Seminar Data Science for Economics

MSc. Economics program

Madina Kurmangaliyeva
m.kurmangaliyeva@uvt.nl

Spring 2020

Tilburg University

Recap/Intro

So far you have learnt

Econometrics:

- OLS
- Method of Moments
- Maximum Likelihood
- IV
- RDD
- Diff-in-Diff
- ?

Machine learning:

- Linear Shrinkage regressions: Lasso, Ridge, Elastic Net
- Neural network
- ?

Core material

This part of the course is mostly based on the following material:

- Textbooks:
  - *Introduction to Statistical Learning* by Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani. Free e-book
  - *Mostly Harmless Econometrics* by Joshua Angrist and Jörn-Steffen Pischke.
- Papers:
  - *High-dimensional methods and inference on structural and treatment effects* by Alexandre Belloni, Victor Chernozhukov, and Christian Hansen (JEP, 2014). Link to paper
  - *Double/debiased machine learning for treatment and structural parameters* by Victor Chernozhukov, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, and Whitney Newey (Econom. J., 2019). Link to paper
  - *Recursive partitioning for heterogeneous causal effects* by Susan Athey and Guido Imbens (PNAS, 2016). Link to paper

Data science versus econometrics (Question)

1. What is the difference between data science questions and empricial econ questions?

Simple example: estimation of μ[1]

Data generating process:

$$y_i = \mu + \epsilon_i \tag{1}$$

Suppose we want to estimate μ:

$$\begin{aligned} \hat{\mu} &= \alpha \frac{1}{n} \sum_{i=1}^{n} y_i \\ &= \alpha \bar{y} \end{aligned} \tag{2}$$

What should be α to have an unbiased estimator of μ?

$$E(\alpha \bar{y}) = \mu \tag{3}$$

The best estimator is: $\alpha = 1 \implies \hat{\mu} = \bar{y}$

[1] This illustrative example comes from the slides of Susan Athey

Simple example: Predictor for y_i

The same setup.

Data generating process:

$$y_i = \mu + \epsilon_i \tag{4}$$

But now we want to predict y by minimizing expected mean squared error of our prediction.

Predictor for y_i:

$$\hat{y} = \alpha \bar{y} \tag{5}$$

$$\min_{\alpha} E(y_i - \alpha \bar{y})^2 \tag{6}$$

What should be α?

$$\alpha = \frac{\mu^2}{\mu^2 + \frac{1}{n} var(\epsilon)} < 1 \implies \text{Shrinking towards zero!} \tag{7}$$

Simple example: Proof

The best predictor minimizes the expected MSE:

$$E\left[\left(\alpha\bar{y} - y_i\right)^2\right] = \tag{8}$$

$$E\left[\alpha^2\bar{y}^2 - 2\alpha\bar{y}y_i + y_i^2\right] = \tag{9}$$

$$E\left[\alpha^2\left(\mu + \frac{1}{n}\sum_{j=1}^{n}\epsilon_j\right)^2 - 2\alpha\left(\mu + \frac{1}{n}\sum_{j=1}^{n}\epsilon_j\right)(\mu + \epsilon_i) + (\mu + \epsilon_i)^2\right] = \tag{10}$$

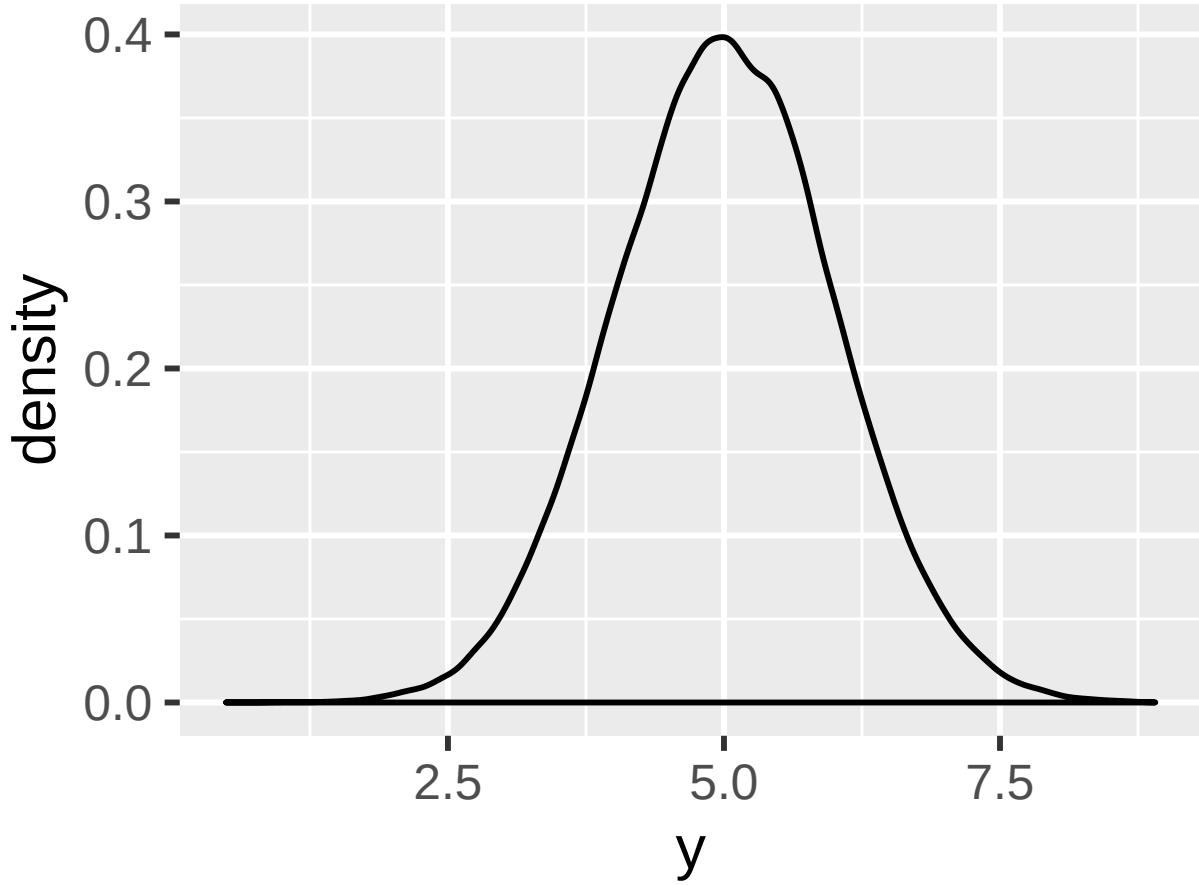
$$\alpha^2\mu^2 + \alpha^2\frac{1}{n}var(\epsilon) - 2\alpha\mu^2 + \mu^2 + var(\epsilon) \tag{11}$$

Take derivative with respect to alpha:

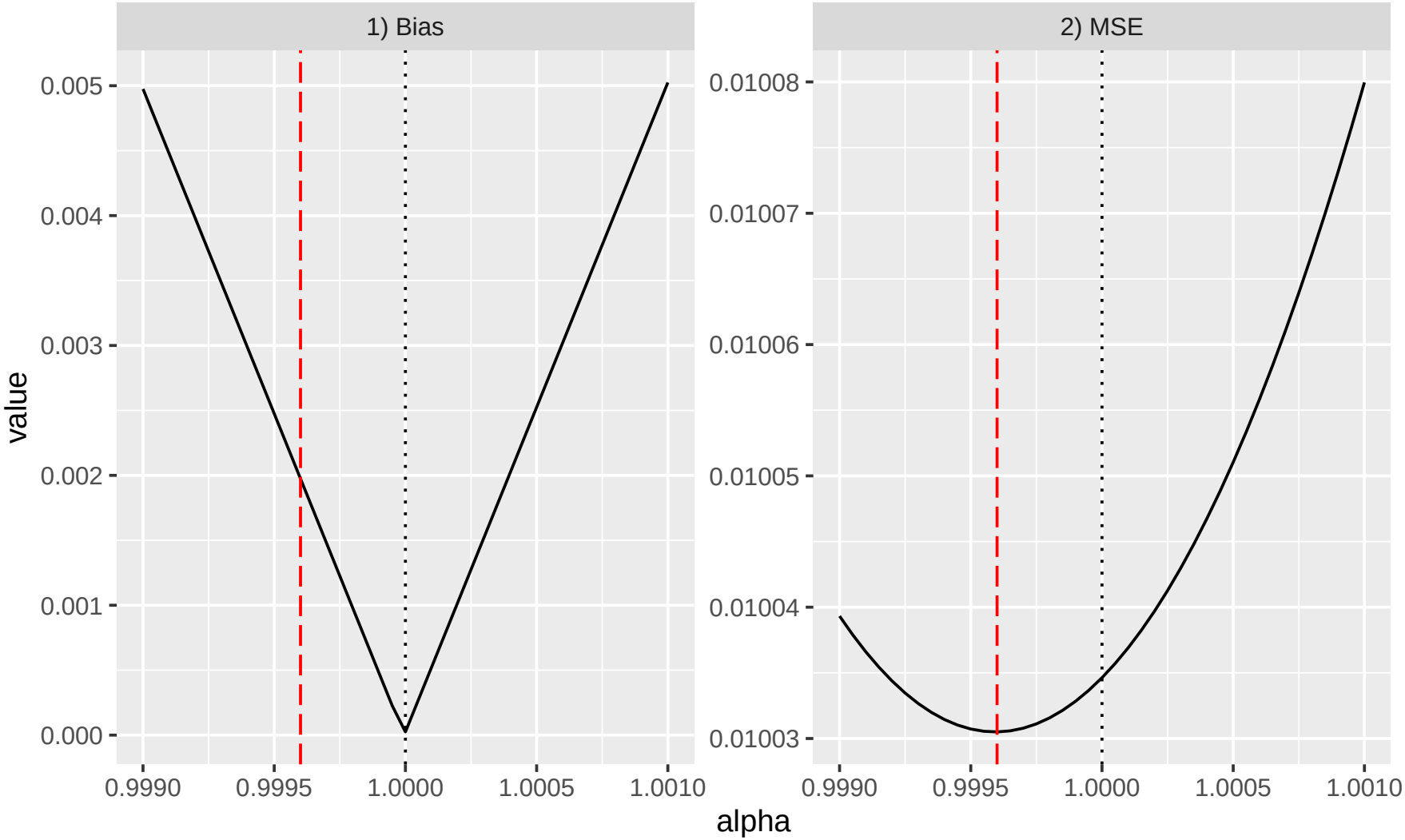
$$-2\alpha\mu^2 + 2\mu^2 - 2\alpha\frac{1}{n}var(\epsilon) = 0 \Longrightarrow \alpha^* = \frac{\mu^2}{\mu^2 + \frac{1}{n}var(\epsilon)} \tag{12}$$

