Final Project

Mary Buczynski

4/27/2022

```
library(rpart)
library(tree)
```

Insert libraries

```
## Registered S3 method overwritten by 'tree':
##   method     from
##   print.tree cli
```

```
library(randomForest)
```

```
## randomForest 4.7-1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
library(gbm)
```

```
## Loaded gbm 2.1.8
```

```
library(ROCR)
library(readr)
```

```
oscars.df <- read_csv("oscars_df - oscars_df.csv")
```

Read in library

```
## New names:
## * `` -> ...1
```

```
## Rows: 571 Columns: 31
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (17): Film, Oscar.Year, Film.Studio.Producer.s, Award, Movie.Genre, Con...
## dbl  (11): ...1, Year.of.Release, Movie.Time, IMDB.Rating, Tomatometer.Ratin...
## date  (2): Original.Release.Date, Streaming.Release.Date
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
oscars.df.ex.na <- read_csv("oscars_df - oscars_df.csv")
```

```
## New names:
## * `` -> ...1
## Rows: 571 Columns: 31-- Column specification --------------------------------------------------------
## Delimiter: ","
```

```
## chr  (17): Film, Oscar.Year, Film.Studio.Producer.s, Award, Movie.Genre, Con...
## dbl  (11): ...1, Year.of.Release, Movie.Time, IMDB.Rating, Tomatometer.Ratin...
## date  (2): Original.Release.Date, Streaming.Release.Date
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
seed.val<-12345
```

```
oscars.df$Award <- factor(oscars.df$Award)
oscars.df$Consolidated.Genre <- factor(oscars.df$Consolidated.Genre)
oscars.df$Content.Rating <- factor(oscars.df$Content.Rating)
oscars.df$Audience.Status <- factor(oscars.df$Audience.Status)
oscars.df$Tomatometer.Status <- factor(oscars.df$Tomatometer.Status)
```

Make columns factors

```
oscars.df <- subset(oscars.df, select = -c(...1, Oscar.Year, Film.Studio.Producer.s, Movie.Info, Critic
oscars.df.ex.na <- oscars.df[!(is.na(oscars.df$Content.Rating) | oscars.df$Content.Rating ==""), ]
```

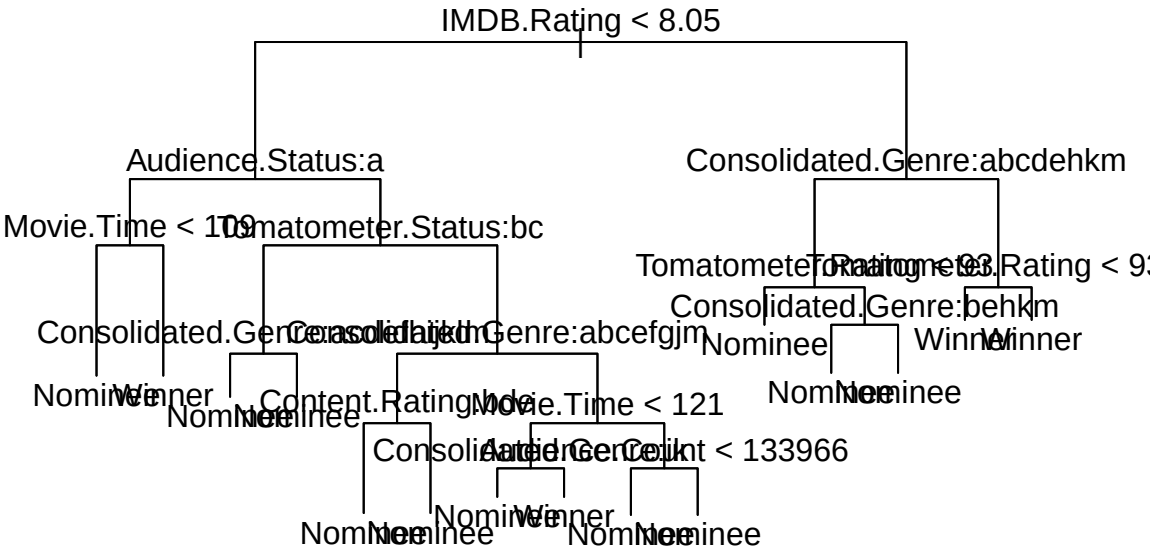
Delete irrelevant/unusable columns and rows

```
set.seed(seed.val)
data.size.tree<-nrow(oscars.df.ex.na)
train.rows.tree<-sample(1:data.size.tree, data.size.tree/2)
train.data.tree<-oscars.df.ex.na[train.rows.tree,]
test.data.tree<-oscars.df.ex.na[-train.rows.tree,]
true.vals.tree<-test.data.tree[, 12]

tree.movies <- tree(Award ~ Movie.Time + Consolidated.Genre + IMDB.Rating + IMDB.Votes + Content.Rating
summary(tree.movies)
```

