Exercises on Machine Learning Tools

STSM2634

2025-05-19

```
library(Ecdat)

## Warning: package 'Ecdat' was built under R version 4.3.3

## Loading required package: Ecfun

## Warning: package 'Ecfun' was built under R version 4.3.3

##
## Attaching package: 'Ecfun'

## The following object is masked from 'package:base':
##
##     sign

##
## Attaching package: 'Ecdat'

## The following object is masked from 'package:datasets':
##
##     Orange

library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse
2.0.0 ──
## ✔ dplyr     1.1.2     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.0
## ✔ ggplot2   3.4.2     ✔ tibble    3.2.1
## ✔ lubridate 1.9.2     ✔ tidyr     1.3.0
## ✔ purrr     1.0.1

## ── Conflicts ──────────────────────────────────────────
tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all
conflicts to become errors

library(e1071)
library(lattice)
library(AER)

## Warning: package 'AER' was built under R version 4.3.3
```

```
## Loading required package: car

## Warning: package 'car' was built under R version 4.3.1

## Loading required package: carData
##
## Attaching package: 'carData'
##
## The following object is masked from 'package:Ecdat':
##
##     Mroz
##
##
## Attaching package: 'car'
##
## The following object is masked from 'package:dplyr':
##
##     recode
##
## The following object is masked from 'package:purrr':
##
##     some
##
## Loading required package: lmtest

## Warning: package 'lmtest' was built under R version 4.3.1

## Loading required package: zoo
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
##
## Loading required package: sandwich
## Loading required package: survival
```

```r
library(neuralnet)
```

```
##
## Attaching package: 'neuralnet'
##
## The following object is masked from 'package:dplyr':
##
##     compute
```

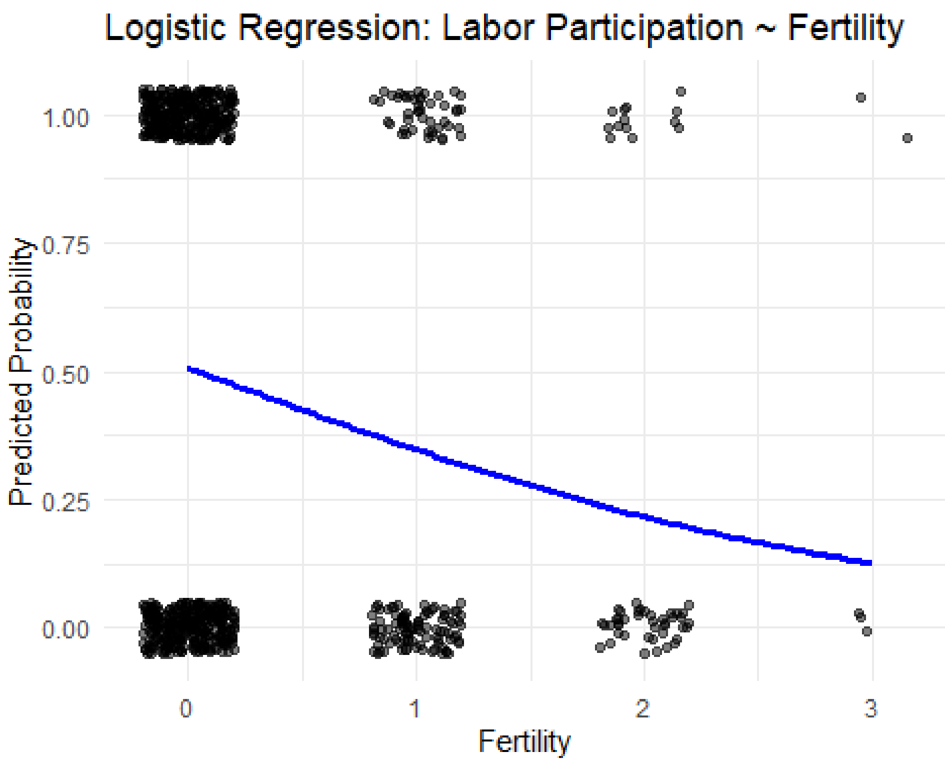
```r
library(MASS)
```

```
##
## Attaching package: 'MASS'
##
```

```
## The following object is masked from 'package:dplyr':
##
##     select
##
## The following object is masked from 'package:Ecdat':
##
##     SP500
```

