Midterm

Srinivasa Phani Madhav

3/28/2024

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library(neuralnet)
```

```
## Warning: package 'neuralnet' was built under R version 4.3.3
```

```
library(keras)
```

```
## Warning: package 'keras' was built under R version 4.3.3
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::compute() masks neuralnet::compute()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x purrr::lift()    masks caret::lift()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(tensorflow)
```

```
## Warning: package 'tensorflow' was built under R version 4.3.3
```

```
##
## Attaching package: 'tensorflow'
##
## The following object is masked from 'package:caret':
##
##     train
```

```
library(dplyr)
library(cloudml)
```

```
## Warning: package 'cloudml' was built under R version 4.3.3
```

```
## Loading required package: tfruns
```

```
## Warning: package 'tfruns' was built under R version 4.3.3
```

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.3.3
```

```
## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:dplyr':
##
##     combine
##
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(rpart)
```

```
## Warning: package 'rpart' was built under R version 4.3.3
```

```
library(rattle)
```

```
## Warning: package 'rattle' was built under R version 4.3.3
```

```
## Loading required package: bitops
## Rattle: A free graphical interface for data science with R.
## Version 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
##
## Attaching package: 'rattle'
##
## The following object is masked from 'package:randomForest':
##
##     importance
```

Part 1

1

```
data <- read.csv("C:/Users/MSP/Downloads/Unknown.csv",header = T)
head(data)
```

```
##     u1   u2 u3   u4  u5 u6   u7 u8    u9   u10   u11   u12 y
## 1 1.59 0.00  0 0.63 0.0  0 1.59  0 0.000 0.275 0.000 0.496 1
## 2 0.00 0.00  0 0.00 0.0  0 0.00  0 0.000 0.729 0.000 0.000 1
## 3 0.76 0.15  0 0.00 0.1  0 0.00  0 0.042 0.101 0.016 0.046 1
## 4 2.94 0.00  0 2.94 0.0  0 0.00  0 0.404 0.404 0.000 0.000 1
## 5 1.16 0.00  0 1.16 0.0  0 0.00  0 0.000 0.133 0.000 0.000 1
## 6 0.00 0.00  0 0.00 0.0  0 0.00  0 0.000 0.196 0.000 0.196 1
```

```
data = na.omit(data)
cat('There are', nrow(data), 'observations left.')
```

```
## There are 4621 observations left.
```

```r
set.seed(123)
training.samples <- data$y %>%
  createDataPartition(p = 0.75, list = FALSE)
train.data  <- data[training.samples, ]
test.data <- data[-training.samples, ]
str(train.data)
```

