Exercise Solutions - Statistical Learning

Alireza Ghorbani

2025-03-20

Contents

1 Exercise 1 (Terminology, Standard Methods)

We consider a $n \times (p+1)$ data matrix containing the data from n individuals: a target variable Y and p covariates $X_1, ..., X_p$.

(a) Which terms are also commonly used in the literature to denote the target variable and the covariates? (give at least two answers for each).

(b) Which data does a prediction rule take as argument? What does it return?

(c) Suppose we want to derive a prediction rule to predict the birth weight at term based on 7 ultrasound measurements made in the 37th week of pregnancy (such as head circumference, femur length, etc) and use a dataset including the data from $n = 500$ infants for this purpose.

(c.1) Which straightforward statistical approach could be used to derive a prediction rule in this setting? Write the corresponding model.

(c.2) Briefly sketch how you would implement this analysis in R.

(c.3) What are its potential limitations?

(d) Answer the questions (c.1), (c.2) and (c.3) again for the case of the binary target variable "birth defect yes/no".

(e) What is a training/learning dataset? And a test dataset?

Answer:

(a) **Target variable** Also called dependent variable or response variable. **Covariates** Also called predictors, independent variables, or features.

(b) The **arguments** are Covariates $X_1, X_2, \ldots, X_p$ and the **return value** is the predicted value $\hat{Y}$ for the target variable.

(c.1) Linear regression can be used with a model like:$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_7 X_7 + \epsilon$, where Y is birth weight, $X_1, \ldots, X_7$ are the ultrasound measurements, and ϵ = error term.

(c.2)

```
# Combine into a dataframe
data <- cbind(Y, X)

# Fit linear regression model
model <- lm(Y ~ ., data = data)
```

(c.3) Limitations are: 1. the model assumes linear relationships between covariates and birth weight and it ignores non-linear or interaction effects unless explicitly modeled. 2. the model is sensitive to outliers and multicollinearity.

(d.1) Here logistic regression can be used with the model: $\log\left(\frac{P(Y=1)}{1-P(Y=1)}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_7 X_7$, where $Y = 1$ (birth defect) or 0 (no birth defect).

(d.2) Steps:
1. Fit a logistic regression model using the 7 covariates.
2. Interpret coefficients as log-odds ratios and evaluate classification accuracy.

```
# Fit logistic regression model
logistic_model <- glm(Y ~ X, family = binomial)
```

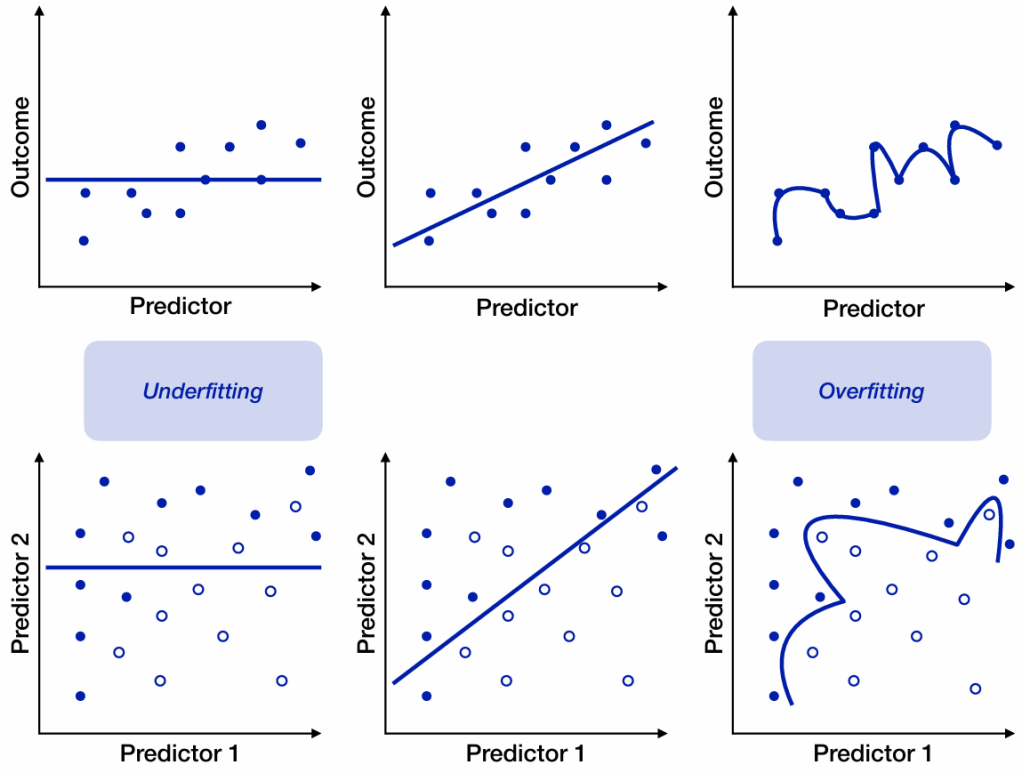
(d.3) Limitations: 1. Again we assume linearity in the log-odds. 2. Poor performance with imbalanced classes (rare birth defects).

