Group 1 – Ridge Regression and Principal Component Analysis

Time Series Analysis and Machine Learning (HS-2025)

Ennio Eberwein, Daria Göbel, Patrick Trost

version 1: 18th Novemeber 2025

Contents

This report presents the submission of **Group 1** for the **Time Series Analysis and Machine Learning** course (HS-2025). There are two exercises in this assignment. **First**, we replicated both panels in *James — An Introduction to Statistical Learning with Applications in R* (Figure 6.4), which displayed the standardized ridge regression coefficients of the credit data set. **Second**, we explained how the two principal component lines in Figure 6.14 were derived, including their meaning and why they are orthogonal to each other.

Exercise 1

1. Data Preparation

We started by importing the necessary libraries used throughout the project.

```
# Analysis and modeling toolkits
library(readxl)
library(dplyr)
library(purrr)

# for lm.ridge
library(MASS)
```

The raw credit_data.csv file is read and the binary Yes/No indicators for the predictors "Own","Student", and "Married" are converted into numeric binary variables. This conversion ensures that the model correctly interprets these predictors as binary inputs, with "Yes" corresponding to 1 and "No" to 0.

```
# Load data and convert categorical Yes/No flags to 0/1 dummies
data <- read.csv("credit_data.csv")

data$Student <- ifelse(data$Student == "Yes", 1, 0)
data$Own     <- ifelse(data$Own     == "Yes", 1, 0)
data$Married <- ifelse(data$Married == "Yes", 1, 0)
```

The categorical predictor "Region" is transformed into three dummy variables representing the regions South, West, and East. These dummy variables are created without including an intercept (in order to avoid perfect multicollinearity), allowing each region to be treated as a separate category in the design matrix. The new indicator variables are appended to the dataset for subsequent analysis.

```
# Build Region indicators and bind them to the main data frame
region_dummies <- model.matrix(~ Region - 1, data = data)
data <- cbind(data[ , !(names(data) %in% "Region")], region_dummies)
```

After implementing the four modifications described above, the cleaned dataset is visualized to inspect for potential inconsistencies or data entry errors.

```
View(data)
```

2. Design Matrix Construction

In this step, we identify the relevant predictors and the response variable to be used in our regression model. The predictor set includes both continuous variables (such as income, limit, and rating) and binary or categorical variables (such as married and region).

To enhance the interpretability and comparability of the model, we standardize all predictors so that each has a mean of zero and a standard deviation of one. Standardization ensures that no variable dominates the estimation simply because it is measured in larger numeric units. It also allows ridge regression to apply the same degree of shrinkage to all coefficients, which improves the stability and comparability of the results.

Without standardization, the penalization term in ridge regression would depend not only on the regularization parameter λ but also on the scale of the jth predictor (see formula 6.6).

```r
# Define predictor matrix X and response vector Y
X <- dplyr::select(
  data,
  Income, Limit, Rating, Cards, Age, Education,
  Own, Student, Married,
  RegionEast, RegionWest, RegionSouth
)
Y <- data$Balance

# Standardize predictors to mean 0, variance 1
# We work with the already build in scale function instead of the equation given in the book
X_standardized <- scale(X)
X_std <- as.data.frame(X_standardized)
```

3. Helper Functions

To make the analysis easier and more consistent, we define two short helper functions that perform the key calculations used throughout the report.
The first function computes the ridge regression coefficients for a given value of the regularization parameter `lambda` following the model:

$$\hat{\beta}_\lambda = \arg\min_\beta \left\{ \sum_{i=1}^{n} (y_i - x_i^\top \beta)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right\}$$

The second function computes the ordinary least squares (OLS) coefficients without an intercept following the model:

$$\hat{\beta}_{OLS} = \arg\min_\beta \left\{ \sum_{i=1}^{n} (y_i - x_i^\top \beta)^2 \right\}$$

```r
# Ridge coefficients for a chosen penalty lambda
# Brauchen wir inter = false und scales = trues -> hat lm.ride diese Argumente?
compute_coefficients <- function(lambda, X, y) {
  ridge_fit <- lm.ridge(y ~ as.matrix(X),
                        lambda = lambda,
                        Inter = FALSE,
                        scales = TRUE)
  coef(ridge_fit)
}

# OLS coefficients with no intercept (keeps all dummies)
compute_ols <- function(X, y) {
  df <- data.frame(y = y, X)
  fit <- lm(y ~ . - 1, data = df)
  b <- coef(fit)
  b[colnames(X)]
}
```

Compute the OLS coefficients once for later comparisons.

```r
ols_coefficients <- compute_ols(X_std, Y)
```

4. Ridge Paths Across Lambda Grid

In this section, we examine how the estimated coefficients change as the regularization strength `lambda` increases.

