Titanic : Machine Learning from disaster

### Introduction

The objective of this project was to complete the analysis of what sorts of people were likely to survive. In particular , the kaggle competition ask's you to apply the tools of machine learning to predict which passengers survived the tragedy.

### First prediction

My first analysis was studing the structure of data and to find out how many passangers have survived and how many have passed away. The table command helped me explore if a variable has any predictive value. The variables that had influence on the survival rate were gender and age. Using these variables i made a simple prediction on the test dataset.

# Structure of training and test set  
str(train)

## 'data.frame': 891 obs. of 12 variables:  
## $ PassengerId: int 1 2 3 4 5 6 7 8 9 10 ...  
## $ Survived : int 0 1 1 1 0 0 0 0 1 1 ...  
## $ Pclass : int 3 1 3 1 3 3 1 3 3 2 ...  
## $ Name : Factor w/ 891 levels "Abbing, Mr. Anthony",..: 109 191 358 277 16 559 520 629 416 581 ...  
## $ Sex : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...  
## $ Age : num 22 38 26 35 35 NA 54 2 27 14 ...  
## $ SibSp : int 1 1 0 1 0 0 0 3 0 1 ...  
## $ Parch : int 0 0 0 0 0 0 0 1 2 0 ...  
## $ Ticket : Factor w/ 681 levels "110152","110413",..: 525 596 662 50 473 276 86 396 345 133 ...  
## $ Fare : num 7.25 71.28 7.92 53.1 8.05 ...  
## $ Cabin : Factor w/ 148 levels "","A10","A14",..: 1 83 1 57 1 1 131 1 1 1 ...  
## $ Embarked : Factor w/ 4 levels "","C","Q","S": 4 2 4 4 4 3 4 4 4 2 ...

str(test)

## 'data.frame': 418 obs. of 11 variables:  
## $ PassengerId: int 892 893 894 895 896 897 898 899 900 901 ...  
## $ Pclass : int 3 3 2 3 3 3 3 2 3 3 ...  
## $ Name : Factor w/ 418 levels "Abbott, Master. Eugene Joseph",..: 210 409 273 414 182 370 85 58 5 104 ...  
## $ Sex : Factor w/ 2 levels "female","male": 2 1 2 2 1 2 1 2 1 2 ...  
## $ Age : num 34.5 47 62 27 22 14 30 26 18 21 ...  
## $ SibSp : int 0 1 0 0 1 0 0 1 0 2 ...  
## $ Parch : int 0 0 0 0 1 0 0 1 0 0 ...  
## $ Ticket : Factor w/ 363 levels "110469","110489",..: 153 222 74 148 139 262 159 85 101 268 ...  
## $ Fare : num 7.83 7 9.69 8.66 12.29 ...  
## $ Cabin : Factor w/ 77 levels "","A11","A18",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ Embarked : Factor w/ 3 levels "C","Q","S": 2 3 2 3 3 3 2 3 1 3 ...

# Passengers that survived vs passengers that passed away  
  
table(train$Survived)

##   
## 0 1   
## 549 342

prop.table(table(train$Survived))

##   
## 0 1   
## 0.6161616 0.3838384

# Males & females that survived vs males & females that passed away  
table(train$Sex, train$Survived)

##   
## 0 1  
## female 81 233  
## male 468 109

prop.table(table(train$Sex, train$Survived), 1)

##   
## 0 1  
## female 0.2579618 0.7420382  
## male 0.8110919 0.1889081

# Create the column child, and indicate whether child or no child  
train$Child <- NA  
train$Child[train$Age < 18] <- 1  
train$Child[train$Age >= 18] <- 0  
  
# Two-way comparison  
table(train$Child, train$Survived)

##   
## 0 1  
## 0 372 229  
## 1 52 61

prop.table(table(train$Child, train$Survived), 1)

##   
## 0 1  
## 0 0.6189684 0.3810316  
## 1 0.4601770 0.5398230

# Prediction based on gender   
test\_one <- test  
test\_one$Survived <- NA  
test\_one$Survived[test\_one$Sex == 'female'] <- 1   
test\_one$Survived[test\_one$Sex == 'male'] <- 0

### Prediction using Decision tree

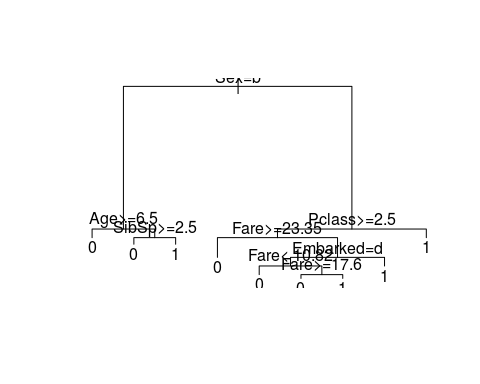
Created a decision tree using rpart function and discovered variables that play an important role whether or not a passenger will survive. Made prediction using the test set and got a result that outperforms a solution using purely gender. To improve the model, manipulated the cp and minisplit in the decision tree.

cp - determines when the splitting up of the decision tree stops.

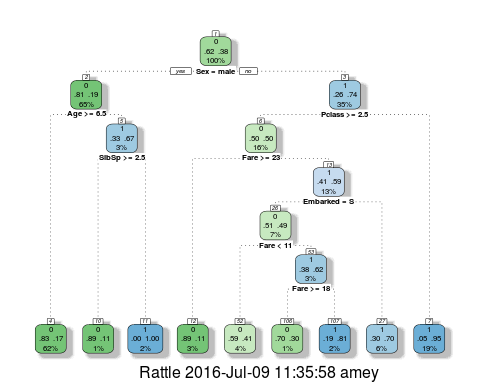
minsplit - determines the minimum amount of observations in a leaf of the tree.

The model genarlizes well compared to previous one.

