction & High-dimensional data
2	Dimension reduction I
3	Dimension reduction II
4	Clustering
5	Model-based clustering
6	Deep learning
7	Text mining I
8	Text mining II

Course proceedings

- 1 **lecture** per week (Wednesday)
  - Read required readings before lecture!
  - Ask questions!
- 1 **lab session** per week (Friday)
  - Take-home exercises before the lab session
  - Additional exercises during the lab session
- 1 **assignments**
  - 25% of your grade
  - With peer feedback round
  - make groups of 3 by end of the week
- 1 **exam** (and a resit)

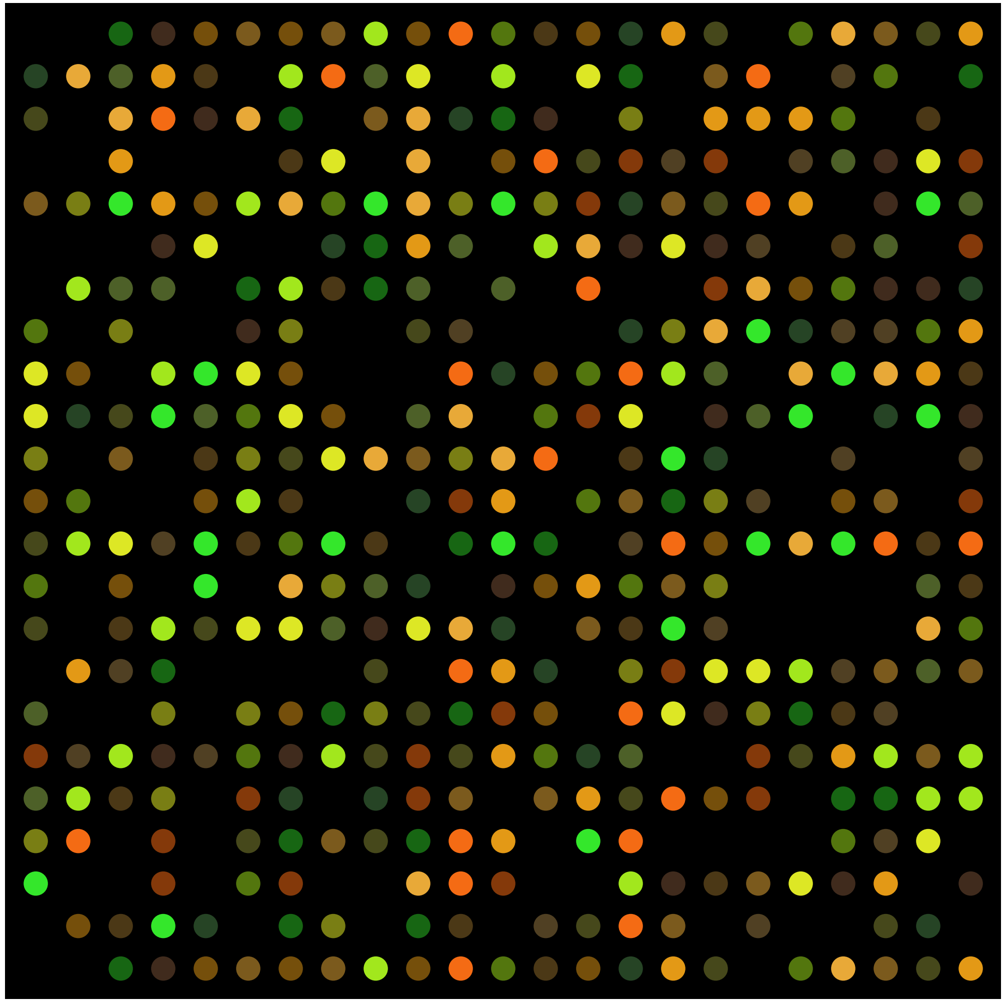
Read the syllabus!

A lot of your questions will be answered by reading the syllabus

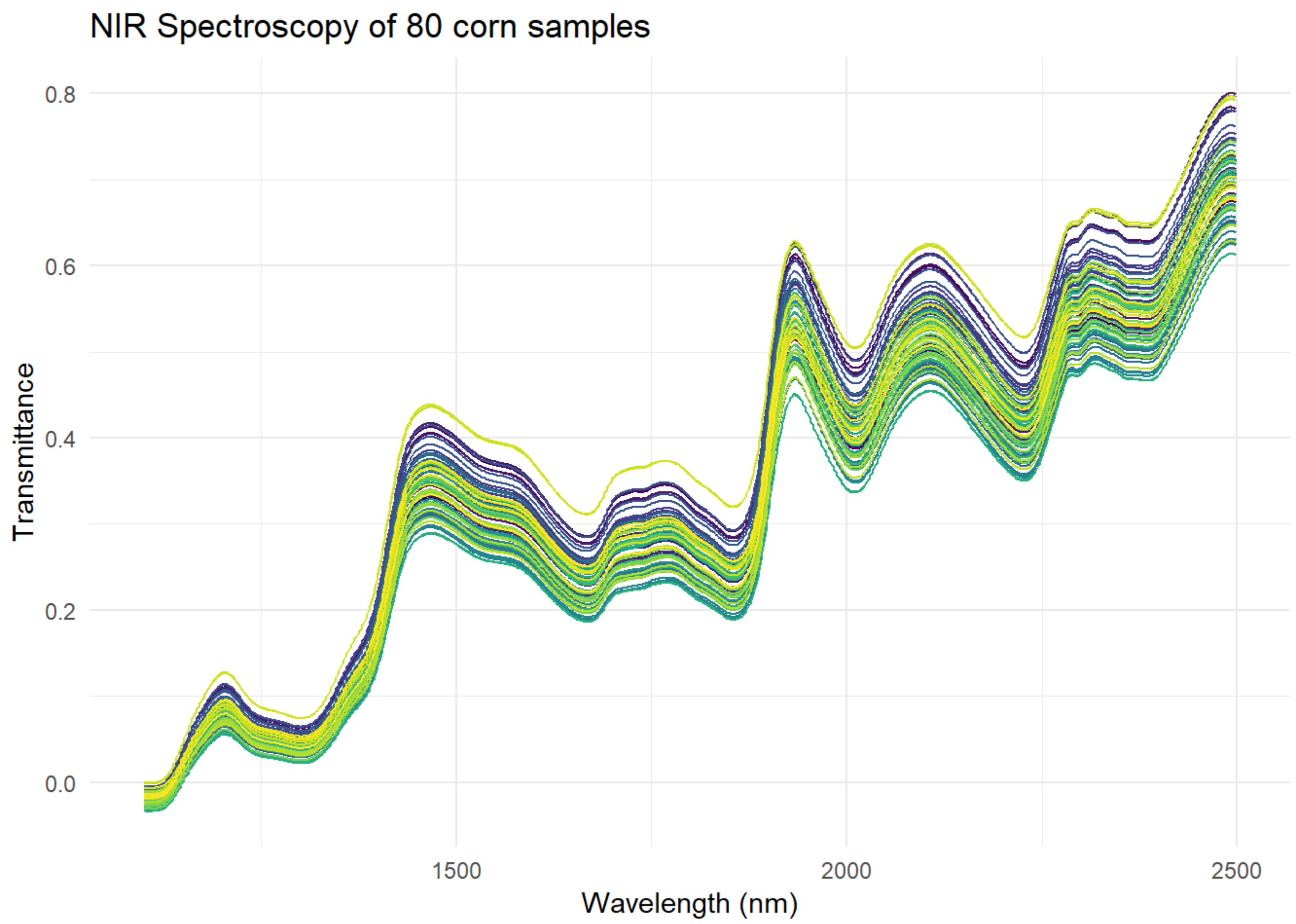
High-dimensionality

What is “high-dimensional”??

Example 1



Example 2



NIR Spectroscopy of 80 corn samples
0.8
0.6
0.4
0.2
0.0
Transmittance
1500
2000
2500
Wavelength (nm)