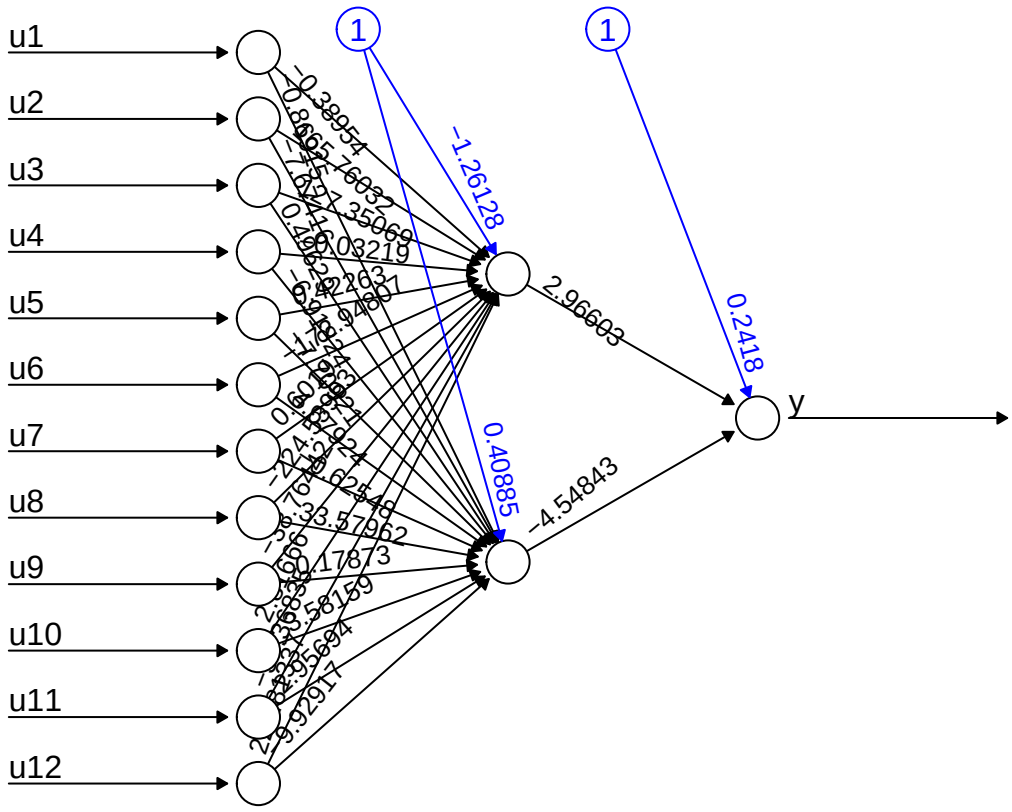
```
## 'data.frame':    3466 obs. of  13 variables:
##  $ u1 : num  1.59 0 0.76 2.94 1.16 0.76 0 0 0.32 0.14 ...
##  $ u2 : num  0 0 0.15 0 0 0.15 1.66 0 0 0.21 ...
##  $ u3 : num  0 0 0 0 0 0 0 0 0 0.94 ...
##  $ u4 : num  0.63 0 0 2.94 1.16 0 0 3.33 0.32 0.14 ...
##  $ u5 : num  0 0 0.1 0 0 0.1 0 0 0 0 ...
##  $ u6 : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ u7 : num  1.59 0 0 0 0 0 0 0 0 0 ...
##  $ u8 : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ u9 : num  0 0 0.042 0.404 0 0.042 0 0 0 0 ...
##  $ u10: num  0.275 0.729 0.101 0.404 0.133 0.101 0 0.352 0 0.132 ...
##  $ u11: num  0 0 0.016 0 0 0.016 0 0 0 0 ...
##  $ u12: num  0.496 0 0.046 0 0 0.046 0 0 0 0.18 ...
##  $ y  : int  1 1 1 1 1 1 1 1 1 1 ...
```

```r
str(test.data)
```

```
## 'data.frame':    1155 obs. of  13 variables:
##  $ u1 : num  0 0 0.63 1.85 0.38 0.34 0.9 0 1.11 0 ...
##  $ u2 : num  0 0 0.31 0 0.25 0 0.9 0 0.18 1.66 ...
##  $ u3 : num  0 0 0.63 0 0 0 0.9 0 0 0 ...
##  $ u4 : num  0 0 0.31 0 0 0.34 0 0 0 0 ...
##  $ u5 : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ u6 : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ u7 : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ u8 : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ u9 : num  0 0 0 0 0.022 0 0 0 0 0 ...
##  $ u10: num  0.196 0.196 0.137 0.223 0.044 0.056 0 0 0.182 0 ...
##  $ u11: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ u12: num  0.196 0.196 0 0 0 0 0 0.37 0 0 ...
##  $ y  : int  1 1 1 1 1 1 1 1 1 1 ...
```

2

```r
set.seed(123)
model1 = neuralnet(y~., data = train.data, hidden = 2, err.fct = "ce",act.fct = "logistic", linear.outpu
plot(model1, rep = "best") # plot the model
```



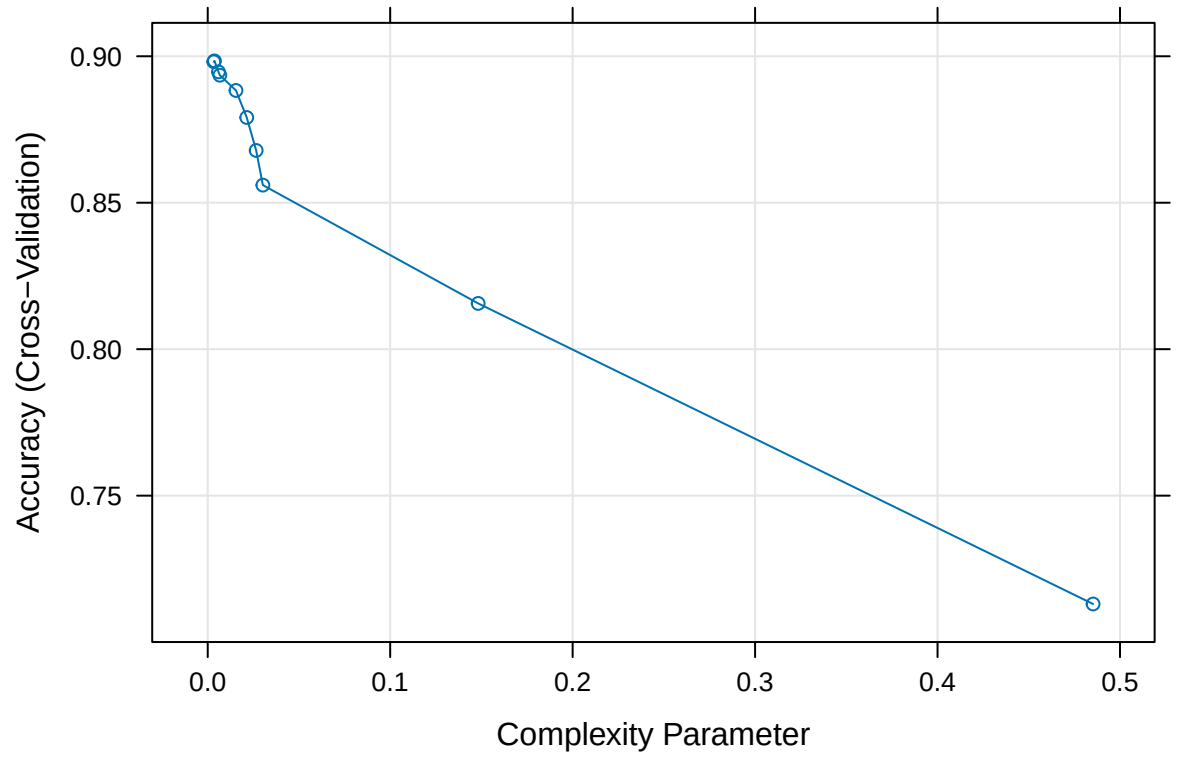

```
probabilities = predict(model1, test.data)
predicted.classes = ifelse(probabilities > 0.5, 1, 0)

(c1 = confusionMatrix(factor(predicted.classes), factor(test.data$y), positive = "1"))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 636  65
##          1  52 402
##
##                Accuracy : 0.8987
##                  95% CI : (0.8798, 0.9155)
##     No Information Rate : 0.5957
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.7888
##
##  Mcnemar's Test P-Value : 0.2673
##
##             Sensitivity : 0.8608
##             Specificity : 0.9244
##          Pos Pred Value : 0.8855
##          Neg Pred Value : 0.9073
##              Prevalence : 0.4043
```

```
##          Detection Rate : 0.3481
##    Detection Prevalence : 0.3931
##       Balanced Accuracy : 0.8926
##
##        'Positive' Class : 1
##
```