# Build the decision tree  
my\_tree\_two <- rpart(Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked,   
 data = train, method = "class")  
  
# Visualize the decision tree   
plot(my\_tree\_two)  
text(my\_tree\_two)



# Plot the tree  
fancyRpartPlot(my\_tree\_two)



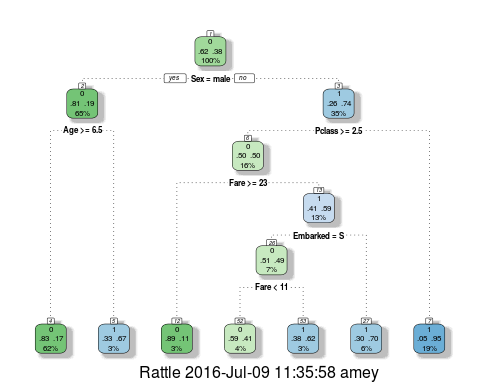
# Make prediction using the test set  
my\_prediction <- predict(my\_tree\_two, test, type="class")  
  
# Create a data frame with two columns: PassengerId & Survived. Survived contains predictions  
my\_solution <- data.frame(PassengerId = test$PassengerId , Survived = my\_prediction)  
  
# Check that data frame has 418 entries  
nrow(my\_solution) == 418

## [1] TRUE

# Write solution to a csv file with the name my\_solution.csv  
write.csv(my\_solution, file = 'my\_solution.csv', row.names = FALSE)  
  
# Create a new decision tree   
my\_tree\_three <- rpart(Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked,   
 data = train, method = "class", control = rpart.control(minsplit = 50, cp = 0))  
 rpart.control(cp = 0, minsplit = 50)

## $minsplit  
## [1] 50  
##   
## $minbucket  
## [1] 17  
##   
## $cp  
## [1] 0  
##   
## $maxcompete  
## [1] 4  
##   
## $maxsurrogate  
## [1] 5  
##   
## $usesurrogate  
## [1] 2  
##   
## $surrogatestyle  
## [1] 0  
##   
## $maxdepth  
## [1] 30  
##   
## $xval  
## [1] 10

