Outlook of the unit

2 Regression

- Linear regression analysis
- Performance evaluation
  - Overfitting and data partitioning
  - Predictive performance
  - Cross-validation
- Polynomial regression & splines
- Penalized regression
  - Lasso
  - Ridge regression
  - Elastic net

Linear regression analysis

Goal of a regression model

- The object of the regression analysis is to explain a so-called **dependent variable** (response variable) by one ("simple regression") or more ("multiple regression") **independent variables** (predictors or covariates).
- In a linear regression model the dependency is modeled by a **linear combination** of the independent variables.

Regression model foundations

In the simplest case the dependency has the following form: $y_i = \beta_0 + \beta_1 x_i$

- y is the **dependent** variable
- x is the **independent** variable
- i is an index over the elements of the data set, from 1 to n.
- In the case of the validity of this strict dependency, in a scatter plot, all observations lie on a straight line with **intercept** β_0 and **slope** β_1.
- As this never happens in practice the model needs to be extended.
- This is done by adding a **random noise** term ϵ_i, for which it is usually assumed that $\epsilon \sim N(0, \sigma^2)$.

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

The regression model in matrix form

The regression model can be written in matrix from:

$$\mathrm{y} = \mathrm{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}.$$

Remember, $\mathbf{y}$ is a numerical outcome variable and $\mathbf{X}$ a set of predictors (covariates).

Estimating the coefficients

- Let $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$ be the prediction of y or the estimated model.
- $e_i = y_i - \hat{y}_i$ represents the i_ith **residual**. This is the difference between the observed value and predicted value by the linear model.
- We define the **residual sum of squares** (RSS) as:

$$RSS = e_1^2 + e_2^2 + \ldots + e_n^2.$$

- The estimates for β are obtained by minimizing the RSS.

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2$$

Illustration: Expenditure survey

Let's consider the following data set from a small survey. An interesting question might be, if the expenditure depends on the age of the surveyed persons.

AGE	GENDER	SATIS	EXHI	TIME	BUDGET	PERSON	JURNEY	VISIT
32	1	3	2	4	30	2	40	1
48	1	4	2	6	60	3	60	5
75	2	4	3	6	50	2	15	15
64	1	5	2	7	100	4	50	20
34	1	2	2	2	10	1	10	0
16	1	1	1	5	5	1	30	0
28	2	4	2	3	7	1	20	1
58	1	4	3	4	55	2	90	20
55	2	5	2	8	70	2	20	10
68	2	5	2	3	25	2	80	8
70	2	4	1	5	40	2	30	30
20	2	2	2	1	3	1	25	0
19	1	3	2	3	10	1	30	1
25	1	3	1	4,5	20	1	15	2
42	2	4	3	6	100	5	25	7
30	1	5	2	4	38	2	10	3
29	2	4	2	3	15	1	45	2
59	1	3	3	5,5	150	6	50	20
41	1	4	1	3	45	3	60	4
67	1	5	3	6	30	1	30	15

Illustration: More than a scatter-plot

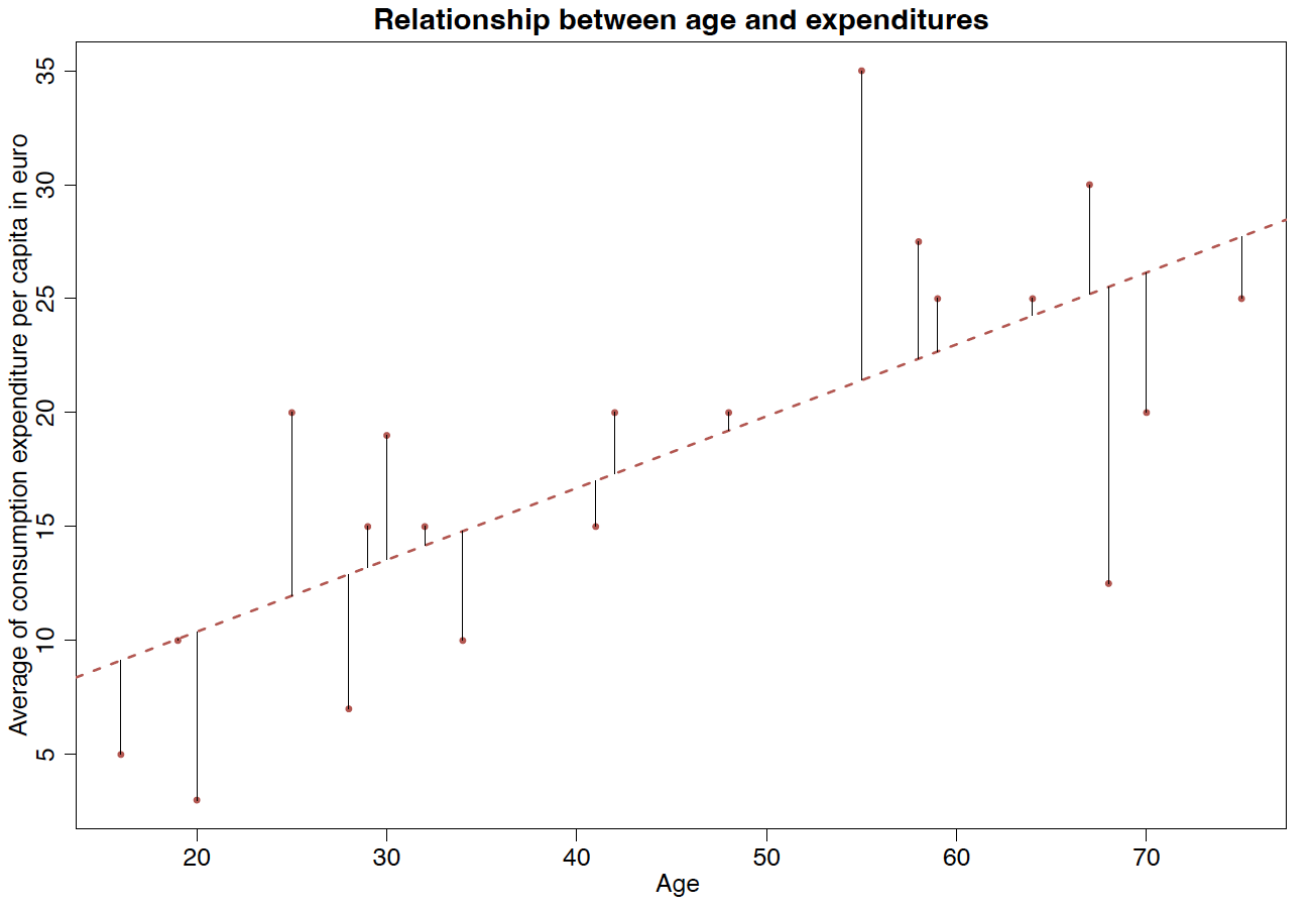


Illustration: Regression line

- In addition to the points, the so-called ***regression line*** is plotted which shows the dependency between the two variables.
- The figure was amended by adding the **estimated residuals** ϵ_i (vertical bars).
- The regressions coefficients β_0 and β_1 were estimated by minimizing the RSS. This is called the least squares method (also known as ***OLS-method***).
- Therefore the regression line lies in such a way that the RSS (differences) is minimized.

Illustration: Interpretation

- The estimated **model parameters** are $\beta_0 = 4.082$ and $\beta_1 = 0.315$.
- By every additional year of age the estimated expenditure rises by 32 cents.
- Therefore, the expected expenditure of a 30 years old would be $4.082 + 0.315 \cdot 30 = 13.54$ and for a 70 years old $4.082 + 0.315 \cdot 70 = 26.14$ which is approximately the double. (See bellow more explanations about prediction in the regression model)

Illustration: Multiple linear regression

- The simple regression model is obviously very limited, in the example a natural extension would be to introduce the influence of gender into the investigation.
- This is reached by extending the model to a **multiple linear regression**:

$$Exp_i = \beta_0 + \beta_1 \cdot AGE_i + \beta_2 \cdot GENDER_i + \epsilon_i$$

Illustration: Multiple linear regression output (first part)

- To make the model easy to interpret the coding for the gender should be **transformed** into a 0/1-coded variable (0: male, 1: female).

	Estimate	Std. Error	t value	p value
(Intercept)	4.74628	3.52149	1.348	0.19541
AGE	0.33376	0.07427	4.494	0.00032
GENDER	-3.70451	2.83212	-1.308	0.20827

- The second column of the table consists of the parameter estimates for β_0 (2nd row), β_1 (3rd row) and β_2 (4th row).
- As these estimations were estimated based on a sample they underlie a certain insecurity. This insecurity is quantified by the **standard error** (SE) contained in the third column.

Illustration: Interpretation in *ceteris paribus*

Estimated model: $\hat{y}_i = 4.7463 + 0.3338 \cdot x_{i1} - 3.7045 \cdot x_{i2}$

- The estimate $\beta = -3.705$ means that women spend in general 3.70 € less than men in the same age.
- Due to the coding of GENDER (male: GENDER=0), the estimated spending of a man is given as: $\beta_0 \cdot \beta_1 \cdot AGE_i + \beta_2 \cdot 0$.
- For women GENDER takes the value 1 resulting in an estimated expenditure of $AGE_i + \beta_2 \cdot 1$.
- Hence, β_2 are the additional expenditures of a woman compared to a man of similar age.
- If x_{i1} (age in years) is increased by one unit, then y_i (mean spending per person) increases by 0.3338 units.

Linear regression model generalization

This model in its general form with P **covariates** is defined by:

$$Y_i = \beta_0 + \beta_1 \cdot x_{1i} + \beta_2 \cdot x_{2i} + \ldots + \beta_P \cdot x_{Pi} + \epsilon_i \qquad (i = 1, \ldots, n)$$

- In the multiple linear regression it is possible to use metric covariates as well as variables with only nominal or ordinal scale that only have a 0/1 coding.
- Those are called **dummy variables**.
- Nominal or ordinal variables with more than two levels need to be recoded into multiple dummy variables.

Assumptions of the linear regression model

The linear regression model relies on the following **assumptions**, that should be validated:

(i) $\epsilon_1, \ldots, \epsilon_n$ are **normally**[1] distributed random variables with

$$E(\epsilon_i) = 0 \quad \textbf{(Zero mean)}$$

$$V(\epsilon_i) = \sigma^2 \quad \textbf{(Constant variance, homoscedasticity)}$$

(ii) $\epsilon_1, \ldots, \epsilon_n$ are **independent**.

(iii) ϵ_i and $X_{i,j}$ with $j = 1, \ldots, p$ are **uncorrelated**.

[1] The normality is only needed to apply standard tests, for the estimation procedure it is not required

Assumptions of the linear regression model: Verification

- If the metric variables have a **linear relationship** with the response, they can be verified with the help of **scatter plots**. Each one the covariate is plotted against the response.
- The **QQ-plot** should be investigated to check for normality.
- The assumption of homoscedasticity and independent residuals can be checked by a ***residual plot***. This is a scatter plot in which the estimated values of the dependent variable are plotted on the abscissa and the estimated residuals on the ordinate. The points should be scattered **unsystematically**.

Assumptions of the linear regression model: Solutions

- If the there are non-linear associations in the data, then a simple approach is to use non-linear transformations of the predictors, such as $\log X$.
- If a funnel-shaped structure is observed in the residual plot, it is an indication for **heteroscedasticity**. One possible solution is to transform the response using a concave function such as $\log Y$.
- Other **systematics** in the residual plot are often due to violations of the independence assumption.
- An **outlier** is a point for which y_i is far from the value predicted by the model. They can be an incorrect recording of an observation during data collection. In this case, they could be omitted from the analysis.

Illustration: Normality verification

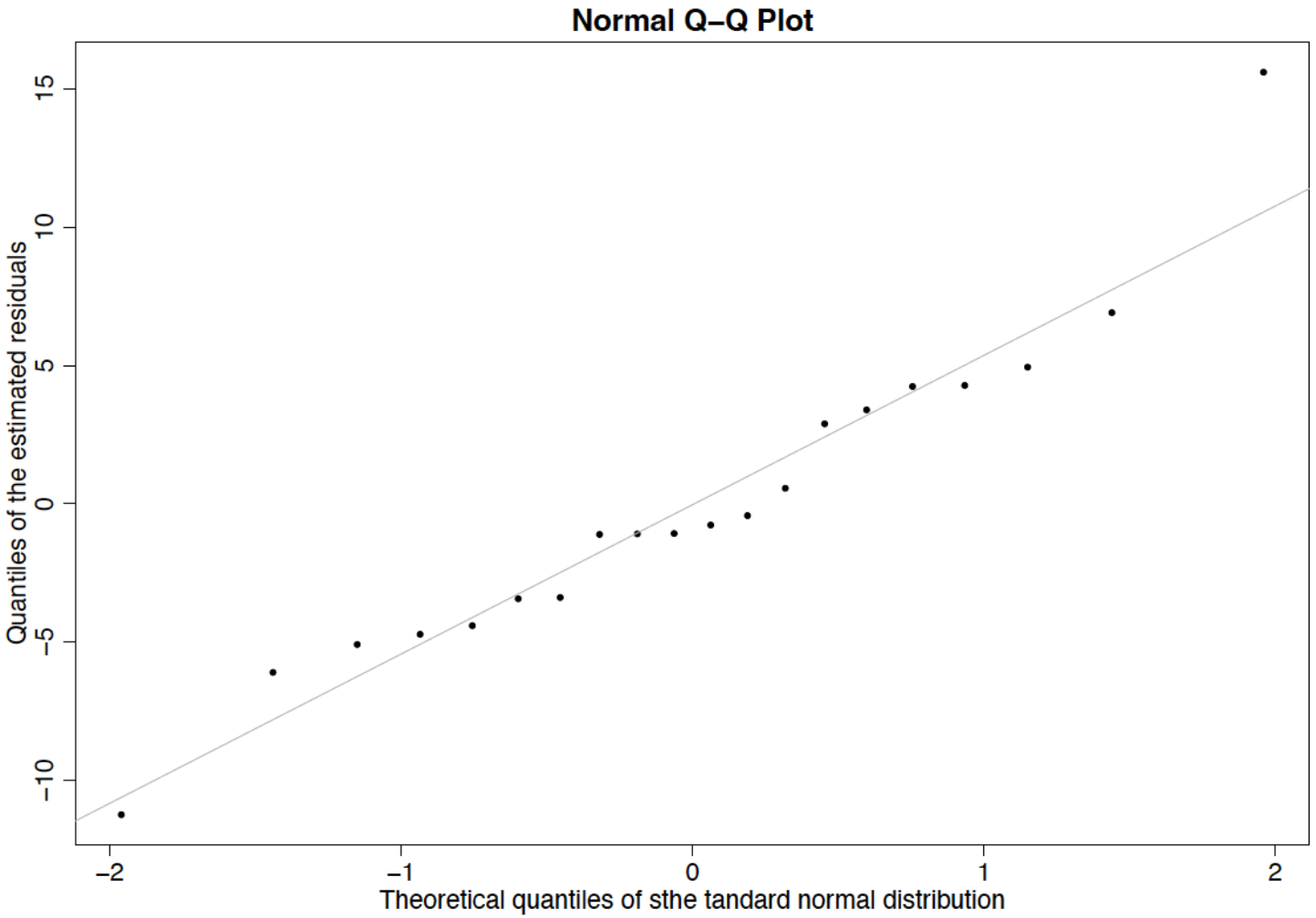
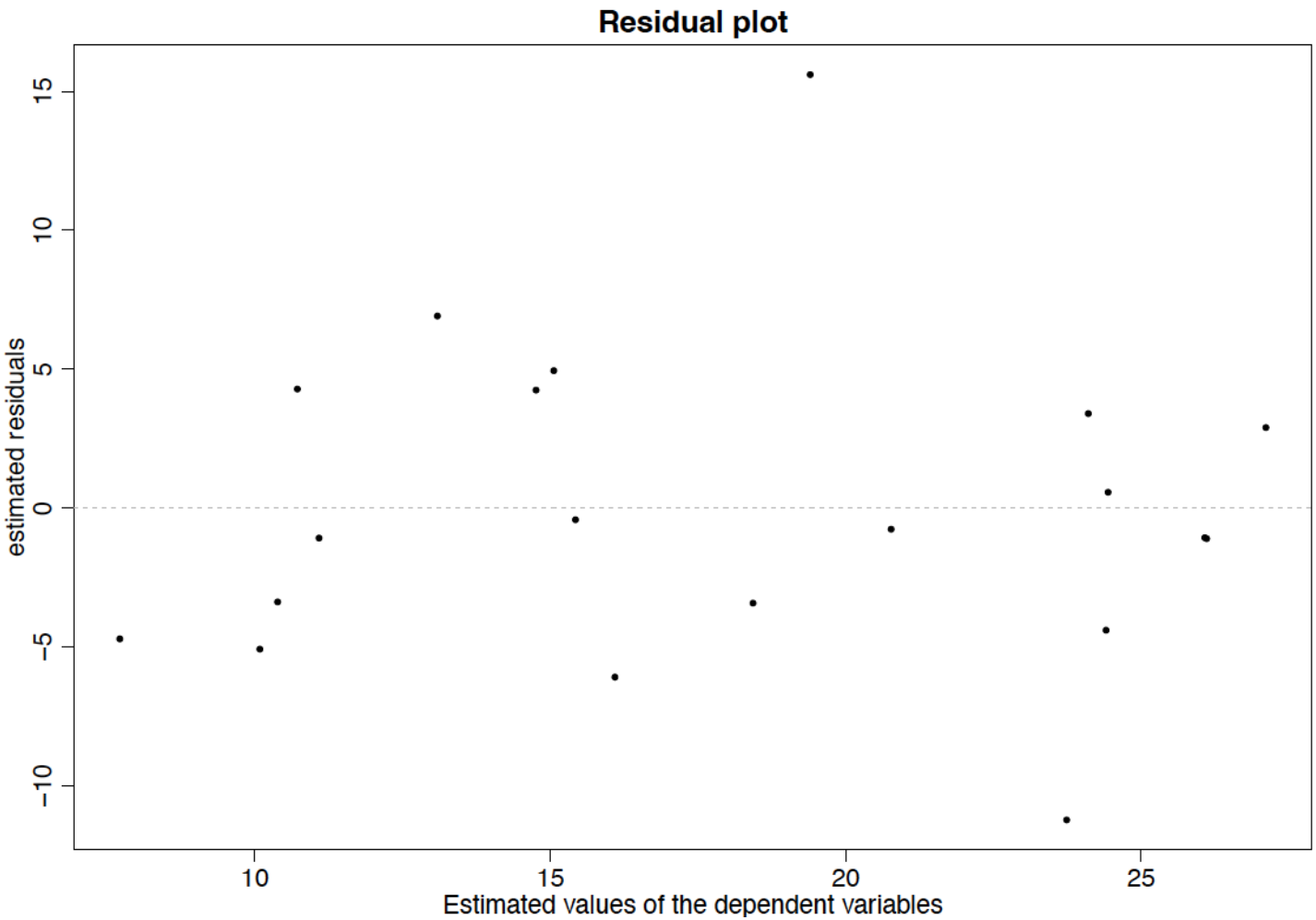


Illustration: Homoscedasticity verification



Is there a relationship between the response and predictors?