3

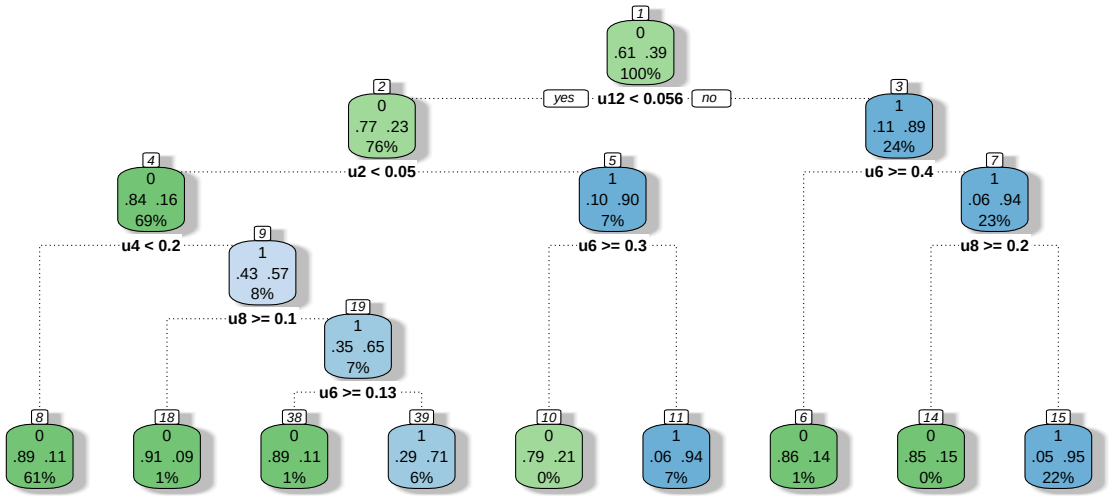
```
detach(package:keras,unload=TRUE)
detach(package:tensorflow,unload=TRUE)

train.data$y = as.factor(train.data$y)
test.data$y = as.factor(test.data$y)

set.seed(123)
model2 = train(
  y ~., data = train.data, method = "rpart",
  trControl = trainControl("cv", number = 10),
  tuneLength = 10)
plot(model2)
```




```
model2$bestTune
```

```
##           cp
## 2 0.003687316
```

```
fancyRpartPlot(model2$finalModel)
```



Rattle 2024−Mar−28 09:48:19 MSP

```
pred = predict(model2, newdata = test.data)
(c2 = confusionMatrix(pred, factor(test.data$y), positive = "1"))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 651  80
##          1  37 387
##
##                Accuracy : 0.8987
##                  95% CI : (0.8798, 0.9155)
##     No Information Rate : 0.5957
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7865
##
##  Mcnemar's Test P-Value : 0.0001032
##
##             Sensitivity : 0.8287
##             Specificity : 0.9462
##          Pos Pred Value : 0.9127
##          Neg Pred Value : 0.8906
##              Prevalence : 0.4043
```

```
##          Detection Rate : 0.3351
##    Detection Prevalence : 0.3671
##       Balanced Accuracy : 0.8875
##
##        'Positive' Class : 1
##
```

#4

```
predictions <- data.frame(
  Neural_Network_Prediction = predicted.classes,
  Pruned_Tree_Prediction = pred
)
# Create a 2x2 table to show agreement between predictions
agreement_table <- table(predictions$Neural_Network_Prediction, predictions$Pruned_Tree_Prediction)

# Output the agreement table
print("Agreement Table:")
```

```
## [1] "Agreement Table:"
```

```
print(agreement_table)
```

```
##
##       0   1
##   0 694   7
##   1  37 417
```