Single Tree

```
##
## Classification tree:
## tree(formula = Award ~ Movie.Time + Consolidated.Genre + IMDB.Rating +
##     IMDB.Votes + Content.Rating + Tomatometer.Status + Tomatometer.Rating +
##     Audience.Status + Audience.Count, data = train.data.tree)
## Variables actually used in tree construction:
## [1] "IMDB.Rating"        "Audience.Status"    "Movie.Time"
## [4] "Tomatometer.Status" "Consolidated.Genre" "Content.Rating"
## [7] "Audience.Count"     "Tomatometer.Rating"
## Number of terminal nodes:  15
## Residual mean deviance:  0.3357 = 68.14 / 203
## Misclassification error rate: 0.08716 = 19 / 218
```

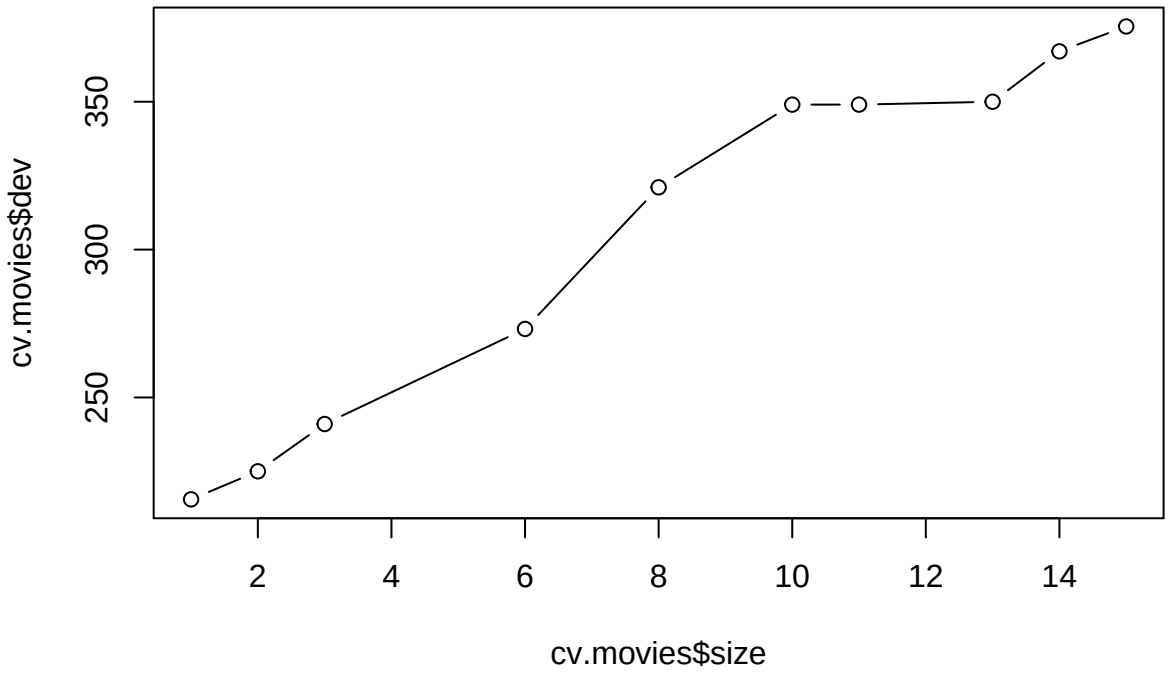
```
plot(tree.movies)
text(tree.movies)
```




```
tree.movies
```