- To formally test for this, the "F-test" can be utilized. Its hypothesis is $\beta_1 = \beta_2 = ... = \beta_P = 0$.
- The F-test of overall significance compares the **joint effect** of all the variables together.
- The alternative is that at least one variable is different from 0 -, in other words, at least one variable has a significant influence on the dependent variable.
- Hence, when there is no relationship between the response and predictors, one would expect the F-statistic to be close to 1.

Deciding on important variables (based on testing)

- t-tests allow for testing the hypothesis $\beta_p = 0$ separately.
- If this can not be rejected, the corresponding variable has no significant influence on the dependent variable and can be removed from the model. For this purpose, only the 2-tailed hypotheses are tested so that the p-values can be used.

Illustration: Multiple linear regression output (second part)

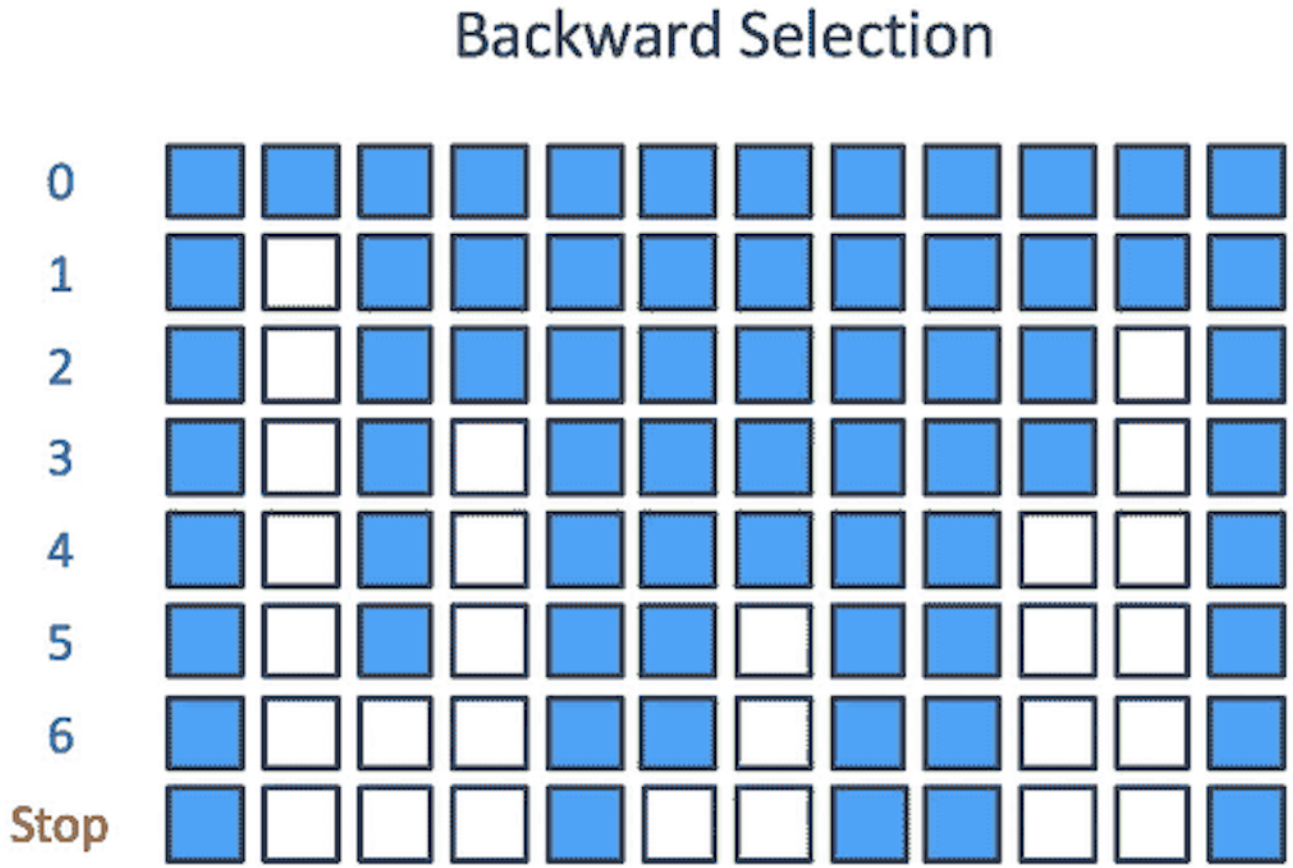
	Estimate	Std. Error	t value	p value
(Intercept)	4.74628	3.52149	1.348	0.19541
AGE	0.33376	0.07427	4.494	0.00032
GENDER	-3.70451	2.83212	-1.308	0.20827

- The p-value in the multiple regression model regarding the overall significance is $0.0005 < \alpha = 0.05$. Therefore, the null hypothesis can be rejected.
- The p-value corresponding to the hypothesis $\beta_2 = 0$ is 0.20827 and is therefore larger than $\alpha = 0.05$.
- Hence, gender should be removed from the model, as it does not have a significant influence on the expenditures.

Deciding on important variables (based on ***variable selection***)

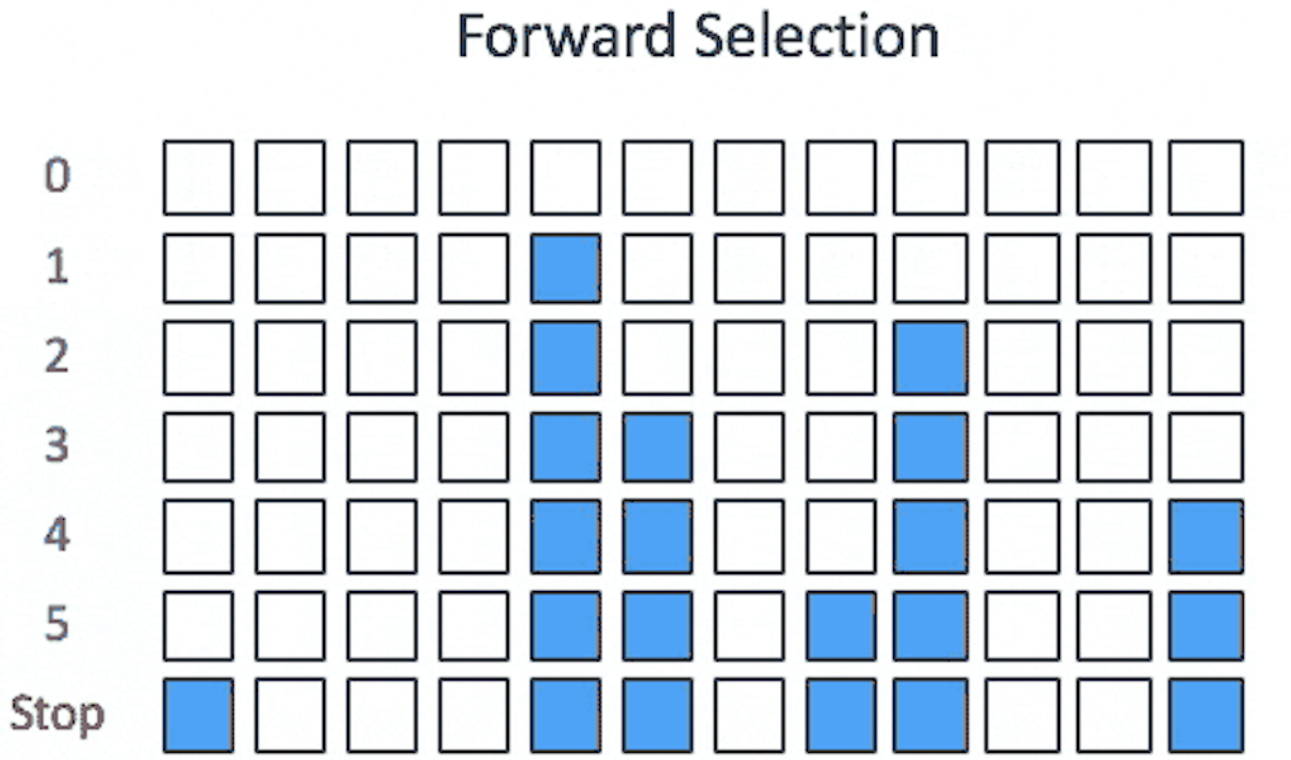
- Using an F-statistic to test for any association between the predictors and the response works when p (number of predictors) is relatively small, and certainly small compared to n (number of individuals).
- However, sometimes we have a very large number of variables.
- If p is large or $p > n$ we are likely to make some **false discoveries**.
- In multiple regression **variable selection** is recommended.
- It refers to determine which predictors are associated with the response, in order to fit a **single model** involving only those predictors.

Backward variable selection



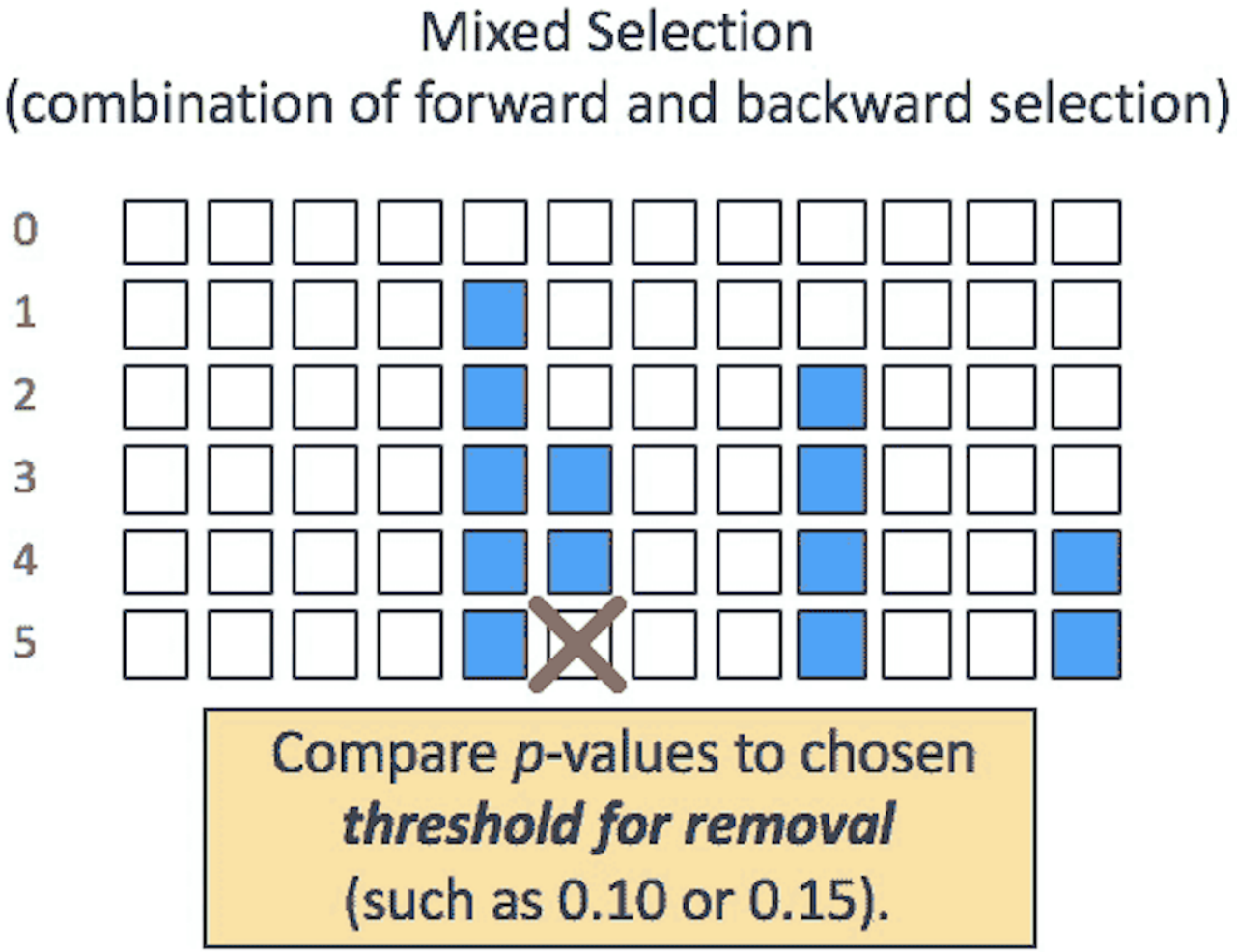
- The first white point is the variable with the highest p-value.

Forward variable selection



- The first blue point is the variable with the lowest p-value.

Stepwise variable selection



Goodness of fit in the regression model

- A popular measure of the **goodness of fit** is the proportion of the variance explained by the model (R^2).
- The value of R^2 **lies between** 0 **and** 1. A value of 0 corresponds to a model that does not explain the dependent variable at all. A value of 1 means that all points lie exactly on the regression line (or (hyper-)plane in the multiple regression model)
- The R^2 is based on the **law of total variance**, which states that the variance of the dependent variable can be decomposed into the sum of the variance that is explained by the residuals and the variance of the residuals (unexplained variance).

The concept behind of R^2

$$R^2 = 1 - \frac{\text{unexplained variance}}{\text{total variance}}$$

$$= \frac{\text{“explained variance”}}{\text{“total variance”}}$$

- By adding more variables to the model the R^2 **never gets worse**.
- To prevent the inflationary adding of useless variables to a model there is the so called “**adjusted** R^2” which penalizes every additional variable. Therefore, the adjusted R^2 only increases if a new variable strongly improves the model.
- For the illustration, the R^2 is 0.5 with one explanatory variable and 0.55 with two explanatory variables. If the R^2 is small, the model might be useless.

Assessing model accuracy: MSE

- Remember that the RSS is defined as $e_1^2 + e_2^2 + \ldots + e_n^2$.
- The **mean square error** (MSE) is a point estimate of the residual variance σ^2 and is defined by:

$$s^2 = \text{MSE} = \frac{RSS}{n - p - 1}$$

- In the regression setting, the MSE is the most commonly-used measure.
- The MSE will be small if the predicted responses are very close to the true responses, and will be large if for some of the observations, the predicted and true responses differ substantially.

Assessing model accuracy

- The **residual standard error** (RSE) is an estimate of the standard deviation of the model error term.
- It is the average amount that the response will deviate from the true regression line defined by:

$$\text{RSE} = \sqrt{\text{MSE}} = \sqrt{\frac{RSS}{n-p-1}}$$

- It is considered a measure of the **lack of fit of** the model. If the predictions obtained using the model are very close to the true outcome values, it will be **small**, and we can conclude that the model fits the data very well.
- If the predictions obtained using the model are one or more observations, then the RSE may be quite **large**, indicating that the model does not fit the data well.

Prediction in the regression model

To predict the values of the outcome variable for a record with predictor values **X**, we use the equation:

$$\hat{y} = X\hat{\beta}.$$

Predictions based on this equation are the best predictions possible in the sense that they will be unbiased and will have the smallest MSE compared to any unbiased estimates under particular assumptions.

Illustration: Predictions

```
New_covariate=list(Age=30, Gender=2)
predict(fit2,New_covariate)
```

- Expected spending of a 30 years old woman:
  $4.7463 + 0.3338 \cdot 30 - 3.7045 \cdot 1 = 11.0558$ Euro.
- Expected spending of a 45 years old man:
  $4.7463 + 0.3338 \cdot 45 - 3.7045 \cdot 0 = 19.7673$ Euro.
- Expected spending of a 70 years old man:
  $4.7463 + 0.3338 \cdot 70 - 3.7045 \cdot 0 = 28.1123$ Euro.

Overfitting and data partitioning

Types of statistical machine learning algorithms (to remember)

Supervised: The algorithm needs that the data scientist acts as a guide to teach the algorithm to which conclusions it should come. It works with explicit inputs and the desired outputs. The y is known and is split into:

- *training* data contain outcomes to train the machine.
- *validation* data are used for select the best performing approach.
- *test* are used for making predictions, which have no outcomes to predict them.

  $\Longrightarrow$ Classification, **regression models**, discriminant analysis, etc.

Unsupervised: The algorithm is able to learn to identify complex structures and patterns of data sets without a data scientist or without using explicitly-provided labels.

Classification vs. Prediction (to remember)

Supervised learning algorithms can be divided into 2 categories: Classification & Prediction (=>Regression)

Classification: Examine data where the classification is unknown, with the goal of predicting what that classification is. Similar data where the classification is known are used to develop rules. Predicts categorical class labels. Examples:

- Recipient of an offer can respond or not respond.
- Bus can be available for service or unavailable.

Prediction: is similar to classification, except that we are trying to predict the value of a numerical variable (e.g., amount of purchase) rather than a class (e.g., purchaser or non-purchaser).

Evaluating predictive performance

Key question: How well will our prediction or classification model perform when we apply it to new data?

*We are particularly interested in comparing the performance of different models so that we can **choose** the one we think will do the best when it is implemented in practice.*

To assure that the chosen model generalizes beyond the current dataset, we

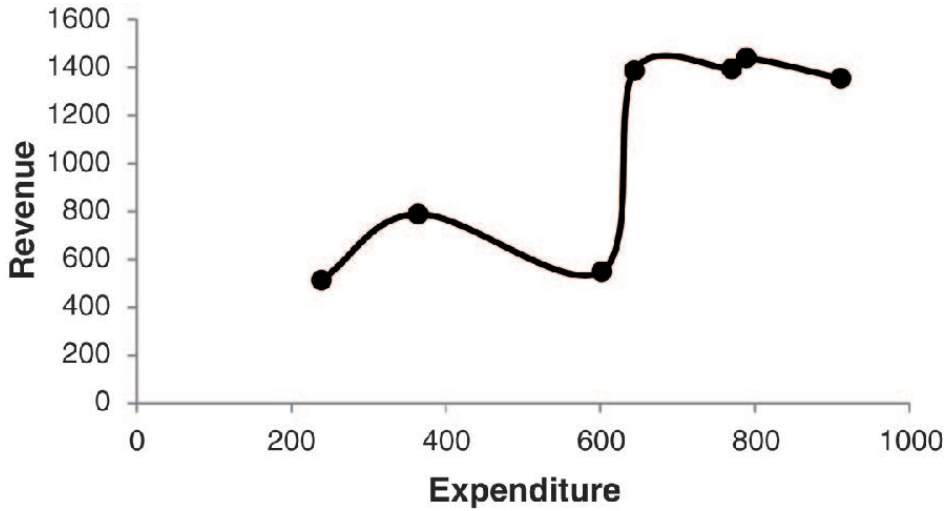
a) use the concept of ***data partitioning*** and

b) try to avoid ***overfitting***.