(e) Training/learning dataset is a subset of the whole dataset used to train the model (estimate parameters). ON the other hand, test dataset is another subset of the whole dataset used to evaluate model performance on unseen data.

---

2 Exercise 2 (Overfitting, Underfitting)

This exercise is about the compromise that has to be found between overfitting and underfitting prediction rules.



(a) Which type of target variable is considered in the top row? And in the bottom row?
(b) Explain the terms "overfitting" and "underfitting" based on the figure.
(c) Why do you expect the prediction rules displayed in the middle column to yield better predictions on an independent test dataset than the prediction rules displayed in the left and right columns?

Answer:

(a) In top row the target variable is **continuous** (e.g., regression problem) and in bottom row the target variable is **binary/categorical** (e.g., classification problem).

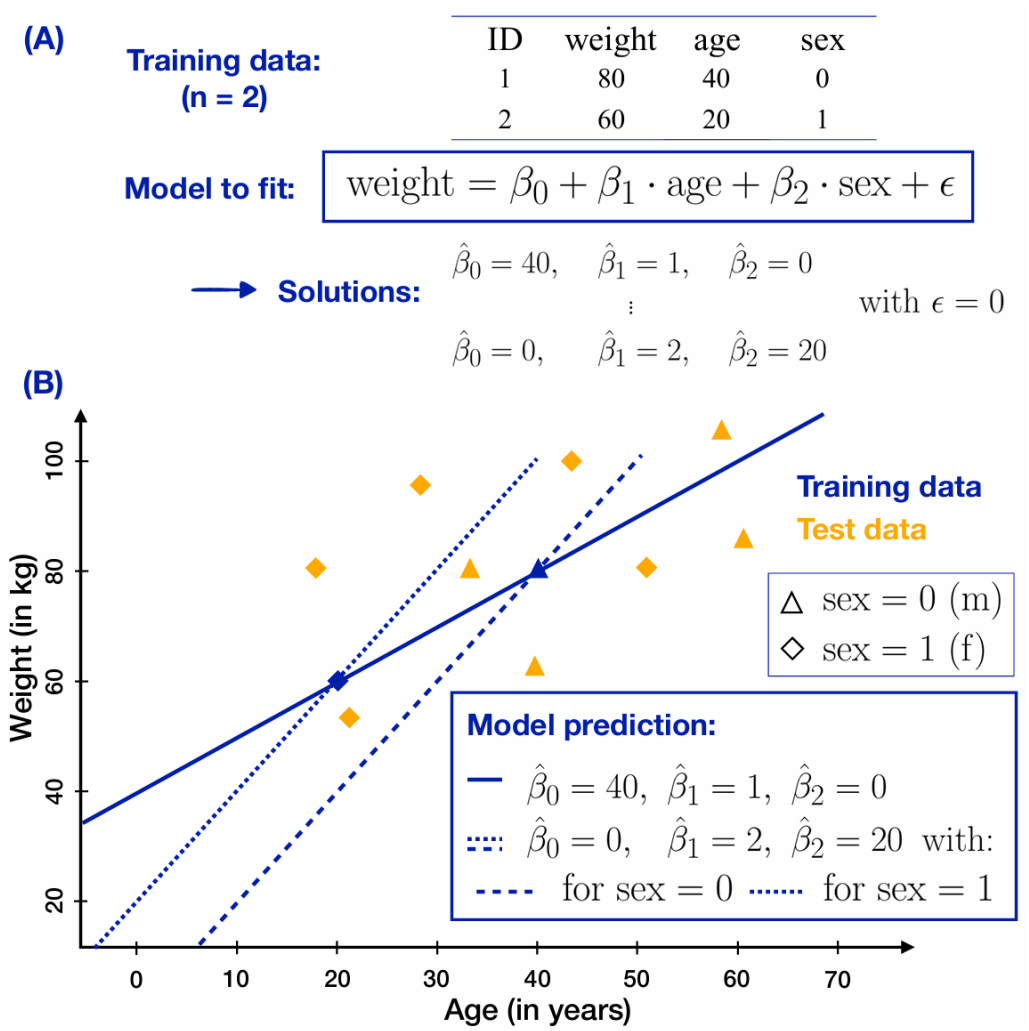
(b) **Overfitting** occurs when a model is overly complex, capturing noise or random fluctuations in the training data ("fits the training data too closely"). This leads to poor generalization on new data (high variance). In the figure, the right column shows overfit models. **Underfitting** occurs when a model is overly simplistic and fails to capture the underlying pattern in the data (high bias). In the figure, the left column shows underfit models.

