STAT 32950 Take Home Exam

Bin Yu

May 23, 2025

Note: In order to maintain the result the same and for replicability, I set all random seed to 2025 before doing the cross validation.

Question 1

(a) LASSO Regression ($\alpha = 1$)

We fit the LASSO model and plot the coefficient paths as a function of $\log(\lambda)$:

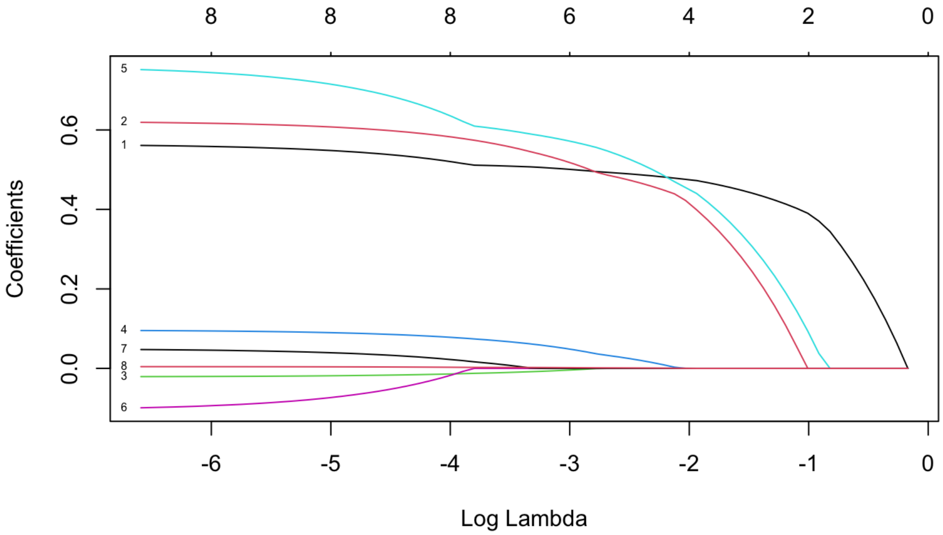

Figure 1: LASSO Coefficient Estimates

The 10-fold cross-validation yields

$$\lambda_{\min} = 0.008050924, \qquad \lambda_{1\text{se}} = 0.2292932.$$

At $\lambda_{1\text{se}}$, the estimated coefficients are:

```
                s1
(Intercept) 0.9340858
```

```
lcavol      0.4409790
lweight     0.2432206
age         .
lbph        .
svi         0.3064360
lcp         .
gleason     .
pgg45       .
```

Hence the fitted LASSO model is

$$E(\text{lpsa}) = 0.9340858 + 0.4409790\,(\text{lcavol}) + 0.2432206\,(\text{lweight}) + 0.3064360\,(\text{svi}).$$

(b) Ridge Regression ($\alpha = 0$)

We fit the ridge model and plot the coefficient paths:

