Linear model selection and regularisation

Federica Eduati

Eindhoven University of Technology
Department of Biomedical Engineering

2024

Learning goals

- **Prediction & Generalisation:** Recognise the importance of prediction on unseen data and understand the concept of generalisation error, including bias and variance trade-off.
- **Model Evaluation:** Understand functioning of cross-validation to assess model performance and generalisation capability and for hyper-parameter tuning.
- **Model Complexity:** Explore methods for subset selection to reduce model complexity and prevent overfitting.
- **Regularisation Techniques:** Understand and apply Ridge regression, the Lasso, and Elastic Net regularisation to control model complexity and improve generalisation.

Material

- Chapters 5 and 6 of "*An introduction to statistical learning with applications in python*, G. James, D. Witten, T. Hastie, R. Tibshirani, J. Taylor"

Overview

- Generalisation error and bias-variance trade-off
- Resampling methods (cross-validation)
- Methods for subset selection
- Ridge regression
- The Lasso
- Elastic Net

Instructions

Go to

www.menti.com

Enter the code

7643 3528

Or use QR code

Quick recap on linear models

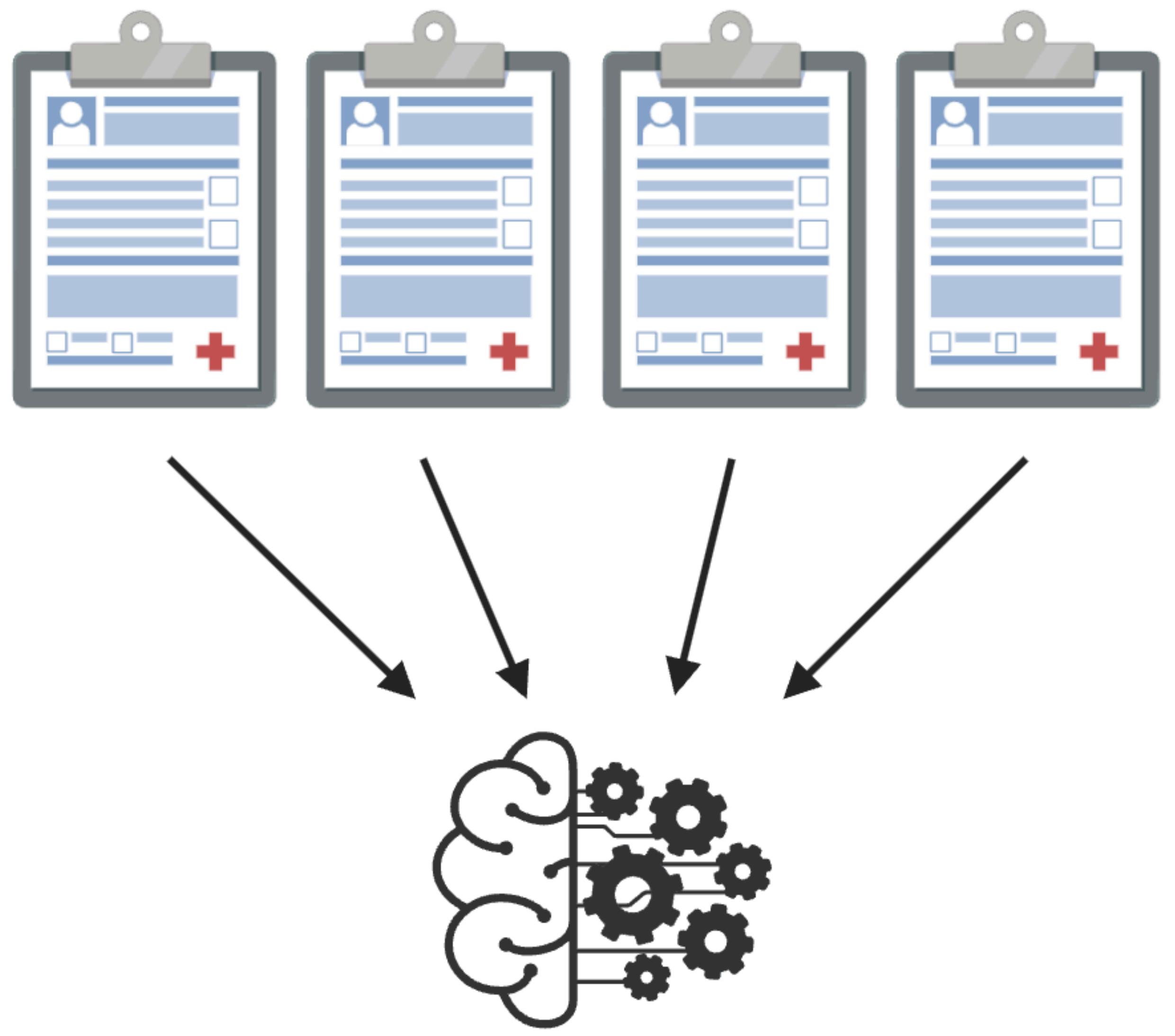
- Definition of linear model for regression and classification (i.e. logistic regression)
- Estimation of model parameters using a training dataset
  - Linear regression: Minimise residual sum of squares (RSS)
  - Logistic regression: Maximise likelihood function
- Interpretation of the model coefficients
- Evaluation of model fit to the training data

Why we care about predictions on unseen data

Training data

$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$, where

each $x_i = (x_{i1}, x_{i2}, \dots, x_{ip})$



Past patients for which we have, e.g. clinical data, lab measurements, imaging and also the outcome (diagnosis, response to treatment).

Test data

$x_0 = (x_{01}, x_{02}, \dots, x_{0p})$

$\hat{y}_0 = ?$

New patient coming to the hospital that needs diagnosis/prognosis

Example: simulated data

To illustrate the impact of model complexity we uses a simulated example ($y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \beta_4 x_{i4} + e_i$, where e_i is randomly generated noise).

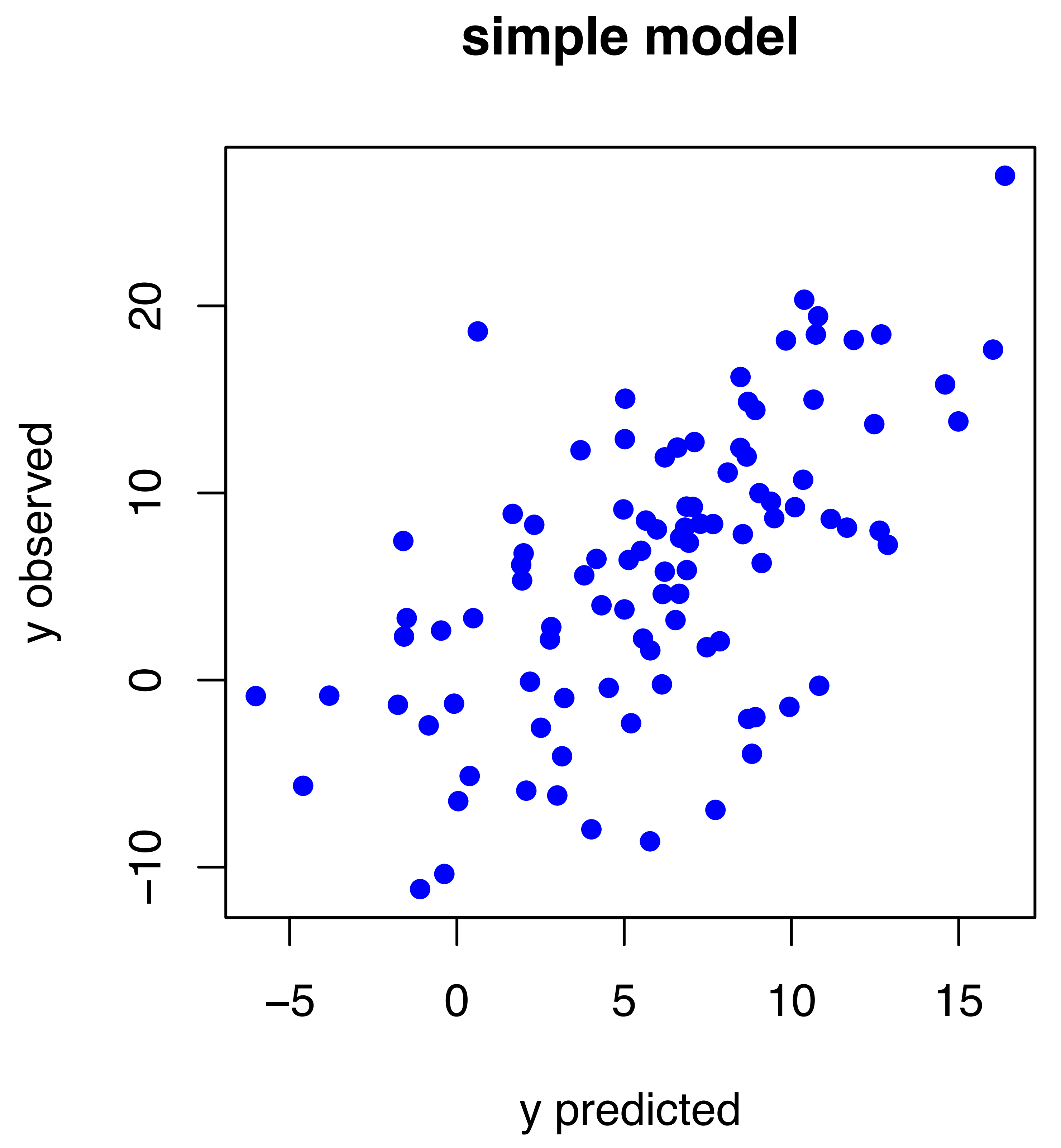
We simulated 20 different datasets with 100 observations each.

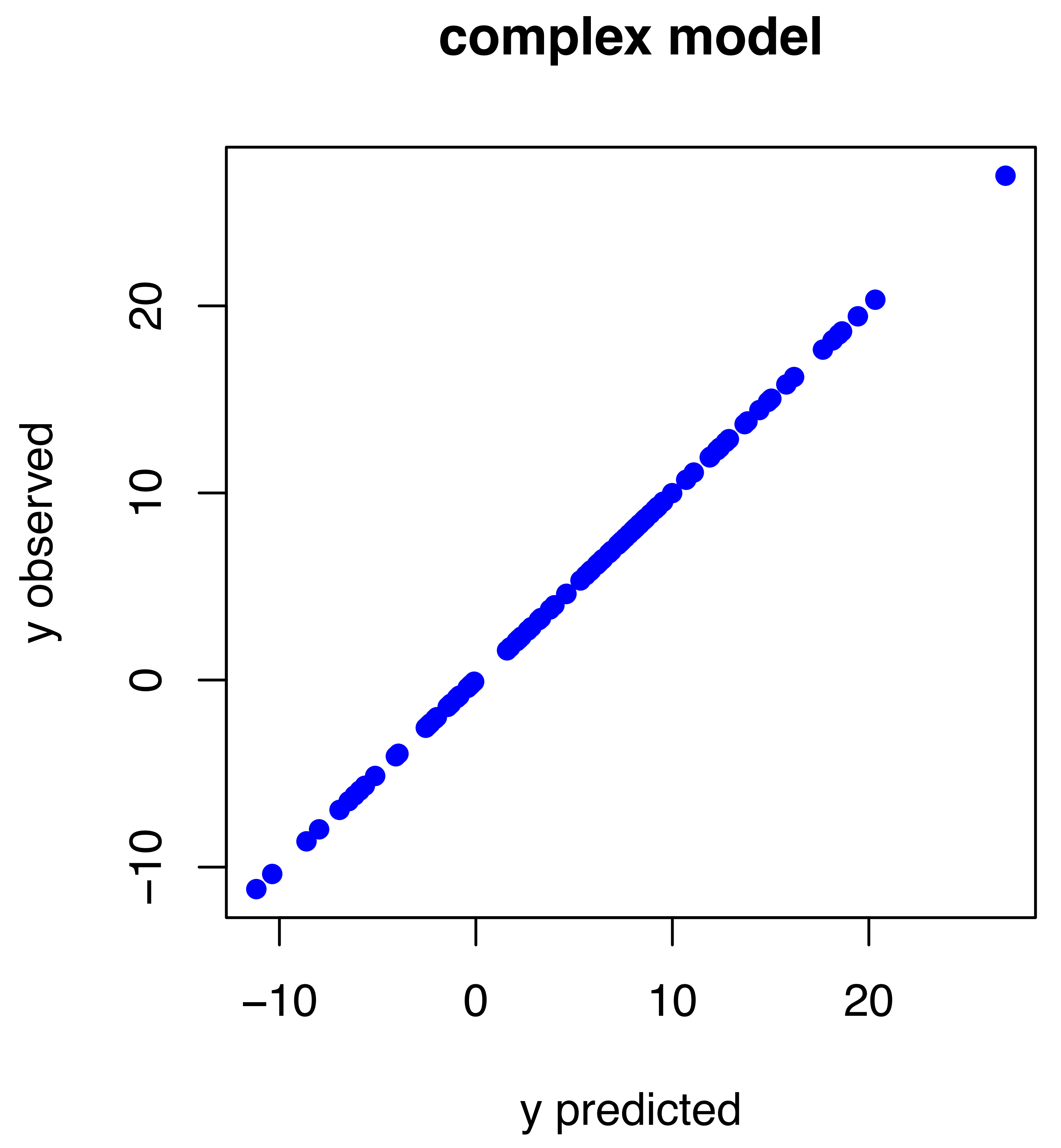
For each dataset, we fit two different models:

- Very *simple model*: $y = \beta_0 + \beta_1 x_1$
- Very *complex model*: $y = \beta_0 + \beta_1 x_1 + \ldots + \beta_{100} x_{100}$

Note: here we know that only 4 features are important, but in general we do not know the true underlying model structure.