To do this, we define a sequence of 100 `lambda` values that span a wide range—from very small values (close to ordinary least squares) to very large ones (strong shrinkage).

For each `lambda`, the ridge regression is re-estimated, and the resulting coefficients are stored.
This creates a set of "ridge paths" that show how each variable's influence gradually shrinks toward zero as the penalty grows.

```
# Log-spaced lambda grid
lambdas <- 10^seq(-2, 5, length.out = 100)

# Collect ridge coefficients for every lambda
coefficients <- sapply(lambdas, function(lambda) {
  compute_coefficients(lambda, X_std, Y)
})

# Clean up into a data frame
df <- t(coefficients)
df <- df[, -1]  # drop intercept column added by lm.ridge
colnames(df) <- gsub("as\\.matrix\\(X\\)", "", colnames(df))
df <- cbind(lambda = lambdas, df)
df <- as.data.frame(df)
```

5. Norm Calculations

In this step, we compare the overall size of the ridge coefficients to the OLS coefficients.
We do this by computing their norms, which measure the combined magnitude of all estimated coefficients.
As the regularization strength `lambda` increases, the ridge coefficients shrink, and their norm becomes smaller relative to the OLS solution.

The resulting ratio (ridge norm divided by OLS norm) provides a convenient way to express the degree of shrinkage on a standardized 0–1 scale.
This ratio will later serve as a rescaled x-axis when visualizing how coefficients change with increasing regularization.

The ratio is given by:

$$\frac{\|\hat{\beta}_\lambda^R\|_2}{\|\hat{\beta}_{OLS}\|_2} = \frac{\sqrt{\sum_{j=1}^{p}\left(\hat{\beta}_{j,\lambda}^R\right)^2}}{\sqrt{\sum_{j=1}^{p}\left(\hat{\beta}_{j,OLS}\right)^2}}$$

```
# Ridge  2 norm per lambda
ridge_norms <- apply(df[, !(names(df) %in% "lambda")], 1,
                     function(row) sqrt(sum(row^2)))
print(ridge_norms)
```

```
##   [1] 563.045823 562.914949 562.761869 562.583003 562.374266 562.131020
##   [7] 561.848039 561.519478 561.138866 560.699126 560.192635 559.611332
##  [13] 558.946895 558.190999 557.335665 556.373710 555.299288 554.108512
##  [19] 552.800106 551.376025 549.841951 548.207547 546.486358 544.695250
##  [25] 542.853340 540.980431 539.095079 537.212473 535.342391 533.487488
##  [31] 531.642110 529.791731 527.912991 525.974209 523.936180 521.753075
##  [37] 519.373263 516.739997 513.791905 510.463371 506.684885 502.383504
##  [43] 497.483580 491.907916 485.579508 478.423993 470.372893 461.367662
##  [49] 451.364416 440.339102 428.292666 415.255616 401.291231 386.496611
##  [55] 371.000877 354.960098 338.548970 321.949886 305.340571 288.881907
```

```
##  [61] 272.707717 256.917985 241.576474 226.712857 212.328726 198.406227
##  [67] 184.917799 171.835518 159.138836 146.819912 134.886168 123.360114
##  [73] 112.276837 101.679790  91.615665  82.129158  73.258346  65.031163
##  [79]  57.463264  50.557282  44.303313  38.680334  33.658206  29.199945
##  [85]  25.263993  21.806297  18.782084  16.147284  13.859597  11.879231
##  [91]  10.169366   8.696391   7.429972   6.342980   5.411352   4.613879
##  [97]   3.931976   3.349428   2.852152   2.427951
```

```
# OLS 2 norm (drop NA caused by omitted dummy)
ols_norm <- sqrt(sum(ols_coefficients^2, na.rm = TRUE))
print(ols_norm)
```

```
## [1] 563.8005
```

```
# Attach the ridge/OLS norm ratio to the data frame
df$norm_ratio <- ridge_norms / ols_norm
```