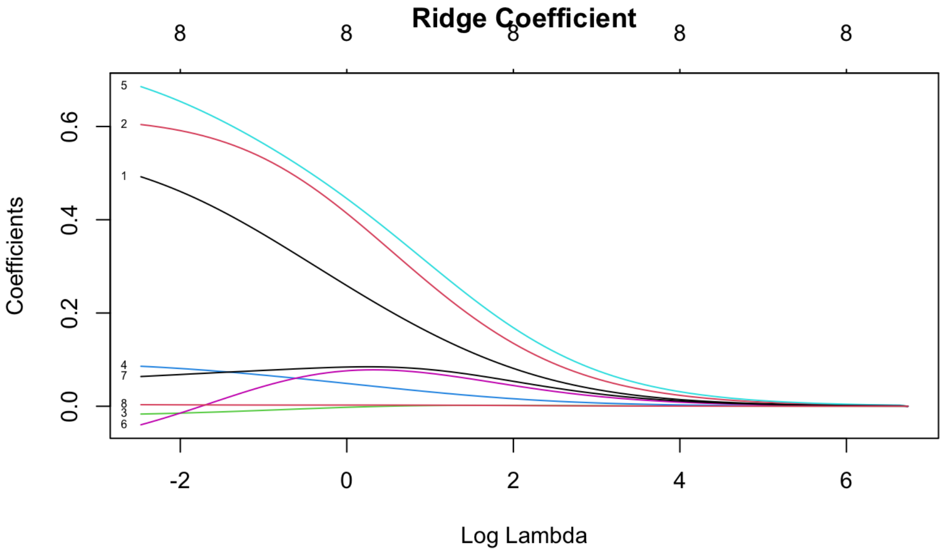

Figure 2: Ridge Coefficient Estimates

Cross-validation yields

$$\lambda_{\text{min}} = 0.08434274, \qquad \lambda_{\text{1se}} = 1.141198.$$

At λ_{1se}, the estimated coefficients are:

```
        s1
(Intercept)  0.097948354
lcavol       0.244343669
lweight      0.394356057
age         -0.001540253
lbph         0.046467792
svi          0.427783630
lcp          0.077476930
gleason      0.084521141
pgg45        0.002613931
```

Hence the fitted ridge model is

$$E(\text{lpsa}) = 0.097948354 + 0.244343669\,(\text{lcavol}) + 0.394356057\,(\text{lweight}) - 0.001540253\,(\text{age}) + 0.046467792\,(\text{lbph})$$

$$+0.427783630\,(\text{svi}) + 0.077476930\,(\text{lcp}) + 0.084521141\,(\text{gleason}) + 0.002613931\,(\text{pgg45}).$$

(c) Elastic Net ($\alpha = 0.4$)

We fit the elastic net model and plot the coefficient paths:

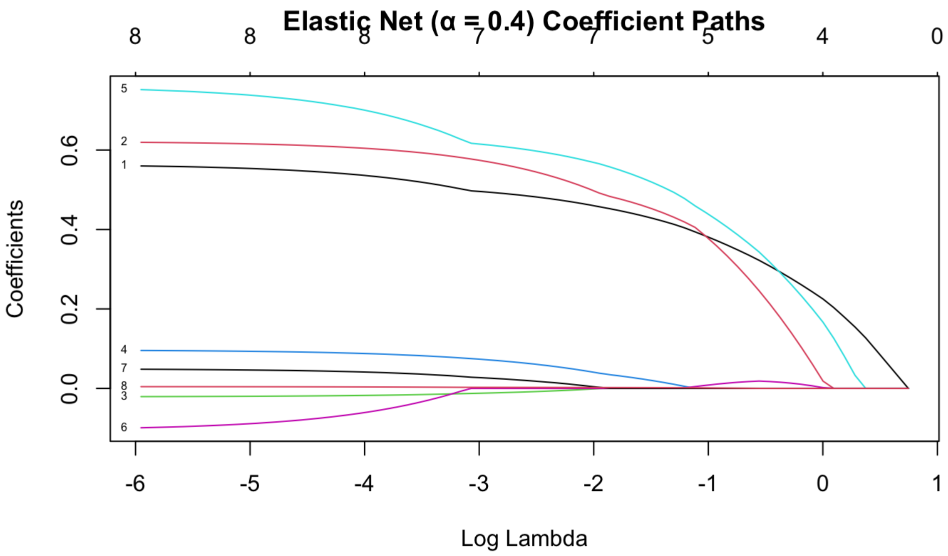

Figure 3: Elastic Net ($\alpha = 0.4$) Coefficient Estimates

Cross-validation yields

$$\lambda_{\text{min}} = 0.002599542, \qquad \lambda_{\text{1se}} = 0.2988842.$$

At λ_{1se}, the estimated coefficients are:

```
                s1
(Intercept) 0.272154037
lcavol      0.404223642
lweight     0.418849782
age         .
lbph        0.005155820
svi         0.477976162
lcp         0.001266486
gleason     .
pgg45       0.001508343
```

Hence the fitted elastic net model is

$$E(\text{lpsa}) = 0.272154037 + 0.404223642\,(\text{lcavol}) + 0.418849782\,(\text{lweight}) + 0.005155820\,(\text{lbph})$$

$$+0.477976162\,(\text{svi}) + 0.001266486\,(\text{lcp}) + 0.001508343\,(\text{pgg45}).$$

(d)

(i)

