Lab 19

2022-10-26

Trees

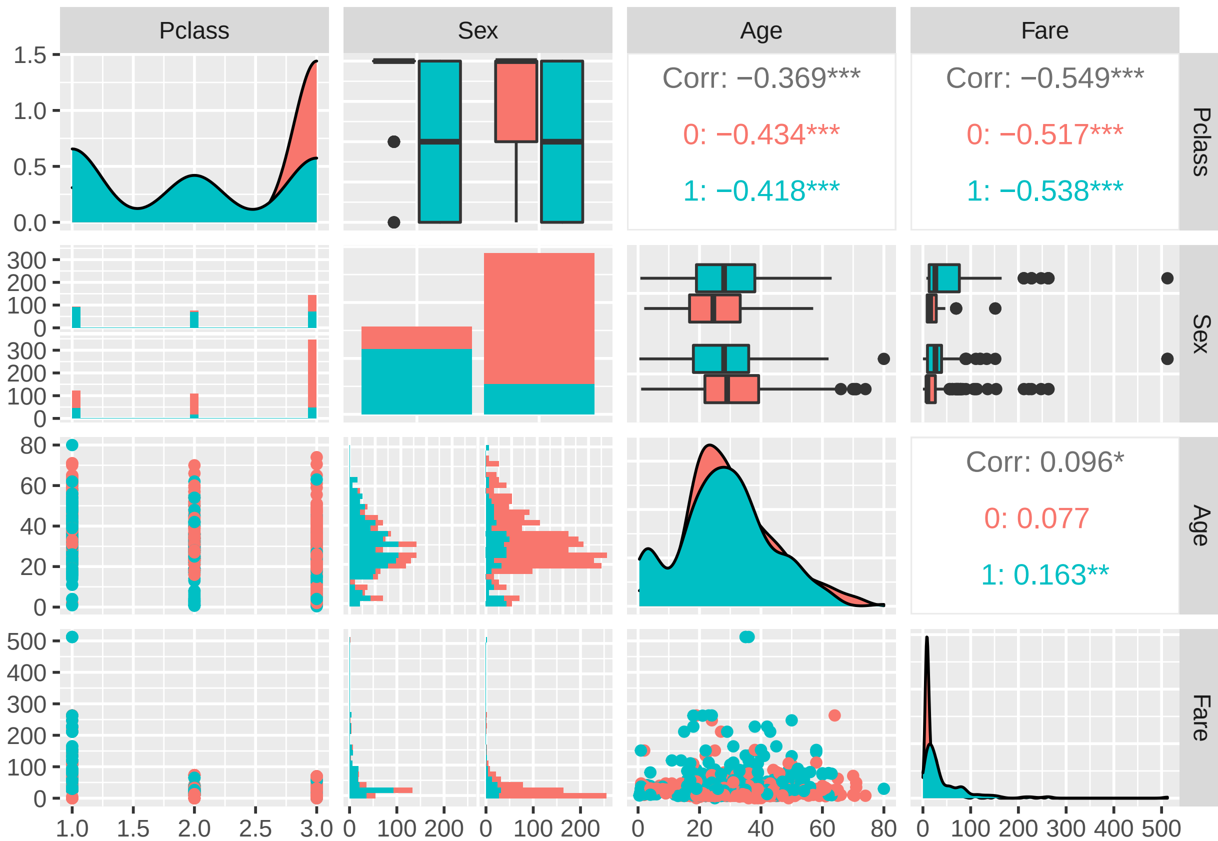
```r
# Classification and Regression Trees (CART)

# Look at data!
head(titanic_train)
```

```
##   PassengerId Survived Pclass
## 1           1        0      3
## 2           2        1      1
## 3           3        1      3
## 4           4        1      1
## 5           5        0      3
## 6           6        0      3
##                                                  Name    Sex Age SibSp Parch
## 1                             Braund, Mr. Owen Harris   male  22     1     0
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38     1     0
## 3                              Heikkinen, Miss. Laina female  26     0     0
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35     1     0
## 5                            Allen, Mr. William Henry   male  35     0     0
## 6                                    Moran, Mr. James   male  NA     0     0
##             Ticket    Fare Cabin Embarked
## 1        A/5 21171  7.2500              S
## 2         PC 17599 71.2833   C85        C
## 3 STON/O2. 3101282  7.9250              S
## 4           113803 53.1000  C123        S
## 5           373450  8.0500              S
## 6           330877  8.4583              Q
```

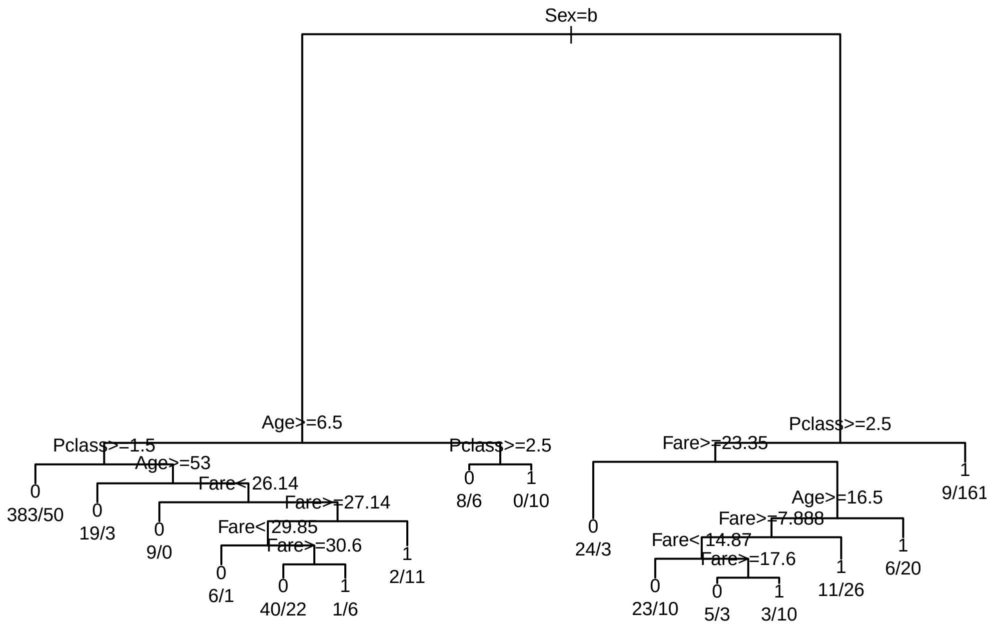
```r
?titanic_train

titanic_train$Survived = as.factor(titanic_train$Survived)

titanic_train %>%
  ggpairs(columns = c("Pclass",
                      "Sex",
                      "Age",
                      "Fare"),
          mapping = aes(color = Survived))
```

```
## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 177 rows containing missing values

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 177 rows containing non-finite values (stat_boxplot).
