Statistical Learning — Practical Exam

Gabriele Socrate

2025-11-20

Exercises

Exercise 1

```
load("data_202511201120.RData")
names(data.all)
```

```
##  [1] "Y"               "age"             "sex"             "baseline_sbp"
##  [5] "BMI"             "smoker"          "exercise_level" "salt_intake"
##  [9] "treatment"       "adherence"       "diabetes"
```

```
num.cov = data.all[, -c(1, 3, 6, 7, 9, 11)]
num.cov.scaled = scale(num.cov)
summary(num.cov.scaled)
```

```
##       age             baseline_sbp          BMI              salt_intake
##  Min.   :-1.74088   Min.   :-3.1830   Min.   :-2.81839   Min.   :-3.453407
##  1st Qu.:-0.84030   1st Qu.:-0.6518   1st Qu.:-0.63045   1st Qu.:-0.657471
##  Median :-0.03251   Median :-0.0177   Median :-0.03243   Median : 0.000434
##  Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.000000
##  3rd Qu.: 0.88970   3rd Qu.: 0.6561   3rd Qu.: 0.71888   3rd Qu.: 0.695864
##  Max.   : 1.77533   Max.   : 3.2019   Max.   : 3.52950   Max.   : 2.765711
##    adherence
##  Min.   :-2.94078
##  1st Qu.:-0.69257
##  Median : 0.08794
##  Mean   : 0.00000
##  3rd Qu.: 0.81430
##  Max.   : 1.77298
```

```
set.seed(1120)
km.res = kmeans(num.cov.scaled, centers = 4, nstart = 10)
km.res$centers
```

```
##           age baseline_sbp        BMI salt_intake  adherence
## 1  0.26061101   -0.1715193  0.1445699  -0.3331798 -1.3480678
## 2  0.90858666    0.2343898 -0.1561342   0.6995590  0.3138607
## 3 -0.93926299    0.4109022  0.3968981   0.3220699  0.1734437
## 4 -0.07625028   -0.5837516 -0.4410942  -0.8231014  0.6876488
```

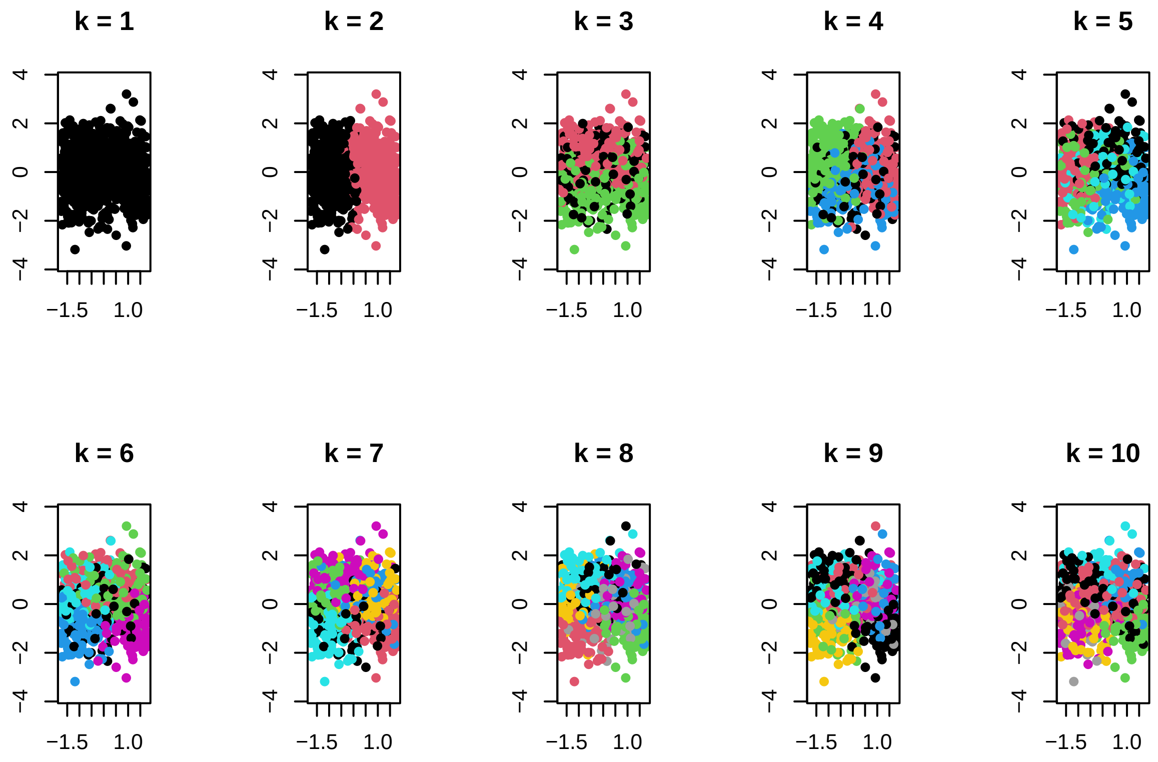
```r
km.res$cluster
```

```
##   [1] 4 4 3 2 3 1 1 3 3 3 3 4 4 2 4 3 1 1 2 2 2 4 1 4 2 2 3 1 3 4 3 3 2 2 4 3 3
##  [38] 4 3 2 4 4 3 1 1 4 2 1 2 2 3 1 2 1 1 4 3 3 3 3 4 4 3 2 4 1 3 4 4 2 2 2 3 4
##  [75] 4 4 4 1 4 1 1 3 3 3 4 4 4 4 3 3 2 2 3 4 3 1 1 1 1 2 1 2 3 4 2 1 3 3 2 1 1
## [112] 4 2 4 3 2 3 1 1 1 2 2 3 2 4 3 1 3 1 1 2 2 4 4 1 4 3 4 4 1 2 2 4 3 1 3 2 4
## [149] 2 1 4 1 1 1 2 4 4 2 2 3 1 3 3 4 2 3 2 3 1 1 3 4 1 2 3 1 3 3 3 1 2 2 4 3 3
## [186] 1 3 4 1 3 1 2 3 1 4 2 2 2 4 3 3 1 1 2 1 2 1 3 1 1 4 1 1 1 3 4 3 2 2 2 2 1
## [223] 1 2 4 4 2 3 4 1 4 2 4 2 1 4 3 3 4 1 2 4 2 1 4 2 3 3 3 3 3 1 1 4 1 4 2 1 1
## [260] 1 4 2 4 3 3 4 2 2 2 2 3 3 1 2 2 1 1 4 3 2 2 2 2 3 2 4 1 2 1 2 1 4 3 3 4 3
## [297] 3 3 1 3 2 4 3 2 4 3 1 2 1 3 1 1 2 3 4 4 2 2 1 1 4 2 4 2 2 4 3 2 2 4 2 3 1
## [334] 3 4 3 3 4 2 4 2 2 2 1 4 3 4 1 2 3 4 2 2 2 2 2 3 4 2 3 2 4 4 4 1 3 1 4 4 3
## [371] 4 4 4 1 2 3 3 4 2 2 4 1 4 4 2 3 2 4 2 1 2 3 3 3 3 3 2 3 2 1 4 4 3 1 4 1 3
## [408] 1 3 1 4 4 1 4 4 2 4 3 4 1 1 3 1 1 3 3 2 4 4 2 3 2 1 2 3 2 1 4 3 4 2 2 4 2
## [445] 4 2 2 4 2 3 3 1 4 3 1 3 4 2 2 3 4 3 2 4 2 3 1 4 2 2 2 1 3 1 3 3 2 3 2 3 2
## [482] 2 2 3 3 3 2 2 3 3 4 2 1 2 2 2 1 3 2 1 4 3 3 4 3 3 1 2 1 3 1 4 3 4 3 4 1 2
## [519] 3 4 1 3 4 3 1 2 1 3 3 4 3 2 1 1 1 1 1 2 3 3 1 4 1 3 2 3 1 1 2 1 3 1 2 4 3
## [556] 3 4 1 4 1 4 1 3 1 3 3 4 3 1 2 2 3 3 1 3 1 2 3 4 3 1 1 2 2 3 4 2 4 2 4 4 4
## [593] 3 3 3 2 4 3 4 4 2 2 1 4 4 1 3 4 3 1 3 2 3 3 4 3 2 3 4 1 3 3 3 4 4 1 3 4 3
## [630] 2 4 3 4 4 2 2 2 3 4 4 3 4 3 4 3 3 2 1 2 2 4 1 2 4 3 3 4 4 2 2 3 2 1 3 3 2
## [667] 2 1 3 1 3 2 1 3 4 4 2 1 1 4 2 3 2 1 2 3 4 3 1 3 1 3 4 2 1 3 4 4 3 3
```