We first check the correlation matrix among explanatory variables:

```
        lcavol lweight  age  lbph   svi   lcp gleason pgg45
lcavol    1.00    0.28 0.22  0.03  0.54  0.68    0.43  0.43
lweight   0.28    1.00 0.35  0.44  0.16  0.16    0.06  0.11
age       0.22    0.35 1.00  0.35  0.12  0.13    0.27  0.28
lbph      0.03    0.44 0.35  1.00 -0.09 -0.01    0.08  0.08
svi       0.54    0.16 0.12 -0.09  1.00  0.67    0.32  0.46
lcp       0.68    0.16 0.13 -0.01  0.67  1.00    0.51  0.63
gleason   0.43    0.06 0.27  0.08  0.32  0.51    1.00  0.75
pgg45     0.43    0.11 0.28  0.08  0.46  0.63    0.75  1.00
```

We could see all three methods can be seen as solving penalized least-squares problems:

$$\hat{\beta}^{\text{lasso}} = \arg\min_{\beta} \Big\{ \|Y - X\beta\|_2^2 \; + \; \lambda\|\beta\|_1 \Big\},$$

$$\hat{\beta}^{\text{ridge}} = \arg\min_{\beta} \Big\{ \|Y - X\beta\|_2^2 \; + \; \lambda\|\beta\|_2^2 \Big\},$$

$$\hat{\beta}^{\text{EN}} = \arg\min_{\beta} \Big\{ \|Y - X\beta\|_2^2 + \lambda\big[(1-\alpha)\|\beta\|_2^2 + \alpha\|\beta\|_1\big] \Big\},$$

where $X'X$ may be ill-conditioned.

Even when $n > p$, if $X'X$ has very small eigenvalues (i.e. is nearly singular), the ordinary least-squares estimator

$$\hat{\beta}_{\text{LS}} = (X'X)^{-1}X'Y$$

acquires huge variance and becomes unstable.

By adding the penalty term, we replace $(X'X)^{-1}$ with

$$(X'X + \lambda I)^{-1} \quad \text{or} \quad \big(X'X + \lambda[(1-\alpha)I]\big)^{-1},$$

which is guaranteed to exist and to have bounded eigenvalues. Although this introduces bias (the penalized estimator is no longer unbiased), the reduction in variance often more than compensates.

Variable selection under correlated predictors

From the correlation matrix, we could see that some variables are highly correlated with others, for example, $Corr(lcp, lcavol) = 0.68$ and $Corr(pgg45, gleason) = 0.75$, thus,

- **LASSO ($\alpha = 1$):** Drives many coefficients to zero (here only *lcavol*, *lweight*, and *svi* remain).

  When predictors are strongly correlated, LASSO tends to pick one and zero out others.
- **Ridge ($\alpha = 0$):** Never sets coefficients exactly to zero, all eight predictors stay in the model, albeit shrink to 0. Correlated covariates share similar shrinkage and are kept together.
- **Elastic Net ($\alpha = 0.4$):** Intermediate sparsity, it keeps six nonzero coefficients (*lcavol*, *lweight*, *lbph*, *svi*, *lcp*, *pgg45*). It is a mixed of ℓ_1/ℓ_2 penalty, which selects some groups of correlated variables rather than just keeping one of those correlated variables in lasso, but not include all correlated variables in the model. Also it shrink the coefficient to 0, like ridge regression.

Thus, if a very sparse model is desired and only a few covariates truly matter, LASSO is suitable, but may drop important members of a correlated block. If stability under multicollinearity is needed, Ridge is suitable (no drop-out, but no sparsity). Elastic Net offers a intermediate choice: it yields sparsity while preserving groups of correlated predictors.

(ii)

Using the CV–MSE vectors at the one-standard-error tuning parameter, we obtain

$$\begin{aligned}\mathrm{MSE}_{\mathrm{lasso},\,\lambda_{1\mathrm{se}}} &= 0.6379557,\\ \mathrm{MSE}_{\mathrm{ridge},\,\lambda_{1\mathrm{se}}} &= 0.6281617,\\ \mathrm{MSE}_{\mathrm{EN},\,\lambda_{1\mathrm{se}}} &= 0.5861445.\end{aligned}$$

