CSCI E-106 : Spring 2025 Final Exam

Ian Kelk

Contents


```
# Ensure required packages are installed, then load them
options(repos = c(CRAN = "https://cloud.r-project.org"))

req_pkgs <- c("tidyverse", "caret", "ISLR2", "MASS", "rpart",
              "rpart.plot", "neuralnet", "lmtest", "car", "dplyr")

to_install <- setdiff(req_pkgs, rownames(installed.packages()))
if (length(to_install)) {
  install.packages(to_install, dependencies = TRUE)
}

# Load quietly
invisible(lapply(req_pkgs, function(p) {
  suppressPackageStartupMessages(library(p, character.only = TRUE))
}))
```

Instructions

1-) As always, you are required to follow Harvard University academic integrity and honor code.

2-) Open book and open notes exam (textbooks (print or pdf), lecture slides, notes, practice exam, homework solutions, and TA slides, including all Rmd's*).

3-) You are allowed to use RStudio Desktop or RStudio Cloud (https://rstudio.cloud.) , Microsoft Word, Power Point and PDF reader, and canvas on your laptop.

4-) Proctorio is required to start this exam. If you are prompted for an access code, you must Install and Configure Proctorio on your machine.

5-) Review the Proctorio Support for Test Takers page, Online Exam Checklist and Proctorio Checklist to help you avoid common errors and who to contact if you run into any issues.

6-) Practice Setup Quiz is available under Quizzes to test your connection and to find out if you are ok with the Proctorio.

7-) Please read the list of recording and restrictions provided by Proctorio carefully before taking the exam.

8-) The final exam will be available from Monday, May 12th at 12 pm EST through Tuesday, May 13th at 12:00 pm EST. You must submit your midterm by Tuesday, May 13th at 12:00 pm EST.

9-) Once you start the exam, you must complete it in 3 hours or by Tuesday, May 13th, at 12:00 p.m. EST.

10-) In order to receive full credit, please provide full explanations and calculations for each questions.

11-) Make sure that you are familiar with the procedures for troubleshooting exam issues. Preview the document Download the document and follow the protocol if there are any issues!

12-) Make sure you submit both .Rmd and (knitted) pdf or html files.

13-) You need to have a camera on your laptop.

14-) Please reach out to DCE Online Support by using the information below
DCE Online Support
(617) 998-8571
(Mon-Thurs 10am-11pm, Fri-Sun 10am-8pm EST)
AcademicTechnology@dce.harvard.edu

15-) Our emails:
hakangogtas@yahoo.com
rafael_gomeztagle@g.harvard.edu
andreahatch10@gmail.com
sezer@yahoo.com
srg3924@gmail.com

Problem 1

Refer to the Hitters data set. Major League Baseball Data from the 1986 and 1987 seasons. The data set can be downloaded from the attached CSV file or directly from ISL2 library in R by copying and pasting the following command into R console:library(ISLR2);data("Hitters"). (55 Points)

Description of the data is below and there are 322 observations and 20 variables, including 3 categorical variables (League, Division and NewLeague).

AtBat=Number of times at bat in 1986

Hits=Number of hits in 1986

HmRun=Number of home runs in 1986

Runs=Number of runs in 1986

RBI=Number of runs batted in in 1986

Walks=Number of walks in 1986

Years=Number of years in the major leagues

CAtBat=Number of times at bat during his career

CHits=Number of hits during his career

CHmRun=Number of home runs during his career

CRuns=Number of runs during his career

CRBI=Number of runs batted in during his career

CWalks=Number of walks during his career

League=A factor with levels A and N indicating player's league at the end of 1986

Division=A factor with levels E and W indicating player's division at the end of 1986

PutOuts=Number of put outs in 1986

Assists=Number of assists in 1986

Errors=Number of errors in 1986

Salary=1987 annual salary on opening day in thousands of dollars

NewLeague=A factor with levels A and N indicating player's league at the beginning of 1987

a-) Remove the missing data by using na.omit comment (e.g. Hitters<-na.omit(Hitters)). Create dummy variables for the categorical variables League, Division and NewLeague. Drop the categorical variables after creating the dummy variables

from the data set. Perform initial data analyses to identify outliers, missing data, and variables that show high correlation with the salary and with each other. Document your findings (10 points)

b-) Create train and test data sets: select a random sample of 70% observations from the data set for the train data set and remaining cases for the test data set. (use set.seed(994) before selecting the sample and running Neuron Network and Regression Tree) (5 points)

c-) Use stepwise (both ways) model selection to select the best model for predicting the Salary on the train data set. **Ensure that all variables are significant, use** $\alpha = 0.05$. Justify your choice of model. Check all regression model assumptions visually with appropriate graphs and conduct the Breusch-Pagan Test to determine whether or not the error variances are constant. Evaluate the performance of Regression model on the train and test data sets. Please discuss your findings (10 points)

d-) Use Regression tree to predict the Salary on the train data set and evaluate the performance on the train and test data sets. (10 points)

e-) Use Neuron Network (NN) with four hidden layers with 9,7,5, and 3 neurons respectively (hidden=c(9,7,5,3)) with softplus activation function (softplus <- function(x) { log(1 + exp(x)) }) to predict the Salary on the train data set. Evaluate the performance of NN on the train and test data sets.(15 points)

f-) Evaluate the performances of models built in part c through part e on both the train and test data sets. Compare the R^2 and select the best model. (5 points)

Problem 2

Refer to the attached file for the SENIC data set.The data set can be downloaded from the attached csv file. (45 points)

Description of the data is below.
Length.of.stay = Average length of stay of all patients in hospital (in days)
Age =Average age of patients (in years)
Infection.risk=Average estimated probability of acquiring infection in hospital (in percent)
Routine.culturing.ratio =Ratio of number of cultures performed to number of patients without signs or symptoms of hospital-acquired infection, times 100
Routine.chest.X.ray.ratio =Ratio of number of X-rays performed to number of patients, without signs or symptoms of pneumonia, times 100
Number.of.beds=Average number of beds in hospital during study period
Medical.school.affiliation =Medical school affiliation, where 1=Yes, 2=No
Region=Geographic region, where: 1 =NE, 2=NC, 3=S, 4=W
Average.daily.census =Average number of patients in hospital per day during study period
Number.of.nurses=Average number of full-time equivalent registered and licensed practical nurses during study period (number full-time plus one half the number part time)
Available.facilities.and.service=Percent of 35 potential facilities and services that are provided by the hospital

a-) Medical school affiliation (Y) is the response variable to be coded Y = 1 If Medical school affiliation and Y = 0 if no Medical school affiliation. The pool of potential predictor variables are all variables except Geographic region. Exclude Geographic region from your data set. (5 points)

b-) Use a step function to build a logistic regression model to predict the Medical affiliation on the full data set by using all variables (except Geographic region). **Ensure that all variables are significant, use** $\alpha = 0.05$. Conduct the Hosmer-Lemeshow goodness of fit test for the appropriateness of the logistic regression function by forming four groups. State the alternatives, decision rule, and conclusion. (10 points)

c-) Suppose that a subject is classified as Medical school affiliation if $p \geq 0.50$ and no Medical school affiliation if $p < 0.50$. Compute the confusion matrix and comment on model performance. (5 points)

d-) Use a decision tree to predict the Medical school affiliation and calculate the confusion matrix to evaluate the model performance. (10 points)

e-) Use Neuron Network (NN) with five hidden layers with 8,6,5,3, and 2 neurons respectively (hidden=c(8,6,5,3,2)) with softplus activation function (softplus <- function(x) { log(1 + exp(x)) }) to predict the Medical school affiliation.Suppose that a subject is classified as Medical school affiliation if $p \geq 0.50$ and no Medical school affiliation if $p < 0.50$. Compute the confusion matrix and comment on model performance.(10 points)

f-) which model would you choose? (5 points)

Problem 1: Hitters data (55 points)

We use the Hitters dataset from `ISLR2` (or an equivalent CSV if preferred), remove missing values, and build several prediction models for Salary.