The following provides a review of the final coefficient table.

```
View(df)
```

6. Visualization

Ensure a `figures/` folder exists and save both figures to disk.

```
if (!dir.exists("figures")) dir.create("figures")
```

6.1 Lambda-on-Log-Scale Plot

This plot shows how each standardized coefficient changes as the regularization strength `lambda` increases, using a logarithmic x-axis.
At small values of `lambda`, the model behaves like OLS with little shrinkage, while larger values gradually pull the coefficients toward zero.
The highlighted variables (Income, Limit, Rating, and Student) represent the most influential predictors, while the others are shown in gray for context.
Overall, the plot illustrates how ridge regression balances model complexity and stability by progressively reducing the size of the coefficients.

```
png("figures/figure_6.4_1.png", width = 768, height = 889, res = 120)

plot(df$lambda, df$Income, type = "l", log = "x", lwd = 2,
     ylim = c(-300, 450), col = "black",
     xlab = expression(lambda),
     ylab = "Standardized Coefficients",
     xaxt = "n", yaxt = "n")
axis(side = 1, at = c(0.01, 1, 100, 10000),
     labels = c("1e-02", "1e+00", "1e+02", "1e+04"))
axis(side = 2, at = seq(-300, 400, by = 100),
     labels = seq(-300, 400, by = 100))

# Highlighted series
lines(df$lambda, df$Limit,   col = "red",        lty = 2, lwd = 2)
lines(df$lambda, df$Rating,  col = "blue",       lty = 3, lwd = 2)
lines(df$lambda, df$Student, col = "goldenrod2", lty = 4, lwd = 2)

# Remaining predictors in gray
lines(df$lambda, df$Cards,       col = "grey70")
```

```
lines(df$lambda, df$Age,          col = "grey70")
lines(df$lambda, df$Education,    col = "grey70")
lines(df$lambda, df$Own,          col = "grey70")
lines(df$lambda, df$Married,      col = "grey70")
lines(df$lambda, df$RegionEast,   col = "grey70")
lines(df$lambda, df$RegionWest,   col = "grey70")
lines(df$lambda, df$RegionSouth,  col = "grey70")

legend("topright",
       legend = c("Income", "Limit", "Rating", "Student"),
       col = c("black", "red", "blue", "goldenrod2"),
       lty = c(1, 2, 3, 4),
       lwd = 2,
       bty = "n",
       cex = 1.2)

dev.off()
```

```
## pdf
##   2
```

6.2 Norm-Ratio Plot

This plot presents the same coefficient paths as before but uses the ratio of the ridge to OLS norms on the x-axis instead of `lambda`.
This ratio provides a standardized measure of shrinkage that ranges from 1 (no regularization, same as OLS) to 0 (maximum shrinkage).
By expressing shrinkage as a relative measure rather than a raw penalty value, this visualization makes it easier to compare the degree of regularization across different models or datasets.

```
df_ratio <- df[order(df$norm_ratio), ]

png("figures/figure_6.4_2.png", width = 768, height = 889, res = 120)

plot(df_ratio$norm_ratio, df_ratio$Income, type = "l", lwd = 2,
     xlim = c(0, 1), ylim = c(-300, 450), col = "black",
     xlab = expression("||" * hat(beta)[lambda]^R * "||"[2] / "||" * hat(beta) * "||"[2]),
     ylab = "Standardized Coefficients")

lines(df_ratio$norm_ratio, df_ratio$Limit,   col = "red",        lty = 2, lwd = 2)
lines(df_ratio$norm_ratio, df_ratio$Rating,  col = "blue",       lty = 3, lwd = 2)
lines(df_ratio$norm_ratio, df_ratio$Student, col = "goldenrod2", lty = 4, lwd = 2)

lines(df_ratio$norm_ratio, df_ratio$Cards,       col = "grey70")
lines(df_ratio$norm_ratio, df_ratio$Age,         col = "grey70")
lines(df_ratio$norm_ratio, df_ratio$Education,   col = "grey70")
lines(df_ratio$norm_ratio, df_ratio$Own,         col = "grey70")
lines(df_ratio$norm_ratio, df_ratio$Married,     col = "grey70")
lines(df_ratio$norm_ratio, df_ratio$RegionEast,  col = "grey70")
lines(df_ratio$norm_ratio, df_ratio$RegionWest,  col = "grey70")
lines(df_ratio$norm_ratio, df_ratio$RegionSouth, col = "grey70")

dev.off()
```

```
## pdf
```