```

```
## Warning: Removed 177 rows containing missing values (geom_point).

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 177 rows containing non-finite values (stat_bin).

## Warning: Removed 177 rows containing non-finite values (stat_density).

## Warning in ggally_statistic(data = data, mapping = mapping, na.rm = na.rm, :
## Removed 177 rows containing missing values

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 177 rows containing missing values (geom_point).
```




```
# Decision tree model
treeModel = rpart(Survived ~ Pclass + Age + Fare + Sex, data=titanic_train, method="class", xval=5, cp=
par(cex = 0.6)
plot(treeModel,margin=.05)
text(treeModel,use.n=TRUE)
```




```
summary(treeModel)
```

```
## Call:
## rpart(formula = Survived ~ Pclass + Age + Fare + Sex, data = titanic_train,
##     method = "class", xval = 5, cp = 0.005)
##   n= 891
##
##             CP nsplit rel error    xerror       xstd
## 1 0.444444444      0 1.0000000 1.0000000 0.04244576
## 2 0.030701754      1 0.5555556 0.5555556 0.03574957
## 3 0.023391813      3 0.4941520 0.5116959 0.03467453
## 4 0.011695906      4 0.4707602 0.5233918 0.03497048
## 5 0.006578947      7 0.4327485 0.5146199 0.03474917
## 6 0.005847953     13 0.3918129 0.4883041 0.03406141
## 7 0.005000000     15 0.3801170 0.4795322 0.03382394
##
## Variable importance
##    Sex   Fare Pclass    Age
##     49     26     18      8
##
## Node number 1: 891 observations,    complexity param=0.4444444
##   predicted class=0  expected loss=0.3838384  P(node) =1
##     class counts:   549   342
##    probabilities: 0.616 0.384
##   left son=2 (577 obs) right son=3 (314 obs)
##   Primary splits:
##       Sex    splits as  RL,           improve=124.426300, (0 missing)
##       Pclass < 2.5      to the right, improve= 43.781830, (0 missing)
##       Fare   < 10.48125 to the left,  improve= 37.941940, (0 missing)
##       Age    < 6.5      to the right, improve=  8.814172, (177 missing)
##   Surrogate splits:
##       Fare < 77.6229  to the left,  agree=0.679, adj=0.089, (0 split)
##
## Node number 2: 577 observations,    complexity param=0.02339181
```

```
##   predicted class=0  expected loss=0.1889081  P(node) =0.647587
##     class counts:   468   109
##    probabilities: 0.811 0.189
##   left son=4 (553 obs) right son=5 (24 obs)
##   Primary splits:
##       Age    < 6.5      to the right, improve=10.78893, (124 missing)
##       Fare   < 26.26875 to the left,  improve=10.21672, (0 missing)
##       Pclass < 1.5      to the right, improve=10.01914, (0 missing)