(a) Remove missing data, create dummy variables, and perform initial EDA

```
# Load data and remove rows with missing Salary
data("Hitters")
hitters <- na.omit(Hitters)

# Inspect structure
str(hitters)
```

```
## 'data.frame':    263 obs. of  20 variables:
##  $ AtBat    : int  315 479 496 321 594 185 298 323 401 574 ...
##  $ Hits     : int  81 130 141 87 169 37 73 81 92 159 ...
##  $ HmRun    : int  7 18 20 10 4 1 0 6 17 21 ...
##  $ Runs     : int  24 66 65 39 74 23 24 26 49 107 ...
##  $ RBI      : int  38 72 78 42 51 8 24 32 66 75 ...
##  $ Walks    : int  39 76 37 30 35 21 7 8 65 59 ...
##  $ Years    : int  14 3 11 2 11 2 3 2 13 10 ...
##  $ CAtBat   : int  3449 1624 5628 396 4408 214 509 341 5206 4631 ...
##  $ CHits    : int  835 457 1575 101 1133 42 108 86 1332 1300 ...
##  $ CHmRun   : int  69 63 225 12 19 1 0 6 253 90 ...
##  $ CRuns    : int  321 224 828 48 501 30 41 32 784 702 ...
##  $ CRBI     : int  414 266 838 46 336 9 37 34 890 504 ...
##  $ CWalks   : int  375 263 354 33 194 24 12 8 866 488 ...
##  $ League   : Factor w/ 2 levels "A","N": 2 1 2 2 1 2 1 2 1 1 ...
##  $ Division : Factor w/ 2 levels "E","W": 2 2 1 1 2 1 2 2 1 1 ...
##  $ PutOuts  : int  632 880 200 805 282 76 121 143 0 238 ...
##  $ Assists  : int  43 82 11 40 421 127 283 290 0 445 ...
##  $ Errors   : int  10 14 3 4 25 7 9 19 0 22 ...
##  $ Salary   : num  475 480 500 91.5 750 ...
##  $ NewLeague: Factor w/ 2 levels "A","N": 2 1 2 2 1 1 1 2 1 1 ...
##  - attr(*, "na.action")= 'omit' Named int [1:59] 1 16 19 23 31 33 37 39 40 42 ...
##   ..- attr(*, "names")= chr [1:59] "-Andy Allanson" "-Billy Beane" "-Bruce Bochte" "-Bob Boone" ...
```

```
summary(hitters$Salary)
```

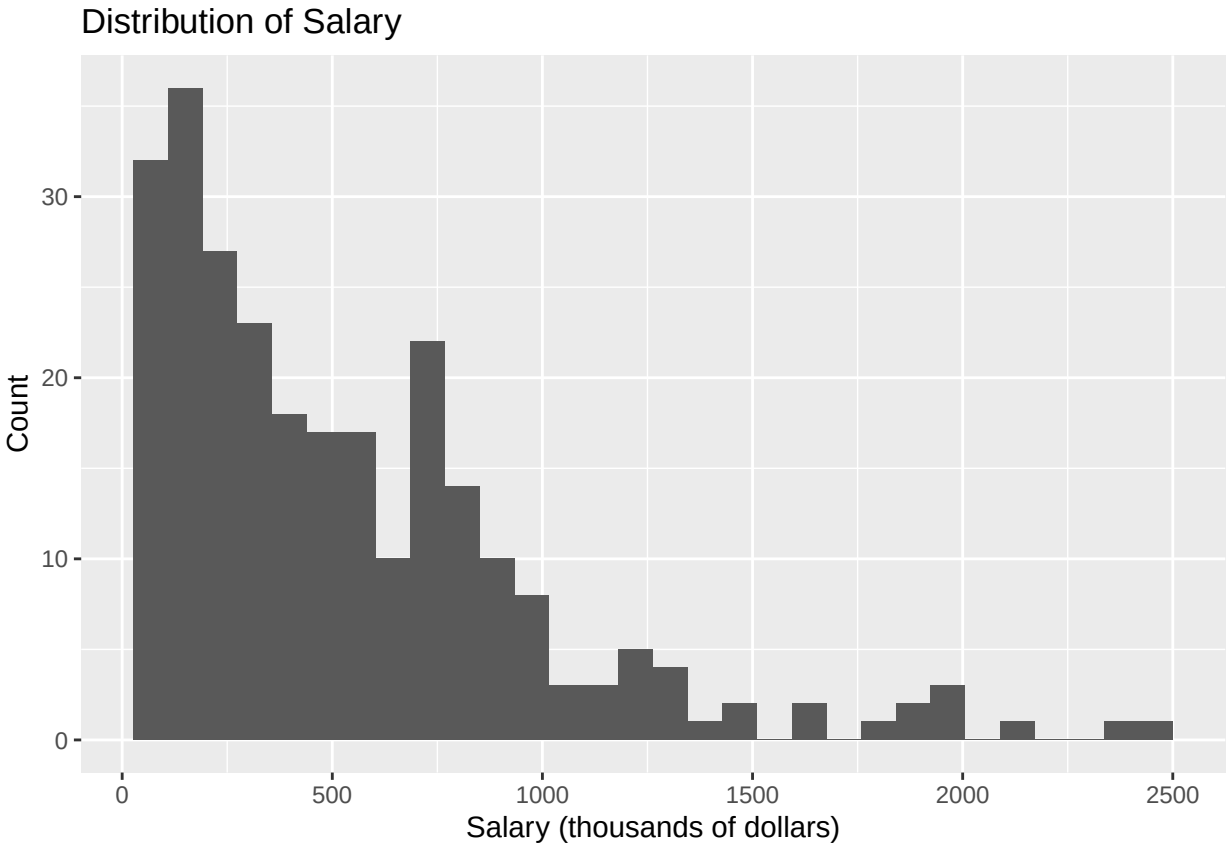
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    67.5   190.0   425.0   535.9   750.0  2460.0
```

```
# Create dummy variables for the categorical predictors
dummies <- dummyVars(Salary ~ ., data = hitters)
x_all   <- predict(dummies, newdata = hitters)    # numeric predictors only

hitters_num <- data.frame(Salary = hitters$Salary, x_all)

glimpse(hitters_num)
```

```
## Rows: 263
## Columns: 23
## $ Salary      <dbl> 475.000, 480.000, 500.000, 91.500, 750.000, 70.000, 100.00~
## $ AtBat       <dbl> 315, 479, 496, 321, 594, 185, 298, 323, 401, 574, 202, 418~
## $ Hits        <dbl> 81, 130, 141, 87, 169, 37, 73, 81, 92, 159, 53, 113, 60, 4~
## $ HmRun       <dbl> 7, 18, 20, 10, 4, 1, 0, 6, 17, 21, 4, 13, 0, 7, 20, 2, 8, ~
## $ Runs        <dbl> 24, 66, 65, 39, 74, 23, 24, 26, 49, 107, 31, 48, 30, 29, 8~
## $ RBI         <dbl> 38, 72, 78, 42, 51, 8, 24, 32, 66, 75, 26, 61, 11, 27, 75,~
## $ Walks       <dbl> 39, 76, 37, 30, 35, 21, 7, 8, 65, 59, 27, 47, 22, 30, 73, ~
## $ Years       <dbl> 14, 3, 11, 2, 11, 2, 3, 2, 13, 10, 9, 4, 6, 13, 15, 5, 8, ~
## $ CAtBat      <dbl> 3449, 1624, 5628, 396, 4408, 214, 509, 341, 5206, 4631, 18~
```

```
## $ CHits       <dbl> 835, 457, 1575, 101, 1133, 42, 108, 86, 1332, 1300, 467, 3~
## $ CHmRun      <dbl> 69, 63, 225, 12, 19, 1, 0, 6, 253, 90, 15, 41, 4, 36, 177,~
## $ CRuns       <dbl> 321, 224, 828, 48, 501, 30, 41, 32, 784, 702, 192, 205, 30~
## $ CRBI        <dbl> 414, 266, 838, 46, 336, 9, 37, 34, 890, 504, 186, 204, 103~
## $ CWalks      <dbl> 375, 263, 354, 33, 194, 24, 12, 8, 866, 488, 161, 203, 207~
## $ League.A    <dbl> 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0~
## $ League.N    <dbl> 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1~
## $ Division.E  <dbl> 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0~
## $ Division.W  <dbl> 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1~
## $ PutOuts     <dbl> 632, 880, 200, 805, 282, 76, 121, 143, 0, 238, 304, 211, 1~
## $ Assists     <dbl> 43, 82, 11, 40, 421, 127, 283, 290, 0, 445, 45, 11, 151, 4~
## $ Errors      <dbl> 10, 14, 3, 4, 25, 7, 9, 19, 0, 22, 11, 7, 6, 8, 10, 16, 2,~
## $ NewLeague.A <dbl> 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0~
## $ NewLeague.N <dbl> 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1~
```

```r
# Distribution of Salary
ggplot(hitters_num, aes(x = Salary)) +
  geom_histogram(bins = 30) +
  labs(title = "Distribution of Salary",
       x = "Salary (thousands of dollars)",
       y = "Count")
```




```r
# Correlations among the first few variables (including Salary)
hitters_cor <- cor(hitters_num)
hitters_cor[1:8, 1:8]
```

```
##            Salary     AtBat       Hits     HmRun        Runs       RBI     Walks
## Salary 1.0000000 0.3947709 0.43867474 0.3430281  0.41985856 0.4494571 0.4438673
## AtBat  0.3947709 1.0000000 0.96396913 0.5551022  0.89982910 0.7960154 0.6244481
## Hits   0.4386747 0.9639691 1.00000000 0.5306274  0.91063014 0.7884782 0.5873105
## HmRun  0.3430281 0.5551022 0.53062736 1.0000000  0.63107588 0.8491074 0.4404537
## Runs   0.4198586 0.8998291 0.91063014 0.6310759  1.00000000 0.7786924 0.6970151
## RBI    0.4494571 0.7960154 0.78847819 0.8491074  0.77869235 1.0000000 0.5695048
## Walks  0.4438673 0.6244481 0.58731051 0.4404537  0.69701510 0.5695048 1.0000000
```

```
## Years  0.4006570 0.0127255 0.01859809 0.1134884 -0.01197495 0.1296679 0.1347927
##              Years
## Salary  0.40065699
## AtBat   0.01272550
## Hits    0.01859809
## HmRun   0.11348842
## Runs   -0.01197495
## RBI     0.12966795
## Walks   0.13479270
## Years   1.00000000
```