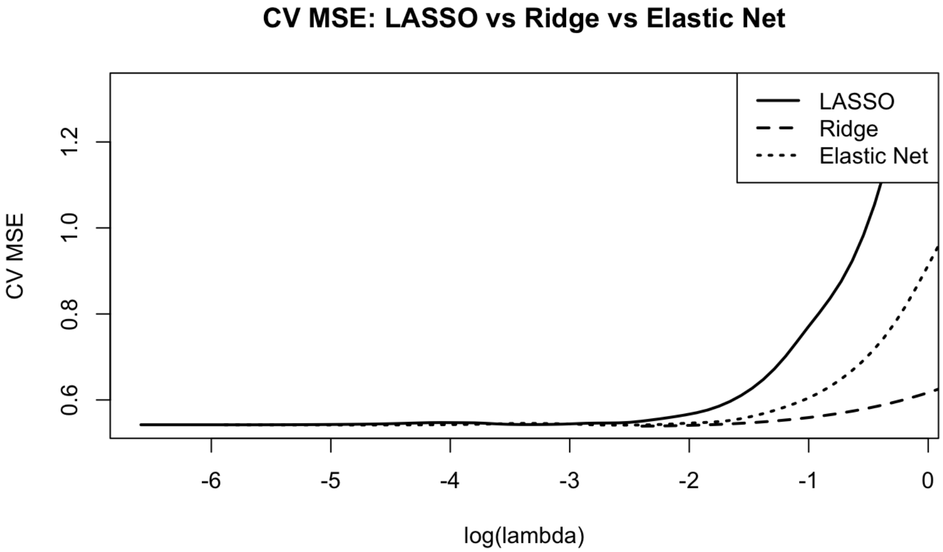

Figure 4: Cross-Validation MSE: LASSO vs Ridge vs Elastic Net

From Figure 4 and the numerical values above:

- **Elastic Net** achieves the lowest MSE (0.5861). By combining ℓ_1 and ℓ_2 penalties, it attains a balanced bias–variance trade-off and got lowest MSE, by only introducing some sparsity while still grouping correlated predictors.
- **Ridge** has MSE = 0.6282. Its ℓ_2-penalty shrinks all coefficients to 0, reducing variance at the cost of a small bias increase, therefore inducing an intermediate MSE.
- **LASSO** has the highest MSE (0.6379). Its ℓ_1-penalty drives many coefficients to zero, which lowers variance but can introduce more bias so perform worst on predictive accuracy in this setting.

However, in the region of small λ (light penalization), all three curves in Figure 4 lie close together—their MSE values differ only very slightly. This indicates that when the penalty is weak, LASSO, Ridge, and Elastic Net have similar predictive performance.

Generally speaking, all three CV–MSE curves in Figure 4 follow the same shape: as λ increases from very small values, the MSE initially remains flat, and then rises sharply under heavy penalization. Ridge regression has the smallest MSE among all three penalties (highest stability), LASSO has highest MSE (most of sparsity), and Elastic Net has the MSE stands in between, illustrating its mixture of variance reduction (Ridge) and coefficient selection (Lasso).