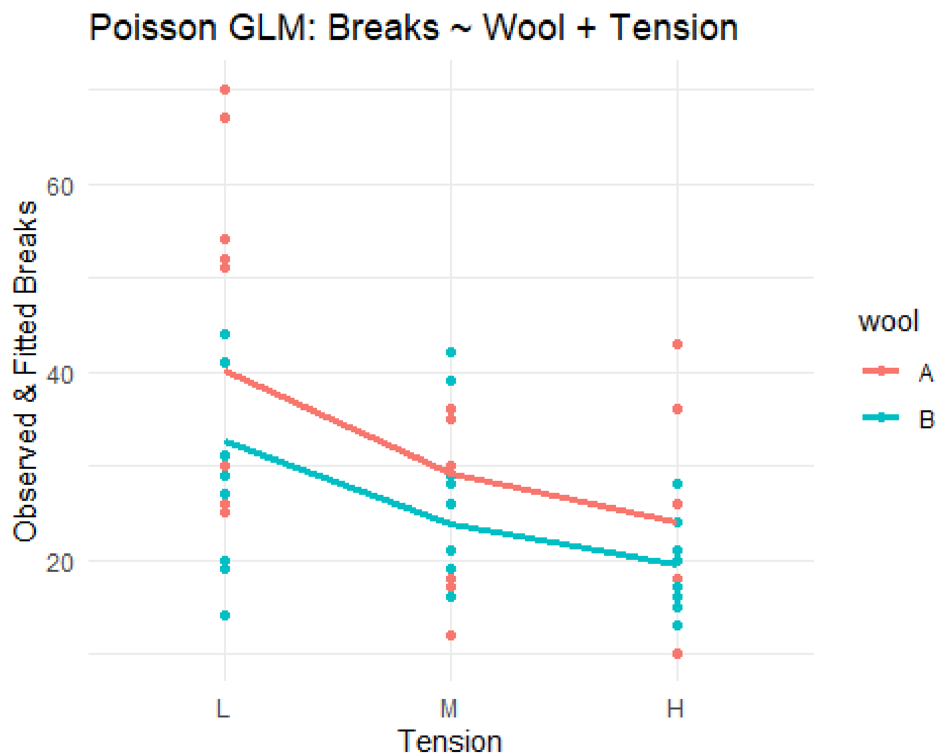
Q1. Logistic Regression on Binary Outcome (SwissLabor dataset).

```
data("SwissLabor")
str(SwissLabor)

## 'data.frame':    872 obs. of  7 variables:
##  $ participation: Factor w/ 2 levels "no","yes": 1 2 1 1 1 2 1 2 1 1 ...
##  $ income       : num  10.8 10.5 11 11.1 11.1 ...
##  $ age          : num  3 4.5 4.6 3.1 4.4 4.2 5.1 3.2 3.9 4.3 ...
##  $ education    : num  8 8 9 11 12 12 8 8 12 11 ...
##  $ youngkids    : num  1 0 0 2 0 0 0 0 0 0 ...
##  $ oldkids      : num  1 1 0 0 2 1 0 2 0 2 ...
##  $ foreign      : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...

# Fit logistic regression model
model1 <- glm(participation ~ youngkids + income, data = SwissLabor, family =
binomial)

# Create prediction grid
newdata1 <- data.frame(youngkids = seq(min(SwissLabor$youngkids),
max(SwissLabor$youngkids), length.out = 100),
                       income = mean(SwissLabor$income))

newdata1$predicted_prob <- predict(model1, newdata = newdata1, type =
"response")

# Plot
ggplot(SwissLabor, aes(x = youngkids, y = as.numeric(participation) - 1)) +
  geom_jitter(height = 0.05, width = 0.2, alpha = 0.5) +
  geom_line(data = newdata1, aes(x = youngkids, y = predicted_prob), color =
"blue", size = 1.2) +
  labs(title = "Logistic Regression: Labor Participation ~ Fertility",
       x = "Fertility", y = "Predicted Probability") +
  theme_minimal()

## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



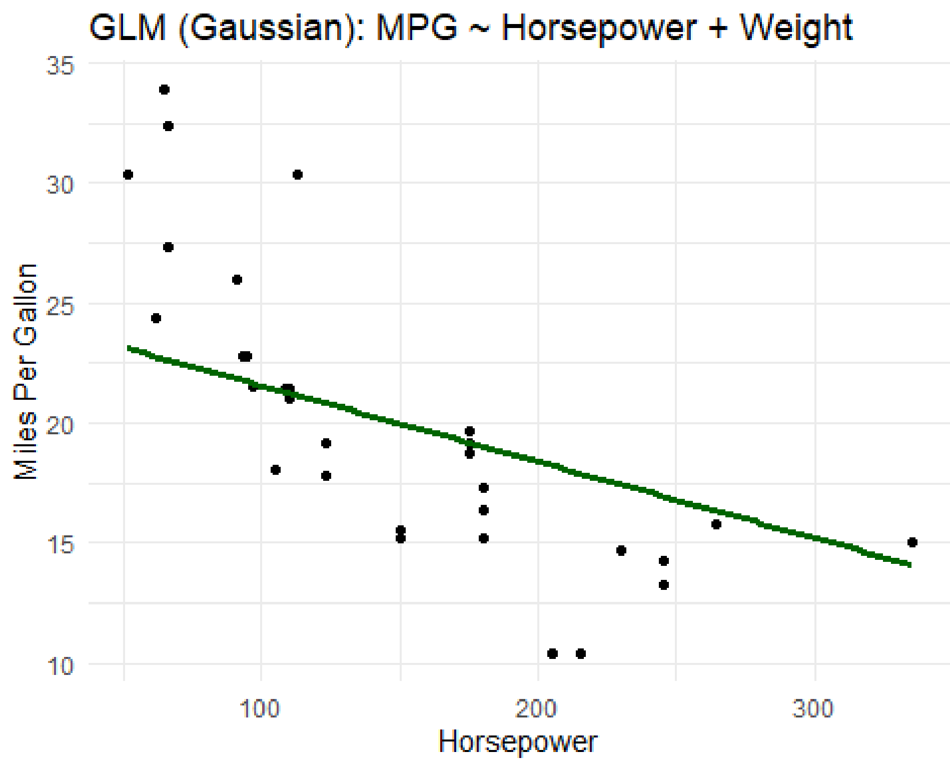
Q2. Poisson Regression for Count Data (warpbreaks dataset).