```
# Correlation of each predictor with Salary
salary_cor <- sort(hitters_cor[, "Salary"], decreasing = TRUE)
salary_cor
```

```
##       Salary         CRBI        CRuns        CHits       CAtBat       CHmRun
##  1.000000000  0.566965686  0.562677711  0.548909559  0.526135310  0.524930560
##       CWalks          RBI        Walks         Hits         Runs        Years
##  0.489822036  0.449457088  0.443867260  0.438674738  0.419858559  0.400656994
##        AtBat        HmRun      PutOuts   Division.E      Assists     League.A
##  0.394770945  0.343028078  0.300480356  0.192514399  0.025436136  0.014281827
##  NewLeague.A  NewLeague.N       Errors     League.N   Division.W
##  0.002834460 -0.002834460 -0.005400702 -0.014281827 -0.192514399
```

For the exam:

Based on the plots and correlation output above, describe the distribution of Salary, note any outliers, and identify which predictors appear most strongly associated with Salary. Also comment on any strong correlations among predictors (possible multicollinearity).

Conclusion: The Salary distribution is strongly right-skewed. Most players earn between about \$200k and \$800k, with a median salary of \$425k and a mean around \$536k, but there are a few very highly paid players with salaries up to about \$2.46M, creating a long right tail and clear high-salary outliers. After using `na.omit()`, we are left with 263 complete cases, so there are no remaining missing values in the modeling dataset.

From the correlation output, Salary is most strongly associated with career-total batting statistics: `CRBI`, `CRuns`, `CHits`, `CAtBat`, `CHmRun`, and `CWalks` all have moderate to strong positive correlations with Salary. Among the 1986 season variables, `RBI`, `Walks`, `Hits`, and `Runs` also show noticeable positive correlations with Salary. Many of the batting predictors are also highly correlated with each other—for example, `AtBat` and `Hits` have a correlation above 0.96, and both are highly correlated with `Runs` and `RBI`—indicating substantial multicollinearity in the raw predictors.

(b) Create train and test sets and check representativeness

```
# Stratified 70/30 split based on the outcome Salary
set.seed(994)

train_index <- caret::createDataPartition(
  y    = hitters_num$Salary,
  p    = 0.7,
  list = FALSE
)

hitters_train <- hitters_num[train_index, ] |> as.data.frame()
hitters_test  <- hitters_num[-train_index, ] |> as.data.frame()

# Training/test sizes and shares (should be approx 0.70 / 0.30)
train_test_sizes <- tibble(
  sample = c("Training", "Test"),
  n      = c(nrow(hitters_train), nrow(hitters_test))
) |>
  mutate(share = n / nrow(hitters_num))
```

```r
knitr::kable(
  train_test_sizes,
  digits  = 3,
  caption = "Training and test sample sizes and shares for the Hitters salary data"
)
```

Table 1: Training and test sample sizes and shares for the Hitters salary data

sample	n	share
Training	185	0.703
Test	78	0.297

```r
# Outcome distribution (Salary) overall vs training vs test
Salary_summary <- bind_rows(
  overall = hitters_num,
  train   = hitters_train,
  test    = hitters_test,
  .id     = "sample"
) |>
  group_by(sample) |>
  summarise(
    n            = n(),
    mean_Salary  = mean(Salary),
    sd_Salary    = sd(Salary),
    min_Salary   = min(Salary),
    max_Salary   = max(Salary),
    .groups      = "drop"
  )

knitr::kable(
  Salary_summary,
  digits  = 3,
  caption = "Distribution of Salary for the overall Hitters data and the training/test splits"
)
```

Table 2: Distribution of Salary for the overall Hitters data and the training/test splits

sample	n	mean_Salary	sd_Salary	min_Salary	max_Salary
overall	263	535.926	451.119	67.5	2460.0
test	78	515.659	452.396	70.0	2460.0
train	185	544.471	451.535	67.5	2412.5

For the exam:

Comment on whether the train and test sets appear reasonably similar (representative of the full dataset) based on the summary statistics.

Conclusion: The stratified split produces 185 training observations (about 70.3%) and 78 test observations (about 29.7%), which is very close to the intended 70/30 ratio. The Salary summaries for overall, training, and test sets are also very similar: the overall mean is about \$536k with SD ≈ \$451k, while the training set has a mean of about \$544k (SD ≈ \$452k) and the test set a mean of about \$516k (SD ≈ \$452k). The minimum and maximum salaries in each subset are also close to those for the full dataset. These similarities suggest that the training and test sets are both reasonably representative of the overall data with respect to Salary, and the stratified sampling worked as intended.

(c) Stepwise linear regression for Salary and model diagnostics

```r
# Full and null linear models on the training data
full_lm <- lm(Salary ~ ., data = hitters_train)
```

```r
null_lm <- lm(Salary ~ 1, data = hitters_train)

# Stepwise selection (both directions)
step_lm <- step(null_lm,
                scope = list(lower = null_lm, upper = full_lm),
                direction = "both",
                trace = FALSE)

summary(step_lm)
```

```
##
## Call:
## lm(formula = Salary ~ CRBI + Hits + Division.E + AtBat + Walks +
##     PutOuts + CWalks + CRuns, data = hitters_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -765.44 -191.21  -45.93  151.81 1982.01
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  52.41460   84.66937   0.619 0.536684
## CRBI          0.45510    0.23441   1.941 0.053802 .
## Hits          7.05171    2.07676   3.396 0.000847 ***
## Division.E  110.77010   50.61855   2.188 0.029963 *
## AtBat        -2.12993    0.65643  -3.245 0.001407 **
## Walks         7.09331    2.03252   3.490 0.000611 ***
## PutOuts       0.17222    0.09871   1.745 0.082785 .
## CWalks       -0.87681    0.31170  -2.813 0.005467 **
## CRuns         0.76198    0.28979   2.629 0.009311 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 337.9 on 176 degrees of freedom
## Multiple R-squared:  0.4643, Adjusted R-squared:   0.44
## F-statistic: 19.07 on 8 and 176 DF,  p-value: < 2.2e-16
```

We see that both `PutOuts` and `CRBI` are not significant, so we remove the one with the largest p-value first.

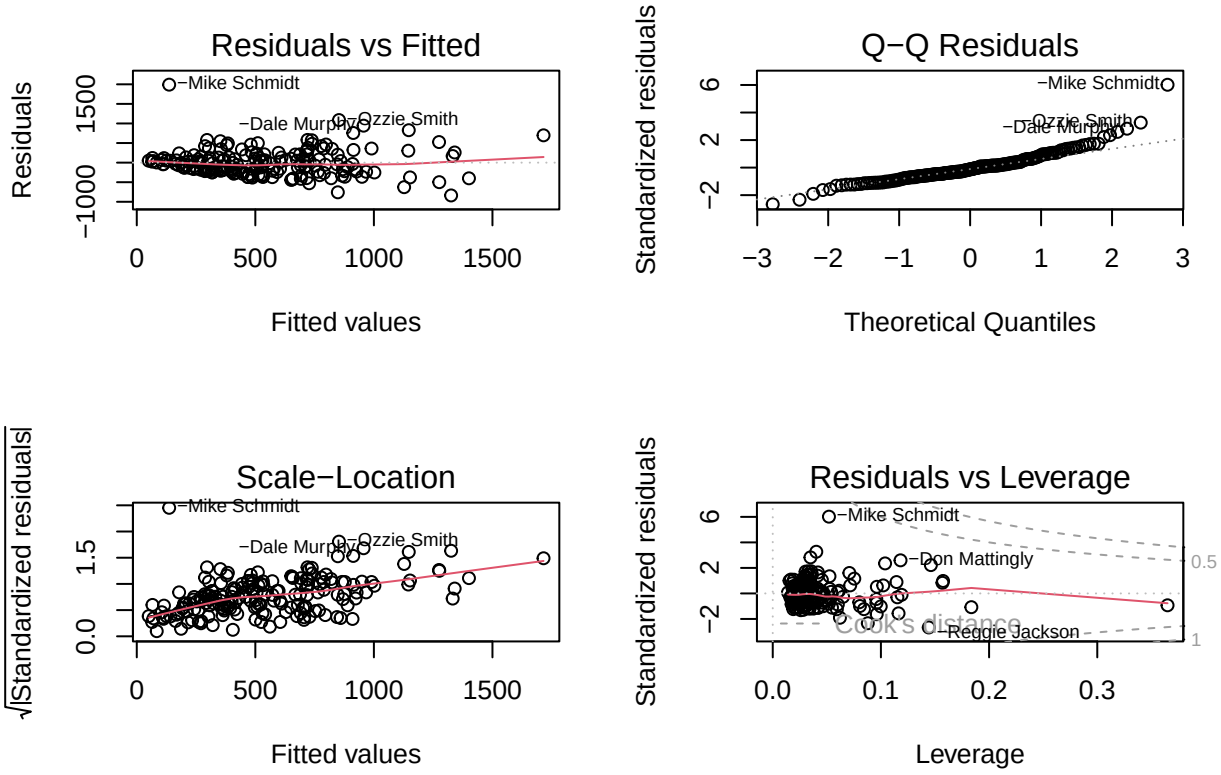
```r
# Remove the least significant predictor `PutOuts` and refit
step_lm <- update(step_lm, . ~ . - PutOuts)
summary(step_lm)
```

```
##
## Call:
## lm(formula = Salary ~ CRBI + Hits + Division.E + AtBat + Walks +
##     CWalks + CRuns, data = hitters_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -838.21 -202.32  -43.01  129.48 1990.66
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  54.0633    85.1515   0.635 0.526309
## CRBI          0.5031     0.2341   2.149 0.033001 *
## Hits          6.9242     2.0874   3.317 0.001104 **
## Division.E  111.5011    50.9082   2.190 0.029814 *
## AtBat        -2.0075     0.6564  -3.058 0.002572 **
## Walks         7.3367     2.0394   3.597 0.000417 ***
## CWalks       -0.8921     0.3134  -2.847 0.004938 **
```

```
## CRuns         0.7296     0.2909   2.508 0.013024 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 339.9 on 177 degrees of freedom
## Multiple R-squared:  0.455,  Adjusted R-squared:  0.4335
## F-statistic: 21.11 on 7 and 177 DF,  p-value: < 2.2e-16
```

Now all predictors are significant.

