Model selection

Overview

Quick review of multicollinearity and polynomial regression

Model selection

- Overfitting
- Statistics and methods useful for model selection

If there is time

- Using ggplot to visualize regression models
- Regularization methods

Review: Multicollinearity

Multicollinearity occurs when two or more variables are closely related to each other

- e.g., if they have a high correlation

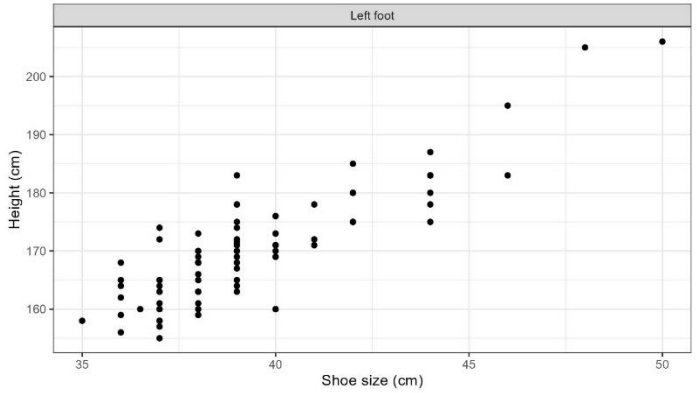
Multicollinearity can make our estimate of the regression coefficients unstable

- e.g., standard error of coefficients become large

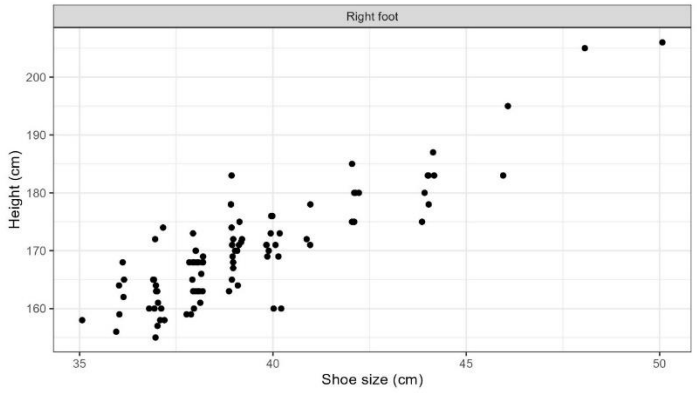
The **variance inflated factor** is a statistic that can be computed to test for multicollinearity

- Rule of thumb: suspect multicollinearity for VIF > 5

Left foot



Right foot



car::vif(lm_fit)

Review: Polynomial regression

Polynomial regression extends linear regression to non-linear relationships by including nonlinear transformations of predictors

$$\text{salary} = \beta_0 + \beta_1 \cdot \text{endowment} + \beta_2 \cdot (\text{endowment})^2 + \beta_3 \cdot (\text{endowment})^3 + \varepsilon$$