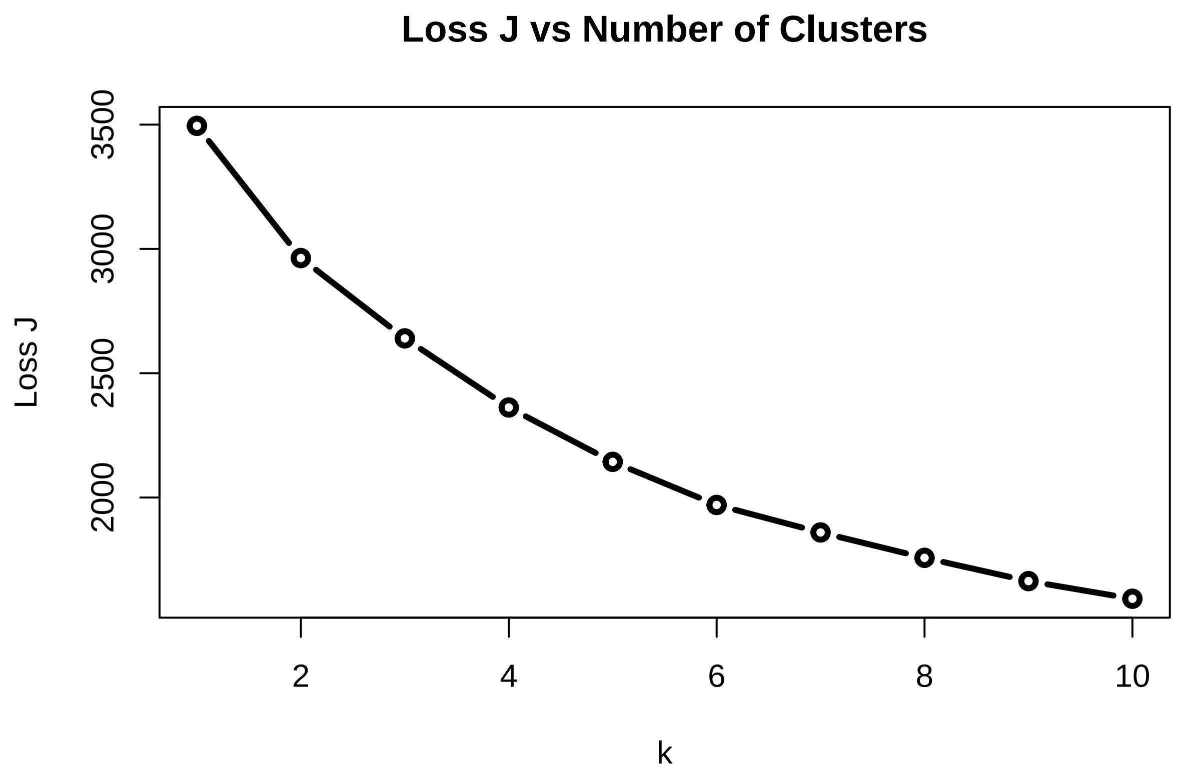
```r
J <- km.res$tot.withinss
J
```

```
## [1] 2362.18
```

```r
J.all = numeric(10)

layout(matrix(1:10, nrow = 2, byrow = TRUE))
for(k in 1:10){
  set.seed(1120)
  kmeans_res = kmeans(num.cov.scaled, centers = k, nstart = 10)

  plot(num.cov.scaled, pch = 16, cex = 1,
       xlab = '', ylab = '', asp = 1,
       col = kmeans_res$cluster,
       main = paste("k =", k))

  J.all[k] = kmeans_res$tot.withinss
}
```