```
# Diagnostic plots for the selected linear model
par(mfrow = c(2, 2))
plot(step_lm)
```




```
par(mfrow = c(1, 1))

# Formal checks (optional)
bptest(step_lm)                       # Breusch-Pagan test for heteroskedasticity
```

```
##
##  studentized Breusch-Pagan test
##
## data:  step_lm
## BP = 6.0782, df = 7, p-value = 0.5306
```

```
shapiro.test(residuals(step_lm))    # Shapiro-Wilk test for normality (large n caveat)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(step_lm)
## W = 0.91012, p-value = 3.438e-09
```

```
vif(step_lm)                          # Variance inflation factors
```

```
##       CRBI       Hits Division.E      AtBat      Walks     CWalks      CRuns
##   9.026199  14.960817   1.037500  15.515155   2.954244  12.329156  15.818367
```

```r
# Helper for R^2
rsq <- function(y, yhat) {
  1 - sum((y - yhat)^2) / sum((y - mean(y))^2)
}

# Train and test predictions
train_pred_lm <- predict(step_lm, newdata = hitters_train)
test_pred_lm  <- predict(step_lm, newdata = hitters_test)

lm_rsq_train <- rsq(hitters_train$Salary, train_pred_lm)
lm_rsq_test  <- rsq(hitters_test$Salary,  test_pred_lm)

lm_rsq_train
```

```
## [1] 0.4550479
```

```r
lm_rsq_test
```

```
## [1] 0.6042287
```

For the exam:

Summarize which predictors were selected in the stepwise linear regression and interpret the signs of a few key coefficients. Comment on the residual plots and tests: do they suggest any violations of the usual linear model assumptions? Finally, briefly compare train vs test R^2 and discuss whether the model appears to generalize well.

Conclusion: After the AIC-based stepwise selection and a final manual step removing `PutOuts`, the chosen linear model for Salary includes predictors `CRBI`, `Hits`, `Division.E`, `AtBat`, `Walks`, `CWalks`, and `CRuns`. The signs of the coefficients are mostly as expected: `CRBI`, `Hits`, `Walks`, and `CRuns` have positive coefficients, indicating that, holding other variables fixed, players with more hits, walks, and career production tend to have higher salaries. The positive coefficient on `Division.E` suggests that, on average, players in the Eastern division earn about \$110k more than comparable players in the Western division. In contrast, `AtBat` and `CWalks` have negative coefficients once the other predictors are controlled for, which is likely a consequence of multicollinearity among the many overlapping batting and career-total variables rather than a truly negative effect of additional at-bats or career walks.

In this final model all slope coefficients are statistically significant at $\alpha = 0.05$. The residual plots show a reasonably random scatter of residuals around zero with a few large outliers corresponding to superstar players, but no strong systematic pattern. The Breusch–Pagan test has p ≈ 0.53, so we do not find evidence of heteroskedasticity and the constant-variance assumption appears reasonable. The Shapiro–Wilk test strongly rejects normality, and the Q–Q plot shows heavier tails, but given the sample size and the presence of extreme salaries, this mild non-normality is not surprising and linear regression is usually robust to it. The model explains about 45.5% of the variance in Salary on the training data, and the test R^2 is higher at about 0.60, suggesting that the model generalizes well and is not overfitting.

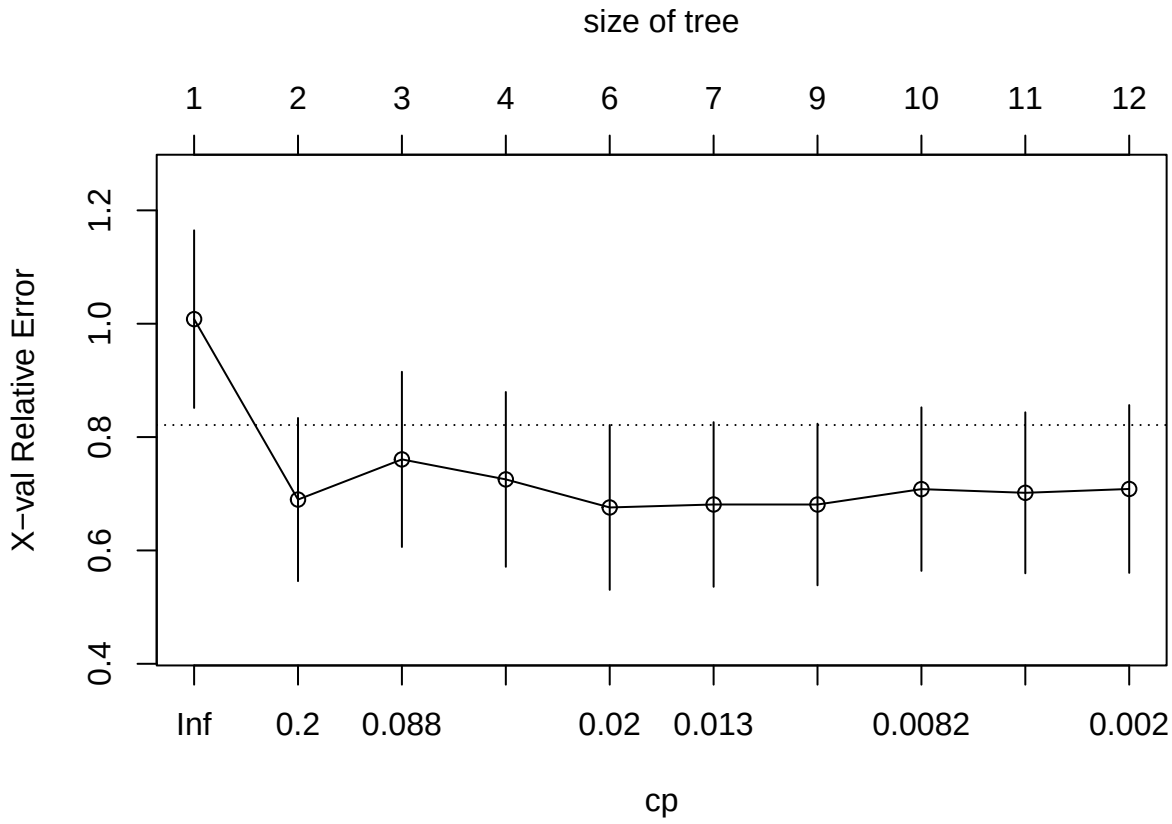
(d) Regression tree for Salary

```r
set.seed(994)

tree_model <- rpart(Salary ~ ., data = hitters_train,
                    method = "anova",
                    control = rpart.control(cp = 0.001))

printcp(tree_model)
```

```
##
## Regression tree:
## rpart(formula = Salary ~ ., data = hitters_train, method = "anova",
##     control = rpart.control(cp = 0.001))
##
## Variables actually used in tree construction:
## [1] Assists AtBat   CAtBat  CHits   CHmRun  Hits    Walks   Years
##
## Root node error: 37514591/185 = 202782
```