##
## Node number 3: 314 observations,    complexity param=0.03070175
##   predicted class=1  expected loss=0.2579618  P(node) =0.352413
##     class counts:    81   233
##    probabilities: 0.258 0.742
##   left son=6 (144 obs) right son=7 (170 obs)
##   Primary splits:
##       Pclass < 2.5      to the right, improve=31.163130, (0 missing)
##       Fare   < 48.2     to the left,  improve=10.114210, (0 missing)
##       Age    < 12       to the left,  improve= 1.891684, (53 missing)
##   Surrogate splits:
##       Fare < 25.69795 to the left,  agree=0.799, adj=0.563, (0 split)
##       Age  < 18.5     to the left,  agree=0.564, adj=0.049, (0 split)
##
## Node number 4: 553 observations,    complexity param=0.006578947
##   predicted class=0  expected loss=0.1681736  P(node) =0.620651
##     class counts:   460    93
##    probabilities: 0.832 0.168
##   left son=8 (433 obs) right son=9 (120 obs)
##   Primary splits:
##       Pclass < 1.5      to the right, improve=11.083720, (0 missing)
##       Fare   < 26.26875 to the left,  improve=10.532060, (0 missing)
##       Age    < 24.75    to the left,  improve= 1.235487, (124 missing)
##   Surrogate splits:
##       Fare < 26.26875 to the left,  agree=0.911, adj=0.592, (0 split)
##
## Node number 5: 24 observations,    complexity param=0.005847953
##   predicted class=1  expected loss=0.3333333  P(node) =0.02693603
##     class counts:     8    16
##    probabilities: 0.333 0.667
##   left son=10 (14 obs) right son=11 (10 obs)
##   Primary splits:
##       Pclass < 2.5      to the right, improve=3.8095240, (0 missing)
##       Fare   < 20.825   to the right, improve=2.6666670, (0 missing)
##       Age    < 1.5      to the right, improve=0.6095238, (0 missing)
##   Surrogate splits:
##       Age  < 0.96     to the right, agree=0.708, adj=0.3, (0 split)
##       Fare < 64.37915 to the left,  agree=0.667, adj=0.2, (0 split)
##
## Node number 6: 144 observations,    complexity param=0.03070175
##   predicted class=0  expected loss=0.5  P(node) =0.1616162
##     class counts:    72    72
##    probabilities: 0.500 0.500
##   left son=12 (27 obs) right son=13 (117 obs)
##   Primary splits:
##       Fare < 23.35    to the right, improve=10.051280, (0 missing)
```

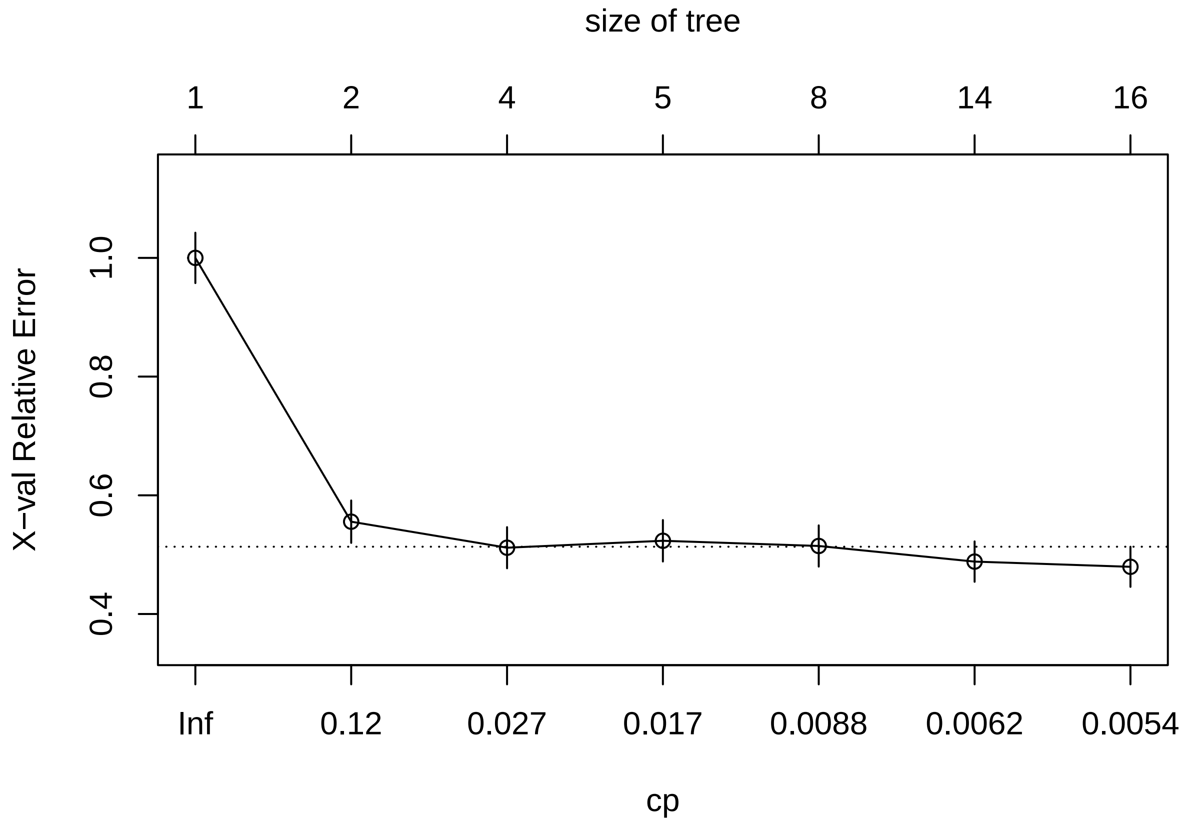
```