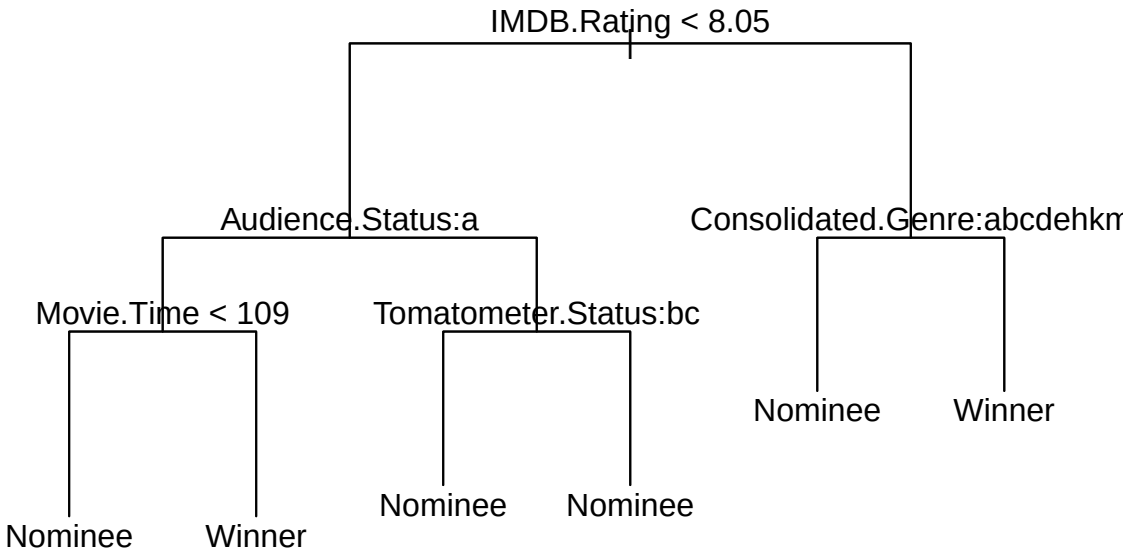
```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 218 192.100 Nominee ( 0.83945 0.16055 )
##    2) IMDB.Rating < 8.05 182 126.000 Nominee ( 0.89011 0.10989 )
##      4) Audience.Status: Spilled 12  16.300 Nominee ( 0.58333 0.41667 )
##        8) Movie.Time < 109 7   0.000 Nominee ( 1.00000 0.00000 ) *
##        9) Movie.Time > 109 5   0.000 Winner ( 0.00000 1.00000 ) *
##      5) Audience.Status: Upright 170 101.500 Nominee ( 0.91176 0.08824 )
##       10) Tomatometer.Status: Fresh,Rotten 82  10.800 Nominee ( 0.98780 0.01220 )
##         20) Consolidated.Genre: Action,Crime,Drama,Fantasy,History,Romance,Romantic Comedy,Sport,Thri
##         21) Consolidated.Genre: Comedy 6   5.407 Nominee ( 0.83333 0.16667 ) *
##       11) Tomatometer.Status: Certified-Fresh 88  77.120 Nominee ( 0.84091 0.15909 )
##         22) Consolidated.Genre: Action,Comedy,Crime,Fantasy,History,Horror,Sport,Western 50  22.700 N
##           44) Content.Rating: NR,PG-13,R 41   0.000 Nominee ( 1.00000 0.00000 ) *
##           45) Content.Rating: G,PG 9  11.460 Nominee ( 0.66667 0.33333 ) *
##         23) Consolidated.Genre: Drama,Romance,Romantic Comedy,Thriller,War 38  45.730 Nominee ( 0.710
##           46) Movie.Time < 121 20  27.530 Nominee ( 0.55000 0.45000 )
##             92) Consolidated.Genre: Romantic Comedy,Thriller 9   9.535 Nominee ( 0.77778 0.22222 ) *
##             93) Consolidated.Genre: Drama,Romance,War 11  14.420 Winner ( 0.36364 0.63636 ) *
##           47) Movie.Time > 121 18  12.560 Nominee ( 0.88889 0.11111 )
##             94) Audience.Count < 133966 13   0.000 Nominee ( 1.00000 0.00000 ) *
##             95) Audience.Count > 133966 5   6.730 Nominee ( 0.60000 0.40000 ) *
##    3) IMDB.Rating > 8.05 36  48.900 Nominee ( 0.58333 0.41667 )
##      6) Consolidated.Genre: Action,Comedy,Crime,Drama,Fantasy,Romance,Thriller,Western 24  24.560 Nom
##       12) Tomatometer.Rating < 93 8   0.000 Nominee ( 1.00000 0.00000 ) *
##       13) Tomatometer.Rating > 93 16  19.870 Nominee ( 0.68750 0.31250 )
##         26) Consolidated.Genre: Comedy,Fantasy,Romance,Thriller,Western 6   0.000 Nominee ( 1.00000 0
##         27) Consolidated.Genre: Action,Crime,Drama 10  13.860 Nominee ( 0.50000 0.50000 ) *
##      7) Consolidated.Genre: History,Romantic Comedy,Sport,War 12  10.810 Winner ( 0.16667 0.83333 )
##       14) Tomatometer.Rating < 93.5 5   6.730 Winner ( 0.40000 0.60000 ) *
##       15) Tomatometer.Rating > 93.5 7   0.000 Winner ( 0.00000 1.00000 ) *
```

```
cv.movies <-cv.tree(tree.movies)
plot(cv.movies$size, cv.movies$dev, type="b")
```



Pruned Single Tree

```
pruned.tree <- prune.tree(tree.movies, best=6)
plot(pruned.tree)
text(pruned.tree)
```