```r
data("warpbreaks")

# Fit Poisson regression model
model2 <- glm(breaks ~ wool + tension, data = warpbreaks, family = poisson)

# Predict fitted values
warpbreaks$fit <- predict(model2, type = "response")

# Plot
ggplot(warpbreaks, aes(x = tension, y = breaks, color = wool)) +
  geom_point() +
  geom_line(aes(y = fit, group = wool), size = 1.2) +
  labs(title = "Poisson GLM: Breaks ~ Wool + Tension",
       x = "Tension", y = "Observed & Fitted Breaks") +
  theme_minimal()
```



Q3. Ordinary Linear Regression (mtcars dataset).

```
data("mtcars")

# Fit GLM (default family = Gaussian)
model3 <- glm(mpg ~ hp + wt, data = mtcars)

# Predict over horsepower range at fixed weight
newdata3 <- data.frame(hp = seq(min(mtcars$hp), max(mtcars$hp), length.out =
100),
                       wt = mean(mtcars$wt))
newdata3$predicted <- predict(model3, newdata = newdata3)

# Plot
ggplot(mtcars, aes(x = hp, y = mpg)) +
  geom_point() +
  geom_line(data = newdata3, aes(x = hp, y = predicted), color = "darkgreen",
size = 1.2) +
  labs(title = "GLM (Gaussian): MPG ~ Horsepower + Weight",
       x = "Horsepower", y = "Miles Per Gallon") +
  theme_minimal()
```



Q4. Applying SVM on Wages data (Binary classification) to model the income status based on the education, experience, marital status, and sex.

```
data(Wages)
str(Wages)

## 'data.frame':    4165 obs. of  12 variables:
##  $ exp    : int  3 4 5 6 7 8 9 30 31 32 ...
##  $ wks    : int  32 43 40 39 42 35 32 34 27 33 ...
##  $ bluecol: Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 2 2 2 ...
##  $ ind    : int  0 0 0 0 1 1 1 0 0 1 ...
##  $ south  : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 1 1 1 ...
##  $ smsa   : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ married: Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 2 2 2 ...
##  $ sex    : Factor w/ 2 levels "female","male": 2 2 2 2 2 2 2 2 2 2 ...
##  $ union  : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 2 ...
##  $ ed     : int  9 9 9 9 9 9 9 11 11 11 ...
##  $ black  : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ lwage  : num  5.56 5.72 6 6 6.06 ...

any(is.na(Wages))

## [1] FALSE

# Create binary target: high wage
mean_wage <- mean(Wages$lwage, na.rm = TRUE)
```

```r
Wages_clean <- na.omit(Wages) # Remove missing values, if any
Wages_clean$high_wage <- as.factor(Wages_clean$lwage > mean_wage)

# Split into training and test sets (e.g., 70/30)
set.seed(123)
n <- nrow(Wages_clean)
train_index <- sample(1:n, size = 0.7 * n)
train_data <- Wages_clean[train_index, ]
test_data <- Wages_clean[-train_index, ]

# Fit SVM
model2 <- svm(high_wage ~ exp + ed + married + sex, data = train_data, kernel
= "radial", probability = TRUE)

# Training accuracy
pred2 <- predict(model2, test_data)
acc2 <- mean(pred2 == test_data$high_wage)
cat("Training Accuracy (Wages):", round(acc2, 4), "\n")

## Training Accuracy (Wages): 0.7152

# Dummy input
dummy2 <- data.frame(exp = 10, ed = 14, married = factor("yes", levels =
levels(Wages_clean$married)), sex = factor("male", levels =
levels(Wages_clean$sex)) )
pred_dummy2 <- predict(model2, dummy2, probability = TRUE)
cat("Dummy Prediction (Wages):", as.character(pred_dummy2), "\n")

## Dummy Prediction (Wages): FALSE
```

Q4. Applying SVM on SwissLabor data (Binary classification) to model the participation status based on the education, age, and income.