##       Age  < 38.5     to the right, improve= 3.875163, (42 missing)
##
## Node number 7: 170 observations
##   predicted class=1  expected loss=0.05294118  P(node) =0.1907969
##     class counts:     9   161
##    probabilities: 0.053 0.947
##
## Node number 8: 433 observations
##   predicted class=0  expected loss=0.1154734  P(node) =0.4859708
##     class counts:   383    50
##    probabilities: 0.885 0.115
##
## Node number 9: 120 observations,    complexity param=0.006578947
##   predicted class=0  expected loss=0.3583333  P(node) =0.1346801
##     class counts:    77    43
##    probabilities: 0.642 0.358
##   left son=18 (22 obs) right son=19 (98 obs)
##   Primary splits:
##       Age  < 53       to the right, improve=3.464646, (21 missing)
##       Fare < 26.14375 to the left,  improve=2.801515, (0 missing)
##
## Node number 10: 14 observations
##   predicted class=0  expected loss=0.4285714  P(node) =0.01571268
##     class counts:     8     6
##    probabilities: 0.571 0.429
##
## Node number 11: 10 observations
##   predicted class=1  expected loss=0  P(node) =0.01122334
##     class counts:     0    10
##    probabilities: 0.000 1.000
##
## Node number 12: 27 observations
##   predicted class=0  expected loss=0.1111111  P(node) =0.03030303
##     class counts:    24     3
##    probabilities: 0.889 0.111
##
## Node number 13: 117 observations,    complexity param=0.01169591
##   predicted class=1  expected loss=0.4102564  P(node) =0.1313131
##     class counts:    48    69
##    probabilities: 0.410 0.590
##   left son=26 (91 obs) right son=27 (26 obs)
##   Primary splits:
##       Age  < 16.5     to the right, improve=2.468587, (34 missing)
##       Fare < 7.8875   to the right, improve=2.032527, (0 missing)
##   Surrogate splits:
##       Fare < 20.8     to the left,  agree=0.747, adj=0.087, (34 split)
##
## Node number 18: 22 observations
##   predicted class=0  expected loss=0.1363636  P(node) =0.02469136
##     class counts:    19     3
##    probabilities: 0.864 0.136
##
## Node number 19: 98 observations,    complexity param=0.006578947
##   predicted class=0  expected loss=0.4081633  P(node) =0.1099888
```

```
##     class counts:    58    40
##    probabilities: 0.592 0.408
##   left son=38 (9 obs) right son=39 (89 obs)
##   Primary splits:
##       Fare < 26.14375 to the left,  improve=3.301995, (0 missing)
##       Age  < 36.5     to the right, improve=1.112992, (21 missing)
##
## Node number 26: 91 observations,    complexity param=0.01169591
##   predicted class=1  expected loss=0.4615385  P(node) =0.1021324
##     class counts:    42    49
##    probabilities: 0.462 0.538
##   left son=52 (54 obs) right son=53 (37 obs)
##   Primary splits:
##       Fare < 7.8875   to the right, improve=3.363902, (0 missing)
##       Age  < 36.5     to the right, improve=1.661815, (31 