Overfitting: illustration

The more variables we include in a model, the greater the risk of overfitting the particular data used for modeling. **What is overfitting?**

Example: Advertising expenditures in one period vs sales in a subsequent period

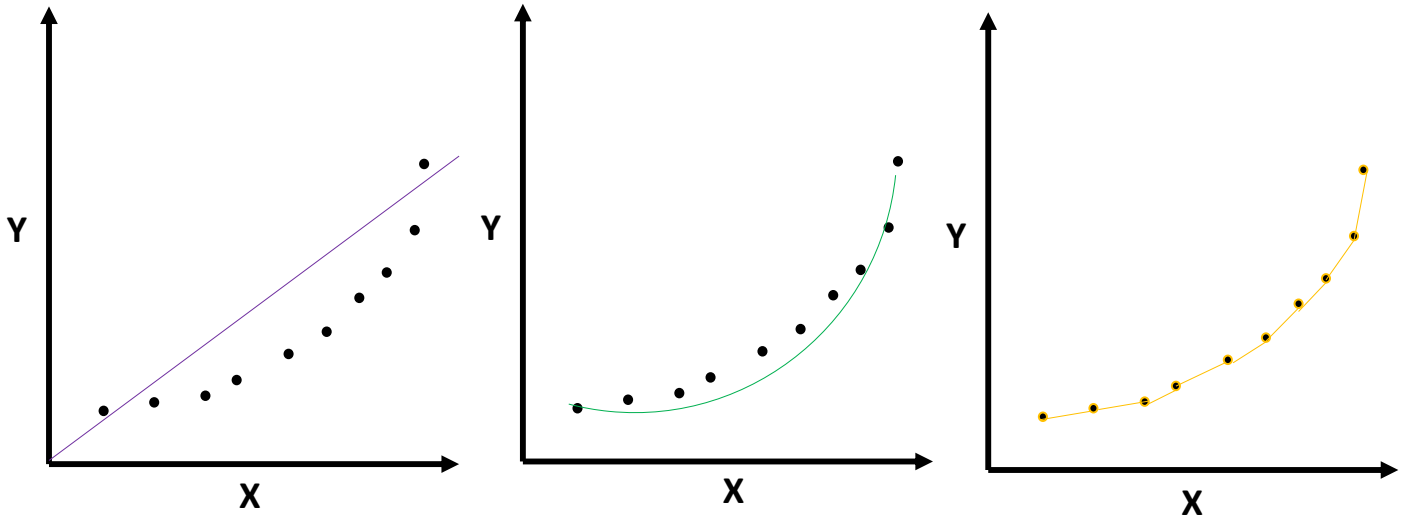


- We could connect points with a smooth **interpolation** - no errors.
- We see that such a curve is **unlikely** to be accurate, or even useful, in predicting future sales.

Overfitting:

- Purpose of building a model is to represent relationships among variables in such a way that this model will do a good job of predicting **future** outcome values based on future predictor values.
- A **simple** straight line might do a better job than the complex function in terms of predicting future sales on the basis of advertising.
- Instead, we devised a **complex** function that fit the data perfectly.
- We **ended up** modeling some variation in the data that is nothing more than chance variation.
- We **mistreated** the noise in the data as if it were a signal.

Overfitting illustration



Linear regression:
- Underfitting
- High bias

Non-linear regression:
- Desired
- Trade-off between variance and bias

Points connection:
- **Overfitting**
- High variance

Overfitting: estimate likely performance

Initial idea:

- Maximizing training accuracy rewards overly complex models.
- We can reach a 100% accuracy but we are not able to generalize well.

Causes:

- The model contains too many predictors (**complexity**).
- The **data set** is too noisy or too small.
- The model has being **refined** over time, but no new data inputs are provided.

Alternative idea:

- We can **split** the initial data set into different sets so that the model can be trained and tested on different data.
- Testing accuracy is a **preferable** than training accuracy.

Creation and use of data partitions

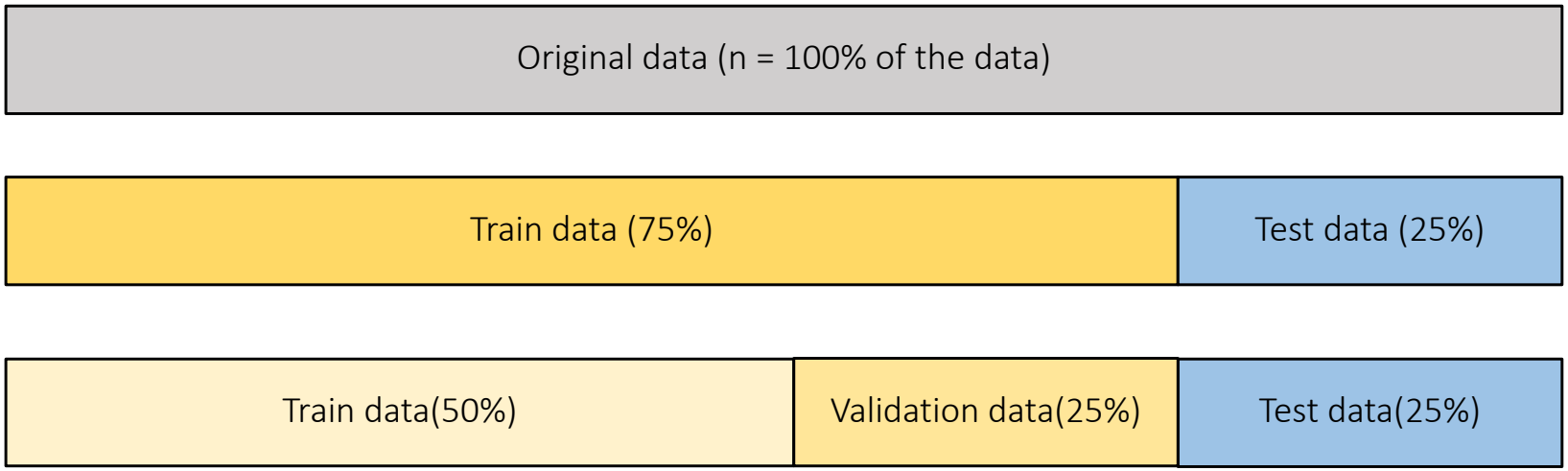
- When we use the same data both to develop the model and to assess its performance, we introduce an ***optimism bias***.
- To address this (overfitting) problem, we simply **partition** our data and develop our model using only one of the partitions.
- After we have selected a model, we try it out on another partition and see how it performs.

Two or three data sets for evaluation

- **Training data**, typically the largest partition, contains the data used to build the various models. The same training partition is generally used to **develop** multiple models.
  Train denotes the set of elements with $|Train| = n_t$.
- **Validation data** is used to **compare** the predictive performance of each model and choose the best one. Sometimes the validation partition may be used in an automated fashion to tune and improve the model.
  Vali denotes the set of elements with $|Vali| = n_v$.
- **Test data** is used to **assess** the performance of the chosen model with new data. *Test* denotes the set of elements with $|Test| = n_{test}$.

Classic partition

In general: $n_t + n_v + n_{test} = n$, where n is the size of the data.



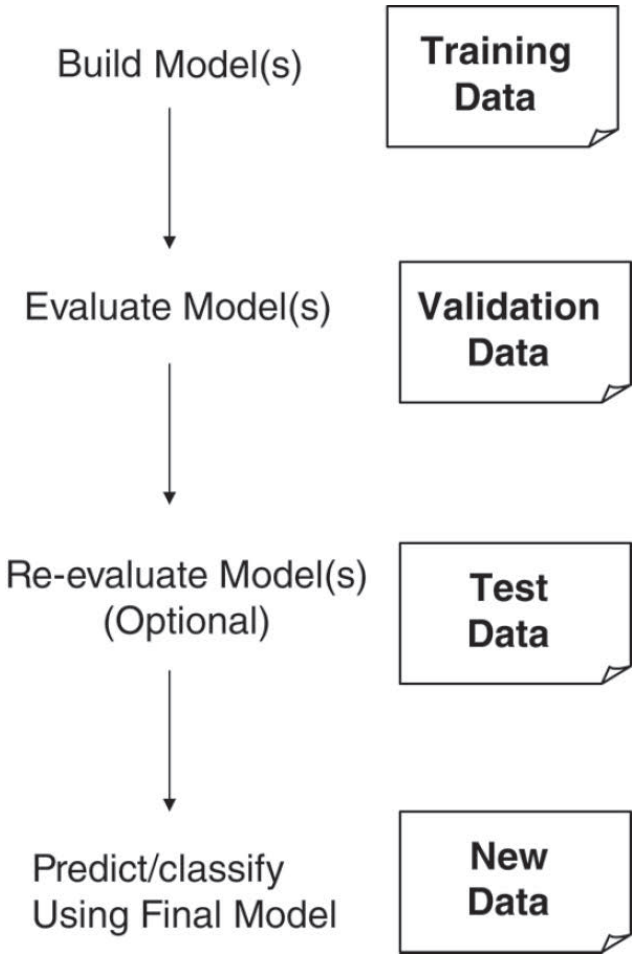
Validation vs. test data sets

Why have both a validation and a test partition?

- We use the validation data to **assess multiple models** and then choose the model that performs best with the validation data.
- The performance of the chosen model on the validation data will be overly **optimistic**.
- Applying the model to the test data, which it has not seen before, will provide **an unbiased estimate** of how well the model will perform with new data.

When we are concerned mainly with **finding** the best model and less with exactly how well it will do, we might use only training and validation partitions.

Data partitions and their role in the data mining process

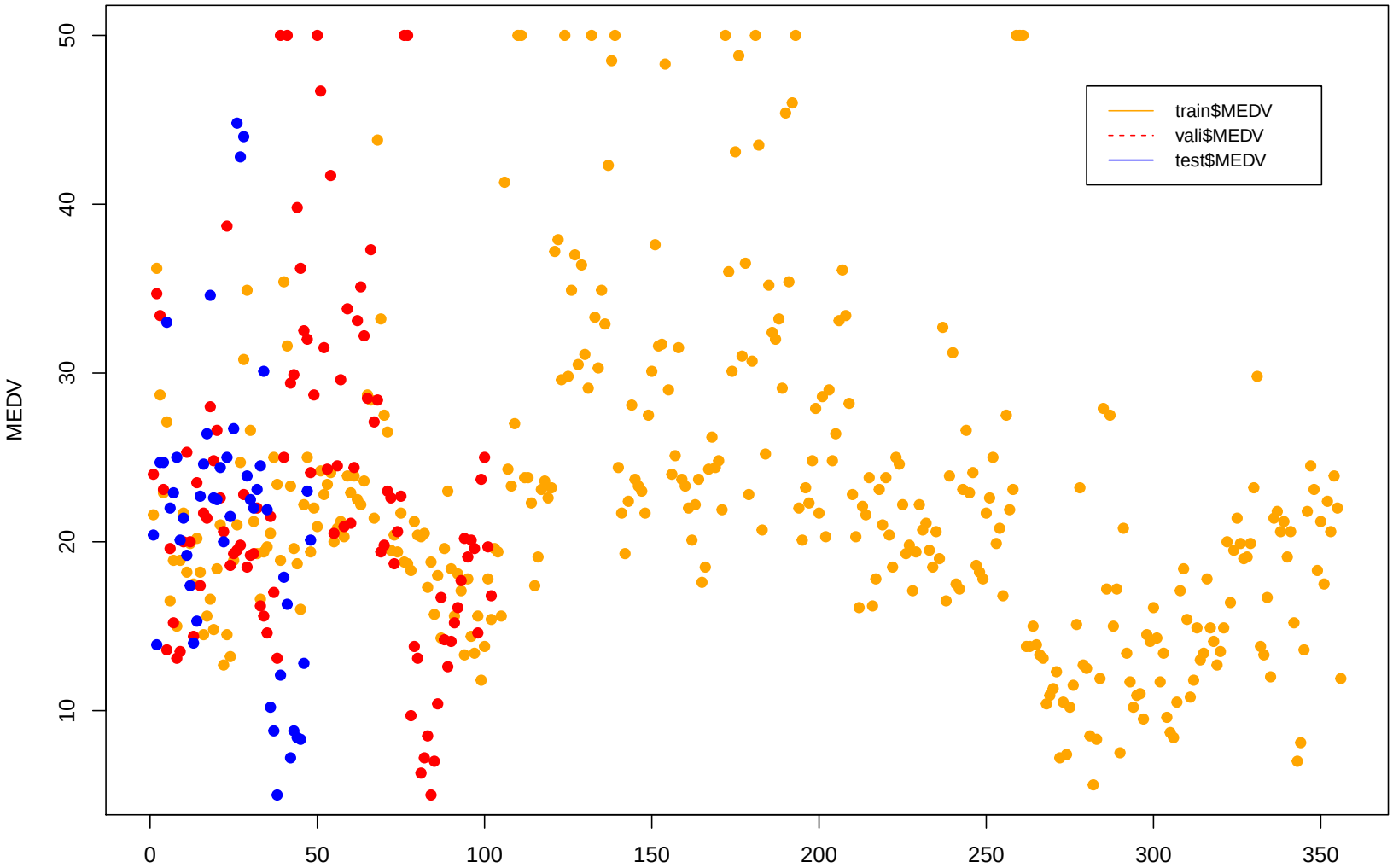


Data partitioning in R

There are several ways to partition the data - one option with the `createDataPartition()`-command:

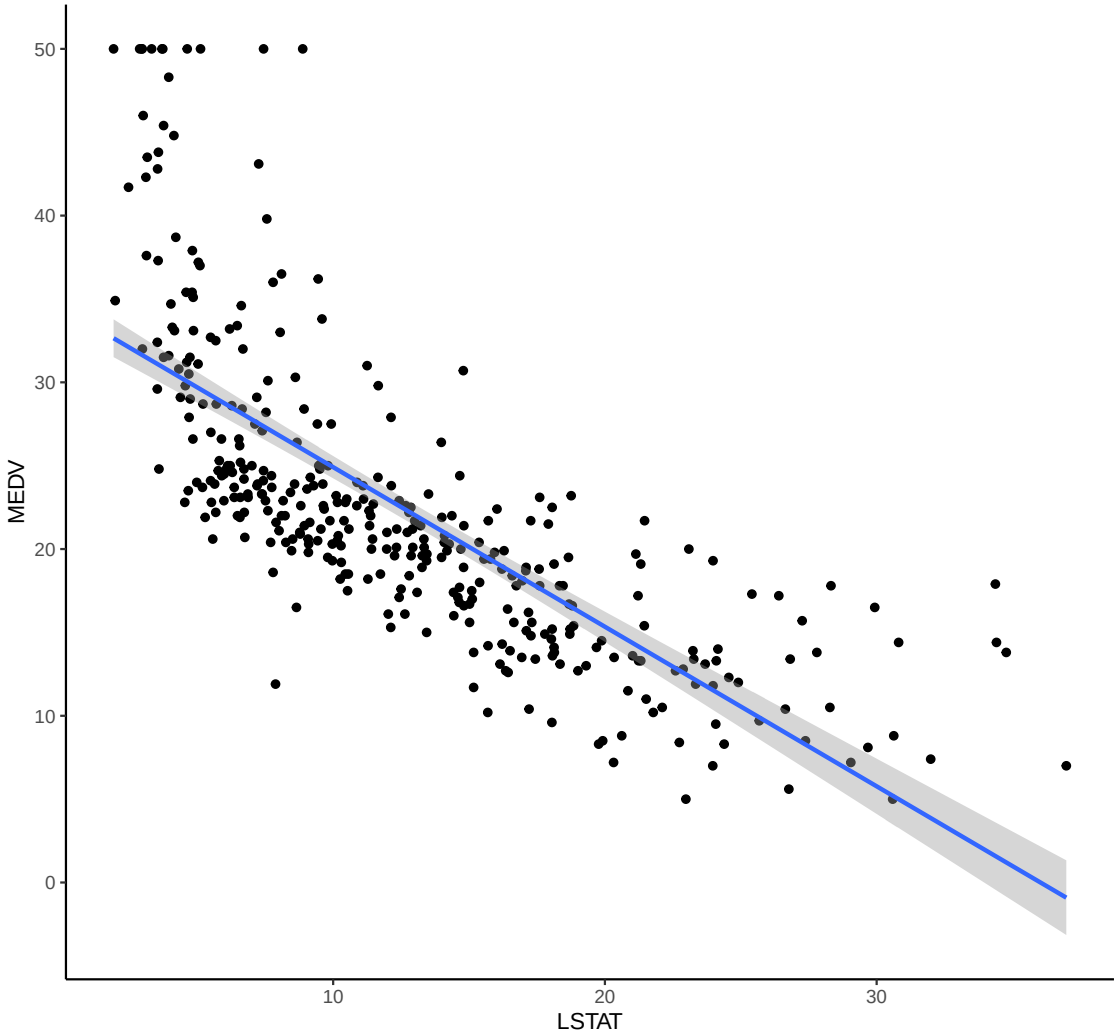
```
> # Loading libraries and the data
> library(caret)
> Daten<-read.csv("BostonHousing.csv")

> # Split data in 70% Training, 20% Validation, 10% Test
> inTrain <- createDataPartition(Daten$CRIM, p = 0.7,
    list = FALSE)
> train <- Daten[inTrain, ]
> inVali <- createDataPartition(Daten$CRIM[-inTrain], p =
    0.666, list = FALSE)
> vali <- Daten[-inTrain,][inVali, ]
> test <- Daten[-inTrain,][-inVali, ]
```