Part2

1

```
data1 = read.csv("C:/Users/MSP/Downloads/Madness.csv")
data1$x15 = ifelse(data1$x15 == 'y', 1, 0)
mean = mean(data1$y)
sd = sd(data1$y)
data1 = data.frame(scale(data1))
data = na.omit(data1)

cat('There are', nrow(data1) - nrow(data), 'missing values.')
```

```
## There are 0 missing values.
```

```
set.seed(123)
training.samples <- data$y %>%
  createDataPartition(p = 0.75, list = FALSE)
train.data  <- data[training.samples, ]
test.data <- data[-training.samples, ]
str(train.data)
```

```
## 'data.frame':    1097 obs. of  16 variables:
##  $ x1 : num  -0.183 -0.977 -0.944 -0.679 -1.109 ...
##  $ x2 : num  -0.0919 0.0735 0.375 0.3605 -0.0434 ...
##  $ x3 : num  -0.0718 0.6513 1.3743 -0.7949 1.3743 ...
##  $ x4 : num  2.179 -0.517 -0.517 -0.517 -0.517 ...
##  $ x5 : num  0.2571 -0.6276 -0.0456 -0.9484 1.3745 ...
```

```
##  $ x6 : num  -0.795 1.189 1.617 0.502 -0.795 ...
##  $ x7 : num  -0.482 0.515 1.299 -0.292 0.34 ...
##  $ x8 : num  0.789 0.789 0.789 -1.026 0.789 ...
##  $ x9 : num  0.164 0.164 1.39 -2.288 0.164 ...
##  $ x10: num  -0.211 -0.211 -0.211 -0.211 -0.211 ...
##  $ x11: num  -0.319 -0.319 1.527 -0.934 0.297 ...
##  $ x12: num  0.6 0.6 0.6 -0.951 0.6 ...
##  $ x13: num  0.312 0.312 1.65 0.312 0.312 ...
##  $ x14: num  -0.0607 0.6315 1.6979 0.0328 0.7625 ...
##  $ x15: num  1.43 1.43 1.43 1.43 1.43 ...
##  $ y  : num  0.00729 0.53597 0.86954 -0.47734 1.58704 ...
```

```
str(test.data)
```

```
## 'data.frame':    363 obs. of  16 variables:
##  $ x1 : num  -1.0429 1.8001 -0.0181 0.2133 0.3125 ...
##  $ x2 : num  -0.2071 -0.0969 -0.0135 0.0684 0.2456 ...
##  $ x3 : num  0.651 0.651 0.651 -0.795 -0.795 ...
##  $ x4 : num  -0.517 -0.517 0.382 -0.517 0.382 ...
##  $ x5 : num  -0.793 -0.522 -0.144 -0.317 -0.648 ...
##  $ x6 : num  1.161 0.937 1.457 -0.795 -0.795 ...
##  $ x7 : num  0.37 0.384 1.093 -0.905 -1.148 ...
##  $ x8 : num  0.789 -1.026 0.789 -1.026 -1.026 ...
##  $ x9 : num  0.164 0.164 0.164 0.164 -1.062 ...
##  $ x10: num  -0.211 -0.211 -0.211 -0.211 -0.211 ...
##  $ x11: num  0.912 0.297 0.297 -0.934 -1.549 ...
##  $ x12: num  -0.951 0.6 2.151 -0.951 -0.951 ...
##  $ x13: num  0.312 1.65 0.312 -1.027 -1.027 ...
##  $ x14: num  0.3509 0.7905 0.0515 -0.4162 -0.5658 ...
##  $ x15: num  1.426 1.426 1.426 -0.701 -0.701 ...
##  $ y  : num  0.347 -0.515 0.24 -0.647 -0.465 ...
```

2

```
library(keras)
```

```
## Warning: package 'keras' was built under R version 4.3.3
```

```
train_x <- as.matrix(subset(train.data, select = -y))
train_y <- as.matrix(subset(train.data, select = y))
test_x <- as.matrix(subset(test.data, select = -y))
test_y <- as.matrix(subset(test.data, select = y))

set.seed(123)

model3 <- keras_model_sequential()
model3 %>%
  layer_dense(units = 2, activation = 'linear', input_shape = c(ncol(train_x))) %>%
  layer_dense(units = 1, activation = "linear")

model3 %>% compile(loss='mse', optimizer='adam', metrics='mse')

summary(model3)
```