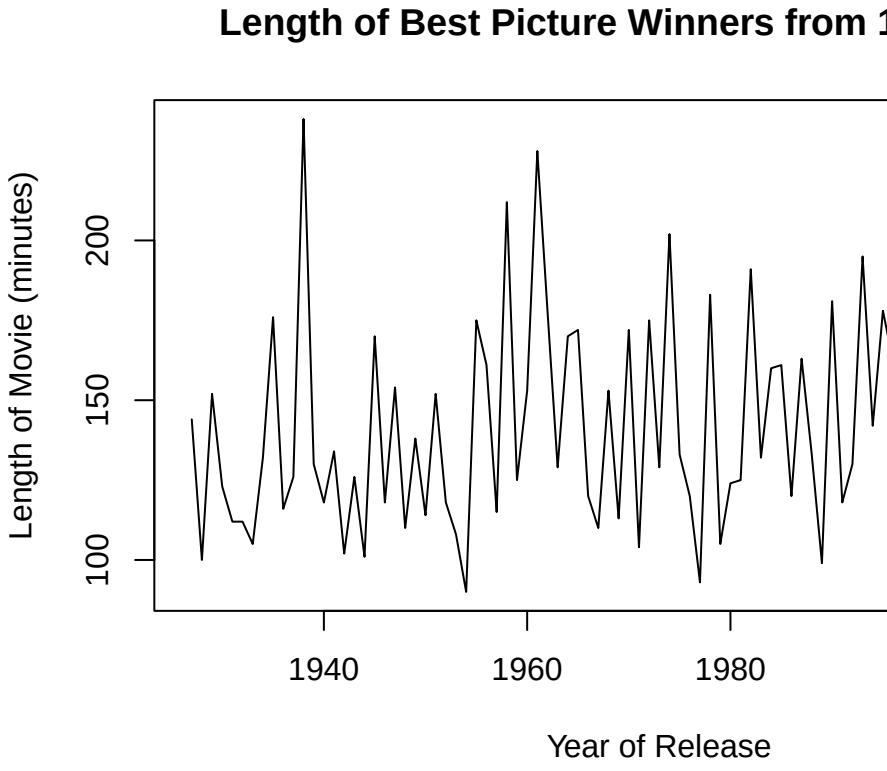



```
pruned.tree
```

```
## node), split, n, deviance, yval, (yprob)
##       * denotes terminal node
##
##  1) root 218 192.10 Nominee ( 0.83945 0.16055 )
##    2) IMDB.Rating < 8.05 182 126.00 Nominee ( 0.89011 0.10989 )
```

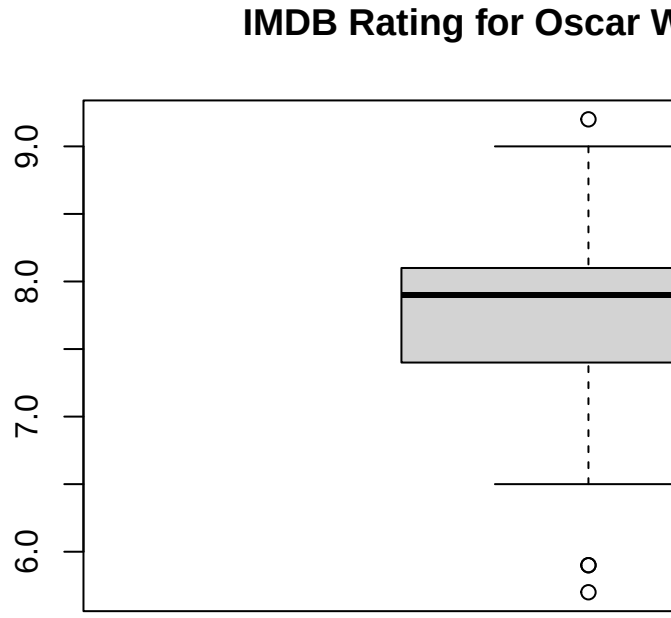
```
##       4) Audience.Status: Spilled 12  16.30 Nominee ( 0.58333 0.41667 )
##         8) Movie.Time < 109 7   0.00 Nominee ( 1.00000 0.00000 ) *
##         9) Movie.Time > 109 5   0.00 Winner ( 0.00000 1.00000 ) *
##       5) Audience.Status: Upright 170 101.50 Nominee ( 0.91176 0.08824 )
##        10) Tomatometer.Status: Fresh,Rotten 82  10.80 Nominee ( 0.98780 0.01220 ) *
##        11) Tomatometer.Status: Certified-Fresh 88  77.12 Nominee ( 0.84091 0.15909 ) *
##     3) IMDB.Rating > 8.05 36  48.90 Nominee ( 0.58333 0.41667 )
##       6) Consolidated.Genre: Action,Comedy,Crime,Drama,Fantasy,Romance,Thriller,Western 24  24.56 Nom
##       7) Consolidated.Genre: History,Romantic Comedy,Sport,War 12  10.81 Winner ( 0.16667 0.83333 ) *
```

```
oscar.winners <- oscars.df[oscars.df$Award == "Winner", ]
movies.vec <- oscar.winners$Movie.Time
movie.ts <- ts(movies.vec, frequency=1, start=c(1927))
movie.ts.plot <- plot(movie.ts, xlab="Year of Release", ylab="Length of Movie (minutes)", main="Length
```