Still a linear equation but non-linear in original predictors

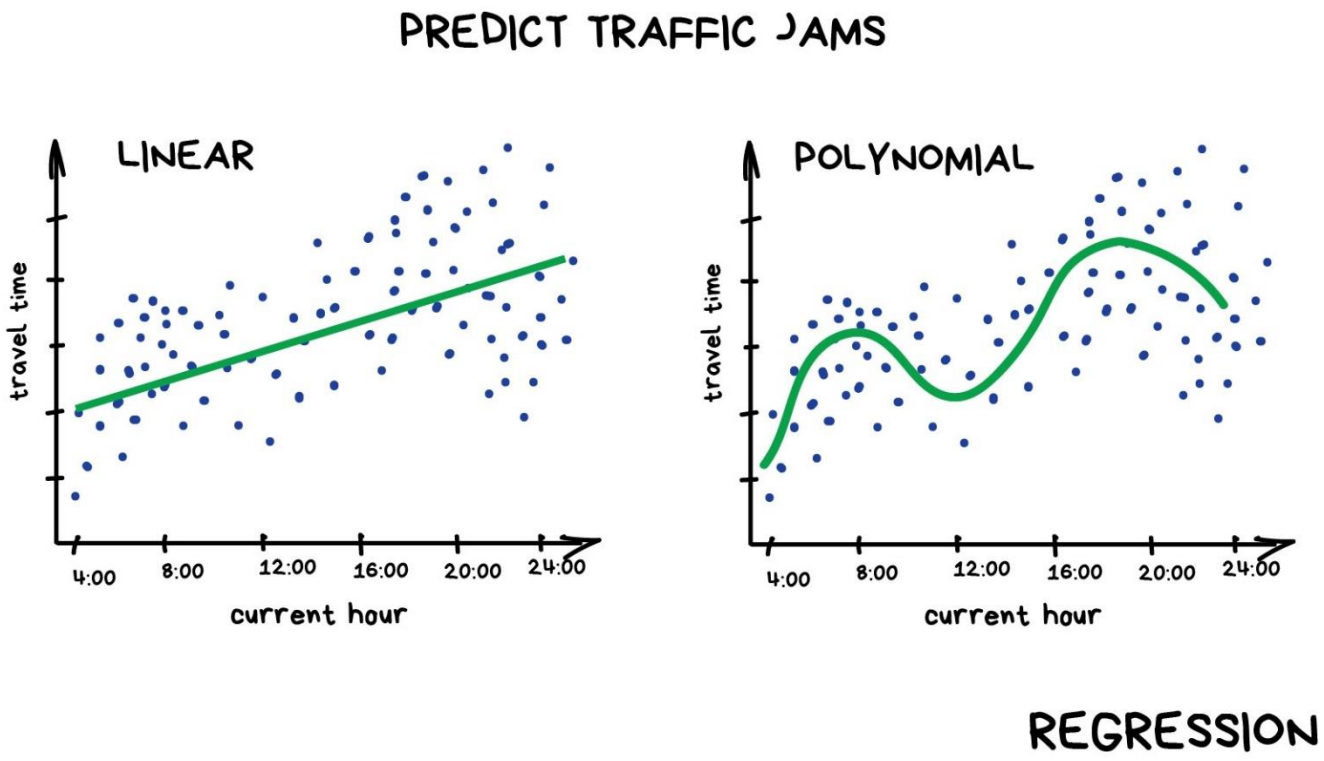
Review: Polynomial regression

Polynomial regression extends linear regression to non-linear relationships by including nonlinear transformations of predictors

We can compare model fits by:

- Assessing if the coefficients on the higher order terms are statistically significant
- Looking at the R^2 values
- Running hypothesis tests comparing nested models
- Etc.

Let’s review this in R…



Model selection

Model selection

Model selection is the process of selecting a statistical model from a set of candidate models

- E.g., which explanatory variables, interaction terms, transformations of variables, etc. to include in a final model

Model selection is a bit of an art

- “All models are wrong but some are useful”

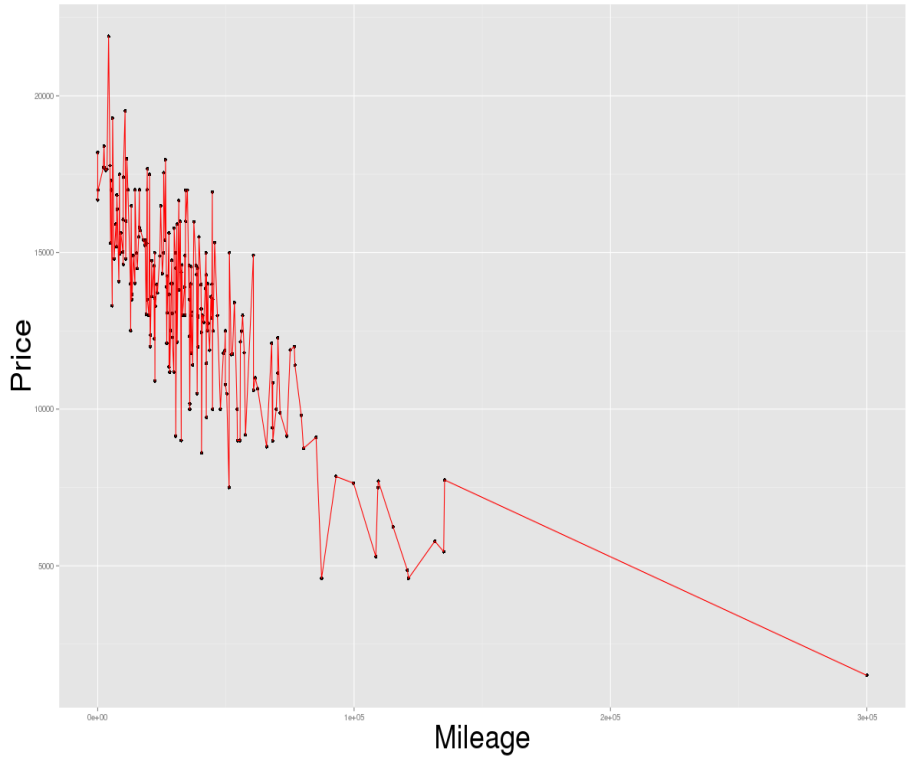
Model selection

Model selection depends on our goal, which usually is either:

- Making accurate predictions
- Understanding relationships in our data

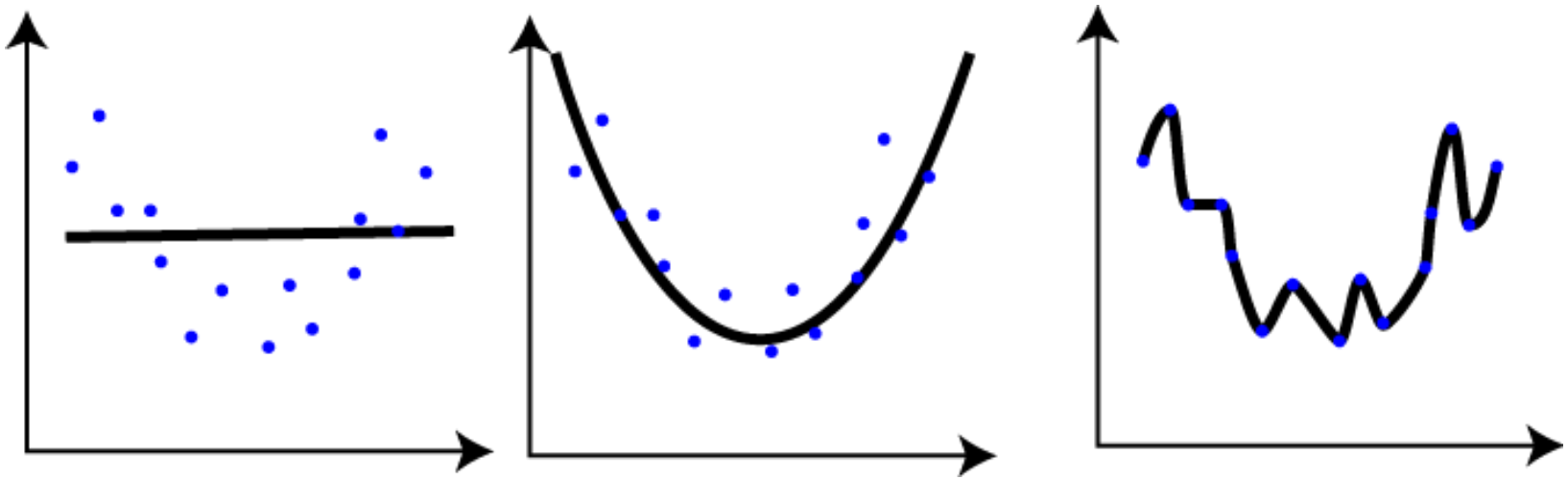
If we fit our model too closely to the data sample we have, we are likely to fail in both of these goals

- i.e., it will be hard to understand relationships between explanatory and response variables
- and model will not make good predictions on new data



Overfitting

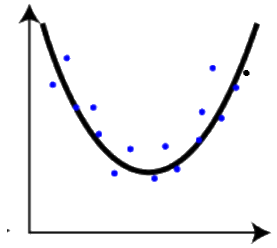
Overfitting occurs when we generate a function that too closely matches random sample we have, but does not generalize to the full probability distribution

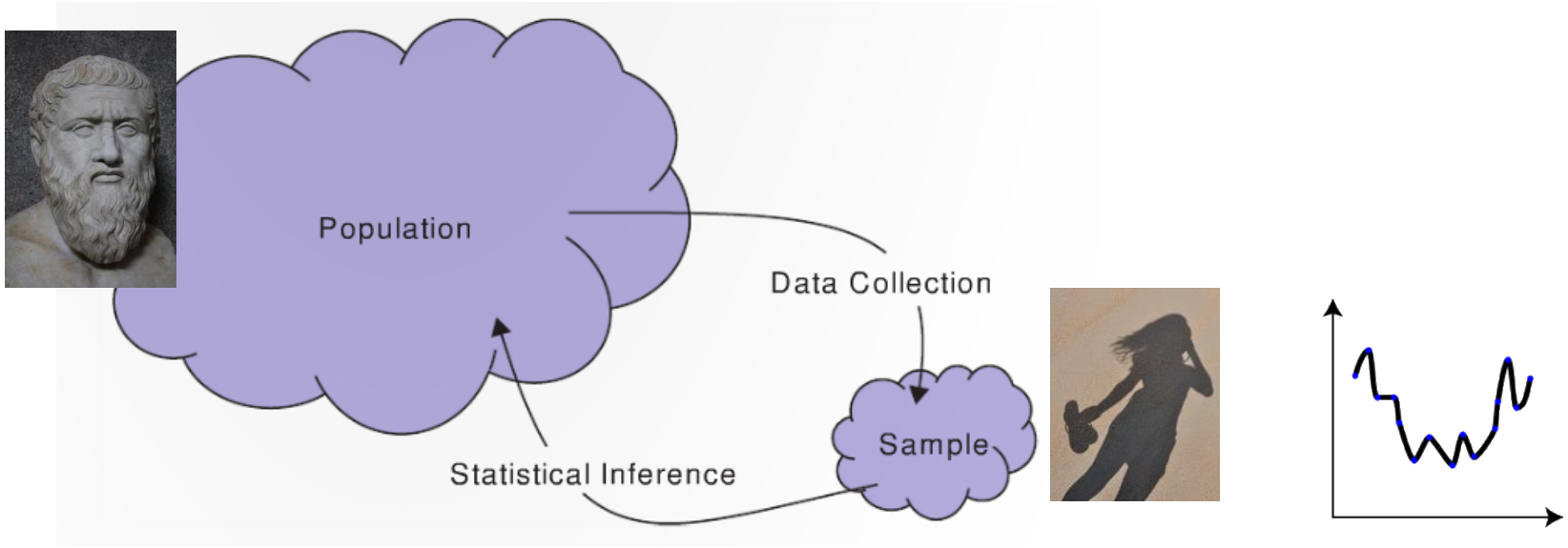


Overfitting

Overfitting occurs when we generate a function that too closely matches random sample we have, but does not generalize to the full probability distribution