# Visualize new decision tree  
fancyRpartPlot(my\_tree\_three)



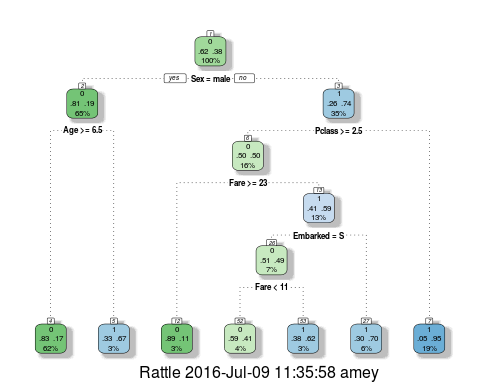
# View my\_solution  
my\_solution

## PassengerId Survived  
## 1 892 0  
## 2 893 0  
## 3 894 0  
## 4 895 0  
## 5 896 1  
## 6 897 0  
## 7 898 1  
## 8 899 0  
## 9 900 1  
## 10 901 0  
## 11 902 0  
## 12 903 0  
## 13 904 1  
## 14 905 0  
## 15 906 1  
## 16 907 1  
## 17 908 0  
## 18 909 0  
## 19 910 0  
## 20 911 1  
## 21 912 0  
## 22 913 0  
## 23 914 1  
## 24 915 0  
## 25 916 1  
## 26 917 0  
## 27 918 1  
## 28 919 0  
## 29 920 0  
## 30 921 0  
## 31 922 0  
## 32 923 0  
## 33 924 0  
## 34 925 0  
## 35 926 0  
## 36 927 0  
## 37 928 0  
## 38 929 0  
## 39 930 0  
## 40 931 0  
## 41 932 0  
## 42 933 0  
## 43 934 0  
## 44 935 1  
## 45 936 1  
## 46 937 0  
## 47 938 0  
## 48 939 0  
## 49 940 1  
## 50 941 1  
## 51 942 0  
## 52 943 0  
## 53 944 1  
## 54 945 1  
## 55 946 0  
## 56 947 0  
## 57 948 0  
## 58 949 0  
## 59 950 0  
## 60 951 1  
## 61 952 0  
## 62 953 0  
## 63 954 0  
## 64 955 1  
## 65 956 0  
## 66 957 1  
## 67 958 1  
## 68 959 0  
## 69 960 0  
## 70 961 1  
## 71 962 1  
## 72 963 0  
## 73 964 0  
## 74 965 0  
## 75 966 1  
## 76 967 0  
## 77 968 0  
## 78 969 1  
## 79 970 0  
## 80 971 1  
## 81 972 1  
## 82 973 0  
## 83 974 0  
## 84 975 0  
## 85 976 0  
## 86 977 0  
## 87 978 1  
## 88 979 0  
## 89 980 1  
## 90 981 1  
## 91 982 1  
## 92 983 0  
## 93 984 1  
## 94 985 0  
## 95 986 0  
## 96 987 0  
## 97 988 1  
## 98 989 0  
## 99 990 0  
## 100 991 0  
## 101 992 1  
## 102 993 0  
## 103 994 0  
## 104 995 0  
## 105 996 1  
## 106 997 0  
## 107 998 0  
## 108 999 0  
## 109 1000 0  
## 110 1001 0  
## 111 1002 0  
## 112 1003 1  
## 113 1004 1  
## 114 1005 1  
## 115 1006 1  
## 116 1007 0  
## 117 1008 0  
## 118 1009 1  
## 119 1010 0  
## 120 1011 1  
## 121 1012 1  
## 122 1013 0  
## 123 1014 1  
## 124 1015 0  
## 125 1016 0  
## 126 1017 1  
## 127 1018 0  
## 128 1019 1  
## 129 1020 0  
## 130 1021 0  
## 131 1022 0  
## 132 1023 0  
## 133 1024 0  
## 134 1025 0  
## 135 1026 0  
## 136 1027 0  
## 137 1028 0  
## 138 1029 0  
## 139 1030 0  
## 140 1031 0  
## 141 1032 0  
## 142 1033 1  
## 143 1034 0  
## 144 1035 0  
## 145 1036 0  
## 146 1037 0  
## 147 1038 0  
## 148 1039 0  
## 149 1040 0  
## 150 1041 0  
## 151 1042 1  
## 152 1043 0  
## 153 1044 0  
## 154 1045 1  
## 155 1046 0  
## 156 1047 0  
## 157 1048 1  
## 158 1049 0  
## 159 1050 0  
## 160 1051 1  
## 161 1052 1  
## 162 1053 0  
## 163 1054 1  
## 164 1055 0  
## 165 1056 0  
## 166 1057 0  
## 167 1058 0  
## 168 1059 0  
## 169 1060 1  
## 170 1061 0  
## 171 1062 0  
## 172 1063 0  
## 173 1064 0  
## 174 1065 0  
## 175 1066 0  
## 176 1067 1  
## 177 1068 1  
## 178 1069 0  
## 179 1070 1  
## 180 1071 1  
## 181 1072 0  
## 182 1073 0  
## 183 1074 1  
## 184 1075 0  
## 185 1076 1  
## 186 1077 0  
## 187 1078 1  
## 188 1079 0  
## 189 1080 0  
## 190 1081 0  
## 191 1082 0  
## 192 1083 0  
## 193 1084 0  
## 194 1085 0  
## 195 1086 0  
## 196 1087 0  
## 197 1088 1  
## 198 1089 0  
## 199 1090 0  
## 200 1091 0  
## 201 1092 1  
## 202 1093 1  
## 203 1094 0  
## 204 1095 1  
## 205 1096 0  
## 206 1097 0  
## 207 1098 1  
## 208 1099 0  
## 209 1100 1  
## 210 1101 0  
## 211 1102 0  
## 212 1103 0  
## 213 1104 0  
## 214 1105 1  
## 215 1106 0  
## 216 1107 0  
## 217 1108 1  
## 218 1109 0  
## 219 1110 1  
## 220 1111 0  
## 221 1112 1  
## 222 1113 0  
## 223 1114 1  
## 224 1115 0  
## 225 1116 1  
## 226 1117 1  
## 227 1118 0  
## 228 1119 1  
## 229 1120 0  
## 230 1121 0  
## 231 1122 0  
## 232 1123 1  
## 233 1124 0  
## 234 1125 0  
## 235 1126 0  
## 236 1127 0  
## 237 1128 0  
## 238 1129 0  
## 239 1130 1  
## 240 1131 1  
## 241 1132 1  
## 242 1133 1  
## 243 1134 0  
## 244 1135 0  
## 245 1136 0  
## 246 1137 0  
## 247 1138 1  
## 248 1139 0  
## 249 1140 1  
## 250 1141 1  
## 251 1142 1  
## 252 1143 0  
## 253 1144 0  
## 254 1145 0  
## 255 1146 0  
## 256 1147 0  
## 257 1148 0  
## 258 1149 0  
## 259 1150 1  
## 260 1151 0  
## 261 1152 0  
## 262 1153 0  
## 263 1154 1  
## 264 1155 1  
## 265 1156 0  
## 266 1157 0  
## 267 1158 0  
## 268 1159 0  
## 269 1160 0  
## 270 1161 0  
## 271 1162 0  
## 272 1163 0  
## 273 1164 1  
## 274 1165 1  
## 275 1166 0  
## 276 1167 1  
## 277 1168 0  
## 278 1169 0  
## 279 1170 0  
## 280 1171 0  
## 281 1172 0  
## 282 1173 1  
## 283 1174 1  
## 284 1175 1  
## 285 1176 0  
## 286 1177 0  
## 287 1178 0  
## 288 1179 0  
## 289 1180 0  
## 290 1181 0  
## 291 1182 0  
## 292 1183 1  
## 293 1184 0  
## 294 1185 0  
## 295 1186 0  
## 296 1187 0  
## 297 1188 1  
## 298 1189 0  
## 299 1190 0  
## 300 1191 0  
## 301 1192 0  
## 302 1193 0  
## 303 1194 0  
## 304 1195 0  
## 305 1196 1  
## 306 1197 1  
## 307 1198 0  
## 308 1199 1  
## 309 1200 0  
## 310 1201 1  
## 311 1202 0  
## 312 1203 0  
## 313 1204 0  
## 314 1205 1  
## 315 1206 1  
## 316 1207 1  
## 317 1208 0  
## 318 1209 0  
## 319 1210 0  
## 320 1211 0  
## 321 1212 0  
## 322 1213 0  
## 323 1214 0  
## 324 1215 0  
## 325 1216 1  
## 326 1217 0  
## 327 1218 1  
## 328 1219 0  
## 329 1220 0  
## 330 1221 0  
## 331 1222 1  
## 332 1223 0  
## 333 1224 0  
## 334 1225 1  
## 335 1226 0  
## 336 1227 0  
## 337 1228 0  
## 338 1229 0  
## 339 1230 0  
## 340 1231 0  
## 341 1232 0  
## 342 1233 0  
## 343 1234 0  
## 344 1235 1  
## 345 1236 0  
## 346 1237 0  
## 347 1238 0  
## 348 1239 1  
## 349 1240 0  
## 350 1241 1  
## 351 1242 1  
## 352 1243 0  
## 353 1244 0  
## 354 1245 0  
## 355 1246 0  
## 356 1247 0  
## 357 1248 1  
## 358 1249 0  
## 359 1250 0  
## 360 1251 1  
## 361 1252 0  
## 362 1253 1  
## 363 1254 1  
## 364 1255 0  
## 365 1256 1  
## 366 1257 0  
## 367 1258 0  
## 368 1259 0  
## 369 1260 1  
## 370 1261 0  
## 371 1262 0  
## 372 1263 1  
## 373 1264 0  
## 374 1265 0  
## 375 1266 1  
## 376 1267 1  
## 377 1268 0  
## 378 1269 0  
## 379 1270 0  
## 380 1271 0  
## 381 1272 0  
## 382 1273 0  
## 383 1274 1  
## 384 1275 1  
## 385 1276 0  
## 386 1277 1  
## 387 1278 0  
## 388 1279 0  
## 389 1280 0  
## 390 1281 0  
## 391 1282 0  
## 392 1283 1  
## 393 1284 0  
## 394 1285 0  
## 395 1286 0  
## 396 1287 1  
## 397 1288 0  
## 398 1289 1  
## 399 1290 0  
## 400 1291 0  
## 401 1292 1  
## 402 1293 0  
## 403 1294 1  
## 404 1295 0  
## 405 1296 0  
## 406 1297 0  
## 407 1298 0  
## 408 1299 0  
## 409 1300 1  
## 410 1301 1  
## 411 1302 1  
## 412 1303 1  
## 413 1304 0  
## 414 1305 0  
## 415 1306 1  
## 416 1307 0  
## 417 1308 0  
## 418 1309 0