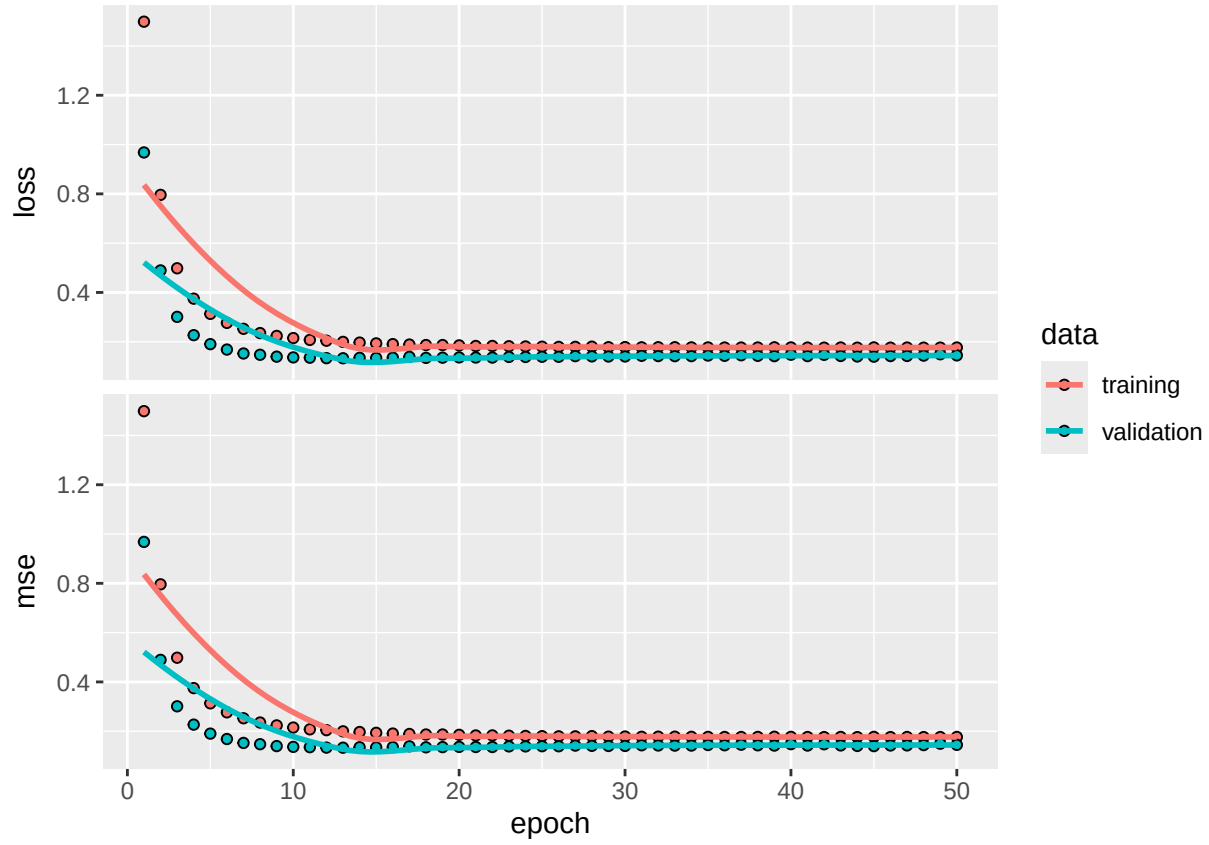
```
## Model: "sequential"
```

```
## ________________________________________________________________________________
##  Layer (type)                       Output Shape                    Param #
## ================================================================================
##  dense_1 (Dense)                    (None, 2)                       32
##  dense (Dense)                      (None, 1)                       3
## ================================================================================
## Total params: 35 (140.00 Byte)
## Trainable params: 35 (140.00 Byte)
## Non-trainable params: 0 (0.00 Byte)
## ________________________________________________________________________________
history <- model3 %>% fit(train_x, train_y, epochs = 50, batch_size = 8, validation_split = 0.1)

## Epoch 1/50
## 124/124 - 1s - loss: 1.4985 - mse: 1.4985 - val_loss: 0.9680 - val_mse: 0.9680 - 788ms/epoch - 6ms/s
## Epoch 2/50
## 124/124 - 0s - loss: 0.7959 - mse: 0.7959 - val_loss: 0.4895 - val_mse: 0.4895 - 200ms/epoch - 2ms/s
## Epoch 3/50
## 124/124 - 0s - loss: 0.4981 - mse: 0.4981 - val_loss: 0.3010 - val_mse: 0.3010 - 174ms/epoch - 1ms/s
## Epoch 4/50
## 124/124 - 0s - loss: 0.3749 - mse: 0.3749 - val_loss: 0.2267 - val_mse: 0.2267 - 176ms/epoch - 1ms/s
## Epoch 5/50
## 124/124 - 0s - loss: 0.3131 - mse: 0.3131 - val_loss: 0.1903 - val_mse: 0.1903 - 170ms/epoch - 1ms/s
## Epoch 6/50
## 124/124 - 0s - loss: 0.2766 - mse: 0.2766 - val_loss: 0.1683 - val_mse: 0.1683 - 173ms/epoch - 1ms/s
## Epoch 7/50
## 124/124 - 0s - loss: 0.2525 - mse: 0.2525 - val_loss: 0.1525 - val_mse: 0.1525 - 186ms/epoch - 1ms/s
## Epoch 8/50
## 124/124 - 0s - loss: 0.2352 - mse: 0.2352 - val_loss: 0.1473 - val_mse: 0.1473 - 166ms/epoch - 1ms/s
## Epoch 9/50
## 124/124 - 0s - loss: 0.2236 - mse: 0.2236 - val_loss: 0.1396 - val_mse: 0.1396 - 173ms/epoch - 1ms/s
## Epoch 10/50
## 124/124 - 0s - loss: 0.2150 - mse: 0.2150 - val_loss: 0.1365 - val_mse: 0.1365 - 174ms/epoch - 1ms/s
## Epoch 11/50
## 124/124 - 0s - loss: 0.2076 - mse: 0.2076 - val_loss: 0.1353 - val_mse: 0.1353 - 162ms/epoch - 1ms/s
## Epoch 12/50
## 124/124 - 0s - loss: 0.2043 - mse: 0.2043 - val_loss: 0.1334 - val_mse: 0.1334 - 166ms/epoch - 1ms/s
## Epoch 13/50
## 124/124 - 0s - loss: 0.1995 - mse: 0.1995 - val_loss: 0.1331 - val_mse: 0.1331 - 168ms/epoch - 1ms/s
## Epoch 