- The model is fit to closely to the shadows and not getting at the Truth





Overfitting song

https://www.youtube.com/watch?v=DQWI1kvmwRg

Selecting models methods

There are a number of different methods for selecting models. Four we will briefly discuss are:

1. Creating measures of fit (statistics) that penalize models with more predictors

2. Creating simpler models by removing predictors

3. Evaluating models using cross-validation

4. If there is time: methods that shrink regression coefficients

Model method selection 1

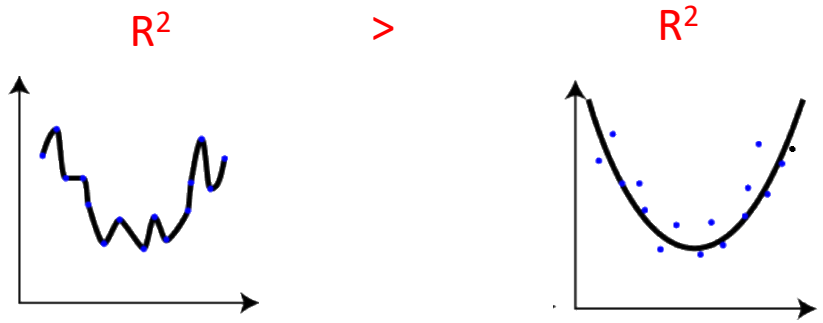
Selected models using statistics that penalize larger models

R^2 as a measure of model fit

We have used the coefficient of multiple determination (R^2) to determine how well our model is fitting the data:

$$R^2 = \frac{SSModel}{SSTotal} = 1 - \frac{SSResidual}{SSTotal} = 1 - \frac{\Sigma_{i=1}^{n}(y_i - \hat{y})^2}{\Sigma_{i=1}^{n}(y_i - \bar{y})^2}$$

R^2 always increases with more predictors x_i because the response variable y can always fit more closely with more predictors



Recall: the standard deviation of the errors: σ_ε

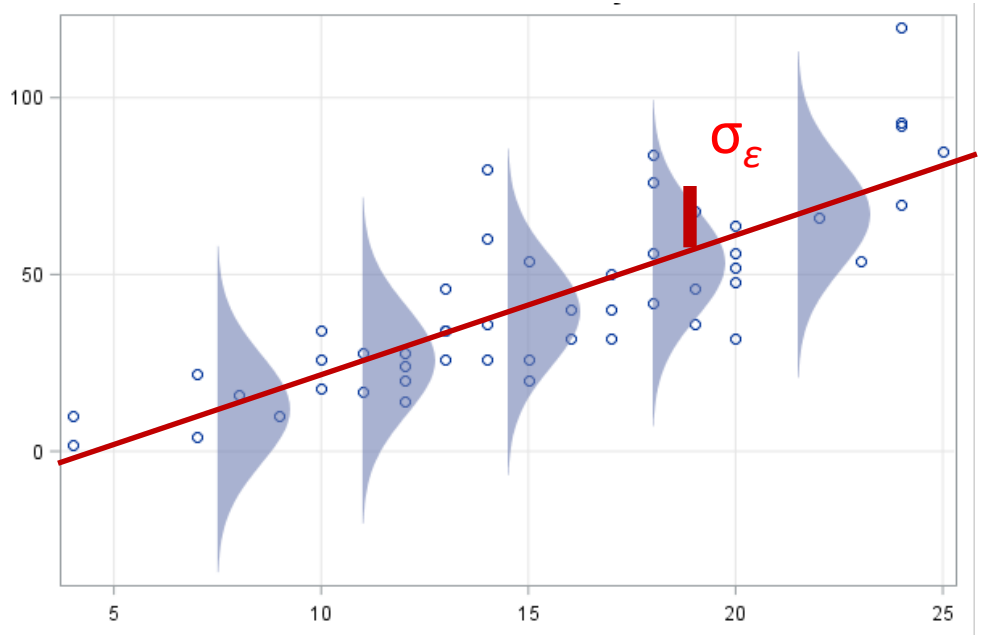
Recall for simple linear regression, the standard deviation of the errors is denoted $\boldsymbol{\sigma_\varepsilon}$ and shows how far the points fall off the true regression line.