Example 3

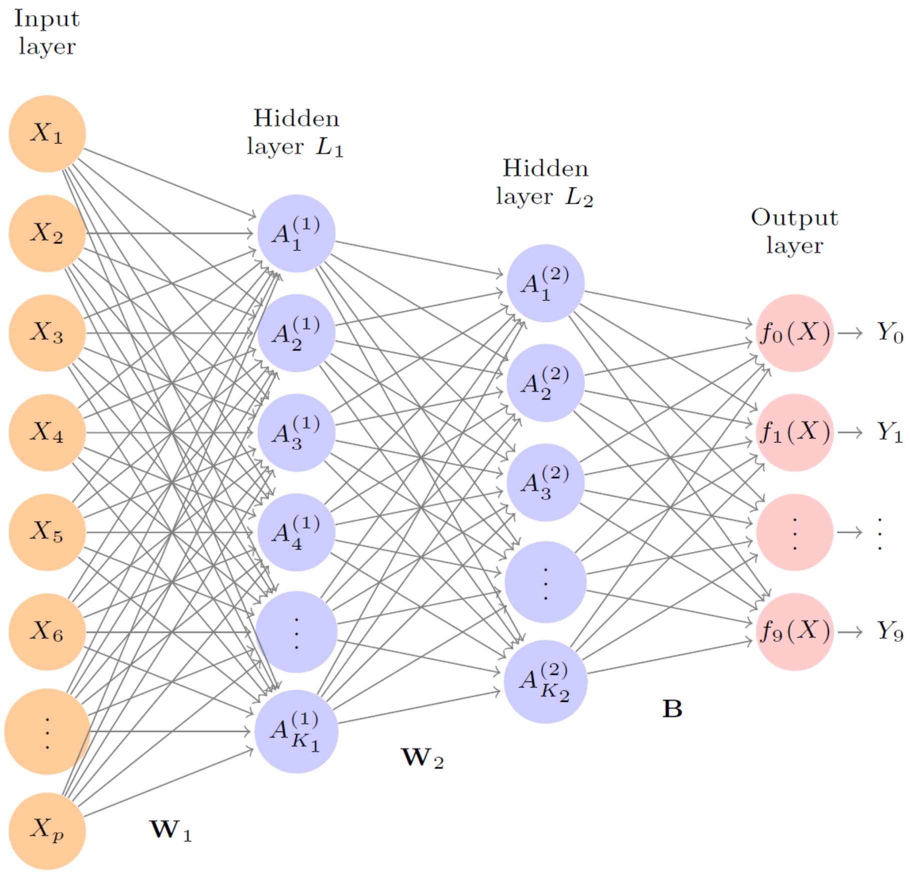
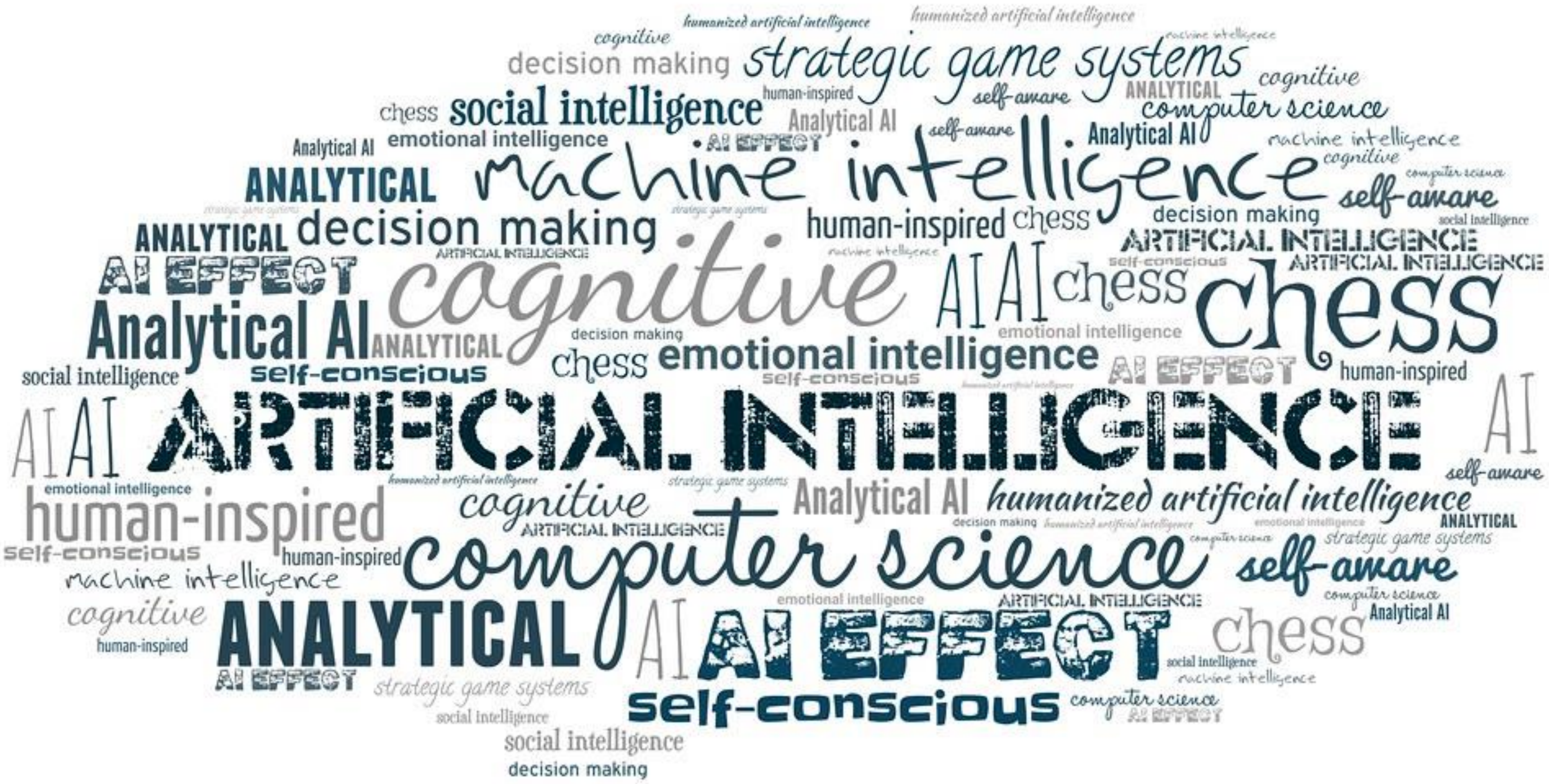


FIGURE 10.4. *Neural network diagram with two hidden layers and multiple*

Example 4


humanized artificial intelligence
cognitive
humanized artificial intelligence
decision making
strategic game systems
cognitive
chess
social intelligence
human-inspired
self-aware
ANALYTICAL
computer science
Analytical AI
emotional intelligence
AI EFFECT
self-aware
Analytical AI
machine intelligence
ANALYTICAL
machine intelligence
self-aware
ANALYTICAL
decision making
human-inspired
chess
decision making
ARTIFICIAL INTELLIGENCE
AI EFFECT
cognitive
AI
AI
chess
chess
Analytical AI
ANALYTICAL
decision making
emotional intelligence
social intelligence
self-conscious
chess
emotional intelligence
AI EFFECT
human-inspired
AI
AI
ARTIFICIAL INTELLIGENCE
AI
self-aware
emotional intelligence
human-inspired
cognitive
Analytical AI
humanized artificial intelligence
ANALYTICAL
ARTIFICIAL INTELLIGENCE
strategic game systems
self-conscious
human-inspired
computer science
self-aware
machine intelligence
cognitive
ANALYTICAL
AI
AI EFFECT
emotional intelligence
ARTIFICIAL INTELLIGENCE
Analytical AI
chess
human-inspired
AI EFFECT
strategic game systems
social intelligence
machine intelligence
self-conscious
computer science
social intelligence
decision making

High-dimensional

- Microarray dataset with many columns (P) and few rows (N)
- Spectroscopy dataset with many measured features and few observations
- Neural network with many parameters (weights) and few observations (examples)
- Text dataset with few documents and many unique words

High-dimensional

Many parameters (P) relative to the amount of information (N) to learn about those parameters

So what?

The curse of dimensionality

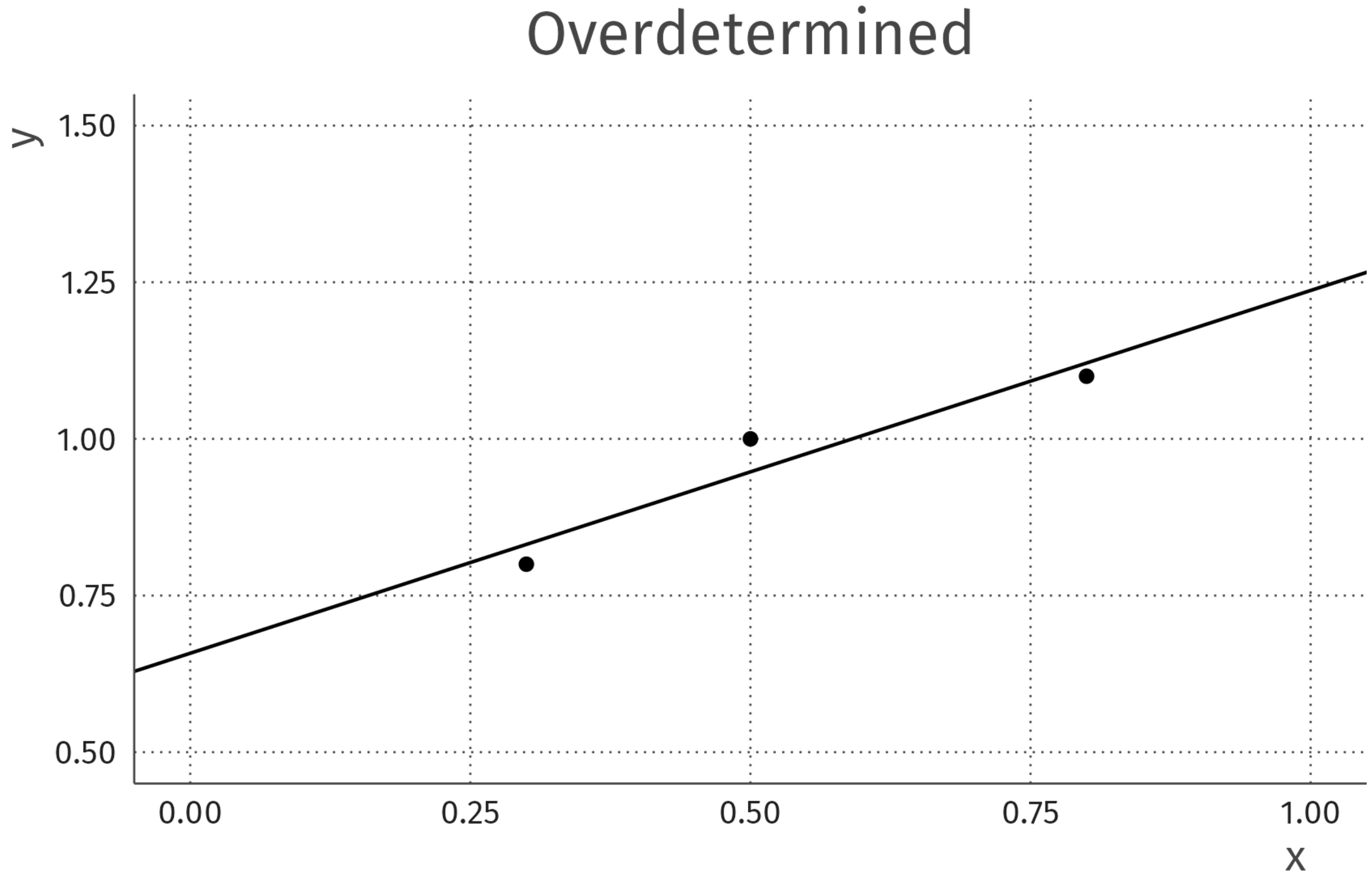
The curse of dimensionality

Let’s do linear regression!