missing)
##
## Node number 27: 26 observations
##   predicted class=1  expected loss=0.2307692  P(node) =0.0291807
##     class counts:     6    20
##    probabilities: 0.231 0.769
##
## Node number 38: 9 observations
##   predicted class=0  expected loss=0  P(node) =0.01010101
##     class counts:     9     0
##    probabilities: 1.000 0.000
##
## Node number 39: 89 observations,    complexity param=0.006578947
##   predicted class=0  expected loss=0.4494382  P(node) =0.09988777
##     class counts:    49    40
##    probabilities: 0.551 0.449
##   left son=78 (76 obs) right son=79 (13 obs)
##   Primary splits:
##       Fare < 27.1354  to the right, improve=4.7919070, (0 missing)
##       Age  < 43       to the right, improve=0.8888889, (17 missing)
##
## Node number 52: 54 observations,    complexity param=0.01169591
##   predicted class=0  expected loss=0.4259259  P(node) =0.06060606
##     class counts:    31    23
##    probabilities: 0.574 0.426
##   left son=104 (33 obs) right son=105 (21 obs)
##   Primary splits:
##       Fare < 14.8729  to the left,  improve=2.563252, (0 missing)
##       Age  < 23.5     to the left,  improve=1.314848, (11 missing)
##
## Node number 53: 37 observations
##   predicted class=1  expected loss=0.2972973  P(node) =0.04152637
##     class counts:    11    26
##    probabilities: 0.297 0.703
##
## Node number 78: 76 observations,    complexity param=0.006578947
##   predicted class=0  expected loss=0.3815789  P(node) =0.08529742
##     class counts:    47    29
##    probabilities: 0.618 0.382
##   left son=156 (7 obs) right son=157 (69 obs)
```

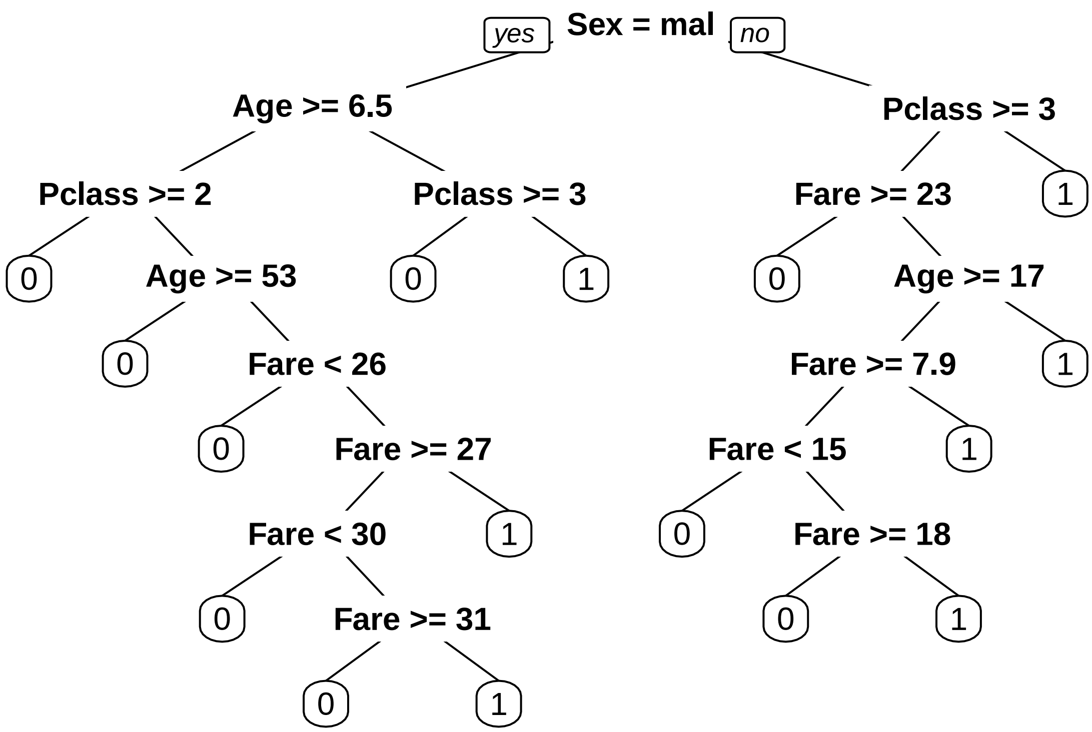
```
##   Primary splits:
##       Fare < 29.85    to the left,  improve=0.8787730, (0 missing)
##       Age  < 43       to the right, improve=0.7179528, (15 missing)
##
## Node number 79: 13 observations
##   predicted class=1  expected loss=0.1538462  P(node) =0.01459035
##     class counts:     2    11
##    probabilities: 0.154 0.846
##
## Node number 104: 33 observations
##   predicted class=0  expected loss=0.3030303  P(node) =0.03703704
##     class counts:    23    10
##    probabilities: 0.697 0.303
##
## Node number 105: 21 observations,    complexity param=0.005847953
##   predicted class=1  expected loss=0.3809524  P(node) =0.02356902
##     class counts:     8    13
##    probabilities: 0.381 0.619
##   left son=210 (8 obs) right son=211 (13 obs)
##   Primary splits:
##       Fare < 17.6     to the right, improve=1.53937700, (0 missing)
##       Age  < 30       to the left,  