Data partitioning



Polynomial regression & splines

Illustration: Motivation



Polynomial regression

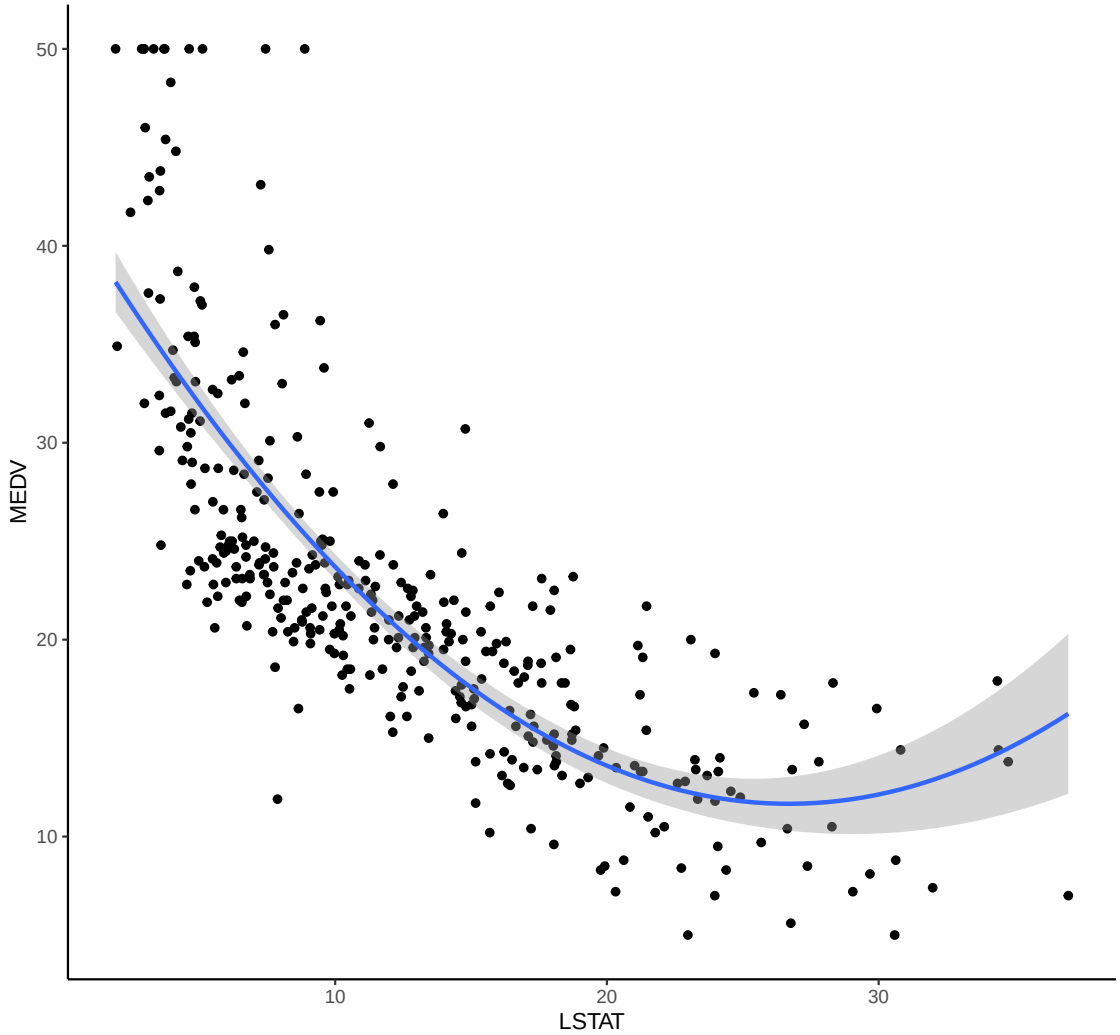
- It **extends** the linear model by adding extra predictors, obtained by raising each of the original predictors to a power.
- For example, a cubic regression uses three variables, X, X^2, and X^3, as predictors.
- This approach provides a simple way to provide a **nonlinear** fit to data.
- It is considered to be a special case of multiple linear regression.
- A polynomial regression may lead to increase in **complexity** as the number of covariates also increases.
- Polynomial models should be hierarchical, containing the terms X, X^2, and X^3, in a hierarchy.

Polynomial regression in R: Boston housing data

We can use the `poly()`-command for specifying a polynomial regression:

```
# To fit a polynomial model
modelfinal <- lm(MEDV ~ poly(LSTAT, 2, raw = TRUE), data
    = train)
# Make predictions
predictions <- modelfinal %>% predict(test)
# Model performance
data.frame(RMSE = RMSE(predictions, test$MEDV),
R2 = R2(predictions, test$MEDV))
# Let's check the curve
ggplot(train, aes(LSTAT, MEDV) ) + geom_point() +
stat_smooth(method = lm, formula = y ~ poly(x, 2, raw =
    TRUE))
# Let's check the assumptions
residuals <- data.frame('Residuals' = modelfinal$
    residuals)
```

Polynomial regression: Boston housing data



Regression splines

- Regression splines is a flexible class of basis functions that extends upon the polynomial regression.
- Instead of fitting a high-degree polynomial over the entire range of X we can make a **zoom** into the data and fit such polynomial there.
- Then, we move to a next small region and fit again such a polynomial.

- As a result, we obtain a connection of these little polynomials so that we end up with a **continuous smooth curve** through the points.
- Smoothing splines is a kind of **piecewise continuous function** composed by those polynomials to model the entire data set.

Regression splines: knots

- It is a way of smoothly interpolating between fixed points, called **knots**.
- The knots can be found throughout cross-validation.
- And after defining the points, the polynomial regression is computed between those knots.
- Regression splines often give superior results to polynomial regression. This is because unlike polynomials, which must use a high degree to produce flexible fits, splines introduce flexibility by increasing the number of knots but keeping the degree fixed.

Smoothing splines: ideas behind

- In statistics and image processing, to **smooth** a data set is to create an approximating function that attempts to capture important patterns in the data, while leaving out noise or other fine-scale structures/rapid phenomena. (Wikipedia)
- Smoothing splines are function estimates obtained from a set of noisy observations in order to balance a measure of goodness of fit with a derivative based measure of the **smoothness**. (Wikipedia)

Smoothing splines: definition

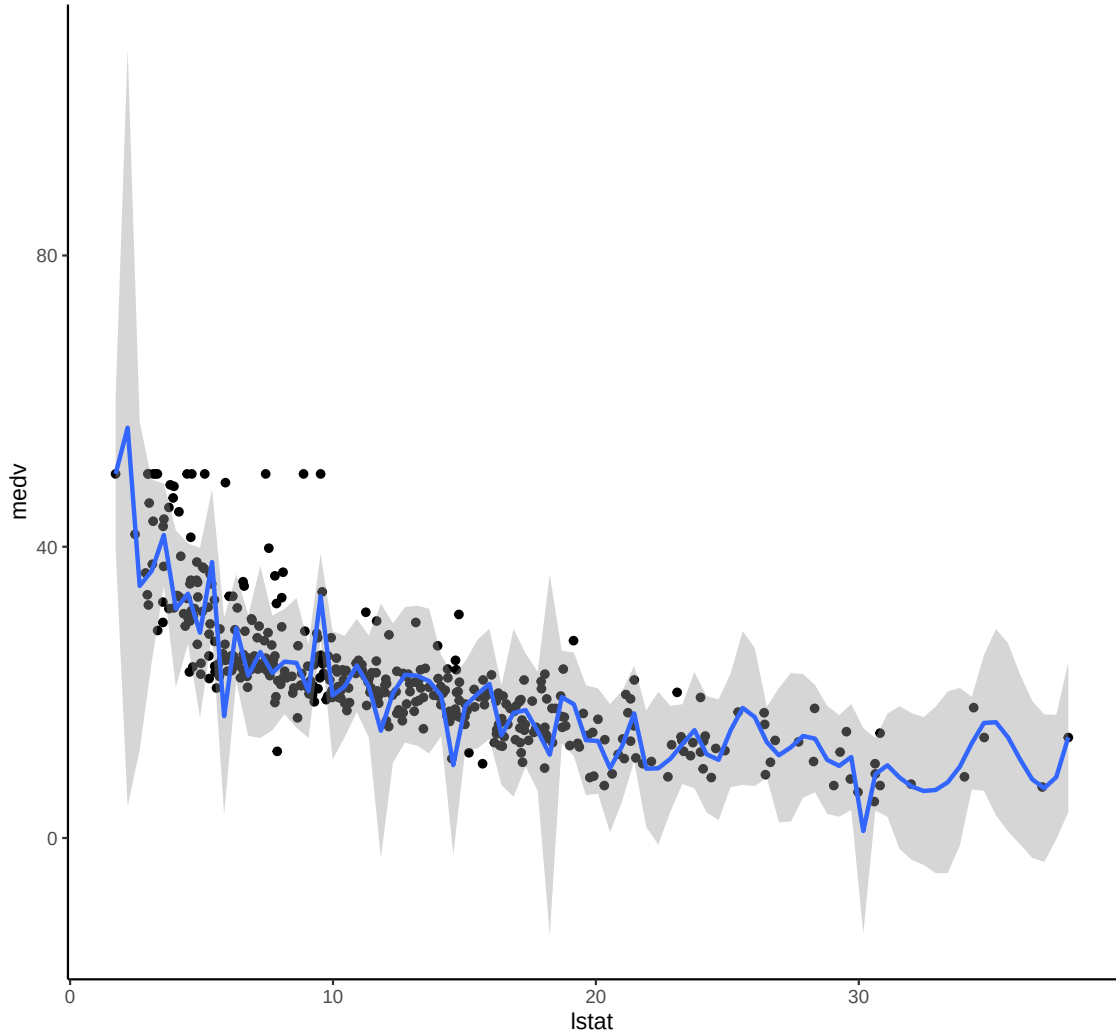
- Smoothing splines are a popular approach for **non-parametric** regression problems.
- The knots and smoothness are controlled via the **tuning parameter** α. This controls the roughness of the smoothing spline, and hence the effective degrees of freedom.
- The knot selection problem completely by using a **maximal set of knots**. The complexity of the fit is controlled by "**regularization**".
- It penalises for the **roughness of the fitting**.
- **Overfitting** disadvantage.

Splines in R: Boston housing data

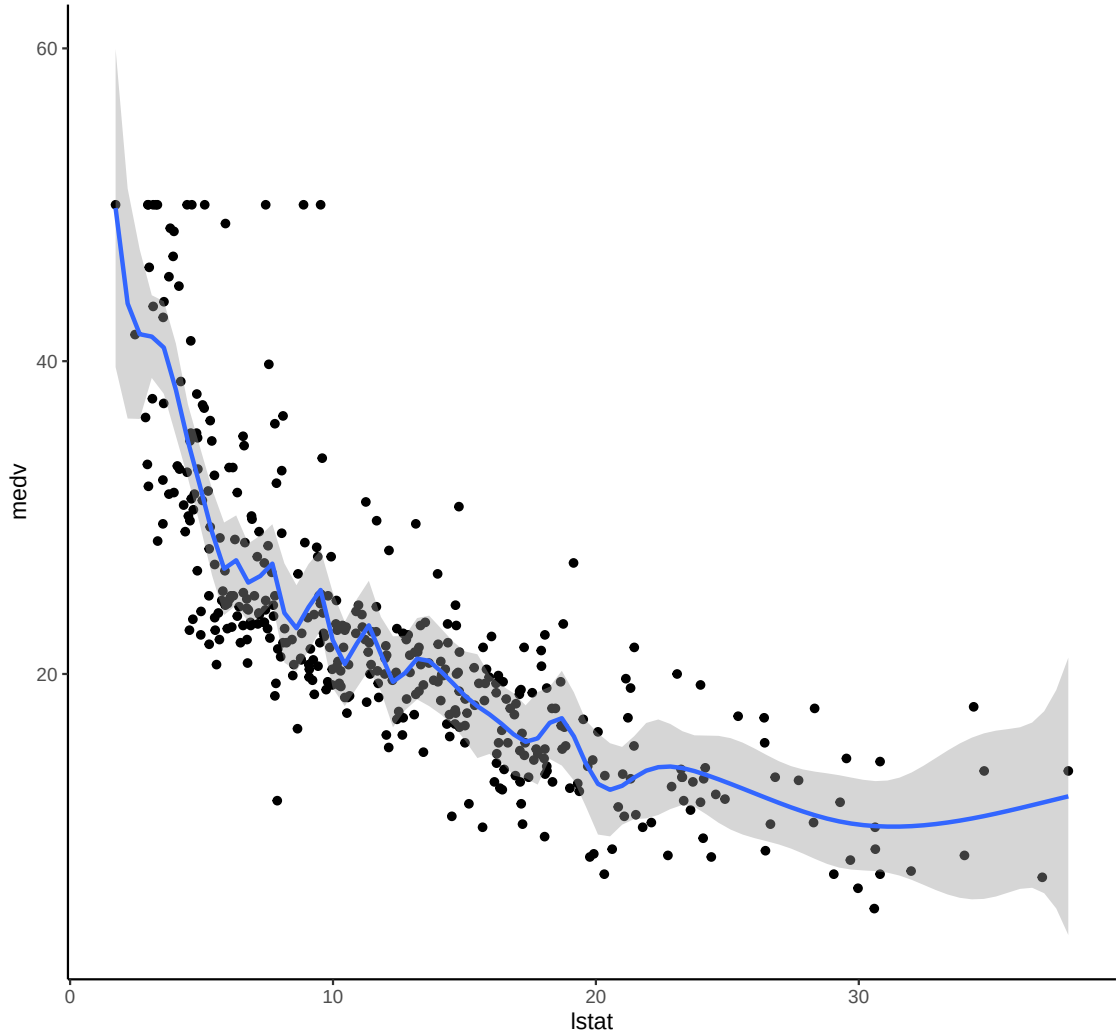
From the library splines we can fit a smoothing splines using the `bs()`-command:

```
# To fit smoothing splines
library(splines)
# Build the model
knots <- quantile(train$LSTAT, p = c(0.25, 0.5, 0.75))
modelss <- lm (MEDV ~ bs(LSTAT, knots = knots), data =
    train)
# Make predictions
predictions <- modelss %>% predict(test)
# Model performance
data.frame(
RMSE = RMSE(predictions, test$MEDV),
R2 = R2(predictions, test$MEDV))
# Let's check the curve
ggplot(train, aes(LSTAT, MEDV) ) + geom_point() +
stat_smooth(method = lm, formula = y ~ splines::bs(x, df
    = 205))
```

Smoothing splines (larger df): Boston housing data



Smoothing splines (smaller df): Boston housing data



Predictive performance

Evaluating predictive performance

Some first comments:

- Predictive accuracy is not the same as goodness-of-fit (R^2).
- Classical statistical measures of performance are aimed at finding a model that fits well to the data on which the model was trained.
- We are interested in models that have high predictive accuracy when applied to new records.
- For assessing prediction performance, we use **several measures**.
- The measures are based on the validation or test set, which serves as a more objective ground than the training set to assess predictive accuracy.

Naive benchmark: The average

- The **benchmark criterion** in prediction is using the average outcome value (thereby ignoring all predictor information).
- The prediction for a new record is simply the average across the outcome values of the records in training set.

$$\hat{y}_j = \frac{1}{n_t}\sum_{i=1}^{n_t} y_i = \overline{y}_t \quad \text{for all } j \in Vali$$

- A good predictive model should outperform the benchmark criterion in terms of predictive accuracy.

A more complex approach: Linear regression

- For making predictions the **multiple linear regression** model is used.
- Two popular but different objectives behind fitting a regression model are:
  - Explaining or **quantifying** the average effect of inputs on an outcome.
  - **Predicting** the outcome value for new records, given their input values.
- In the course, the focus is on predicting new individual records. Here we are not interested in the coefficients themselves, nor in the *average unit*, but rather in the **predictions** that this model can generate for new records.

A more complex approach: Linear regression

Remember that to predict the values of the outcome variable for a record with predictor values **X**, we use the equation:

$$\hat{y} = X\hat{\beta}.$$

Predictions based on this equation are the best predictions possible in the sense that they will be unbiased and will have the smallest MSE compared to any unbiased estimates under particular assumptions.

More prediction accuracy measures: R output

Remember, the **prediction error** for unit i is defined as: $e_i = y_i - \hat{y}_i$.
A few popular numerical measures of predictive accuracy are:

- Mean absolute error $MAE = \frac{1}{n}\sum_{i=1}^{n}|e_i|$.
- Mean error $ME = \frac{1}{n}\sum_{i=1}^{n} e_i$, gives an indication of whether the predictions are on average over- or underpredicting the outcome.
- Mean percentage error $MPE = \frac{100}{n}\sum_{i=1}^{n}\frac{e_i}{y_i}$, gives the percentage score of how predictions deviate from the actual values, taking into account the direction.
- Mean absolute percentage error $MAPE = \frac{100}{n}\sum_{i=1}^{n}|\frac{e_i}{y_i}|$.
- Root mean squared error $RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_i^2}$.

Note that all these measures are influenced by **outliers**. To check outlier influence, we can compute median-based measures or simply plot a histogram or boxplot of the errors.