Example: model fit on training data

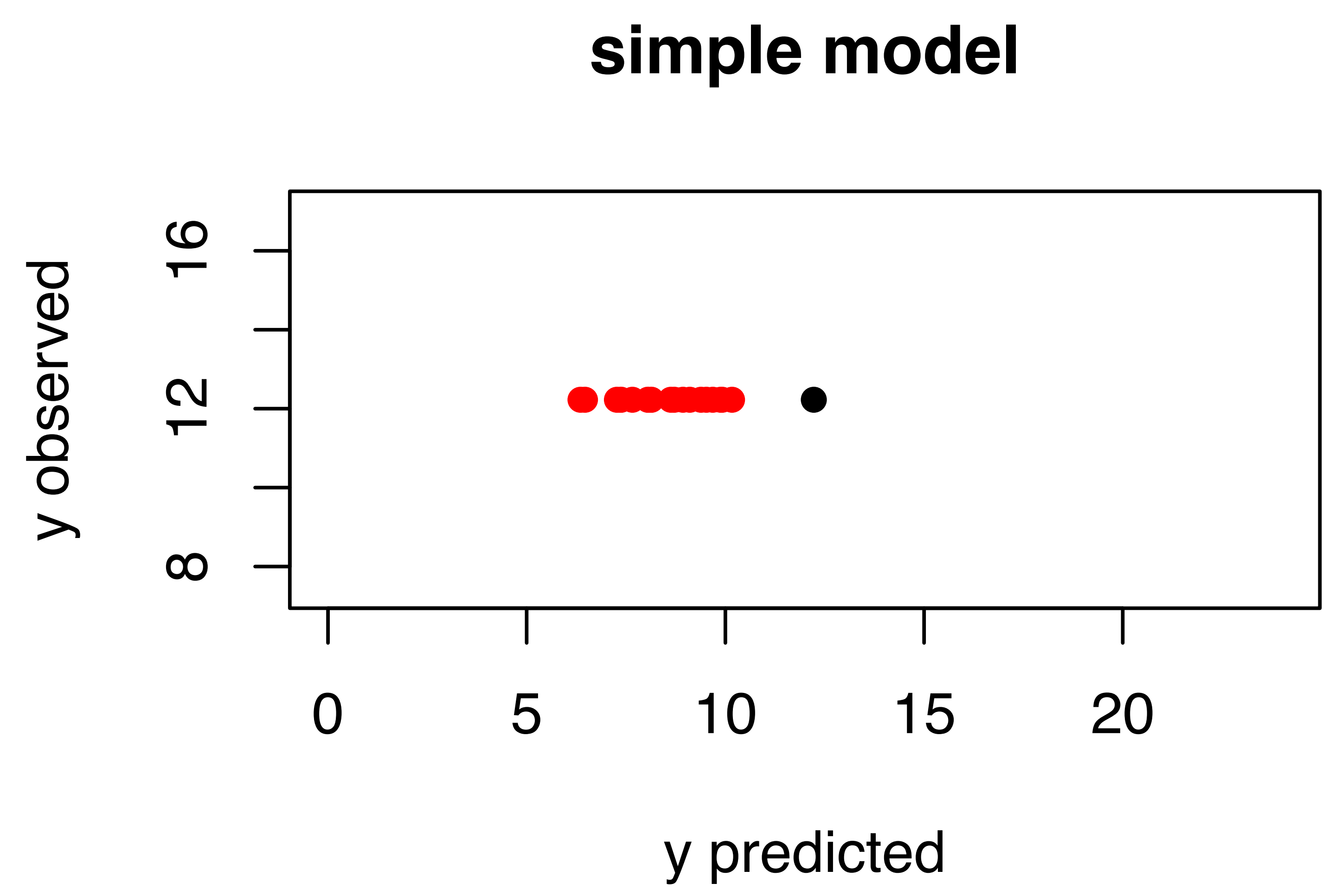


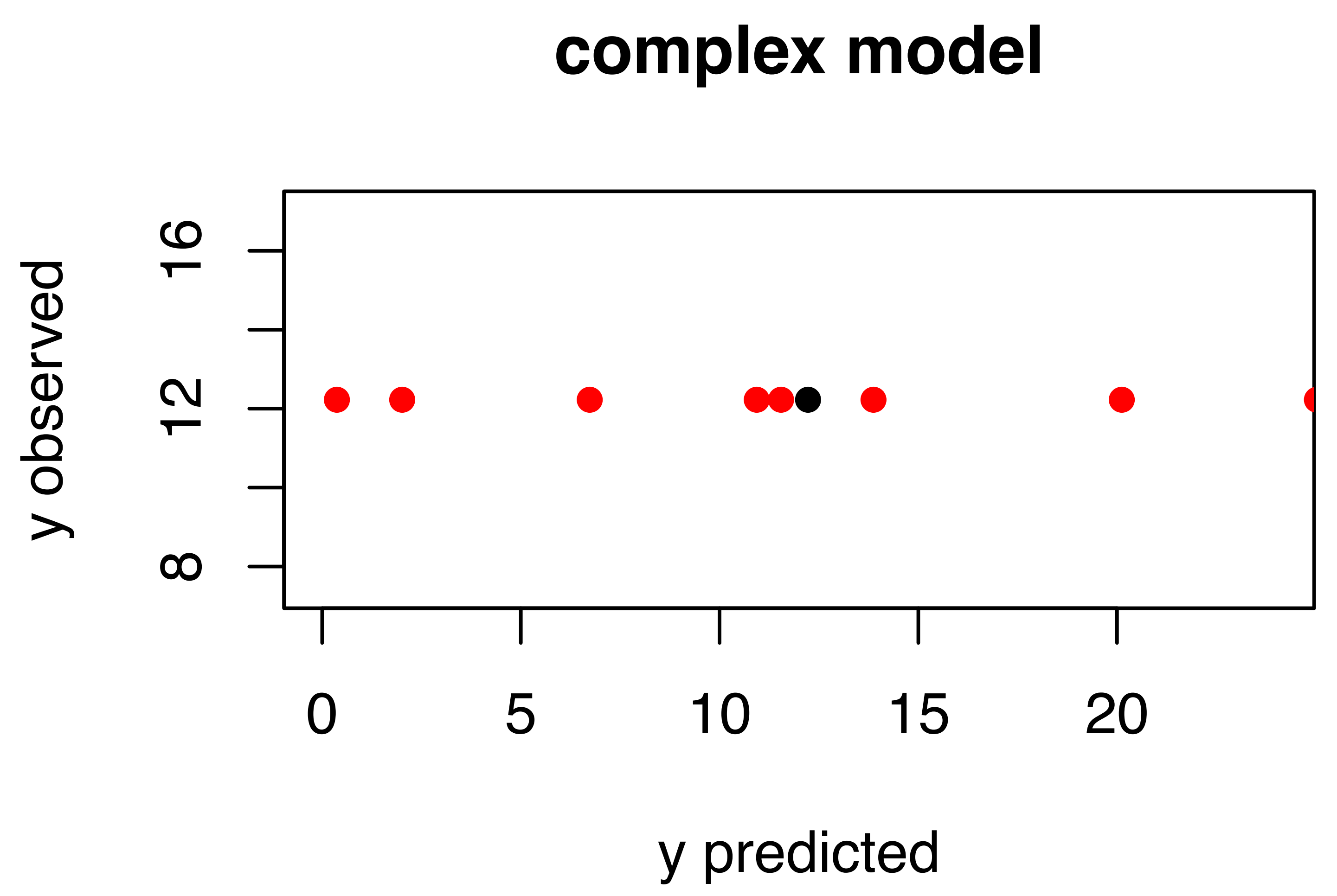


- The very *complex model* always perfectly fit the training data
- The very *simple model* capture some trend but cannot describe the training data very well

Example: What happens when we make a new prediction?

We use the 20 trained models to make predictions for a new observation:





● Ideal prediction

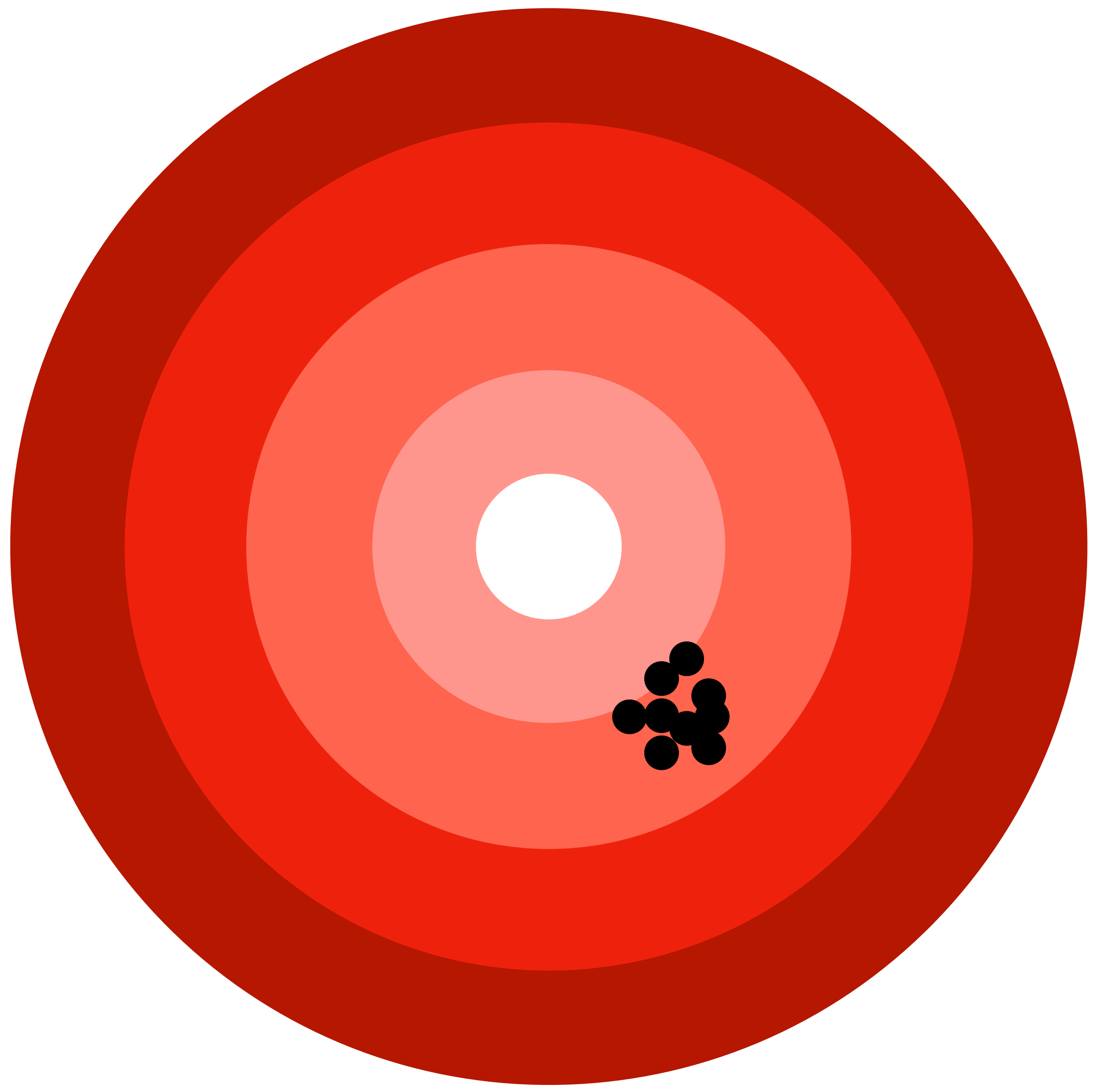
🔴 Actual predictions

Both models make some error on the new data:

- Very simple model: systematic error
- Very complex model: stochastic error

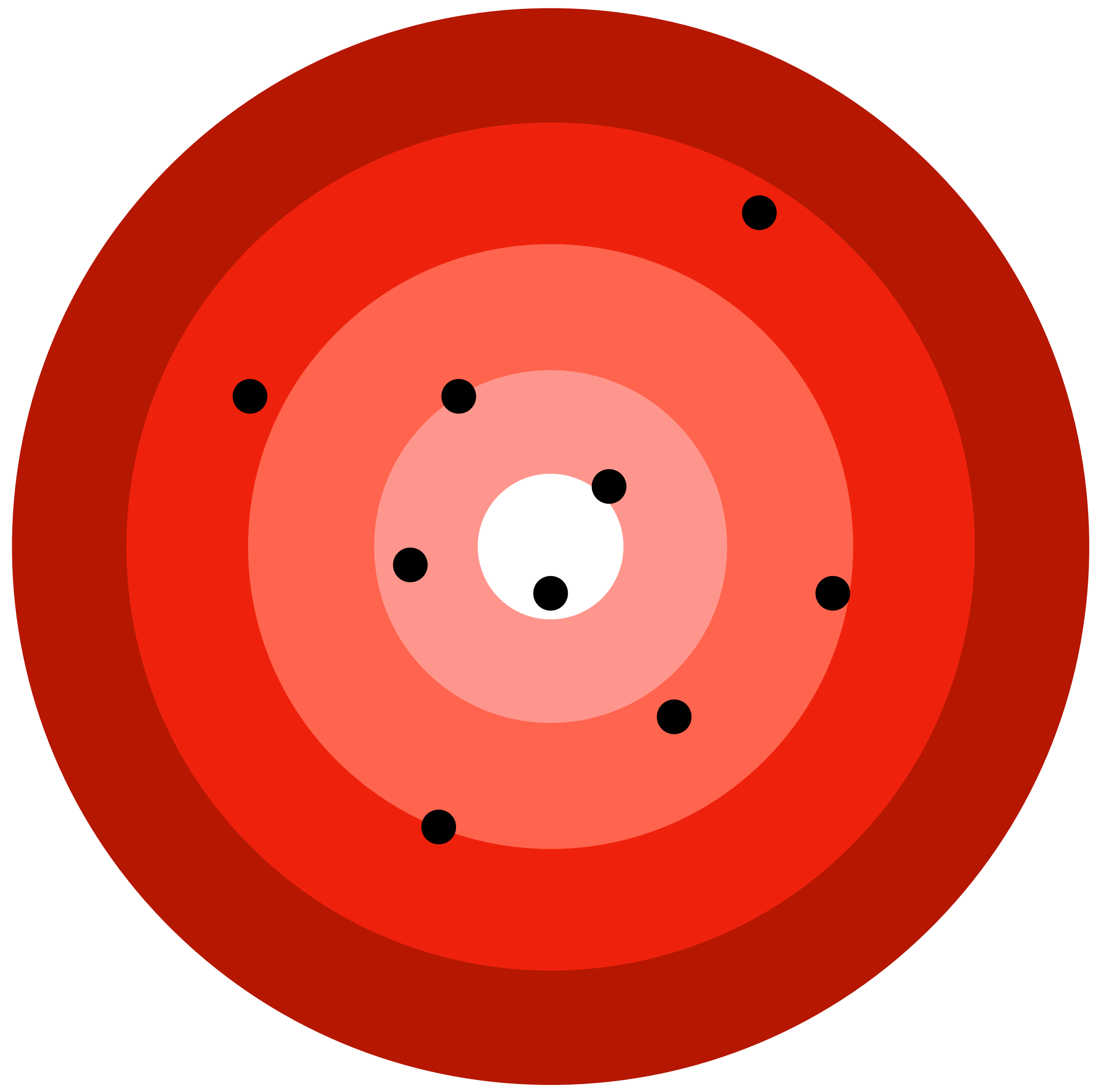
Bias and variance

Visual example: hitting a target on a dartboard



High bias

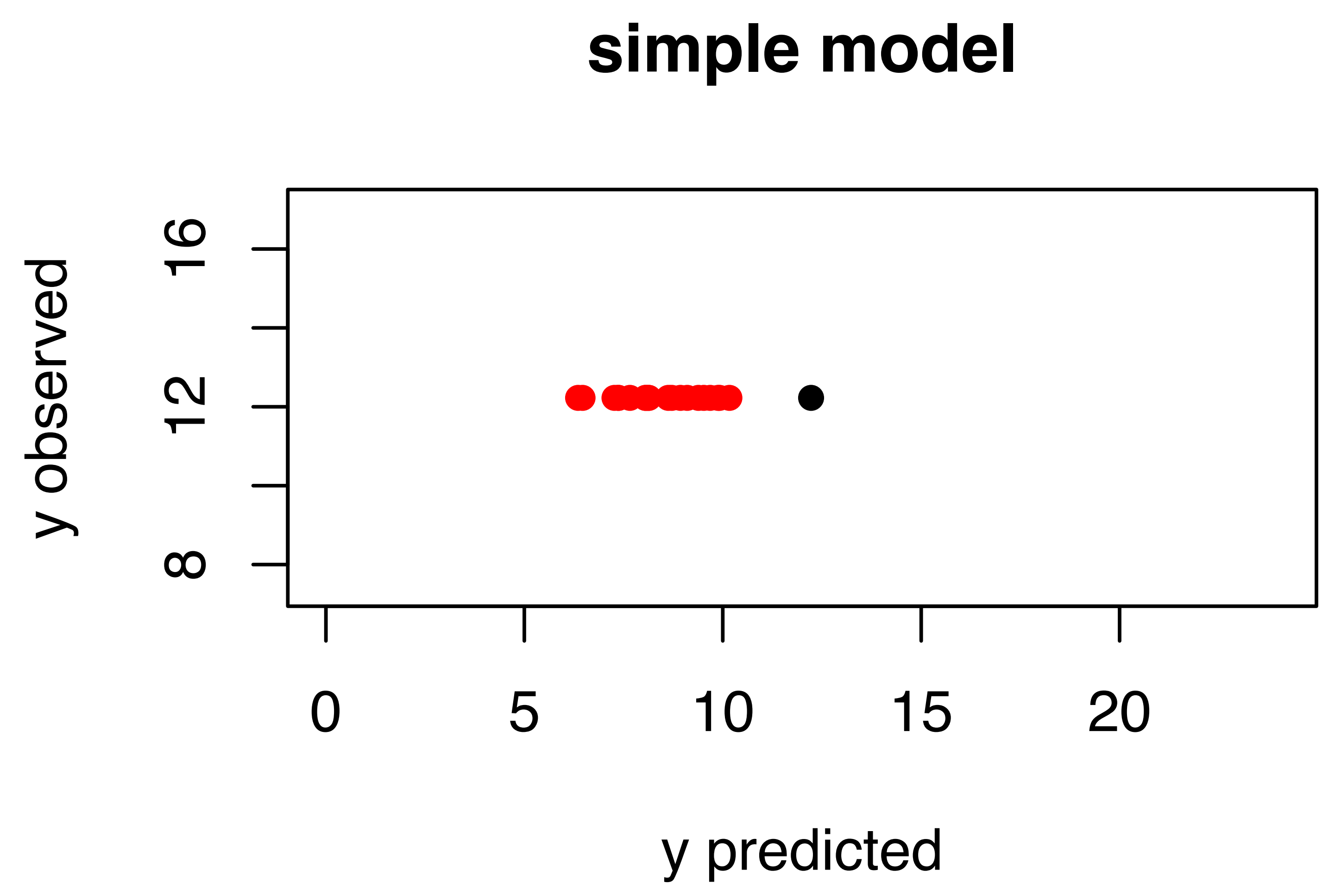
When the model consistently makes errors by oversimplifying the problem.



High variance

When the model is too sensitive to training data, leading to inconsistent results with new data.