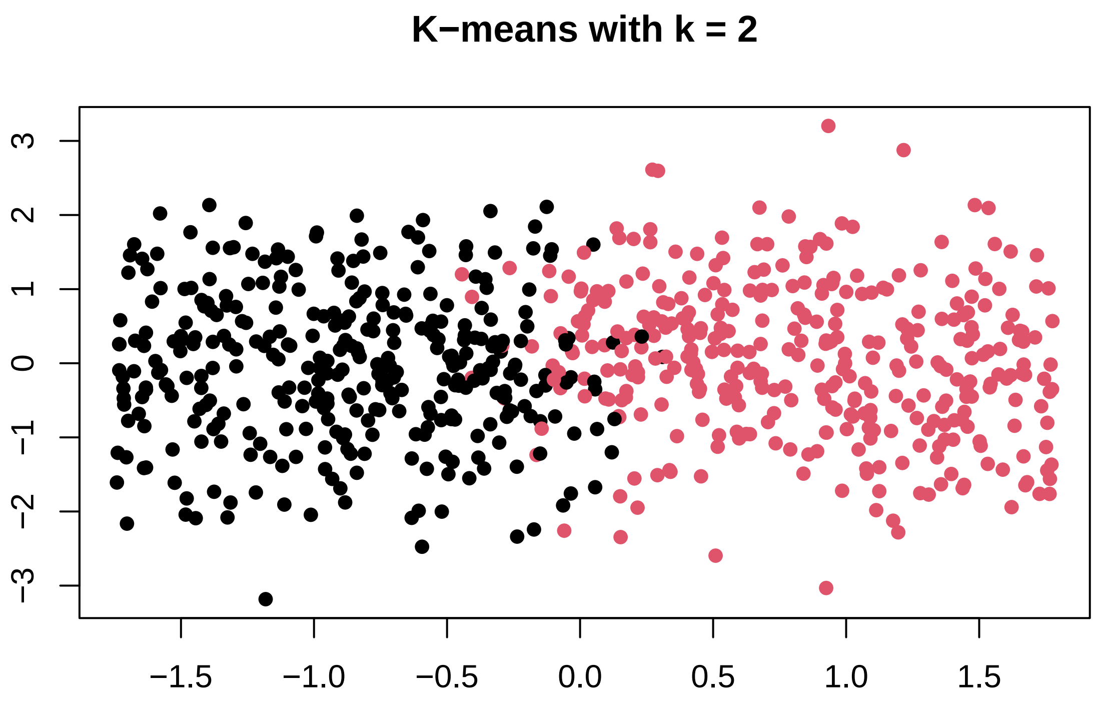
```r
layout(1)
plot(1:10, J.all, type = 'b', lwd = 3,
     xlab = "k", ylab = "Loss J",
     main = "Loss J vs Number of Clusters")
```



The plot of does not show a clear elbow point: the decrease in the loss is smooth and gradual for all values of k from 1 to 10. When inspecting the cluster plots, none of the solutions reveals well-separated or naturally distinguishable groups, and the partitions appear mostly arbitrary. For these reasons, the data do not seem to exhibit a meaningful cluster structure, and no particular value of k is clearly supported.

Although the plot of J(k) does not show a clear elbow and the data do not seem to contain well-separated groups, if we are required to select a number of clusters with k>1, the most reasonable choice is k=2. The decrease in the loss function is largest when moving from k=1 to k=2, while further increases in k lead to smaller marginal reductions. The plot for k=2 shows a mild but visible separation into two groups, whereas higher values of k produce partitions that appear mostly arbitrary. Therefore, k=2 represents the most interpretable and justifiable clustering solution under the constraint k>1.

```
set.seed(1120)
km.final <- kmeans(num.cov.scaled, centers = 2, nstart = 10)

plot(num.cov.scaled,
     pch = 16, cex = 1,
     col = km.final$cluster,
     xlab = "", ylab = "",
     main = "K-means with k = 2")
```



The algorithm assigns each observation to the cluster whose center (mean vector) is closest in Euclidean distance, so each group collects points that lie closer to one centroid than to the other. However, the two clusters strongly overlap, and no clear separation emerges. This is consistent with the behavior of the loss J(k), which does not display a clear elbow. So, the solution with k=2 should be seen just as a rough split of the data based on distance to the two centers, not as two real or clearly separated groups. ## Exercise 2

```r
set.seed(1120)
Y_cat = factor(ifelse(data.all$Y < 17,0,1))
data.all.cat = data.all
data.all.cat$Y = Y_cat
n = dim(data.all)[1]
select.train = sample(1:n,n*7/10)
train.cat = data.all.cat[select.train,]
test.cat = data.all.cat[-select.train,]
summary(train.cat)
```

```
##  Y            age             sex         baseline_sbp          BMI
##  0:296   Min.   :30.04   Male  :233   Min.   : 99.41   Min.   :15.99
##  1:194   1st Qu.:42.69   Female:257   1st Qu.:137.90   1st Qu.:24.62
##          Median :54.07                Median :148.56   Median :27.03
##          Mean   :54.44                Mean   :148.66   Mean   :27.13
##          3rd Qu.:66.75                3rd Qu.:159.43   3rd Qu.:29.97
##          Max.   :79.91                Max.   :189.55   Max.   :40.74
##  smoker     exercise_level  salt_intake       treatment     adherence
##  No :355   Low     :197    Min.   : 1.124   Control:188   Min.   :0.1260
##  Yes:135   Moderate:206    1st Qu.: 6.655   DrugA  :154   1st Qu.:0.5376
##             High    : 87    Median : 7.963   DrugB  :148   Median :0.6809
##                             Mean   : 7.985                 Mean   :0.6631
##                             3rd Qu.: 9.348                 3rd Qu.:0.8166
##                             Max.   :13.433                 Max.   :0.9784
##  diabetes
##  No :373
##  Yes:117
##
##
##
##
```

```
library(e1071)
svm.model <- svm(
  Y ~ . ,
  data   = train.cat,
  kernel = "radial",
  cost   = 10,
  gamma  = 1,
  scale = TRUE
)


pred.train <- predict(svm.model, newdata = train.cat)
pred.test  <- predict(svm.model, newdata = test.cat)
acc.train <- mean(pred.train == train.cat$Y)
acc.test  <- mean(pred.test  == test.cat$Y)
acc.train
```

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

```
acc.test
```

```
## [1] 0.7142857
```

```
library(caret)
```

```
## Warning: il pacchetto 'caret' è stato creato con R versione 4.4.3
```

```
## Caricamento del pacchetto richiesto: ggplot2
```

```
## Caricamento del pacchetto richiesto: lattice
```

```
set.seed(1120)
tune.radial = tune(
  svm,
  Y ~ .,
  data   = train.cat,
  kernel = "radial",
  ranges = list(
    cost  = c(0.1, 1, 10),
    gamma = c(1, 2)
  ),
  scale = TRUE
)

summary(tune.radial)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##    10     1
##
## - best performance: 0.277551
##
## - Detailed performance results:
##   cost gamma     error dispersion
## 1  0.1     1 0.3959184 0.08117787
## 2  1.0     1 0.3040816 0.07723411
## 3 10.0     1 0.2775510 0.07017032
## 4  0.1     2 0.3959184 0.08117787
## 5  1.0     2 0.3979592 0.07830520
## 6 10.0     2 0.3979592 0.08120637
```

```
best.svm = tune.radial$best.model

pred.svm = predict(best.svm, newdata = test.cat)
confusionMatrix(pred.svm, test.cat$Y)
```

