**ST4061 – Statistical Methods for Machine Learning II**

**ST6041 – Machine Learning and Statistical Analytics II**

2023-24

Continuous Assessment 2

**Answers to Question 1**

|  |  |
| --- | --- |
| **Question** | **Your answer** |
| **1** | Mean OOB RMSE for 0.001 shrinkage rate – 0.8414  Mean OOB RMSE for 0.05 shrinkage rate – 0.5053  Mean OOB RMSE for 0.01 shrinkage rate – 0.6195  Mean OOB RMSE for 0.1 shrinkage rate – 0.4933 |
| **2** |  |
| **3** |  |

**R code for Question 1**

## Question 1

rm(list=ls())

require(gbm)

require(ISLR)

df = na.omit(Hitters)

df$Salary = log(df$Salary)

rates = c(0.001, 0.05, 0.01, 0.1)

L = length(rates)

set.seed(4061)

n = nrow(df)

B = 100

OOB\_RMSEs = matrix(NA, nrow = B, ncol = L)

colnames(OOB\_RMSEs) = rates

for (i in 1:L) {

for (j in 1:B){

idxs = sample(1:n, n, replace=TRUE)

X\_train = df[idxs,]

X\_test = df[-idxs,]

Y\_test = df[-idxs,]$Salary

gbm\_tree = gbm(Salary~., data=X\_train, distribution = 'gaussian', shrinkage = rates[i])

test\_preds = predict(gbm\_tree, X\_test)

OOB\_RMSEs[j, i] = sqrt(mean((test\_preds - Y\_test)^2))

}

}

head(OOB\_RMSEs)

## Question 1(1)

OOB\_RMSEs\_mean = apply(OOB\_RMSEs, 2, mean)

round(OOB\_RMSEs\_mean, 4)

## Question 1(2)

par(mfrow=c(1,1))

boxplot(OOB\_RMSEs, main="Boxplot of OOB-RMSEs across shrinkage rates",xlab="Shrinkage-values",ylab="Bootstrap RMSE-estimates", col = 'cyan')

**Answers to Question 2**

|  |  |
| --- | --- |
| **Question** | **Your answer** |
| **1** | “Sex” is a categorical feature with 2 levels (“Female”, “Male”).  Grade is also a categorical column but it’s not a feature here but the outcome (labels). |
| **2** | There are more number of Males compared to the number of females in the data set by |
| **3** |  |
| **4** |  |
| **5** | p-value for Wilcoxon test between Age and Sex – **0.7259**  p-value for Wilcoxon test between Age and X.Max – **0.07627**  p-value for Wilcoxon test between Age and X.Mean – **0.07691** |
| **6** | Mean of Age after Min-Max Scaling – **0.4465**  Mean of X.Mean after Min-Max Scaling – **0.2477**  Mean of X.Max after Min-Max Scaling – **0.1655** |
| **7** | Overall Error quoted by neuralnet function used - **7.543743** (seed set as 4061) |
| **8** | (i) Overall Accuracy – **0.92208 (92.208%)**  (ii) Class-wise Sensitivity for this fit –     |  |  |  | | --- | --- | --- | | **“high”** | **“int”** | **“low”** | | 0.9733  (97.33%) | 0.963  (96.3%) | 0.68  (68%) | |
| **9** | **26** columns identified by the correlation filter to remove with 0.95 as cutoff –  ['compactness1', 'compactness2', 'sphericity', 'l.major', 'major.axis.length', 'l.minor', 'minor.axis.length', 'l.least', 'least.axis.length', 'RMS', 'mean\_HIST', 'sum.avg\_GLCM', 'auto.corr\_GLCM', 'var\_HIST', 'joint.var\_GLCM', 'sum.var\_GLCM', 'joint.max\_GLCM', 'energy\_GLCM', 'entropy\_GLCM', 'homogeneity\_GLCM', 'inv.diff.mom\_GLCM', 'diff.entropy\_GLCM', 'dissimilarity\_GLCM', 'homogeneity.norm\_GLCM', 'contrast\_GLCM', 'inv.diff.mom.norm\_GLCM'] |

**R code for Question 2**

## Question 2

require(caret)

require(neuralnet)

require(DataExplorer)

df = read.csv(file="uws.csv", stringsAsFactors=TRUE)

subdf = df[,c("grade","sex","age","x.mean","x.max")]

y = df$grade

x = df

x$grade = NULL

str(df)

## Question 2(1)

## sex is a categorical feature

## Question 2(2)

plot\_bar(df)

## Question 2(3)

plot\_boxplot(subdf, by = 'grade')

## Question 2(4)

plot\_boxplot(subdf, by = 'sex')

## Question 2(5)

age = subdf$age

sex = subdf$sex

x\_max = subdf$x.max

x\_mean = subdf$x.mean

wilcox.test(age ~ sex, alternative = "two.sided")

wilcox.test(x\_max ~ sex, alternative = "two.sided")

wilcox.test(x\_mean ~ sex, alternative = "two.sided")

## Question 2(6)

conversion <- function(x){

# function recoding levels into numerical values

if(is.factor(x)){

levels(x)

return(as.numeric(x))

} else {

return(x)

}

}

scaling <- function(x){

# function applying normalization to [0,1] scale

mins = min(x,na.rm=TRUE)

maxs = max(x,na.rm=TRUE)

return((x-mins)/(maxs-mins))

}

df\_inter = data.frame(lapply(df,conversion))

df\_scaled = data.frame(lapply(df\_inter,scaling))

means = apply(df\_scaled[, c("age", "x.mean", "x.max")], 2, mean)

round(means, 4)

df\_scaled$grade = NULL

## Question 2(7)

set.seed(4061)

mod = neuralnet(y~., data = df\_scaled, hidden=c(5), linear.output = FALSE)

(error = mod$result.matrix["error",])

## Question 2(8)

col\_names = colnames(mod$response)

final\_preds = as.factor(col\_names[max.col(predict(mod, df\_scaled))])

cf\_mat = caret::confusionMatrix(final\_preds, y)

(overall\_accuracy = cf\_mat$overall[1])

specificity\_class = cbind(cf\_mat$byClass[1],cf\_mat$byClass[2],cf\_mat$byClass[3])

colnames(specificity\_class) = c("high","int","low")

round(specificity\_class, 4)

## Question 2(9)

x$sex = as.numeric(x$sex)

correlation\_matrix <- cor(x)

cols = colnames(x)

pairs = c()

for (i in cols) {

for (j in cols) {

if (i!=j & abs(correlation\_matrix[i,j]) > 0.95) {

pairs = c(pairs, c(i,j))

}

}

}

cat("Columns to remove : ", unique(pairs))

unique(pairs)