(c) **Middle column models** balance bias and variance, avoiding extremes of being under or overfit. **Left column (underfit)** models have high bias due to oversimplification; fails to capture true relationships, leading to systematic prediction errors. **Right column (overfit)** models have high variance due to excessive complexity; adapts to training noise, leading to inconsistant predictions on new data. Therefore, **middle column** models are Optimally balances flexibility and simplicity, capturing true patterns without overreacting to noise. This minimizes test error (generalization error) on independent datasets.

---

3 Exercise 3 (Dimension Problems)

We consider the case of a training dataset with size n smaller than $p+1$, where p is the number of candidate predictor variables. To explain the problems occurring when $n < p+1$, we will consider a trivial toy example with $n = 2$ and $p = 2$, as displayed in the figure.



(a) In the linear regression model displayed in panel (A), if we want to estimate β_0, β_1, β_2 such that $\epsilon = 0$ for the $n = 2$ patients, how many equations and how many unknown do we have? Briefly explain your answer.

(b) Explain all components of the panel (B).

(c) Are the two solutions displayed in (B) the only ones that yield $\epsilon = 0$ for the $n = 2$ patients? If no, derive another possible solution.

(d) Can the training data (of size $n = 2$) tell us which of these solutions yielding $\epsilon = 0$ will be better on test data?

(e) Give three concrete examples of situations with a prediction problem in the $n < p$ setting (sample size (much) smaller than the number of covariates).

Answer:

(a) Number of equations is $n = 2$ (equation per observation) and number of unknowns is $p + 1 = 3$ $(\beta_0, \beta_1, \beta_2)$. The system is underdetermined because $n < p + 1$, resulting in infinitely many solutions. In a **linear regression model**, using **least squares** the regression coefficients β are estimated by $\hat{\beta} = (X^T X)^{-1} X^T y$ where X is the **design matrix** containing the data (with n observations and $p + 1$ predictors). The columns of the matrix X are **linearly dependent**, meaning that there are fewer than $p + 1$ independent columns. This implies that $X^T X$ does not have full rank of $p + 1$ and is thus **singular**. In linear algebra, a matrix is invertible (non-singular) if and only if it has full rank. For $X^T X$. When $X^T X$ is singular (i.e., it has rank less than $p + 1$), the inverse $(X^T X)^{-1}$ does not exist because its **determinant is zero**. Thus, we cannot compute $(X^T X)^{-1}$.