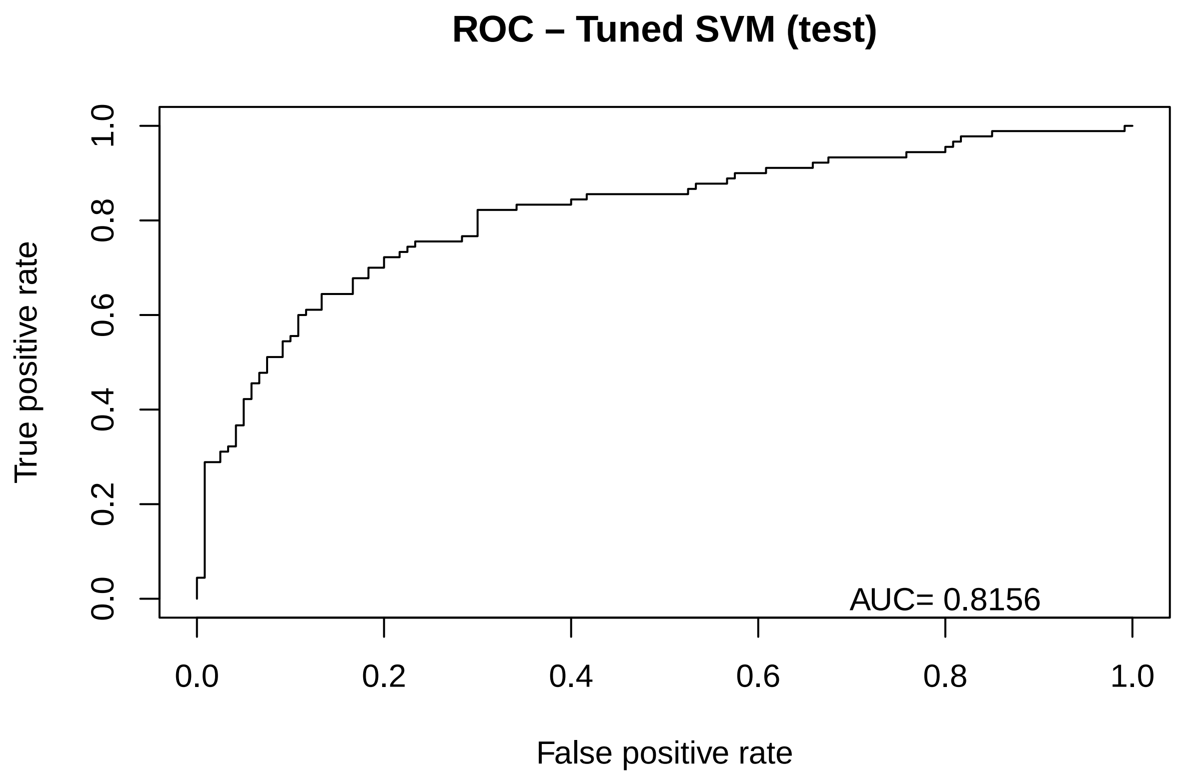
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 114  54
##          1   6  36
##
##                Accuracy : 0.7143
##                  95% CI : (0.6481, 0.7743)
##     No Information Rate : 0.5714
##     P-Value [Acc > NIR] : 1.348e-05
##
##                   Kappa : 0.375
##
##  Mcnemar's Test P-Value : 1.298e-09
##
##             Sensitivity : 0.9500
##             Specificity : 0.4000
##          Pos Pred Value : 0.6786
##          Neg Pred Value : 0.8571
##               Prevalence : 0.5714
##           Detection Rate : 0.5429
##     Detection Prevalence : 0.8000
##        Balanced Accuracy : 0.6750
##
##         'Positive' Class : 0
##
```

The best model is obtained with cost = 10 and gamma = 1, which achieves the lowest cross-validation error (0.278). On the test set, this SVM reaches an accuracy of about 71.4%, above the No Information Rate. Since the positive class is 0, the sensitivity is very high (0.95), meaning that most observations in class 0 are correctly classified. The specificity is much lower (0.40), showing that the model is less accurate in detecting class 1.

```r
library(ROCR)
```

```
## Warning: il pacchetto 'ROCR' è stato creato con R versione 4.4.2
```

```r
rocplot=function(pred, truth, ...){
  predob = prediction(pred, truth)
  perf = performance(predob , "tpr", "fpr")
  plot(perf ,...)
  auc = performance(predob,"auc")
  auc <- auc@y.values[[1]]
  auc = signif(auc,4)
  text(0.8,0,paste("AUC=",auc))
}


fitted.best.test = -attributes(
  predict(svm.model, test.cat, decision.values = TRUE)
)$decision.values

rocplot(fitted.best.test, test.cat$Y,
        main="ROC - Tuned SVM (test)")
```



The tuned SVM and the original SVM actually use the same hyperparameters (the tuning step confirms the initial choice), so their ROC curves and AUC are expected to be essentially the same. On the test set, the model achieves an AUC of 0.8156, indicating good discrimination between the two classes across thresholds. Overall, tuning the hyperparameters improves the model's generalization ability and leads to a clear increase in predictive accuracy. ## Exercise 3

```
n = dim(data.all)[1]
set.seed(1120)
select.train = sample(1:n,n*7/10)
train = data.all[select.train,]
test = data.all[-select.train,]