Simple example: Distribution of y when $\mu = 5$

$y_i = 5 + \epsilon_i$, $\epsilon \sim \mathcal{N}(0, 1)$, and $n = 100 \Longrightarrow \alpha^* \approx 0.9996$



Simple example: Bias vs MSE



Red line at $\alpha = 0.9996$, black dotted line at $\alpha = 1$. See the code `simulation_prediction_vs_estimation.R`

Bottomline

With this relatively simple setting, we can see that:

$$\text{Inference} \neq \text{Prediction}$$

Prediction tasks are about bias-variance trade-off

Exercise: Inference or Prediction?

Is the following a prediction or an inference task?

- Identify poor households eligible for welfare program based on the house characteristics (#rooms, sqm, neighborhood) and observable durable goods (tv, fridge, toilet)
- Find which advertisement campaign is the most effective for boosting product sales
- Find the price elasticity of demand for a product
- Measure GDP growth from sattelite night luminosity images
- Find whether uber is helping or hurting of the long-term employment prospects of the poor.
- Does infant daycare affect later-life IQ of children?

The rest of the course

In the last half of the course we will study the following techniques.

Prediction:

- Theory behind cross-validation
- Regularization: Lasso vs Ridge
- Decision trees, random forest, bagging, boosting

Prediction for causal inference:

- Post-regularization causal inference
- Double Machine Learning
- Causal trees

Working with data, a replication study

Prediction

Data generating process

Ingredients and recipes:

- X:
  - X is an $n \times p$ matrix of covariates
  - X is a random **sample** from the **population of** X
- Y:
  - Y is an n-length vector of outcomes corresponding to X
  - Y is also a random **sample** from the **population of** Y
- The **population of** X and the **population of** Y are related through the following data generating process:

$$Y = \underbrace{f(X)}_{\text{CEF}} + \underbrace{\varepsilon}_{E(\varepsilon|X)=0} \tag{13}$$

Notation: CEF = Conditional Expectation Function

Examples of data generating processes

$f(X)$ means "some" function of X. It can be:

- Continuous additive: $Y = \beta_1 X_1 + ... + \beta_k X_p$
- Continuous non-linear: $Y = \beta_0 + \frac{\exp\{X_2 * ... * X_p\}}{X_1}$.

Hypothetical example

A UvT student spends on average €400 per month on food. The spending is distributed normally with €80 standard deviation.

$$X \sim N(400, 80)$$

Now, suppose that a student that spends €X on food, also spends on average $0.5X$ per month on entertainment with variance €50. And the conditional distribution of spending on entertainment is also normal.

The data generating process for any sample of X and Y is:

- $Y = 0.5X + \epsilon$
- $\epsilon \sim N(0, 50)$
- $X \sim N(400, 80)$

Prediction

Let's get a bit mathy.

Given the sample X^T and Y^T provide prediction of Y for any new sample X?

Prediction: $\hat{Y} = \hat{f}(X) \mid \underbrace{X^T, Y^T}_{\text{training data}}$

- It means that we trained $\hat{f}(.)$ using a particular sample realization X^T, Y^T.
- We are not concerned with the exact form of $\hat{f}$.
- We are searching for $\hat{f}$ that minimizes Mean Squared Error in expectation for any new realization of X.

Expected Mean Squared Error

Error:

$$Y - \hat{Y} = \underbrace{f(X) - \hat{f}(X)}_{\text{reducible error}} + \underbrace{\varepsilon}_{\text{irreducible error}} \tag{14}$$

Expected MSE:

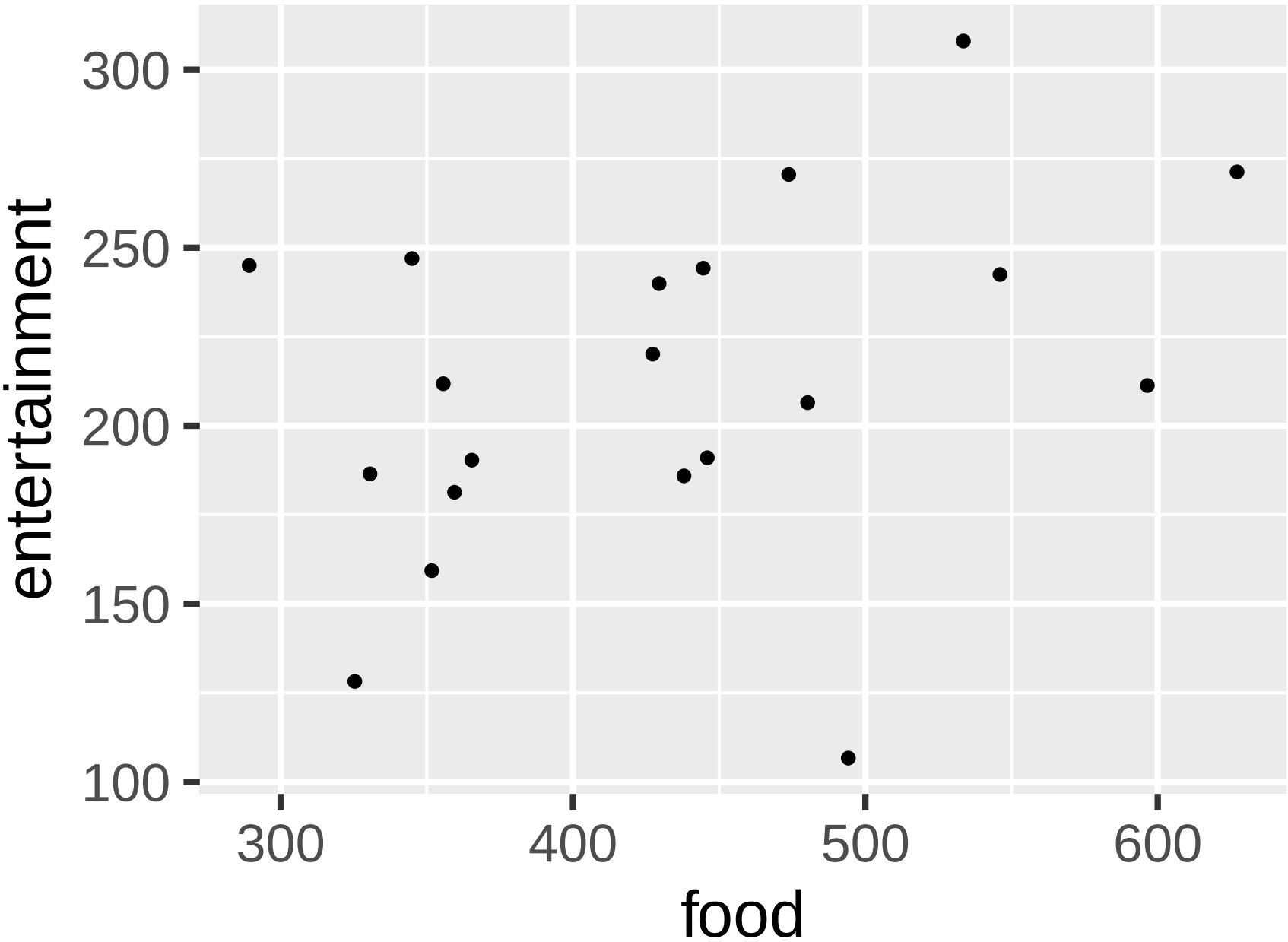
$$E_Y[(Y - \hat{Y})^2 | X, X^T] = \underbrace{E\left[\left(f(X) - \hat{f}(X)\right)^2\right]}_{\text{reducible}} + \underbrace{var(\varepsilon)}_{\text{irreducible}} \tag{15}$$