Prediction errors: Boston housing data

We investigate prediction error metrics from a model for the median value of a home: Training and validation.

```
# Run linear regression model: Median value ~ Crime rate
   + River 1/0 + Number of rooms + Teacher/Pupil ratio
> fit<-lm(MEDV~CRIM+CHAS+RM+PTRATIO,data=train)
> summary(fit)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.28257    4.32789  -0.527  0.59820
CRIM        -0.20104    0.03201  -6.280 8.84e-10 ***
CHAS         2.97189    1.14598   2.593  0.00985 **
RM           7.24178    0.42543  17.022  < 2e-16 ***
PTRATIO     -1.09026    0.14484  -7.527 3.45e-13 ***

Residual standard error: 5.691 on 401 degrees of freedom
Multiple R-squared:  0.6265,  Adjusted R-squared:  0.6228
F-statistic: 168.2 on 4 and 401 DF,  p-value: < 2.2e-16
```

Prediction errors: Boston housing data

The `accuracy()`-command returns the prediction error metrics.

```
# Loading libraries and the data
> library(forecast)

# Create predictions
pred_t_reg<-predict(fit,newdata = train)
pred_v_reg<-predict(fit,newdata = vali)

# Evaluate performance
  # Training - linear regression
>   accuracy(pred_t_reg,train$MEDV)
           ME    RMSE     MAE      MPE     MAPE
Test set    0 5.6556 3.8768 -6.5779 21.0999
  # Validation - linear regression
>   accuracy(pred_v_reg,vali$MEDV)
                ME  RMSE     MAE      MPE     MAPE
Test set -0.2754 6.367 4.0833 -8.3027 21.1321
```

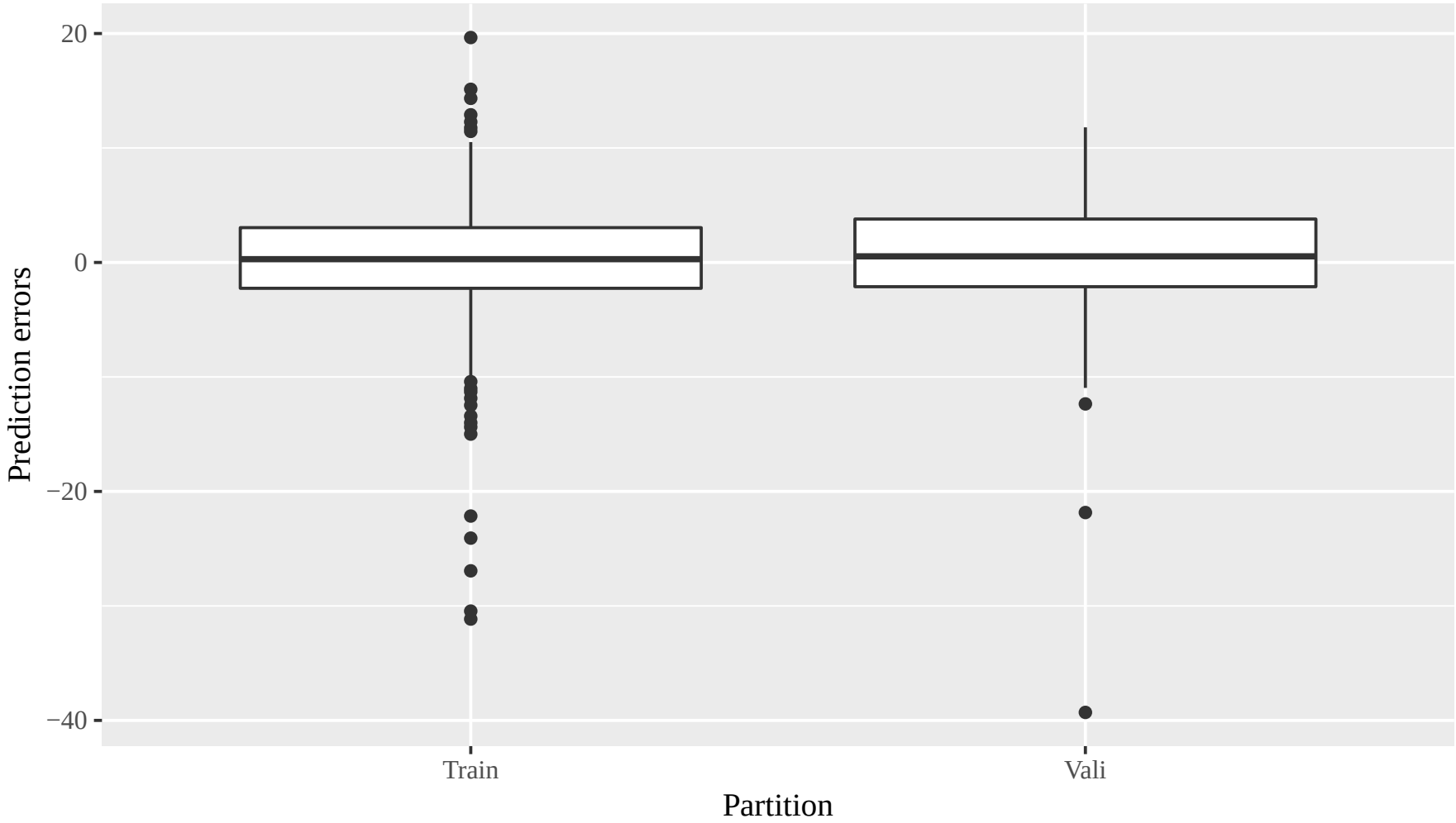
Prediction errors: Boston housing data

Comparing **training and validation performance**:

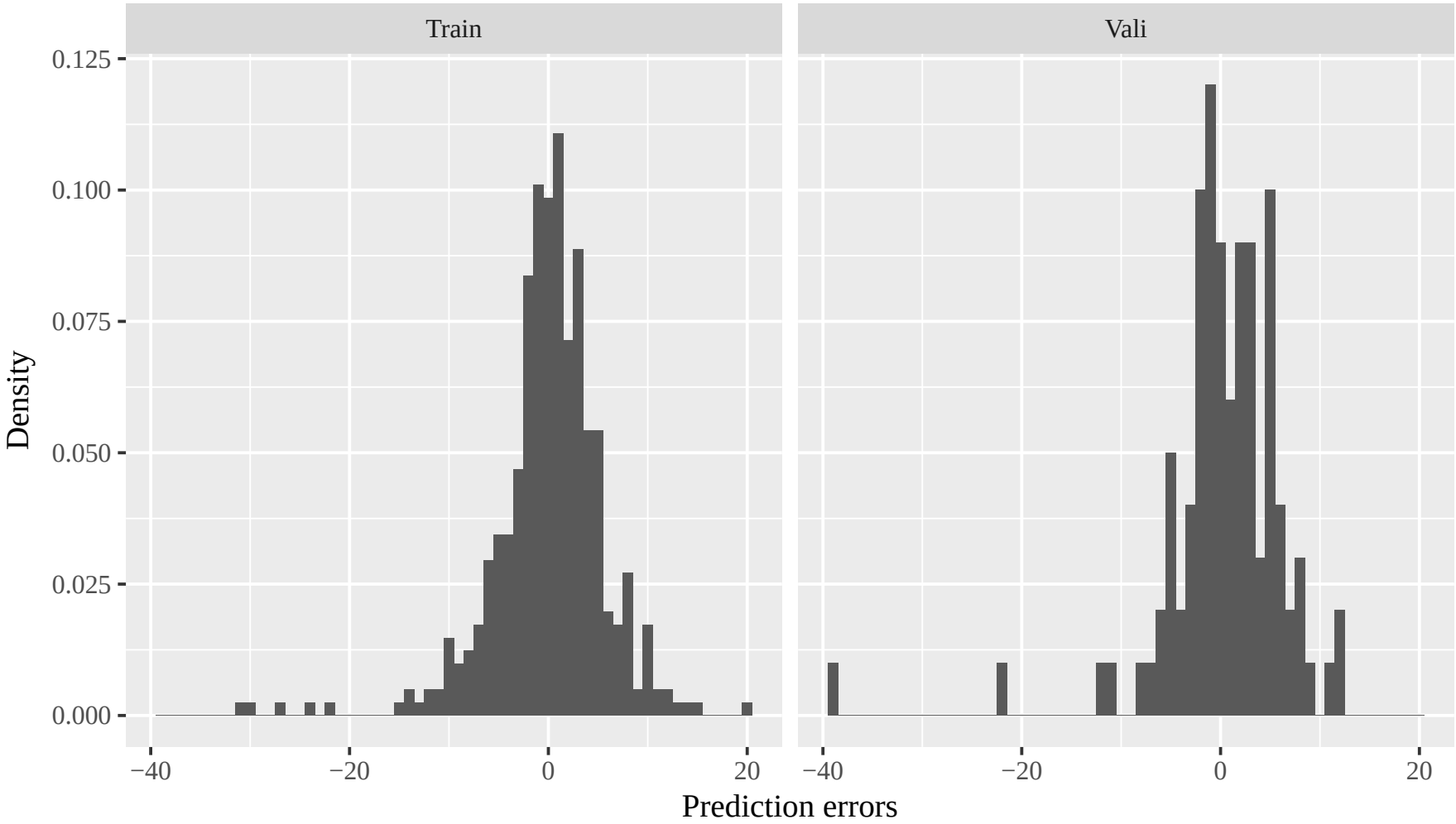
- Errors that are based on the training set tell us about model fit, whereas those that are based on the validation set measure the model's ability to predict new data.
- We expect training errors to be smaller than the validation errors, and the more complex the model, the greater the likelihood that it will overfit the training data.
- In an extreme case of overfitting, the training errors would be zero (perfect fit of the model to the training data), and the validation errors would be non-zero and non-negligible.

For this reason, it is important to compare the error plots and metrics of the training and validation sets.

Prediction errors: Boston housing data



Prediction errors: Boston housing data



Cross-validation

Cross-validation

When the **number of units is small**, data partitioning might not be advisable as each partition will contain too few records for model building and performance evaluation.

One alternative to data partitioning is cross-validation:

- **Cross-validation** (CV) is a procedure that starts with partitioning the data into *folds,* or non-overlapping subsamples.
- Each time, one of the **folds** is used as the validation set and the remaining $k - 1$ folds serve as the training set.
- Combine the model's predictions on each of the k validation sets in order to evaluate the overall performance of the model.

Cross-validation is also used for choosing/ tuning the parameters in algorithms.

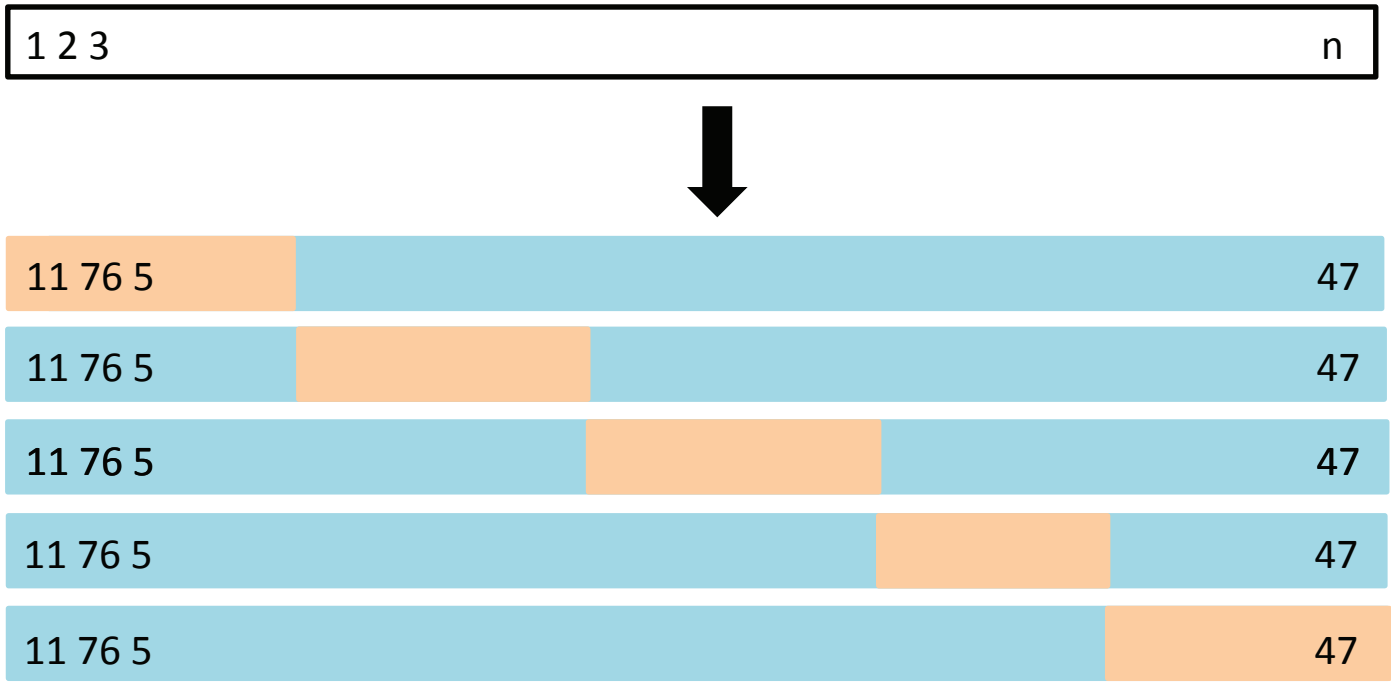
k-fold cross-validation

- Approach involves randomly k-**fold CV** dividing the set of observations into k groups, or folds, of approximately equal size.
- The first fold is treated as a validation set, and the method is fit on the remaining $k-1$ folds.
- The mean squared error, MSE_1, is then computed on the observations in the held-out fold.
- This procedure is repeated k times; each time, a different group of observations is treated as a validation set.
- The k-fold CV estimate is computed by averaging these values

$$CV_{(k)} = \frac{1}{k}\sum_{i=1}^{k} MSE_i.$$

- In practice, one typically performs k-fold CV using $k = 5$ or $k = 10$.

5-fold cross-validation: A schematic display



A set of n observations is randomly split into 5 non-overlapping groups. Each of these fifths acts as a validation set, and the remainder as a training set. The test error is estimated by averaging the 5 resulting MSE estimates.

k-fold cross-validation

What is the advantage of using $k = 5$ or $k = 10$ rather than $k = n$?

- **Leave-one-out cross-validation (LOOCV)** with $k = n$ requires fitting the statistical learning method n times - computational expensive.
- k-fold CV gives more accurate estimates of the test error rate than does LOOCV.

Bias-variance trade-off for k-fold cross-validation:

- LOOCV gives approximately unbiased estimates of the test error, since each training set contains $n - 1$ observations, which is almost as many as the number of observations in the full data set.
- LOOCV has much higher variance than does k-fold CV with $k < n$.
- LOOCV averages outputs of n fitted models on an almost identical data - outputs are highly (positively) correlated.
- Since the mean of highly correlated quantities has higher variance than does the mean of quantities that are not as highly correlated, the test error estimate (LOOCV) has higher variance than the one (k-fold CV).

True and estimated test MSE: Two goals

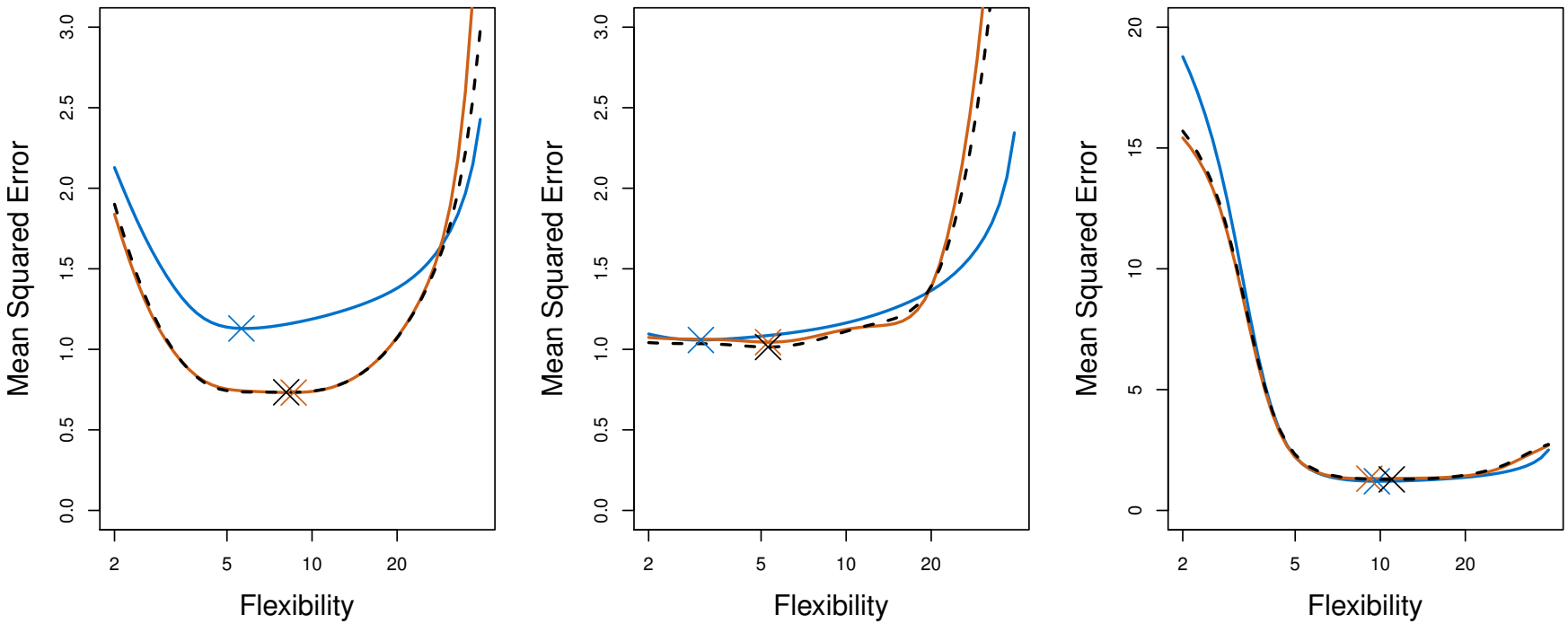
First goal:

- ... might be to determine how well a given statistical learning procedure can be expected to perform on independent data - the actual estimate of the test MSE is of interest.

Second goal:

- ... might be to determine the location of the minimum point in the estimated test MSE curve.
- We perform CV on a number of statistical learning methods, or on a single method using different levels of flexibility to identify the method that results in the lowest test error.
- Location of the minimum point in the estimated test MSE curve is important, but the actual value of the estimated test MSE is not.

True and estimated test MSE: Two goals



True and estimated test MSE for the simulated data sets. The true test MSE is shown in blue, the LOOCV estimate is shown as a black dashed line, and the 10-fold CV estimate is shown in orange. The crosses indicate the minimum of each of the MSE curves.