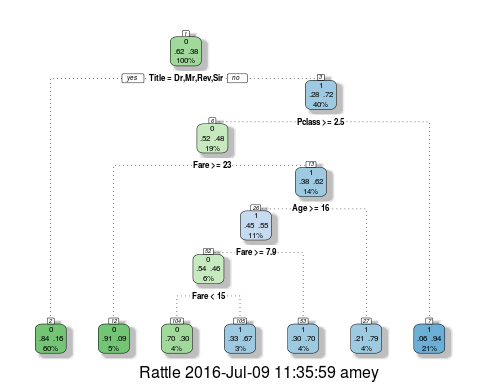
### Improve prediction

To improve prediction, a valid assumption is that larger families need more time to get together on a sinking ship, and hence have less chance of surviving. Family size is determined by the variables SibSp and Parch, which indicate the number of family members a certain passenger is traveling with. So we need to add a new variable family\_size, which is the sum of SibSp and Parch plus one (the observation itself), to the test and train set. In model five another important variable 'Title' is added to the decision tree.

# Create a new train set with the new variable  
train\_two <- train  
train\_two$family\_size <- train$SibSp + train$Parch + 1  
  
# Create a new decision tree   
my\_tree\_four <- rpart(Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked + family\_size, data = train\_two, method = "class", control = rpart.control(minsplit = 50, cp = 0))  
   
# Visualize new decision tree  
fancyRpartPlot(my\_tree\_four)



train\_new = all\_data[1:891,]  
train\_new = subset(train\_new , select = -c(family\_size))  
test\_new = all\_data[892:1309,]  
test\_new = subset(test\_new , select = -c(family\_size))  
  
# Create a new decision tree  
my\_tree\_five <- rpart(Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked + Title,  
 data = train\_new, method = 'class')  
  
# Visualize new decision tree  
fancyRpartPlot(my\_tree\_five)



# Make prediction using `my\_tree\_five` and `test\_new`  
my\_prediction <- my\_prediction <- predict(my\_tree\_five, test\_new, type = 'class')  
  
# Create a data frame with two columns: PassengerId & Survived. Survived contains predictions  
my\_solution2 <- data.frame(PassengerId = test\_new$PassengerId, Survived = my\_prediction)  
  
# Write solution away to a csv file with the name my\_solution.csv  
write.csv(my\_solution2, file = 'my\_solution2.csv', row.names = FALSE)  
  