(b) Components of Panel (B): **Training Data** has two observations ($n = 2$) with covariates `age`, `sex`, and target `weight`. **Test Data** has 9 observation visualized as triangles/diamonds for sex. Two models solutions achieving perfect training fit ($\epsilon = 0$):

1. $\hat{\beta}_0 = 40, \hat{\beta}_1 = 1, \hat{\beta}_2 = 0$ for the line
2. $\hat{\beta}_0 = 0, \hat{\beta}_1 = 2, \hat{\beta}_2 = 20$ for the two dashed line each for one sex. It shows how different coefficient sets produce identical training fits but divergent predictions for test data.

(c) No, many solutions exist. Assume $\beta_1 = t$ (free parameter). Solve for β_0 and β_2:

$$\begin{cases} 80 = \beta_0 + 40\beta_1 + 0\beta_2 \\ 60 = \beta_0 + 20\beta_1 + 1\beta_2 \end{cases}$$

Subtract equations:

$$20 = 20\beta_1 - \beta_2 \implies \beta_2 = 20\beta_1 - 20$$

Let $\beta_1 = 0.5$, then $\beta_2 = -10$ and $\beta_0 = 80 - 40(0.5) = 60$.

(d) No, the training data ($n = 2$) cannot determine which solution generalizes better. All models achieve $\epsilon = 0$ on training data, but test performance depends on the true (unknown) relationship.

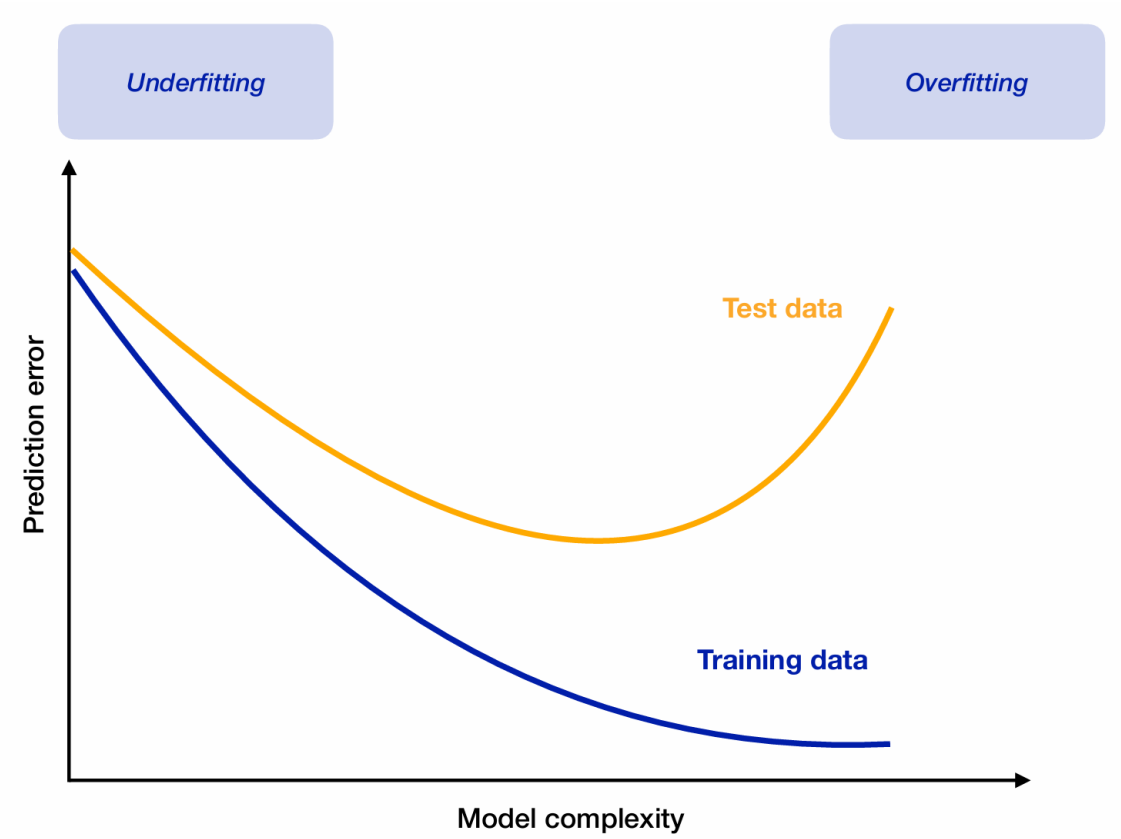
(e) Examples of $n < p$ settings:

1. **Genomics**: Predicting cancer subtypes using $p = 20{,}000$ gene expressions from $n = 50$ patients.
2. **Neuroimaging**: Classifying brain disorders using $p = 10{,}000$ MRI voxels from $n = 30$ scans.

---

4 Exercise 4 (Estimating Prediction Error)

We suppose that we have constructed a prediction rule $\hat{f}$ based on a training dataset of size n.



(a) Which straightforward error measure can be used to evaluate whether $\hat{f}$ predicts Y well in the case of a binary Y? And in the case of a continuous Y?

(b) Explain based on the figure from Exercise 3 and based on the following figure why it is not a good idea to estimate the prediction error of $\hat{f}$ by computing the error measures mentioned in (a) that results when simply applying $\hat{f}$ to the training dataset (that was used to construct it).

(c) Explain based on this figure why one should try to construct $\hat{f}$ such that it neither overfits nor underfits the training data.

Answer:

(a) In binary cases Y **classification error rate** which is the proportion of misclassified observations. For continuous Y **mean squared error (MSE)**; MSE $= \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{f}(X_i))^2$.

AUC also look for this?

(b) In exercise 3, When $n < p$, models can perfectly fit training data ($\epsilon = 0$), but solutions are non-unique and likely overfit. Training error $= 0$ does not reflect test error. Also in this question, training error decreases monotonically with model complexity, but test error follows a U-shape. Using training error underestimates true prediction error, especially for complex models.

(c) To select **Optimal model complexity** and to avoid **underfitting** which is when model is too simple to capture patterns, leading to high test error and **overfitting** which is when model adapts to noise, causing high test error. Therefore it is better to minimizes test error.

---

5 Exercise 5 (Classification Methods, General)

In this exercise we consider (supervised) classification methods, i.e. methods to construct $\hat{f}$ in the case of a categorical target variable Y.

(a) Explain the basic principle (2-3 well-chosen sentences) of the following classification methods in the case of binary classification (i.e. for a binary Y):

(a.1) logistic regression

(a.2) linear discriminant analysis

(a.3) nearest neighbors

(a.4) classification trees

(b) Which of the methods outlined in (a) can in principle be applied to data with more covariates than observations ($p > n$)?

(c) Imagine you have a dataset with $n = 100$ and $p = 1,000$ and a collaborator saying that you should use linear discriminant analysis to construct $\hat{f}$ because it was shown to perform well in a previous study. The collaborator says you just have to (i) perform, for each covariate, a two-sample t-test to test the null-hypothesis that the mean of this covariate is equal in both groups ($Y = 0$ and $Y = 1$), (ii) select the 5 covariates (out of 1,000) with the smallest p-values, and (iii) proceed with linear discriminant analysis using these 5 covariates. Explain at least one inconvenience of this approach.

Answer:

(a.1) **Logistic Regression** models the probability of the target variable $Y = 1$ as a function of the covariates $X_1, X_2, \ldots, X_p$, where the probability is constrained to the interval $[0, 1]$. It uses the logistic function to transform the linear combination of the predictors into a probability:

$$\log\left(\frac{P(Y=1)}{1-P(Y=1)}\right) = \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p.$$

Logistic regression estimates the log-odds of the outcome as a linear combination of the predictors, making it a powerful tool for classification in the binary case. It is particularly useful when we expect a relationship between the covariates and the log-odds of the outcome.