14/50
## 124/124 - 0s - loss: 0.1967 - mse: 0.1967 - val_loss: 0.1349 - val_mse: 0.1349 - 160ms/epoch - 1ms/s
## Epoch 15/50
## 124/124 - 0s - loss: 0.1935 - mse: 0.1935 - val_loss: 0.1343 - val_mse: 0.1343 - 170ms/epoch - 1ms/s
## Epoch 16/50
## 124/124 - 0s - loss: 0.1904 - mse: 0.1904 - val_loss: 0.1354 - val_mse: 0.1354 - 160ms/epoch - 1ms/s
## Epoch 17/50
## 124/124 - 0s - loss: 0.1887 - mse: 0.1887 - val_loss: 0.1388 - val_mse: 0.1388 - 170ms/epoch - 1ms/s
## Epoch 18/50
## 124/124 - 0s - loss: 0.1867 - mse: 0.1867 - val_loss: 0.1347 - val_mse: 0.1347 - 168ms/epoch - 1ms/s
## Epoch 19/50
## 124/124 - 0s - loss: 0.1867 - mse: 0.1867 - val_loss: 0.1355 - val_mse: 0.1355 - 184ms/epoch - 1ms/s
## Epoch 20/50
## 124/124 - 0s - loss: 0.1853 - mse: 0.1853 - val_loss: 0.1365 - val_mse: 0.1365 - 161ms/epoch - 1ms/s
## Epoch 21/50
## 124/124 - 0s - loss: 0.1828 - mse: 0.1828 - val_loss: 0.1360 - val_mse: 0.1360 - 168ms/epoch - 1ms/s
```

```
## Epoch 22/50
## 124/124 - 0s - loss: 0.1830 - mse: 0.1830 - val_loss: 0.1359 - val_mse: 0.1359 - 181ms/epoch - 1ms/s
## Epoch 23/50
## 124/124 - 0s - loss: 0.1818 - mse: 0.1818 - val_loss: 0.1385 - val_mse: 0.1385 - 182ms/epoch - 1ms/s
## Epoch 24/50
## 124/124 - 0s - loss: 0.1809 - mse: 0.1809 - val_loss: 0.1384 - val_mse: 0.1384 - 174ms/epoch - 1ms/s
## Epoch 25/50
## 124/124 - 0s - loss: 0.1800 - mse: 0.1800 - val_loss: 0.1392 - val_mse: 0.1392 - 172ms/epoch - 1ms/s
## Epoch 26/50
## 124/124 - 0s - loss: 0.1792 - mse: 0.1792 - val_loss: 0.1390 - val_mse: 0.1390 - 233ms/epoch - 2ms/s
## Epoch 27/50
## 124/124 - 0s - loss: 0.1793 - mse: 0.1793 - val_loss: 0.1417 - val_mse: 0.1417 - 188ms/epoch - 2ms/s
## Epoch 28/50
## 124/124 - 0s - loss: 0.1801 - mse: 0.1801 - val_loss: 0.1407 - val_mse: 0.1407 - 190ms/epoch - 2ms/s
## Epoch 29/50
## 124/124 - 0s - loss: 0.1780 - mse: 0.1780 - val_loss: 0.1402 - val_mse: 0.1402 - 181ms/epoch - 1ms/s
## Epoch 30/50
## 124/124 - 0s - loss: 0.1785 - mse: 0.1785 - val_loss: 0.1403 - val_mse: 0.1403 - 177ms/epoch - 1ms/s
## Epoch 31/50
## 124/124 - 0s - loss: 0.1774 - mse: 0.1774 - val_loss: 0.1430 - val_mse: 0.1430 - 165ms/epoch - 1ms/s
## Epoch 32/50
## 124/124 - 0s - loss: 0.1779 - mse: 0.1779 - val_loss: 0.1421 - val_mse: 0.1421 - 164ms/epoch - 1ms/s
## Epoch 33/50
## 124/124 - 0s - loss: 0.1779 - mse: 0.1779 - val_loss: 0.1416 - val_mse: 0.1416 - 154ms/epoch - 1ms/s
## Epoch 