OLLSCOIL NA hÉIREANN, CORCAIGH
THE NATIONAL UNIVERSITY OF IRELAND, CORK

COLÁISTE NA hOLLSCOILE, CORCAIGH
UNIVERSITY COLLEGE, CORK

ST4060 - Statistical Methods for Machine Learning I
ST6040 - Machine Learning and Statistical Analytics I
2024-25 Final Exam
ANSWER DOCUMENT

Instructions

- Please provide your answers **in this answer sheet**.
- **Only upload the pdf version of your answer document.**
- **Provide all answers hereafter. Paste relevant figures and R output within this document where appropriate.**

Answers to Question 1

(a)	RMSE may be used for Model 1 as it is simple linear line AUROC maybe used for Model 2 as it is curve RMSE may be used for Model3 as it is spline
(b)	Model 2 fits the data the best as it has no smoothing like model 3. Model 1 is just a linear line.
(c)	Model 2 yields the best fit as it has lowest RMSE and and is not smoothed to fit the data like Model 3
(d)	Yes Model 3 looks like it Model 2 but with penalty term added to it. The penalty term penalizes Model 3 for roughness to make it smoother as seen in the plots. Model 3 has a higher RMSE than Model 2 because the smoother it gets the less it ll fit the data better.
(e)	Model 2 training: B2 Model 2 testing: B4 Model 3 training: B1 Model 3 testing: B3 The variance for training is typically lower than testing so this would suggest B1 and B2 are training errors. B3 and B4 have larger variances in the error and this would suggest they are testing errors. Model 2 had lower RMSE than Model 3 in part (c) so this would suggest Model 2 would have the lower training and test error as seen above.

No R code required for Question 1

Answers to Question 2

(a)	Both RMSE: 272.3369
(b)	
(c)	Training error: 15.21442
(d)	
(e)	Lower variance for repeated k fold

R code for Question 2

```
## Q 2 ----------------------------------------------------------------------------------------------------
data(rock)
x = rock$peri
y = rock$perm

plot(x, y)

# (a) Fit two smoothing splines (i.e., P-splines) to the entire dataset (X, Y ). Set their
smoothing
# parameter spar to .8 and 0.95 respectively. Compute and quote the RMSE for both fits.
library(splines)
length(rock$perm)
length(rock$peri)
ssp1 = smooth.spline(x,y, spar=0.8)
y.ssp1 = approx(ssp1$x,ssp1$y,xout=x)$y
(rmse.ssp1 = sqrt( mean((y.ssp1-y)^2) ))



ssp2 = smooth.spline(x,y, spar=0.95)
y.ssp2 = approx(ssp2$x,ssp2$y,xout=x)$y
(rmse.ssp2 = sqrt( mean((y.ssp2-y)^2) ))
rmse.ssp2


# (b) Implement simple (i.e., not repeated, and not stratified) 10-fold cross-validation (CV)
# of the second smoothing spline (i.e. using spar=0.95). Set the random seed to 4060
# (set.seed(4060)) before running the analysis. Compute and quote the estimated training
# and test RMSEs for this model.
set.seed(4060)
# (a)
```

```
# Initialize variables for 10-fold cross-validation of a lasso model with fixed lambda = 10
# for the regularization param. Compute and store the training and test RMSE from the
# cross validated fits. Quote the mean estimates of training and test prediction errors.
n = nrow(x)  # Number of observations
K = 10  # Number of folds
folds = cut(1:n, K, labels = FALSE)  # Create folds
fit.rmse.lmo = p.rmse.lmo = numeric(K)  # To store training and test RMSE


for (k in 1:K) {
  # Create training and test sets
  i.train = which(folds != k)
  i.test = which(folds == k)
  x.train = as.matrix(x[i.train, ])
  y.train = y[i.train]
  x.test = x[i.test]
  y.test = y[i.test]

  # Train lmo model
  lmo = lm(y.train ~ x.train)
  # Compute training RMSE
  y.hat = predict(lmo, newx = x.train)
  length(y.hat)
  length(y.train)
  fit.rmse.lmo[k] = sqrt(mean((y.hat - y.train)^2))

  # Compute test RMSE
  # y.p = predict(lmo, newx = x.test)
  # length(y.p)
  # length(y.test)
  # p.rmse.lmo[k] = sqrt(mean((y.p - y.test)^2))
}

# Mean RMSEs
sqrt(mean(fit.rmse.lmo))

# mean_test_rmse_lasso = mean(p.rmse.lasso)

# (d)
sd(mean_fit.rmse.lmo)


data(rock)
x = rock$peri
y = rock$perm
set.seed(4060)
K = 10
```

```
folds = cut(1:n,K,labels=FALSE)
err = numeric(K)
for(k in 1:K){
  i.train = which(folds!=k)
  lm.fit = lm(y[i.train]~., data=x[i.train,])
  lm.pred = predict(lm.fit, newdata=x[-i.train,])
  err[k] = mean((lm.pred-y[-i.train])^2)
}
```