```r
# Load data
data(SwissLabor)
str(SwissLabor)

## 'data.frame':    872 obs. of  7 variables:
##  $ participation: Factor w/ 2 levels "no","yes": 1 2 1 1 1 2 1 2 1 1 ...
##  $ income       : num  10.8 10.5 11 11.1 11.1 ...
##  $ age          : num  3 4.5 4.6 3.1 4.4 4.2 5.1 3.2 3.9 4.3 ...
##  $ education    : num  8 8 9 11 12 12 8 8 12 11 ...
##  $ youngkids    : num  1 0 0 2 0 0 0 0 0 0 ...
##  $ oldkids      : num  1 1 0 0 2 1 0 2 0 2 ...
##  $ foreign      : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...

Swiss_clean <- na.omit(SwissLabor) # Remove missing values, if any

# Target: participate (already binary: yes/no)
Swiss_clean$participate <- factor(Swiss_clean$participation)
```

```r
# Split into train/test
set.seed(123)
n <- nrow(Swiss_clean)
train_idx <- sample(1:n, size = 0.7 * n)
train_data <- Swiss_clean[train_idx, ]
test_data <- Swiss_clean[-train_idx, ]

# Fit SVM
model_swiss <- svm(participation ~ age + income + education,
                   data = train_data, kernel = "radial", probability = TRUE)

# Test accuracy
pred_swiss <- predict(model_swiss, test_data)
acc_swiss <- mean(pred_swiss == test_data$participate)
cat("Test Accuracy (SwissLabor):", round(acc_swiss, 4), "\n")
```

```
## Test Accuracy (SwissLabor): 0.6221
```

```r
# Dummy input
dummy_swiss <- data.frame(age = 35, income = 5, education = 12)
pred_dummy_swiss <- predict(model_swiss, dummy_swiss, probability = TRUE)
cat("Dummy Prediction (SwissLabor):", as.character(pred_dummy_swiss), "\n")
```

```
## Dummy Prediction (SwissLabor): no
```

Q5. Housing Price Categorization using Boston data from MASS package with SVM.

```r
data(Boston)
str(Boston)
```

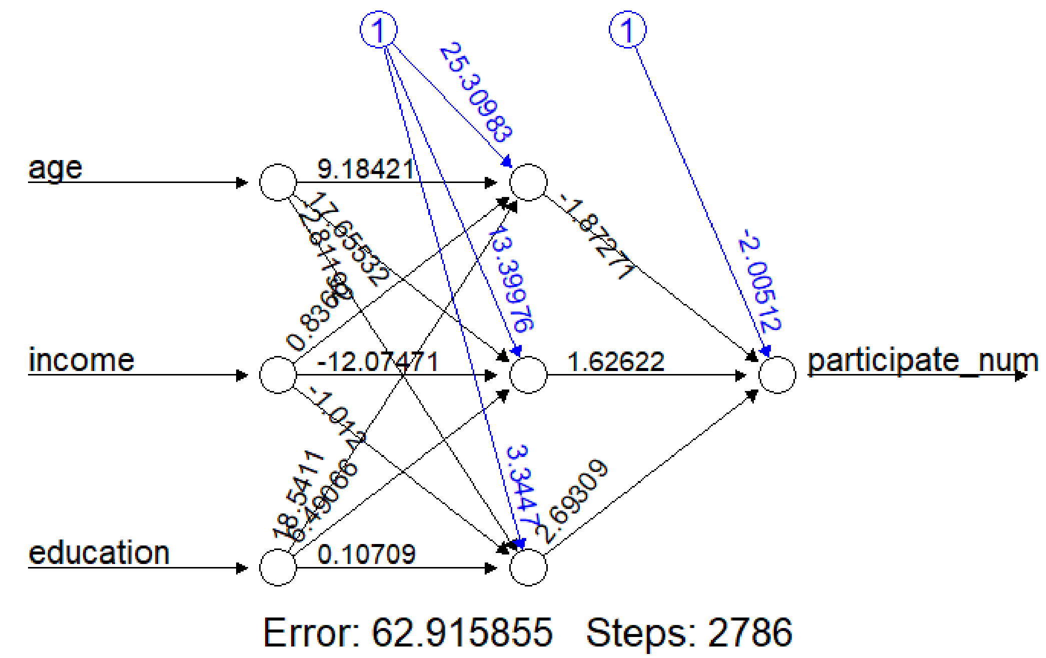
```
## 'data.frame':    506 obs. of  14 variables:
##  $ crim   : num  0.00632 0.02731 0.02729 0.03237 0.06905 ...
##  $ zn     : num  18 0 0 0 0 0 12.5 12.5 12.5 12.5 ...
##  $ indus  : num  2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...
##  $ chas   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ nox    : num  0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 
0.524 ...
##  $ rm     : num  6.58 6.42 7.18 7 7.15 ...
##  $ age    : num  65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...
##  $ dis    : num  4.09 4.97 4.97 6.06 6.06 ...
##  $ rad    : int  1 2 2 3 3 3 5 5 5 5 ...
##  $ tax    : num  296 242 242 222 222 222 311 311 311 311 ...
##  $ ptratio: num  15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...
##  $ black  : num  397 397 393 395 397 ...
##  $ lstat  : num  4.98 9.14 4.03 2.94 5.33 ...
##  $ medv   : num  24 21.6 34.7 33.4 36.2 28.7 22.9 27.1 16.5 18.9 ...
```