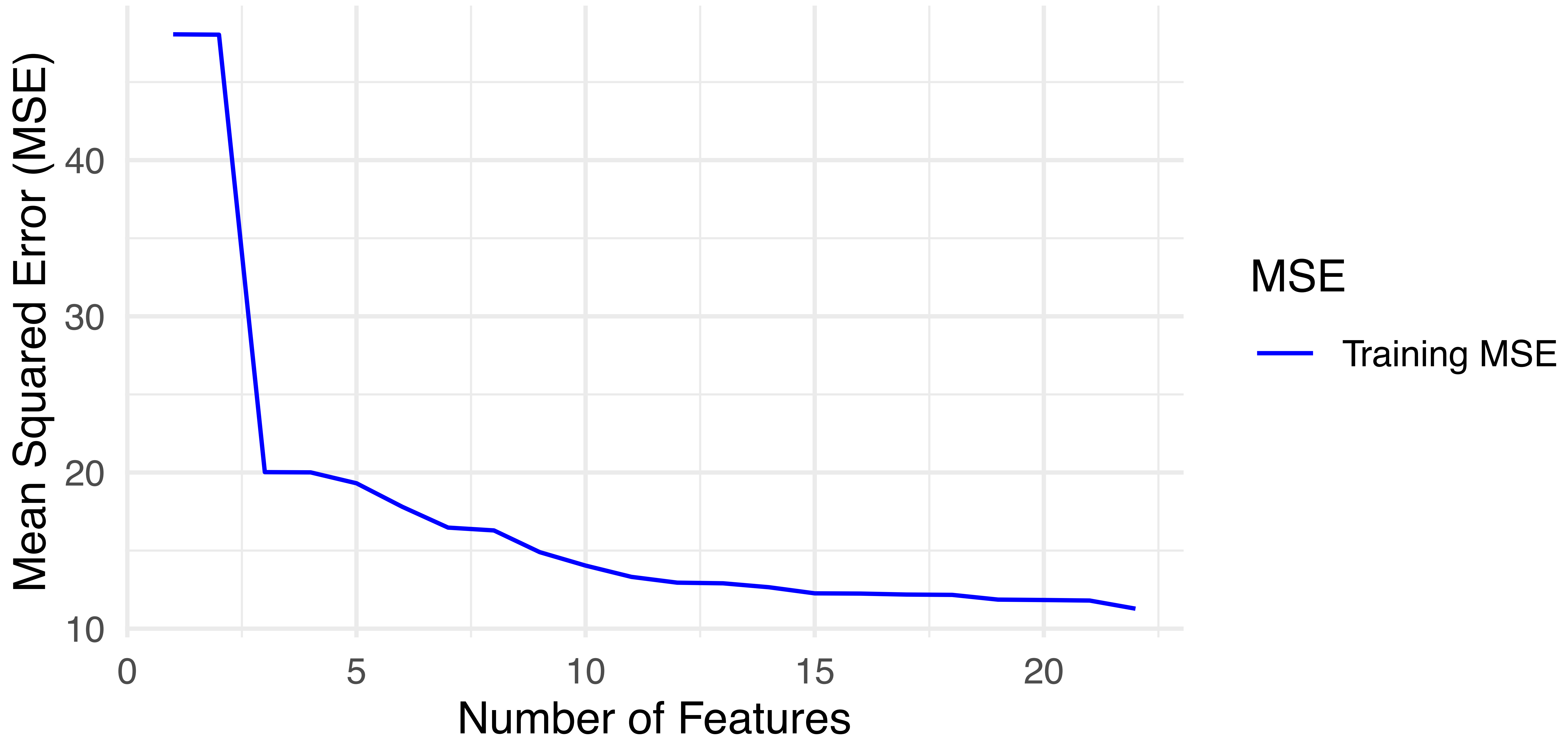
A too simple model has:



1. High bias and low variance
2. High bias and high variance
3. Low bias and high variance
4. Low bias and low variance

Varying model complexity: effect on training set

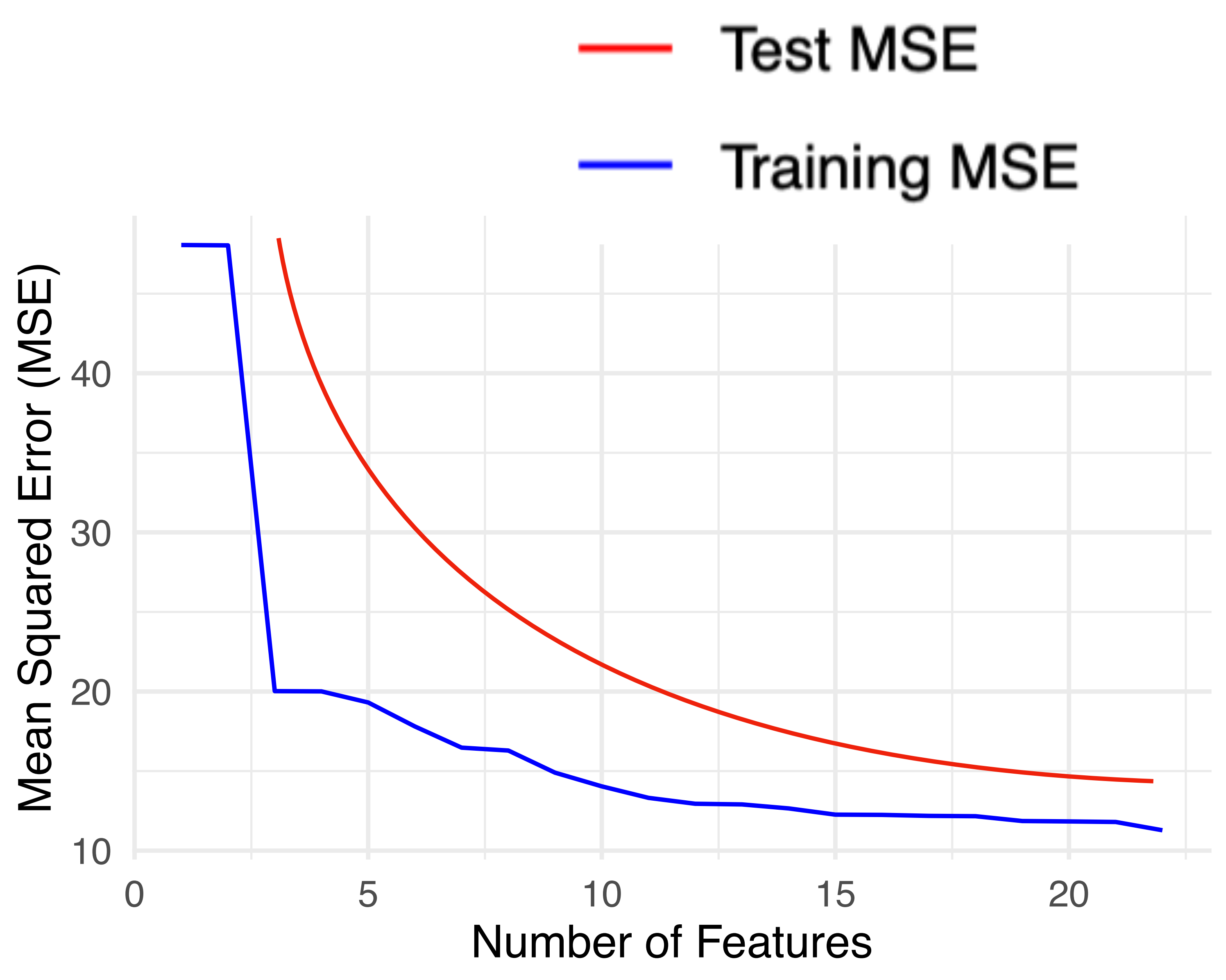
Model error on the training set when fitting models with increasing number of features

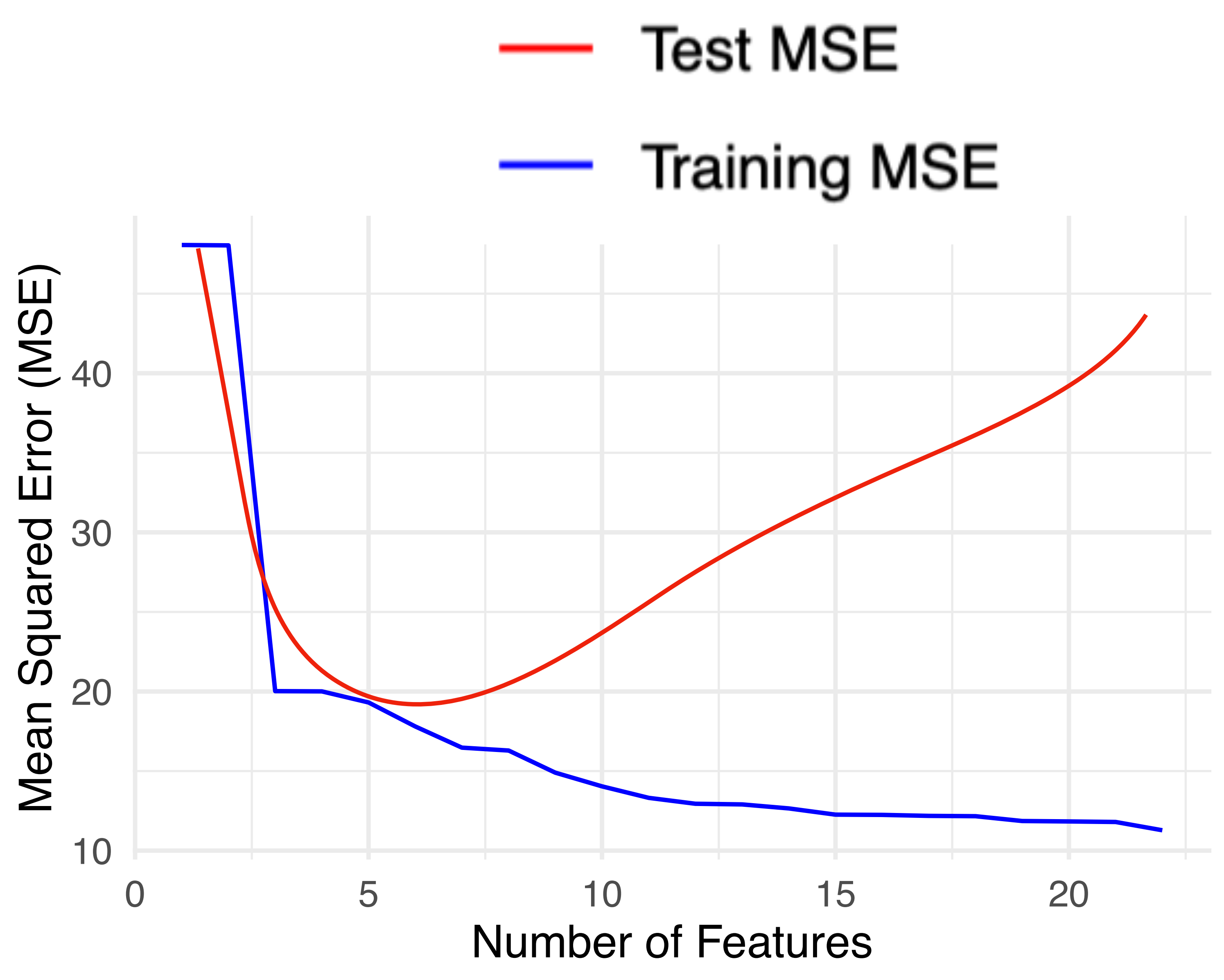


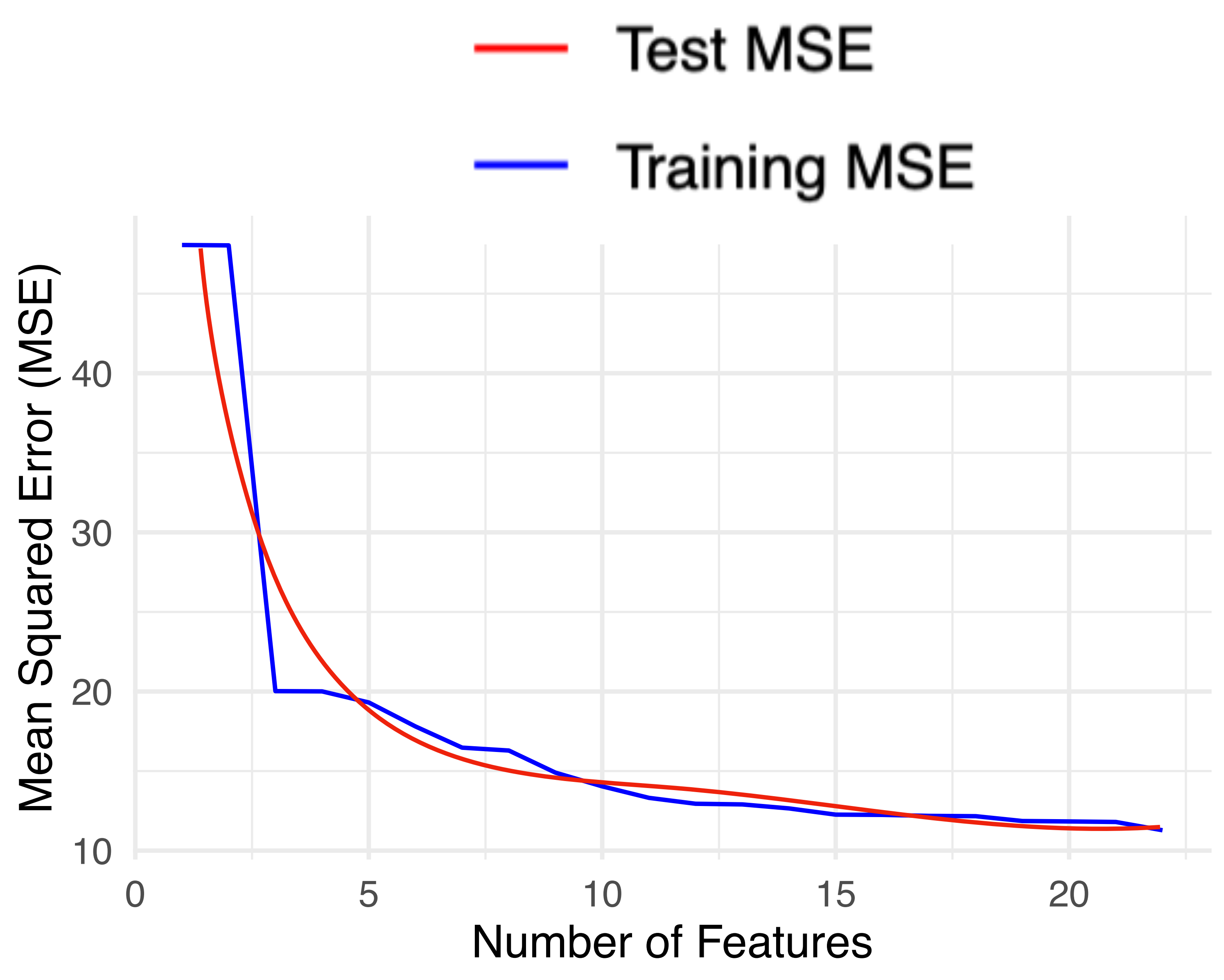
Synthetic dataset ($n = 100$) simulated using 4 features ($p = 4$).

70% of the data used for training, 30% for testing.

What is the trend of the test error for increasing model complexity?



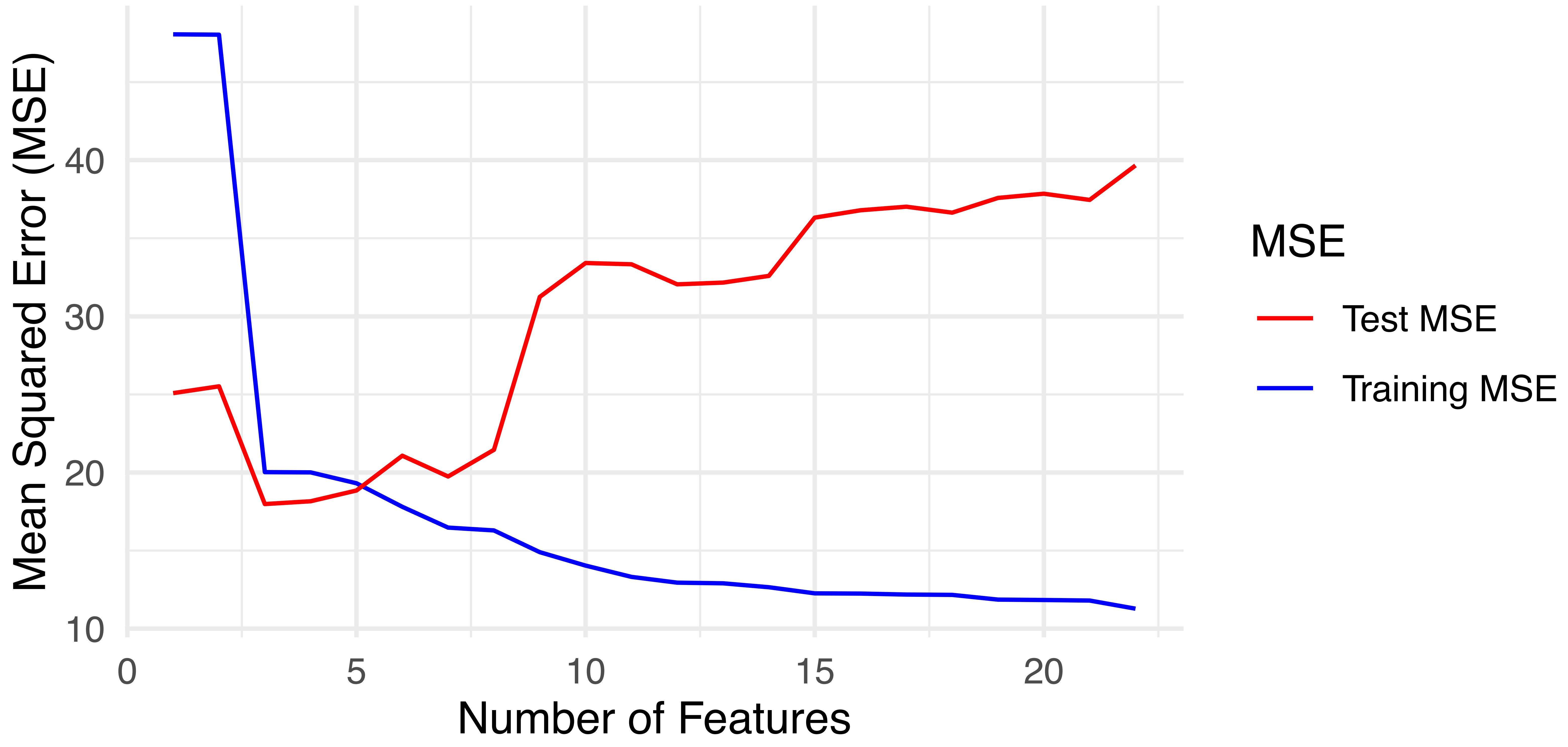




1. Similar trend as the training error but higher error for the test set
2. The fit improves to a certain point but can get worse when too many parameters are included
3. The fit to the test set is always comparable to the one of the training set