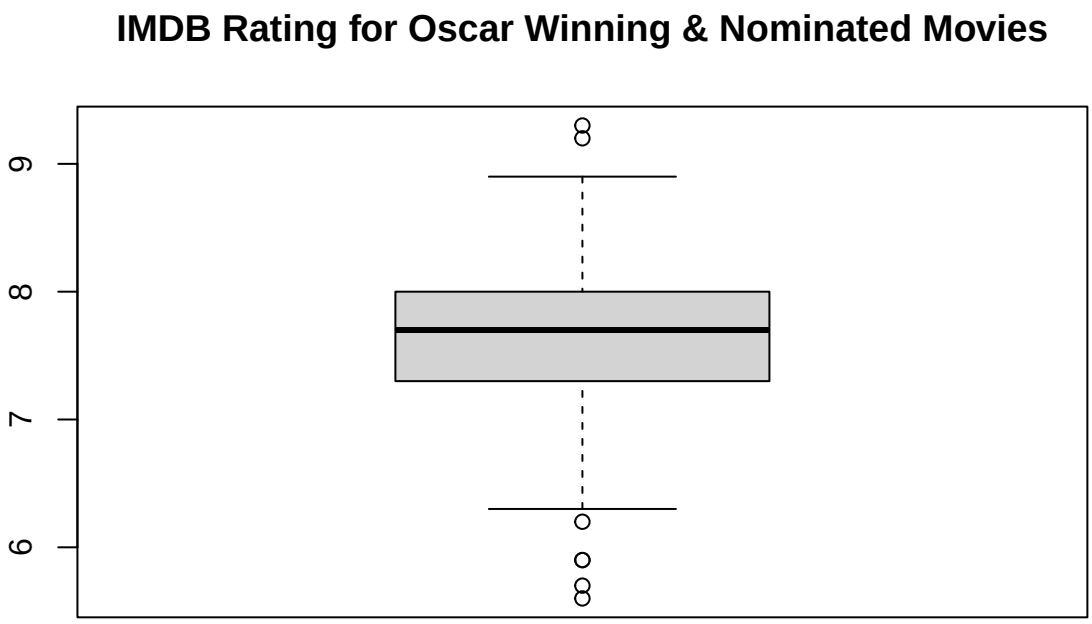
Time Series of winners & length of movie

```
boxplot(oscar.winners$IMDB.Rating, main="IMDB Rating for Oscar Winning Movies")
```

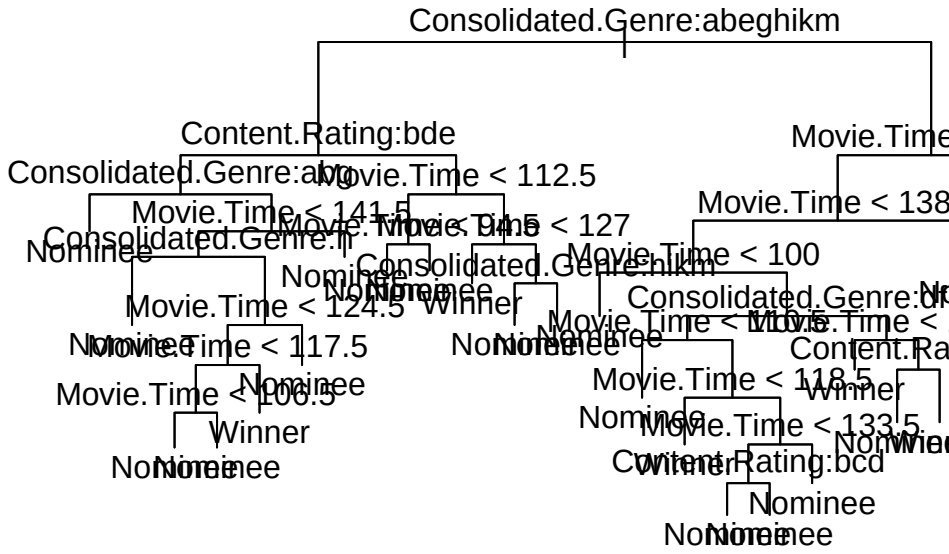


Boxplots of IMDB Ratings for all movies and oscar winners

```
boxplot(oscars.df.ex.na$IMDB.Rating, main="IMDB Rating for Oscar Winning & Nominated Movies")
```

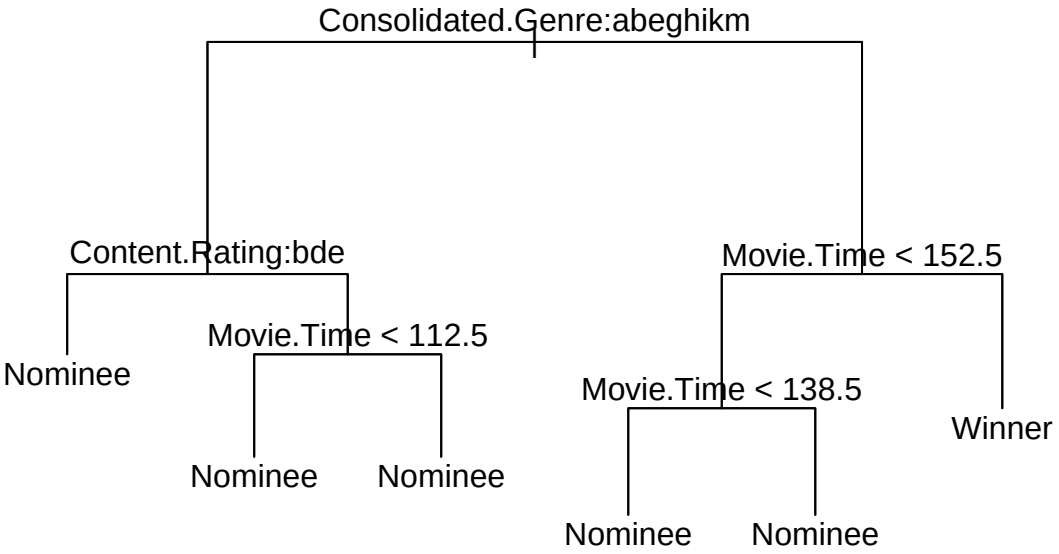



```
movie.attributes.df <- subset(oscars.df.ex.na, select=c("Award", "Movie.Time", "Consolidated.Genre", "C

data.size.rf <- nrow(movie.attributes.df)
train.rows.rf <- sample(1:data.size.rf, data.size.rf/2)
train.data.rf <- movie.attributes.df[train.rows.rf,]
test.data.rf <- movie.attributes.df[-train.rows.rf,]
true.vals.rf <- test.data.rf[,1]

tree.attributes <- tree(Award ~ Movie.Time + Consolidated.Genre + Content.Rating, train.data.rf)
plot(tree.attributes)
text(tree.attributes)
```