Simulate

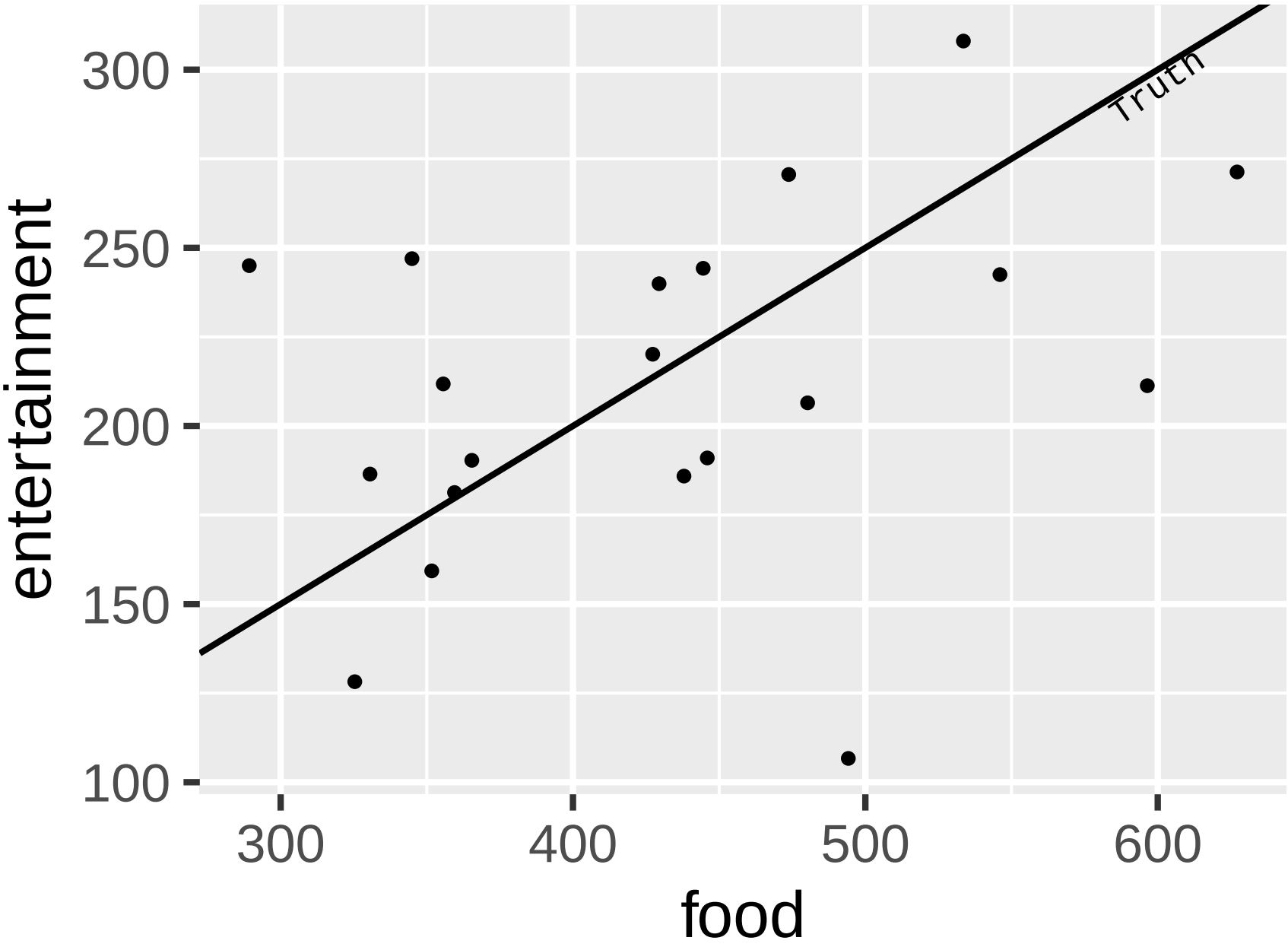
```
set.seed(911)
train <- tibble(food = rnorm(n = 20, 400, 80)) %>%
          mutate(entertainment = 0.5*food +
                   rnorm(n = 20, 0, 50))

# Plot the training data
train %>% ggplot(aes(x = food, y = entertainment)) +
  geom_point()
```