We can use the **standard deviation of residuals ($\hat{\sigma}_e$)** as an estimate for σ_ε

For simple linear regression we had:

$$\hat{\sigma}_e = \sqrt{\frac{1}{n-2} SSRes}$$

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

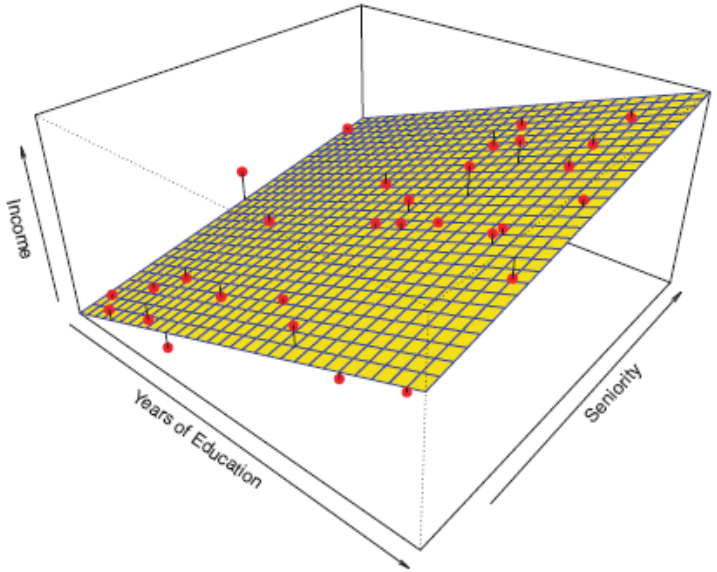


Recall: the standard deviation of the errors: σ_ε

For multiple regression, we with k parameters (i.e., k - 1 predictors) an (almost) unbiased estimate of $\hat{\sigma}_\varepsilon$ is:

$$\hat{\sigma}_e = \sqrt{\frac{1}{n-k} SSRes}$$

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



The residual standard deviation $\hat{\sigma}_\varepsilon$ corrects for bias by dividing the SSResidual by 1/(n - k)

This estimate does not always decrease with more predictors x

Adjusted R^2

The **adjusted R^2** helps account for the number of predictors in the model by using $\hat{\sigma}_\varepsilon^2$

$$R^2_{adj} = 1 - \frac{SSRes/(n-k)}{SSTotal/n-1} = 1 - \frac{\hat{\sigma}^2_e}{s^2_y}$$

The adjusted R^2 does not always give a higher values to the model with the more predictors

- i.e., using this statistic, we will not always say that a model with the most predictors is a “better” fit to the data

Other statistics that penalize larger models

There are several other statistics that also ***penalize models that have more predictors***

- These statistics are only meaningful for within data set comparisons

Akaike information criterion: $AIC \propto 2 \cdot k + n \cdot ln(SSRes/n)$ R: AIC(lm_fit)

Bayesian information criterion: $BIC \propto k \cdot ln(n) + n \cdot ln(SSRes/n)$ R: BIC(lm_fit)

One should select the model with the lowest value on these statistics

Let’s try it in R…

Model method selection 2

Using algorithms to select a subset of variables (stepwise regression)

Brief mention: Variable selection

Variable selection refers to finding models that rely on a small subset of predictors

- This can help make the regression model more interpretable as well

We could use individual predictor's (x's) p-values to determine which predictors to use, however…

- Some of these will be spuriously significant
  - i.e., if H_0 is true for all predictors, ~5 will be significant at $\alpha = .05$ level
- The p-values change as predictors are added and removed
  - Due to multicollinearity

```
lm_fit_mult <-
    lm(log(endowment)  ~
     salary_tot,
     salary_men,
     salary_women
    )
```

Feature selection: deciding which variables to use

Ideally, we would like to try all combinations of predictors, however, if there are k features, there are 2^k possible models which can be intractable

A few heuristic methods exist for selecting smaller models

- Forward selection: start with a model with no predictors and add predictors (until you have enough)
- Backward selection: Start with the full model and delete predictors
- Mixed selection: Use a combination of forward and backward selection

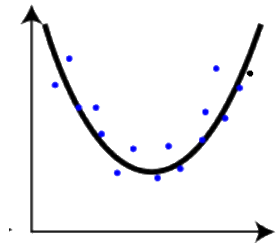
```
Coefficients:
             Estimate  Std. Error t value             Pr(>|t|)
(Intercept) 12.85948900  0.17913330  71.787 < 0.0000000000000002 ***
salary_tot   0.00013686  0.00003950   3.465             0.000551 ***
salary_men  -0.00001569  0.00002019  -0.777             0.437160
salary_women -0.00004304  0.00002156  -1.997             0.046099 *
```

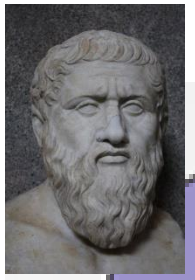
In R: leaps::regsubsets()

Model method selection 3

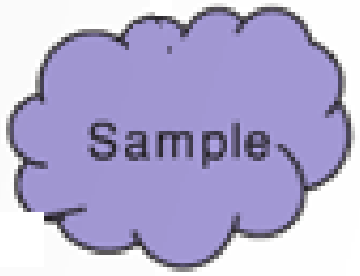
Choosing a model through cross-validation