improve=0.03809524, (6 missing)
##
## Node number 156: 7 observations
##   predicted class=0  expected loss=0.1428571  P(node) =0.007856341
##     class counts:     6     1
##    probabilities: 0.857 0.143
##
## Node number 157: 69 observations,    complexity param=0.006578947
##   predicted class=0  expected loss=0.4057971  P(node) =0.07744108
##     class counts:    41    28
##    probabilities: 0.594 0.406
##   left son=314 (62 obs) right son=315 (7 obs)
##   Primary splits:
##       Fare < 30.5979  to the right, improve=3.1739800, (0 missing)
##       Age  < 43       to the right, improve=0.8596491, (12 missing)
##
## Node number 210: 8 observations
##   predicted class=0  expected loss=0.375  P(node) =0.008978676
##     class counts:     5     3
##    probabilities: 0.625 0.375
##
## Node number 211: 13 observations
##   predicted class=1  expected loss=0.2307692  P(node) =0.01459035
##     class counts:     3    10
##    probabilities: 0.231 0.769
##
## Node number 314: 62 observations
##   predicted class=0  expected loss=0.3548387  P(node) =0.06958474
##     class counts:    40    22
##    probabilities: 0.645 0.355
##
## Node number 315: 7 observations
##   predicted class=1  expected loss=0.1428571  P(node) =0.007856341
```

```
##      class counts:     1     6
##     probabilities: 0.143 0.857
```

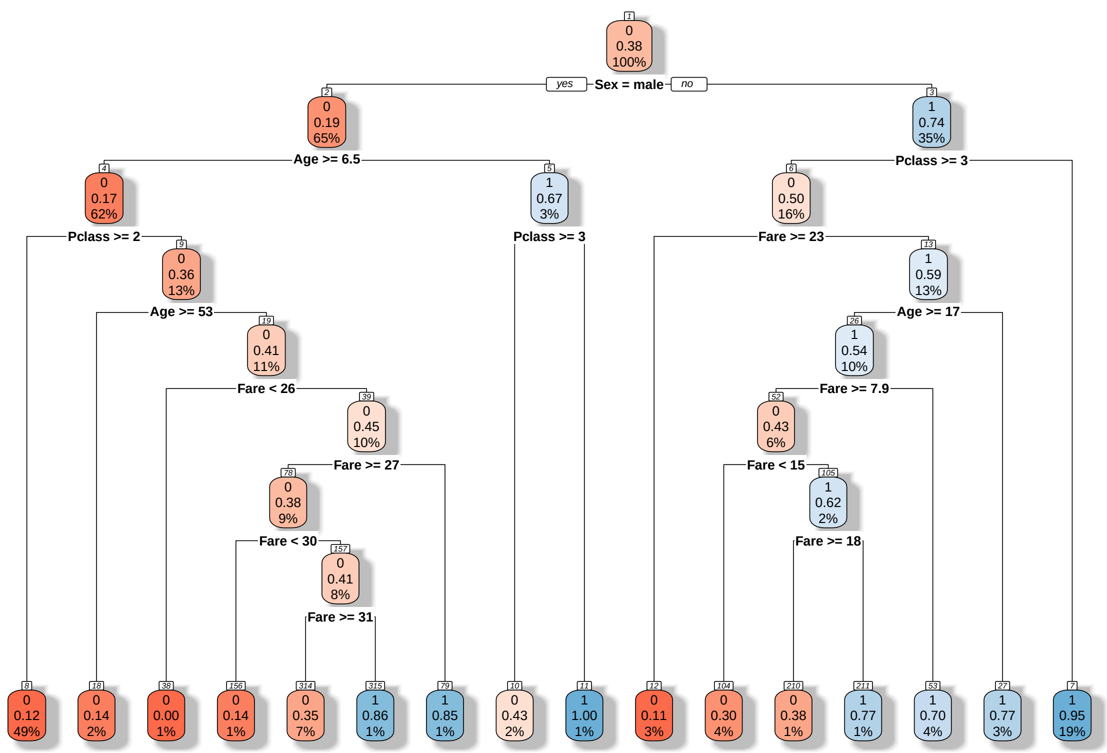
```r
par(cex = 1)
plotcp(treeModel)
```




```r
# Some nicer plots
prp(treeModel)
```




```
rpart.plot(treeModel, box.palette="RdBu", shadow.col="gray", nn=TRUE)
```

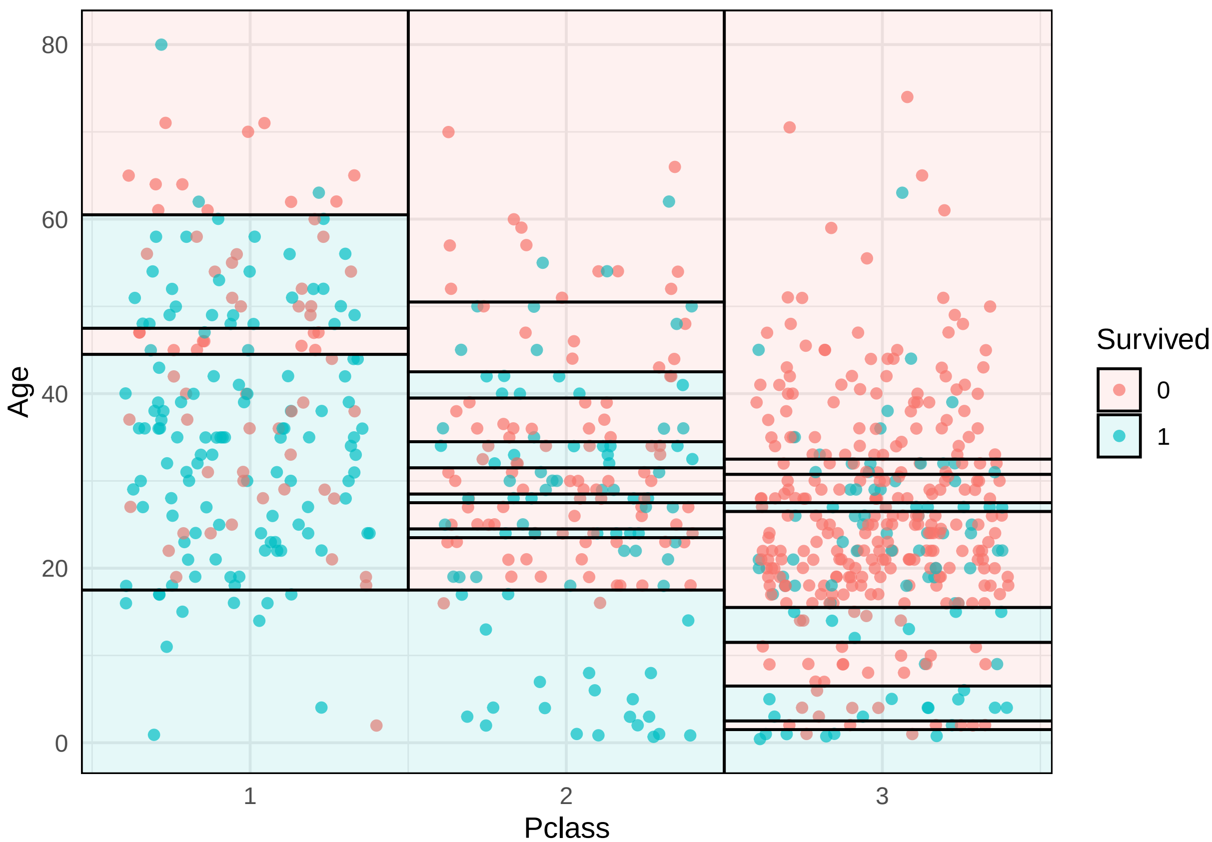


Plotting decision boundaries

```
# Plotting decision boundaries
```

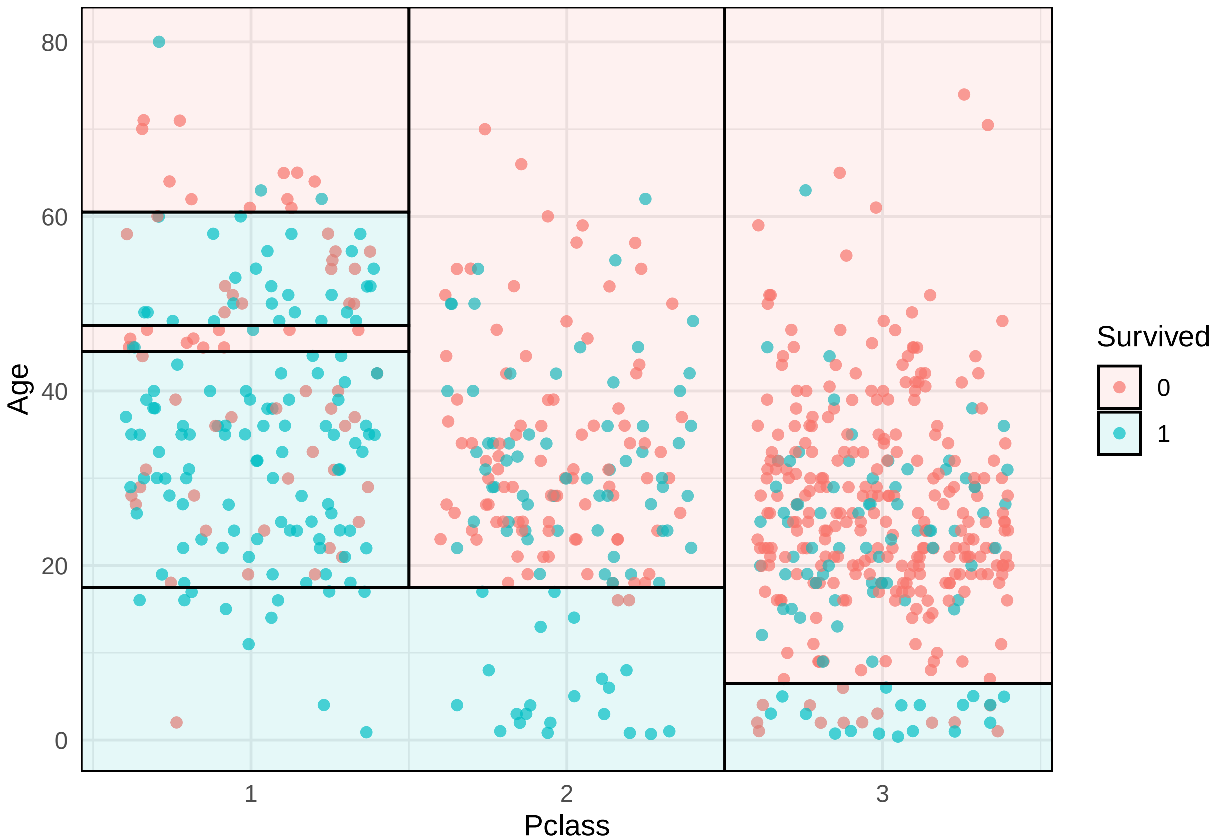
```r
# Create a model with 2 predictors
treeModel = rpart(Survived ~ Pclass + Age, data=titanic_train, method="class", xval=5, cp=.00005)


# Plot decision boundaries
titanic_train %>%
  ggplot(aes(x=Pclass, y=Age)) +
  geom_jitter(aes(col=Survived), alpha=0.7) +
  geom_parttree(data = treeModel, aes(fill=Survived), alpha = 0.1) +
  theme_minimal()
```

```
## Warning: Removed 177 rows containing missing values (geom_point).
```



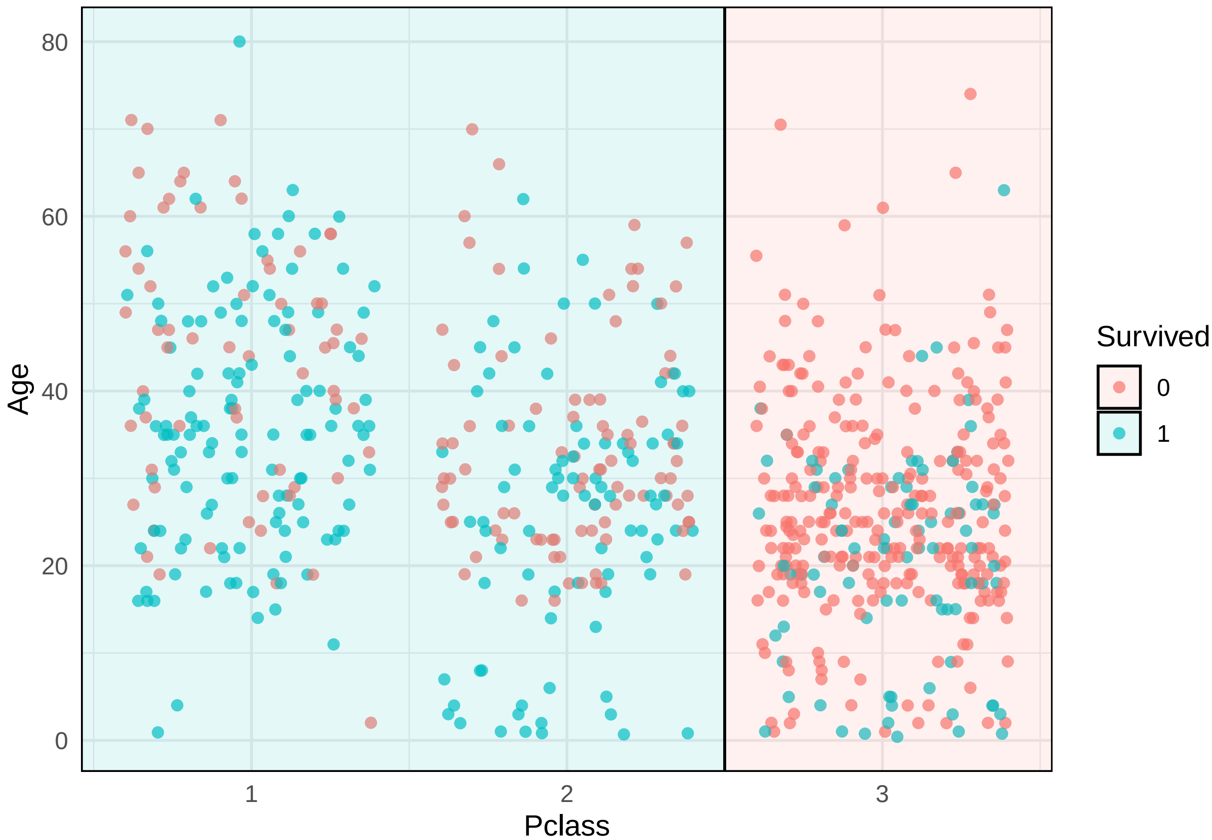

```r
# What happens if we tweak the complexity parameter a.k.a. cp
# cp = 0.01
treeModel = rpart(Survived ~ Pclass + Age, data=titanic_train, method="class", xval=5, cp=.01)

# Plot decision boundaries
titanic_train %>%
  ggplot(aes(x=Pclass, y=Age)) +
  geom_jitter(aes(col=Survived), alpha=0.7) +
  geom_parttree(data = treeModel, aes(fill=Survived), alpha = 0.1) +
  theme_minimal()
```