```r
# Create binary target
median_price <- median(Boston$medv)
Boston$expensive <- as.factor(Boston$medv > median_price)

# Split into train/test
set.seed(123)
n <- nrow(Boston)
train_idx <- sample(1:n, size = 0.7 * n)
train_data <- Boston[train_idx, ]
test_data <- Boston[-train_idx, ]

# Fit SVM
model_boston <- svm(expensive ~ lstat + rm + crim, data = train_data, kernel
= "radial", probability = TRUE)

# Test accuracy
pred_boston <- predict(model_boston, test_data)
acc_boston <- mean(pred_boston == test_data$expensive)
cat("Test Accuracy (Boston):", round(acc_boston, 4), "\n")

## Test Accuracy (Boston): 0.8553

# Dummy input
dummy_boston <- data.frame(lstat = 5, rm = 7, crim = 0.05)
pred_dummy_boston <- predict(model_boston, dummy_boston, probability = TRUE)
cat("Dummy Prediction (Boston):", as.character(pred_dummy_boston), "\n")

## Dummy Prediction (Boston): TRUE
```

Q6. Applying ANN on SwissLabor data (Binary classification) to model the participation status based on the education, age, and income.

```r
data(SwissLabor)
str(SwissLabor)

## 'data.frame':    872 obs. of  7 variables:
##  $ participation: Factor w/ 2 levels "no","yes": 1 2 1 1 1 2 1 2 1 1 ...
##  $ income       : num  10.8 10.5 11 11.1 11.1 ...
##  $ age          : num  3 4.5 4.6 3.1 4.4 4.2 5.1 3.2 3.9 4.3 ...
##  $ education    : num  8 8 9 11 12 12 8 8 12 11 ...
##  $ youngkids    : num  1 0 0 2 0 0 0 0 0 0 ...
##  $ oldkids      : num  1 1 0 0 2 1 0 2 0 2 ...
##  $ foreign      : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...

Swiss_clean <- na.omit(SwissLabor)
Swiss_clean$participate_num <- ifelse(Swiss_clean$participation == "yes", 1,
0)

# Normalize inputs
Swiss_clean$income <- scale(Swiss_clean$income)
```

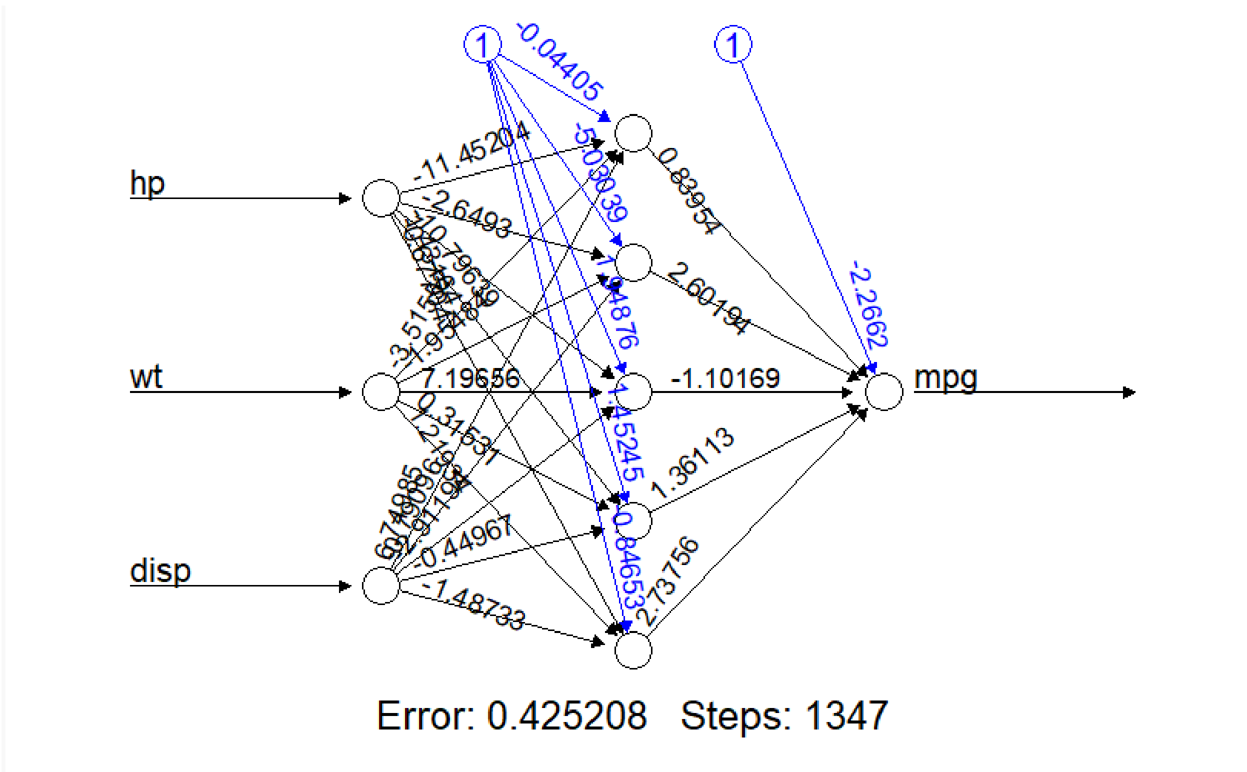
```r
Swiss_clean$education <- scale(Swiss_clean$education)
Swiss_clean$age <- scale(Swiss_clean$age)