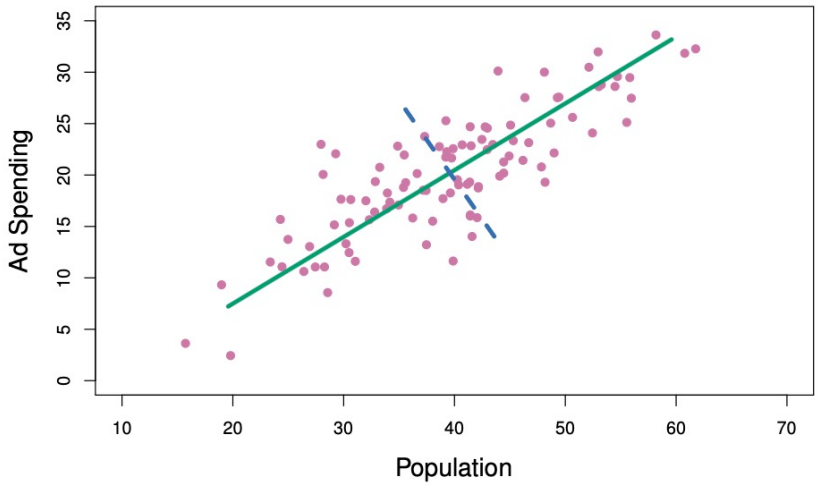
```
##   2
```

7. Interpretation Guide

- The ridge paths illustrate how each standardized coefficient shrinks toward zero as `lambda` grows, with Student reacting more quickly than Income/Limit/Rating.
- The norm-ratio axis recasts shrinkage strength on a common 0–1 scale: values near 1 behave like OLS; values near 0 indicate heavy penalization.

Exercise 2

Exercise 2 is split in three parts: 1. Explain in detail how the two lines in Figure 6.14 are derived. 2. What do they mean? 3. Why are they orthogonal to each other?



The population size (pop) and ad spending (ad) for 100 different cities are shown as purple circles. The green solid line indicates the first principal component, and the blue dashed line indicates the second principal component.

Overview

The overarching topic of this exercise is principal-component-analysis (PCA) with non-standardized variables population size (pop) and ad spending (ad). PCA is a dimension reduction technique to reduce the predictor space p to M principal components, which are linear combinations of the p variables.

Derivation of principal components

Principal components (PC) are denoted as $Z_1, Z_2, \ldots, Z_M$, where M is the number of principal components. Each principal component is a linear combination of the original predictors $X_1, X_2, \ldots, X_p$.

In our case, we can, at most, derive two principal components as we have two variables. Our PCs take the form $PC_i = a \cdot \text{pop} + b \cdot \text{ad}$ with a, b being the principal component loadings (also referred to as weights).

To derive the first principal component, we want to find a linear combination of pop and ad that captures the maximum variance in the input variables while minimizing the sum of the squared perpendicular distances between each point and the line.

For this, it is vital to restrict a, b because they would otherwise blow up the variance arbitrarily. Therefore, we can only consider linear combination of the form $a^2 + b^2 = 1$.