```
## Warning: Removed 177 rows containing missing values (geom_point).
```



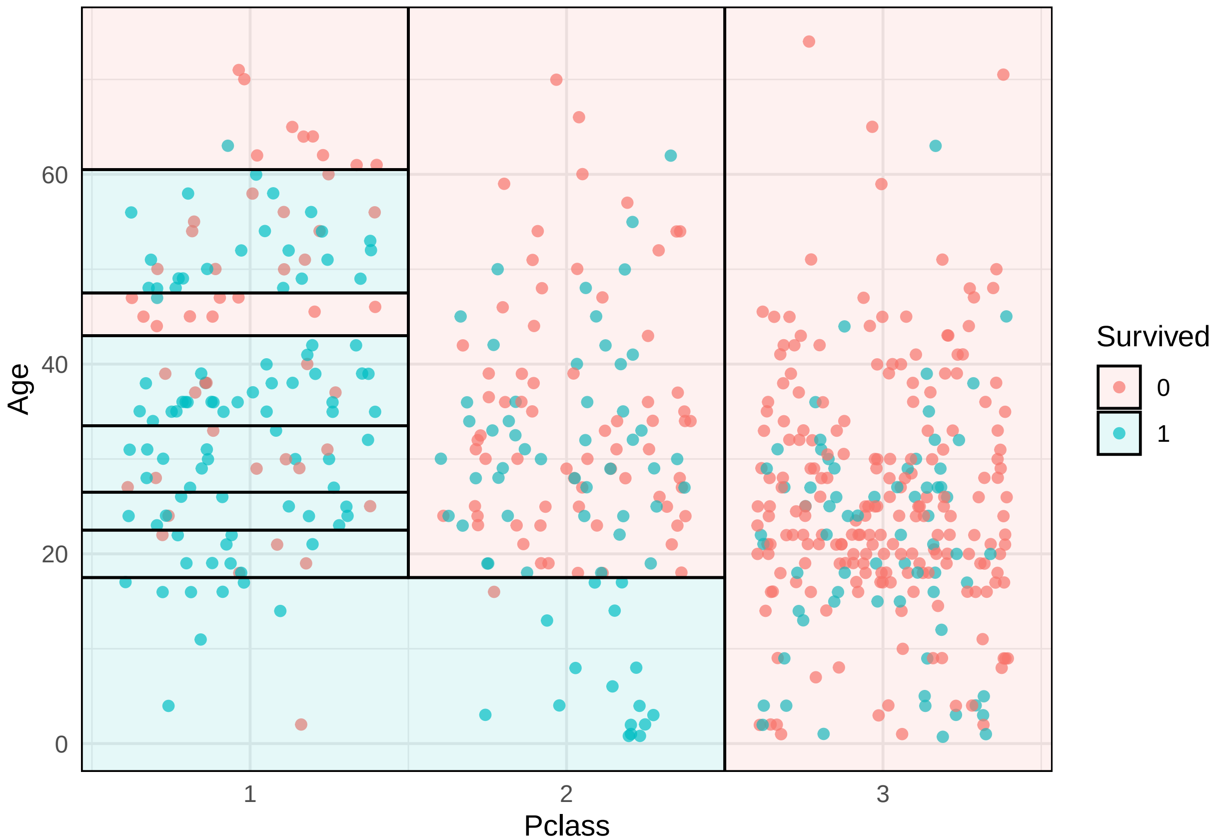

```r
# cp = 0.1
treeModel = rpart(Survived ~ Pclass + Age, data=titanic_train, method="class", xval=5, cp=.1)


# Plot decision boundaries
titanic_train %>%
  ggplot(aes(x=Pclass, y=Age)) +
  geom_jitter(aes(col=Survived), alpha=0.7) +
  geom_parttree(data = treeModel, aes(fill=Survived), alpha = 0.1) +
  theme_minimal()
```

```
## Warning: Removed 177 rows containing missing values (geom_point).
```




```
# Let us split into training and test sets
## 75% of the sample size
train.index <- createDataPartition(titanic_train$Survived, p = .75, list = FALSE)
train <- titanic_train[ train.index,]
test  <- titanic_train[-train.index,]

# Train a model and check performance on the test set
treeModel = rpart(Survived ~ Pclass + Age, data=train, method="class", xval=5, cp=.01)

# Plot decision boundaries
train %>%
  ggplot(aes(x=Pclass, y=Age)) +
  geom_jitter(aes(col=Survived), alpha=0.7) +
  geom_parttree(data = treeModel, aes(fill=Survived), alpha = 0.1) +
  theme_minimal()
```