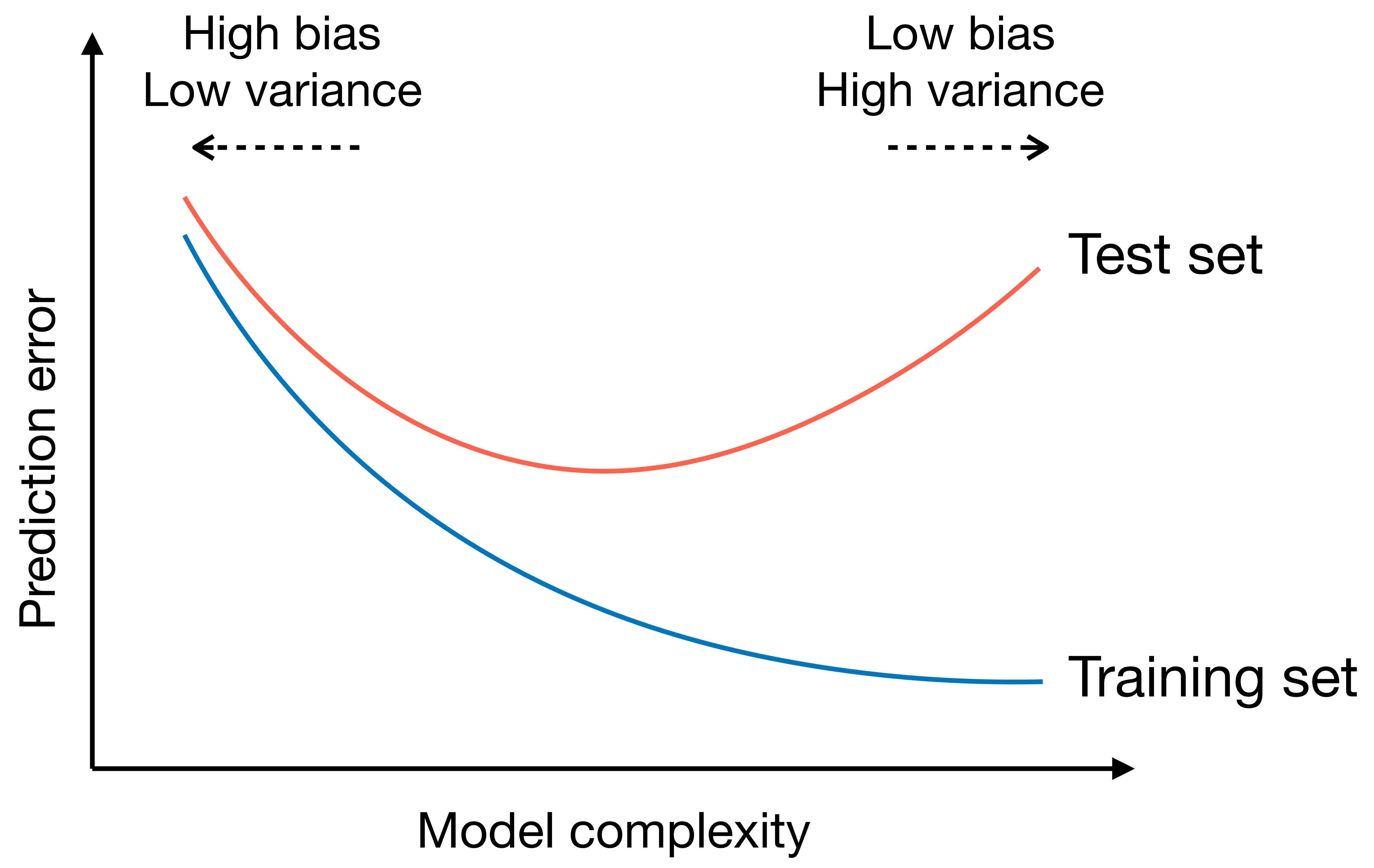
Varying model complexity: effect on training and test set

Model error on the training set when fitting models with increasing number of features



Synthetic dataset ($n = 100$) simulated using 4 features ($p = 4$).
70% of the data used for training, 30% for testing.

Bias and variance trade-off



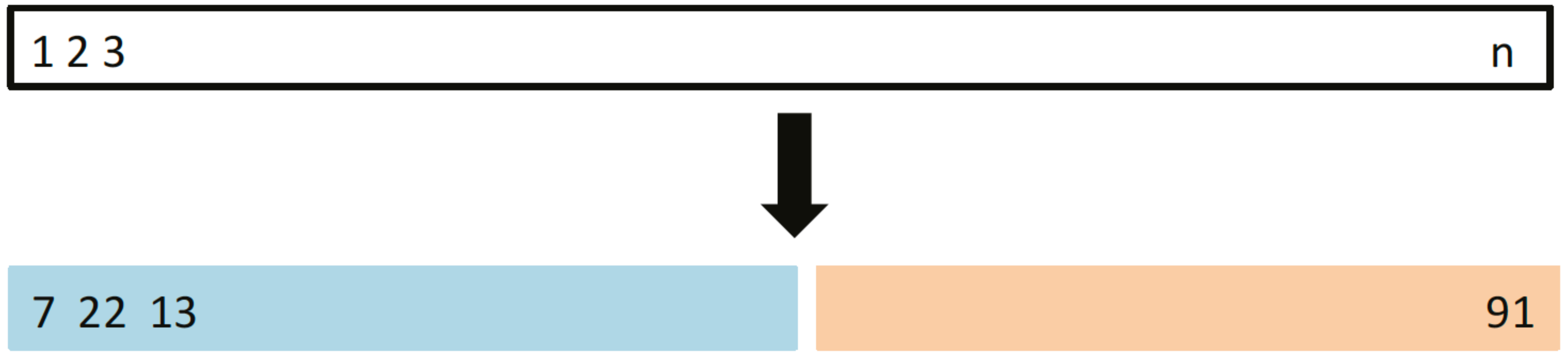
Complex models tend to overfit the training data and perform poorly on the test data

Resampling methods

When we do not have access to a very large designated test set, we can estimate the test error rate by *holding out* a subset of the training observations from the fitting process and then apply our model to those held out observations.

Resampling methods can be useful also to tune model *hyper parameters* (a.k.a. *tuning parameters*). Model hyperparameters are settings that define the model's structure or how it learns from data (e.g., the number of neighbors in k-NN) and are set before the training process begins.

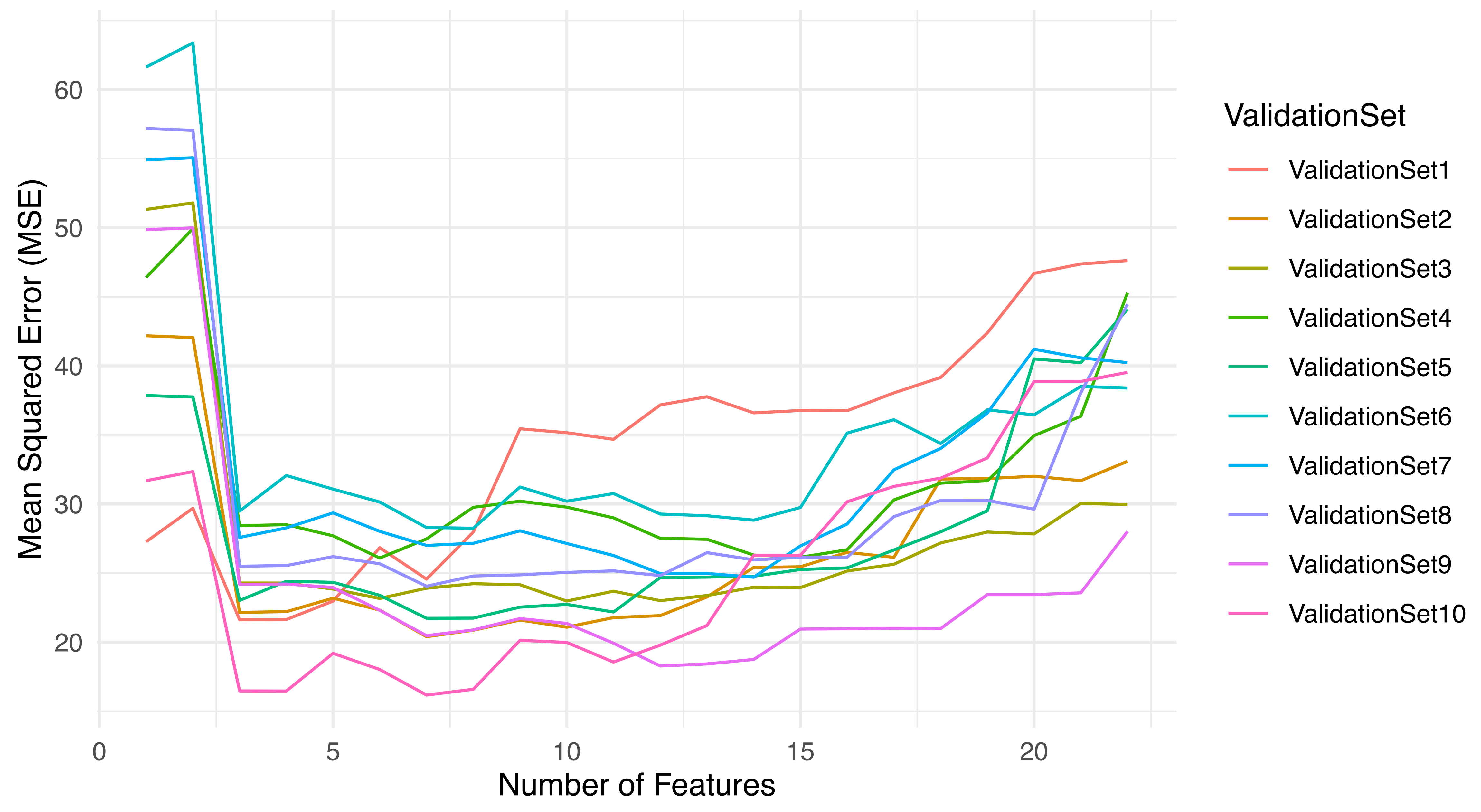
The validation set approach



- The n observations are randomly divided in a *training set* and a *validation set*
- The model is trained on the training set
- The fitted model is used to predict responses for the observations in the validation set

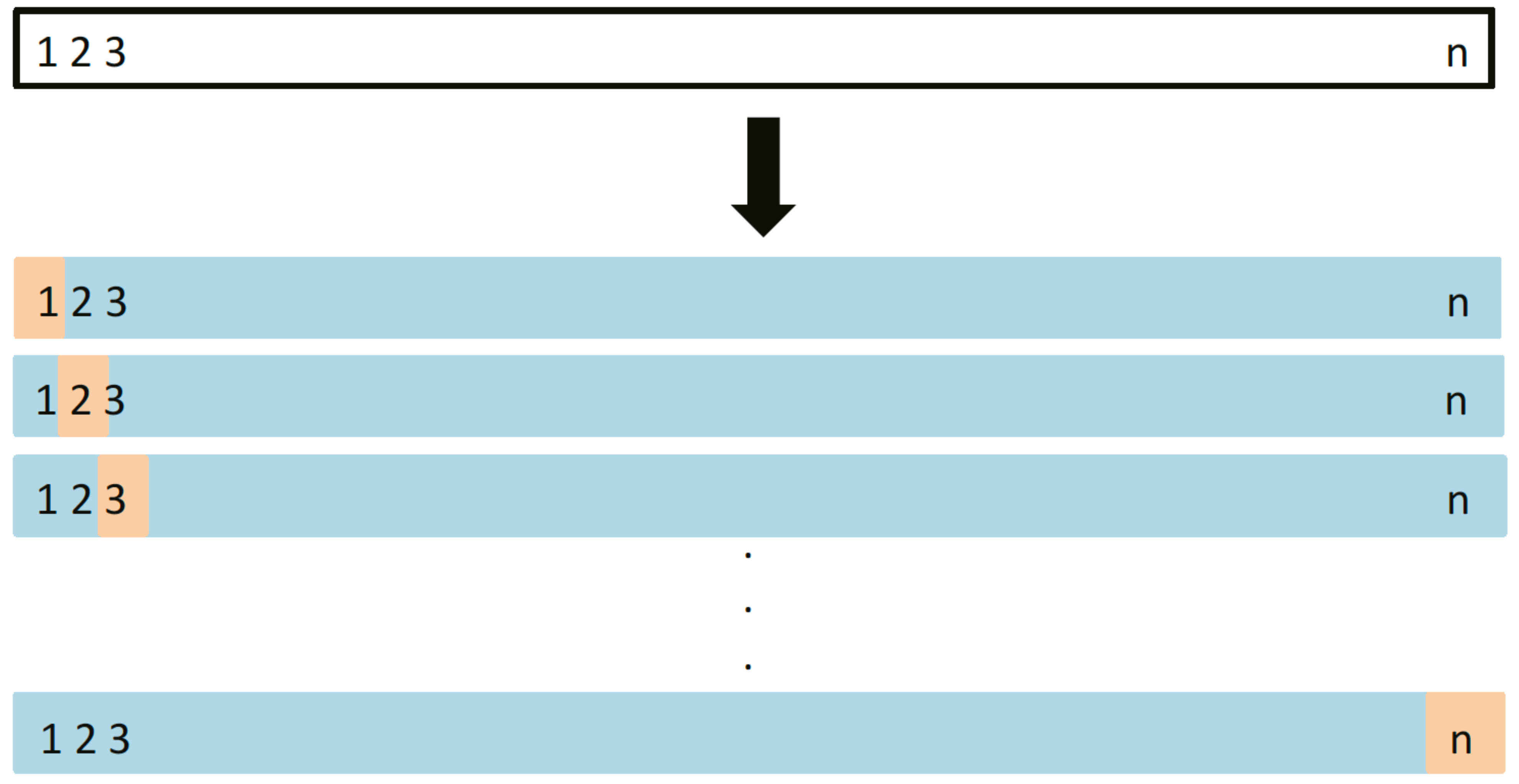
The validation set approach

Using the same synthetic dataset ($n = 100$, 50%-50% training-validation split)



- Validation estimate of test error can be highly variable depending on which observations were assigned to training and test sets
- Test error is overestimated as only some observations were used for training

Leave-one-out cross-validation (LOOCV)

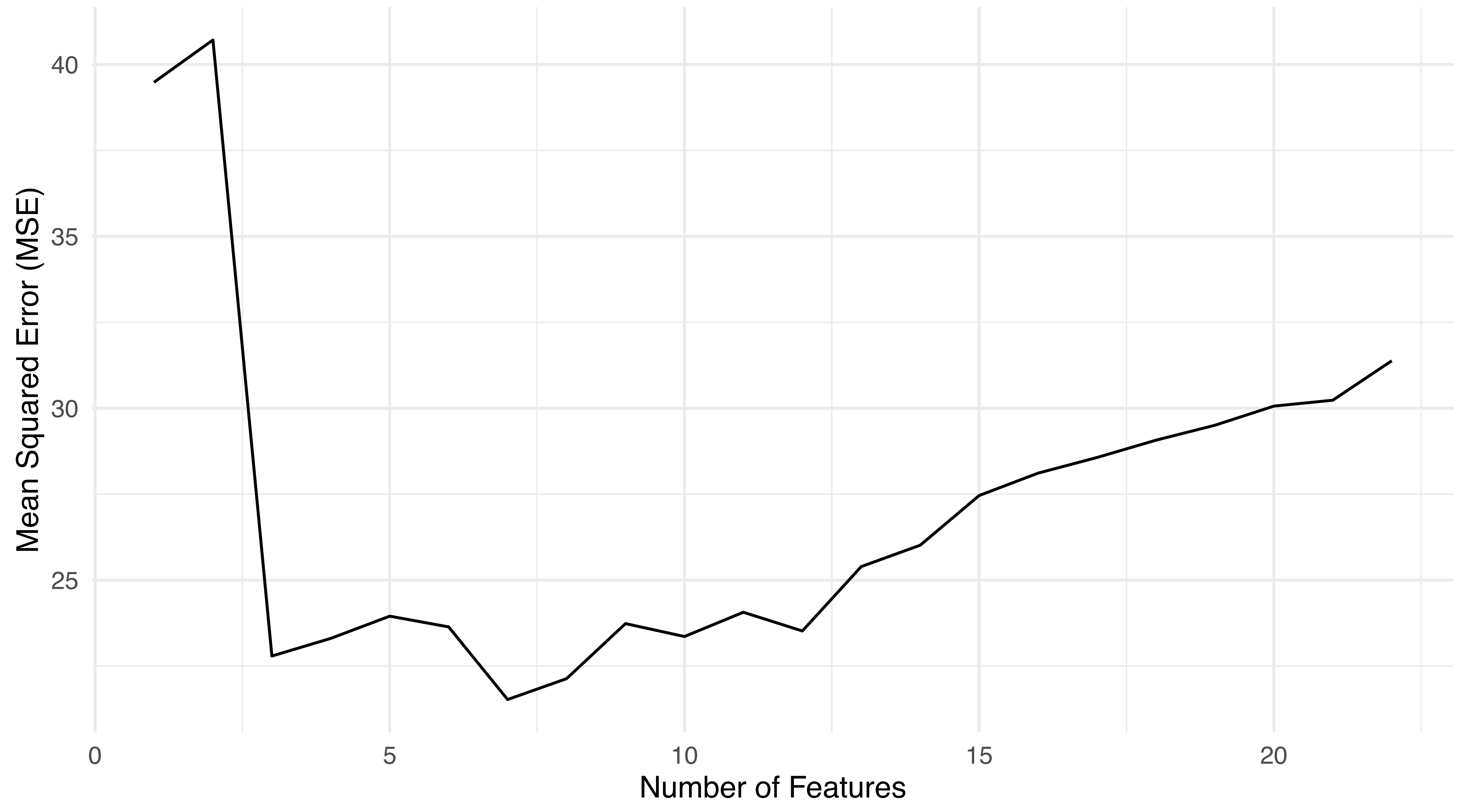


- Only one observation is used for validation
- LOOCV estimate for the test MSE is the average of the n error estimates