# View my\_solution2  
my\_solution2

## PassengerId Survived  
## 892 892 0  
## 893 893 1  
## 894 894 0  
## 895 895 0  
## 896 896 0  
## 897 897 0  
## 898 898 1  
## 899 899 0  
## 900 900 1  
## 901 901 0  
## 902 902 0  
## 903 903 0  
## 904 904 1  
## 905 905 0  
## 906 906 1  
## 907 907 1  
## 908 908 0  
## 909 909 0  
## 910 910 0  
## 911 911 1  
## 912 912 0  
## 913 913 1  
## 914 914 1  
## 915 915 0  
## 916 916 1  
## 917 917 0  
## 918 918 1  
## 919 919 0  
## 920 920 0  
## 921 921 0  
## 922 922 0  
## 923 923 0  
## 924 924 1  
## 925 925 0  
## 926 926 0  
## 927 927 0  
## 928 928 0  
## 929 929 0  
## 930 930 0  
## 931 931 0  
## 932 932 0  
## 933 933 0  
## 934 934 0  
## 935 935 1  
## 936 936 1  
## 937 937 0  
## 938 938 0  
## 939 939 0  
## 940 940 1  
## 941 941 1  
## 942 942 0  
## 943 943 0  
## 944 944 1  
## 945 945 1  
## 946 946 0  
## 947 947 0  
## 948 948 0  
## 949 949 0  
## 950 950 0  
## 951 951 1  
## 952 952 0  
## 953 953 0  
## 954 954 0  
## 955 955 1  
## 956 956 1  
## 957 957 1  
## 958 958 1  
## 959 959 0  
## 960 960 0  
## 961 961 1  
## 962 962 1  
## 963 963 0  
## 964 964 0  
## 965 965 0  
## 966 966 1  
## 967 967 0  
## 968 968 0  
## 969 969 1  
## 970 970 0  
## 971 971 1  
## 972 972 1  
## 973 973 0  
## 974 974 0  
## 975 975 0  
## 976 976 0  
## 977 977 0  
## 978 978 1  
## 979 979 0  
## 980 980 1  
## 981 981 1  
## 982 982 0  
## 983 983 0  
## 984 984 1  
## 985 985 0  
## 986 986 0  
## 987 987 0  
## 988 988 1  
## 989 989 0  
## 990 990 1  
## 991 991 0  
## 992 992 1  
## 993 993 0  
## 994 994 0  
## 995 995 0  
## 996 996 1  
## 997 997 0  
## 998 998 0  
## 999 999 0  
## 1000 1000 0  
## 1001 1001 0  
## 1002 1002 0  
## 1003 1003 1  
## 1004 1004 1  
## 1005 1005 1  
## 1006 1006 1  
## 1007 1007 0  
## 1008 1008 0  
## 1009 1009 1  
## 1010 1010 0  
## 1011 1011 1  
## 1012 1012 1  
## 1013 1013 0  
## 1014 1014 1  
## 1015 1015 0  
## 1016 1016 0  
## 1017 1017 1  
## 1018 1018 0  
## 1019 1019 1  
## 1020 1020 0  
## 1021 1021 0  
## 1022 1022 0  
## 1023 1023 1  
## 1024 1024 0  
## 1025 1025 0  
## 1026 1026 0  
## 1027 1027 0  
## 1028 1028 0  
## 1029 1029 0  
## 1030 1030 0  
## 1031 1031 0  
## 1032 1032 0  
## 1033 1033 1  
## 1034 1034 0  
## 1035 1035 0  
## 1036 1036 0  
## 1037 1037 0  
## 1038 1038 0  
## 1039 1039 0  
## 1040 1040 0  
## 1041 1041 0  
## 1042 1042 1  
## 1043 1043 0  
## 1044 1044 0  
## 1045 1045 0  
## 1046 1046 0  
## 1047 1047 0  
## 1048 1048 1  
## 1049 1049 1  
## 1050 1050 0  
## 1051 1051 0  
## 1052 1052 1  
## 1053 1053 1  
## 1054 1054 1  
## 1055 1055 0  
## 1056 1056 0  
## 1057 1057 1  
## 1058 1058 0  
## 1059 1059 0  
## 1060 1060 1  
## 1061 1061 0  
## 1062 1062 0  
## 1063 1063 0  
## 1064 1064 0  
## 1065 1065 0  
## 1066 1066 0  
## 1067 1067 1  
## 1068 1068 1  
## 1069 1069 0  
## 1070 1070 1  
## 1071 1071 1  
## 1072 1072 0  
## 1073 1073 0  
## 1074 1074 1  
## 1075 1075 0  
## 1076 1076 1  
## 1077 1077 0  
## 1078 1078 1  
## 1079 1079 0  
## 1080 1080 0  
## 1081 1081 0  
## 1082 1082 0  
## 1083 1083 0  
## 1084 1084 1  
## 1085 1085 0  
## 1086 1086 1  
## 1087 1087 0  
## 1088 1088 1  
## 1089 1089 1  
## 1090 1090 0  
## 1091 1091 0  
## 1092 1092 1  
## 1093 1093 1  
## 1094 1094 1  
## 1095 1095 1  
## 1096 1096 0  
## 1097 1097 0  
## 1098 1098 1  
## 1099 1099 0  
## 1100 1100 1  
## 1101 1101 0  
## 1102 1102 0  
## 1103 1103 0  
## 1104 1104 0  
## 1105 1105 1  
## 1106 1106 1  
## 1107 1107 0  
## 1108 1108 1  
## 1109 1109 0  
## 1110 1110 1  
## 1111 1111 0  
## 1112 1112 1  
## 1113 1113 0  
## 1114 1114 1  
## 1115 1115 0  
## 1116 1116 1  
## 1117 1117 1  
## 1118 1118 0  
## 1119 1119 1  
## 1120 1120 0  
## 1121 1121 0  
## 1122 1122 0  
## 1123 1123 1  
## 1124 1124 0  
## 1125 1125 0  
## 1126 1126 0  
## 1127 1127 0  
## 1128 1128 0  
## 1129 1129 0  
## 1130 1130 1  
## 1131 1131 1  
## 1132 1132 1  
## 1133 1133 1  
## 1134 1134 0  
## 1135 1135 0  
## 1136 1136 0  
## 1137 1137 0  
## 1138 1138 1  
## 1139 1139 0  
## 1140 1140 1  
## 1141 1141 0  
## 1142 1142 1  
## 1143 1143 0  
## 1144 1144 0  
## 1145 1145 0  
## 1146 1146 0  
## 1147 1147 0  
## 1148 1148 0  
## 1149 1149 0  
## 1150 1150 1  
## 1151 1151 0  
## 1152 1152 0  
## 1153 1153 0  
## 1154 1154 1  
## 1155 1155 1  
## 1156 1156 0  
## 1157 1157 0  
## 1158 1158 0  
## 1159 1159 0  
## 1160 1160 0  
## 1161 1161 0  
## 1162 1162 0  
## 1163 1163 0  
## 1164 1164 1  
## 1165 1165 1  
## 1166 1166 0  
## 1167 1167 1  
## 1168 1168 0  
## 1169 1169 0  
## 1170 1170 0  
## 1171 1171 0  
## 1172 1172 0  
## 1173 1173 1  
## 1174 1174 1  
## 1175 1175 1  
## 1176 1176 1  
## 1177 1177 0  
## 1178 1178 0  
## 1179 1179 0  
## 1180 1180 0  
## 1181 1181 0  
## 1182 1182 0  
## 1183 1183 1  
## 1184 1184 0  
## 1185 1185 0  
## 1186 1186 0  
## 1187 1187 0  
## 1188 1188 1  
## 1189 1189 0  
## 1190 1190 0  
## 1191 1191 0  
## 1192 1192 0  
## 1193 1193 0  
## 1194 1194 0  
## 1195 1195 0  
## 1196 1196 1  
## 1197 1197 1  
## 1198 1198 0  
## 1199 1199 1  
## 1200 1200 0  
## 1201 1201 0  
## 1202 1202 0  
## 1203 1203 0  
## 1204 1204 0  
## 1205 1205 1  
## 1206 1206 1  
## 1207 1207 1  
## 1208 1208 0  
## 1209 1209 0  
## 1210 1210 0  
## 1211 1211 0  
## 1212 1212 0  
## 1213 1213 0  
## 1214 1214 0  
## 1215 1215 0  
## 1216 1216 1  
## 1217 1217 0  
## 1218 1218 1  
## 1219 1219 0  
## 1220 1220 0  
## 1221 1221 0  
## 1222 1222 1  
## 1223 1223 0  
## 1224 1224 0  
## 1225 1225 1  
## 1226 1226 0  
## 1227 1227 0  
## 1228 1228 0  
## 1229 1229 0  
## 1230 1230 0  
## 1231 1231 1  
## 1232 1232 0  
## 1233 1233 0  
## 1234 1234 0  
## 1235 1235 1  
## 1236 1236 1  
## 1237 1237 1  
## 1238 1238 0  
## 1239 1239 1  
## 1240 1240 0  
## 1241 1241 1  
## 1242 1242 1  
## 1243 1243 0  
## 1244 1244 0  
## 1245 1245 0  
## 1246 1246 1  
## 1247 1247 0  
## 1248 1248 1  
## 1249 1249 0  
## 1250 1250 0  
## 1251 1251 1  
## 1252 1252 0  
## 1253 1253 1  
## 1254 1254 1  
## 1255 1255 0  
## 1256 1256 1  
## 1257 1257 0  
## 1258 1258 0  
## 1259 1259 0  
## 1260 1260 1  
## 1261 1261 0  
## 1262 1262 0  
## 1263 1263 1  
## 1264 1264 0  
## 1265 1265 0  
## 1266 1266 1  
## 1267 1267 1  
## 1268 1268 0  
## 1269 1269 0  
## 1270 1270 0  
## 1271 1271 0  
## 1272 1272 0  
## 1273 1273 0  
## 1274 1274 0  
## 1275 1275 1  
## 1276 1276 0  
## 1277 1277 1  
## 1278 1278 0  
## 1279 1279 0  
## 1280 1280 0  
## 1281 1281 1  
## 1282 1282 0  
## 1283 1283 1  
## 1284 1284 1  
## 1285 1285 0  
## 1286 1286 0  
## 1287 1287 1  
## 1288 1288 0  
## 1289 1289 1  
## 1290 1290 0  
## 1291 1291 0  
## 1292 1292 1  
## 1293 1293 0  
## 1294 1294 1  
## 1295 1295 0  
## 1296 1296 0  
## 1297 1297 0  
## 1298 1298 0  
## 1299 1299 0  
## 1300 1300 1  
## 1301 1301 1  
## 1302 1302 1  
## 1303 1303 1  
## 1304 1304 1  
## 1305 1305 0  
## 1306 1306 1  
## 1307 1307 0  
## 1308 1308 0  
## 1309 1309 1