library(gam)
```

```
## Warning: il pacchetto 'gam' è stato creato con R versione 4.4.3

## Caricamento del pacchetto richiesto: splines

## Caricamento del pacchetto richiesto: foreach

## Loaded gam 1.22-6
```

```
gam.model <- gam(
  Y ~  s(age, df = 3) +
       s(baseline_sbp, df = 3) +
```

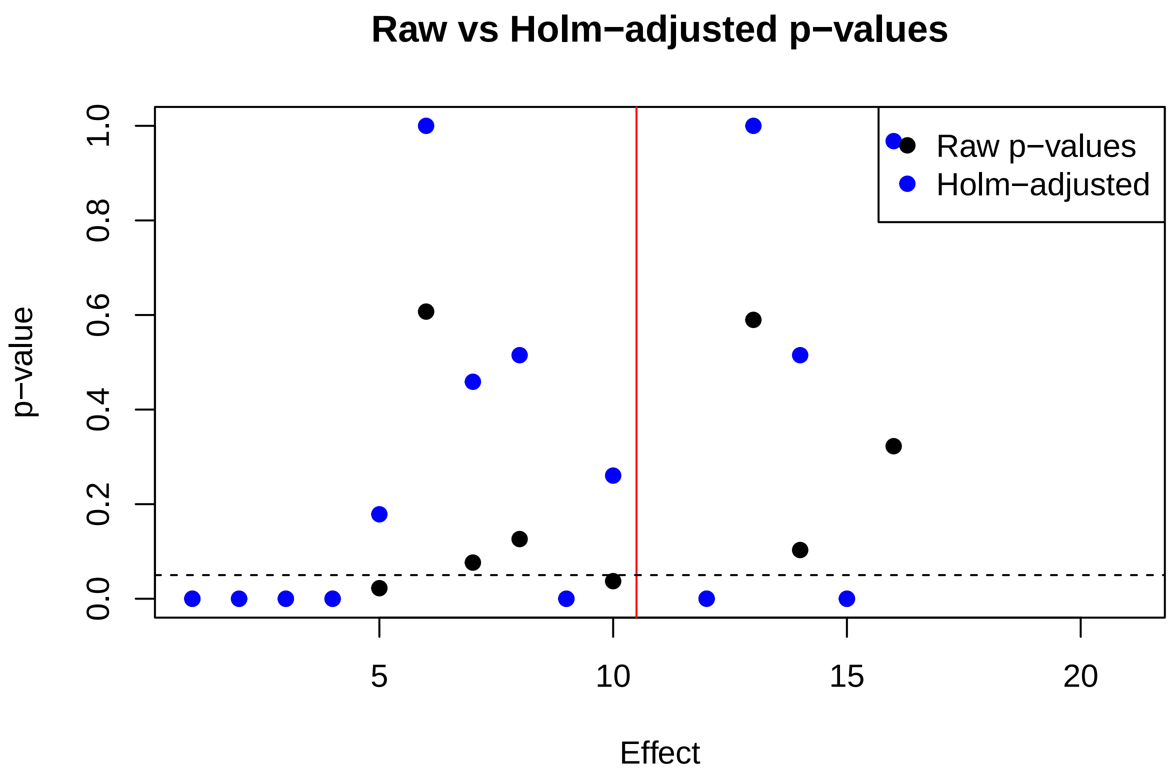
```r
       s(BMI, df = 3) +
       s(salt_intake, df = 3) +
       s(adherence, df = 3) +
    sex + smoker + exercise_level + treatment + diabetes,
  data = train
)


pred.train.gam <- predict(gam.model, newdata = train)
pred.test.gam  <- predict(gam.model, newdata = test)
mse.train <- mean((train$Y - pred.train.gam)^2)
mse.test  <- mean((test$Y - pred.test.gam)^2)

mse.train
```

```
## [1] 121.845
```

```r
mse.test
```

```
## [1] 77.18914
```

```r
gam.sum <- summary(gam.model)

p <- nrow(gam.sum$parametric.anova) - 1
p.values.par <- gam.sum$parametric.anova[1:p, 5]
p.values.nonpar <- gam.sum$anova[, 3]

p.values <- c(p.values.par, p.values.nonpar)

holm.p.values <- p.adjust(p.values, method = "holm")

plot(p.values, col = "black", pch = 19, ylim = c(0,1),
     xlab = "Effect", ylab = "p-value", main="Raw vs Holm-adjusted p-values")

points(holm.p.values, col = "blue", pch = 19)
abline(v = p + 0.5, col = "red")    # separates linear vs smooth
abline(h = 0.05, lty = 2)
legend("topright", legend=c("Raw p-values", "Holm-adjusted"),
       col=c("black","blue"), pch=19)
```

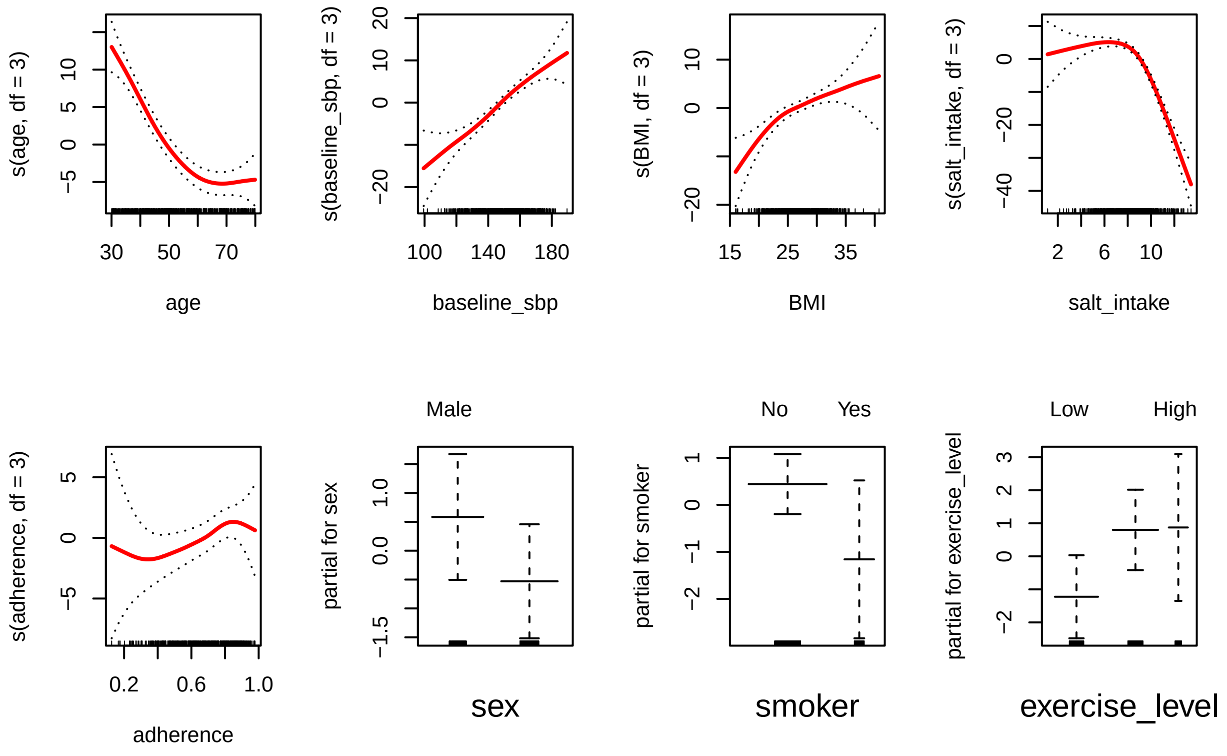


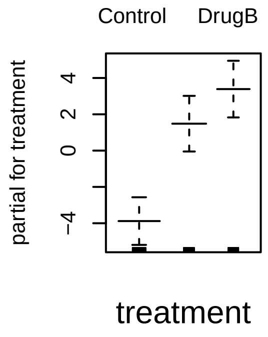
From the plot comparing raw and Holm-adjusted p-values, we observe that the Holm correction (controlling the Family-Wise Error Rate) makes the significance criterion more stringent, so fewer smooth terms remain significant at the 0.05 level. Several terms have small raw p-values, but after Holm adjustment only the strongest effects stay below the 0.05 threshold. Overall, this indicates that while the GAM initially suggests multiple potential nonlinear relationships, only the most statistically robust nonlinear effects are supported once multiple testing is taken into account.

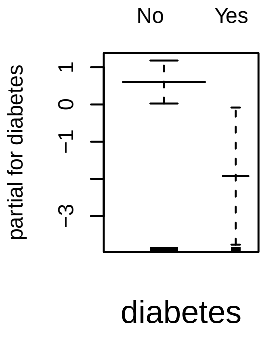
```r
summary(gam.model)
```