Code for Question 1

```
rm(list=ls())
set.seed(2025)

Prostate = read.table("/Users/yubin/Desktop/Multivariate Analysis/pcancer.dat", header=T)
library(MASS)
library(glmnet)

Y=as.numeric(Prostate[,9])
X=as.matrix(Prostate[,1:8])

# (a) LASSO regression (alpha = 1)

lassofit = glmnet(X,Y,alpha=1)

plot(lassofit,label=T, xvar="lambda")

cvfit=cv.glmnet(X,Y,alpha=1)

cvfit$lambda.min
cvfit$lambda.1se
coef(cvfit,s="lambda.1se")

# (b) Ridge regression (alpha = 0)
ridgefit <- glmnet(X, Y, alpha = 0)

plot(ridgefit, xvar = "lambda", label = TRUE)
title("Ridge Coefficient")

cvfit_ridge <- cv.glmnet(X, Y, alpha = 0)

cvfit_ridge$lambda.min
cvfit_ridge$lambda.1se
coef(cvfit_ridge,s="lambda.1se")

# (c) Elastic Net regression (alpha = 0.4)

enfit <- glmnet(X, Y, alpha = 0.4)

plot(enfit, xvar = "lambda", label = TRUE)
title("Elastic Net ( = 0.4) Coefficient Paths")

cv_en <- cv.glmnet(X, Y, alpha = 0.4)
cv_en$lambda.min
cv_en$lambda.1se
coef(cv_en,s="lambda.1se")

cor_mat <- cor(X)
print(round(cor_mat, 2))
```

```
mse_lasso  <- cvfit$cvm
lam_lasso  <- cvfit$lambda
mse_ridge  <- cvfit_ridge$cvm
lam_ridge  <- cvfit_ridge$lambda
mse_en     <- cv_en$cvm
lam_en     <- cv_en$lambda

idx_l_1se  <- which(lam_lasso == cvfit$lambda.1se)
idx_r_1se  <- which(lam_ridge == cvfit_ridge$lambda.1se)
idx_en_1se <- which(lam_en == cv_en$lambda.1se)

cat("LASSO MSE at lambda.1se =", mse_lasso[idx_l_1se], "\n\n")
cat("Ridge MSE at lambda.1se =", mse_ridge[idx_r_1se], "\n\n")
cat("EN MSE at lambda.1se =",     mse_en[idx_en_1se],     "\n\n")

plot(log(lam_lasso), mse_lasso, type = "l", lwd = 2,
     xlab = "log(lambda)", ylab = "CV MSE",
     main = "CV MSE: LASSO vs Ridge vs Elastic Net")
lines(log(lam_ridge), mse_ridge, lwd = 2, lty = 2)
lines(log(lam_en),     mse_en,     lwd = 2, lty = 3)
legend("topright",
       legend = c("LASSO", "Ridge", "Elastic Net"),
       lty    = c(1, 2, 3),
       lwd    = c(2, 2, 2))
```

Question 2

(a) Regression Tree Model

We randomly split the n observations of the Prostate data into a training set (two-thirds of the rows) and a test set (one-third), using simple random sampling.

On the training set we fit a full regression tree

$$\text{lpsa} \sim \text{lcavol} + \text{lweight} + \text{age} + \text{lbph} + \text{svi} + \text{lcp} + \text{gleason} + \text{pgg45},$$

and then perform 10-fold cross–validation (via cv.tree()) to compare subtrees of different sizes. The resulting plot of CV–deviance versus the tree sizes shows:

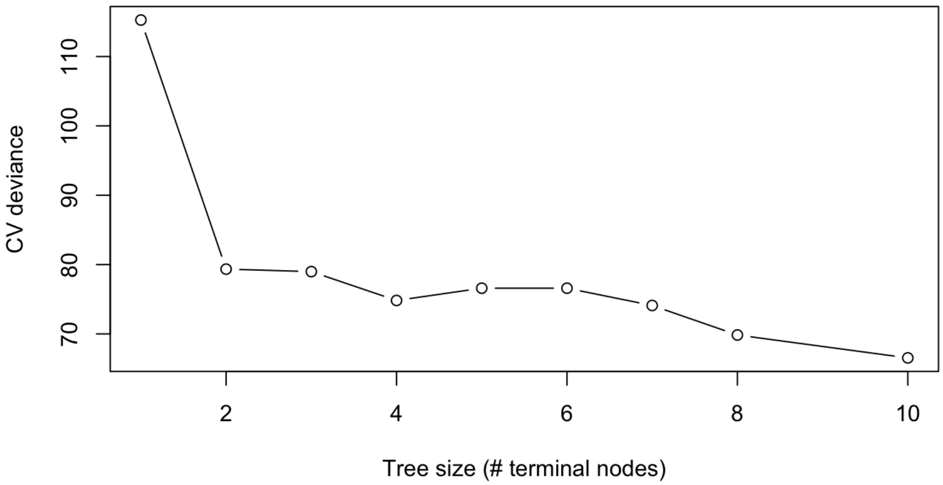

Figure 5: CV-deviance vs tree size

- A global minimum of CV–deviance at size = 10 leaves.
- An "elbow" at size = 4 leaves, beyond which additional splits yield only marginal improvement.

Model size decision

- **Minimum-deviance rule:** choose the 10-leaf subtree to minimize cross-validated squared error.
- **Elbow-rule (parsimony):** choose the 4-leaf subtree for a simple tree with greater interpretability, while accepting a small increase in CV–deviance.