### Random Forest

Random forest technique handles the overfitting problem faced in decision trees. To implement Random Forest all the missing values in the data set should be filled via predicition model.

# All data, both training and test set  
str(all\_data)

## 'data.frame': 1309 obs. of 14 variables:  
## $ PassengerId: int 1 2 3 4 5 6 7 8 9 10 ...  
## $ Survived : int 0 1 1 1 0 0 0 0 1 1 ...  
## $ Pclass : int 3 1 3 1 3 3 1 3 3 2 ...  
## $ Name : chr "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence Briggs Thayer)" "Heikkinen, Miss. Laina" "Futrelle, Mrs. Jacques Heath (Lily May Peel)" ...  
## $ Sex : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...  
## $ Age : num 22 38 26 35 35 NA 54 2 27 14 ...  
## $ SibSp : int 1 1 0 1 0 0 0 3 0 1 ...  
## $ Parch : int 0 0 0 0 0 0 0 1 2 0 ...  
## $ Ticket : Factor w/ 929 levels "110152","110413",..: 524 597 670 50 473 276 86 396 345 133 ...  
## $ Fare : num 7.25 71.28 7.92 53.1 8.05 ...  
## $ Cabin : Factor w/ 187 levels "","A10","A14",..: 1 83 1 57 1 1 131 1 1 1 ...  
## $ Embarked : Factor w/ 4 levels "","C","Q","S": 4 2 4 4 4 3 4 4 4 2 ...  
## $ family\_size: num 2 2 1 2 1 1 1 5 3 2 ...  
## $ Title : Factor w/ 11 levels "Col","Dr","Lady",..: 7 8 5 8 7 7 7 4 8 8 ...

# Passenger on row 62 and 830 do not have a value for embarkment.   
# Since many passengers embarked at Southampton, we give them the value S.  
# Code all embarkment codes as factors.  
all\_data$Embarked[c(62,830)] = "S"  
all\_data$Embarked <- factor(all\_data$Embarked)  
  
# Passenger on row 1044 has an NA Fare value. Replace it with the median fare value.  
all\_data$Fare[1044] <- median(all\_data$Fare, na.rm=TRUE)  
  
# To fill the missing age value  
# Make a prediction of a passengers Age using the other variables and a decision tree model.   
# method="anova" is used since we are predicting a continuous variable.  
predicted\_age <- rpart(Age ~ Pclass + Sex + SibSp + Parch + Fare + Embarked + Title + family\_size,  
 data=all\_data[!is.na(all\_data$Age),], method="anova")  
all\_data$Age[is.na(all\_data$Age)] <- predict(predicted\_age, all\_data[is.na(all\_data$Age),])  
  