$$CV_n = \frac{1}{n}\sum_{i=1}^{n} MSE_i$$

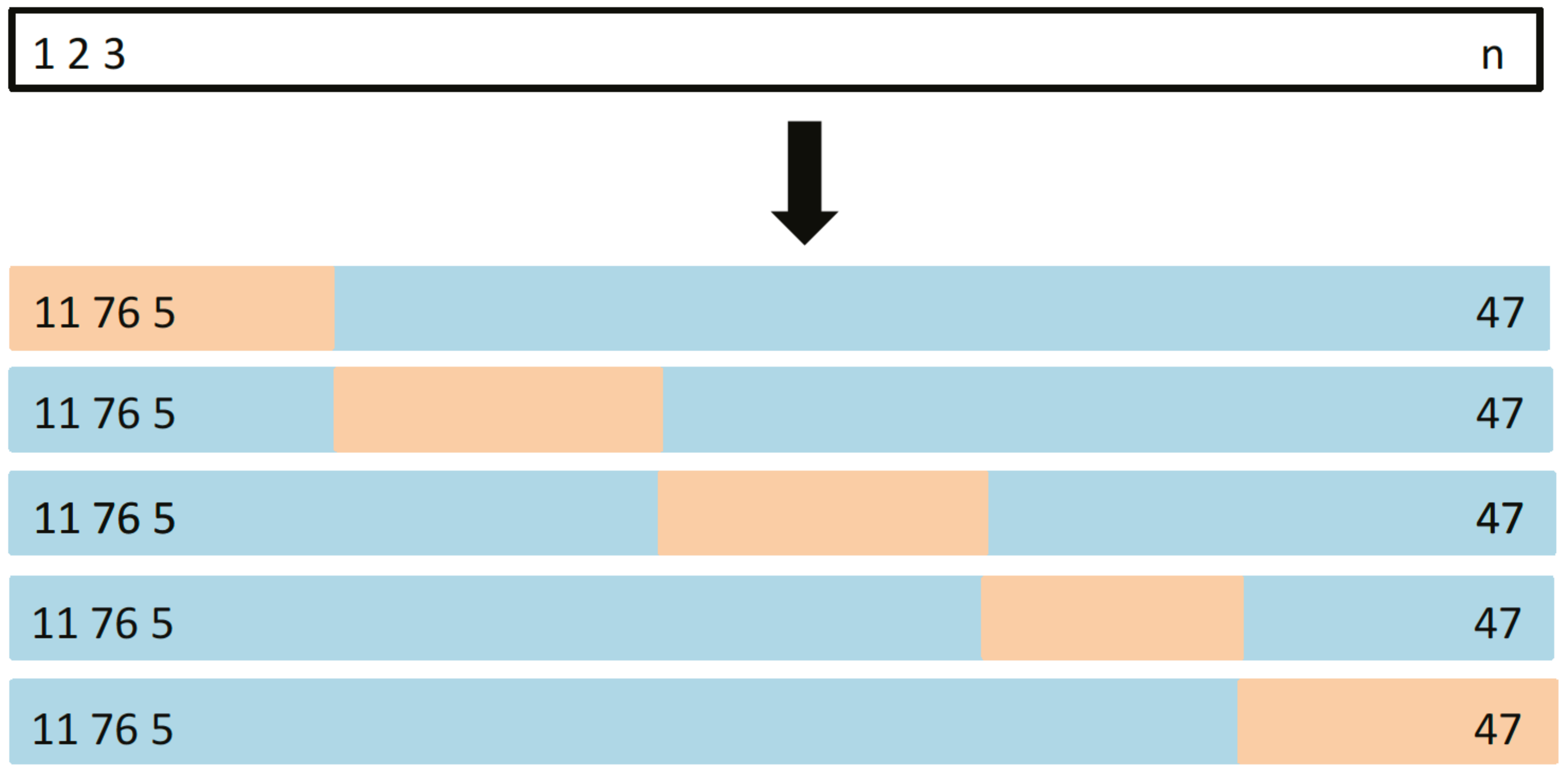
Leave-one-out cross-validation (LOOCV)

Using the same synthetic dataset ($n = 100$, LOOCV)



- Almost all the observations used for training, test error is not overestimated
- Computationally very expensive: model has to be retrained n times

k-fold cross-validation (k-fold CV)



- Observations are randomly divided in k groups
- For each fold $i = 1, \ldots, k$, the i-th fold is used as validation set and the remaining $k - 1$ folds are used as training set
- K-fold CV estimate for the test MSE is the average of the MSE computed for each validation set $(MSE_1, MSE_2, \ldots, MSE_k)$

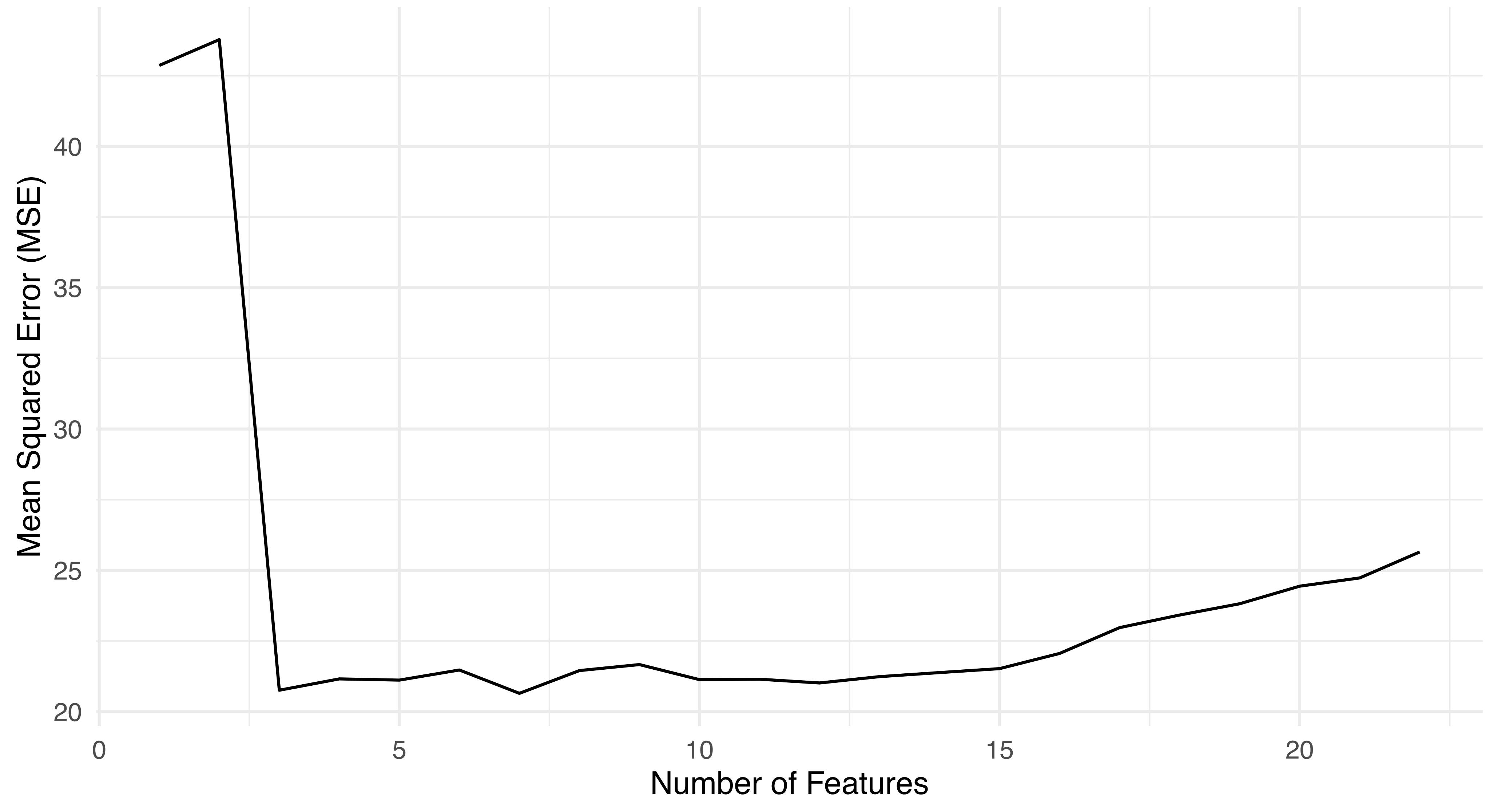
$$CV_k = \frac{1}{k} \sum_{i=1}^{n} MSE_i$$

What is the relationship between LOOCV and k-fold CV

- LOOCV is a spacial case of k-fold CV with $k = n$
- LOOCV is a spacial case of k-fold CV with $k = 1$
- There is no relationship between LOOCV and k-fold CV

k-fold cross-validation (k-fold CV)

Using the same synthetic dataset ($n = 100$, 10-fold CV)



- Similar results as with LOOCV but computationally less expensive

Quick recap on model complexity & bias-variance trade-off

Model complexity:

- Simple models (low complexity) may not capture all patterns in the data $\rightarrow$ High Bias
- Complex models (high complexity) may overfit the training data $\rightarrow$ High Variance

Bias-variance trade-off:

- High Bias: Underfitting, overly simple models, poor performance on both training and unseen data.
- High Variance: Overfitting, overly complex models, great performance on training data but poor generalisation to

Goal: Find the right balance between bias and variance to minimise generalisation error on unseen data.

Problems of complex models

Especially when we have a large number of features (p large compared to N), least square estimated can suffer from:

- *Low prediction accuracy*: high model complexity gives low bias but high variance on the predictions
- *Poor interpretability*: we do not know which predictors are really useful to explain the output

Setting some coefficients to zero can help to:

- Reduce the variance of predictions (at the price of increasing the bias)
- Help to identify important predictors

Subset selection

With subset selection we want to retain only a subset of the predictors and eliminate the rest from the model.

We can try a lot of different models, each containing a different subset of the predictors, and check which model is the best using cross-validation.

Problem: there are a total of 2^p models that contain subsets of p predictors!!

Subset selection

Three classical approaches to step-wise *subset selection*:

- *Forward selection*. Start from the null model (i.e. only intercept) and sequentially add to the model the variable that gives lowest RSS (i.e. higest improvement of the fit).
- *Backward selection*. Start with the full model (i.e. all variables) and sequentially remove the predictor with the largest p-value (i.e. lower impact on the fit).
- *Mixed selection*. Alternation of forward and backward steps to add variables that improves RSS while maintaining the p-value below a certain threshold.

Subset selection

Subset selection tends to:

- improve *interpretability*: only the most relevant predictors useful to explain the output are selected.
- still suffer from low *prediction accuracy*: high variance (discrete process in retaining and discarding predictors) often does not reduce prediction error.

Improving least square estimates with regularisation

- Least square estimates generally provide all non-zero coefficient.
- if $p > N$, solutions are not unique (i.e. multiple solutions with same minimum, often overfitting the data)

Need to constrain or regularize the estimation process.

Idea: shrink regression coefficients by imposing a penalty on their size.

Ridge regression

While least square fitting consist in estimating the coefficients $\hat{\beta}$ that minimise

$$RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij})^2$$

Ridge regression additionally penalises the sum of squares of the coefficients (L2 norm) by estimating the coefficients $\hat{\beta}^R$ that minimise

$$RSS + \lambda \sum_{j=1}^{p} \beta_j^2$$

Ridge regression: the meaning of λ

$$RSS + \lambda \sum_{j=1}^{p} \beta_j^2$$ where $\lambda \sum_{j=1}^{p} \beta_j^2$ is the *shrinkage penalty*

$\lambda \geq 0$ is a *tuning parameter*

- $\lambda = 0$ corresponds to the least square estimates
- For larger values of λ, coefficients $\beta_1, \ldots, \beta_p$ will shrink towards zero

Note: no shrinkage applied to the intercept β_0

Importance of feature scaling in shrinkage methods

Idea: make sure that features are on a similar scale

$$x_j = \frac{x_j - \mu_j}{\theta_j}$$

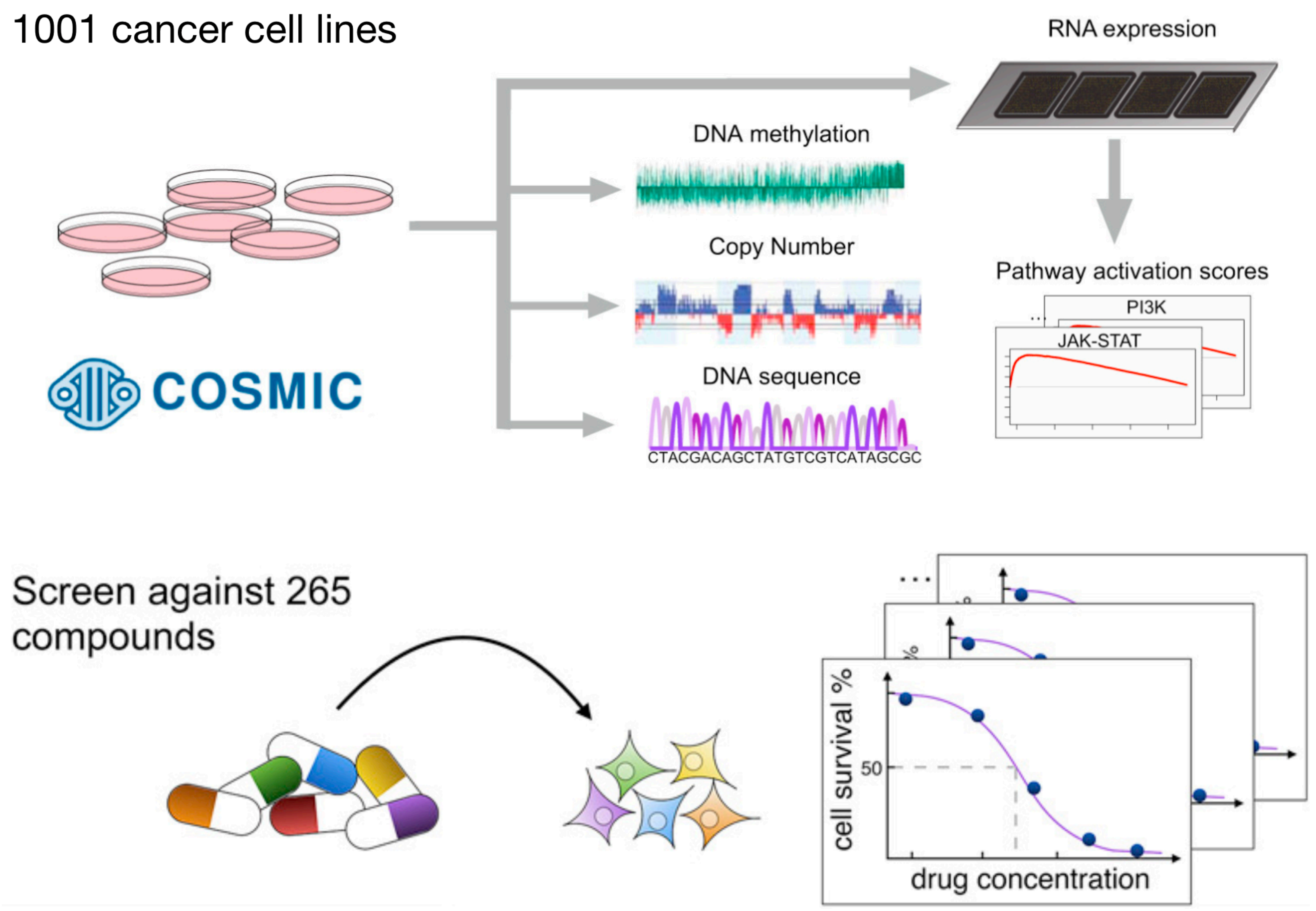
Where μ_i and θ_i are respectively the average and the standard deviation of all the values of feature j

If we standardise the predictors, the estimated intercept will be

$$\hat{\beta}_0 = \bar{y} = \sum_{i=1}^{n} \frac{y_i}{n}$$

Example: predicting drug response from genomic data

We will use as example data from the Genomics of Drug Sensitivity in Cancer (GDSC) project*



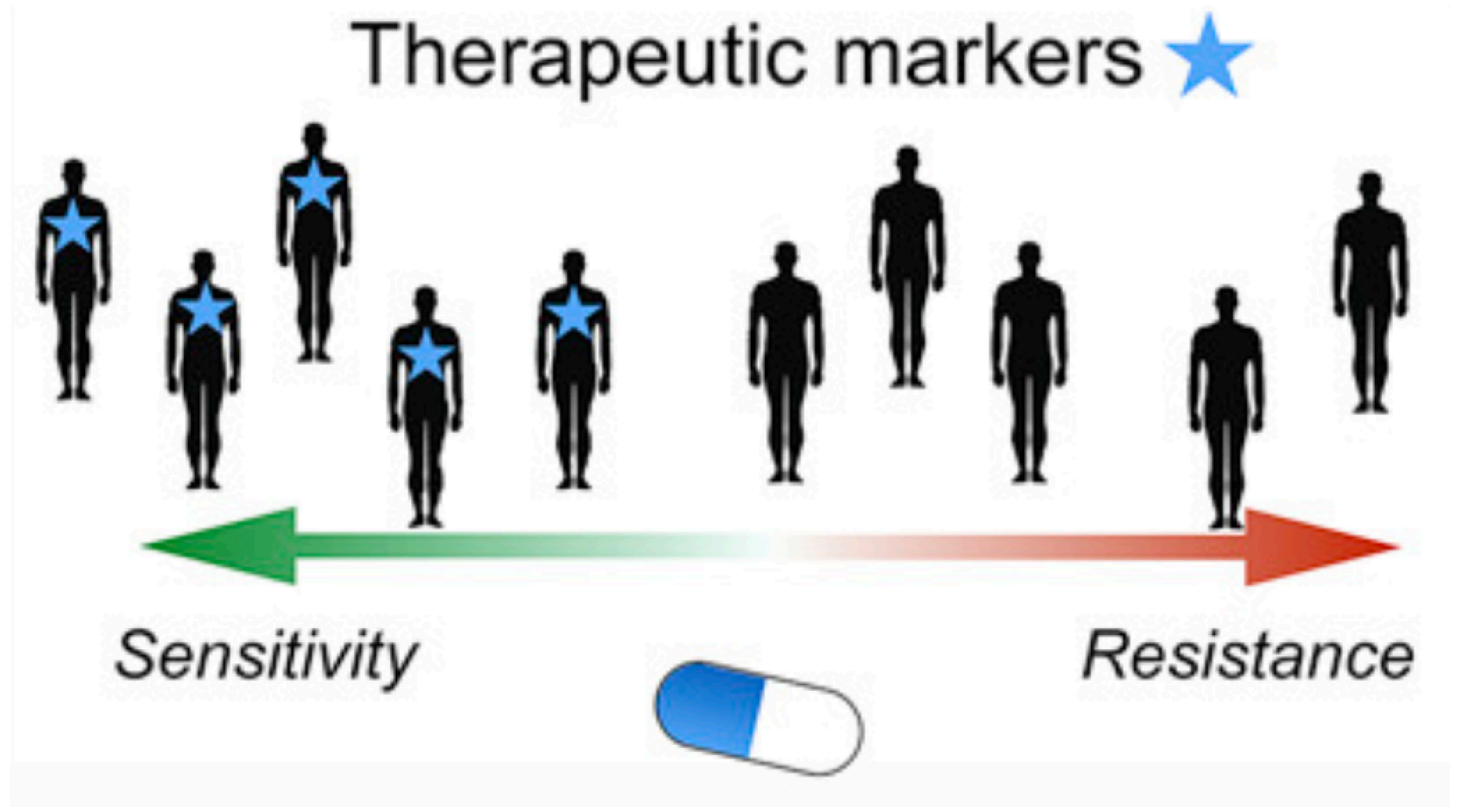
* https://www.cancerrxgene.org/ Iorio et al. “A landscape of pharmacogenomic interactions in cancer”, Cell, 2016

Example: predicting drug response from genomic data

Cancer patients tend to respond very differently to drugs. A big challenge in cancer research is to find ways to assign the optimal treatment to each patient.