Cross-validation: Boston housing data

Cross-validation can be automatically computed for any generalized linear model using the `cv.glm()`-command.

```
> # Run linear regression model: Median value ~ Crime
    rate + Number of rooms + Teacher/Pupil ratio
> fit<-glm(MEDV~CRIM+RM+PTRATIO,data=Daten)

> # Leave-one-out cross-validation
> cv_one_err<-cv.glm(Daten,fit)
> cv_one_err$delta
[1] 35.00064 34.99989

> # 5-fold cross-validation
> cv_5_err<-cv.glm(Daten,fit,K=5)
> cv_5_err$delta
[1] 34.98018 34.89865
```

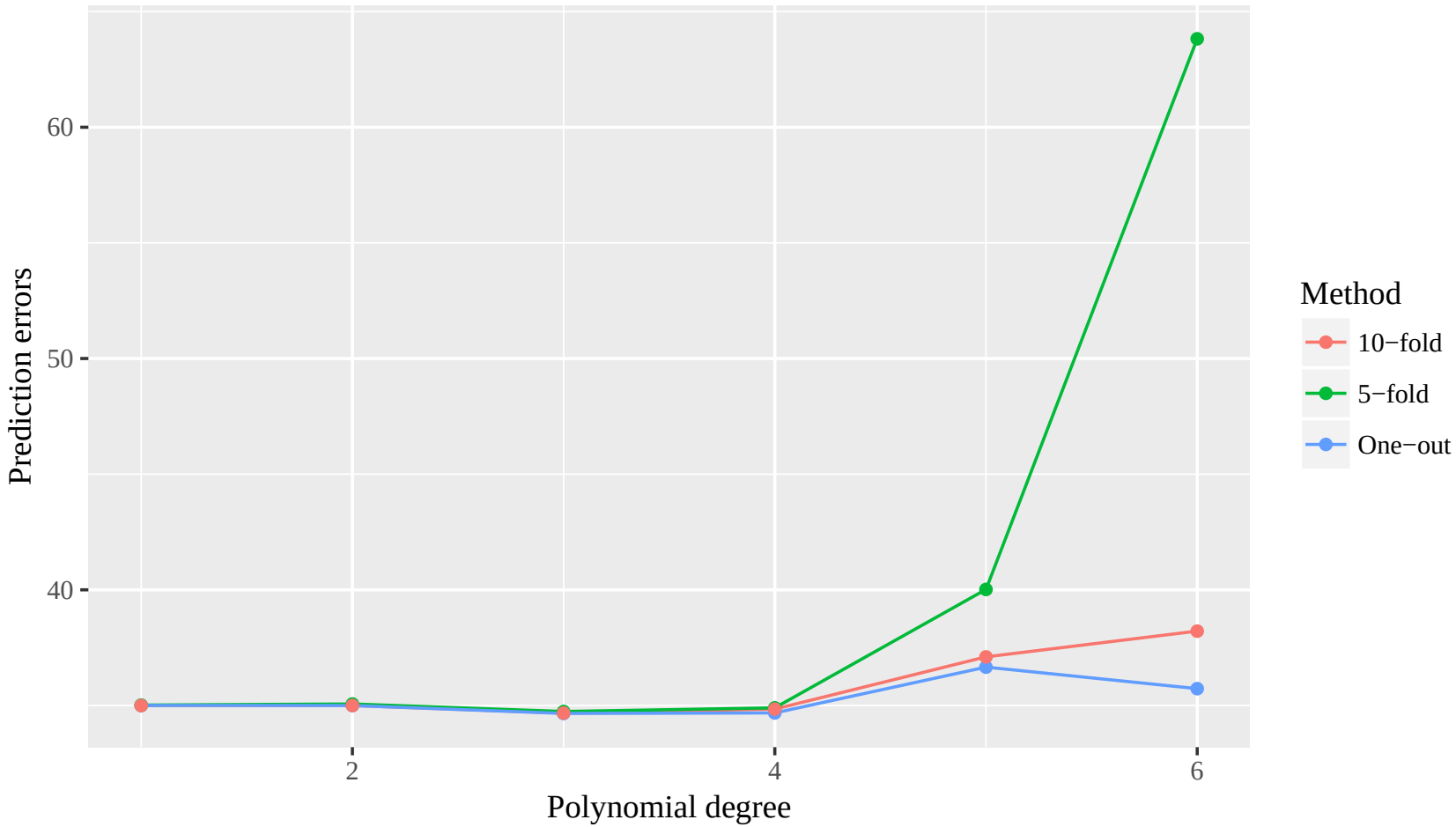
Cross-validation: Boston housing data

We investigate prediction error metrics from a model for the median value of a home: Linear vs. polynomial model for variable *Crime rate*.

```
> cv_error<-NULL
>
> for(i in 1:6){
+   fit_poly<-glm(MEDV~poly(CRIM,degree=i)+RM+PTRATIO,
    data=Daten)
+   cv_error[i]<-cv.glm(Daten,fit_poly,K=10)$delta[1]
+ }
> cv_error
[1] 34.708 34.813 35.076 34.423 41.874 51.530
```

The results don't show a clear improvement from using higher-order (> 4) polynomials. However, results may depend on the random split.

Cross-validation: Boston housing data



Each CV approach was run 100 separate times, each with a different random split of the data into K parts. The figure shows the median prediction error over replications.

Penalized regression

Example of Failure? Linear regression on the Flights Dataset

Linear Regression on a subsample (**n=9787**) of the Flights dataset with variables:

- Response variable: Arrival Delay
- Explanatory variables: Month, DayofMonth, DayOfWeek, DepTime, CRSDepTime, ArrTime, CRSArrTime, UniqueCarrier, FlightNum, TailNum, ActualElapsedTime, CRSElapsedTime, AirTime, DepDelay, Distance, TaxiIn, TaxiOut

A total of **9723 variables**, including the necessary dummy variables.

Results of the Analysis

254 variables are reported as significant (p<0.05).

	Coef	p	Name
1	-3.586672e-16	1.401054e-02	DepTime
2	4.370166e-16	2.851509e-03	CRSDepTime
3	1.709959e-16	3.341626e-02	ArrTime
4	-4.057648e-16	1.100155e-05	CRSArrTime
5	-2.059659e-12	2.852775e-02	FlightNum5
6	-1.990751e-12	3.064923e-02	FlightNum8
..	...	...	...

- The coefficients are **negligible**.
- Adding more observations would probably produce more significant coefficients. Would that be helpful?
- With fewer observations we would have had $n < p$ and not been given **any estimates**.
- **p-values, R^2, etc. are not good measures** of model fit anymore.

Penalized Regression: idea behind

- They are also known as shrinkage methods or regularization models.
- We would like to continue using linear regression models, but we need to **adjust** them to be usable with big or high-dimensional datasets.
- We introduce a **penalty** for too many or too large coefficients.
- We can fit a model containing all p predictors using a technique that **constrains** or **regularizes** the coefficient estimates, or equivalently, that **shrinks** the coefficient estimates towards zero.

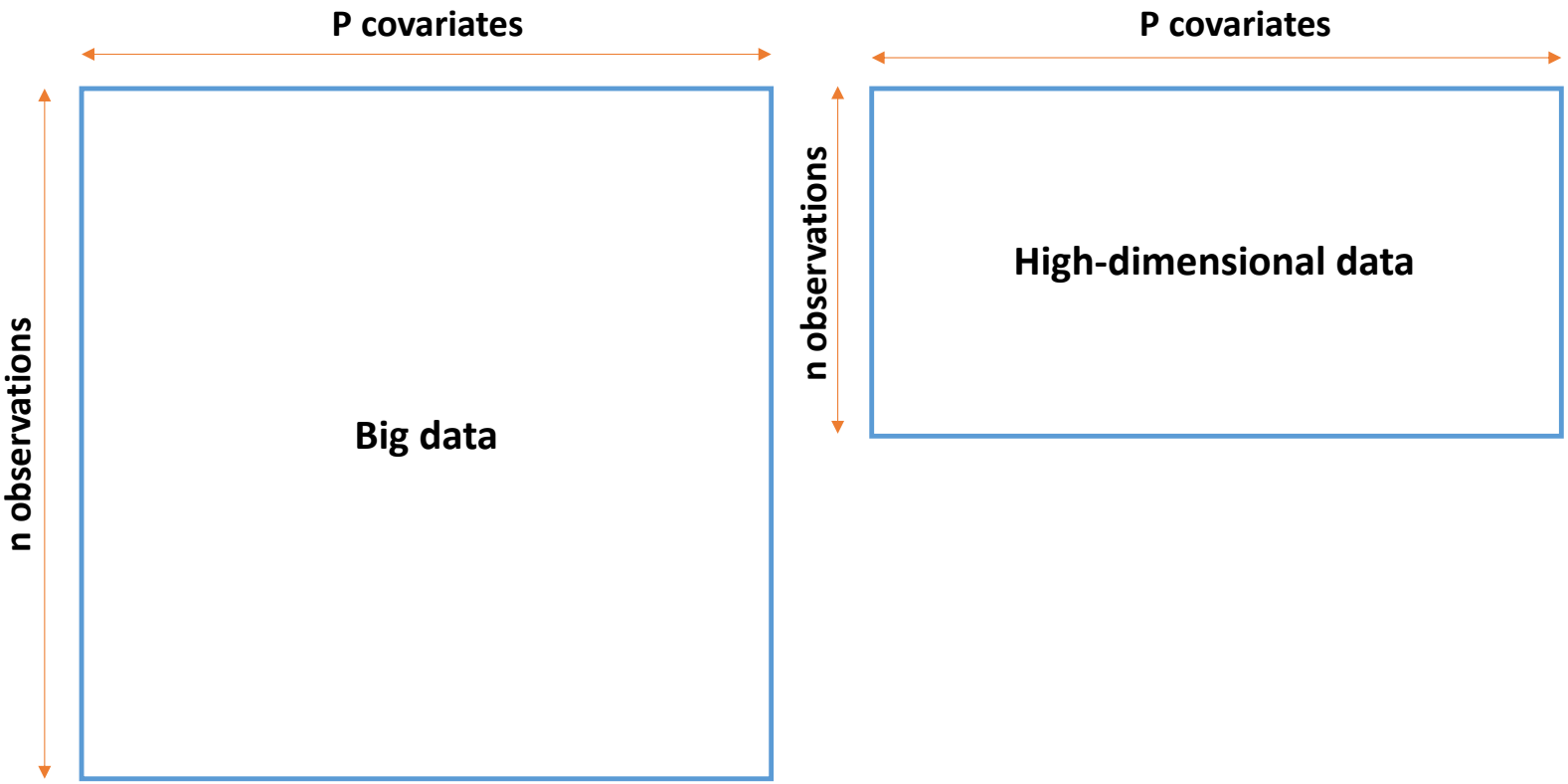
Penalized regression: when?

Regularization methods can be used when at least one of the following conditions is met

- large number of variables
- more variables than observations $n \ll p$
- strong multicollinearity
- a sparse solution is wanted/needed (feature selection)
- "The word 'high-dimensional' refers to a situation where the number of unknown parameters which are to be estimated is one or several orders of magnitude larger than the number of samples in the data."[2]

[2] Peter Bühlmann, Sara van de Geer - Statistics for High-Dimensional Data, Springer 2011

Big data vs. high-dimensional data



Examples of high-dimensional data

Typically, high-dimensional data arise in a number of settings:

- genomics (microarrays, proteomics)
- signal processing
- image analysis
- market basket data and portfolio allocation
- industry (3d-printing)

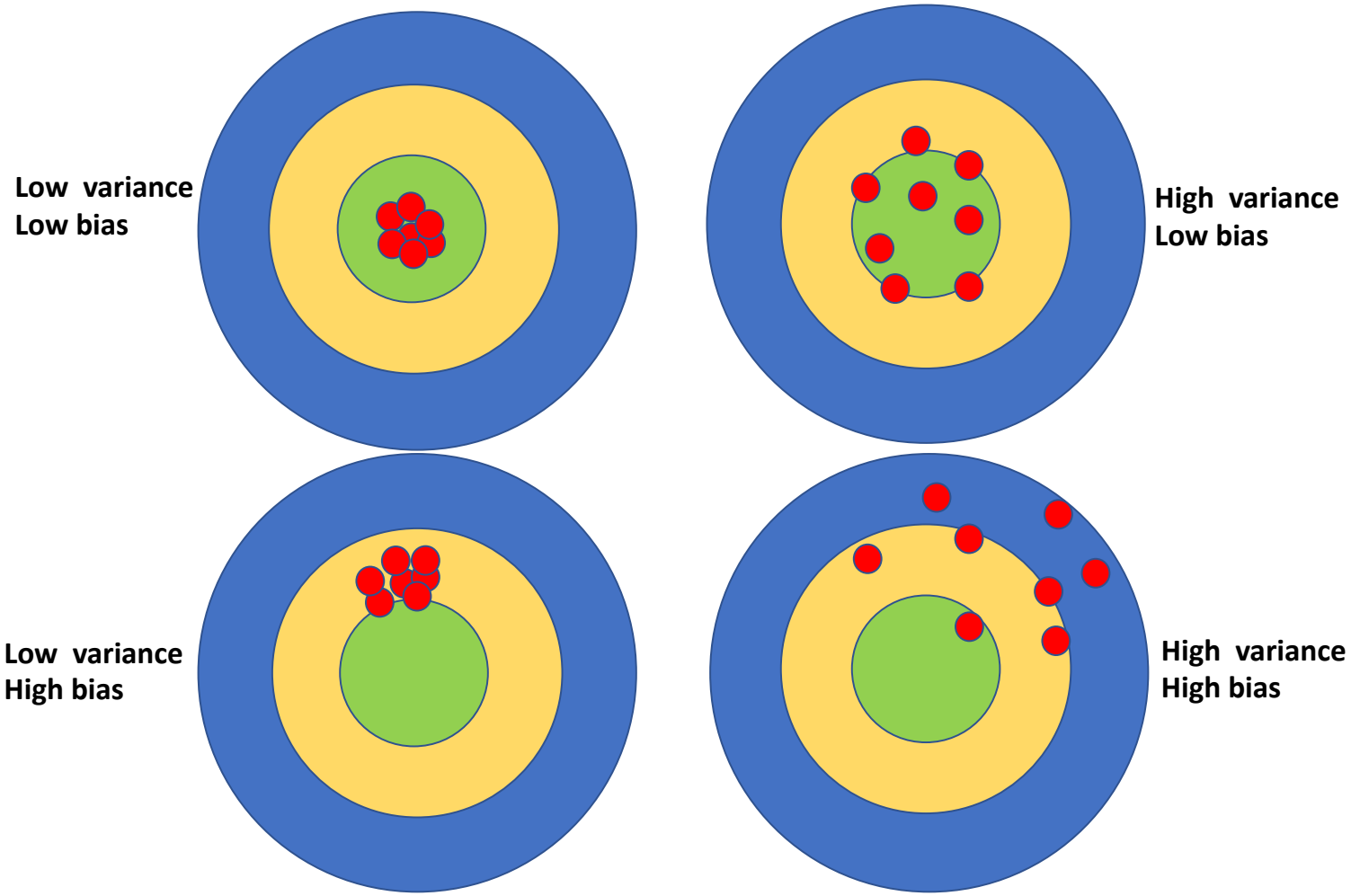
MSE of a predictor: remeber

We use the MSE together with cross validation to assess our model fit.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{Y}_i - Y_i\right)^2$$

Or more exactly, the mean squared prediction error.

Bias-variance trade-off

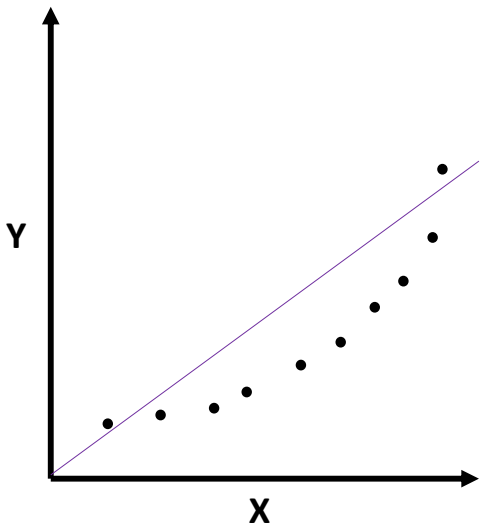


MSE of an estimator

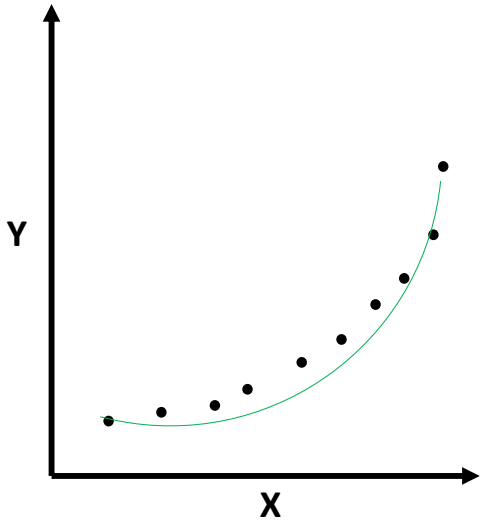
The MSE of an estimator can also be written as the sum of the bias squared and the variance

$$\begin{aligned}\text{MSE}\left(\hat{\theta}\right) &:= \mathbb{E}\left[(\hat{\theta}-\theta)^2\right] \\ &= \mathbb{E}\left[\left(\hat{\theta}-\mathbb{E}[\hat{\theta}]\right)^2\right] + \left(\mathbb{E}[\hat{\theta}]-\theta\right)^2 \\ &= \mathit{Var}\left(\hat{\theta}\right) + \mathit{Bias}\left(\hat{\theta},\theta\right)^2.\end{aligned}$$

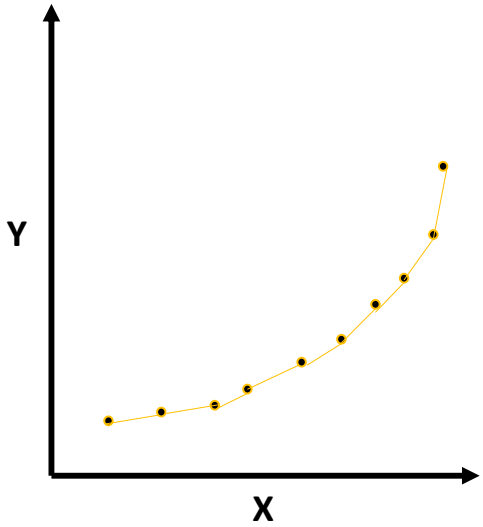
Why penalized regression?



Linear regression:
- High bias
- Small variance



Polynomial regression:
- Trade-off between variance and bias (moderate bias and variance)



Splines:
- Small bias
- High variance

Deviance

- Our **second criterion** for assessing model fit will be the deviance ratio.
- The deviance for a model is defined as

$$dev = -2 \ln \frac{\text{likelihood of the fitted model}}{\text{likelihood of the saturated model}},$$

where the saturated model has one parameter per observation.

- The deviance ratio can then be defined as

$$devratio = 1 - \frac{dev}{nulldev},$$

where *nulldev* is the deviance of the null model, the model with only an intercept.

- **Think of the deviance ratio as the R^2 from simple linear regression.**

Illustration generalities: sample from Airline data

- We will use a sample (n = 39153) of the Flights 2008 dataset.
- The models will be fit to a subsample of size 27594 and evaluated on a test set of size 11559.
  - Response variable: Arrival Delay
  - Explanatory variables: Month, DayofMonth, DayOfWeek, DepTime, CRSDepTime, ArrTime, CRSArrTime, UniqueCarrier, FlightNum, TailNum, ActualElapsedTime, CRSElapsedTime, AirTime, DepDelay, Distance, TaxiIn, TaxiOut
- We will be working on normalized Data.