```
##
## Call: gam(formula = Y ~ s(age, df = 3) + s(baseline_sbp, df = 3) +
##     s(BMI, df = 3) + s(salt_intake, df = 3) + s(adherence, df = 3) +
##     sex + smoker + exercise_level + treatment + diabetes, data = train)
## Deviance Residuals:
##        Min         1Q     Median         3Q        Max
## -108.81721   -4.71090    0.05131    5.36467   29.45625
##
## (Dispersion Parameter for gaussian family taken to be 127.8471)
##
##     Null Deviance: 136155.9 on 489 degrees of freedom
## Residual Deviance: 59704.55 on 466.9998 degrees of freedom
## AIC: 3791.912
##
## Number of Local Scoring Iterations: NA
##
## Anova for Parametric Effects
##                          Df Sum Sq Mean Sq  F value    Pr(>F)    
## s(age, df = 3)             1  15896 15896.3 124.3387 < 2.2e-16 ***
## s(baseline_sbp, df = 3)    1  12013 12012.7  93.9619 < 2.2e-16 ***
## s(BMI, df = 3)             1   4750  4750.2  37.1557 2.286e-09 ***
## s(salt_intake, df = 3)     1  18855 18854.6 147.4777 < 2.2e-16 ***
## s(adherence, df = 3)       1    672   671.9   5.2556   0.02232 *  
## sex                        1     34    33.8   0.2647   0.60717    
## smoker                     1    403   403.0   3.1525   0.07646 .  
## exercise_level             2    532   265.8   2.0791   0.12620    
## treatment                  2   4681  2340.7  18.3082 2.214e-08 ***
## diabetes                   1    558   558.2   4.3661   0.03720 *  
## Residuals                467  59705   127.8                      
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                         Npar Df Npar F     Pr(F)    
## (Intercept)                                         
## s(age, df = 3)                2 11.287 1.632e-05 ***
## s(baseline_sbp, df = 3)       2  0.529    0.5897    
## s(BMI, df = 3)                2  2.284    0.1030    
## s(salt_intake, df = 3)        2 57.288 < 2.2e-16 ***
## s(adherence, df = 3)          2  1.134    0.3226    
## sex                                                 
## smoker                                              
## exercise_level                                      
## treatment                                           
## diabetes                                            
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
layout(matrix(1:8,nrow=2,byrow=TRUE))
plot.Gam(gam.model , se=TRUE , col ="red",lwd=2)
```


s(age, df = 3)
age
s(baseline_sbp, df = 3)
baseline_sbp
s(BMI, df = 3)
BMI
s(salt_intake, df = 3)
salt_intake
s(adherence, df = 3)
adherence
partial for sex
Male
sex
partial for smoker
No
Yes
smoker
partial for exercise_level
Low
High
exercise_level





As we can see from the smooth plots and from the ANOVA table for nonparametric effects, only age and salt_intake show a statistically significant non-linear effect (both have $p < 0.05$ in the nonparametric test). In contrast, the smooth terms for baseline_sbp, BMI, and adherence are not significantly non-linear (their nonparametric p-values are > 0.05), so their effects can be reasonably approximated as linear (or treated as weak/non-essential nonlinearities).

From the parametric part of the model and the corresponding partial-effect plots, the categorical variables sex and exercise_level are not significant (p>0.05) and their partial effects largely overlap around zero. Smoker is also not significant at the 5% level (it is only borderline at the 10% level). On the other hand, treatment is clearly significant and shows a strong difference between Control and DrugB, and diabetes is significant as well, with a visible shift between levels.

```r
best.gam<- gam(
  Y ~  s(age, df = 3) +
       baseline_sbp +
       BMI +
       s(salt_intake, df = 3) +
       adherence +
    treatment + diabetes,
  data = train
)


pred.train.gam <- predict(best.gam, newdata = train)
pred.test.gam  <- predict(best.gam, newdata = test)
mse.train <- mean((train$Y - pred.train.gam)^2)
mse.test  <- mean((test$Y - pred.test.gam)^2)
mse.train
```