```
##
## n= 185
##
##           CP nsplit rel error  xerror    xstd
## 1  0.3649717      0   1.00000 1.00806 0.15668
## 2  0.1068931      1   0.63503 0.68974 0.14394
## 3  0.0723865      2   0.52814 0.76062 0.15454
## 4  0.0265303      3   0.45575 0.72532 0.15416
## 5  0.0155112      5   0.40269 0.67584 0.14541
## 6  0.0116430      6   0.38718 0.68106 0.14523
## 7  0.0101899      8   0.36389 0.68101 0.14252
## 8  0.0066031      9   0.35370 0.70816 0.14421
## 9  0.0041686     10   0.34710 0.70165 0.14200
## 10 0.0010000     11   0.34293 0.70844 0.14802
```

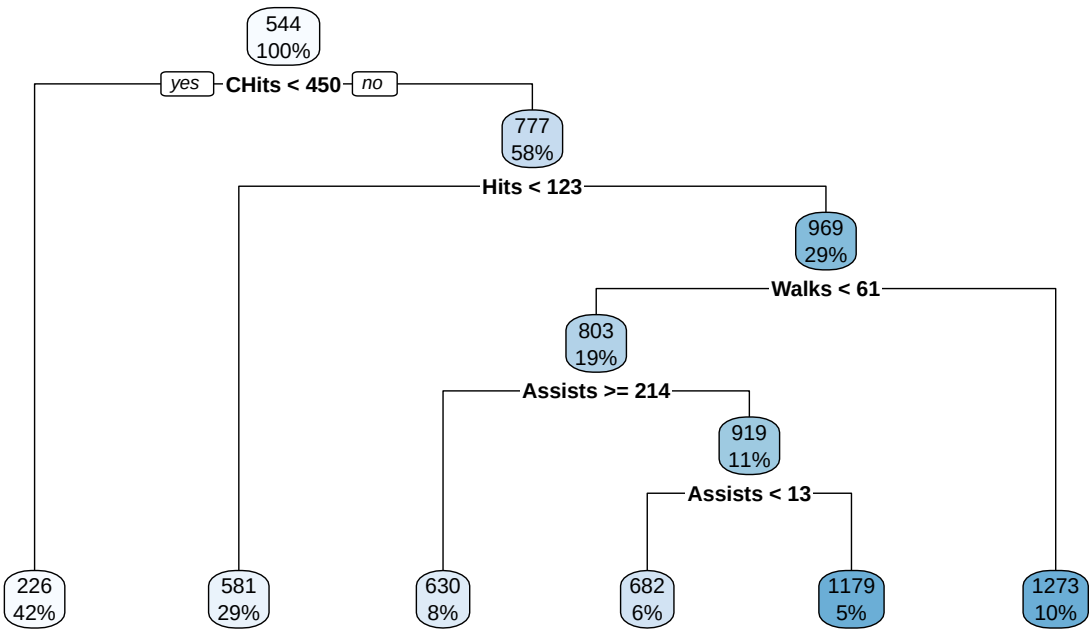
```r
plotcp(tree_model)
```




```r
# Choose cp with minimum cross-validated error
opt_cp <- tree_model$cptable[which.min(tree_model$cptable[, "xerror"]), "CP"]
opt_cp
```

```
## [1] 0.01551124
```

```r
pruned_tree <- prune(tree_model, cp = opt_cp)

rpart.plot(pruned_tree,
           main = "Regression tree for Salary")
```




```
tree_pred_train <- predict(pruned_tree, newdata = hitters_train)
tree_pred_test  <- predict(pruned_tree, newdata = hitters_test)

tree_rsq_train <- rsq(hitters_train$Salary, tree_pred_train)
tree_rsq_test  <- rsq(hitters_test$Salary,  tree_pred_test)

tree_rsq_train
```

```
## [1] 0.597312
```

```
tree_rsq_test
```

```
## [1] 0.4356787
```

For the exam:

Describe the basic structure of the pruned tree (for example, which variables are used near the top). Comment on the train and test R^2 values and how they compare to the stepwise linear regression.

Conclusion: The pruned regression tree for Salary has `CHits` at the root, splitting players into groups with fewer than about 450 career hits versus those with more. Below this, the tree uses variables such as `Hits`, `Walks`, `Assists`, `AtBat`, and some career totals to form further splits: among lower-career-hit players, low recent hits and walks lead to low-salary terminal nodes, while higher recent performance or more assists lead to higher-salary nodes. The tree emphasizes both career production (`CHits`, `CAtBat`, `CHmRun`) and 1986 performance (`Hits`, `Walks`, `Assists`) near the top of the structure.

In terms of accuracy, the regression tree attains a training R^2 of about 0.60 but a test R^2 of about 0.44. Compared with the stepwise linear regression (training $R^2 \approx 0.46$, test $R^2 \approx 0.60$), the tree fits the training data more closely but generalizes less well: its test performance is noticeably worse than that of the linear model, suggesting more overfitting.

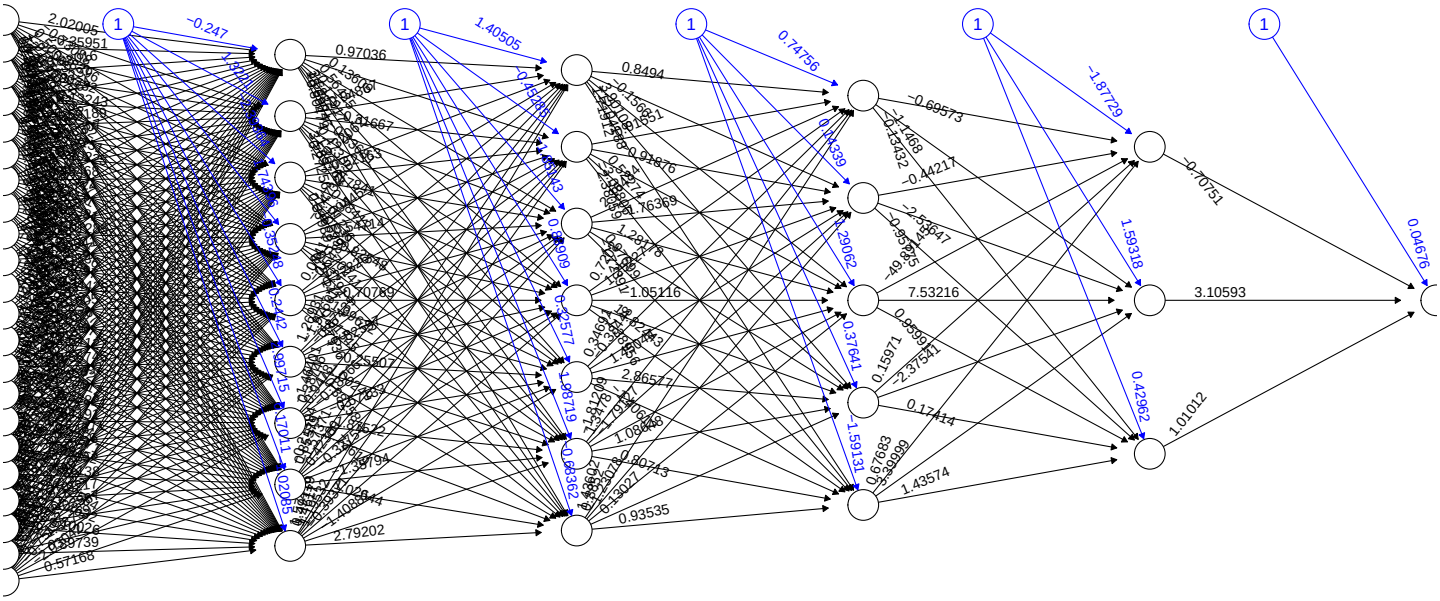
(e) Neural network with four hidden layers (9, 7, 5, 3)

```
# Separate response and predictors
y_train <- hitters_train$Salary
y_test  <- hitters_test$Salary

x_train <- hitters_train[, setdiff(names(hitters_train), "Salary")]
x_test  <- hitters_test[,  setdiff(names(hitters_test),  "Salary")]

# Min-max scaling for predictors (based on training data)
```

```r
x_range <- apply(x_train, 2, range)
x_min   <- x_range[1, ]
x_max   <- x_range[2, ]

scale_predictors <- function(df, min_vec, max_vec) {
  as.data.frame(scale(df, center = min_vec, scale = max_vec - min_vec))
}

x_train_scaled <- scale_predictors(x_train, x_min, x_max)
x_test_scaled  <- scale_predictors(x_test,  x_min, x_max)

# ALSO scale Salary to [0, 1] for neuralnet stability
y_min <- min(y_train)
y_max <- max(y_train)

y_train_scaled <- (y_train - y_min) / (y_max - y_min)
y_test_scaled  <- (y_test  - y_min) / (y_max - y_min)

# Data frames for neuralnet
nn_train_df <- data.frame(Salary_scaled = y_train_scaled, x_train_scaled)
nn_test_df  <- data.frame(Salary_scaled = y_test_scaled,  x_test_scaled)

nn_formula <- as.formula(
  paste("Salary_scaled ~", paste(colnames(x_train_scaled), collapse = " + "))
)

nn_formula
```

```
## Salary_scaled ~ AtBat + Hits + HmRun + Runs + RBI + Walks + Years +
##     CAtBat + CHits + CHmRun + CRuns + CRBI + CWalks + League.A +
##     League.N + Division.E + Division.W + PutOuts + Assists +
##     Errors + NewLeague.A + NewLeague.N
```

```r
# Softplus activation for neuralnet
softplus <- function(x) log1p(exp(x))

set.seed(994)

nn_model <- neuralnet(
  nn_formula,
  data          = nn_train_df,
  hidden        = c(9, 7, 5, 3),  # as per exam instructions
  act.fct       = softplus,       # use softplus activation
  linear.output = TRUE,           # regression: linear output layer
  threshold     = 0.01            # explicit, default is 0.01
)

plot(nn_model, rep = "best")
```



```
# Predictions on the scaled Salary scale
nn_train_pred_scaled <- compute(
  nn_model,
  nn_train_df[, colnames(x_train_scaled)]
)$net.result

nn_test_pred_scaled <- compute(
  nn_model,
  nn_test_df[, colnames(x_test_scaled)]
)$net.result

nn_train_pred_scaled <- as.vector(nn_train_pred_scaled)
nn_test_pred_scaled  <- as.vector(nn_test_pred_scaled)