In this example (and in your applied practical 1.3) we aim to:

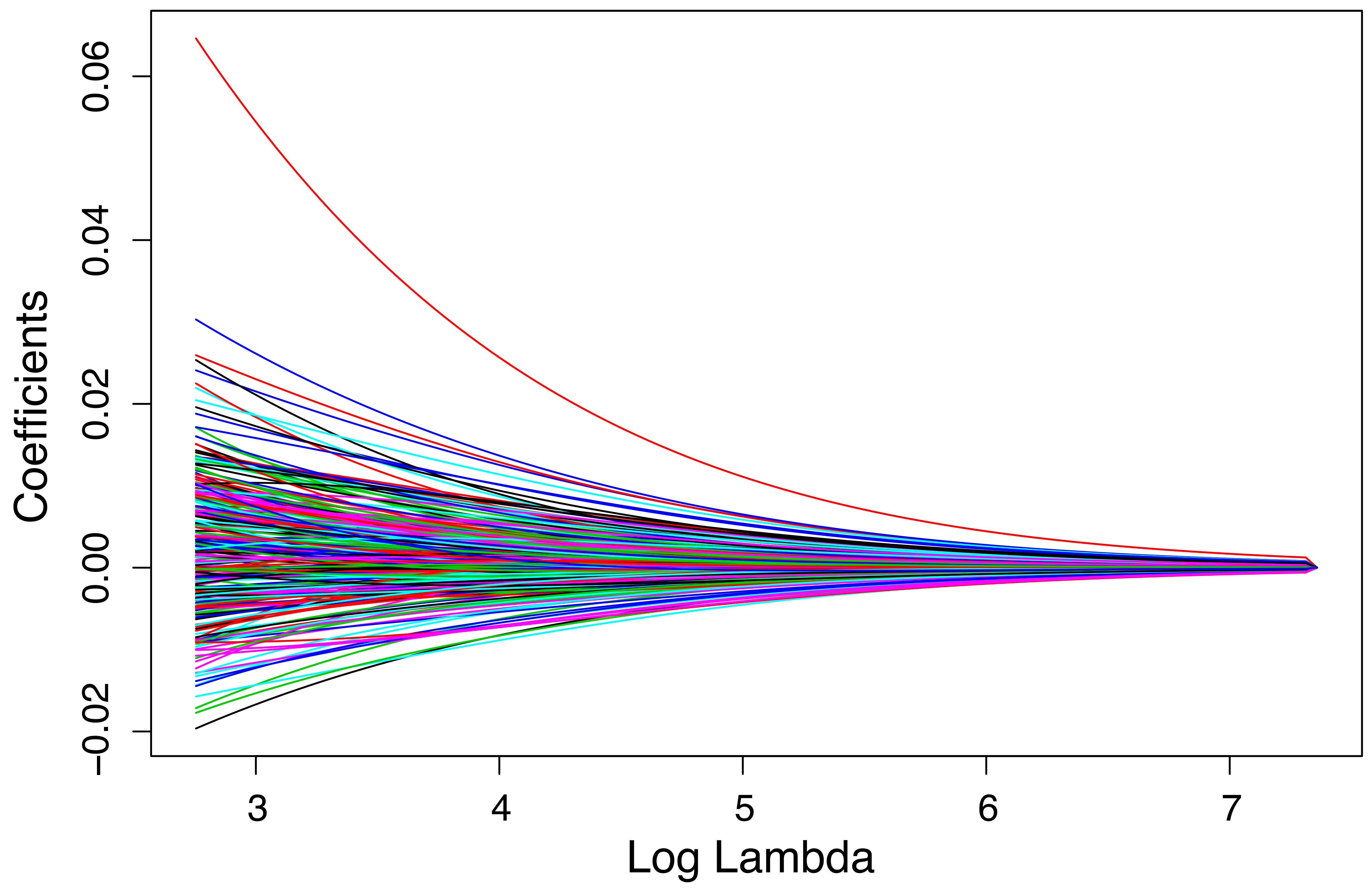
- Predict drug response from genomic data;
- Find biomarkers of drug sensitivity.



Here we will use 148 cell lines from four cancer types to predict response to YM155 drug using expression of 244 genes (Note: this is different from the practical).

Example: application of Ridge regression

When we apply *Ridge regression* to the GDSC example we obtain the following profile for the *Ridge regression* coefficients.

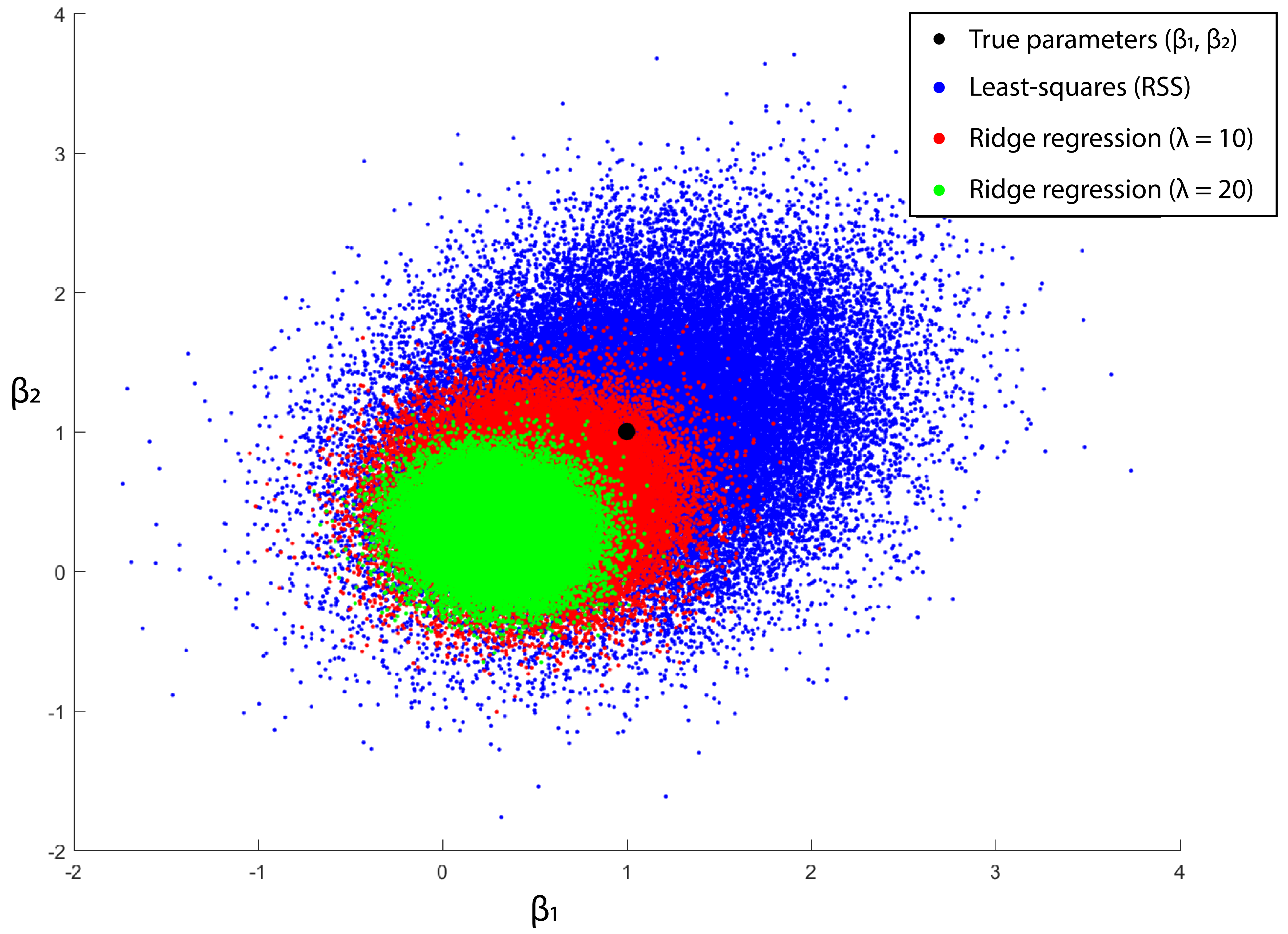


What is the λ that gives minimum variance in Ridge regression

- $\lambda = 1$
- $\lambda = Inf$
- $\lambda = 0$

The effect of regularisation on bias and variance

Effect of λ on bias and variance in a simulated example ($y_i = \beta_1 x_{i1} + \beta_2 x_{i2} + e_i, i = 1,\ldots,20$). The model is simulated 50k times, each time the noise e_i is randomly generated.



The Lasso

The Lasso penalises the sum of the absolute value of the coefficients (L1 norm) by estimating the coefficients $\hat{\beta}^L$ that minimise

$$\sum_{i=1}^{n}(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij})^2 + \lambda\sum_{j=1}^{p}|\beta_j| = RSS + \lambda\sum_{j=1}^{p}|\beta_j|$$

The Lasso: the meaning of λ

$$RSS + \lambda \sum_{j=1}^{p} |\beta_j| \quad \text{where } \lambda \sum_{j=1}^{p} |\beta_j| \text{ is the } \textit{shrinkage penalty}$$

$\lambda \geq 0$ is a *tuning parameter*

- $\lambda = 0$ corresponds to the least square estimates
- For larger values of λ, coefficients $\beta_1, \ldots, \beta_p$ will shrink exactly to zero, performing a continuous subset selection

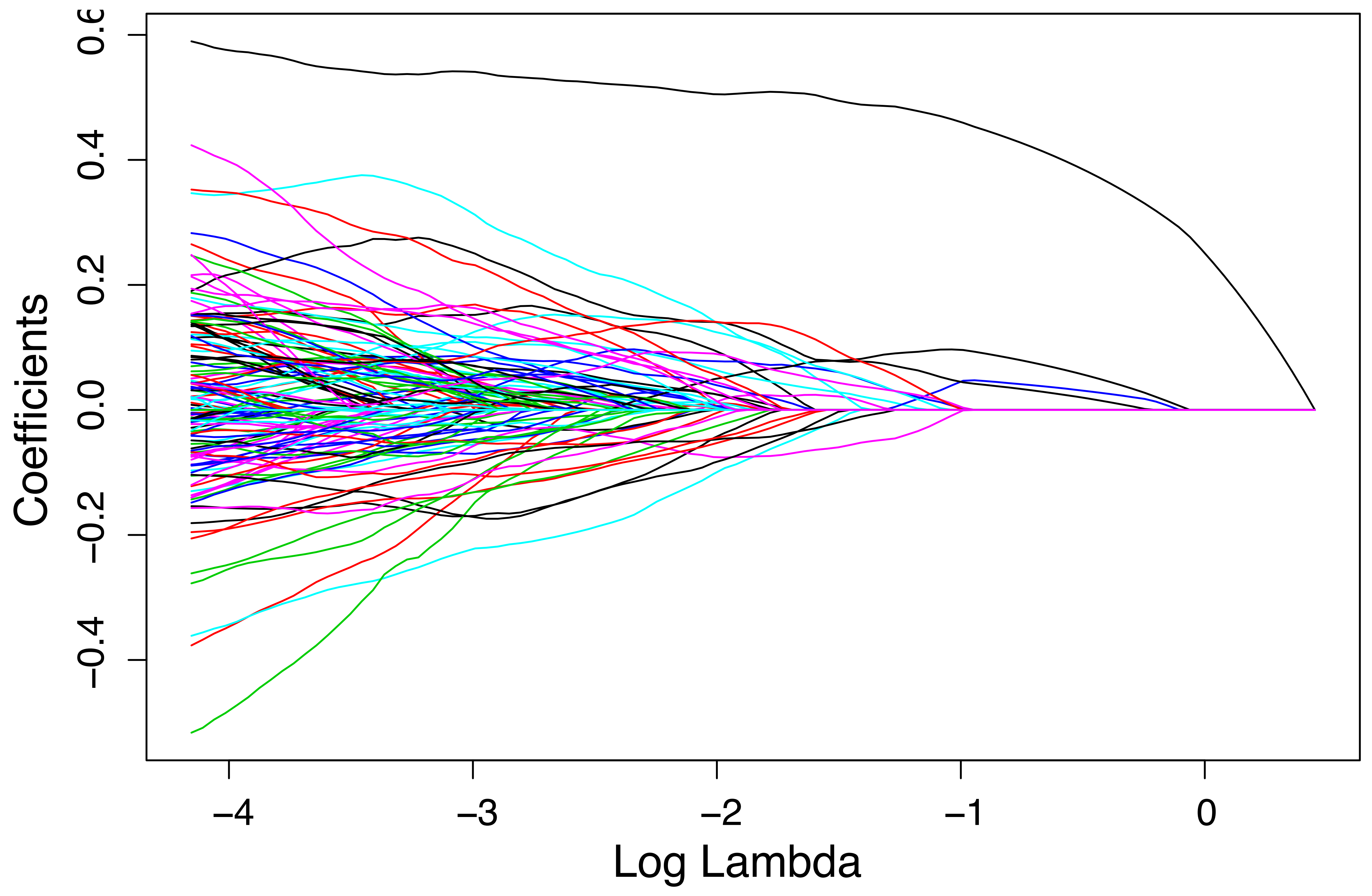
Note: no shrinkage applied to the intercept β_0

A high value of λ in the Lasso:

- Increases model complexity
- Does not affect model complexity
- Reduces model complexity

Example: application of Lasso regression

When we apply the Lasso to the GDSC example we obtain the following profile for the Lasso coefficients.