Thus, I pruned the tree with tree size = 4 and 10 respectively:

Prune tree with size = 4

Under the 4-leaf model, the fitted predictor is a piecewise constant function of lcavol alone:

$$\widehat{\text{lpsa}} = \begin{cases} 0.55, & \text{lcavol} < -0.4786, \\ 2.00, & -0.4786 \le \text{lcavol} < 1.0508, \\ 2.70, & 1.0508 \le \text{lcavol} < 2.7917, \\ 4.10, & \text{lcavol} \ge 2.7917. \end{cases}$$

Plot as shown below:

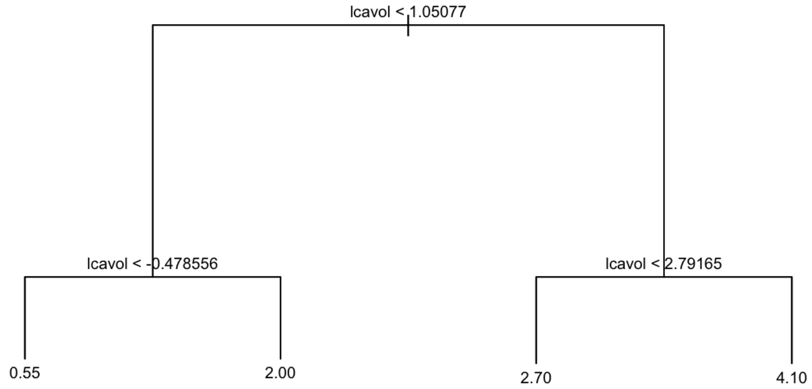

Figure 6: Pruned Tree (Size =4)

Prune tree with size = 10

By applying size = 10 to the full tree, we obtain a subtree with exactly ten terminal nodes (at which point the cross-validated deviance is minimized), and the decision rule is shown in the plot:

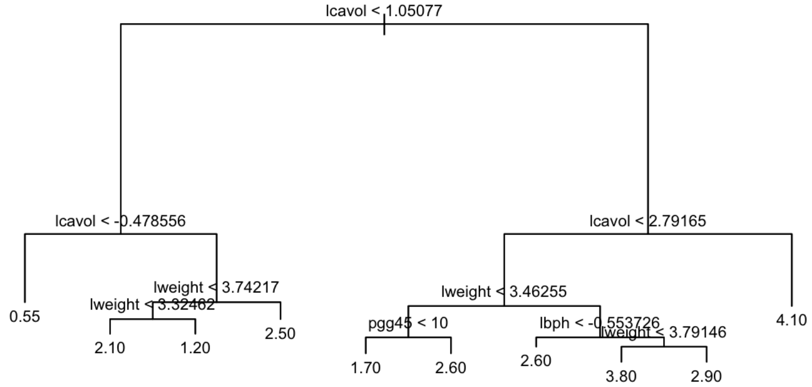

Figure 7: Pruned Tree (Size =10)

Test-set performance

Let $n_{\text{test}} = 32$ be the size of our held-out test set. Denote by $\hat{y}_i^{(4)}$ and $\hat{y}_i^{(10)}$ the predictions from the 4-leaf and 10-leaf trees, respectively, and by y_i the true lpsa. We then compute

$$\mathrm{MSE}_4 = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \big(\hat{y}_i^{(4)} - y_i\big)^2 = 0.6226361, \quad \mathrm{RSS}_4 = \sum_{i=1}^{n_{\text{test}}} \big(\hat{y}_i^{(4)} - y_i\big)^2 = 19.9244,$$

$$\mathrm{MSE}_{10} = \frac{1}{n_{\text{test}}} \sum_{i=1}^{n_{\text{test}}} \big(\hat{y}_i^{(10)} - y_i\big)^2 = 0.4058258, \quad \mathrm{RSS}_{10} = \sum_{i=1}^{n_{\text{test}}} \big(\hat{y}_i^{(10)} - y_i\big)^2 = 12.9864.$$

Thus, the 10-leaf tree attains both lower mean squared error and lower residual sum of squares on the test set, whereas the 4-leaf tree remains more parsimonious and interpretable. Thus, if one wished to optimize for predictive accuracy, the 10-leaf version would be selected; for a simpler, more interpretable rule, the 4-leaf model is preferred.