x	y
0.5	1
0.3	0.8
0.8	1.1

```
lm(formula = y ~ x, data = dat)
```

The curse of dimensionality



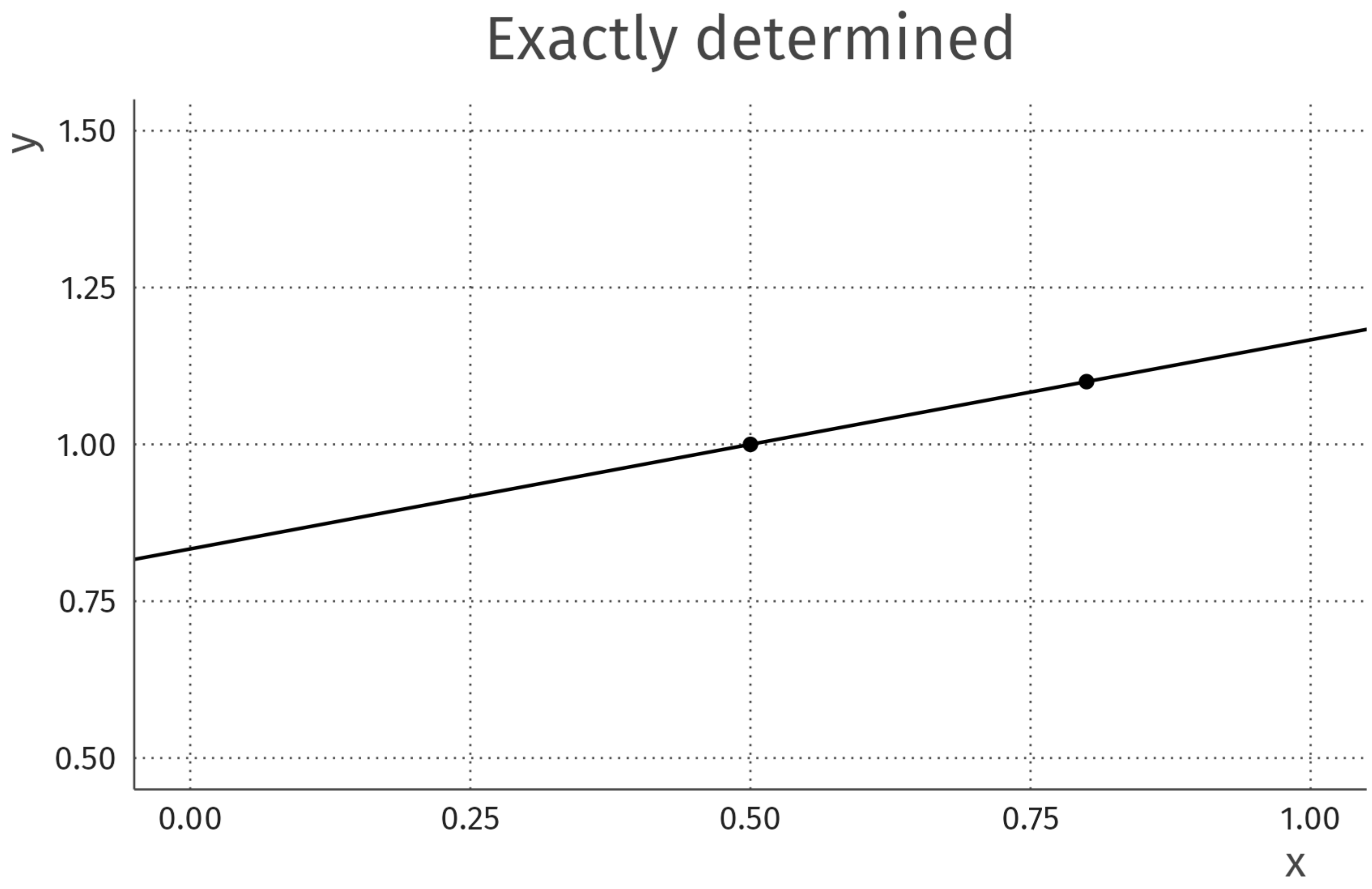
The curse of dimensionality

Let’s do linear regression!

x	y
0.5	1
0.8	1.1

```
lm(formula = y ~ x, data = dat)
```

The curse of dimensionality



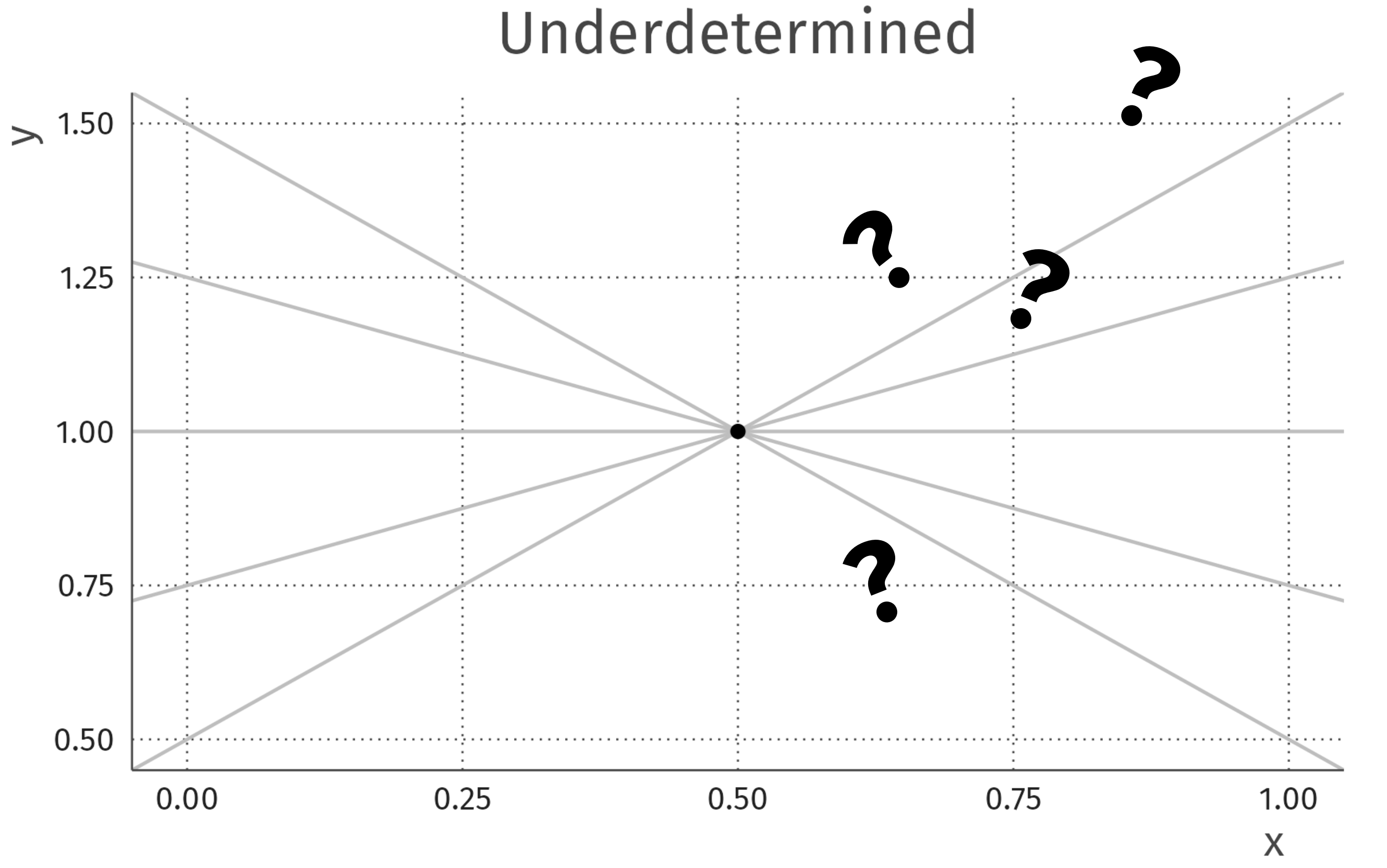
The curse of dimensionality

Let's do linear regression!

x	y
0.5	1

```
lm(formula = y ~ x, data = dat)
```

The curse of dimensionality



The curse of dimensionality

Overdetermined

With $P < N$, we estimate the parameters to best represent the data (e.g., using least-squares)

Exactly determined

- With $P = N$, we can always fit the data perfectly
- N points always fall on an $(N - 1)$-dimensional hyperplane (having $P = N$ parameters)

Underdetermined

- There are infinitely many lines going through a point
- There are infinitely many planes going through two points
- There are infinitely many N-dimensional hyperplanes going through N points

The curse of dimensionality

In R

x	y
0.5	1

```
Call:
lm(formula = y ~ x, data = dat)

Residuals:
ALL 1 residuals are 0: no residual degrees of freedom!

Coefficients: (1 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)
(Intercept)        1         NA      NA       NA
x                 NA         NA      NA       NA

Residual standard error: NaN on 0 degrees of freedom
```