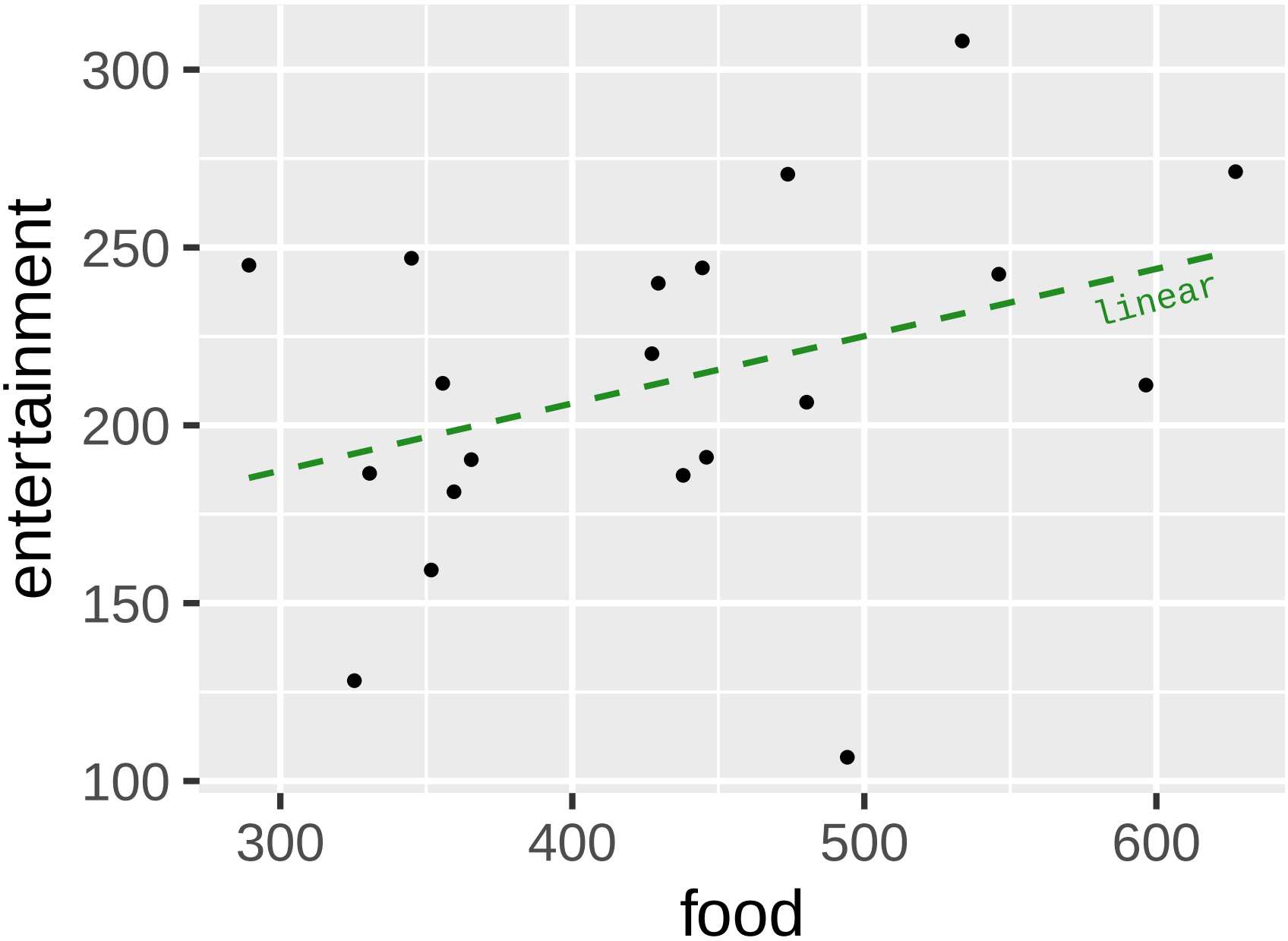
Training data, 20 observations



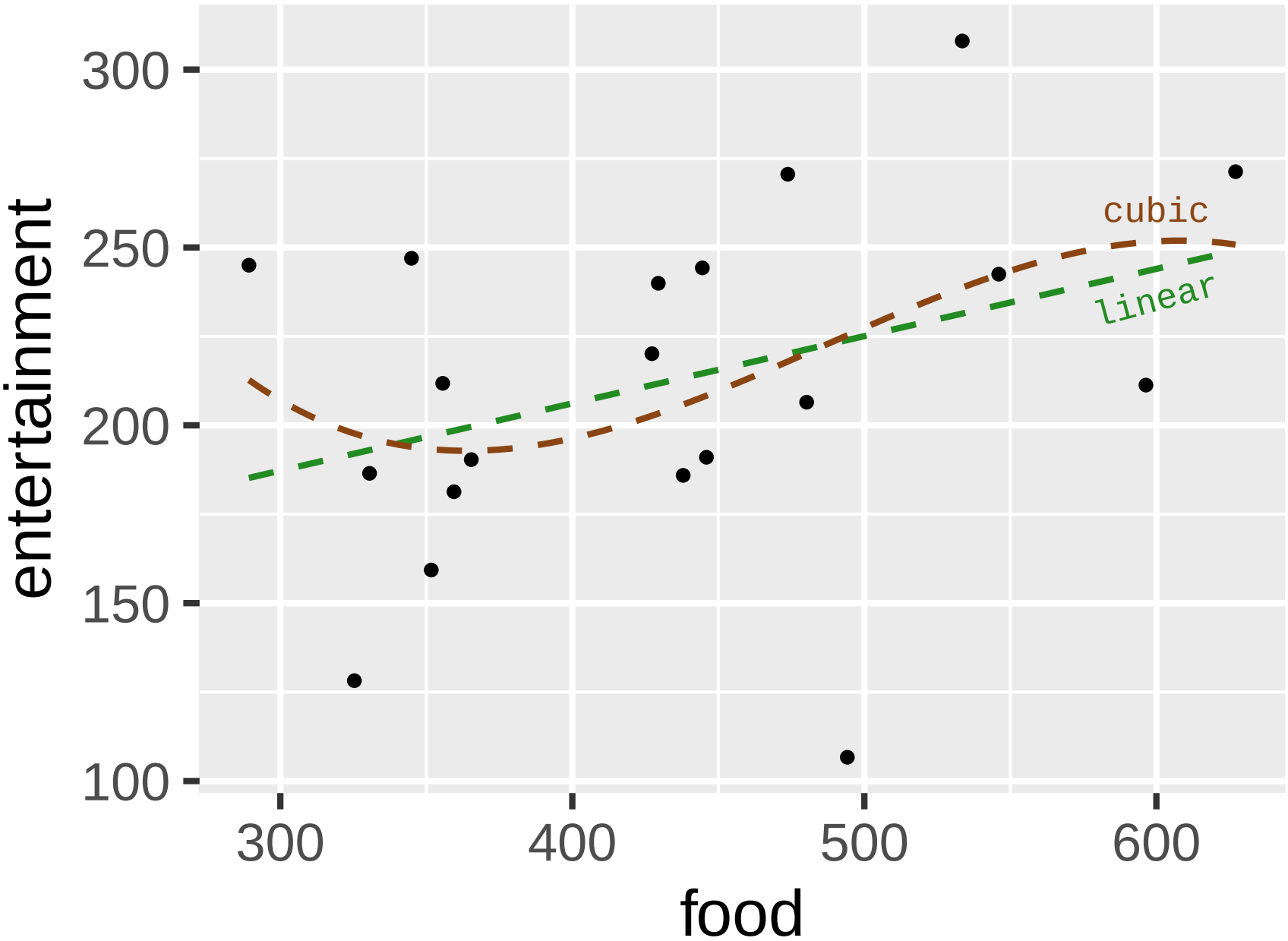
the true data generating process



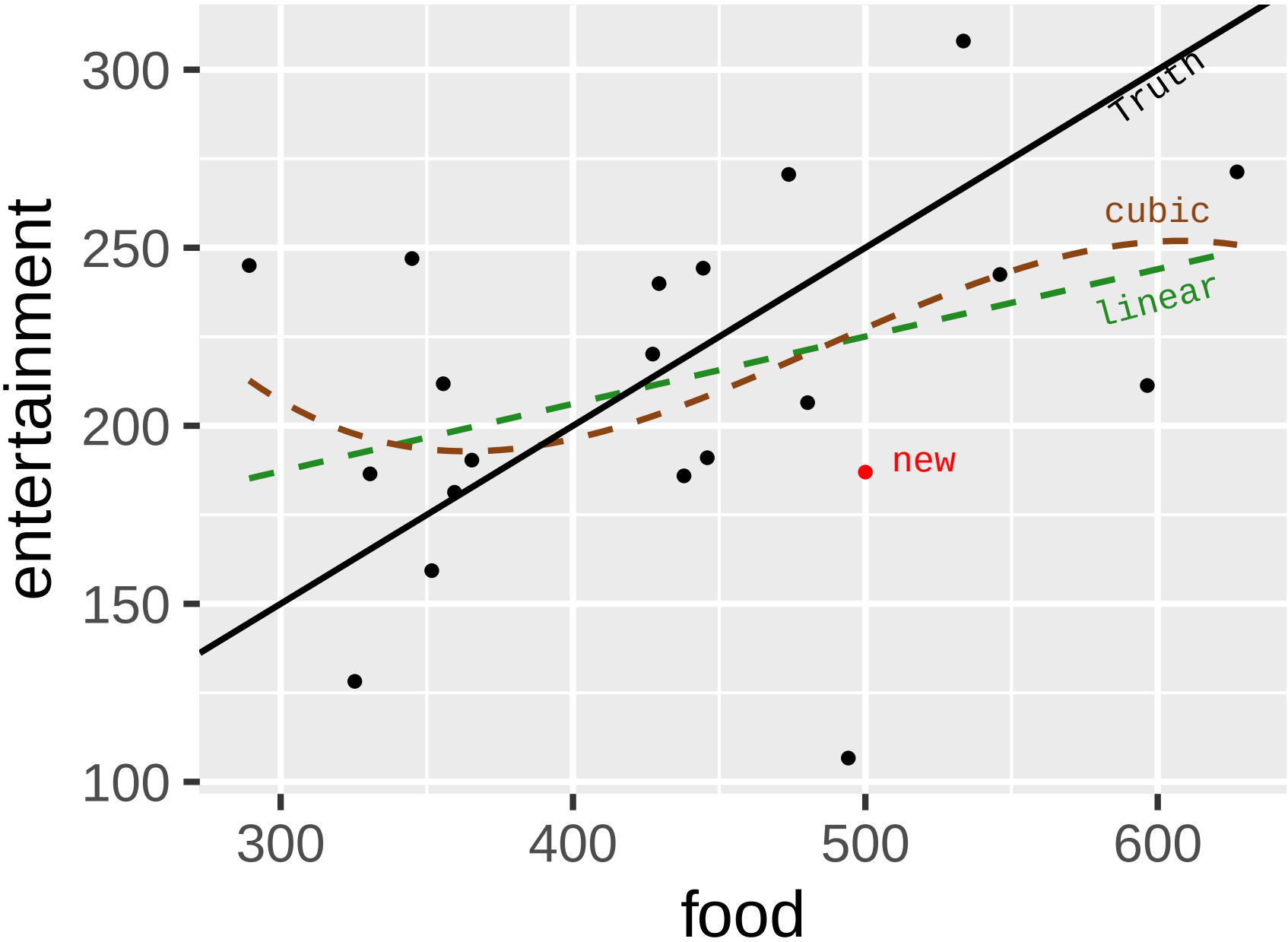
can approximate by linear function



can approximate by cubic function



the new observation



Approximation

As you can see, based on **this particular training sample** both cubic or linear approximation will give almost the same prediction at $X = 500$.

However, both approximations to $\hat{f}$ will be systematically **under-predicting on average** the spending on entertainment for new draws from the population of UvT students who have food expenditure of 500 euro.

But these approximations will still be **over-predicting in some cases** (like in the example), due to the variation around the "truth".

However, this is only based on this training sample. When we re-estimate the models on a new training sample, the predictions at $X = 500$ will change!

What can theory say about the MSE of any predictive model in expectation?

Bias-Variance decomposition

We can decompose the expected MSE into:

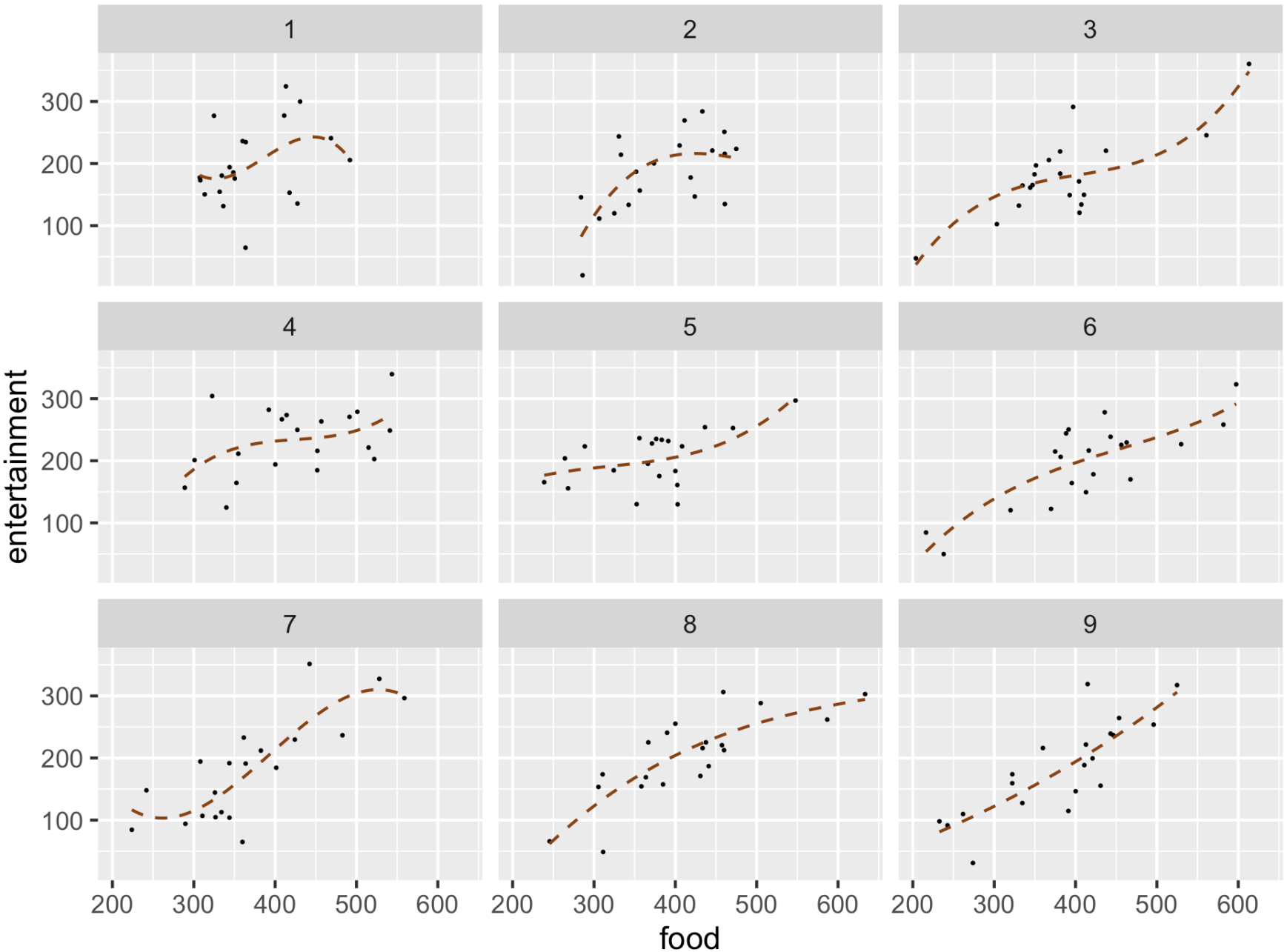
$$E_{Y,Y^T}[(Y - \hat{Y})^2 | X, X^T] = \underbrace{E\left[\left(f(X) - \hat{f}(X)\right)^2\right]}_{\text{reducible}} + \underbrace{var(\varepsilon)}_{\text{irreducible}} \tag{16}$$

which can be even further decomposed into:

$$\begin{aligned} E_{Y,Y^T}[(Y - \hat{Y})^2 | X, X^T] = & \underbrace{E_{Y^T}\left[\left(\hat{f}(X) - E_{Y^T}[\hat{f}(X)|X, X^T]\right)^2 | X, X^T\right]}_{var(\hat{f}(X))} \\ & + \underbrace{(f(X) - E_{Y^T}[\hat{f}(X)|X, X^T])^2}_{Bias^2(\hat{f}(X))} \\ & + \underbrace{var(\varepsilon)}_{\text{irreducible}} \end{aligned} \tag{17}$$

The decomposition shows the sensitivity of *prediction errors* to the *sampling variability* of Y^T (the training data).