Lasso regression

Lasso regression

Least **A**bsolute **S**hrinkage and **S**election **O**perator introduced by Tibshirani in 1996.

Advantages:

- statistical accuracy in prediction
- variable selection
- computational feasibility

Does not perform well, when groups with high collinearity are present.

Definition of lasso regression

The lasso estimate is defined by

$$\hat{\beta}^{lasso} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2$$

$$\text{subject to } \sum_{j=1}^{p} |\beta_j| \leq t.$$

Or equivalently in *Lagrangian form*

$$\hat{\beta}^{lasso} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\},$$

with $\lambda \geq 0$.

Example: 2008 Flights

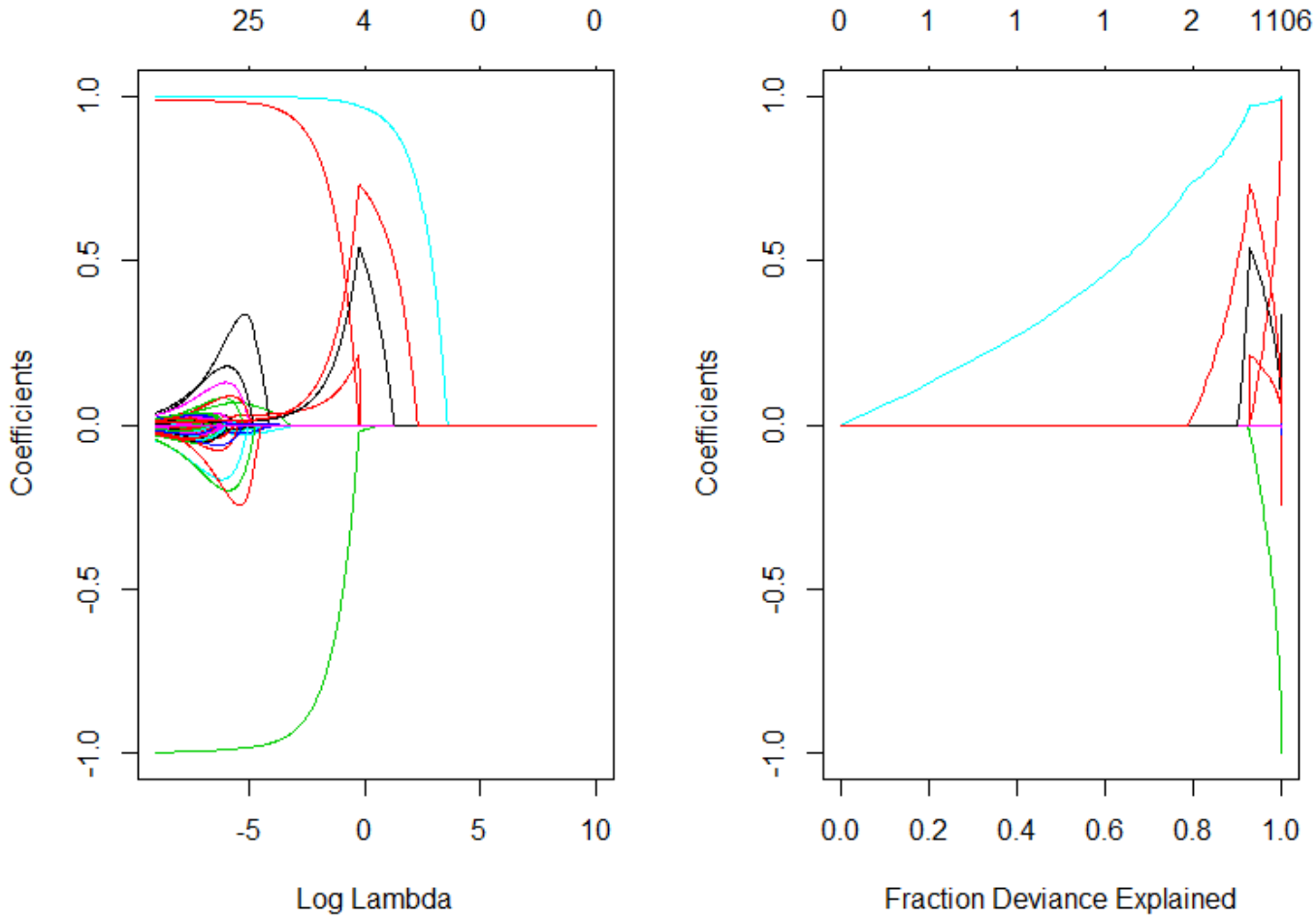


Figure: Estimates of the coefficients against λ (left) and deviance (right) for the Lasso. Upper x-axis indicates number of coefficients in the model.

Example: 2008 Flights

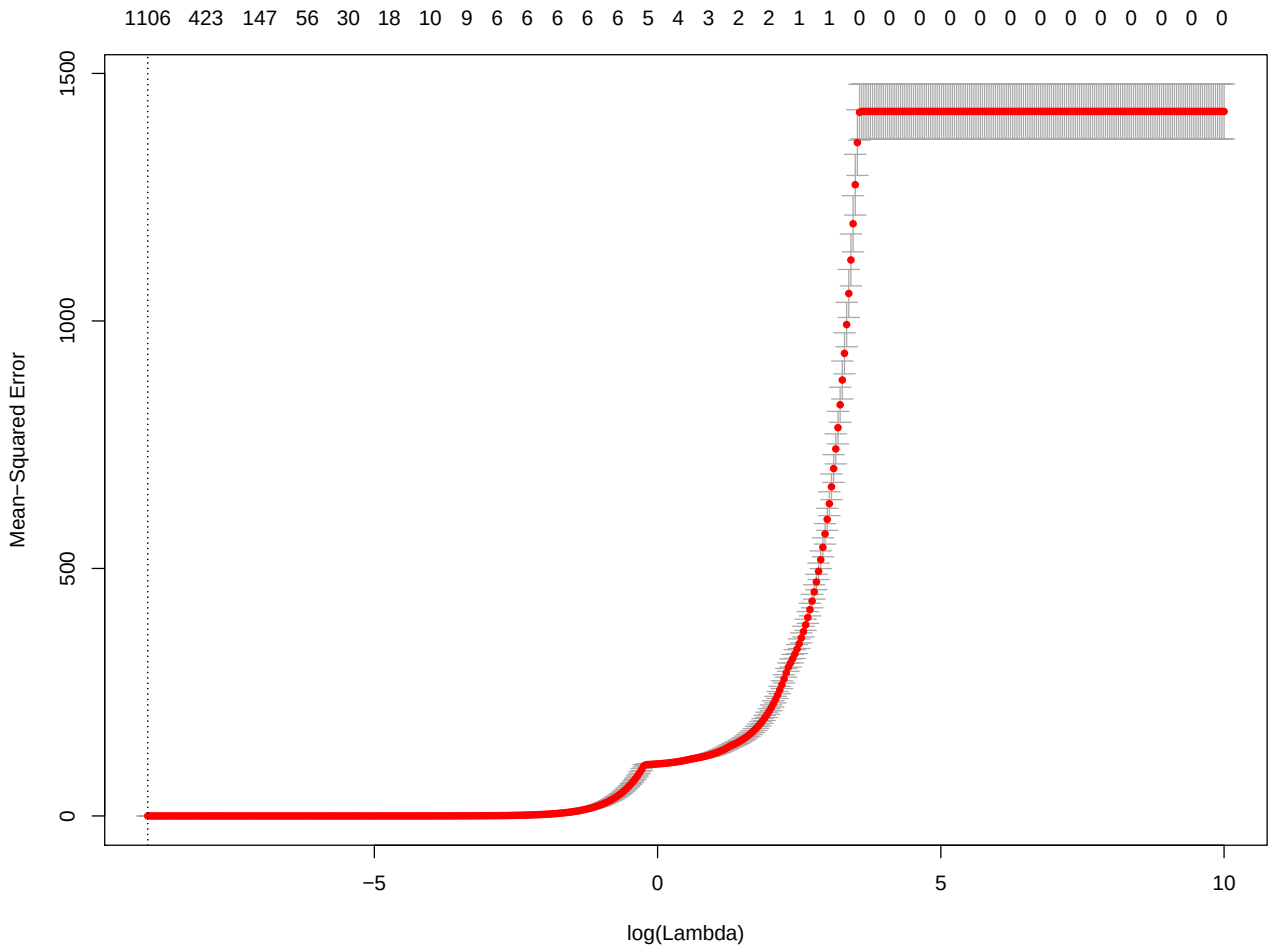


Figure: MSEs calculated through cross-validation for different values of λ. Upper x-axis indicates number of coefficients in the model.

Lasso Coefficients

- The Lasso shrinks most parameters to zero and selects the important variables.
- In this case at $\log(\lambda) = -1$ we get the following estimates:

Name	Estimate
(Intercept)	-6.94
UniqueCarrierWN	0.12
ActualElapsedTime	0.53
CRSElapsedTime	-0.54
DepDelay	0.99
TaxiIn	0.26
TaxiOut	0.34

- All other coefficients are equal to zero.

Ridge Regression

Definition of ridge regression

- The ridge estimate is defined by

$$\hat{\beta}^{ridge} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2$$

$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 \leq t.$$

- Or equivalently in *Lagrangian form*

$$\hat{\beta}^{ridge} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\},$$

with $\lambda \geq 0$.

Ridge regression philosophy

- Ridge regression can be seen as making a bias-variance trade-off. This makes sense if we consider that the MSE of an estimator consists of the variance and the bias.

$$\textbf{MSE}\left(\hat{\theta}\right) = \mathit{Var}\left(\hat{\theta}\right) + \mathit{Bias}\left(\hat{\theta}, \theta\right)^2$$

- Ridge regression **does not exclude variables**, but **reduces effect estimates to near zero**. It is therefore not suited for finding parsimonious models.

Example: 2008 Flights

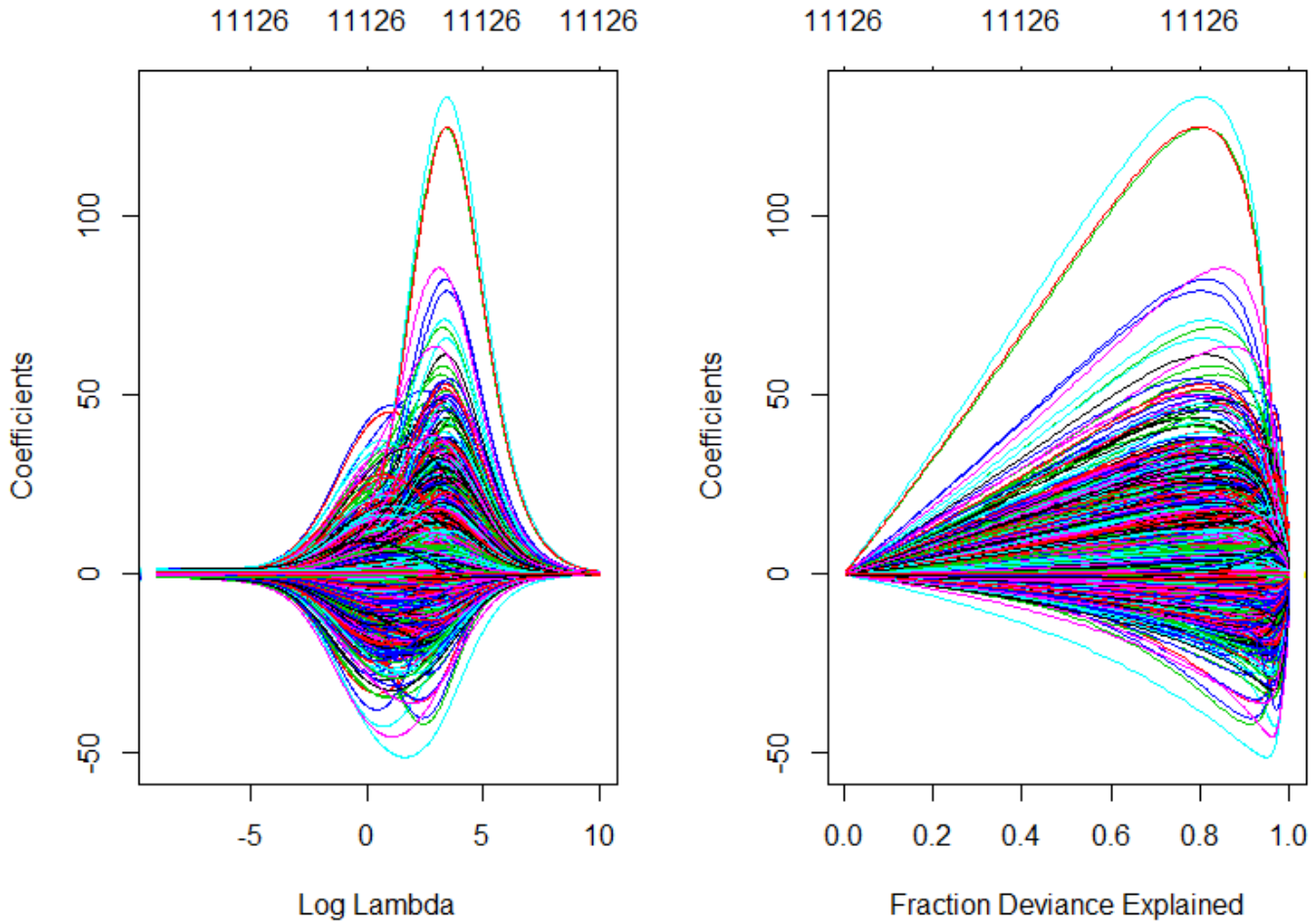


Figure: Estimates of the coefficients against λ (left) and deviance (right) for ridge regression. Upper x-axis indicates number of coefficients in the model.

Example: 2008 Flights

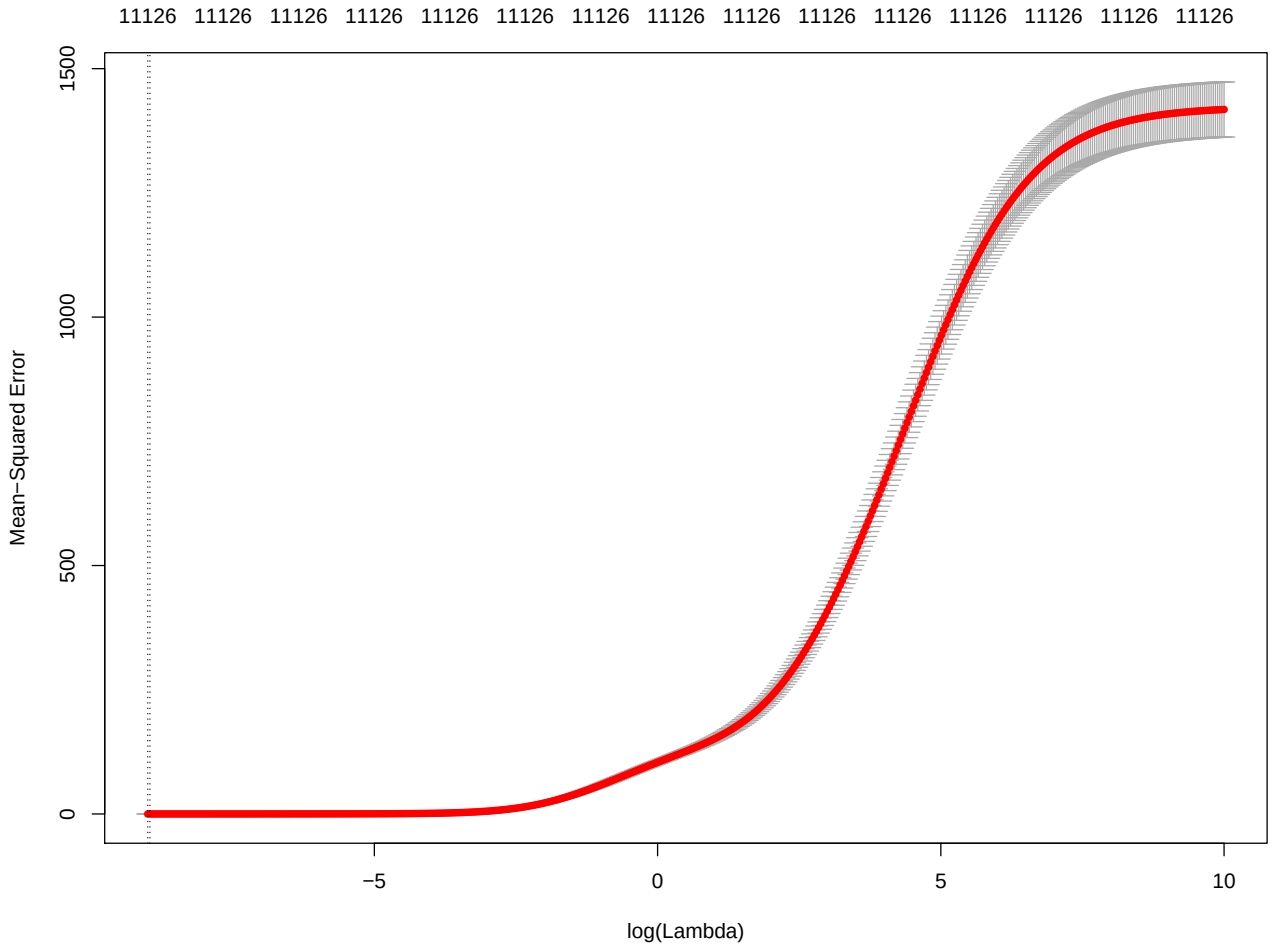


Figure: MSEs calculated through cross-validation for different values of λ. Upper x-axis indicates number of coefficients in the model.

Differences between ridge and lasso

- Lasso will perform variable selection, whereas Ridge will shrink the coefficients close to zero, but nearly never to zero.