(a.2) **Linear Discriminant Analysis** (LDA) assumes that the features follow a Gaussian (normal) distribution within each class, and that the classes share the same covariance matrix. LDA finds the linear combination of the predictors that maximizes the separation between the means of the different classes, while minimizing the within-class variance. The goal is to identify a projection of the data that best discriminates between the classes. LDA is effective when the normality assumption holds and can provide a probabilistic interpretation of class membership.

(a.3) **Nearest Neighbors** method (often k-nearest neighbors or KNN) classifies a data point based on the majority class among its k-closest neighbors in the training set. The distance between points is typically measured using Euclidean distance, but other distance metrics can also be used depending on the data's characteristics. KNN is a **non-parametric** method, meaning it makes fewer assumptions about the underlying distribution of the data. While intuitive and powerful, KNN can suffer from the "curse of dimensionality" when the number of features is large, as distances in high-dimensional spaces become less informative.

(a.4) **Classification Trees** recursively partition the feature space into distinct regions, where the observations in each region are as homogeneous as possible with respect to the target class. The algorithm splits the data at the predictor variable that maximizes a purity measure (such as Gini impurity or entropy) at each step, building a binary tree of decisions. The final predictions are determined by the majority class in each leaf node of the tree. Decision trees are interpretable and capable of handling non-linear relationships, but they are prone to overfitting, especially in cases with many features and limited data.

(b) Applicability to $p > n$

- **Logistic Regression**: Logistic regression can be applied to high-dimensional data (i.e., $p > n$) if regularization techniques, such as **Lasso** or **Ridge** regression, are used to penalize the coefficients. This helps prevent overfitting by reducing the complexity of the model and selecting relevant covariates.
- **Linear Discriminant Analysis (LDA)**: LDA assumes that the number of predictors p is smaller than the number of observations n (i.e., $p < n$) because it requires the covariance matrix of the predictors to be invertible. When $p > n$, LDA fails because $X^T X$ becomes singular. However, regularization techniques can sometimes be used to mitigate this problem. Also, if we assume that the covariance matrix is diagonal this approach can work in HD settings.
- **Nearest Neighbors**: Nearest neighbors can be applied even when $p > n$, as the method does not rely on a parametric model. However, in high-dimensional spaces, the concept of "closeness" becomes less useful due to the curse of dimensionality. The distance between points becomes less meaningful, which can degrade the model's performance.
- **Classification Trees**: Classification trees can handle high-dimensional data, including the case where $p > n$. The tree construction process involves selecting variables at each node, which inherently handles the issue of having more features than observations. Trees are also robust to overfitting when pruned, although they can become very complex in high-dimensional settings.

(c) The proposed approach has several drawbacks that can lead to suboptimal results:

1. **Multiple Testing Problem**: When performing individual t-tests for each of the 1,000 covariates, the risk of Type I errors increases. With multiple comparisons, some covariates will be falsely identified as significant purely by chance. This increases the likelihood of overfitting, where the selected features are not genuinely predictive.
2. **Ignores Correlation Among Covariates**: Linear Discriminant Analysis assumes that the covariates are jointly normally distributed and may not handle correlated predictors well. By using a univariate approach (i.e., t-tests), we ignore the potential interactions and correlations between covariates, which can lead to incorrect feature selection and bias in the resulting model.
3. **Overfitting Due to Feature Selection**: Selecting the 5 covariates with the smallest p-values is a form of feature selection that can lead to overfitting, especially in high-dimensional datasets. This approach may give too much importance to features that are marginally significant or even irrelevant. A more robust feature selection method (e.g., regularization or cross-validation) would be more appropriate to avoid overfitting.
4. **Assumptions of LDA**: LDA assumes that the covariates are normally distributed within each class and have the same covariance matrix across classes. If these assumptions do not hold, LDA's performance may degrade, and the model may not generalize well to new data. Without careful assessment of these assumptions, the suggested approach could lead to poor predictive performance.
5. **Multicolinearatiy Problem**:

---

6 Exercise 6 (Lasso Regression for High-Dimensional data)

This exercise focuses on a penalized regression method called Lasso regression, which is applicable to different types of target variables Y (including continuous variables as considered here for simplicity, binary variables or time-to-event variables), also when $n < p$.