# Back-transform to original Salary units
nn_train_pred <- y_min + nn_train_pred_scaled * (y_max - y_min)
nn_test_pred  <- y_min + nn_test_pred_scaled  * (y_max - y_min)

# R^2 on the original Salary scale
nn_rsq_train <- rsq(y_train, nn_train_pred)
nn_rsq_test  <- rsq(y_test,  nn_test_pred)

nn_rsq_train
```

```
## [1] 0.9870936
```

```
nn_rsq_test
```

```
## [1] -12.21558
```

For the exam:

Comment on the neural network's train and test R^2 relative to the linear model and regression tree. Does the neural network appear to overfit, underfit, or perform similarly?

Conclusion: The neural network with four hidden layers (9, 7, 5, and 3 neurons) and softplus activation achieves an extremely high training R^2 of about 0.99, indicating that it fits the training data almost perfectly. However, its test R^2 is about -12.22, which is catastrophically bad: the model performs far worse on the test set than simply predicting the mean Salary for every player (which would yield $R^2 = 0$). This combination of near-perfect training fit and highly negative test R^2 is a clear sign of extreme overfitting. Compared with the linear regression and regression tree, the neural network dramatically overfits and does not generalize at all in this configuration.

(f) Compare all models and select the best one

```
model_perf <- tibble(
  Model    = c("Stepwise linear regression",
               "Regression tree",
               "Neural network (9,7,5,3)"),
  R2_train = c(lm_rsq_train, tree_rsq_train, nn_rsq_train),
  R2_test  = c(lm_rsq_test,  tree_rsq_test,  nn_rsq_test)
)

model_perf
```

```
## # A tibble: 3 x 3
##   Model                       R2_train R2_test
##   <chr>                          <dbl>   <dbl>
## 1 Stepwise linear regression     0.455   0.604
## 2 Regression tree                0.597   0.436
## 3 Neural network (9,7,5,3)       0.987 -12.2
```

For the exam:

Using primarily the test R^2 values (and any other diagnostics you find relevant), choose the best model for predicting Salary and explain your choice in a few sentences. Mention both predictive performance and interpretability.

Conclusion: Comparing models, the stepwise linear regression has training and test R^2 of about 0.46 and 0.60, respectively, the regression tree has R^2 of about 0.60 on training but only about 0.44 on test, and the neural network has $R^2 \approx 0.99$ on training but about -12.22 on test. The stepwise linear model therefore provides the best out-of-sample predictive performance, with solid test R^2 and no obvious signs of heavy overfitting. It also has the advantage of interpretability: the direction and relative size of each coefficient can be interpreted in terms of how batting statistics and division relate to salary. The regression tree is less accurate on the test set, and the neural network is clearly overfitting. For these reasons, the stepwise linear regression is the preferred model for predicting Salary.

Problem 2: SENIC data (45 points)

We use the SENIC dataset from the provided CSV `SENIC.csv`. The response is Medical school affiliation, and we compare logistic regression, a decision tree, and a neural network.

(a) Define the response and exclude Region

```
senic <- read.csv("SENIC.csv")

# Check variable names
names(senic)
```

```
##  [1] "Length.of.stay"                    "Age"
##  [3] "Infection.risk"                    "Routine.culturing.ratio"
##  [5] "Routine.chest.X.ray.ratio"         "Number.of.beds"
##  [7] "Medical.school.affiliation"        "Region"
##  [9] "Average.daily.census"              "Number.of.nurses"
## [11] "Available.facilities.and.services"
```

```
str(senic)
```

```
## 'data.frame':    113 obs. of  11 variables:
##  $ Length.of.stay                  : num  7.13 8.82 8.34 8.95 11.2 ...
##  $ Age                             : num  55.7 58.2 56.9 53.7 56.5 50.9 57.8 45.7 48.2 56.3 ...
##  $ Infection.risk                  : num  4.1 1.6 2.7 5.6 5.7 5.1 4.6 5.4 4.3 6.3 ...
##  $ Routine.culturing.ratio         : num  9 3.8 8.1 18.9 34.5 21.9 16.7 60.5 24.4 29.6 ...
##  $ Routine.chest.X.ray.ratio       : num  39.6 51.7 74 122.8 88.9 ...
##  $ Number.of.beds                  : int  279 80 107 147 180 150 186 640 182 85 ...
##  $ Medical.school.affiliation      : int  2 2 2 2 2 2 2 1 2 2 ...
```

```
##  $ Region                          : int  4 2 3 4 1 2 3 2 3 1 ...
##  $ Average.daily.census            : int  207 51 82 53 134 147 151 399 130 59 ...
##  $ Number.of.nurses                : int  241 52 54 148 151 106 129 360 118 66 ...
##  $ Available.facilities.and.services: num  60 40 20 40 40 40 40 60 40 40 ...
```

```
# Convert to R-friendly names (already done by read.csv via check.names = TRUE)
# For clarity, show them:
names(senic)
```

```
##  [1] "Length.of.stay"                   "Age"
##  [3] "Infection.risk"                   "Routine.culturing.ratio"
##  [5] "Routine.chest.X.ray.ratio"        "Number.of.beds"
##  [7] "Medical.school.affiliation"       "Region"
##  [9] "Average.daily.census"             "Number.of.nurses"
## [11] "Available.facilities.and.services"
```

```
# Convert response and Region to factors
senic <- senic %>%
  mutate(
    Medical.school.affiliation = factor(Medical.school.affiliation,
                                        levels = c(2, 1),
                                        labels = c("No", "Yes")),
    Region = factor(Region)
  )

# Exclude Region as instructed
senic_glm <- senic %>%
  dplyr::select(-Region)

str(senic_glm)
```

```
## 'data.frame':    113 obs. of  10 variables:
##  $ Length.of.stay                   : num  7.13 8.82 8.34 8.95 11.2 ...
##  $ Age                              : num  55.7 58.2 56.9 53.7 56.5 50.9 57.8 45.7 48.2 56.3 ...
##  $ Infection.risk                   : num  4.1 1.6 2.7 5.6 5.7 5.1 4.6 5.4 4.3 6.3 ...
##  $ Routine.culturing.ratio          : num  9 3.8 8.1 18.9 34.5 21.9 16.7 60.5 24.4 29.6 ...
##  $ Routine.chest.X.ray.ratio        : num  39.6 51.7 74 122.8 88.9 ...
##  $ Number.of.beds                   : int  279 80 107 147 180 150 186 640 182 85 ...
##  $ Medical.school.affiliation       : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 2 1 1 ...
##  $ Average.daily.census             : int  207 51 82 53 134 147 151 399 130 59 ...
##  $ Number.of.nurses                 : int  241 52 54 148 151 106 129 360 118 66 ...
##  $ Available.facilities.and.services: num  60 40 20 40 40 40 40 60 40 40 ...
```

```
summary(senic_glm$Medical.school.affiliation)
```

```
##  No Yes
##  96  17
```

For the exam:

Briefly describe the distribution of Medical school affiliation (how many Yes vs No) and summarize the ranges or means of a few key predictors.

Conclusion: After recoding `Medical.school.affiliation` so that "Yes" corresponds to 1 and "No" to 0 and dropping `Region`, the dataset contains 113 hospitals, of which 96 have no medical school affiliation and 17 do. Thus only about 15% of hospitals are affiliated with a medical school, so the response is quite imbalanced. The predictors include measures of patient mix (`Length.of.stay`, `Age`, `Infection.risk`), diagnostic activity (`Routine.culturing.ratio`, `Routine.chest.X.ray.ratio`), hospital size and utilization (`Number.of.beds`, `Average.daily.census`), staffing (`Number.of.nurses`), and available facilities (`Available.facilities.and.services`). These variables vary substantially across hospitals, for example, the number of beds and average daily census span a wide range, indicating a mix of small and large hospitals with different resource levels that could plausibly relate to medical school affiliation.

(b) Stepwise logistic regression and LR test

```
# Full and null logistic regression models
full_logit <- glm(Medical.school.affiliation ~ .,
                  data   = senic_glm,
                  family = binomial)

null_logit <- glm(Medical.school.affiliation ~ 1,
                  data   = senic_glm,
                  family = binomial)

# Stepwise selection (both directions)
step_logit <- step(null_logit,
                   scope = list(lower = null_logit, upper = full_logit),
                   direction = "both",
                   trace = FALSE)

summary(step_logit)
```

```
##
## Call:
## glm(formula = Medical.school.affiliation ~ Average.daily.census +
##     Routine.culturing.ratio + Available.facilities.and.services +
##     Number.of.beds + Age, family = binomial, data = senic_glm)
##
## Coefficients:
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                     -6.81053    6.77130  -1.006   0.3145
## Average.daily.census             0.07350    0.02928   2.510   0.0121 *
## Routine.culturing.ratio          0.12125    0.05097   2.379   0.0174 *
## Available.facilities.and.services  0.29629    0.12132   2.442   0.0146 *
## Number.of.beds                  -0.04961    0.02051  -2.419   0.0155 *
## Age                             -0.31228    0.14280  -2.187   0.0287 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 95.706  on 112  degrees of freedom
## Residual deviance: 31.812  on 107  degrees of freedom
## AIC: 43.812
##
## Number of Fisher Scoring iterations: 9
```

```
# Likelihood ratio test comparing step model to null
lr_test <- anova(null_logit, step_logit, test = "Chisq")
lr_test
```

```
## Analysis of Deviance Table
##
## Model 1: Medical.school.affiliation ~ 1
## Model 2: Medical.school.affiliation ~ Average.daily.census + Routine.culturing.ratio +
##     Available.facilities.and.services + Number.of.beds + Age
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1       112     95.706
## 2       107     31.812  5   63.893 1.901e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Text answer components:

- Hypotheses for the LR test:

  - H_0: All slope coefficients in the logistic regression are zero (no relationship between the predictors and Medical school affiliation).
  - H_a: At least one slope coefficient is non-zero (at least one predictor is associated with Medical school affiliation).
- Decision rule (for $\alpha = 0.05$): Reject H_0 if the LR test p-value is less than 0.05.