Answers to Question 3

(a)	Area = 0.09133379 Permi = -0.3440246

(b)	# b) Compute and quote the bootstrap estimate of the bias of the estimator of the regression
	# coefficient associated with variable shape in this model fit. Briefly indicate how you
	# calculated this value (you can paste R code to do this).
	# Bootstrap bias for the p-value: bar on top p* - p^0
	# statistic of the original sample:
	p0 = coefficients(summary(lm.fit))[4,4]
	
	# BS bias:
	mean(pb) - p0
	
	answer is 0

(c)	0.08307001
(d)	No increased MC repetitions to 1000 and 10000 and got same values for estimates in (a) which are same as
(e)	Rock is a small dataset with 48 values so bootstrapping should have lower variances of the distributions of training errors

R code for Question 3

```
## Q 3 ----------------------------------------------------------------------------------------------

lm.fit = lm(perm~., data=rock)
summary(lm.fit)

# (a) Implement a bootstrap analysis of the dataset, fitting a linear regression similar to the
# one above, using all variables in the data and 100 bootstrap resamples. Do not scale any
# of the variables. Do not use function boot for this question. Set the random seed to
# 4060 (set.seed(4060)) before running the analysis. Quote the bootstrap estimates of
# the effect of area and of peri on perm.

set.seed(4060)
B = 1000
dat = rock

# Initialize vectors to store bootstrap estimates
areaB <- numeric(B)
```

```
periB <- numeric(B)
pb = numeric(B)

# Bootstrap procedure
for (b in 1:B) {
  # Sample with replacement
  indices <- sample(1:nrow(rock), size = nrow(rock), replace = TRUE)
  dat_boot <- rock[indices, ]

  # Fit linear model on bootstrap sample
  model <- lm(perm~., data = rock)
  pb[b] = summary(model)$coefficients[4, 4]
  #boot_pvalues[b] <- summary(model)$coefficients[2, 4]

  # Store values
  areaB[b] <- coef(model)[2]
  periB[b] <- coef(model)[3]
}

# (a) Bootstrap estimates
mean(areaB)
mean(periB)

# b) Compute and quote the bootstrap estimate of the bias of the estimator of the
regression
# coefficient associated with variable shape in this model fit. Briefly indicate how you
# calculated this value (you can paste R code to do this).
# Bootstrap bias for the p-value: bar on top p* - p^0
# statistic of the original sample:
p0 = coefficients(summary(lm.fit))[4,4]

# BS bias:
mean(pb) - p0

# (c) Compute and quote the naive 95% bootstrap confidence interval around the standard
# error of the estimator of the regression coefficient associated with variable shape in this
# model fit. Briefly indicate how you calculated this value (you can paste R code to do
#                                          this).
pb
mean(pb) + c(1, -1) * 1.96*sd(pb)

# (d) Modify your bootstrap implementation to assess whether this model is prone to
overfitting.
# Is there evidence of overfitting? Briefly comment on your findings, using numerical output
# of your choice.
set.seed(4060)
B = 10000
```

```
dat = rock
# Initialize vectors to store bootstrap estimates
areaB <- numeric(B)
periB <- numeric(B)
pb = numeric(B)

# Bootstrap procedure
for (b in 1:B) {
  # Sample with replacement
  indices <- sample(1:nrow(rock), size = nrow(rock), replace = TRUE)
  dat_boot <- rock[indices, ]

  # Fit linear model on bootstrap sample
  model <- lm(perm~., data = rock)
  pb[b] = summary(model)$coefficients[4, 4]
  #boot_pvalues[b] <- summary(model)$coefficients[2, 4]

  # Store values
  areaB[b] <- coef(model)[2]
  periB[b] <- coef(model)[3]
}
mean(areaB)
mean(periB)


# e) Briefly explain what difference you would expect to see between the variances of the
# distributions of training errors obtained from this bootstrap analysis and from simple
# 5-fold cross-validation, and why. [5]
```

24/5

Answers to Question 4

(a)	First 3 explain more than 5% PC1 PC2 PC3 PC4 PC5 PC6 PC7 PC8 PC9 PC10 0.57602 0.26496 0.05972 0.02695 0.02223 0.02101 0.01329 0.00807 0.00537 0.00238
(b)	PC1 PC2 PC3 0.3305343 0.3168385 0.2611487
(c)	Fit Sum Squared Residuals: 147.4944 Fit1 adjusted R squares: 0.8066423 RMSE slightly higher for Z
(d)	PCA is distance based and there is binary data in the dataset like am and bs which are binary and not distance based.
(e)	The binary data and count data eg number of cyclinders drove the k means clustering

R code for Question 4

```
x = mtcars
x$mpg = NULL
y = mtcars$mpg

scaled_pca <- prcomp(x, scale = TRUE)

# Variance explained by each principal component
explained_variance <- summary(scaled_pca)$importance[2, ]


# (b)
z = scaled_pca$rotation
apply(z,2,sd)[1:3]

# (c)
fit1 = lm(y~., data=x)
fit2 = lm(y~., data=z)

# (e)
ko = kmeans(x, centers=3)
plot(x[,1:2], pch=20, cex=2, col=ko$cluster)
```