```
## Warning: Removed 125 rows containing missing values (geom_point).
```




```r
surv_pred <- predict(treeModel, newdata=test, type='class')
surv_pred
```

```
##   5   6  13  14  17  19  20  24  28  29  33  36  39  42  43  46  48  50  51  56
##   0   0   0   0   0   0   0   1   1   0   0   1   0   0   0   0   0   0   0   1
##  57  59  60  66  73  77  84  86  91  94  97  99 101 108 111 128 136 140 142 153
##   0   1   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   1   0   0
## 154 158 162 167 173 176 179 180 181 184 187 188 195 196 200 210 218 219 221 232
##   0   0   0   1   0   0   0   0   0   1   0   0   0   1   0   1   0   1   0   0
## 234 236 237 239 242 243 244 247 257 263 264 265 271 278 282 283 284 306 307 312
##   0   0   0   0   0   0   0   0   1   1   1   0   1   0   0   0   0   1   1   1
## 314 317 324 327 335 336 345 351 353 355 357 361 365 366 370 377 380 383 386 393
##   0   0   0   0   1   0   0   0   0   0   1   0   0   0   1   0   0   0   0   0
## 394 400 403 408 413 419 420 421 426 427 430 433 438 457 459 462 469 475 476 482
##   1   0   0   1   1   0   0   0   0   0   0   0   0   0   0   0   0   0   1   0
## 487 488 503 512 517 518 520 523 524 526 535 536 540 543 549 551 558 562 563 574
##   1   1   0   0   0   0   0   0   0   0   0   1   1   0   0   1   1   0   0   0
## 577 580 584 585 586 588 593 598 599 602 604 610 615 617 618 620 623 624 627 629
##   0   0   1   0   1   1   0   0   0   0   0   1   0   0   0   0   0   0   0   0
## 631 632 634 637 644 645 650 653 661 666 668 681 682 686 688 690 693 696 699 700
##   0   0   1   0   0   0   0   0   1   0   0   0   1   0   0   1   0   0   1   0
## 708 713 719 723 736 738 739 742 745 750 752 756 771 774 775 780 790 792 793 794
##   1   1   0   0   0   1   0   1   0   0   0   1   0   0   0   0   0   1   0   1
## 802 804 810 814 817 825 829 830 833 834 835 837 840 843 844 848 857 868 871 881
##   0   0   1   0   0   0   0   0   0   0   0   0   1   1   0   0   0   1   0   0
## 887 891
##   0   0
```

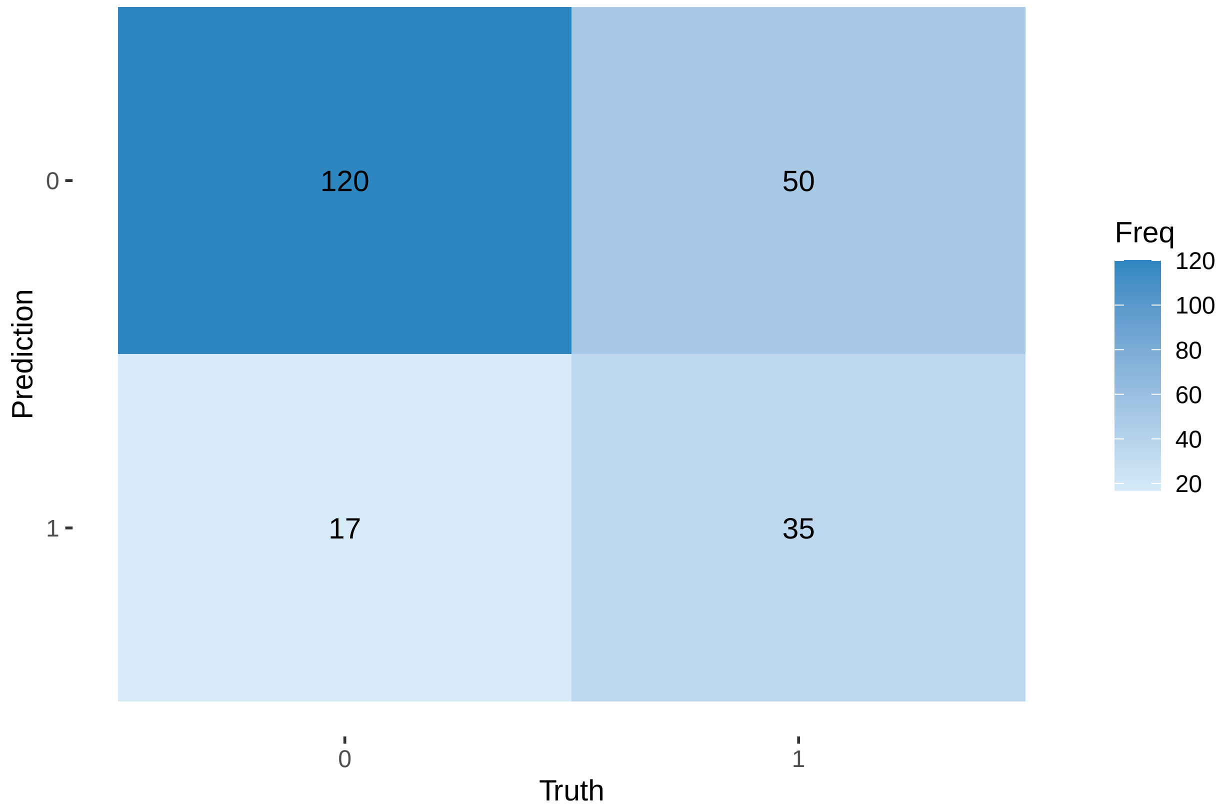
```
## Levels: 0 1
```

```r
confusionMatrix(surv_pred, test$Survived)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 120  50
##          1  17  35
##
##                Accuracy : 0.6982
##                  95% CI : (0.6332, 0.7578)
##     No Information Rate : 0.6171
##     P-Value [Acc > NIR] : 0.007179
##
##                   Kappa : 0.3106
##
##  Mcnemar's Test P-Value : 9.252e-05
##
##             Sensitivity : 0.8759
##             Specificity : 0.4118
##          Pos Pred Value : 0.7059
##          Neg Pred Value : 0.6731
##              Prevalence : 0.6171
##          Detection Rate : 0.5405
##    Detection Prevalence : 0.7658
##       Balanced Accuracy : 0.6438
##
##        'Positive' Class : 0
##
```

```r
# plotting the confusion matrix
trueAndPredFr <- data.frame(surv_pred, test$Survived)
confMat <- conf_mat(trueAndPredFr, truth=test.Survived, estimate=surv_pred)

autoplot(confMat, type = "heatmap") +
  scale_fill_gradient(low="#D6EAF8",high = "#2E86C1") +
  theme(legend.position = "right")
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```




```
# Take home exercise: Can you tune cp using a train and validation set, and then
# test the performance on a test set?
```