Different training samples



e.g., bias and variance of the cubic function

In our example, the true CEF is linear ($Y = 0.5X + \epsilon$)

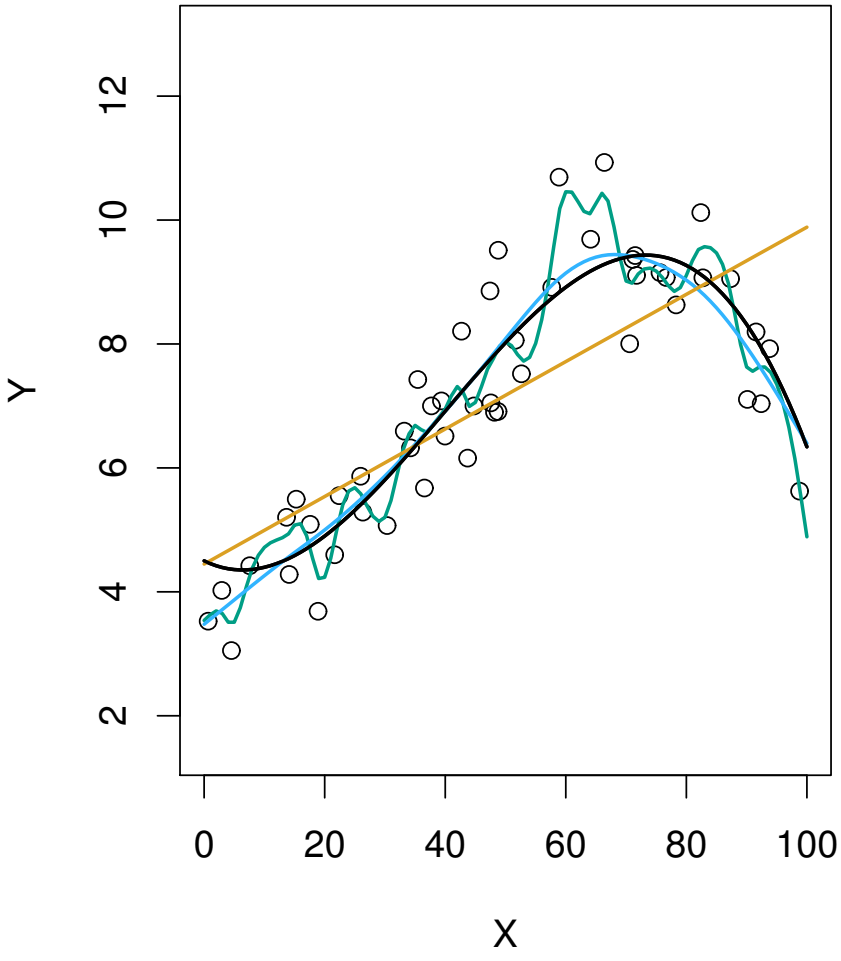
If we approximate it with a cubic function, we will on average capture the true CEF. (i.e., think what happens if we average all those cubic polynomials we estimate in different subsamples on the previous slide) $\Longrightarrow$

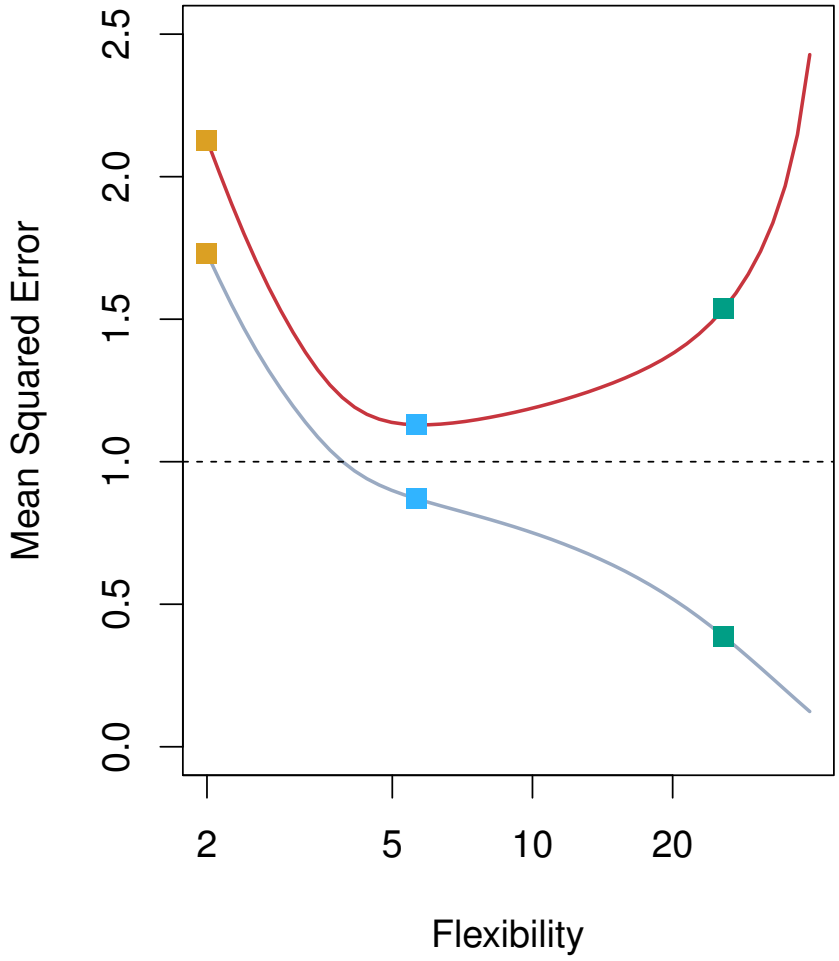
$$(f(X) - E_{Y^T}[\hat{f}(X)|X, X^T])^2 = 0$$

However, in any particular sample, the cubic polynomial function will vary a lot around its own expected shape, which in this case is the true CEF.

$$E_{Y^T}\left[\left(\hat{f}(X) - E_{Y^T}[\hat{f}(X)|X, X^T]\right)^2 |X, X^T\right] > 0$$

U-shaped test MSE





(Fig. 2.9 from "An Introduction to Statistical Learning, with applications in R")

Bias vs Variance

- $Bias(\hat{f}(x))$ = error from approximating the true f with some simplified $\hat{f}$
- $var(\hat{f}(x))$ = changes in $\hat{f}$ in different samples of X
- Restrictive models with few interaction terms are usually:
  - too simple, not capturing the population CEF $\longrightarrow$ High bias.
  - robust in different samples $\longrightarrow$ Low variance
- Flexible models wih many interaction terms are usually:
  - sophisticated enough to capture the population CEF on average $\longrightarrow$ Low bias.
  - but change a lot in different samples $\longrightarrow$ High variance
- Hence, the **bias-variance trade-off**

Questions

When more flexible model will be preferred over less flexible?

- the shape of CEF is linear in parameters
- the shape of CEF is highly non-linear in parameters
- the variance of noise is high
- the variance of noise is low
- the number of observations per number of parameters (n/p) is high
- (n/p) is low

Let's return back to exploring properties of Ridge and Lasso

Linear model selection and regularization (+ tutorial 1)

OLS

What do we know about OLS properties when the CEF is linear, the errors are uncorrelated, have equal variances and expectation value of zero?

Answer:

- Gauss-Markov theorem: OLS is BLUE – Best Linear Unbiased Estimator

What if CEF is non-linear? Answer:

- Then OLS is the Best Linear Unbiased Estimator of the linear approximation of the CEF (See Angrist & Pischke)

To sum up: OLS is unbiased for the linear approximation of the CEF, and among the unbiased estimators it has the lowest variance under homoskedasticity. OLS is good for inference, but unlikely to be the best tool for prediction.

Revisiting the simple example

Remember in the example:

$$y_i = \mu + \epsilon_i \tag{18}$$

- The CEF of y_i is linear (a constant).
- The assumptions satisfy Gauss-Markov theorem
- OLS estimator would be $\bar{y}$
- And yet, we know that the best predictor: $\hat{y} < \bar{y}$

Example 2

$$y_i = \alpha + \beta x_i + \epsilon \tag{19}$$

and $E(x_i) = 0,\ var(x_i) = 1$

Estimates:

Parameter	Est.	S.E.
α	0.000	0.001
β	20.000	100.000

Question:

- What should be our prediction for y_i when $x = 1$?

Example 2

How about this set of estimates?