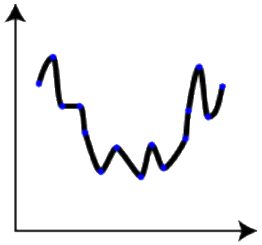
Cross-validation





Training set





Test set

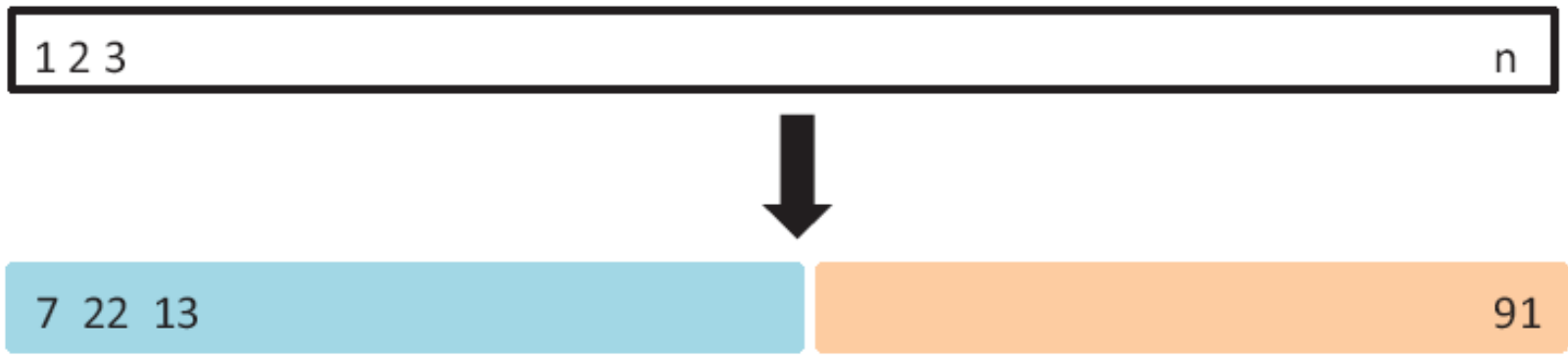
Good predictions on the test set give an estimate of how accurate the model will be on new data from the population

Cross-validation

We run cross-validation by splitting data into two sets:

A training set in which the parameters of a regression model are fit (estimated)

A test set in which the prediction accuracy of our model is assessed



Mean squared prediction error

To evaluate how effective a model is, we can use the mean squared prediction error (MSPE) using the following steps:

1. Fit a model using the training data
2. Make predictions on the test data
3. We can use the MSPE to assess how accurate the predictions are:

$$MSPE = \frac{1}{n_t}\sum_{i=1}^{n_t}(y_i - \hat{y}_i)^2 = \frac{1}{n_t}\sum_{i=1}^{n_t}(y_i - (\hat{\beta}_0 + \hat{\beta}_1 x_1 + \ldots + \hat{\beta}_k x_k))^2$$