# Split the data back into a train set and a test set  
train <- all\_data[1:891,]  
test <- all\_data[892:1309,]  
  
# Set seed for reproducibility  
set.seed(111)  
  
# Apply the Random Forest Algorithm  
my\_forest <- randomForest(as.factor(Survived) ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked + Title,   
 data=train, importance=TRUE, ntree=1000)  
  
# Make prediction using the test set  
my\_prediction <- predict(my\_forest, test)  
  
# Create a data frame with two columns: PassengerId & Survived. Survived contains predictions  
my\_solution3 <- data.frame(PassengerId = test$PassengerId, Survived = my\_prediction)  
  
# Write solution away to a csv file with the name my\_solution.csv  
write.csv(my\_solution3, file = "my\_solution3.csv", row.names = FALSE)  
  
# View my\_solution3  
my\_solution3

## PassengerId Survived  
## 892 892 0  
## 893 893 0  
## 894 894 0  
## 895 895 0  
## 896 896 1  
## 897 897 0  
## 898 898 0  
## 899 899 0  
## 900 900 1  
## 901 901 0  
## 902 902 0  
## 903 903 0  
## 904 904 1  
## 905 905 0  
## 906 906 1  
## 907 907 1  
## 908 908 0  
## 909 909 0  
## 910 910 0  
## 911 911 1  
## 912 912 0  
## 913 913 1  
## 914 914 1  
## 915 915 0  
## 916 916 1  
## 917 917 0  
## 918 918 1  
## 919 919 0  
## 920 920 0  
## 921 921 0  
## 922 922 0  
## 923 923 0  
## 924 924 1  
## 925 925 0  
## 926 926 1  
## 927 927 0  
## 928 928 0  
## 929 929 0  
## 930 930 0  
## 931 931 1  
## 932 932 0  
## 933 933 0  
## 934 934 0  
## 935 935 1  
## 936 936 1  
## 937 937 0  
## 938 938 0  
## 939 939 0  
## 940 940 1  
## 941 941 1  
## 942 942 0  
## 943 943 0  
## 944 944 1  
## 945 945 1  
## 946 946 0  
## 947 947 0  
## 948 948 0  
## 949 949 0  
## 950 950 0  
## 951 951 1  
## 952 952 0  
## 953 953 0  
## 954 954 0  
## 955 955 1  
## 956 956 1  
## 957 957 1  
## 958 958 1  
## 959 959 0  
## 960 960 0  
## 961 961 1  
## 962 962 1  
## 963 963 0  
## 964 964 0  
## 965 965 0  
## 966 966 1  
## 967 967 0  
## 968 968 0  
## 969 969 1  
## 970 970 0  
## 971 971 1  
## 972 972 1  
## 973 973 0  
## 974 974 0  
## 975 975 0  
## 976 976 0  
## 977 977 0  
## 978 978 1  
## 979 979 0  
## 980 980 1  
## 981 981 1  
## 982 982 1  
## 983 983 0  
## 984 984 1  
## 985 985 0  
## 986 986 0  
## 987 987 0  
## 988 988 1  
## 989 989 0  
## 990 990 0  
## 991 991 0  
## 992 992 1  
## 993 993 0  
## 994 994 0  
## 995 995 0  
## 996 996 1  
## 997 997 0  
## 998 998 0  
## 999 999 0  
## 1000 1000 0  
## 1001 1001 0  
## 1002 1002 0  
## 1003 1003 1  
## 1004 1004 1  
## 1005 1005 1  
## 1006 1006 1  
## 1007 1007 0  
## 1008 1008 0  
## 1009 1009 1  
## 1010 1010 0  
## 1011 1011 1  
## 1012 1012 1  
## 1013 1013 0  
## 1014 1014 1  
## 1015 1015 0  
## 1016 1016 0  
## 1017 1017 1  
## 1018 1018 0  
## 1019 1019 1  
## 1020 1020 0  
## 1021 1021 0  
## 1022 1022 0  
## 1023 1023 0  
## 1024 1024 0  
## 1025 1025 0  
## 1026 1026 0  
## 1027 1027 0  
## 1028 1028 0  
## 1029 1029 0  
## 1030 1030 0  
## 1031 1031 0  
## 1032 1032 0  
## 1033 1033 1  
## 1034 1034 0  
## 1035 1035 0  
## 1036 1036 0  
## 1037 1037 0  
## 1038 1038 0  
## 1039 1039 0  
## 1040 1040 0  
## 1041 1041 0  
## 1042 1042 1  
## 1043 1043 0  
## 1044 1044 0  
## 1045 1045 1  
## 1046 1046 0  
## 1047 1047 0  
## 1048 1048 1  
## 1049 1049 0  
## 1050 1050 0  
## 1051 1051 1  
## 1052 1052 1  
## 1053 1053 1  
## 1054 1054 1  
## 1055 1055 0  
## 1056 1056 0  
## 1057 1057 1  
## 1058 1058 0  
## 1059 1059 0  
## 1060 1060 1  
## 1061 1061 0  
## 1062 1062 0  
## 1063 1063 0  
## 1064 1064 0  
## 1065 1065 0  
## 1066 1066 0  
## 1067 1067 1  
## 1068 1068 1  
## 1069 1069 0  
## 1070 1070 1  
## 1071 1071 1  
## 1072 1072 0  
## 1073 1073 0  
## 1074 1074 1  
## 1075 1075 0  
## 1076 1076 1  
## 1077 1077 0  
## 1078 1078 1  
## 1079 1079 0  
## 1080 1080 0  
## 1081 1081 0  