Parameter	Est.	S.E.
α	0.000	0.001
β	20.000	1.000

Question:

- What should be our prediction for y_i when $x = 1$?

Shrinkage and selection models

Regression regularization:

- Ridge and Lasso regressions shrink estimates of OLS regression towards zero to decrease the variance
- **Ridge** regression shrinks all parameters **towards zero**
- **Lasso** regression shrinks some parameters **towards zero** and some may become **exactly zero** $\Longrightarrow$ Lasso is used also as variable selection
- There is also **best subset selection** method, but computationally complex (See ISLR Chapter 6.1 for self-study)

Ridge regression

$$\min_{\beta} \underbrace{(y_i - \beta_0 - \sum_{j=1}^{k} \beta_j x_{ij})^2}_{RSS} + \underbrace{\lambda \sum_{j=1}^{p} \beta_j^2}_{\text{shrinkage penalty}} \tag{20}$$

Because we penalize the square of β coefficients, Ridge will give smaller – or shrunk – coefficients as compared to OLS.

Questions:

- What happens if $\lambda = 0$?
- What happens if $\lambda \longrightarrow \infty$?

λ is a tuning parameter $\Longrightarrow$ find the optimal λ through cross-validation.

Disadvantage: Keeps all predictors in the model.

Lasso

$$\min_{\beta} \underbrace{(y_i - \beta_0 - \sum_{j=1}^{k} \beta_j x_{ij})^2}_{RSS} + \underbrace{\lambda \sum_{j=1}^{p} |\beta_j|}_{\text{shrinkage penalty}} \tag{21}$$

Because we penalize the **absolute value** of β coefficients, Lasso may give corner solutions compared to Ridge (i.e., $\beta_j = 0$ for some j).

Standartization

Since Ridge and Lasso regression is not invariant to rescaling, the common approach is to standardize all variables:

$$\tilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \bar{x}_j)^2}} \tag{22}$$

i.e., each predictor is centered around its own mean and the standard deviation is set to one.

`glmnet` package in `R` standardizes all predictors by default, but you can switch it off if you really need to.

How to choose lambda?

If you have sample large enough:

- Randomly split your data into three disjoint subsamples:
  1. Training set (1/3)
  2. Validation set (1/3)
  3. Test set (1/3)
- Use training set to run Ridge/Lasso regressions at different values of λ (and maybe other predictive models of your choice)
- Once you get $\hat{f}(\lambda)$, predict y for the validation set and choose the model (and corresponding λ^*) that minimizes the MSE
- Finally, estimate the test MSE of the chosen model.
- Question: Why don't we use the MSE of the chosen model in the validation set? Why do we need a test set?

Validation vs test MSE

Suppose that λ_1 and λ_2 are, in fact, equivalent in terms of prediction accuracy.

Due to sampling uncertainty:

$$MSE(\lambda_1) \sim N(5, 1);\ MSE(\lambda_2) \sim N(5, 1)$$

In the validation step, we choose the model with the lowest MSE.

Simulation

Pseudocode:

- Generate a table (matrix) of two random variables:

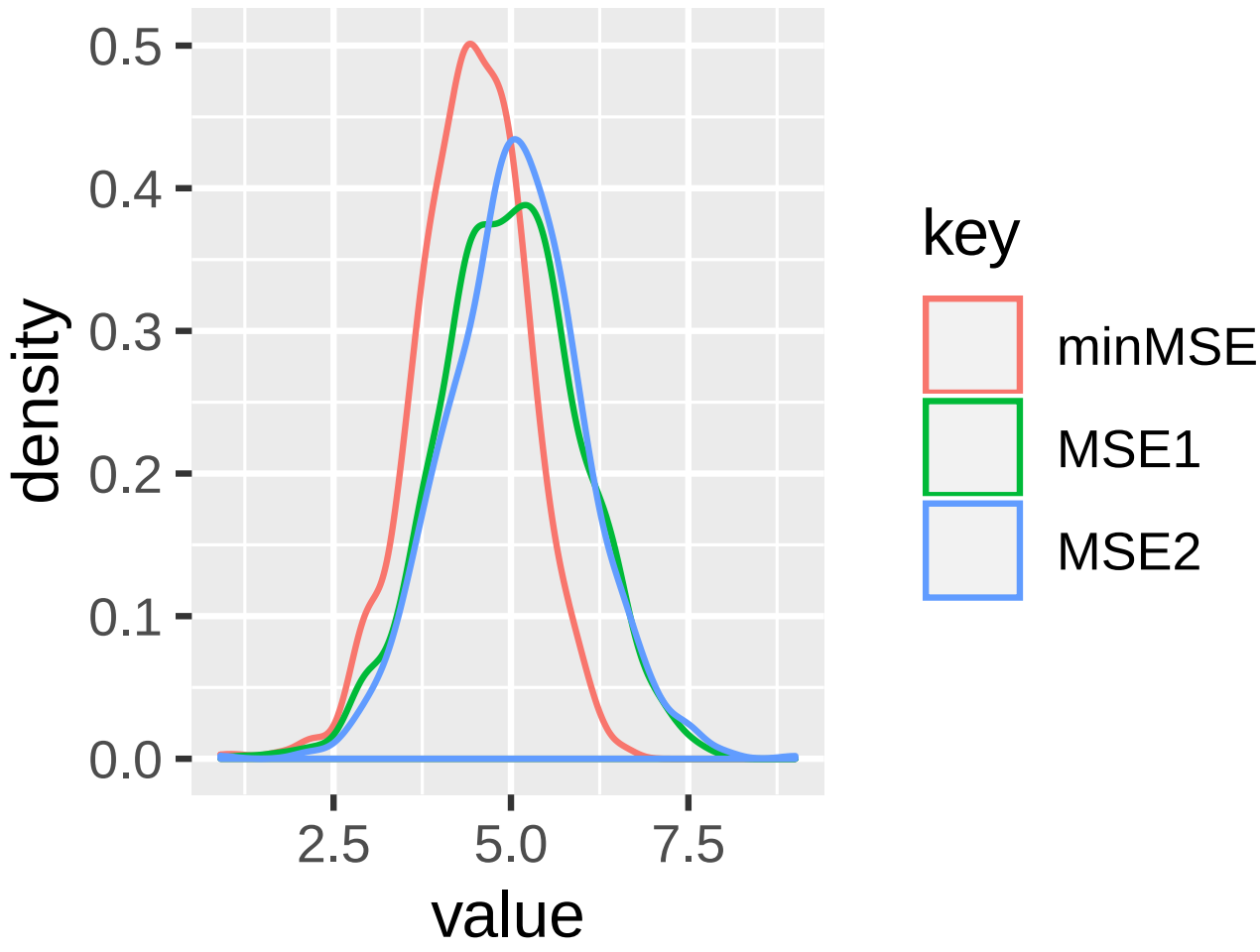
$$MSE1 \sim N(5, 1);\ MSE2 \sim N(5, 1)$$

- Create a new variable `minMSE`:

$$minMSE_i = \min(MSE1_i, MSE2_i)$$

	MSE1	MSE2	minMSE
1	5.26	5.05	5.05
2	5.90	6.56	5.90
3	4.63	4.86	4.63
4	4.54	4.09	4.09
5	4.64	6.35	4.64
6	3.83	5.89	3.83

Plot MSE validation vs test



Cross-validation

What to do when the sample size is not that large? (Usual scenario)

Use k-fold cross-validation:

1. Pick λ
2. Split the sample randomly into k parts
3. Pick the first split as a validation set, and the rest $k-1$ splits use together as a training set
4. Calculate MSE_1
5. Repeat using different splits as a validation set at the time, and obtain $MSE_2, \dots, MSE_k$

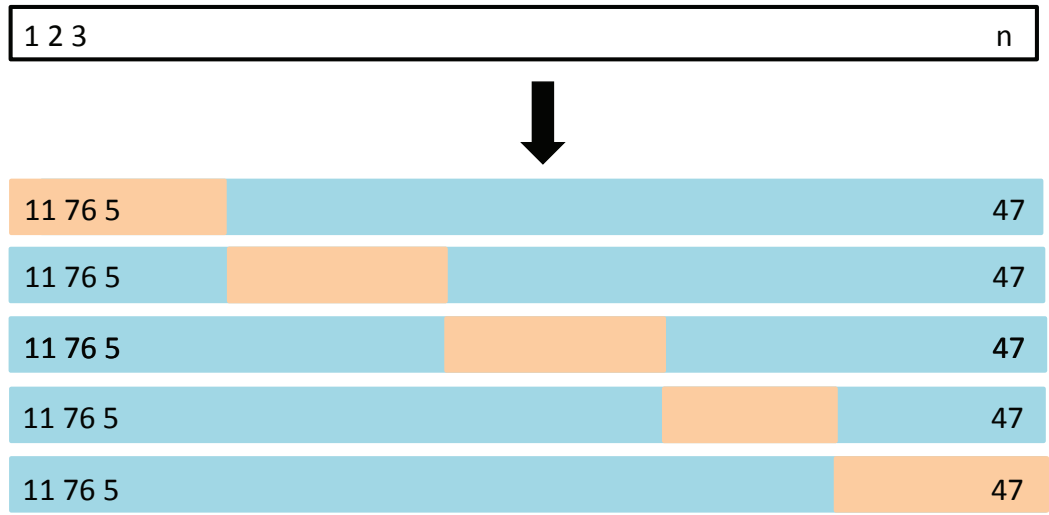


Fig. 5.5 from ISLR

$$\underbrace{CV(\lambda) = \frac{1}{k}\sum_{i=1}^{k} MSE_i}_{\text{Approximates test error}}$$

Cross-validation (whole procedure)

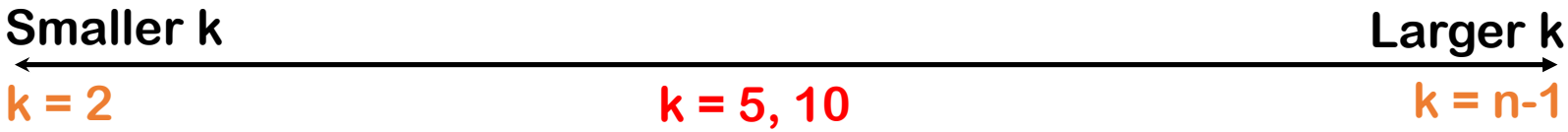
1. Perform cross-validation at different values of λ
2. Pick $\lambda^\star$ with the minimum $CV(\lambda)$
3. Re-run the model using $\lambda^\star$ on the whole sample
4. Use $CV(\lambda^\star)$ as proxy for test error

Cross-validation: Which k to use?