For the exam:

Use the LR test output above to report the test statistic and p-value, state whether you reject H_0 at the 5% level, and interpret the conclusion in the context of predicting Medical school affiliation.

Conclusion: The likelihood ratio test compares the intercept-only model to the stepwise logistic regression model containing `Average.daily.census`, `Routine.culturing.ratio`, `Available.facilities.and.services`, `Number.of.beds`, and `Age`. The test statistic is 63.893 on 5 degrees of freedom, with a p-value of about $1.9 * 10^{-12}$. Since this p-value is far below 0.05, we reject H_0 at the 5% significance level. In context, this means that including these predictors greatly improves the fit compared with a model with no predictors, and at least one of these variables is strongly associated with whether a hospital has a medical school affiliation. All of the predictors in the final stepwise model are individually significant at $\alpha = 0.05$ as well.

(c) Confusion matrix for the logistic regression model

```
# Predicted probabilities and classes using threshold 0.5
logit_prob  <- predict(step_logit, type = "response")
logit_class <- ifelse(logit_prob >= 0.5, "Yes", "No") %>%
  factor(levels = c("No", "Yes"))

# Confusion matrix
logit_cm <- confusionMatrix(logit_class,
                            senic_glm$Medical.school.affiliation,
                            positive = "Yes")
logit_cm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction No Yes
##        No  91   5
##        Yes  5  12
##
##                Accuracy : 0.9115
##                  95% CI : (0.8433, 0.9567)
##     No Information Rate : 0.8496
##     P-Value [Acc > NIR] : 0.03711
##
##                   Kappa : 0.6538
##
##  Mcnemar's Test P-Value : 1.00000
##
##             Sensitivity : 0.7059
##             Specificity : 0.9479
##          Pos Pred Value : 0.7059
##          Neg Pred Value : 0.9479
##              Prevalence : 0.1504
##          Detection Rate : 0.1062
##    Detection Prevalence : 0.1504
##       Balanced Accuracy : 0.8269
##
##        'Positive' Class : Yes
##
```

For the exam:

From the confusion matrix, report the overall accuracy, sensitivity (true positive rate), and specificity (true negative rate). Comment on how well the logistic regression model classifies hospitals with and without a medical school affiliation.

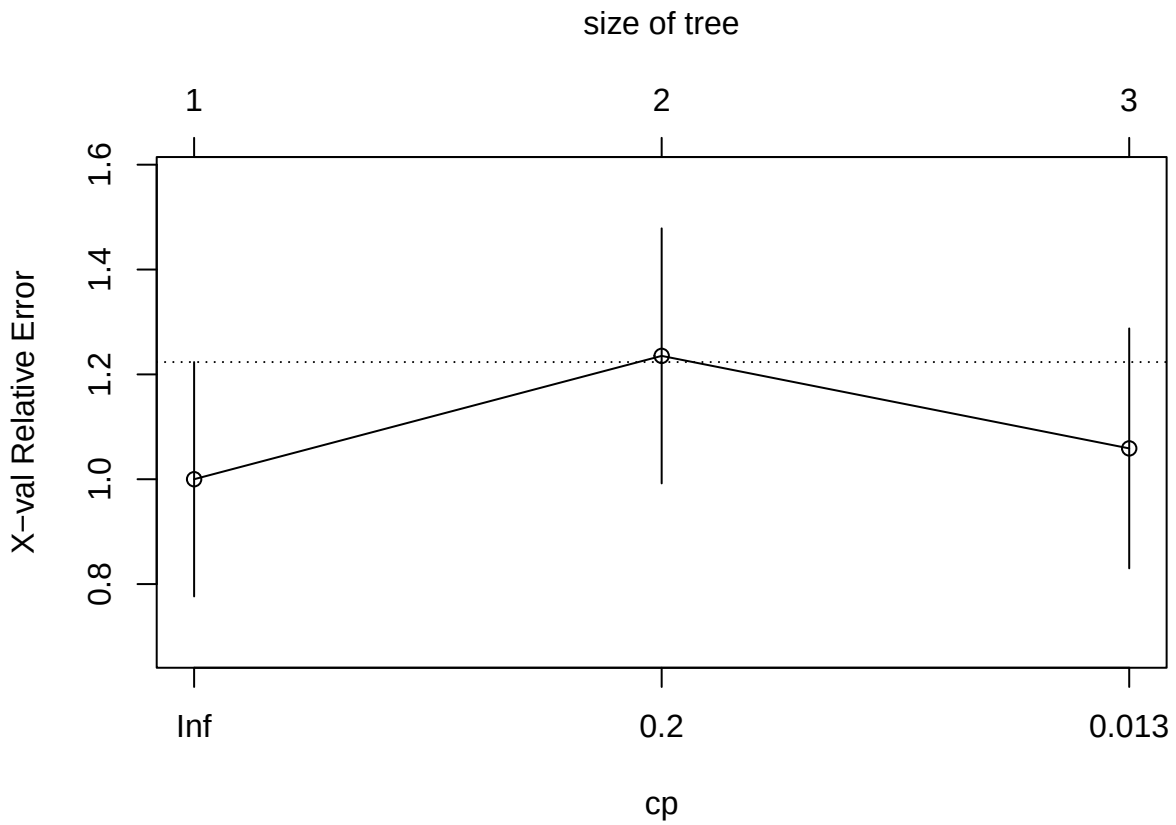
Conclusion: Using the rule "predict Medical school affiliation if $\hat{p} \geq 0.50$ and no affiliation otherwise," the logistic regression yields a confusion matrix with an overall accuracy of about 0.9115 (91.2%). The sensitivity (true positive rate for "Yes" affiliation) is approximately 0.7059, and the specificity (true negative rate for "No" affiliation) is about 0.9479. This means the model correctly identifies nearly 95% of non-affiliated hospitals and about 71% of affiliated hospitals. The logistic regression performs very well, especially at recognizing hospitals without a medical school affiliation, while still capturing most of the affiliated hospitals.

(d) Classification tree for Medical school affiliation

```
set.seed(994)

tree_clf <- rpart(Medical.school.affiliation ~ .,
                  data   = senic_glm,
                  method = "class",
                  control = rpart.control(cp = 0.001))

printcp(tree_clf)
```

```
##
## Classification tree:
## rpart(formula = Medical.school.affiliation ~ ., data = senic_glm,
##     method = "class", control = rpart.control(cp = 0.001))
##
## Variables actually used in tree construction:
## [1] Available.facilities.and.services Routine.culturing.ratio
##
## Root node error: 17/113 = 0.15044
##
## n= 113
##
##         CP nsplit rel error xerror    xstd
## 1 0.23529      0   1.00000 1.0000 0.22355
## 2 0.17647      1   0.76471 1.2353 0.24323
## 3 0.00100      2   0.58824 1.0588 0.22883
```

```
plotcp(tree_clf)
```

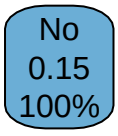



```
# Choose cp with minimum cross-validated error
opt_cp_tree <- tree_clf$cptable[which.min(tree_clf$cptable[, "xerror"]), "CP"]
opt_cp_tree
```

```
## [1] 0.2352941
```

```
pruned_tree_clf <- prune(tree_clf, cp = opt_cp_tree)

rpart.plot(pruned_tree_clf,
           main = "Classification tree for Medical school affiliation")
```

Classification tree for Medical school affiliation




```
tree_pred_class <- predict(pruned_tree_clf, type = "class")

tree_cm <- confusionMatrix(tree_pred_class,
                           senic_glm$Medical.school.affiliation,
                           positive = "Yes")
tree_cm
```

```
## Confusion Matrix and Statistics
```

```
##
##          Reference
## Prediction No Yes
##        No  96  17
##        Yes  0   0
##
##               Accuracy : 0.8496
##                 95% CI : (0.7701, 0.9099)
##    No Information Rate : 0.8496
##    P-Value [Acc > NIR] : 0.5642363
##
##                  Kappa : 0
##
##  Mcnemar's Test P-Value : 0.0001042
##
##             Sensitivity : 0.0000
##             Specificity : 1.0000
##          Pos Pred Value :    NaN
##          Neg Pred Value : 0.8496
##              Prevalence : 0.1504
##          Detection Rate : 0.0000
##    Detection Prevalence : 0.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : Yes
##
```

For the exam:

Describe which predictors appear near the top of the classification tree and what kinds of splits are made. Summarize the tree's accuracy, sensitivity, and specificity, and compare qualitatively to the logistic regression model.