N = 20

P = 50

```
Call:
lm(formula = y ~ ., data = hidim_data)

Residuals:
ALL 20 residuals are 0: no residual degrees of freedom!

Coefficients: (31 not defined because of singularities)
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   17.278        NaN     NaN      NaN
X1             4.284        NaN     NaN      NaN
X2            17.150        NaN     NaN      NaN
X3            -4.112        NaN     NaN      NaN
X4            12.518        NaN     NaN      NaN
X5            -7.034        NaN     NaN      NaN
X6            -7.189        NaN     NaN      NaN
X7            12.484        NaN     NaN      NaN
X8           -10.152        NaN     NaN      NaN
X9             1.135        NaN     NaN      NaN
X10           26.725        NaN     NaN      NaN
X11            8.157        NaN     NaN      NaN
X12          -18.407        NaN     NaN      NaN
X13           19.460        NaN     NaN      NaN
X14           11.567        NaN     NaN      NaN
X15           -4.300        NaN     NaN      NaN
X16          -11.200        NaN     NaN      NaN
X17          -11.089        NaN     NaN      NaN
X18            1.665        NaN     NaN      NaN
X19          -16.621        NaN     NaN      NaN
X20               NA         NA      NA       NA
X21               NA         NA      NA       NA
X22               NA         NA      NA       NA
X23               NA         NA      NA       NA
X24               NA         NA      NA       NA
X25               NA         NA      NA       NA
X26               NA         NA      NA       NA
X27               NA         NA      NA       NA
X28               NA         NA      NA       NA
X29               NA         NA      NA       NA
X30               NA         NA      NA       NA
```

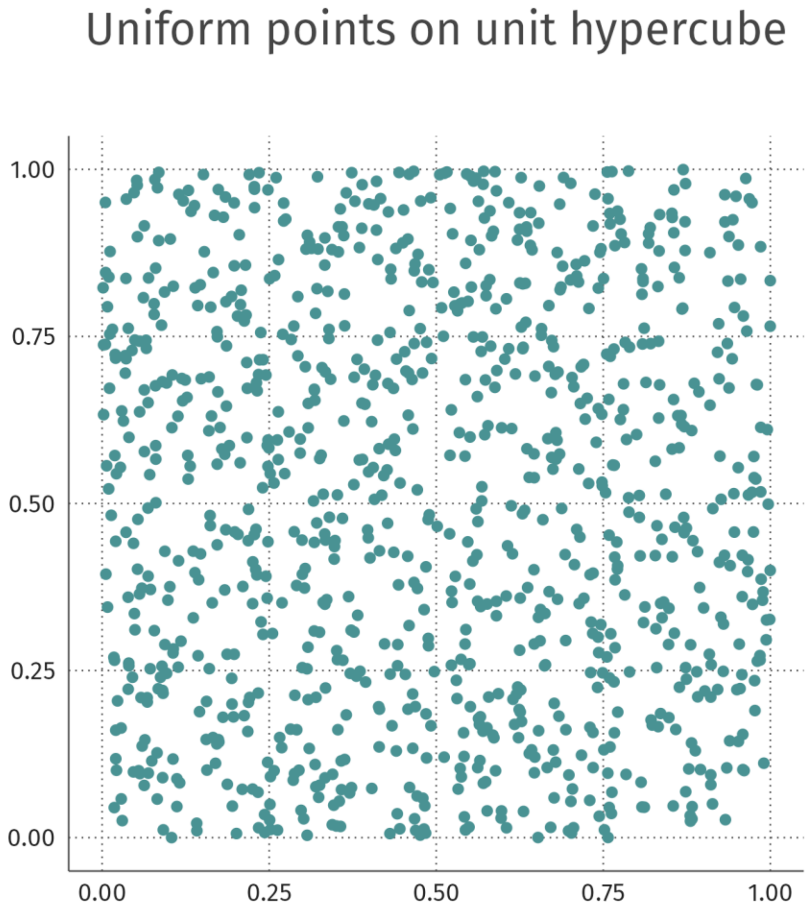
The curse of dimensionality

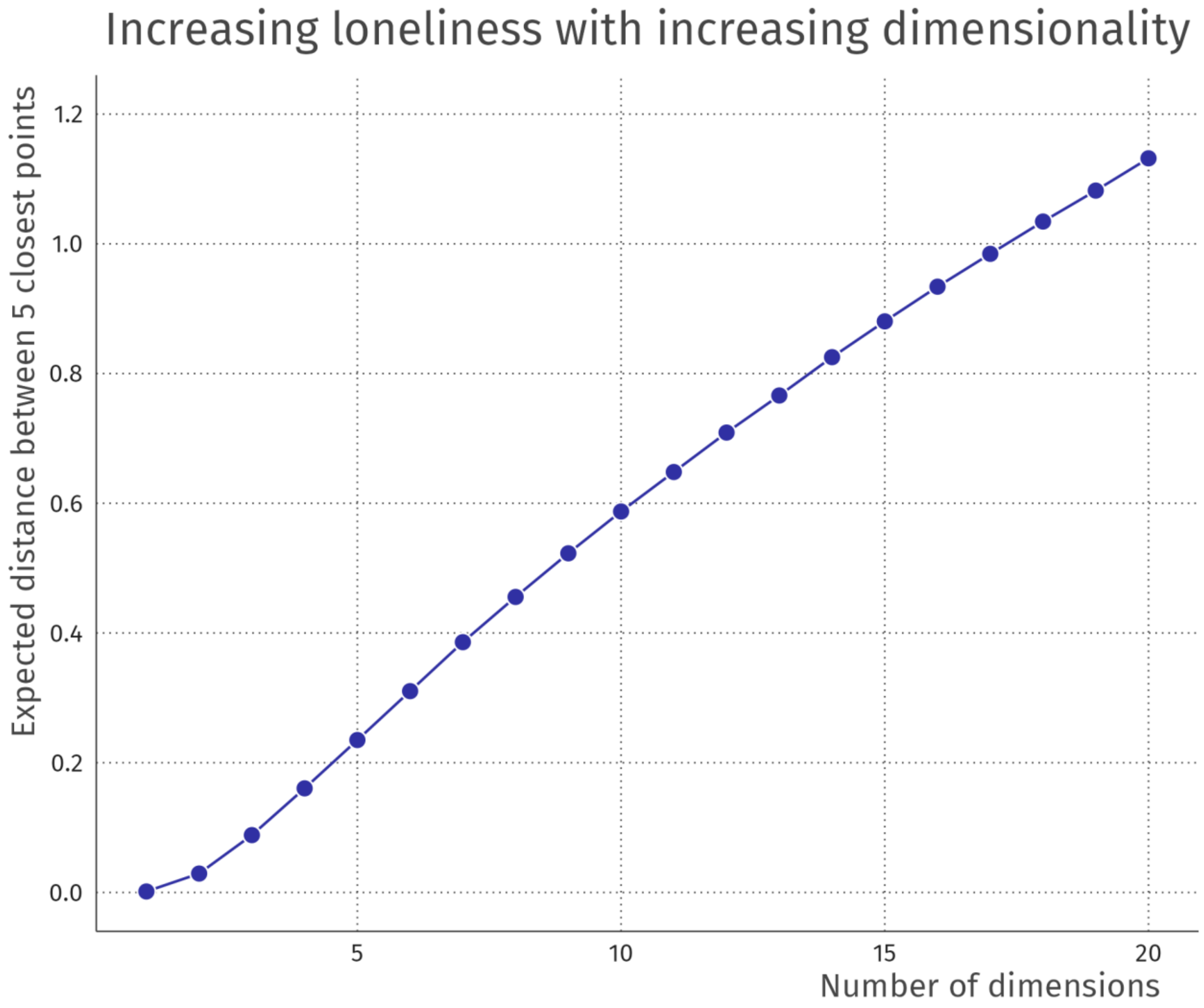
Bernhard Schölkopf
@bschoelkopf

a high-dimensional space is a lonely place

4:50 PM · Aug 24, 2014 · Twitter Web Client

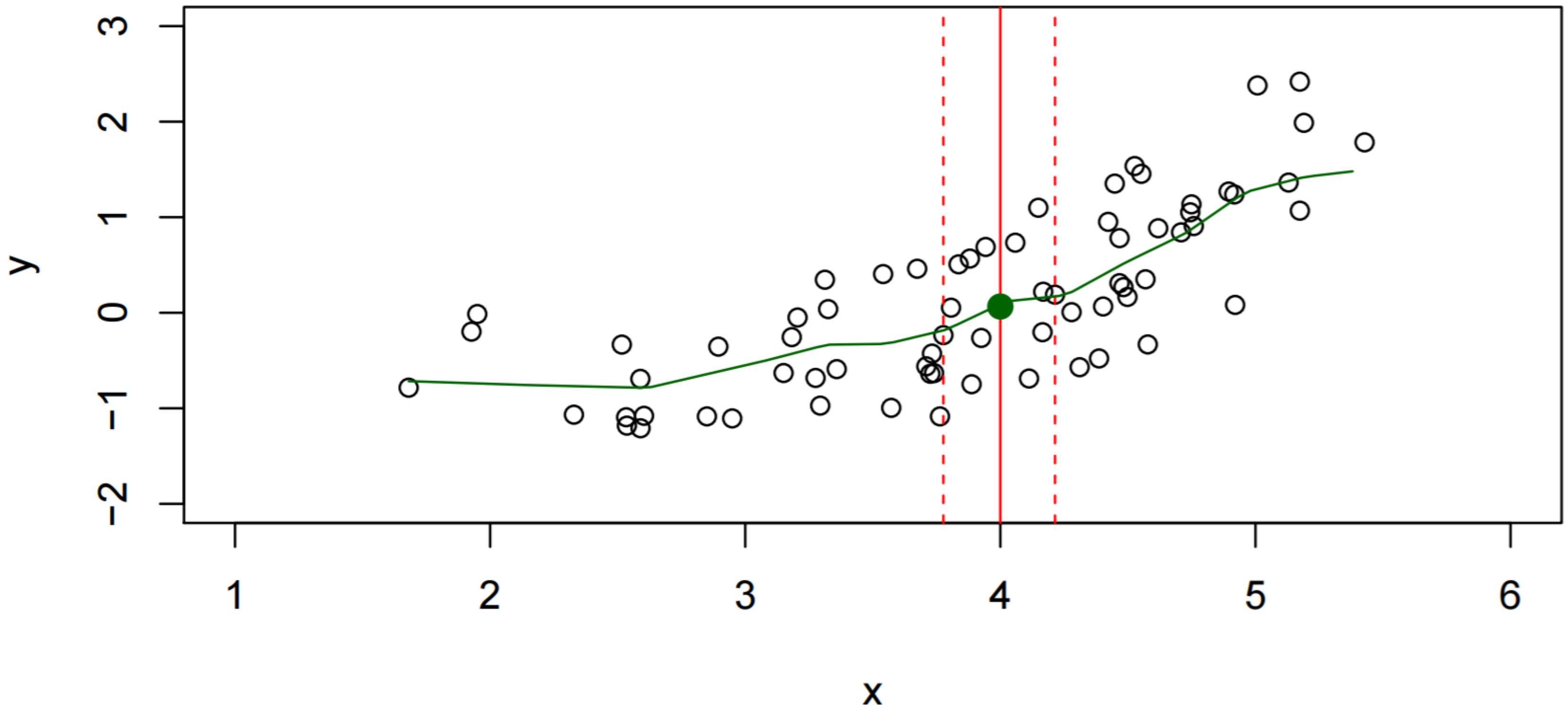
47 Retweets **4** Quote Tweets **122** Likes