34/50
## 124/124 - 0s - loss: 0.1773 - mse: 0.1773 - val_loss: 0.1418 - val_mse: 0.1418 - 163ms/epoch - 1ms/s
## Epoch 35/50
## 124/124 - 0s - loss: 0.1770 - mse: 0.1770 - val_loss: 0.1441 - val_mse: 0.1441 - 161ms/epoch - 1ms/s
## Epoch 36/50
## 124/124 - 0s - loss: 0.1767 - mse: 0.1767 - val_loss: 0.1431 - val_mse: 0.1431 - 172ms/epoch - 1ms/s
## Epoch 37/50
## 124/124 - 0s - loss: 0.1766 - mse: 0.1766 - val_loss: 0.1454 - val_mse: 0.1454 - 165ms/epoch - 1ms/s
## Epoch 38/50
## 124/124 - 0s - loss: 0.1768 - mse: 0.1768 - val_loss: 0.1429 - val_mse: 0.1429 - 154ms/epoch - 1ms/s
## Epoch 39/50
## 124/124 - 0s - loss: 0.1776 - mse: 0.1776 - val_loss: 0.1418 - val_mse: 0.1418 - 162ms/epoch - 1ms/s
## Epoch 40/50
## 124/124 - 0s - loss: 0.1771 - mse: 0.1771 - val_loss: 0.1475 - val_mse: 0.1475 - 213ms/epoch - 2ms/s
## Epoch 41/50
## 124/124 - 0s - loss: 0.1765 - mse: 0.1765 - val_loss: 0.1419 - val_mse: 0.1419 - 177ms/epoch - 1ms/s
## Epoch 42/50
## 124/124 - 0s - loss: 0.1762 - mse: 0.1762 - val_loss: 0.1471 - val_mse: 0.1471 - 165ms/epoch - 1ms/s
## Epoch 43/50
## 124/124 - 0s - loss: 0.1767 - mse: 0.1767 - val_loss: 0.1424 - val_mse: 0.1424 - 166ms/epoch - 1ms/s
## Epoch 44/50
## 124/124 - 0s - loss: 0.1761 - mse: 0.1761 - val_loss: 0.1403 - val_mse: 0.1403 - 204ms/epoch - 2ms/s
## Epoch 45/50
## 124/124 - 0s - loss: 0.1771 - mse: 0.1771 - val_loss: 0.1395 - val_mse: 0.1395 - 174ms/epoch - 1ms/s
## Epoch 46/50
## 124/124 - 0s - loss: 0.1762 - mse: 0.1762 - val_loss: 0.1421 - val_mse: 0.1421 - 175ms/epoch - 1ms/s
## Epoch 47/50
## 124/124 - 0s - loss: 0.1766 - mse: 0.1766 - val_loss: 0.1424 - val_mse: 0.1424 - 161ms/epoch - 1ms/s
## Epoch 48/50
## 124/124 - 0s - loss: 0.1762 - mse: 0.1762 - val_loss: 0.1438 - val_mse: 0.1438 - 173ms/epoch - 1ms/s
```

```
## Epoch 49/50
## 124/124 - 0s - loss: 0.1763 - mse: 0.1763 - val_loss: 0.1494 - val_mse: 0.1494 - 169ms/epoch - 1ms/s
## Epoch 50/50
## 124/124 - 0s - loss: 0.1767 - mse: 0.1767 - val_loss: 0.1447 - val_mse: 0.1447 - 169ms/epoch - 1ms/s
```

```
plot(history)
```




```
preds <- predict(model3, test_x)
```

```
## 12/12 - 0s - 71ms/epoch - 6ms/step
```

```
scaled_RMSE <- sqrt(mean((test_y - preds)^2))

mean_y <- mean(test.data$y)
sd_y <- sd(test.data$y)
test_RMSE <- sqrt(mean(((test_y * sd_y + mean_y) - (preds * sd_y + mean_y))^2))

scaled_RMSE
```

```
## [1] 0.6307747
```

```
test_RMSE
```