# Train-test split
set.seed(123)
n <- nrow(Swiss_clean)
train_idx <- sample(1:n, size = 0.7 * n)
train_data <- Swiss_clean[train_idx, ]
test_data <- Swiss_clean[-train_idx, ]

# Train neural network
model1 <- neuralnet(participate_num ~ age + income + education,
                    data = train_data, hidden = c(3), linear.output = FALSE)

plot(model1)
# Predict on test set
test_pred1 <- compute(model1, test_data[, c("age", "income",
"education")])$net.result
test_class1 <- ifelse(test_pred1 > 0.5, 1, 0)
acc1 <- mean(test_class1 == test_data$participate_num)
cat("Test Accuracy (SwissLabor):", round(acc1, 4), "\n")

## Test Accuracy (SwissLabor): 0.6374

# Dummy input
dummy1 <- data.frame(
  age = scale(35, attr(Swiss_clean$age, "scaled:center"),
attr(Swiss_clean$age, "scaled:scale")),
  income = scale(5, attr(Swiss_clean$income, "scaled:center"),
attr(Swiss_clean$income, "scaled:scale")),
  education = scale(12, attr(Swiss_clean$education, "scaled:center"),
attr(Swiss_clean$education, "scaled:scale"))
)

pred_dummy1 <- compute(model1, dummy1)$net.result
cat("Dummy Prediction (SwissLabor):", round(pred_dummy1, 4), "\n")

## Dummy Prediction (SwissLabor): 0.0952

# Convert probability to class label
class_dummy1 <- ifelse(pred_dummy1 > 0.5, "yes", "no")
cat("Dummy Prediction (SwissLabor):", class_dummy1, "\n")

## Dummy Prediction (SwissLabor): no
```



Q7. Fit an ANN model on the mtcars dataset to predict the mpg values based on the hp, wt, disp values.

```
data(mtcars)
str(mtcars)

## 'data.frame':    32 obs. of  11 variables:
##  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
##  $ disp: num  160 160 108 258 360 ...
##  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
##  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ qsec: num  16.5 17 18.6 19.4 17 ...
##  $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
##  $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
##  $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
##  $ carb: num  4 4 1 1 2 1 4 2 2 4 ...

# Normalize all numeric columns
mtcars_scaled <- as.data.frame(scale(mtcars))

# Train-test split
set.seed(123)
n <- nrow(mtcars_scaled)
```

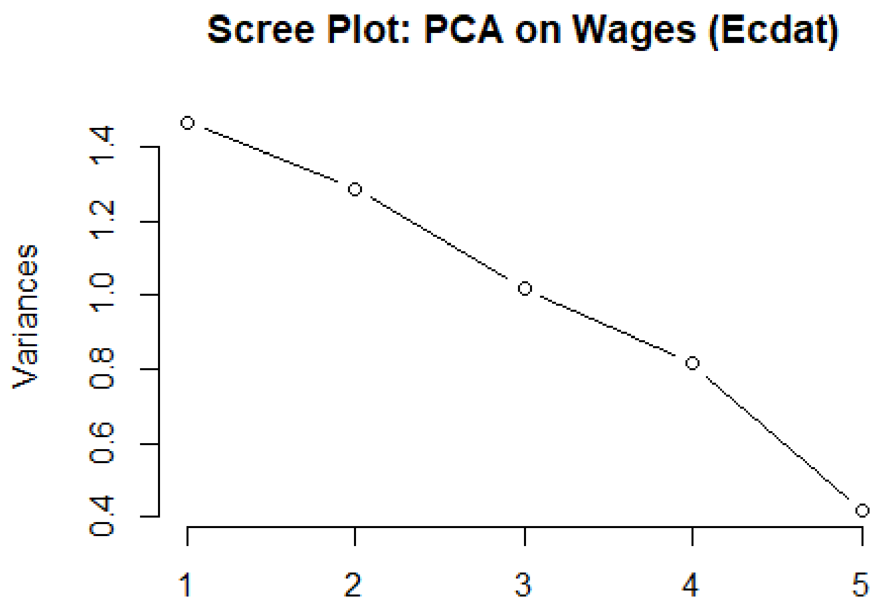
```r
train_idx <- sample(1:n, size = 0.7 * n)
train_data <- mtcars_scaled[train_idx, ]
test_data <- mtcars_scaled[-train_idx, ]