```
## [1] 125.5153
```

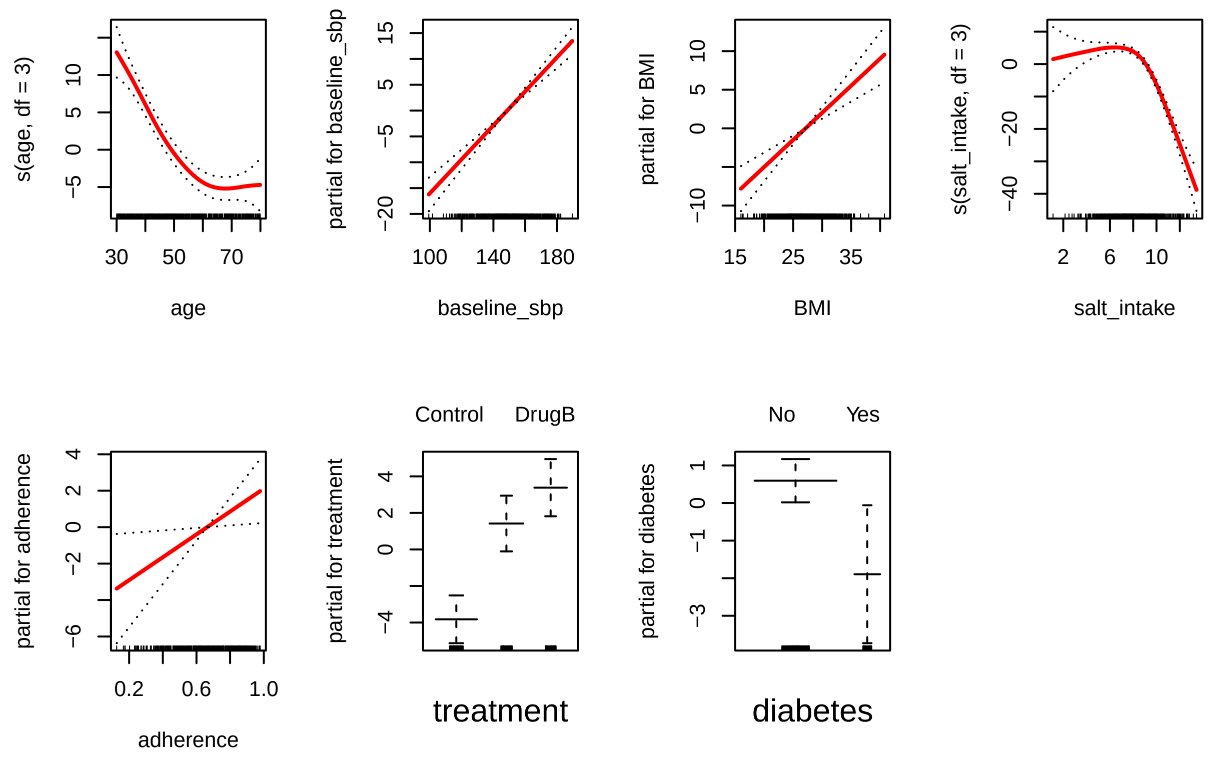
```r
mse.test
```

```
## [1] 82.36546
```

```r
summary(best.gam)
```

```
##
## Call: gam(formula = Y ~ s(age, df = 3) + baseline_sbp + BMI + s(salt_intake,
##     df = 3) + adherence + treatment + diabetes, data = train)
## Deviance Residuals:
##       Min        1Q    Median        3Q       Max
## -107.2509   -4.7632    0.5207    5.6693   31.6731
##
## (Dispersion Parameter for gaussian family taken to be 128.9369)
##
##     Null Deviance: 136155.9 on 489 degrees of freedom
## Residual Deviance: 61502.86 on 476.9998 degrees of freedom
## AIC: 3786.453
##
## Number of Local Scoring Iterations: NA
##
## Anova for Parametric Effects
##                         Df Sum Sq Mean Sq  F value    Pr(>F)
## s(age, df = 3)           1  15947 15947.1 123.6817 < 2.2e-16 ***
## baseline_sbp             1  11844 11843.9  91.8579 < 2.2e-16 ***
## BMI                      1   4714  4713.6  36.5575 2.994e-09 ***
## s(salt_intake, df = 3)   1  19003 19003.0 147.3819 < 2.2e-16 ***
## adherence                1    764   764.5   5.9292   0.01526 *
## treatment                2   4634  2317.0  17.9699 2.991e-08 ***
## diabetes                 1    549   549.4   4.2611   0.03954 *
## Residuals              477  61503   128.9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                        Npar Df Npar F     Pr(F)
## (Intercept)
## s(age, df = 3)               2 11.455 1.383e-05 ***
## baseline_sbp
## BMI
## s(salt_intake, df = 3)       2 60.123 < 2.2e-16 ***
## adherence
## treatment
## diabetes
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
layout(matrix(1:8,nrow=2,byrow=TRUE))
plot.Gam(best.gam , se=TRUE , col ="red",lwd=2)
```



Age shows a clear nonlinear decreasing effect: younger patients experience a larger reduction in systolic pressure, while the improvement becomes progressively smaller with increasing age, so older patients benefit less. Salt intake has a nonlinear effect that is relatively stable at low–moderate levels and decreases only for very high daily salt consumption, suggesting that excessive salt intake reduces the effectiveness of the treatment.

In contrast, baseline_sbp, BMI, and adherence have significant linear effects in the final model (all with $p < 0.05$), while their nonlinear components are not significant, so a linear specification is appropriate for these covariates. Treatment group plays an important role: patients in the control arm achieve the smallest reduction, whereas DrugB shows a clearly larger reduction in systolic pressure. Finally, diabetes has a negative impact on the outcome: diabetic patients tend to experience a lower reduction in systolic blood pressure compared to non-diabetic individuals.

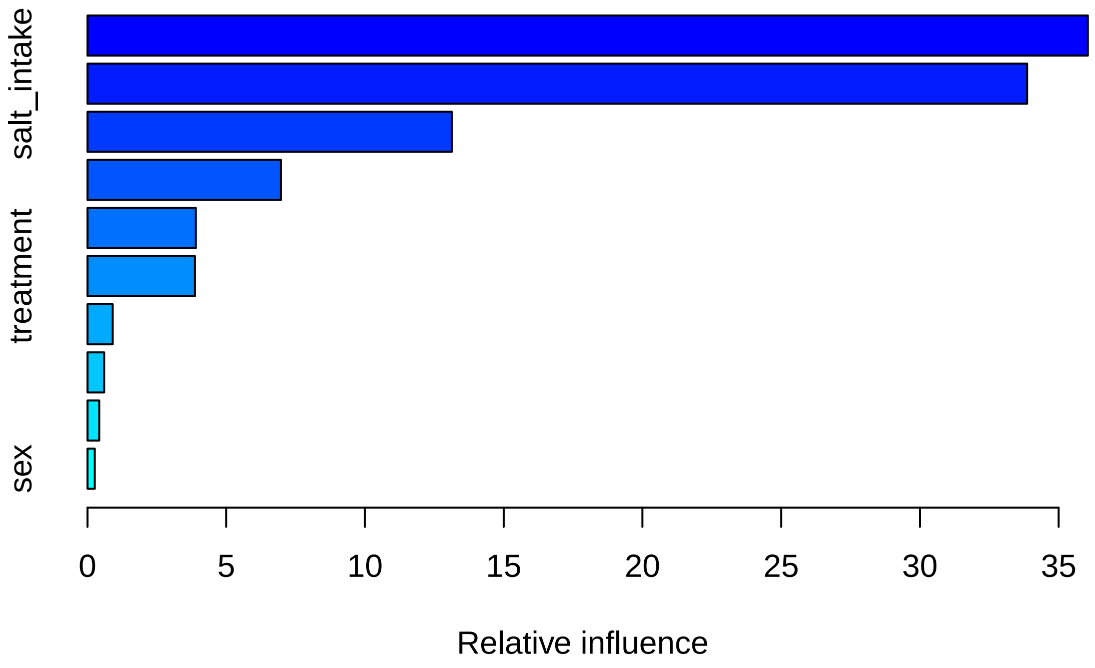
Exercise 4