Actual y values in the test set

Predicted y values on the test set

Parameters estimated on the training set

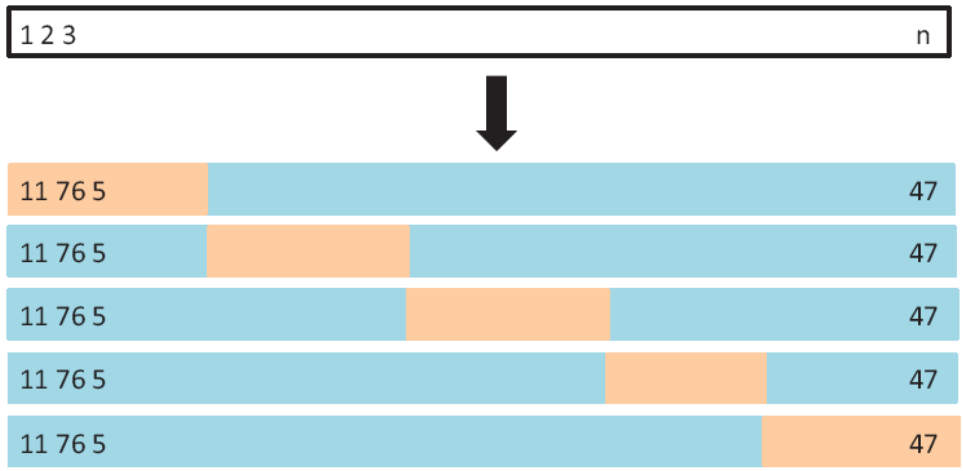
Predictions assessed on the test set

n_t is the number of points in the test set

K-fold cross-validation

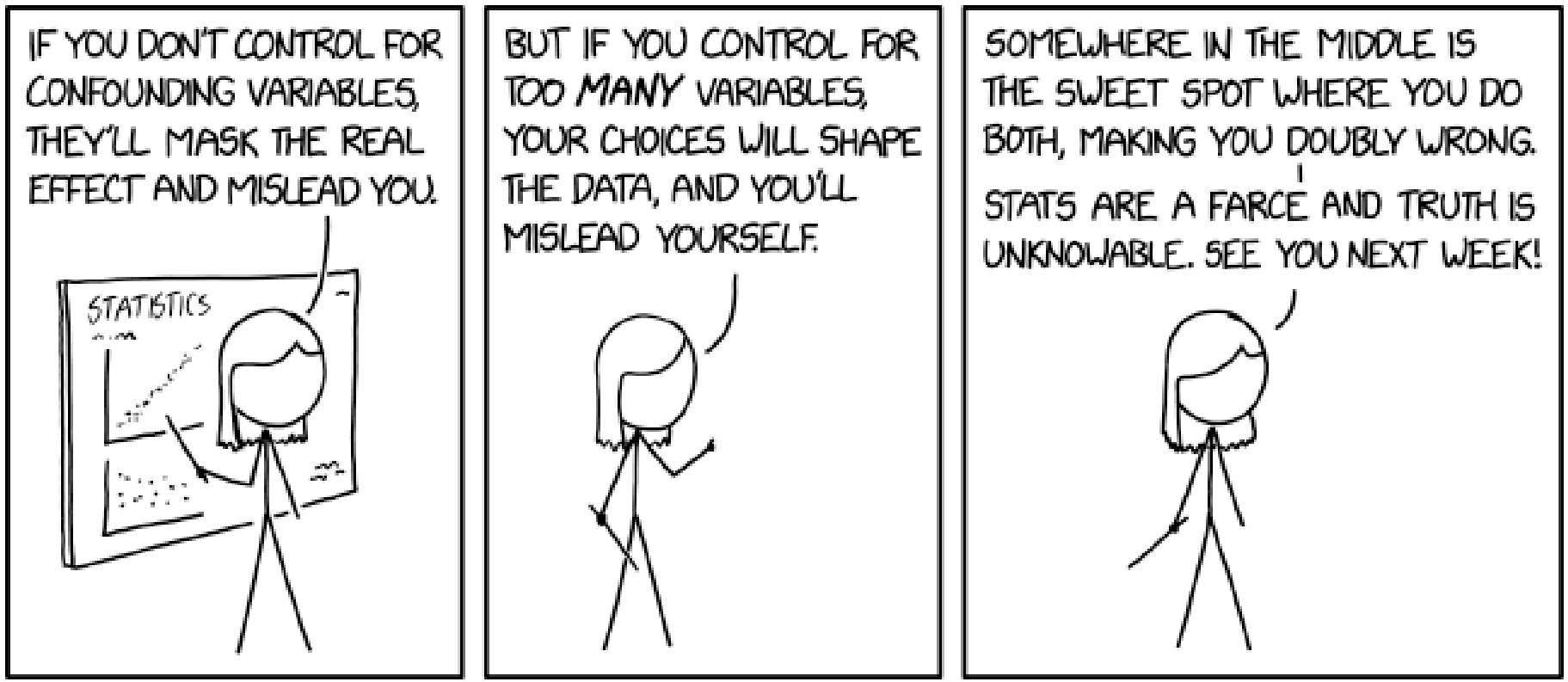
K-fold cross-validation

- Split the data into k parts
- Train on k-1 of these parts and test on the left out part
- Repeat this process for all k parts
- Average the prediction accuracies to get a final estimate of the generalization error



Leave-one-out (LOO) cross-validation: k = n

Let’s try it in R…

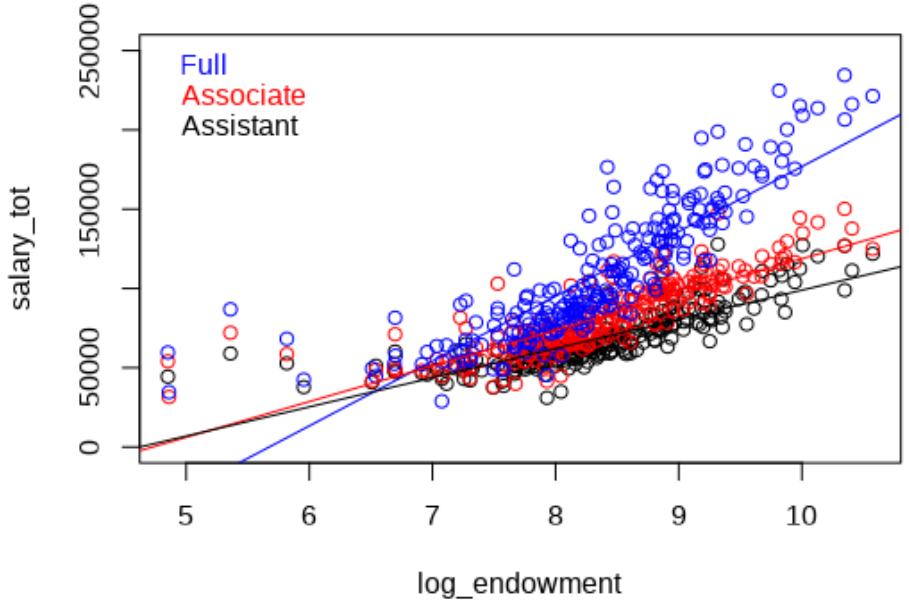


Plotting multiple regression models with ggplot

So far we have plotted our multiple regression models using base R graphics

This was useful for seeing the relationship between how R fits linear models, and what these models represent

However, if you want an easier/prettier way to visualize linear models, we can use ggplot!




```
Call:
lm(formula = salary_tot ~ log_endowment + rank_name + log_endowment:rank_name,
    data = IPED_2)
```

Let's try it in R!