What can guide the choice of k?

How many folds is it optimal to use for cross-validation?

Choice of k: bias-variance trade-off



Smaller training sample

Larger training sample

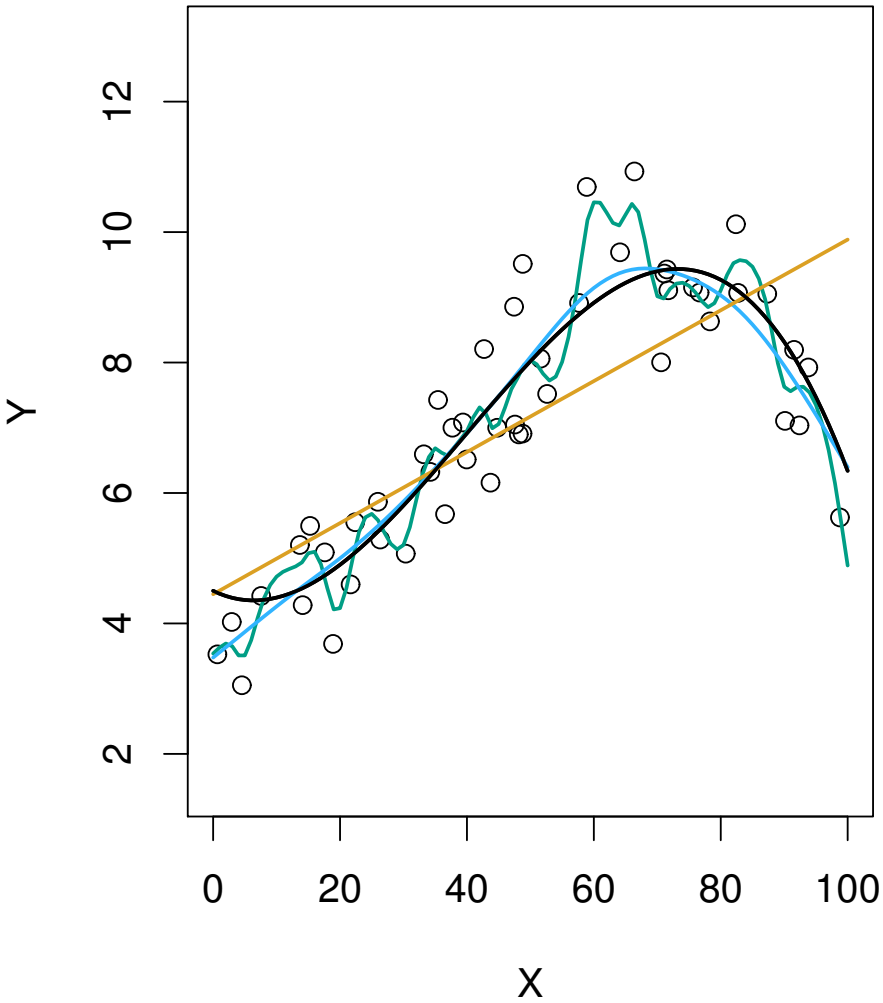
Estimates of errors biased upwards with respect to errors of the model trained on the full sample

But lower variance

Estimates of errors unbiased, since using almost all sample for training

But correlated errors, thus, higher variance

The ISLR example



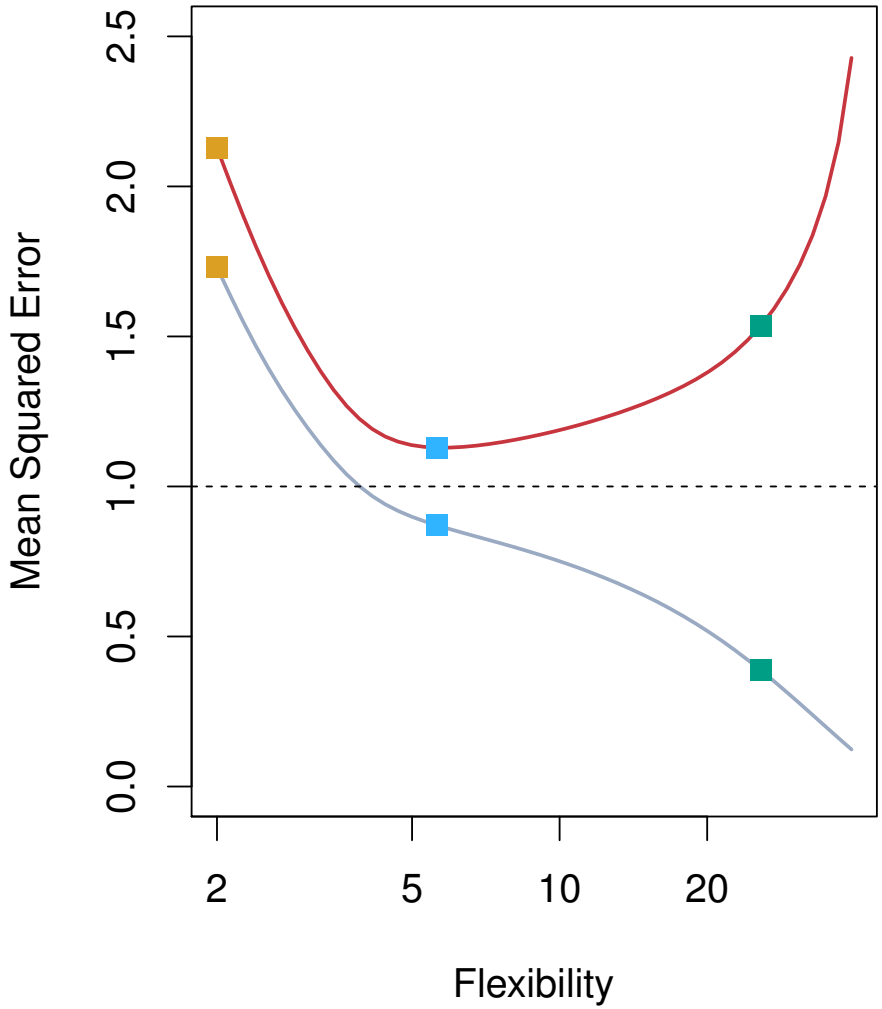


Fig. 2.9 from ISLR

The ISLR example: true vs CV MSE

Blue line: True MSE. Black line: $(n-1)$-fold CV errors. Orange line: 10-fold CV errors.

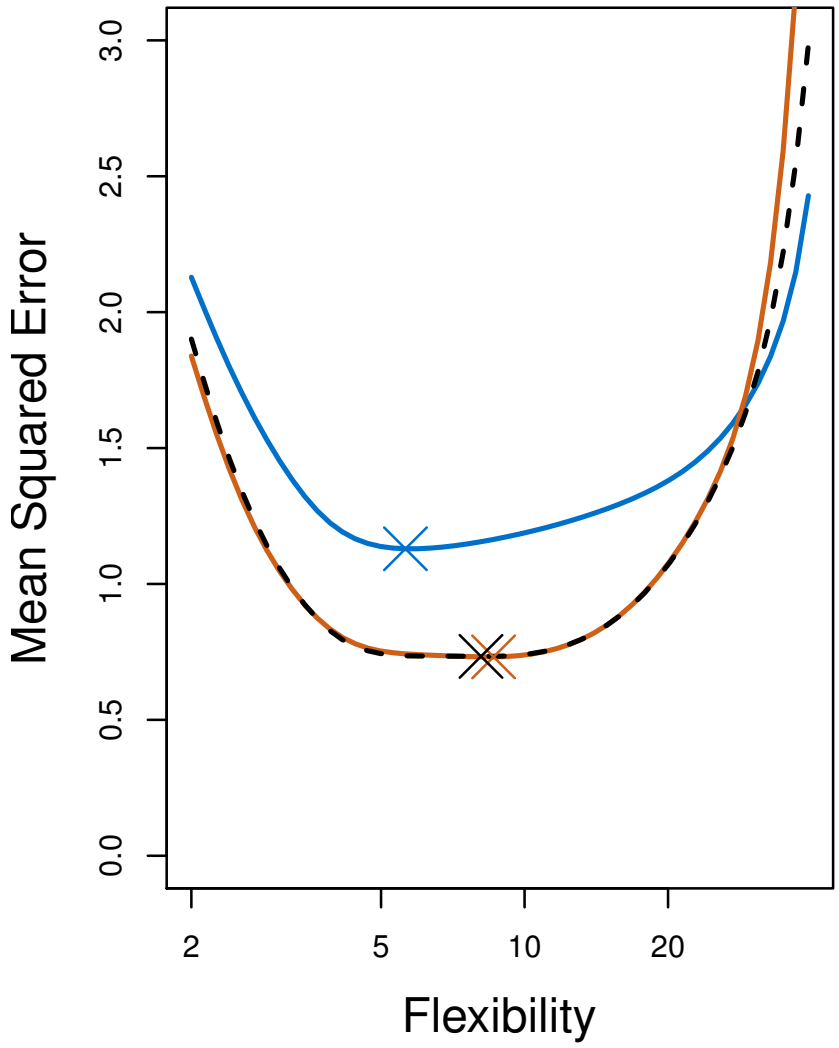


Fig. 5.6 from ISLR.