(a) Briefly explain the principle of Lasso and how it addresses the type of problems outlined in Exercise 3 (the problem that, in least-squares regression, there is an infinity of solutions when $n < p + 1$).

(b) A user ran Lasso regression on his dataset with different λ values: $\lambda = 0.1$ and $\lambda = 1$. He remembers that one of the resulting models included 6 covariates, while the other one included 12 covariates. Can you help him remembering which λ value yielded which model? You may use the formula of the criterion minimized by Lasso regression to substantiate your answer:

$$SSE + \lambda \sum_{j=1}^{p} |\beta_j|$$

(c) Give the name of an R function implementing Lasso regression.

(d) Suppose we want to build a prediction rule $\hat{f}$ using both a handful of well-established clinical covariates (such as age, sex, tumor stage, etc.) and thousands of potential omics markers. Explain why it may be suboptimal to ignore the distinction between these two types of covariates when fitting a Lasso regression model. Why could the Lasso-based "offset" approach be more appropriate?

(e) Give another example of penalized regression method and briefly explain its difference to Lasso.

Answer:

(a) Lasso Regression addresses the issue of infinite solutions in least squares when $n < p + 1$ by adding an L1 penalty term $\lambda \sum_{j=1}^{p} |\beta_j|$ to the Sum of Squared Errors (SSE). This penalty shrinks some coefficients to exactly zero, effectively performing variable selection and yielding a unique, sparse solution. The regularization imposed by λ ensures a trade-off between model fit and complexity, resolving the underdetermined system problem.

(b) A higher λ increases the penalty, shrinking more coefficients to zero. For example, $\lambda = 1$ produces the model with 6 covariates (stronger penalty), while $\lambda = 0.1$ results in 12 covariates (weaker penalty, allowing more non-zero coefficients).

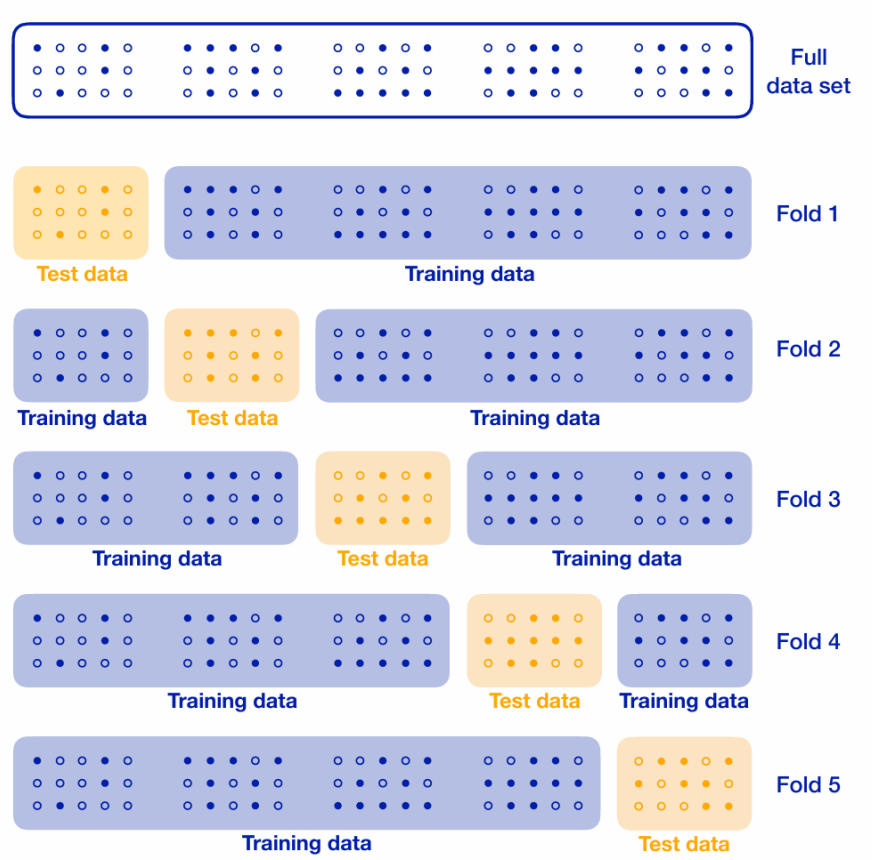
(c) The R function `glmnet()` (from the **glmnet** package) implements Lasso regression by setting `alpha = 1`.