Random Forest for Movie Attributes

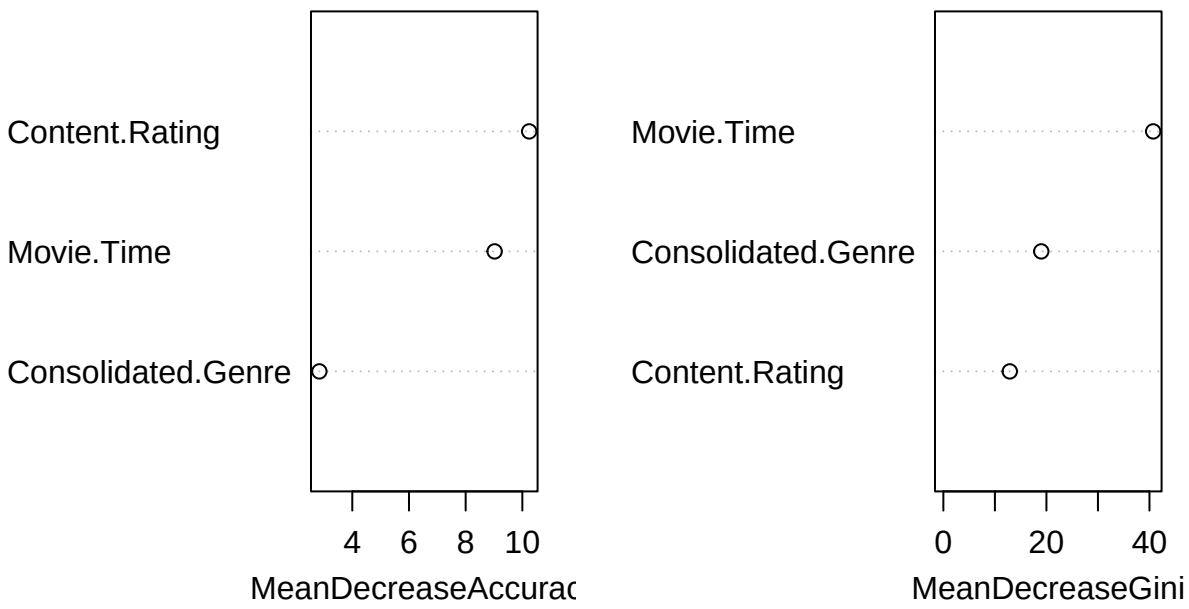
```
pruned.attributes <- prune.tree(tree.attributes, best=5)
plot(pruned.attributes)
text(pruned.attributes)
```




```
rf.movie.att <- randomForest(Award ~., data=train.data.rf, mtry=3, importance=TRUE)
rf.pred <- predict(rf.movie.att, newdata=test.data.rf)
importance(rf.movie.att)
```

```
##                          Nominee     Winner MeanDecreaseAccuracy MeanDecreaseGini
## Movie.Time          11.498037 -0.7239374             9.022379         40.72513
## Consolidated.Genre   7.503193 -7.5810898             2.837966         18.99958
## Content.Rating      12.715622 -1.3241165            10.242196         12.89235
```

```
varImpPlot(rf.movie.att)
```