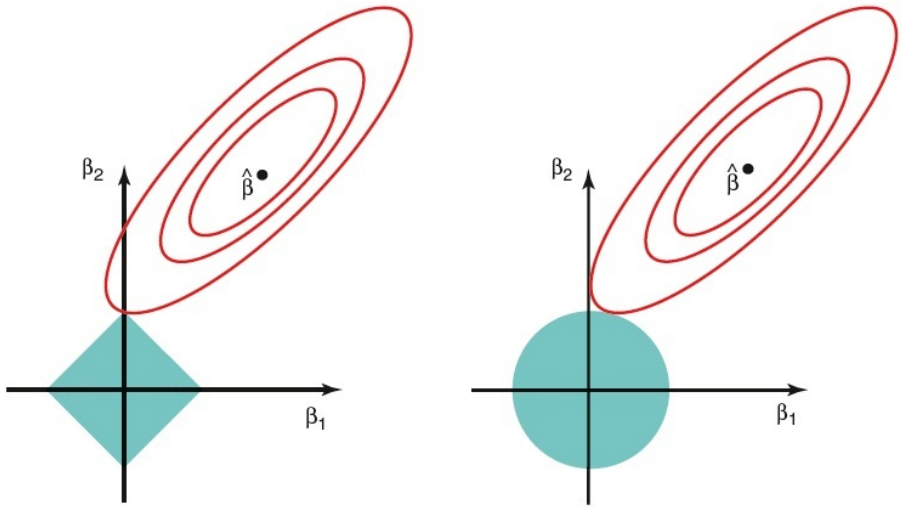


Figure: Contours of the error and constraint functions for the lasso (left) and ridge regression (right). The solid blue areas are the constraint regions, $|\beta_1| + |\beta_2| \leq t$ and $\beta_1^2 + \beta_2^2 \leq t$, while the red ellipses are the contours of the least squares error function.

[1] Trevor Hastie, Robert Tibshirani, Jerome Friedman (2009). Elements of Statistical Learning, Springer, Second Edition, p. 71

Problems with Ridge and Lasso

The lasso

- It will only select one variable out of a group of highly correlated variables.
- It can have at most n non-zero coefficients (possibly not good for prediction).

Ridge regression

- It is harder or impossible to interpret.

Elastic net

Elastic net: Ideas behind

- To address the problems of lasso and ridge, the elastic net was created. It generalizes both and combines the l_1 and l_2 penalties.
- The elastic net **combines** the benefits of both lasso and ridge Regression.
  - It will shrink coefficients for groups of highly correlated variables like Ridge
  - It will set variables to zero like Lasso
  - It can give more than n non-zero coefficients in the $n < p$ case

Elastic net definition

The Elastic Net estimate is given by

$$\hat{\beta}^{elnet} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2$$

$$\text{subject to } \sum_{j=1}^{p} \left(\alpha|\beta_j| + (1-\alpha)\,\beta_j^2 \right) \leq t.$$

Or equivalently in *Lagrangian form*

$$\hat{\beta}^{elnet} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 + \lambda \sum_{j=1}^{p} \left(\alpha|\beta_j| + (1-\alpha)\,\beta_j^2 \right) \right\}$$

with $\lambda \geq 0$ and $\alpha \in [0, 1]$.

Example: 2008 Flights

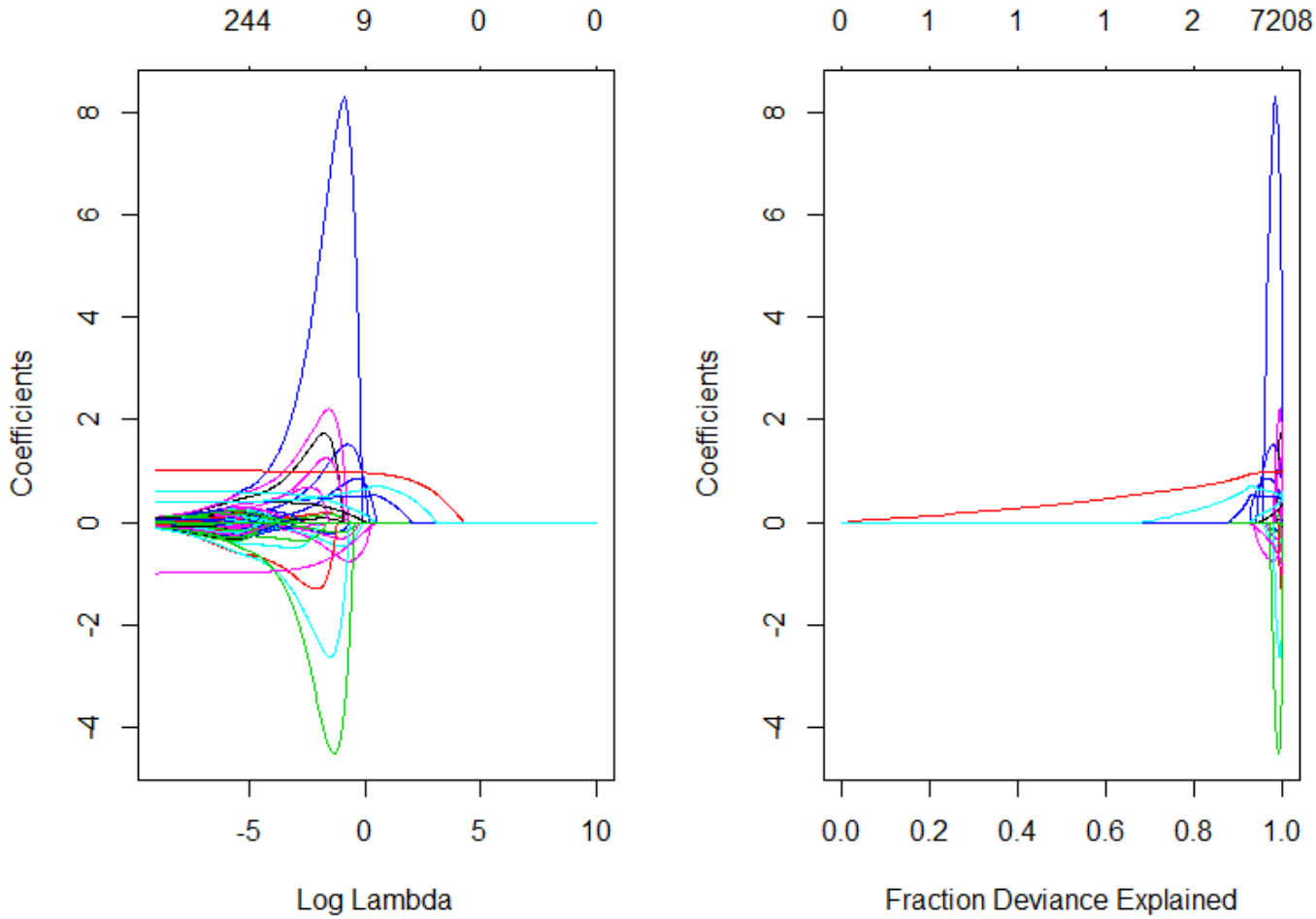


Figure: Estimates of the coefficients against λ (left) and deviance (right) for the elastic net with $\alpha = 0.5$. Upper x-axis indicates number of coefficients in the model.

Example: 2008 Flights

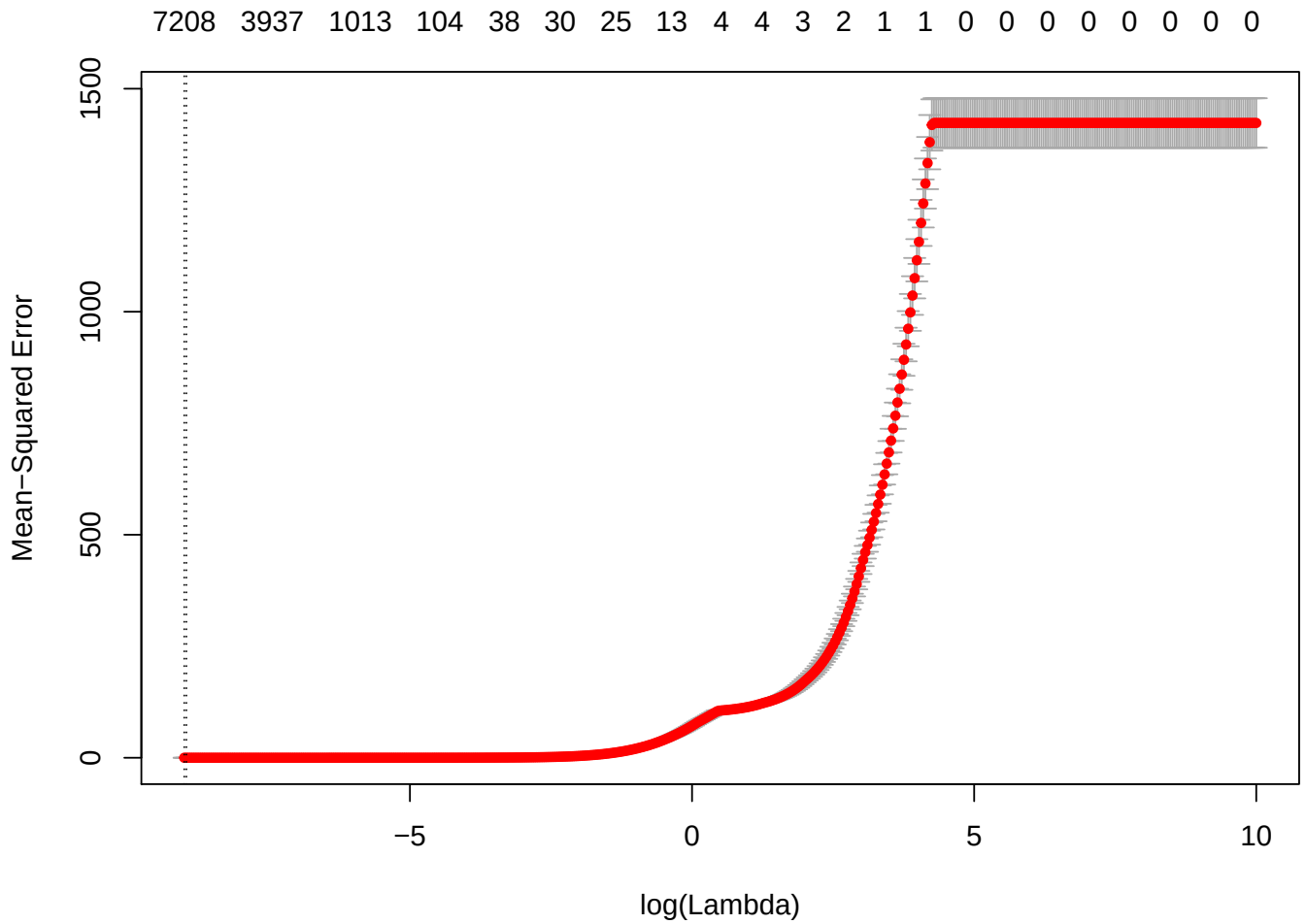


Figure: MSEs calculated through cross-validation for different values of λ. Upper x-axis indicates number of coefficients in the model.

Comparison of results

We compare the models using the mean squared prediction error.

- A smaller value indicates a better prediction.
- We will consider both cross-validated MSE and test sample MSE

Example: 2008 Flights

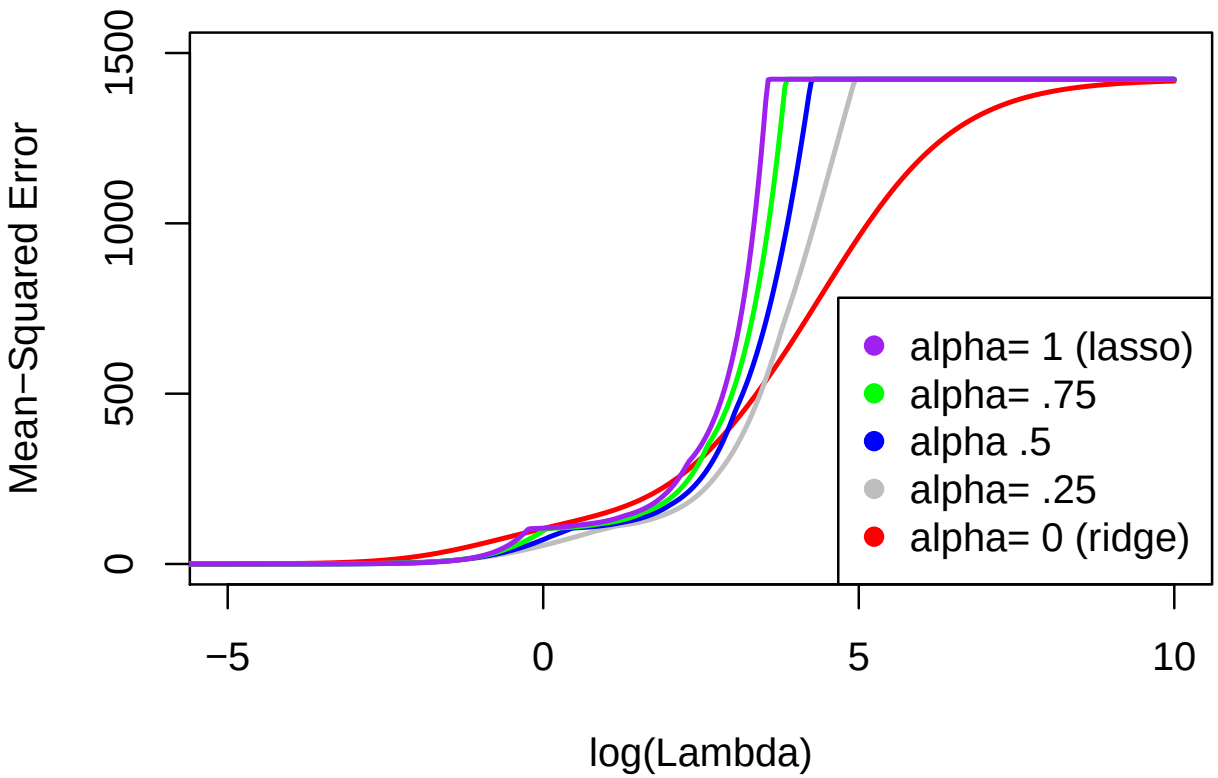


Figure: Crossvalidated MSEs of alle five models. The improvements with decreasing λ become increasingly smaller. An unrestricted model seems to have the smallest MSE.

Example: 2008 Flights

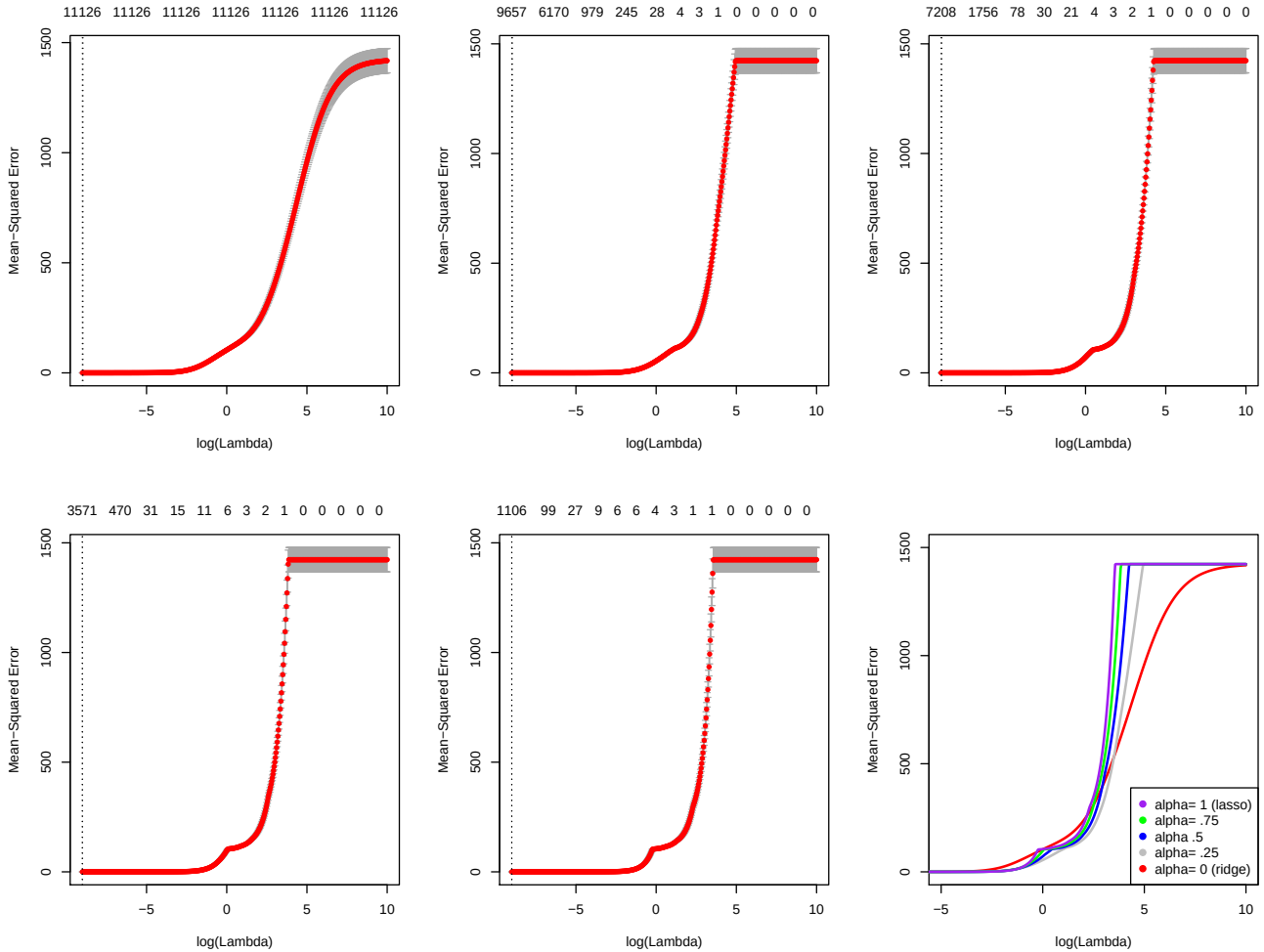


Figure: MSEs for all five models. Upper x-axis indicates the number of parameters in the model. Top left: ridge regression, bottom middle: lasso.

Results interpretation

- We do indeed decrease the MSE by allowing the λ to become even smaller, but the gains become minuscule.
- An optimal balance between low MSE and small number of coefficients is probably around $\log(\lambda) = -1$. The lasso can achieve this with only 6 variables.
- We would expect with these data, that we need a lot of variables to achieve a very small MSE, since the arrival delay has certainly been optimized by many airlines and easy things to fix are probably not around anymore.
  - The MSEs at $\log(\lambda) = -1$ from the hold-out test dataset are:

α	Ridge	0.25	0.5	0.75	Lasso
MSE	45.87	18.26	19.66	21.61	22.17