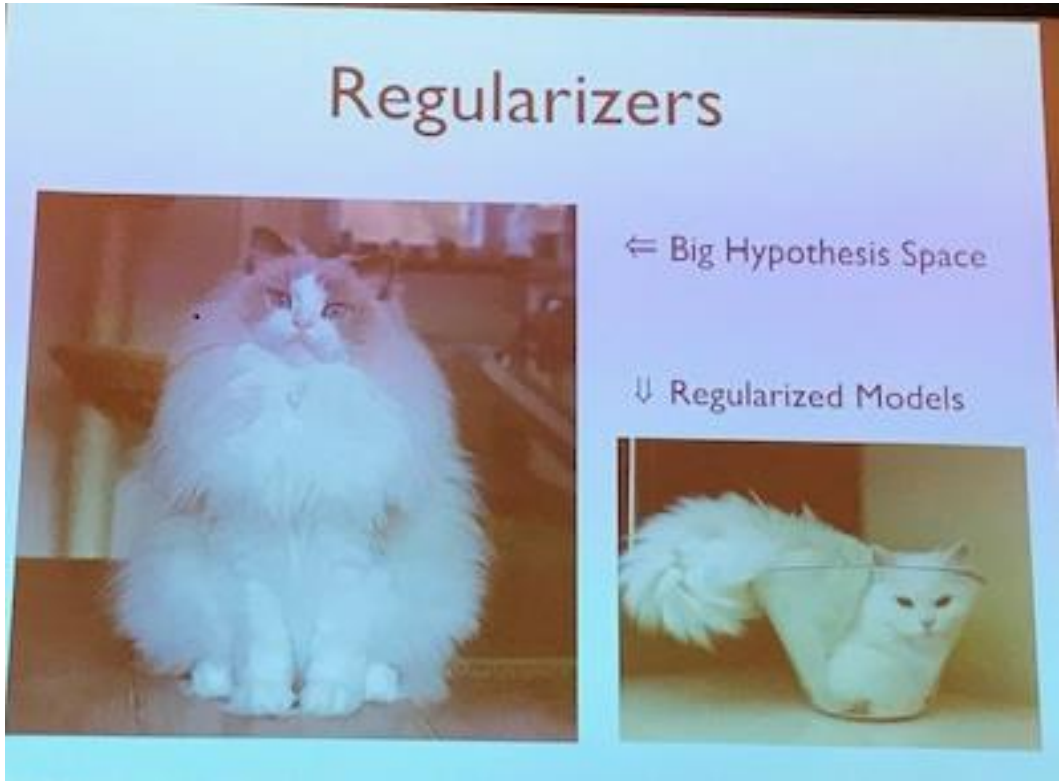
```
Residual standard error: 16260 on 705 degrees of freedom
Multiple R-squared:  0.7806,    Adjusted R-squared:  0.7791
F-statistic: 501.7 on 5 and 705 DF,  p-value: < 2.2e-16
```

Side note

Controlling model complexity with regularization

Brief mention: shrinkage methods

Rather than finding the coefficients $\hat{\beta}_i$ by just minimizing the SSRes, one can also add penalties in the fitting procedure to find simpler models



We will very briefly discuss two techniques:

- Ridge regression (l_2 norm penalty)
- The lasso (l_1 norm penalty)

Brief mention: Ridge regression

Ridge regression finds the coefficients $\hat{\beta}_i$ that minimize:

$$\sum_{i=1}^{n}(y_i - (\hat{\beta}_0 + \sum_{j=1}^{k}\hat{\beta}_j x_{ij}))^2 + \lambda \sum_{j=1}^{k}\hat{\beta}_j^2$$

SSRes

shrinkage penalty

Tuning parameter

What happens if:

- λ = 0
- λ → infinity
- (the coefficients depend on the tuning parameter value)

see the glmnet package

Brief mention: The Lasso

The Lasso finds the coefficients $\hat{\beta}_i$ that minimize:

$$\sum_{i=1}^{n}(y_i - (\hat{\beta}_0 + \sum_{j=1}^{k}\hat{\beta}_j x_{ij}))^2 + \lambda \sum_{j=1}^{k}|\hat{\beta}_j|$$

SSRes

shrinkage penalty

Tuning parameter

Similar to ridge regression but penalizes $|\hat{\beta}_i|$ instead of $\hat{\beta}_i^2$

- i.e., uses the L_1 penalty instead of the L_2 penalty

<u>Advantages</u>

- Final model will often have many set $\hat{\beta}_i$ to 0
- i.e., does variable selection and creates a 'sparse' model

see the glmnet package

Shrinkage/regularization methods

Regularization methods often work very well when we care about making accurate predictions on new data

Theory suggests that these methods work by minimizing the MSPE through a bias-variance tradeoff

- Average MSPE = bias^2 + variance
- We use a biased method (via regularization) to get models that vary less from one random data set to the next, reducing the average MSPE

To learn more about regularization methods, take a class on Machine Learning!