Uniform points on unit hypercube
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00


Increasing loneliness with increasing dimensionality
Expected distance between 5 closest points
Number of dimensions
0.0
0.2
0.4
0.6
0.8
1.0
1.2
5
10
15
20

The curse of dimensionality

K-nearest neighbours regression



The curse of dimensionality

K-nearest neighbours regression

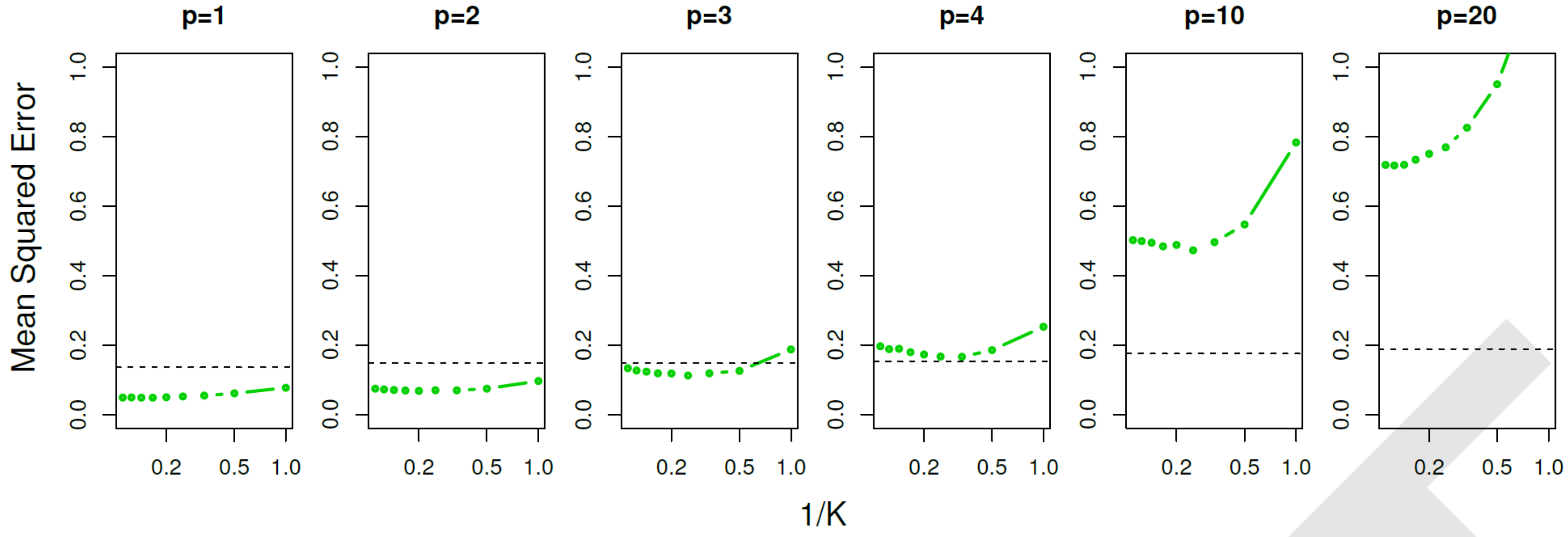
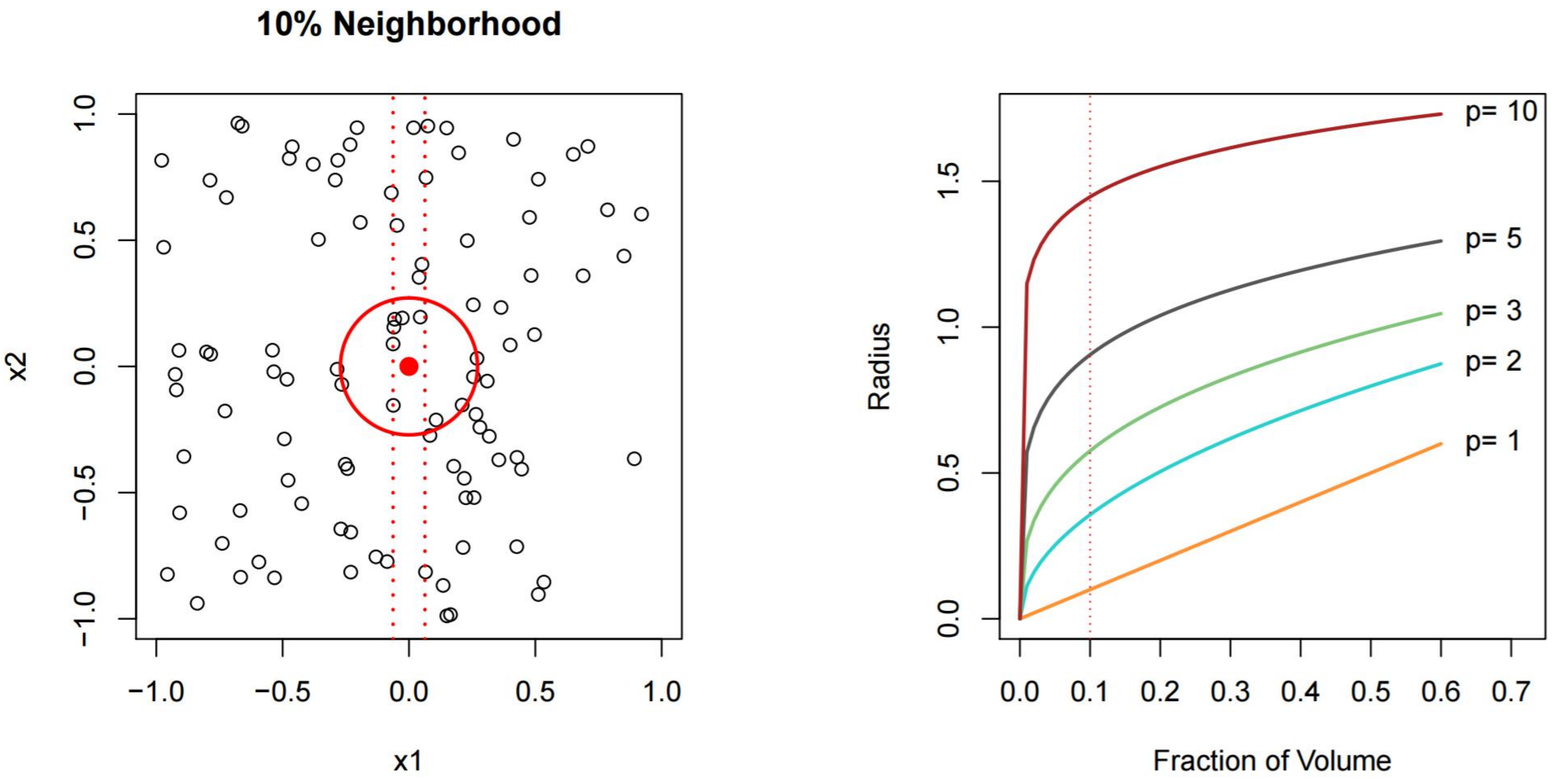


FIGURE 3.20. *Test MSE for linear regression (black dashed lines) and KNN*

ISLR2, p. 110

The curse of dimensionality

What is “nearest”?



ISLR

Break

What can we do?

Battling the curse of dimensionality

- **Feature selection**
  Remove some features

- **Penalization / regularization / shrinkage**
  Constrain model parameters, possibly setting some to 0

- **Dimension reduction (next 2 weeks!)**
  Summarize P features in Q < P features

Feature selection

Filter

Select features based on some (usually univariate) criterion

- *Variance filter*: drop features with (very) low variance from the dataset
- *Correlation filter*: drop features with low correlation with outcome from dataset
- What other filter methods can you come up with?

Wrapper

Fit models with different # of features, select best model

- Forward / backward selection
- What is “best”? P-value? AIC/BIC? Out-of-sample MSE?
- Computationally expensive

Battling the curse of dimensionality

- **Feature selection**
  Remove some features

- **Penalization / regularization / shrinkage**
  Constrain model parameters, possibly setting some to 0

- **Dimension reduction (next 2 weeks!)**
  Summarize P features in Q < P features

Penalization / regularization

Ridge regression penalty:

$$\lambda \cdot \sum_{p=1}^{P} \beta_p^2 = \lambda \cdot \|\beta\|_2^2$$

LASSO regression penalty:

$$\lambda \cdot \sum_{p=1}^{P} |\beta_p| = \lambda \cdot \|\beta\|_1$$

Penalization / regularization

Penalties put constraints on the parameter space

With ridge, the Euclidian (L_2) distance from 0 is constrained

With LASSO, the Manhattan distance (L_1) from 0 is constrained.

