Lose It! Code

The following code was used for the analysis of the Lose It! data. Note that the rpart, rattle, and pROC packages were used.

Here is the overview paper of the rpart package:

http://cran.r-project.org/web/packages/rpart/vignettes/longintro.pdf

```
library(rpart)
library(rattle)
```

```
## Rattle: A free graphical interface for data mining with R.
## Version 3.4.1 Copyright (c) 2006-2014 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
##
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

To subset the data once the data are read in:

```
a <- subset(lose1, setupage>=18 & setupage<=70)
b <- subset(a, LastActiveAge>=18)
c <- subset(b, startbmi>=25)
d <- subset(c, weightChange>=0)
```

```
# Rename the clean dataset.
lose1_overagefw <- d
```

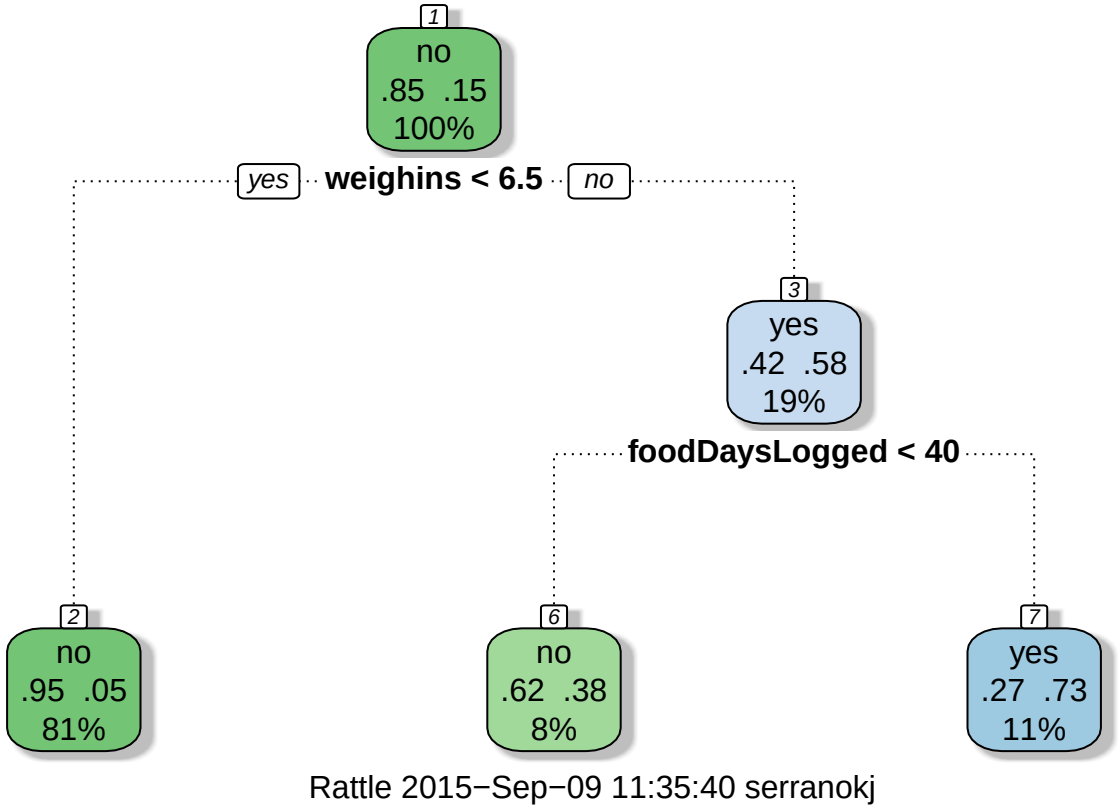
Now, we run rpart using the clean dataset. Note that our dependent variable is outcome5new, a dichotomous variable indicating whether a user was successful at losing 5% or more of his/her starting weight.

```
# Run rpart with the clean dataset.
wtloss_1 <- rpart(outcome5new ~ LastActiveAge + gender + foodDaysLogged
                  + exerciseDaysLogged + percDaysExercise + foodCaloriesLogged
                  + foodCaloriesLoggedyn + foodcaloriesAverage + exerciseCaloriesLogged
                  + exerciseCaloriesLoggedyn + exerciseCaloriesAverage + weighins
                  + startWeight + startbmi + daysSinceCreated + setupage + AetnaCarePass
                  + Facebook + Fitbit + FitbitAria + JawboneUp + Loseit_Scale + MapMyRun
                  + MisfitShine + NikeFuel + NikePlus + RunKeeper + Strava + Twitter
                  + Walgreens + user_android + user_ios + user_web + friendCount
                  + friendcountyn + groupCount + groupcountyn + challengeCount
                  + challengecountyn + usesReminders + customGoalCount
                  + customGoalCountyn + customFoodCount + customFoodCountyn
```