(b) Random Forest Model

We fit a Random Forest on the training set, manually tuning to find m that minimize test-set MSE, with ntrees set to 1000. We found

$$mtry = 7, ntree = 1000$$

gave the lowest test MSE:

$$\mathrm{MSE}_{\mathrm{RF,test}} = 0.1367894.$$

The summary for the final model is:

```
Type of random forest: regression
Number of trees: 1000
No. of variables tried at each split: 7
Mean of squared residuals (OOB MSE): 0.7345914
% Var explained: 48.88
```

Variable importance (IncNodePurity) is:

Variable	IncNodePurity
lcavol	50.6095
lweight	15.7670
age	4.6008
lbph	5.0622
svi	6.5236
lcp	3.6768
gleason	1.0857
pgg45	5.3252

Thus, the decision-rule parameters for the Random Forest are

$$\{\, mtry = 7,\ ntree = 1000 \,\},$$

which govern how many predictors are randomly sampled at each split and how many trees comprise the ensemble. The importance scores indicate that *lcavol* is the most influential predictor in reducing node impurity.

(c)

We compare the pruned regression trees (4-leaf and 10-leaf) to the Random Forest (mtry = 7, ntree = 1000) on the held-out test set ($n_{\text{test}} = 32$).

We use MSE and RSS as performance measures:

Mean Squared Error (MSE): $\frac{1}{n_{\text{test}}}\sum_i(\hat{y}_i - y_i)^2$

Residual Sum of Squares (RSS):$\sum_i(\hat{y}_i - y_i)^2$

$$\begin{aligned} \text{4-leaf tree:} \quad & \text{MSE}_4 = 0.6226, \quad \text{RSS}_4 = 19.9244, \\ \text{10-leaf tree:} \quad & \text{MSE}_{10} = 0.4058, \quad \text{RSS}_{10} = 12.9864, \\ \text{Random Forest:} \quad & \text{MSE}_{\text{RF}} = 0.1368, \quad \text{RSS}_{\text{RF}} = 4.3773, \end{aligned}$$

Which is better? The Random Forest achieves by far the lowest test-set MSE and RSS, indicating better predictive accuracy on the testing set. The single trees, while more interpretable, has higher MSE and RSS, which means that it suffers from poorer generalization ability on the test set (especially the 4-leaf tree).

Thus, in terms of out-of-sample prediction error (MSE/RSS), the Random Forest outperforms single tree. If interpretability is paramount, one might prefer a small pruned tree; for best accuracy, the random forest is superior.

Code for Question 2

Question (a):

```
library(MASS)
library(tree)
library(rpart)
library(rpart.plot)
library(randomForest)

set.seed(2025)
n <- nrow(Prostate)

test  <- sample(1:n, size = (n/3))
train <- setdiff(1:n, test)

tree_full <- tree(
  lpsa ~ lcavol + lweight + age + lbph + svi + lcp + gleason + pgg45,
  data   = Prostate,
  subset = train
)

plot(tree_full)
text(tree_full,cex=0.7,digits=2)

cv_res <- cv.tree(tree_full)
plot(cv_res$size, cv_res$dev, type="b",
     xlab="Tree size (# terminal nodes)",
     ylab="CV deviance")

tree_pruned <- prune.tree(tree_full, best = 4)
plot(tree_pruned)
text(tree_pruned,cex=0.7,digits=2)

tree_pruned_10 <- prune.tree(tree_full, best = 10)
plot(tree_pruned_10)
text(tree_pruned_10,cex=0.7,digits=2)

yhat=predict(tree_pruned,newdata=Prostate[-train,])
```

```
test=Prostate[-train,"lpsa"]

mean((yhat-test)^2)

yhat=predict(tree_pruned_10,newdata=Prostate[-train,])
test=Prostate[-train,"lpsa"]

mean((yhat-test)^2)
```

Question (b):

```
set.seed(2025)
rf_mod <- randomForest(
  lpsa ~ lcavol + lweight + age + lbph + svi + lcp + gleason + pgg45,
  data     = Prostate,
  subset   = train,
  mtry     = 7,
  ntree    = 1000,
)

yhat = predict(rf_mod,newdata=Prostate[-train,])

mean((yhat-test)^2)

print(rf_mod)

imp <- importance(rf_mod)
print(imp)
varImpPlot(rf_mod)
```

Time taken: 3 Hours.