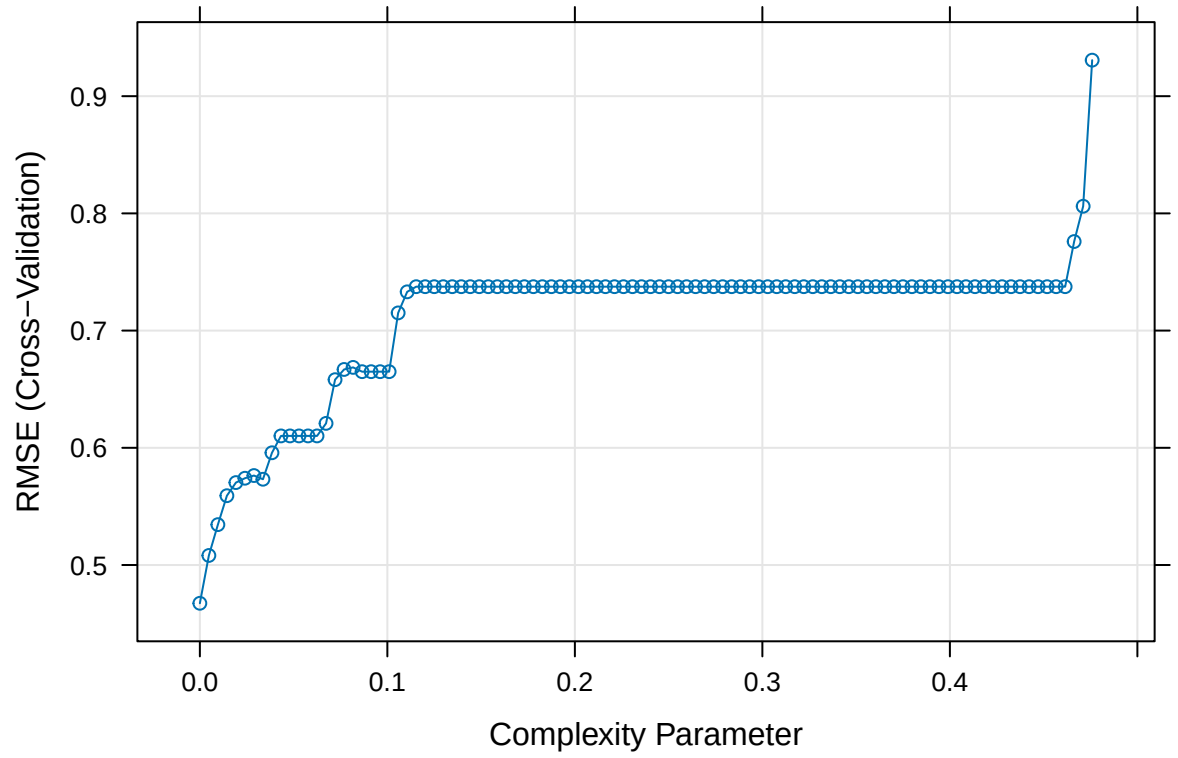
```
## [1] 0.590129
```

3

```
detach(package:keras,unload=TRUE)


model4 = train(
  y ~., data = train.data, method = "rpart",
  trControl = trainControl("cv", number = 10),
  tuneLength = 100)
```

```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo,
## : There were missing values in resampled performance measures.
```

```
plot(model4)
```

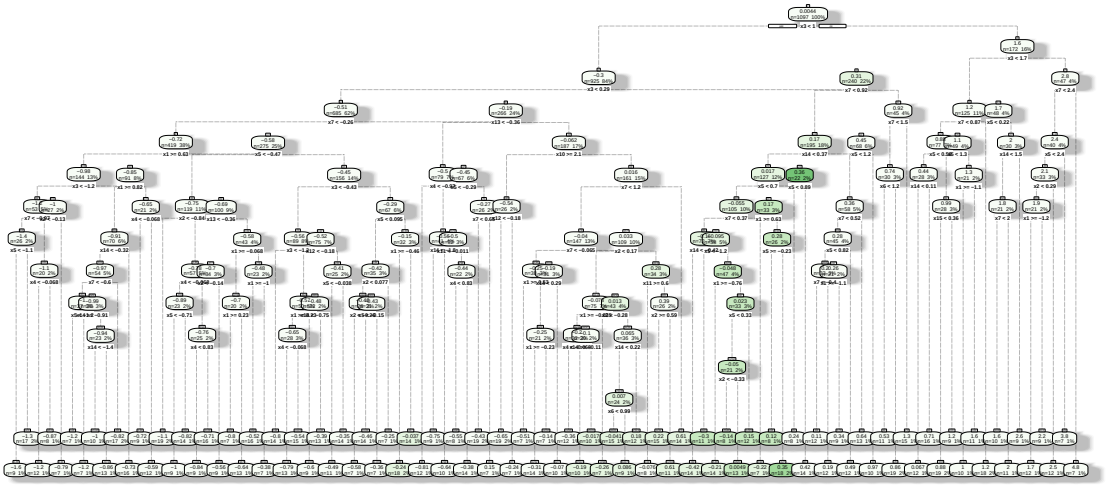



```
model4$bestTune
```

```
##   cp
## 1  0
```

```
fancyRpartPlot(model4$finalModel)
```

```
## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```



Rattle 2024−Mar−28 09:48:36 MSP

```
predictions = model4 %>% predict(test.data)

# scaled test RMSE
RMSE(test.data$y, predictions)
```

```
## [1] 0.5609893
```

```
# test RMSE
RMSE(test.data$y*sd+mean, predictions*sd+mean)
```

```
## [1] 445.664
```