Another formulation for Ridge regression and the Lasso

An equivalent way of writing the Ridge and Lasso problem is:

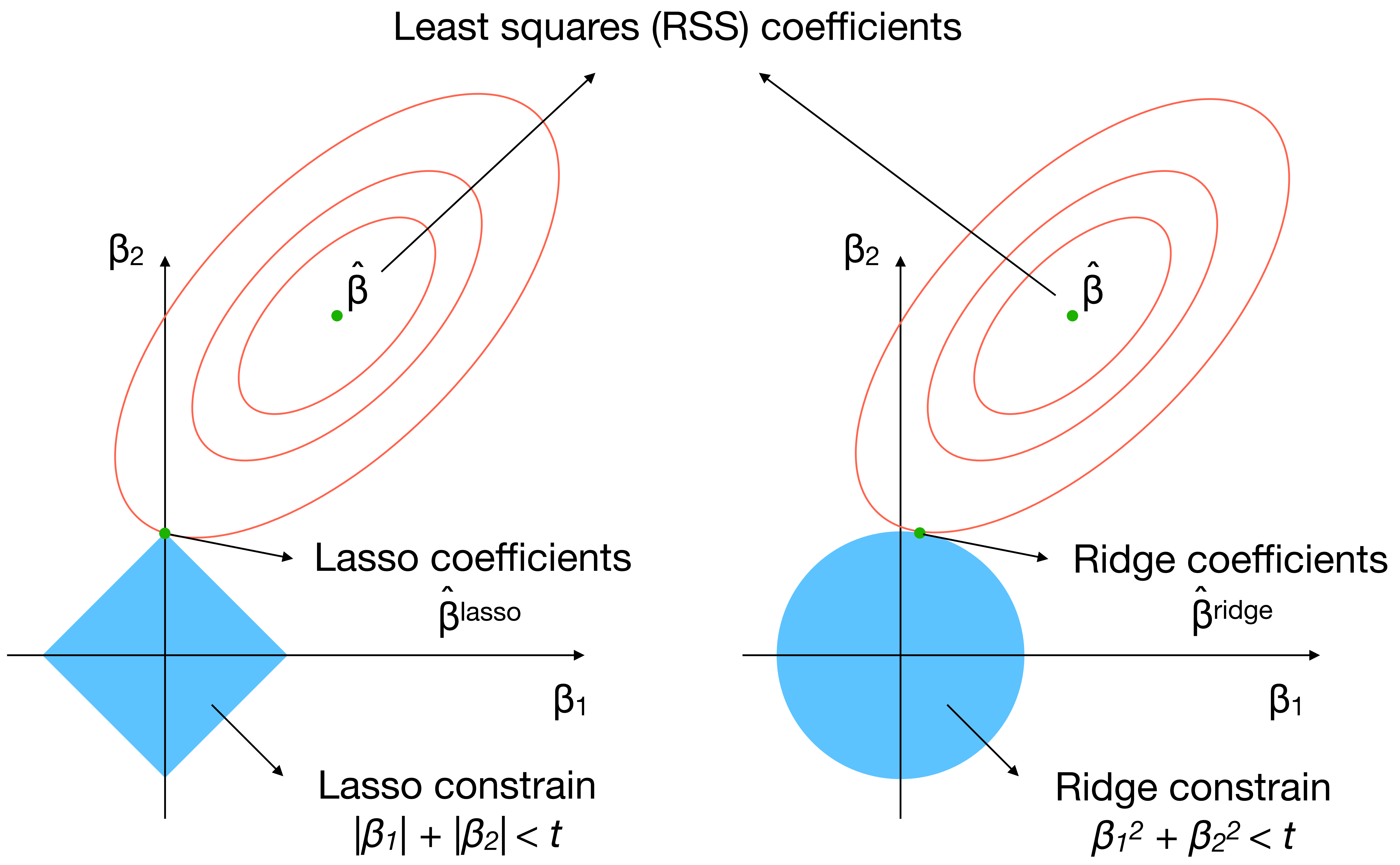
$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\}$$

Subject to:

- $\sum_{j=1}^{p} |\beta_j| \leq s$ for Ridge regression
- $\sum_{j=1}^{p} \beta_j^2 \leq s$ for the Lasso

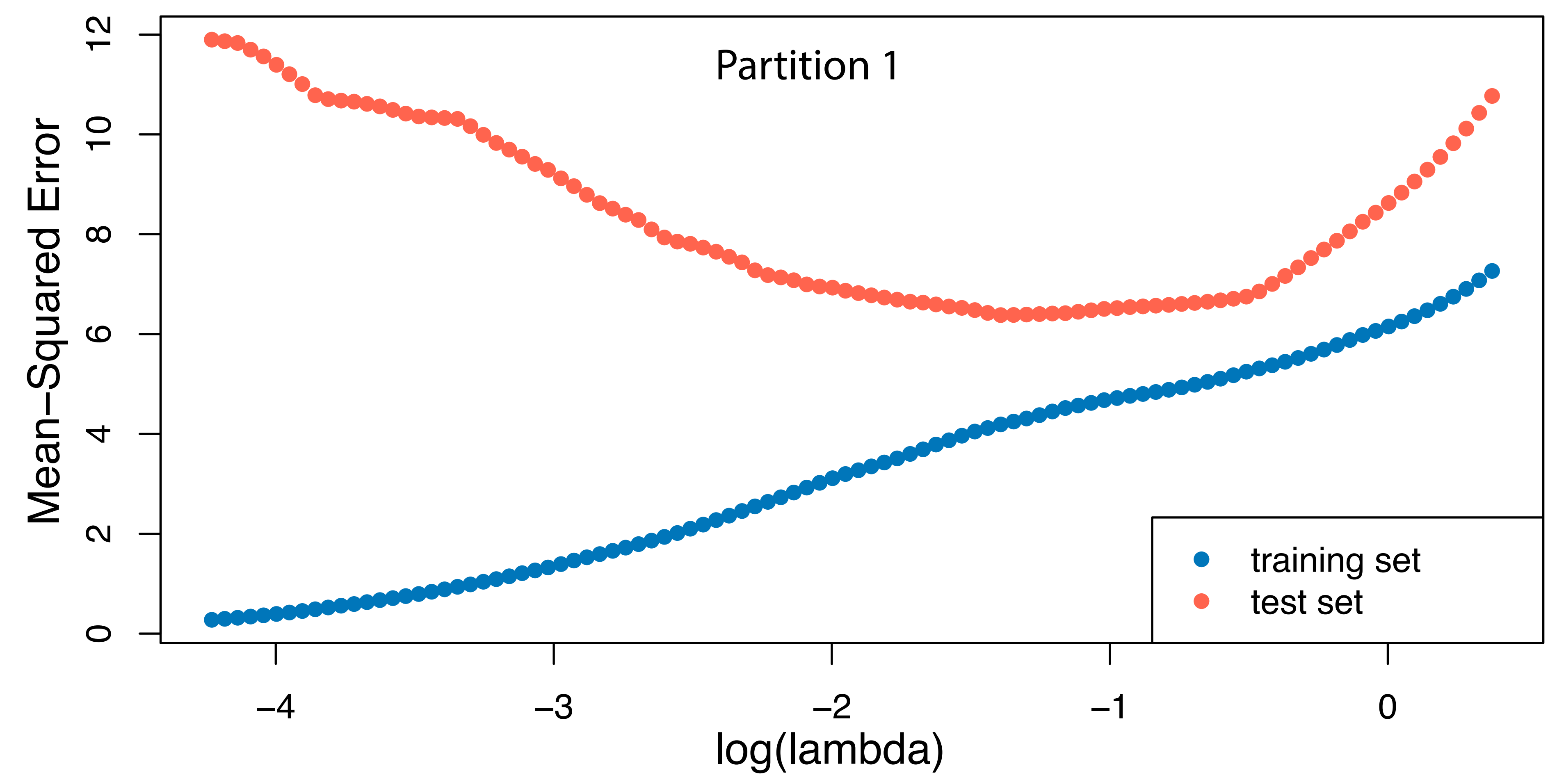
NOTE: There is a one-to-one correspondence between tuning parameter λ and s

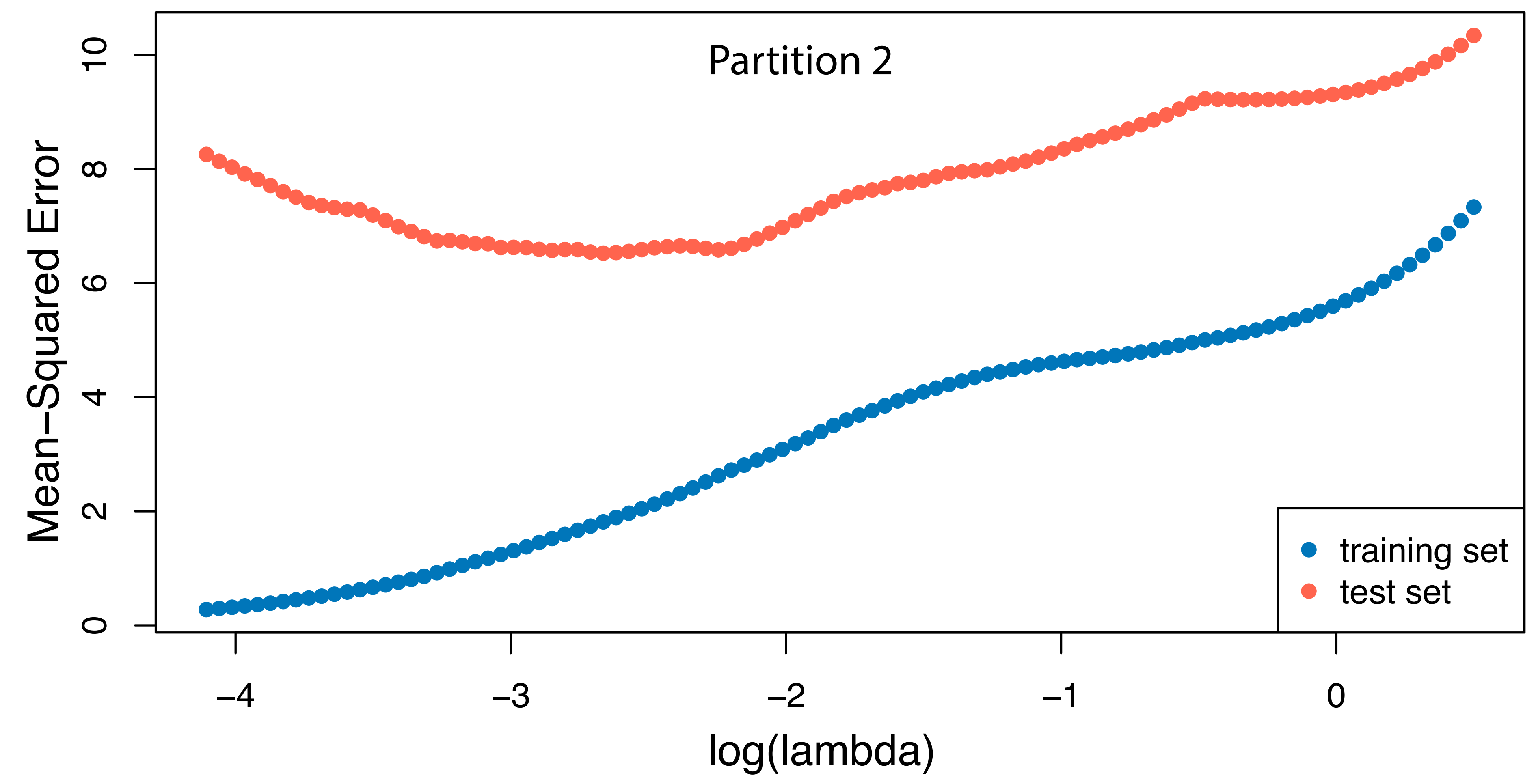
Visual interpretation of Ridge and Lasso regression



Selecting the tuning parameter λ

We want the model that provides the best predictions for the test set.





The choice can be sensitive to the partition used.

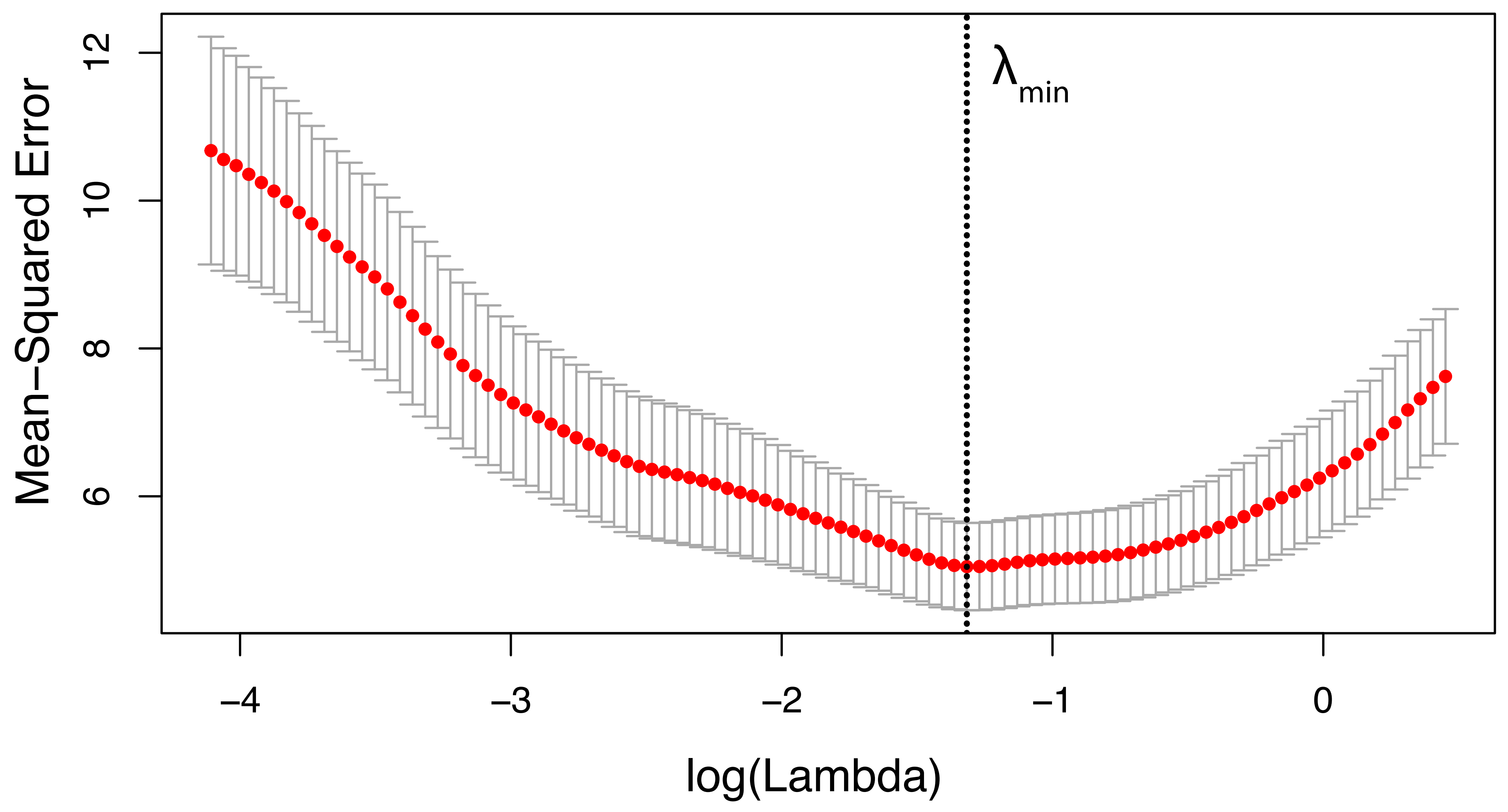
Example of two data partitions for GDSC dataset.

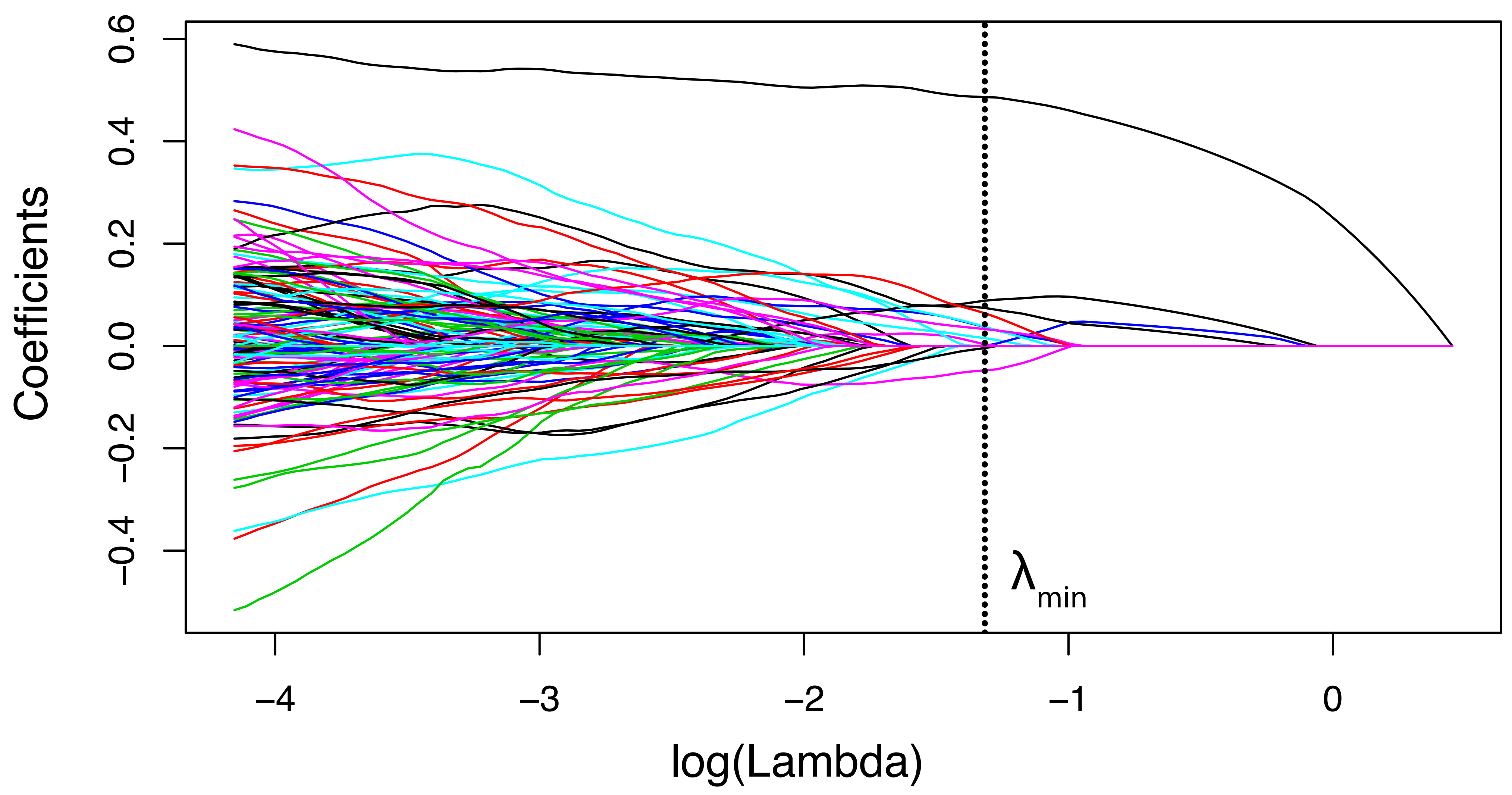
Selecting the tuning parameter λ

For more robust selection we can use cross-validation following these steps:

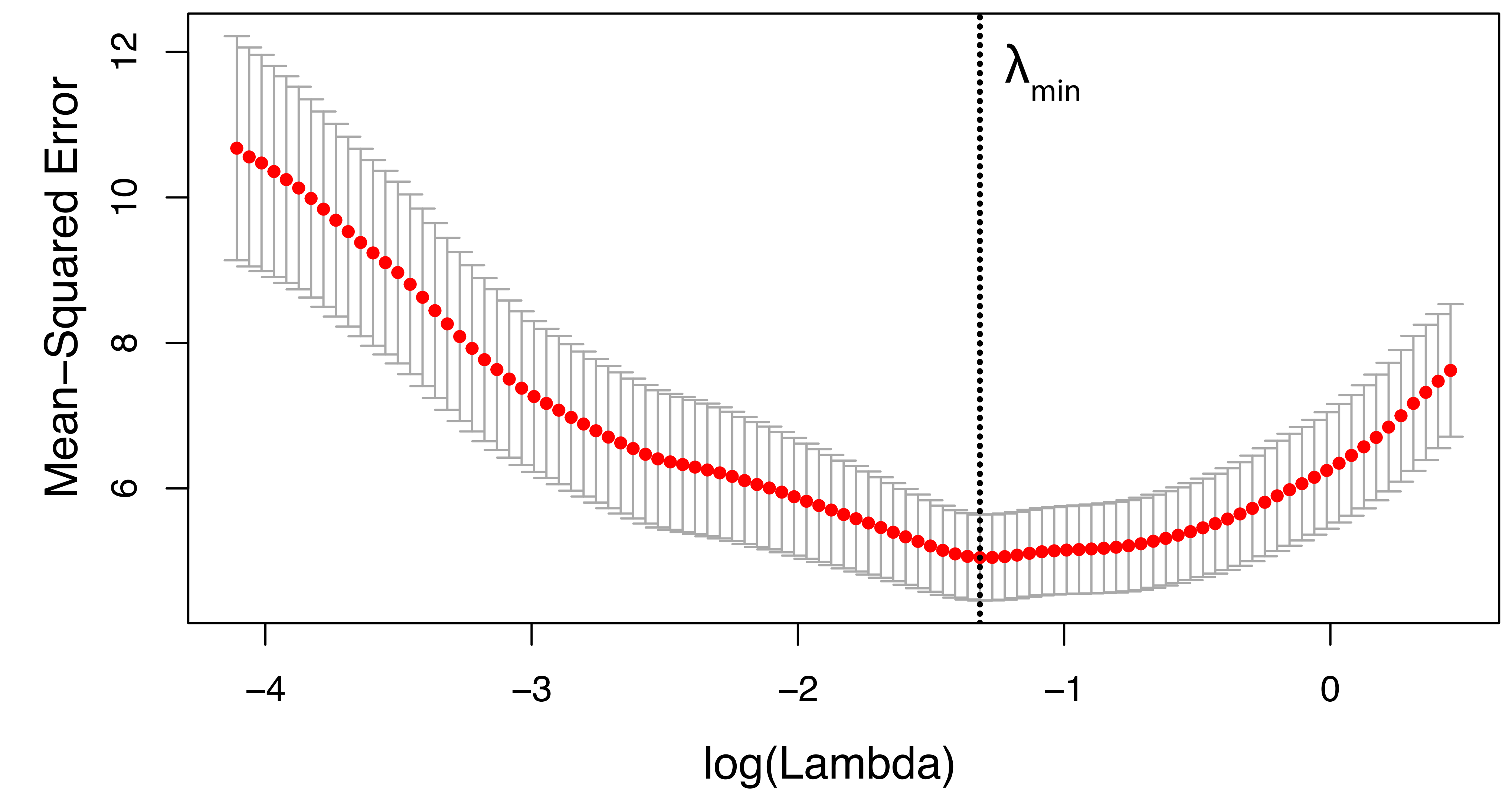
1. Choose a grid of λ
2. Optimize the model for each training set and compute the Mean-Squared Error (MSE) for each validation set
3. Choose the model with the smallest cross-validation error
4. Re-fit using all the available observations and the selected tuning parameter

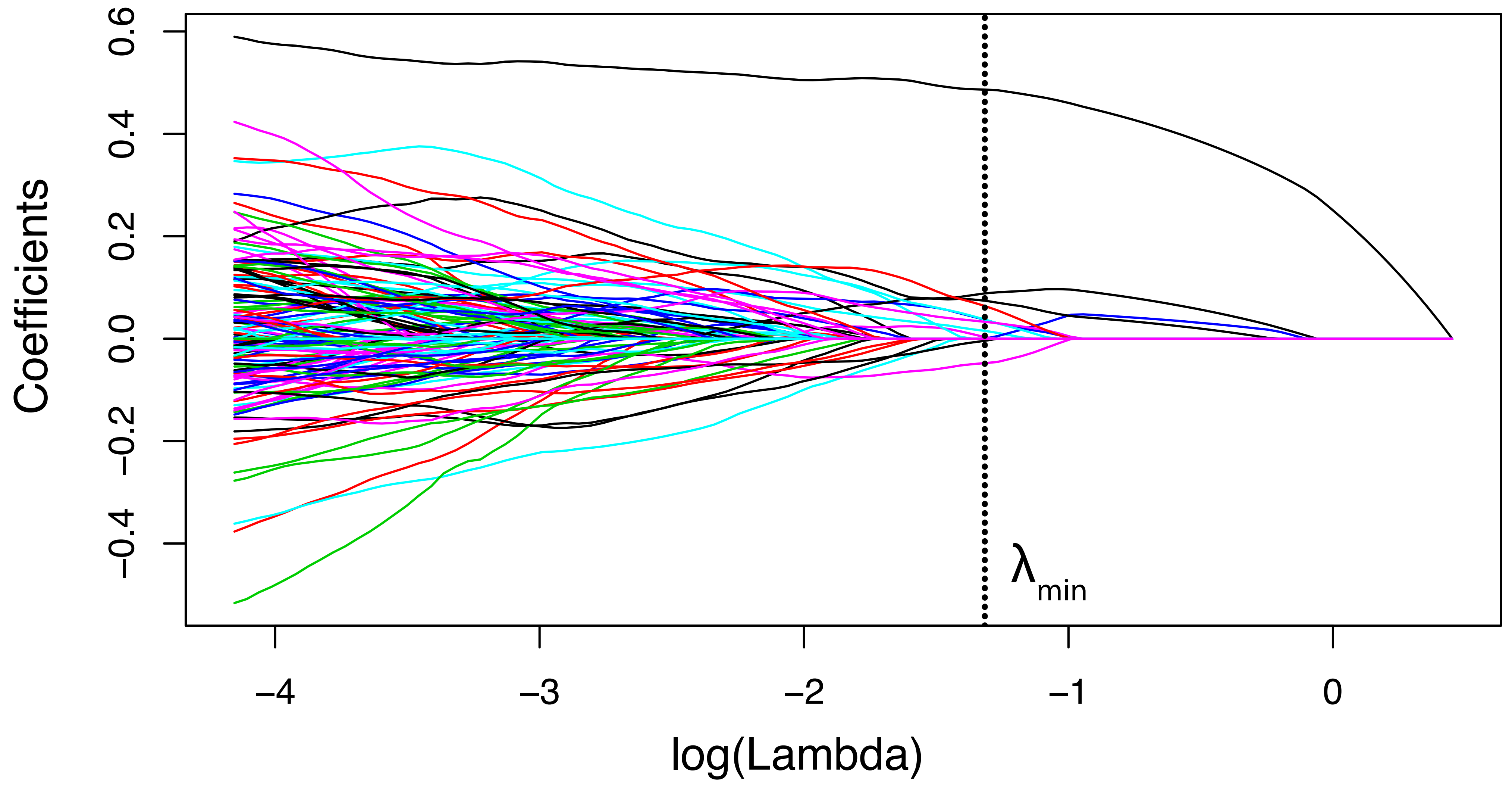
Selecting λ: GDSC example (10-fold CV)





How many predictors are used in the optimal model



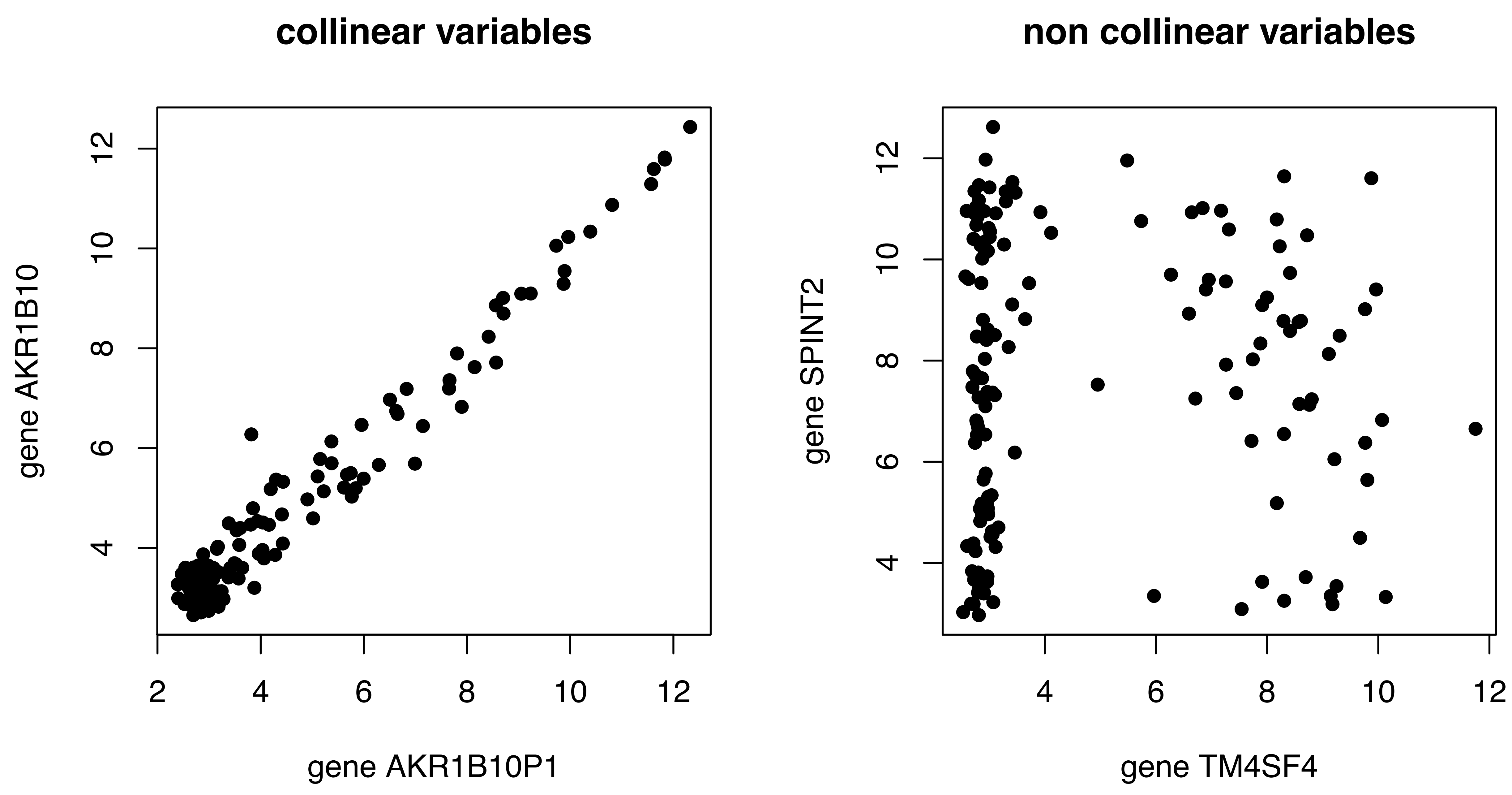


- 1
- 10
- 40
- 100

Collinearity

Two or more predictors can be closely related to each other and show high correlation.

This is quite common in biological and clinical data.



Collinearity

- The effect of collinear variables can be difficult to separate in the regression context.
- Collinearity reduces the accuracy of the estimates of the coefficients of the correlated variables.
- Using the Lasso, collinearity can result in an arbitrary selection of one or the other correlated features, depending on the data used for training.

Elastic Net regression

Elastic Net provide a compromise between Ridge regression and the Lasso, by selecting variables like the Lasso and shrinking together correlated variables like Ridge regression.

Elastic Net combines L1- and L2-norm in the shrinkage parameter, estimating the coefficients $\hat{\beta}^{EN}$ that minimise:

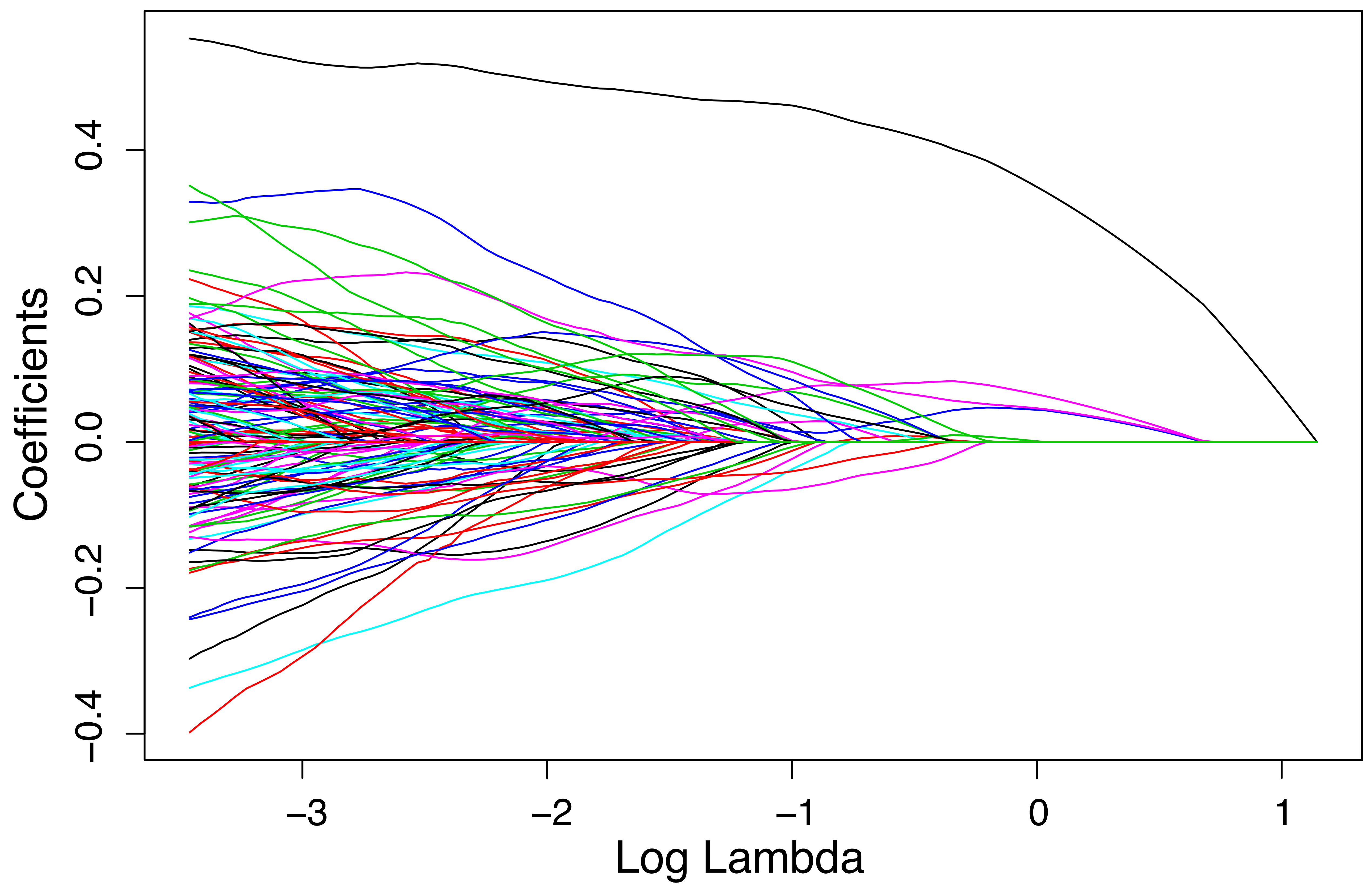
$$RSS + \lambda \sum_{j=1}^{p} ((\alpha\beta_j^2 + (1 - \alpha)|\beta_j|)$$

Where α is a tuning parameter. *Elastic Net* corresponds to:

- Ridge regression for $\alpha = 1$
- The Lasso for $\alpha = 0$

Example: application of Elastic Net

When we apply Elastic Net to the GDSC example (with $\alpha = 0.5$) we obtain the following profile for the Elastic Net coefficients.



Extension to linear classification

- **Same concepts apply:** The principles of generalization, cross-validation, and regularization (Ridge, Lasso, Elastic Net) apply not only to regression but also to linear classification models like logistic regression.
- **Objective function:** In classification, the objective function changes (from least squares to log-likelihood), but the role of regularization remains the same—to control model complexity and improve performance on unseen data.
- **Bias-variance trade-off:** Just as with linear regression, the bias-variance trade-off and the need for model generalization are key concerns in classification tasks.

Questions?