# Train model
model2 <- neuralnet(mpg ~ hp + wt + disp,
                    data = train_data, hidden = 5, linear.output = TRUE)

plot(model2)
# Predict on test set
test_pred2 <- compute(model2, test_data[, c("hp", "wt", "disp")])$net.result
rmse2 <- sqrt(mean((test_pred2 - test_data$mpg)^2))
cat("Test RMSE (mtcars):", round(rmse2, 4), "\n")

## Test RMSE (mtcars): 0.4037

# Dummy input
dummy2 <- data.frame(
  hp = (120 - mean(mtcars$hp)) / sd(mtcars$hp),
  wt = (2.8 - mean(mtcars$wt)) / sd(mtcars$wt),
  disp = (180 - mean(mtcars$disp)) / sd(mtcars$disp)
)

pred_dummy2 <- compute(model2, dummy2)$net.result
# Rescale prediction
pred_mpg <- pred_dummy2 * sd(mtcars$mpg) + mean(mtcars$mpg)
cat("Dummy Prediction (mpg):", round(pred_mpg, 2), "\n")

## Dummy Prediction (mpg): 17.76
```



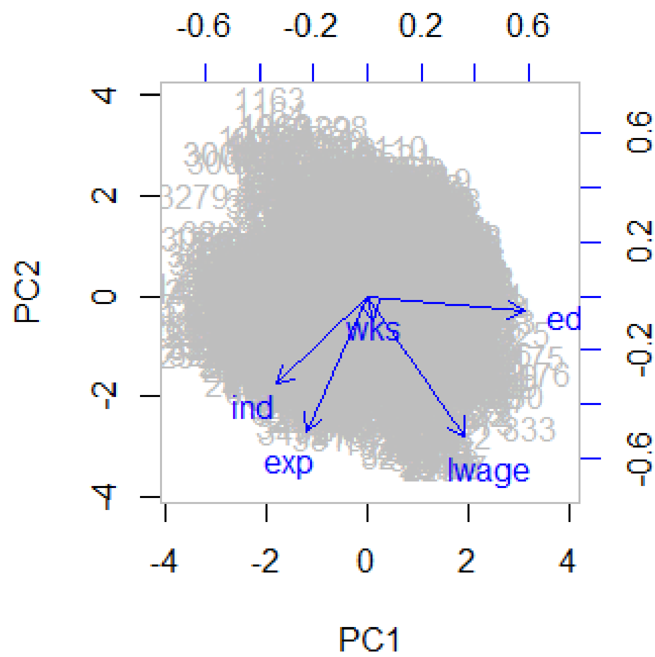
Q8. Analyze relationships between wage-related numeric variables with PCA.

```
# Load and clean data
data(Wages)
Wages_clean <- na.omit(Wages)

# Select numeric variables for PCA
vars <- Wages_clean[, c("exp", "wks", "ind", "ed", "lwage")]

# Perform PCA
pca_wages <- prcomp(vars, center = TRUE, scale. = TRUE)

# View variance explained
summary(pca_wages)

## Importance of components:
##                           PC1    PC2    PC3    PC4    PC5
## Standard deviation     1.2105 1.1345 1.0076 0.9025 0.6465
## Proportion of Variance 0.2931 0.2574 0.2030 0.1629 0.0836
## Cumulative Proportion  0.2931 0.5505 0.7535 0.9164 1.0000
```

```r
# Scree plot (plot the principal components according to the variances
explained by them)
plot(pca_wages, type = "l", main = "Scree Plot: PCA on Wages (Ecdat)")
```