Bottomline: CV or not CV

Lesson 1

If you do NOT care about the test MSE, then 5–, 10– CV is good for searching the optimal λ.

Lesson 2

If you DO care about the test MSE, then better keep a separate test set and use the remaining sample for CV procedure, then estimate the error rate on on the test set that has not been used in fitting the model. (like we do in the tutorial)

Post-selection inference

OLS

Assume we are interested in estimating β in Regression 23:

$$y_i = \beta d_i + \alpha_1 z_{1,i} + \alpha_2 z_{2,i} + \ldots + \alpha_k z_{p,i} + \varepsilon_i \tag{23}$$

where d_i is a variable of interest and $\{z_{1,i}, \ldots, z_{p,i}\}$ is a set of controls (incl. the constant).

Partialling out estimator

Alternatively, we can estimate β by first partialling out $Z = \{z_1, \dots, z_p\}$ from y and x:

$$y_i = \delta Z_i + \epsilon_y \tag{24}$$

$$d_i = \psi_1 Z_i + \epsilon_d \tag{25}$$

And then regressing the residuals of y on the residuals of x:

$$\epsilon_y = \tilde{\beta} \epsilon_d \tag{26}$$

We can show mathematically that:

$$\tilde{\beta} = \beta$$

(proof)

Setting with many potential controls (Chernozhukov et al., JEP 2014)

$$y_i = \beta d_i + \alpha_1 z_{1,i} + \alpha_2 z_{2,i} + \ldots + \alpha_k z_{p,i} + \varepsilon_i$$

Setting:

- d_i is exogenous after conditioning on controls Z
- p controls, n observations, and $p > n$
- **Approximate Sparsity Condition**:
  only some $s << \sqrt{n}$ controls are important, but we do not know which exactly.
  In other words, only a small number of non-zero coefficients are needed to drive the approximation errors down.

Setting with many potential controls (Chernozhukov et al., JEP 2014)

We want to use model selection (e.g., Lasso regression), but doing it naively can **introduce substantial biases**. [See simulations in Tutorial 2]

- because **Lasso** tackles prediction **not for inference**

Transforming inference problem into prediction problem (Chernozhukov et al., JEP 2014)

The partialled out estimation can be also represented as:

$$\underbrace{(y_i - \hat{y}_i(Z))}_{\epsilon_y} = \tilde{\beta} \underbrace{(d_i - \hat{d}_i(Z))}_{\epsilon_d} \tag{27}$$

We transformed our *impossible* inference problem of finding β when $p > n$ into:

- Solving two prediction tasks – i.e., $\hat{y}_i(Z))$ and $\hat{d}_i(Z))$ – for each we can use Lasso
- + one simpler inference task

Double Selection procedure (Chernozhukov et al., JEP 2014)

The Double Selection procedure in three steps:

1. Run Rigorous Lasso for y on Z to select a subset $Z^{\star,y} \in Z$ that best predicts y
2. Repeat the same for d to select a subset $Z^{\star,d} \in Z$ that best predicts y
3. Regress y on d and the union of $Z^{\star,y} \cup Z^{\star,d} \Longrightarrow$ get β

Final ingredient: Rigorous Lasso

The choice of λ when the final goal is inference:

- We have seen how cross-validation can help selecting λ for prediction, but it is not theory grounded
  i.e., CV does not guarantee any Post-Lasso properties
  Post-Lasso is an OLS estimator after we used Lasso to select variables)
- However, we are interested in the right choice of λ because we also care about efficiency of the estimator in our post-Lasso regressions
- Belloni et al. (2012) propose Rigorous Lasso which uses feasible estimation of theoretically-founded optimal λ
- As we can see from Tutorial 2, Rigorous Lasso indeed performs better than cross-validated Lasso.

Rigorous Lasso: optimal λ under homoscedasticity

For example, under homoscedasticity, an X-independent choice of λ:

$$\lambda = 2c\sqrt{n}\hat{\sigma}\Phi^{-1}\left(1 - \frac{\gamma}{2p}\right)$$

where

- $c = 1.1$ for Post-Lasso (which is our case; otherwise 0.5 for Lasso)
- $\hat{\sigma}$ an estimate of $sd(\epsilon_i)$.
- Φ^{-1} is the inverse of a the cumulative standard normal distribution
- γ is the probability level ("of mistakenly not removing X's when all of them have zero coefficients")
- n is # of obs and p is # of predictors

The search for optimal λ starts with some first proxy of $\hat{\sigma}$, and continues using iterative approach until convergence. $\Leftarrow$ Automatically implemented in rlasso() function in R.

(See details in Appendix A in this paper)

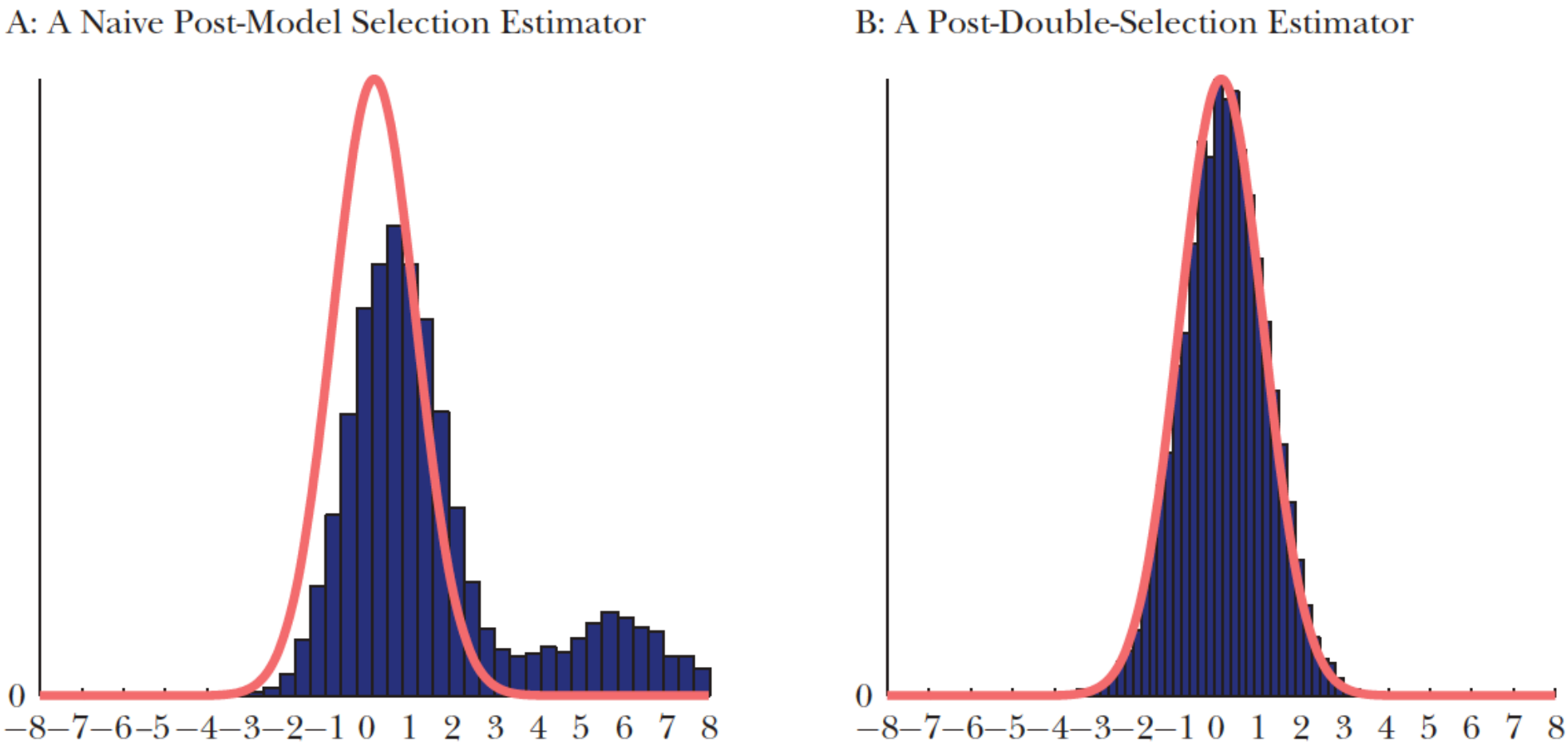
Rigorous Lasso: other cases

There are other theory-based formulas for cases with heteroscedastic errors or clustered errors.

Further reading:

- documentation for R's *hdm* package (link)
- documentation for Stata's package rigorous Lasso (link)
- Belloni, Chernozhukov, and Hansen (2012) (link)

Fig 1 from Chernozhukov et al. (JEP, 2014)



Histogram of simulation results in blue and density of the true distribution of the estimator in red.

We see similar results in Tutorial 2.

When useful?

Cases when Double Selection procedure can be very useful:

- Theory-agnostic approach with many observable characteristics.
- Theory tells which variables are important, but you want to check for robustness of results to the inclusion of possible interaction terms and higher-order terms.

In general, any situation with **conditional independence assumption**:

"The dependence between treatment assignment and treatment-specific outcomes can be removed by conditioning on the observable variables."