Constraints on parameters introduce **bias** (ests. different from ML ests.) but reduce **variance**.

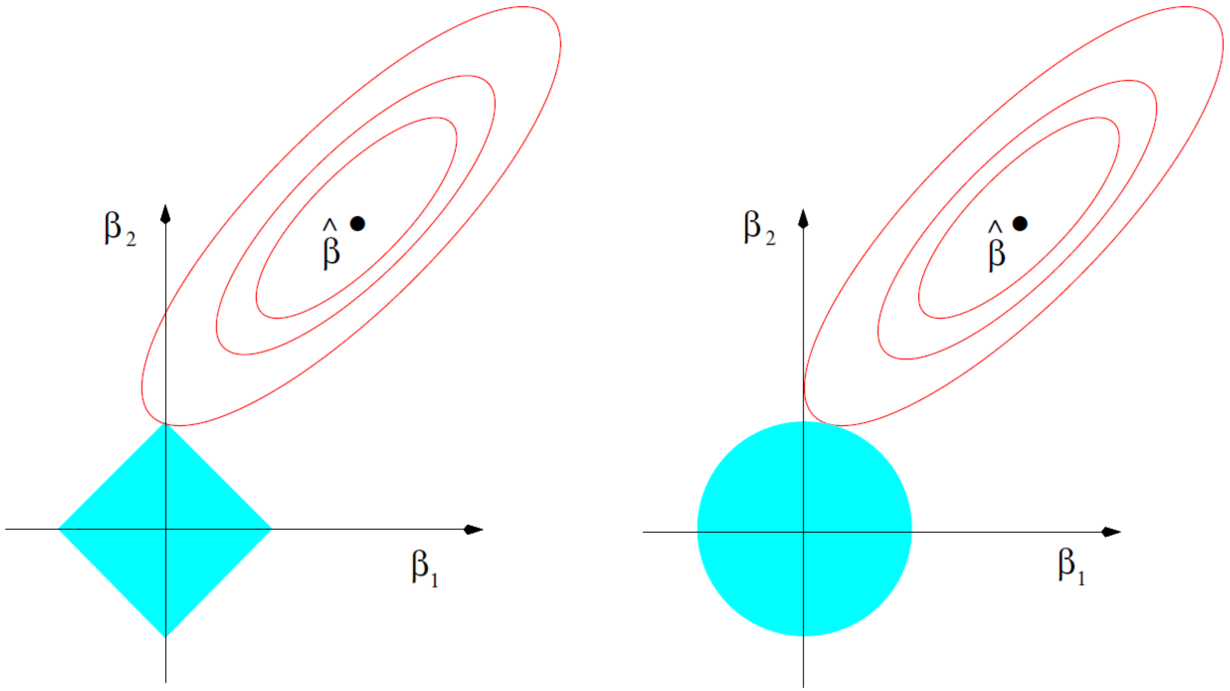


Figure 2.2 *Estimation picture for the lasso (left) and ridge regression (right). The solid blue areas are the constraint regions* $|\beta_1|+|\beta_2| \le t$ *and* $\beta_1^2+\beta_2^2 \le t^2$*, respectively, while the red ellipses are the contours of the residual-sum-of-squares function. The point* $\widehat{\beta}$ *depicts the usual (unconstrained) least-squares estimate.*

SLS, p. 11

Bet on sparsity

Assume underlying process is sparse.

With LASSO we can recover it (“oracle property”)

What if our assumption is false (i.e., process is not sparse)?

Then no method can do well anyway

Then LASSO is not necessarily worse than other methods

Let's try it

Battling the curse of dimensionality with $p = 2$ and $n = 1$

Ridge regression

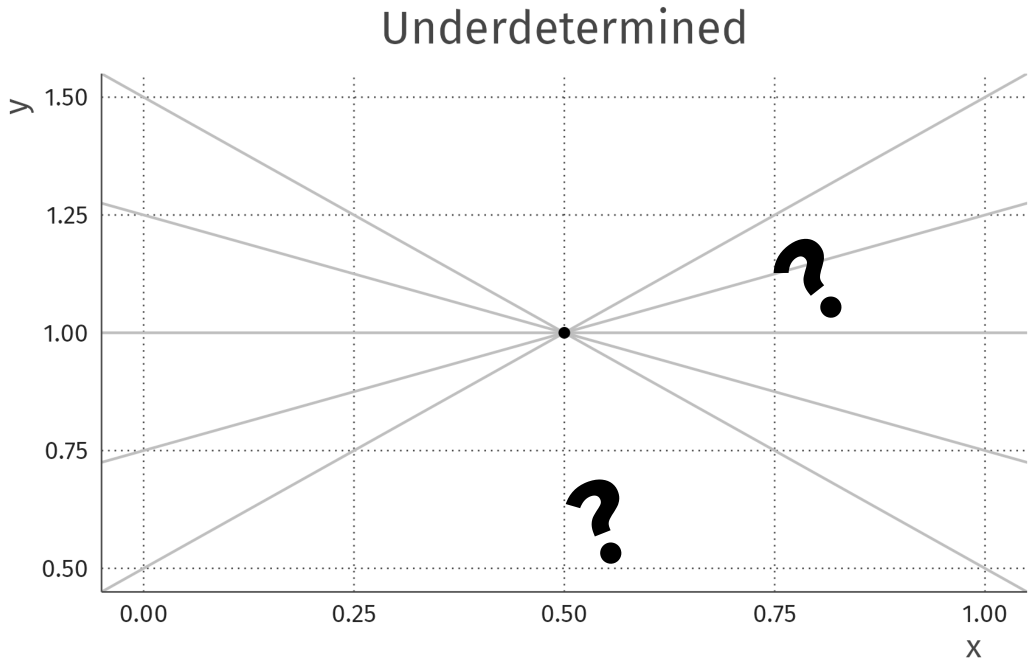
Loss function

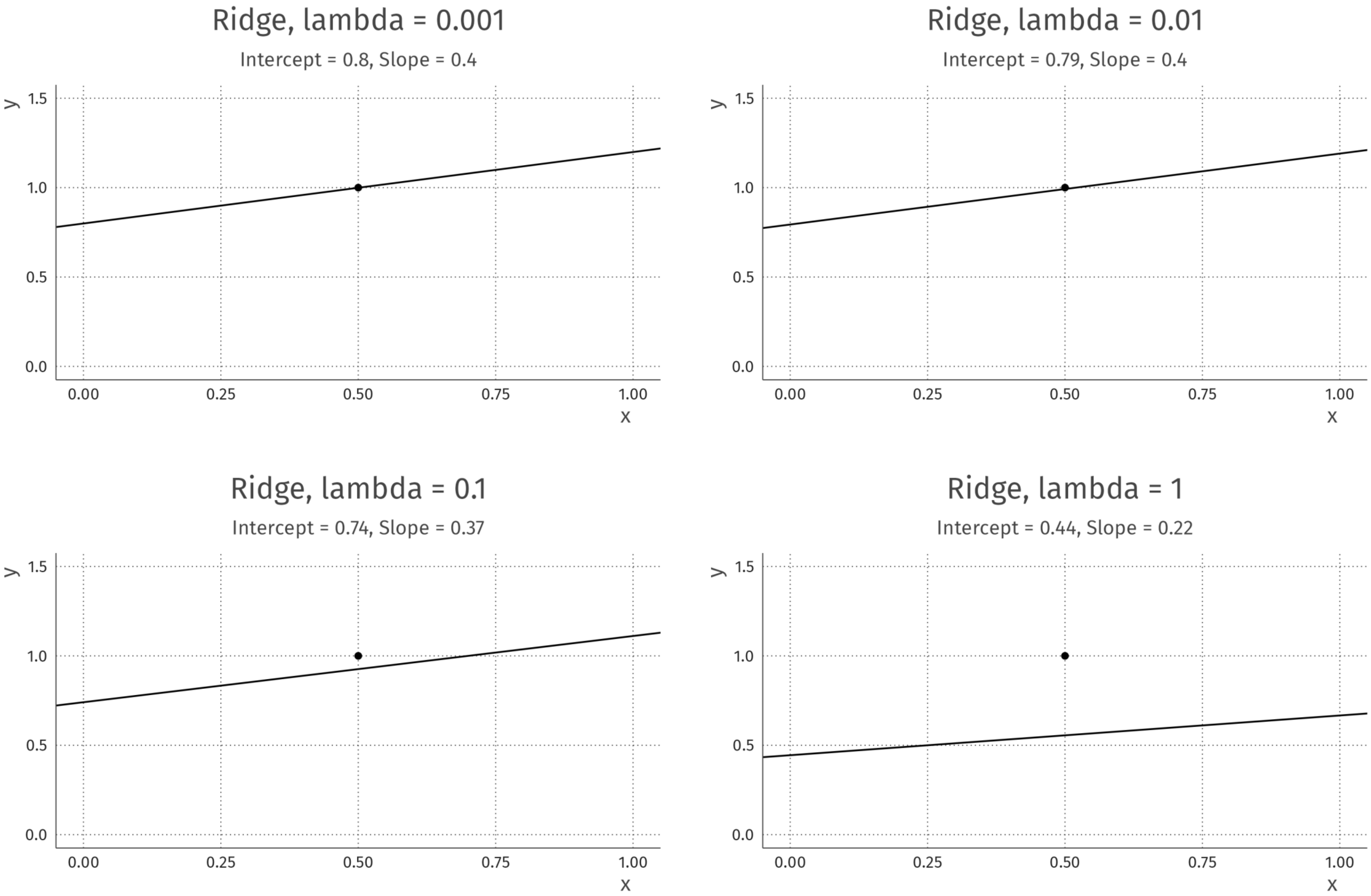
OLS + ridge penalty

Formally, precisely:

$$f(\beta_0, \beta_1) = (y - \beta_0 - \beta_1 x)^2 + \lambda \cdot \sum_{p=1}^{P} \beta_p^2$$