```r
# Biplot for visualizing variable loadings and scores
biplot(pca_wages, scale = 0, col = c("gray", "blue"))
```

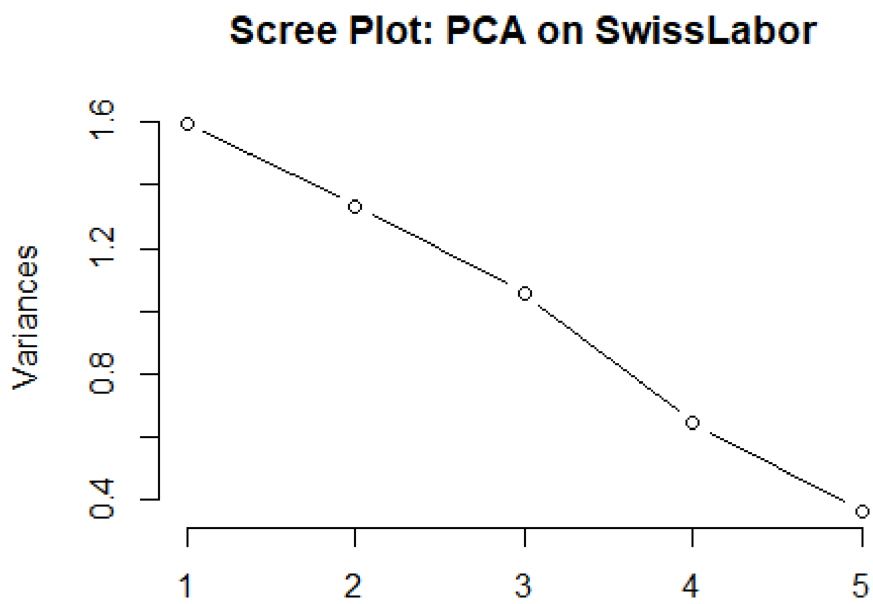


Q9. Reduce socio-economic predictors of labor participation using PCA.

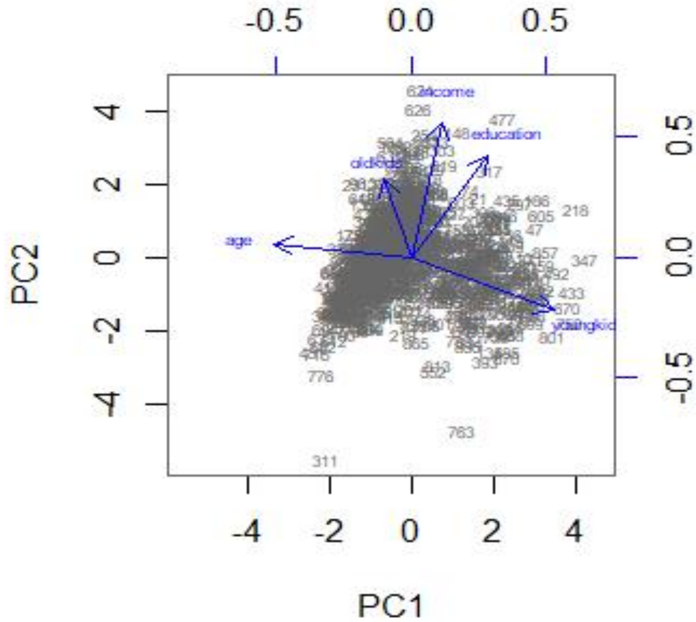
```
data(SwissLabor)
str(SwissLabor)

## 'data.frame':    872 obs. of  7 variables:
##  $ participation: Factor w/ 2 levels "no","yes": 1 2 1 1 1 2 1 2 1 1 ...
##  $ income       : num  10.8 10.5 11 11.1 11.1 ...
##  $ age          : num  3 4.5 4.6 3.1 4.4 4.2 5.1 3.2 3.9 4.3 ...
##  $ education    : num  8 8 9 11 12 12 8 8 12 11 ...
##  $ youngkids    : num  1 0 0 2 0 0 0 0 0 0 ...
##  $ oldkids      : num  1 1 0 0 2 1 0 2 0 2 ...
##  $ foreign      : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...

Swiss_clean <- na.omit(SwissLabor)

# Use relevant numeric predictors
vars <- SwissLabor[, c("income", "age", "education", "youngkids", "oldkids")]


# Perform PCA
pca_swiss <- prcomp(vars, center = TRUE, scale. = TRUE)

# Summary of variance explained
summary(pca_swiss)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4     PC5
## Standard deviation     1.2633 1.1554 1.0289 0.8032 0.60423
## Proportion of Variance 0.3192 0.2670 0.2117 0.1290 0.07302
## Cumulative Proportion  0.3192 0.5862 0.7979 0.9270 1.00000
```

```r
# Scree plot
plot(pca_swiss, type = "l", main = "Scree Plot: PCA on SwissLabor")
```




```r
# Biplot: PC1 vs PC2
biplot(pca_swiss, scale = 0, col = c("gray40", "blue"), cex = 0.5)
```


PC1
PC2
income
education
oldkids
age
youngkid