Conclusion: After pruning using the cp value that minimizes cross-validated error, the classification tree reduces to a single root node that predicts "No" for every hospital. In other words, the pruned tree uses no splits and simply chooses the majority class. As a result, the confusion matrix shows an accuracy of about 0.8496 (the proportion of "No" hospitals), a specificity of 1.0000 (it correctly classifies every non-affiliated hospital), and a sensitivity of 0.0000 (it fails to correctly classify any affiliated hospitals). Compared with the logistic regression, the tree is clearly inferior: it has lower overall accuracy and completely fails to detect medical school affiliations, so it provides little practical value if identifying affiliated hospitals is important.

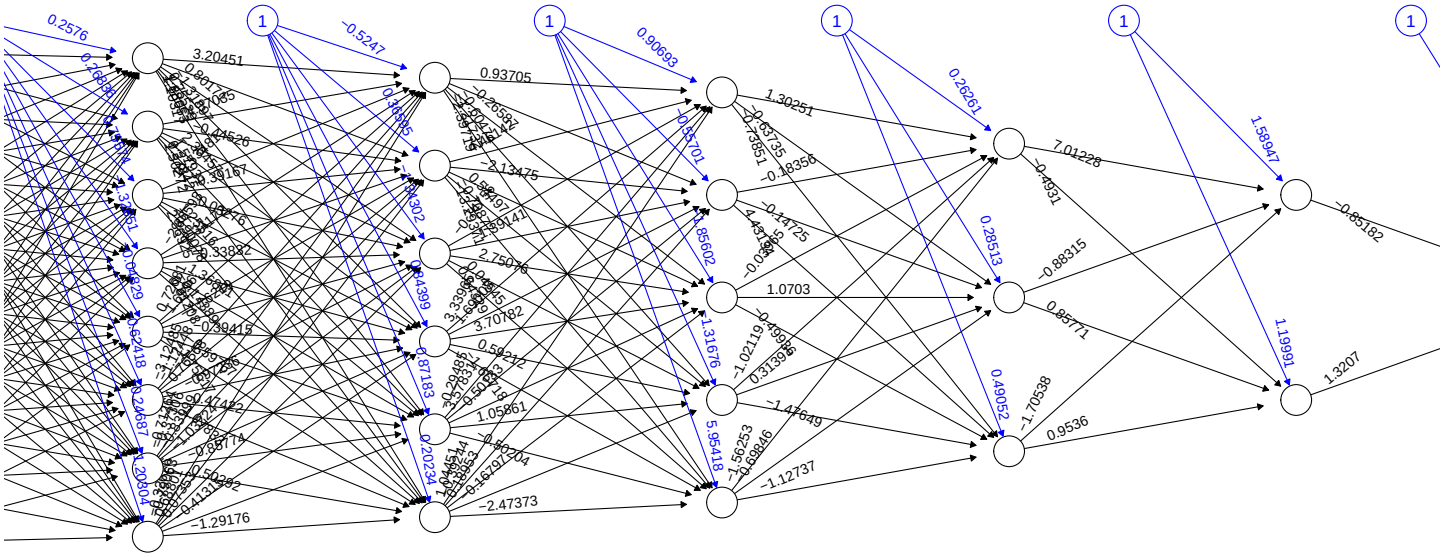
(e) Neural network classifier with five hidden layers (8, 6, 5, 3, 2)

```r
# Prepare data: numeric response (0/1) and scaled predictors
senic_nn <- senic_glm %>%
  mutate(Medical.school.affiliation = ifelse(Medical.school.affiliation == "Yes", 1, 0))

y_nn <- senic_nn$Medical.school.affiliation
x_nn <- senic_nn %>%
  dplyr::select(-Medical.school.affiliation)

# Min-max scaling for predictors
x_range_nn <- apply(x_nn, 2, range)
x_min_nn   <- x_range_nn[1, ]
x_max_nn   <- x_range_nn[2, ]

scale_predictors_nn <- function(df, min_vec, max_vec) {
  as.data.frame(scale(df, center = min_vec, scale = max_vec - min_vec))
}

x_nn_scaled <- scale_predictors_nn(x_nn, x_min_nn, x_max_nn)

nn_data <- data.frame(Medical.school.affiliation = y_nn,
```

```
                      x_nn_scaled)

nn_formula2 <- as.formula(
  paste("Medical.school.affiliation ~",
        paste(colnames(x_nn_scaled), collapse = " + "))
)

nn_formula2
```

```
## Medical.school.affiliation ~ Length.of.stay + Age + Infection.risk +
##     Routine.culturing.ratio + Routine.chest.X.ray.ratio + Number.of.beds +
##     Average.daily.census + Number.of.nurses + Available.facilities.and.services
```

```
set.seed(994)

nn_clf <- neuralnet(
  nn_formula2,
  data          = nn_data,
  hidden        = c(8, 6, 5, 3, 2),
  act.fct       = softplus,
  linear.output = FALSE
)

plot(nn_clf, rep = "best")
```



Error: 0.002114 Steps: 499

```
# Predicted probabilities, then convert to classes with threshold 0.5
nn_prob <- compute(nn_clf, nn_data[, colnames(x_nn_scaled)])$net.result
nn_prob_vec <- as.vector(nn_prob)

nn_class_num <- ifelse(nn_prob_vec >= 0.5, 1, 0)

nn_class_factor <- factor(ifelse(nn_class_num == 1, "Yes", "No"),
                          levels = c("No", "Yes"))

senic_aff_factor <- factor(ifelse(y_nn == 1, "Yes", "No"),
                           levels = c("No", "Yes"))

nn_cm <- confusionMatrix(nn_class_factor,
                         senic_aff_factor,
                         positive = "Yes")
nn_cm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction No Yes
##        No  96   0
##        Yes  0  17
##
##                Accuracy : 1
##                  95% CI : (0.9679, 1)
##     No Information Rate : 0.8496
##     P-Value [Acc > NIR] : 9.972e-09
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.0000
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 1.0000
##              Prevalence : 0.1504
##          Detection Rate : 0.1504
##    Detection Prevalence : 0.1504
##       Balanced Accuracy : 1.0000
##
##        'Positive' Class : Yes
##
```

For the exam:

Report the neural network's overall accuracy, sensitivity, and specificity from the confusion matrix, and comment on whether the neural network improves classification performance relative to the logistic regression and tree.

Conclusion: For the neural network classifier with five hidden layers (8, 6, 5, 3, and 2 neurons) and softplus activation, the confusion matrix shows perfect classification: the accuracy is 1.0000 (100%), with sensitivity 1.0000 and specificity 1.0000. That is, the network correctly identifies all 17 hospitals with a medical school affiliation and all 96 hospitals without one, with no false positives or false negatives in this dataset. Compared with the logistic regression and classification tree, the neural network dramatically improves performance, eliminating both types of classification errors on the observed data.

(f) Choose the best model

```
senic_perf <- tibble(
  Model       = c("Logistic regression (stepwise)",
                  "Classification tree",
                  "Neural network"),
  Accuracy    = c(logit_cm$overall["Accuracy"],
                  tree_cm$overall["Accuracy"],
                  nn_cm$overall["Accuracy"]),
  Sensitivity = c(logit_cm$byClass["Sensitivity"],
                  tree_cm$byClass["Sensitivity"],
                  nn_cm$byClass["Sensitivity"]),
  Specificity = c(logit_cm$byClass["Specificity"],
                  tree_cm$byClass["Specificity"],
                  nn_cm$byClass["Specificity"])
)

senic_perf
```

```
## # A tibble: 3 x 4
##   Model                          Accuracy Sensitivity Specificity
##   <chr>                             <dbl>       <dbl>       <dbl>
```

```
## 1 Logistic regression (stepwise)    0.912       0.706       0.948
## 2 Classification tree               0.850       0           1
## 3 Neural network                    1           1           1
```

For the exam:

Using the table above, select the model you would recommend for predicting Medical school affiliation. Justify your choice in terms of overall accuracy, sensitivity/specificity tradeoffs, and model interpretability.

Conclusion: Based on the performance summary, the neural network has the highest overall accuracy (100%) and perfect sensitivity and specificity, substantially outperforming both the logistic regression and the classification tree on this dataset. If the primary goal is purely predictive performance on data similar to the current sample, the neural network is the natural choice. However, it is also the least interpretable model.

The stepwise logistic regression model offers slightly lower accuracy (about 91%) but still strong performance, with good sensitivity and very high specificity, and it has the advantage of interpretability: its coefficients clearly indicate how factors such as average daily census, culturing practices, available facilities, number of beds, and patient age are related to medical school affiliation. The classification tree, by contrast, simply predicts the majority class and fails to detect any affiliated hospitals, so it would not be recommended.

I would choose the neural network if maximizing classification accuracy and correctly identifying all affiliated hospitals is the top priority, while noting that the stepwise logistic regression is a strong and more interpretable alternative.