We are now maximizing this variance using the Lagrange method:

$$\max \text{Var}(a \cdot \text{pop} + b \cdot \text{ad}) \quad \text{s.t.} \quad a^2 + b^2 = 1$$

For the case that X_1 and X_2 are standardized variables (mean equal to zero, variance equal to one), there is a simple, analytical solution to the maximization problem:

The Lagrangian function is:

$$(*) \quad \mathcal{L}(a, b, \lambda) = a^2 \cdot Var(X_1) + b^2 \cdot Var(X_2) + 2ab \cdot Cov(X_1, X_2) + \lambda(a^2 + b^2 - 1)$$

As we work with standardized predictors, the variance (and thus also the standard deviation) equals one. We can thus drop the variance term and set the covariance equal to the correlation. Hence, the equation simplifies as follows:

$$(**) \quad \mathcal{L}(a, b, \lambda) = a^2 + b^2 + 2ab \cdot \rho + \lambda(a^2 + b^2 - 1)$$

To continue, we are defining the three conditions for the maximization problem. First, we take the two partial derivatives of the Lagrangian with respect to a and b and then include our constraint as the third equation:

$$(1) \quad \frac{\partial \mathcal{L}}{\partial a} = 2a + 2b \cdot \rho + \lambda \cdot 2a \stackrel{!}{=} 0$$

$$(2) \quad \frac{\partial \mathcal{L}}{\partial b} = 2b + 2a \cdot \rho + \lambda \cdot 2b \stackrel{!}{=} 0$$

$$(3) \quad a^2 + b^2 = 1$$

Solving this system of equations, yields the first principal component:

$$PC_1 = \frac{1}{\sqrt{2}} X_1 + \frac{1}{\sqrt{2}} X_2$$

However, this derivation assumes standardization of variables which is not the case in the given example (pop and ad are solely mean adjusted). Thus, while the initial equation $(*)$ still holds, it does no longer simplify to $(**)$ as the variance (and hence the standard deviation) is not necessarily equal to one.

The idea of the maximization problem is still the same and results in the following:

$$\max \text{Var}(aX_1 + bX_2) = \max \; a^2\sigma^2_{X_1} + b^2\sigma^2_{X_2} + 2ab \cdot \sigma_{X_1X_2} \quad \text{s.t.} \quad a^2 + b^2 = 1$$

The corresponding Lagrangian function is:

$$\mathcal{L}(a, b, \lambda) = a^2 \cdot \sigma^2_{X_1} + b^2 \cdot \sigma^2_{X_2} + 2ab \cdot \sigma_{X_1X_2} + \lambda(a^2 + b^2 - 1)$$

This gives us the following three conditions:

$$\frac{\partial \mathcal{L}}{\partial a} = 2a \cdot \sigma^2_{X_1} + 2b \cdot \sigma_{X_1X_2} + \lambda \cdot 2a \stackrel{!}{=} 0$$

$$\frac{\partial \mathcal{L}}{\partial b} = 2b \cdot \sigma^2_{X_2} + 2a \cdot \sigma_{X_1X_2} + \lambda \cdot 2b \stackrel{!}{=} 0$$

$$a^2 + b^2 = 1$$

Solving this set of equations for a and b is not possible as we don't have access to the data set used.

The data used in the book gives us the loadings $a = 0.839$ and $b = 0.544$, resulting in the equation for the first principal component:

$$Z_1 = 0.839 \times (\text{pop} - \overline{\text{pop}}) + 0.544 \times (\text{ad} - \overline{\text{ad}})$$

PC_2 explains the rest of the variance in the data that is not captured by PC_1. Hence PC_2 has to be uncorrelated with PC_1.

$$\begin{bmatrix} a_1 \\ b_1 \end{bmatrix} \cdot \begin{bmatrix} a_2 \\ b_2 \end{bmatrix} = 0$$

$$a_1 \cdot a_2 + b_1 \cdot b_2 = 0$$

$$0.839 \cdot a_2 + 0.544 \cdot b_2 = 0$$

$$\Rightarrow (a_2, b_2) = (0.544, -0.839)$$

$$Z_2 = 0.544 \times (\text{pop} - \overline{\text{pop}}) - 0.839 \times (\text{ad} - \overline{\text{ad}})$$

Summary

With the previous calculations, we found the linear combinations of predictors that explain the highest variance in the predictors. The first principal component, depicted as the green line in figure 6.14 is the component that captures the most variance in the scatter plot and passes directly through the cloud in the direction that has the greatest spread of data. Hence PC_1 can be used to reduce the dimensions of predictors from two to one while capturing the highest degree of variance possible in the initial variables.