## 1082 1082 0  
## 1083 1083 0  
## 1084 1084 1  
## 1085 1085 0  
## 1086 1086 1  
## 1087 1087 0  
## 1088 1088 1  
## 1089 1089 1  
## 1090 1090 0  
## 1091 1091 0  
## 1092 1092 1  
## 1093 1093 1  
## 1094 1094 1  
## 1095 1095 1  
## 1096 1096 0  
## 1097 1097 0  
## 1098 1098 1  
## 1099 1099 0  
## 1100 1100 1  
## 1101 1101 0  
## 1102 1102 0  
## 1103 1103 0  
## 1104 1104 0  
## 1105 1105 1  
## 1106 1106 0  
## 1107 1107 0  
## 1108 1108 1  
## 1109 1109 0  
## 1110 1110 1  
## 1111 1111 0  
## 1112 1112 1  
## 1113 1113 0  
## 1114 1114 1  
## 1115 1115 0  
## 1116 1116 1  
## 1117 1117 1  
## 1118 1118 0  
## 1119 1119 1  
## 1120 1120 0  
## 1121 1121 0  
## 1122 1122 0  
## 1123 1123 1  
## 1124 1124 0  
## 1125 1125 0  
## 1126 1126 0  
## 1127 1127 0  
## 1128 1128 0  
## 1129 1129 0  
## 1130 1130 1  
## 1131 1131 1  
## 1132 1132 1  
## 1133 1133 1  
## 1134 1134 0  
## 1135 1135 0  
## 1136 1136 1  
## 1137 1137 0  
## 1138 1138 1  
## 1139 1139 0  
## 1140 1140 1  
## 1141 1141 0  
## 1142 1142 1  
## 1143 1143 0  
## 1144 1144 0  
## 1145 1145 0  
## 1146 1146 0  
## 1147 1147 0  
## 1148 1148 0  
## 1149 1149 0  
## 1150 1150 1  
## 1151 1151 0  
## 1152 1152 0  
## 1153 1153 0  
## 1154 1154 1  
## 1155 1155 1  
## 1156 1156 0  
## 1157 1157 0  
## 1158 1158 0  
## 1159 1159 0  
## 1160 1160 0  
## 1161 1161 0  
## 1162 1162 0  
## 1163 1163 0  
## 1164 1164 1  
## 1165 1165 1  
## 1166 1166 0  
## 1167 1167 1  
## 1168 1168 0  
## 1169 1169 0  
## 1170 1170 0  
## 1171 1171 0  
## 1172 1172 0  
## 1173 1173 1  
## 1174 1174 1  
## 1175 1175 0  
## 1176 1176 1  
## 1177 1177 0  
## 1178 1178 0  
## 1179 1179 0  
## 1180 1180 0  
## 1181 1181 0  
## 1182 1182 0  
## 1183 1183 0  
## 1184 1184 0  
## 1185 1185 0  
## 1186 1186 0  
## 1187 1187 0  
## 1188 1188 1  
## 1189 1189 0  
## 1190 1190 0  
## 1191 1191 0  
## 1192 1192 0  
## 1193 1193 0  
## 1194 1194 0  
## 1195 1195 0  
## 1196 1196 1  
## 1197 1197 1  
## 1198 1198 0  
## 1199 1199 1  
## 1200 1200 0  
## 1201 1201 0  
## 1202 1202 0  
## 1203 1203 0  
## 1204 1204 0  
## 1205 1205 0  
## 1206 1206 1  
## 1207 1207 1  
## 1208 1208 0  
## 1209 1209 0  
## 1210 1210 0  
## 1211 1211 0  
## 1212 1212 0  
## 1213 1213 0  
## 1214 1214 0  
## 1215 1215 1  
## 1216 1216 1  
## 1217 1217 0  
## 1218 1218 1  
## 1219 1219 0  
## 1220 1220 0  
## 1221 1221 0  
## 1222 1222 1  
## 1223 1223 0  
## 1224 1224 0  
## 1225 1225 1  
## 1226 1226 0  
## 1227 1227 0  
## 1228 1228 0  
## 1229 1229 0  
## 1230 1230 0  
## 1231 1231 1  
## 1232 1232 0  
## 1233 1233 0  
## 1234 1234 0  
## 1235 1235 1  
## 1236 1236 1  
## 1237 1237 1  
## 1238 1238 0  
## 1239 1239 1  
## 1240 1240 0  
## 1241 1241 1  
## 1242 1242 1  
## 1243 1243 0  
## 1244 1244 0  
## 1245 1245 0  
## 1246 1246 1  
## 1247 1247 0  
## 1248 1248 1  
## 1249 1249 0  
## 1250 1250 0  
## 1251 1251 1  
## 1252 1252 0  
## 1253 1253 1  
## 1254 1254 1  
## 1255 1255 0  
## 1256 1256 1  
## 1257 1257 0  
## 1258 1258 0  
## 1259 1259 0  
## 1260 1260 1  
## 1261 1261 0  
## 1262 1262 0  
## 1263 1263 1  
## 1264 1264 0  
## 1265 1265 0  
## 1266 1266 1  
## 1267 1267 1  
## 1268 1268 0  
## 1269 1269 0  
## 1270 1270 0  
## 1271 1271 0  
## 1272 1272 0  
## 1273 1273 0  
## 1274 1274 1  
## 1275 1275 1  
## 1276 1276 0  
## 1277 1277 1  
## 1278 1278 0  
## 1279 1279 0  
## 1280 1280 0  
## 1281 1281 0  
## 1282 1282 0  
## 1283 1283 1  
## 1284 1284 1  
## 1285 1285 0  
## 1286 1286 0  
## 1287 1287 1  
## 1288 1288 0  
## 1289 1289 1  
## 1290 1290 0  
## 1291 1291 0  
## 1292 1292 1  
## 1293 1293 0  
## 1294 1294 1  
## 1295 1295 0  
## 1296 1296 0  
## 1297 1297 0  
## 1298 1298 0  
## 1299 1299 0  
## 1300 1300 1  
## 1301 1301 1  
## 1302 1302 1  
## 1303 1303 1  
## 1304 1304 0  
## 1305 1305 0  
## 1306 1306 1  
## 1307 1307 0  
## 1308 1308 0  
## 1309 1309 1