Minimize loss for different values of λ




Ridge, lambda = 0.001
Intercept = 0.8, Slope = 0.4
Ridge, lambda = 0.01
Intercept = 0.79, Slope = 0.4
Ridge, lambda = 0.1
Intercept = 0.74, Slope = 0.37
Ridge, lambda = 1
Intercept = 0.44, Slope = 0.22
y
1.5
1.0
0.5
0.0
0.00
0.25
0.50
0.75
1.00
x

LASSO regression

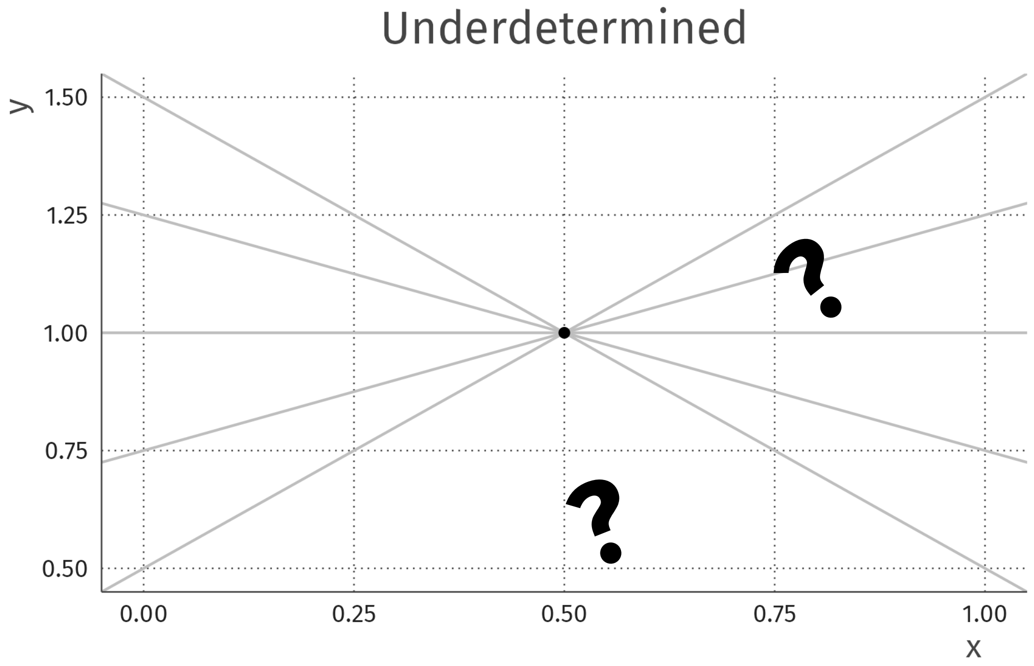
Loss function

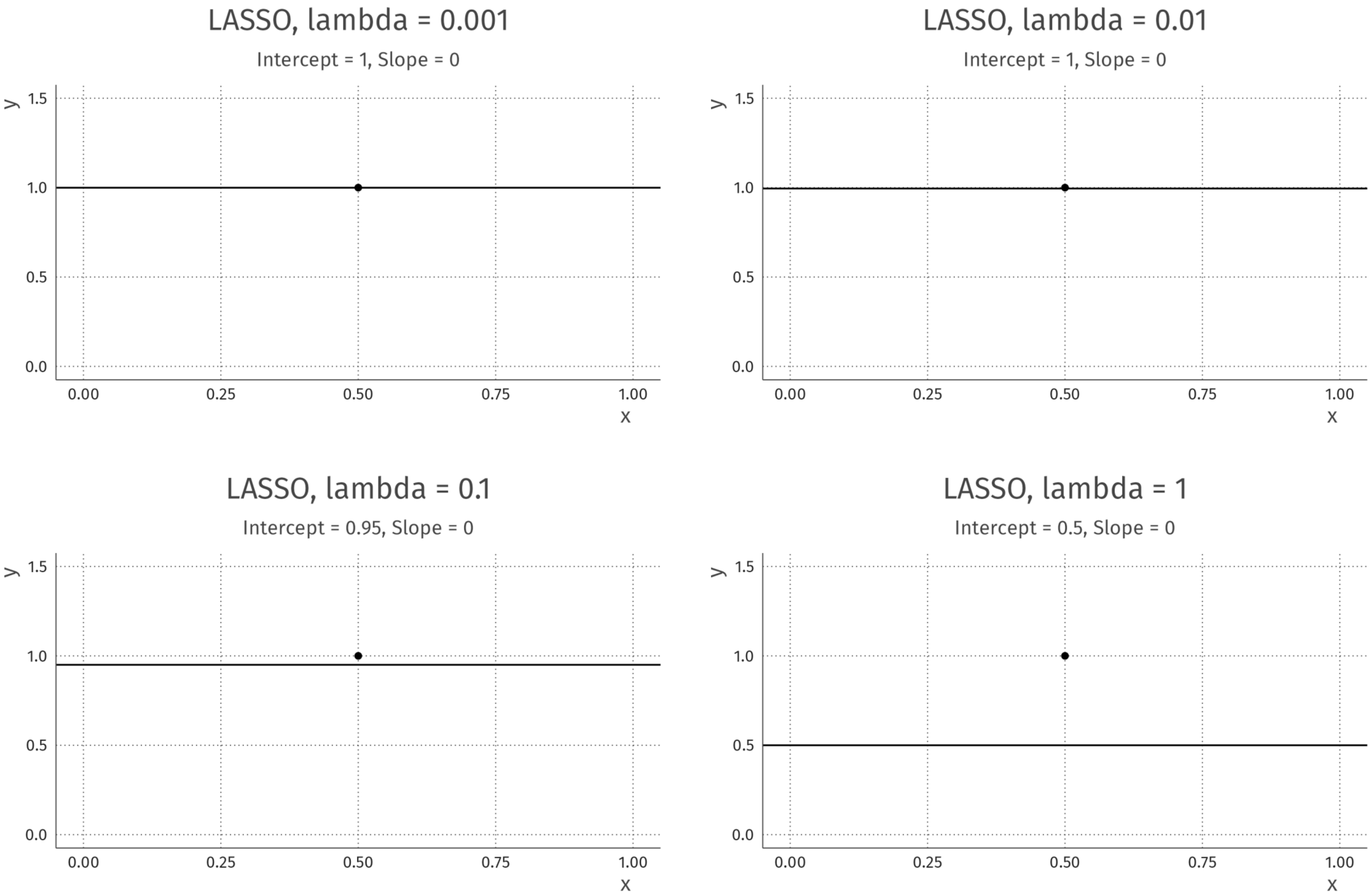
OLS + LASSO penalty

Formally, precisely:

$$f(\beta_0, \beta_1) = (y - \beta_0 - \beta_1 x)^2 + \lambda \cdot \sum_{p=1}^{P} |\beta_p|$$

Minimize loss for different values of λ




LASSO, lambda = 0.001
Intercept = 1, Slope = 0
LASSO, lambda = 0.01
Intercept = 1, Slope = 0
LASSO, lambda = 0.1
Intercept = 0.95, Slope = 0
LASSO, lambda = 1
Intercept = 0.5, Slope = 0
y
x
0.0
0.5
1.0
1.5
0.00
0.25
0.50
0.75
1.00

New problem: choosing lambda

- Lambda is a hyperparameter
- We have to choose it in some way

Common approach: predictive accuracy

- Choose lambda to minimize MSE on unseen data
- We don't *know* out-of-sample MSE, so estimate it
- Cross-validation, train-validation split, LOOCV, BIC, AIC
- Model selection problem!

Technical Bayesian side note

In Bayesian data analysis, we put priors on parameters

- a priori we state that e.g., $\beta_0 \sim N(0, \sigma^2), \beta_1 \sim N(0, \sigma^2)$
- then we use data to update this belief about our parameter
- without data proving otherwise, $\mathrm{E}[\beta_0] = 0, \mathrm{E}[\beta_1] = 0$

This is shrinkage / regularization!

- Priors put constraints on parameter space
- Normal distribution prior ≈ Ridge regression
- Laplace distribution prior ≈ LASSO regression
- σ^2 is the hyperparameter (similar to λ^{-1})

Downside: computationally more expensive

LASSO generalizations

LASSO generalization: elastic net

Elastic net loss function:

$$\frac{1}{2}\sum_{n=1}^{N}\left(y_n - \beta_0 - x_n^T\beta\right)^2 + \lambda\left[\frac{1}{2}(1-\alpha)||\beta||_2^2 + \alpha||\beta||_1\right]$$

- Combination of LASSO and ridge penalty
- Encourages parameter sharing with correlated vars
- Additional hyperparameter: α

LASSO generalization: elastic net

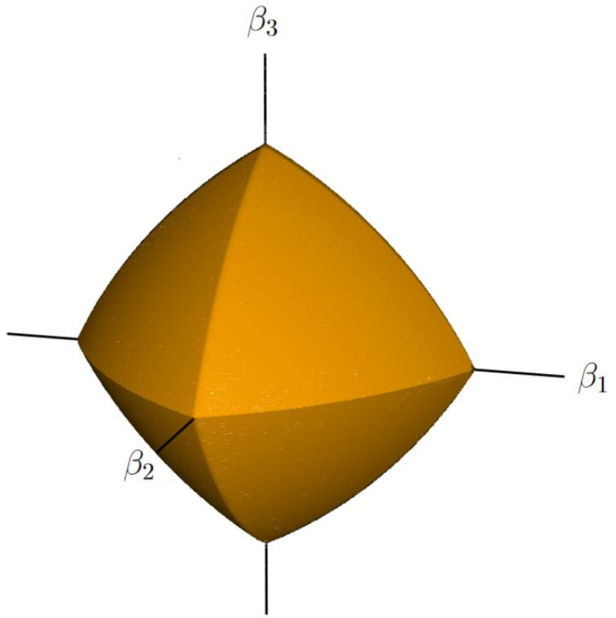


Figure 4.2 *The elastic-net ball with* $\alpha = 0.7$ *(left panel) in* $\mathbb{R}^3$*, compared to the* ℓ_1 *ball (right panel). The curved contours encourage strongly correlated variables to share coefficients (see Exercise 4.2 for details).*