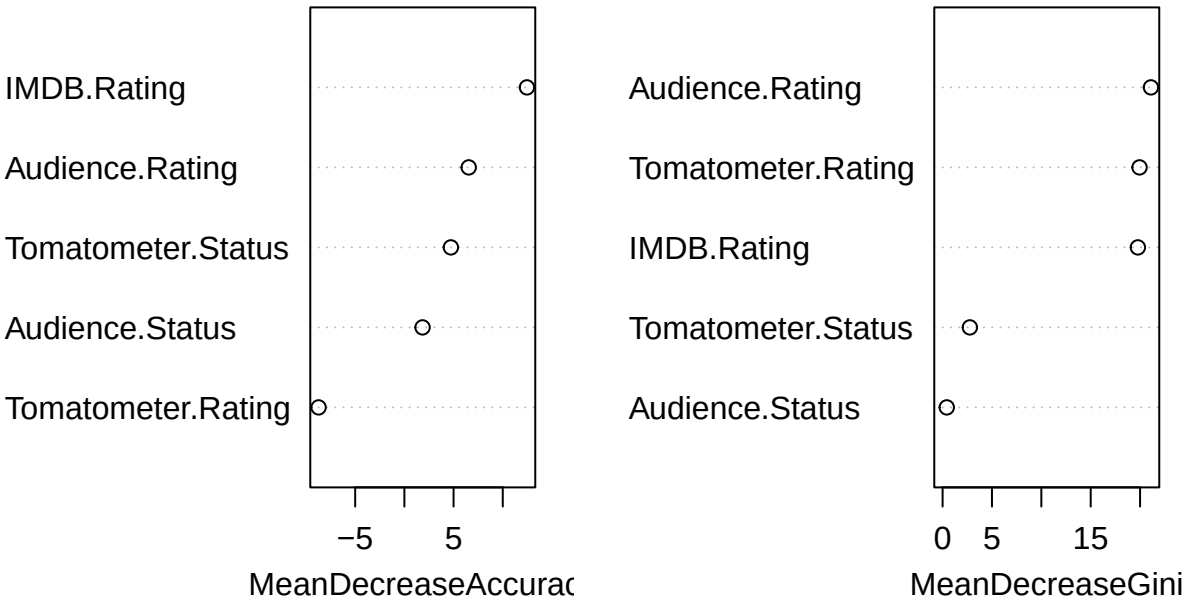



```
movie.ratings.df <- subset(oscars.df.ex.na, select=c("Award", "IMDB.Rating", "Tomatometer.Status", "Toma
movie.ratings.df <- movie.ratings.df[!(is.na(movie.ratings.df$Audience.Status) | movie.ratings.df$Audier

data.size.rf.rate <- nrow(movie.ratings.df)
train.rows.rf.rate <- sample(1:data.size.rf.rate, data.size.rf.rate/2)
train.data.rf.rate <- movie.ratings.df[train.rows.rf.rate,]
test.data.rf.rate <- movie.ratings.df[-train.rows.rf.rate,]
true.vals.rf.rate <- test.data.rf.rate[,1]

rf.movie.rate <- randomForest(Award ~., data=train.data.rf.rate, mtry=5, importance=TRUE)
rf.pred <- predict(rf.movie.rate, newdata=test.data.rf.rate)
importance(rf.movie.rate)
```

Random Forest for Movie Ratings

```
##                          Nominee    Winner MeanDecreaseAccuracy MeanDecreaseGini
## IMDB.Rating         15.30669920 -4.845122            12.457089       19.7777674
## Tomatometer.Status   2.09134808  6.280200             4.727895        2.7610374
## Tomatometer.Rating  -8.10693059 -2.558183            -8.717273       19.9465280
## Audience.Rating      9.87430468 -6.250084             6.541870       21.1006756
## Audience.Status     -0.01608162  3.167674             1.846399        0.4217747
```

```
varImpPlot(rf.movie.rate)
```



Logistic Regression for Ratings Data

```
mod.all.rate <- glm(Award ~ ., family = binomial(logit), data = movie.ratings.df)
summary(mod.all.rate)
```

```
##
## Call:
## glm(formula = Award ~ ., family = binomial(logit), data = movie.ratings.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5658  -0.7375  -0.4072  -0.2862   2.6510
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -3.29880    2.75372  -1.198   0.2309
## IMDB.Rating                0.91754    0.47650   1.926   0.0542 .
## Tomatometer.StatusFresh   -1.65515    0.41471  -3.991 6.58e-05 ***
## Tomatometer.StatusRotten  -0.80166    1.08008  -0.742   0.4580
## Tomatometer.Rating        -0.02215    0.01719  -1.288   0.1977
## Audience.Rating           -0.00793    0.02822  -0.281   0.7787
## Audience.StatusUpright    -2.29637    0.90622  -2.534   0.0113 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 421.54  on 435  degrees of freedom
```

```
## Residual deviance: 378.45  on 429  degrees of freedom
## AIC: 392.45
##
## Number of Fisher Scoring iterations: 5
```

```
mod.step.rate <- step(mod.all.rate)
```

```
## Start:  AIC=392.45
## Award ~ IMDB.Rating + Tomatometer.Status + Tomatometer.Rating +
##     Audience.Rating + Audience.Status
##
##                       Df Deviance    AIC
## - Audience.Rating      1   378.53 390.53
## - Tomatometer.Rating   1   380.09 392.09
## <none>                     378.45 392.45
## - IMDB.Rating          1   382.11 394.11
## - Audience.Status      1   384.76 396.76
## - Tomatometer.Status   2   400.35 410.35
##
## Step:  AIC=390.53
## Award ~ IMDB.Rating + Tomatometer.Status + Tomatometer.Rating +
##     Audience.Status
##
##                       Df Deviance    AIC
## - Tomatometer.Rating   1   380.26 390.26
## <none>                     378.53 390.53
## - IMDB.Rating          1   385.34 395.34
## - Audience.Status      1   389.20 399.20
## - Tomatometer.Status   2   400.41 408.41
##
## Step:  AIC=390.26
## Award ~ IMDB.Rating + Tomatometer.Status + Audience.Status
##
##                       Df Deviance    AIC
## <none>                     380.26 390.26
## - IMDB.Rating          1   386.11 394.11
## - Audience.Status      1   390.98 398.98
## - Tomatometer.Status   2   402.29 408.29
```

```
summary(mod.step.rate)
```

```
##
## Call:
## glm(formula = Award ~ IMDB.Rating + Tomatometer.Status + Audience.Status,
##     family = binomial(logit), data = movie.ratings.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6105  -0.7483  -0.3946  -0.2950   2.6524
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -4.4898     2.2796  -1.970 0.048889 *
## IMDB.Rating                0.7491     0.3131   2.392 0.016738 *
## Tomatometer.StatusFresh   -1.5338     0.3979  -3.855 0.000116 ***
```

```
## Tomatometer.StatusRotten   0.1487     0.8178   0.182 0.855739
## Audience.StatusUpright    -2.4829     0.7193  -3.452 0.000557 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 421.54  on 435  degrees of freedom
## Residual deviance: 380.26  on 431  degrees of freedom
## AIC: 390.26
##
## Number of Fisher Scoring iterations: 5
```

```
mod.all.att <- glm(Award ~ ., family = binomial(logit), data = movie.attributes.df)
summary(mod.all.att)
```