```
set.seed(1120)
library(gbm)
```

```
## Warning: il pacchetto 'gbm' è stato creato con R versione 4.4.3
```

```
## Loaded gbm 2.2.2
## This version of gbm is no longer under development. Consider transitioning to gbm3, https://github.c
```

```
boost.model = gbm(
  Y ~ .,
  data              = train,
  distribution      = "gaussian",
  n.trees           = 5000,
  interaction.depth = 4,
  shrinkage         = 0.01
)

summary(boost.model)
```



```
##                             var    rel.inf
## age                         age 36.0529157
## salt_intake         salt_intake 33.8648213
## baseline_sbp       baseline_sbp 13.1278944
## BMI                         BMI  6.9743856
## adherence             adherence  3.9051428
## treatment             treatment  3.8749759
## exercise_level exercise_level  0.9099273
## diabetes               diabetes  0.6021578
## smoker                   smoker  0.4240749
## sex                         sex  0.2637043
```

```r
yhat.train = predict(boost.model, newdata = train, n.trees = 5000)
MSE.train  = mean((train$Y - yhat.train)^2)

yhat.test = predict(boost.model, newdata = test, n.trees = 5000)
MSE.test  = mean((test$Y - yhat.test)^2)

MSE.train
```

```
## [1] 6.310776
```

```r
MSE.test
```

```
## [1] 35.08872
```

```r
d.explore = 1:7
params = data.frame(d = d.explore)


set.seed(1120)
indexes = sample(1:2, size = nrow(train), replace = TRUE, prob = c(0.8, 0.2))
subtrain = train[indexes == 1, ]
subtest  = train[indexes == 2, ]

MSE = numeric(nrow(params))

for(ii in 1:nrow(params)){
  model.boost = gbm(
    Y ~ .,
    data = subtrain,
    distribution = "gaussian",
    n.trees = 5000,
    interaction.depth = params$d[ii],
    shrinkage = 0.01,
    verbose = FALSE
  )

  pred = predict(model.boost, newdata = subtest, n.trees = 5000)

  MSE[ii] = mean( (pred - subtest$Y)^2 )
}

opt.index  = which.min(MSE)
opt.params = params[opt.index, ]
opt.params
```

```
## [1] 3
```

```r
MSE_min = MSE[opt.index]
MSE_min
```

```
## [1] 37.07161
```

By evaluating different values of the interaction depth d from 1 to 7 on an internal validation split, the smallest MSE is obtained for d=3. The interaction depth controls how many interaction effects the boosting model can capture: d=1 allows only additive (non-interacting) effects, larger values of d allow interactions between predictors. An optimal value of d=3 indicates that the model benefits from including up to third-order interactions, meaning that the relationship between the predictors and the response is not purely additive but involves meaningful combinations of multiple variables.

The variable importance plot shows that age and salt intake are by far the most informative predictors, with relative influence values above 30%. These two covariates contribute the most to reducing prediction error, meaning that variation in blood-pressure reduction after 6 months is strongly driven by the patient's age and daily salt consumption. Overall, the boosting model suggests that blood-pressure reduction is mainly associated with age and salt-intake patterns, while the remaining covariates contribute substantially less to the predictive performance.

Exercise 5

```r
lm.model = lm(Y ~ ., data = train)
summary(lm.model)
```

```
##
## Call:
## lm(formula = Y ~ ., data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -123.128   -5.418    1.242    6.723   29.717
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)             -10.64911    7.93549  -1.342 0.180247
## age                      -0.39225    0.04135  -9.487  < 2e-16 ***
## sexFemale                -1.53321    1.17084  -1.309 0.190997
## baseline_sbp              0.28759    0.03717   7.737 6.10e-14 ***
## BMI                       0.75773    0.14810   5.116 4.52e-07 ***
## smokerYes                -1.92669    1.30648  -1.475 0.140947
## exercise_levelModerate    3.20281    1.28610   2.490 0.013102 *
## exercise_levelHigh        2.65497    1.65538   1.604 0.109411
## salt_intake              -3.03806    0.29035 -10.463  < 2e-16 ***
## treatmentDrugA            5.30902    1.40917   3.767 0.000186 ***
## treatmentDrugB            7.16256    1.41035   5.079 5.46e-07 ***
## adherence                 4.70773    3.14938   1.495 0.135625
## diabetesYes              -2.20343    1.36503  -1.614 0.107144
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 12.75 on 477 degrees of freedom
## Multiple R-squared:  0.4303, Adjusted R-squared:  0.4159
## F-statistic: 30.02 on 12 and 477 DF,  p-value: < 2.2e-16
```

```r
yhat.test = predict(lm.model, newdata = test)
MSE.test.lm  = mean((test$Y - yhat.test)^2)

MSE.test.lm
```

```
## [1] 111.7485
```

```r
boost.model = gbm(
  Y ~ .,
  data              = train,
  distribution      = "gaussian",
  n.trees           = 5000,
  interaction.depth = 3,
  shrinkage         = 0.01
)

yhat.train = predict(boost.model, newdata = train, n.trees = 5000)
MSE.train  = mean((train$Y - yhat.train)^2)
yhat.test = predict(boost.model, newdata = test, n.trees = 5000)
MSE.test  = mean((test$Y - yhat.test)^2)

MSE.train
```

```
## [1] 9.589668
```

```r
MSE.test
```

```
## [1] 34.76147
```

Based on the test MSE values, the boosting model is clearly the best-performing one, with an error of about 34 compared to 111 for the linear model and 82 for the GAM. This indicates that the relationship between the covariates and the reduction in systolic blood pressure is not well captured by additive (as we saw before d = 3 not 1) or linear structures, while boosting successfully models the nonlinear effects and interactions in the data. The results show that age and salt intake are the strongest predictors of blood-pressure reduction, with BMI, adherence, and treatment also contributing but to a smaller extent. A further improvement could be obtained by tuning the number of trees used in the boosting model. In addition, the shrinkage parameter could also be tuned.