(d) Ignoring the distinction between clinical covariates and omics markers may lead Lasso to penalize clinically important variables, potentially excluding them. The "offset" approach treats established clinical covariates as fixed (unpenalized) and applies Lasso only to omics markers, ensuring clinical variables are retained while selecting predictive omics features.

(e) Ridge regression is another penalized method, using an L2 penalty $\lambda \sum_{j=1}^{p} \beta_j^2$. Unlike Lasso, Ridge shrinks coefficients proportionally but does not set them to zero, retaining all variables. It is suited for scenarios where predictors are correlated and all potentially relevant variables should be considered.

---

7 Exercise 7 (Cross-Validation)



(a) What is the goal of K-fold cross-validation?

(b) Explain in 2-3 sentences the principle of cross-validation using the figure.

(c) How large is K in the case of leave-one-out cross-validation? What is its inconvenience?

(d) Explain why the following cross-validation procedure is flawed: "Using a dataset consisting of $n = 160$ patients we aim at deriving a prediction model for the response status (responder vs. non-responder) after neoadjuvant therapy using gene expression data as covariates and linear discriminant analysis as a classification method. We first select the 10 genes (out of 24,000 genes) that are most associated with the response status according to the t-test and eliminate the remaining 23990 genes. We then partition the available dataset into $K = 5$ folds to estimate the classification error of linear discriminant analysis based on these 10 genes".

(e) In practice, cross-validation is not only used for performance estimation but also for the choice of parameters (a choice which is based on performance estimation, however, here performance estimation is not the goal but the way to the goal). Explain how cross-validation can be used to select a good value for the parameter λ in Lasso regression. You may refer to the function *cv.glmnet* from the package *glmnet* to substantiate your explanation.

Answer:

(a) The goal of K-fold cross-validation is to estimate a model's test error (generalization performance) by repeatedly partitioning the data into K subsets (folds). Each fold serves once as a validation set, while the remaining $K-1$ folds train the model. The average validation error across all K iterations provides a robust estimate of how the model will perform on unseen data.

(b) The figure illustrates splitting the dataset into $K = 5$ folds. In each iteration, one fold is held out as a validation set, and the model is trained on the remaining $K-1$ folds. This process rotates until every fold has been used for validation. The final error estimate is the average of the validation errors across all K trials, reducing variability compared to a single train-test split.

(c) In leave-one-out cross-validation (LOOCV), $K = n$ (equal to the number of observations). The inconvenience is its computational expense: fitting the model n times becomes impractical for large n. Additionally, it may have higher variance in error estimation compared to K-fold CV with smaller K.

(d) The procedure is flawed because feature selection (choosing 10 genes) was performed on the entire dataset before cross-validation. This leaks information from the validation folds into the training process, biasing the error estimate (making it overly optimistic). To avoid this, gene selection should occur within each training fold of the CV loop, ensuring no validation data influences feature selection.

(e) To select λ in Lasso regression, cross-validation is used to evaluate performance across a set of λ values. For each λ, K-fold CV computes the average validation error. The optimal $\lambda_{\min}$ in `cv.glmnet` minimizes this error. Alternatively, λ_{1se} selects the largest λ within one standard error of the minimum, balancing simplicity and performance. The `cv.glmnet` function automates this by training Lasso models for different λ values across folds and returns the optimal choice based on CV results.