Logistic Regression for Attributes Data

```
##
## Call:
## glm(formula = Award ~ ., family = binomial(logit), data = movie.attributes.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.2679  -0.6685  -0.5343  -0.2920   2.6425
##
## Coefficients:
##                                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)                     -3.644262   1.140827  -3.194  0.00140 **
## Movie.Time                       0.015049   0.005136   2.930  0.00339 **
## Consolidated.GenreComedy        -0.063435   0.875230  -0.072  0.94222
## Consolidated.GenreCrime          0.957539   0.761903   1.257  0.20884
## Consolidated.GenreDrama          0.752212   0.732035   1.028  0.30416
## Consolidated.GenreFantasy       -1.214613   1.226047  -0.991  0.32184
## Consolidated.GenreHistory        0.699464   0.743811   0.940  0.34702
## Consolidated.GenreHorror       -12.174586 622.882086  -0.020  0.98441
## Consolidated.GenreRomance        0.374681   0.773524   0.484  0.62812
## Consolidated.GenreRomantic Comedy  0.444773   0.856938   0.519  0.60374
## Consolidated.GenreSport          1.331617   0.974253   1.367  0.17169
## Consolidated.GenreThriller      -0.661453   1.242265  -0.532  0.59441
## Consolidated.GenreWar            1.392107   0.775977   1.794  0.07281 .
## Consolidated.GenreWestern        0.455693   1.059648   0.430  0.66717
## Content.RatingNR                -0.416297   0.464410  -0.896  0.37004
## Content.RatingPG                -0.210130   0.468228  -0.449  0.65359
## Content.RatingPG-13             -0.502356   0.505566  -0.994  0.32039
## Content.RatingR                 -0.452693   0.432541  -1.047  0.29529
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 422.79  on 438  degrees of freedom
## Residual deviance: 389.40  on 421  degrees of freedom
## AIC: 425.4
```

```
##
## Number of Fisher Scoring iterations: 13
```

```
mod.step.att <- step(mod.all.att)
```

```
## Start:  AIC=425.4
## Award ~ Movie.Time + Consolidated.Genre + Content.Rating
##
##                       Df Deviance    AIC
## - Content.Rating       4   390.98 418.98
## - Consolidated.Genre 12   407.18 419.18
## <none>                     389.40 425.40
## - Movie.Time           1   397.99 431.99
##
## Step:  AIC=418.98
## Award ~ Movie.Time + Consolidated.Genre
##
##                       Df Deviance    AIC
## - Consolidated.Genre 12   408.70 412.70
## <none>                     390.98 418.98
## - Movie.Time           1   400.41 426.41
##
## Step:  AIC=412.7
## Award ~ Movie.Time
##
##              Df Deviance    AIC
## <none>            408.70 412.70
## - Movie.Time  1   422.79 424.79
```

```
summary(mod.step.att)
```

```
##
## Call:
## glm(formula = Award ~ Movie.Time, family = binomial(logit), data = movie.attributes.df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.1984  -0.6571  -0.5719  -0.4845   2.1265
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.674236   0.611369  -6.010 1.86e-09 ***
## Movie.Time   0.016925   0.004488   3.771 0.000163 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 422.79  on 438  degrees of freedom
## Residual deviance: 408.70  on 437  degrees of freedom
## AIC: 412.7
##
## Number of Fisher Scoring iterations: 4
```