LASSO generalization: group LASSO

Group LASSO loss function:

- Put variables Z & coefficients θ in groups $j = 1, \dots, J$

$$\frac{1}{2}\sum_{n=1}^{N}\left(y_n - \theta_0 - \sum_{j=1}^{J} z_{nj}^{T}\theta_j\right)^2 + \lambda \sum_{j=1}^{J} \|\theta_j\|_2$$

- When we know certain parameters belong together
- Think about dummy coding categorical variables

LASSO generalization: group LASSO

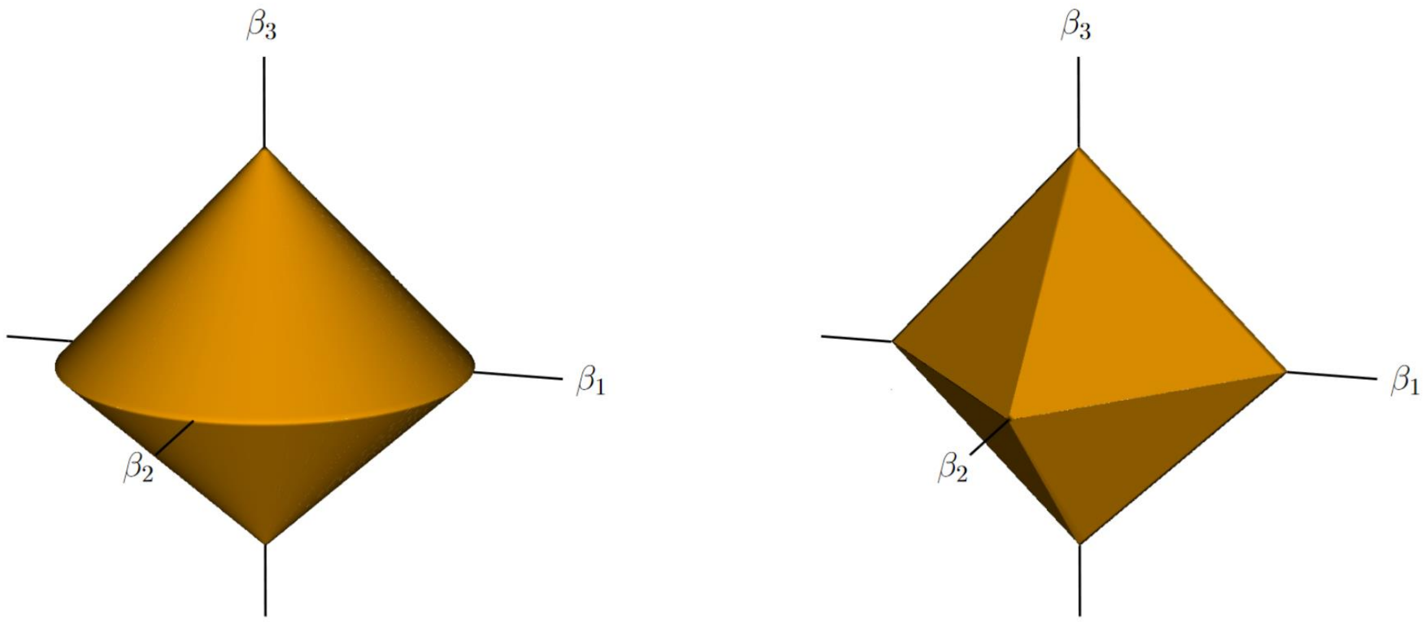


Figure 4.3 *The group lasso ball (left panel) in $\mathbb{R}^3$, compared to the ℓ_1 ball (right panel). In this case, there are two groups with coefficients $\theta_1 = (\beta_1, \beta_2) \in \mathbb{R}^2$ and $\theta_2 = \beta_3 \in \mathbb{R}^1$.*

Implementation: glmnet

Implementation: glmnet

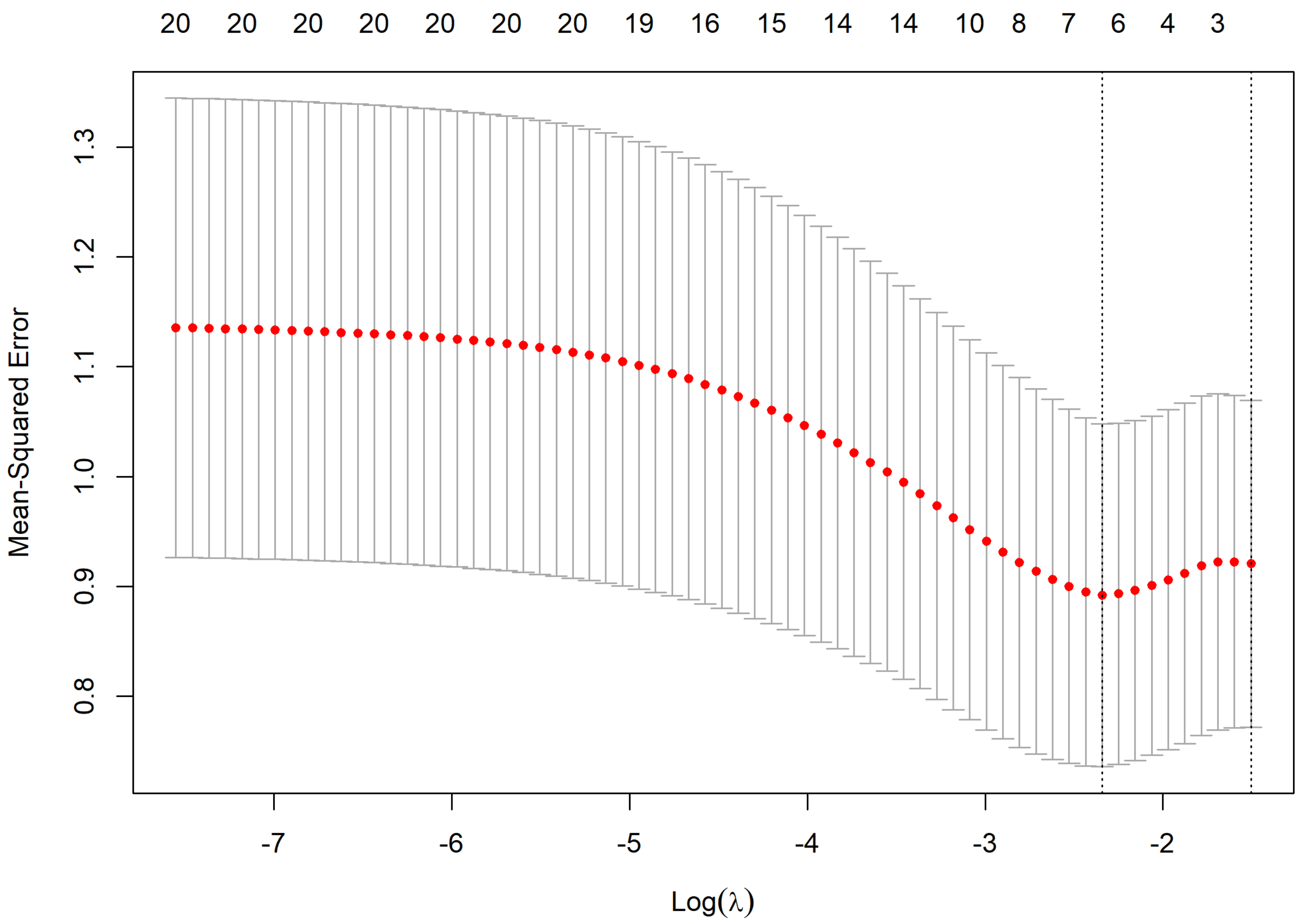
- glmnet can perform LASSO, ridge regression for several types of outcomes
  - Linear regression
  - Logistic regression
  - Poisson (count) regression
  - Multinomial logistic regression
  - …
- It can also perform elastic net regression
- It has cross-validation built-in to select lambda

Implementation: glmnet

```r
library(glmnet)

# generate some fake data matrices
x <- matrix(rnorm(100 * 20), nrow = 100, ncol = 20)
y <- rnorm(100)

# estimate LASSO regression model
fit <- glmnet(x = x, y = y, lambda = 0.1, alpha = 1)

# generate predictions
predict(fit, newx = x)
```

Implementation: glmnet

```
# automatically estimate the lambda parameter
fit_cv <- cv.glmnet(x = x, y = y, alpha = 1)

# generate predictions from the best model
predict(fit_cv, newx = x, s = "lambda.min")

# plot the lambdas vs mse
plot(fit_cv)
```


20 20 20 20 20 20 20 19 16 15 14 14 10 8 7 6 4 3
Mean-Squared Error
0.8
0.9
1.0
1.1
1.2
1.3
-7
-6
-5
-4
-3
-2
Log(λ)

First practical: perform penalized regression with glmnet

Real high-dimensional gene expression dataset for cancer prediction

Practical

- Look at infomda2.nl practical 1
- Do exercises **1-5** before Friday
- In class on Friday: discussing exercises 1-5
- Making the remaining assignments

Have a nice day!