```
                + recipeCount + recipeCountyn + customExerciseCount
                + customExerciseCountyn + hasPicture + challengeAdmin
                + challengeAdminyn + usesEmailReports + goalWeight + goalplanb
                + device , lose1_overagefw, method="class")
```

To plot this tree, we use rattle and the following command:

```
fancyRpartPlot(wtloss_1)
```



Now, we can obtain the predicted responses from this rpart object, which we will use to evaluate the area under the ROC curve in subsample 2. Note that subsample 2 (called lose2_overagefw) was subsetted the same way as subsample 1.

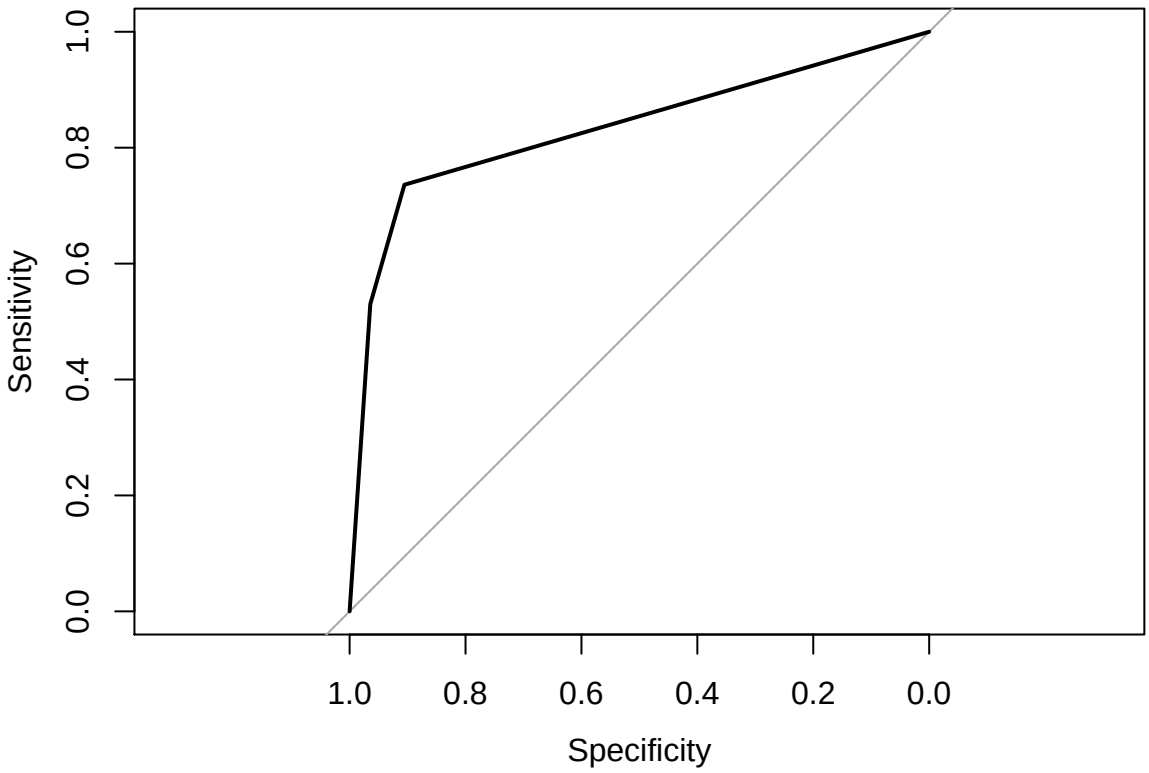
```
# We want only the probabilities for those categorized as yes, so we specify [,2].
pred2<-predict(wtloss_1, lose2_overagefw)[,2]
```

```
# Now, we use the predicted responses to build an ROC curve with subsample 2.
roc_test <- roc(lose2_overagefw$outcome5new, pred2)
```

```
# We then compute the area under the ROC curve.
a_test <- auc(roc_test)
a_test
```

```
## Area under the curve: 0.8327
```

```
# We also can plot the area under the ROC curve.
plot(roc_test)
```




```
##
## Call:
## roc.default(response = lose2_overagefw$outcome5new, predictor = pred2)
##
## Data: pred2 in 275500 controls (lose2_overagefw$outcome5new no) < 48563 cases (lose2_overagefw$outcor
## Area under the curve: 0.8327
```

We then evaluate the results from the tree we obtained with subsample 1 with subsample 3. We repeat the steps above with subsample 3. Note that subsample 3 (called lose5_overagefw) was subsetted the same way as subsamples 1 and 2.

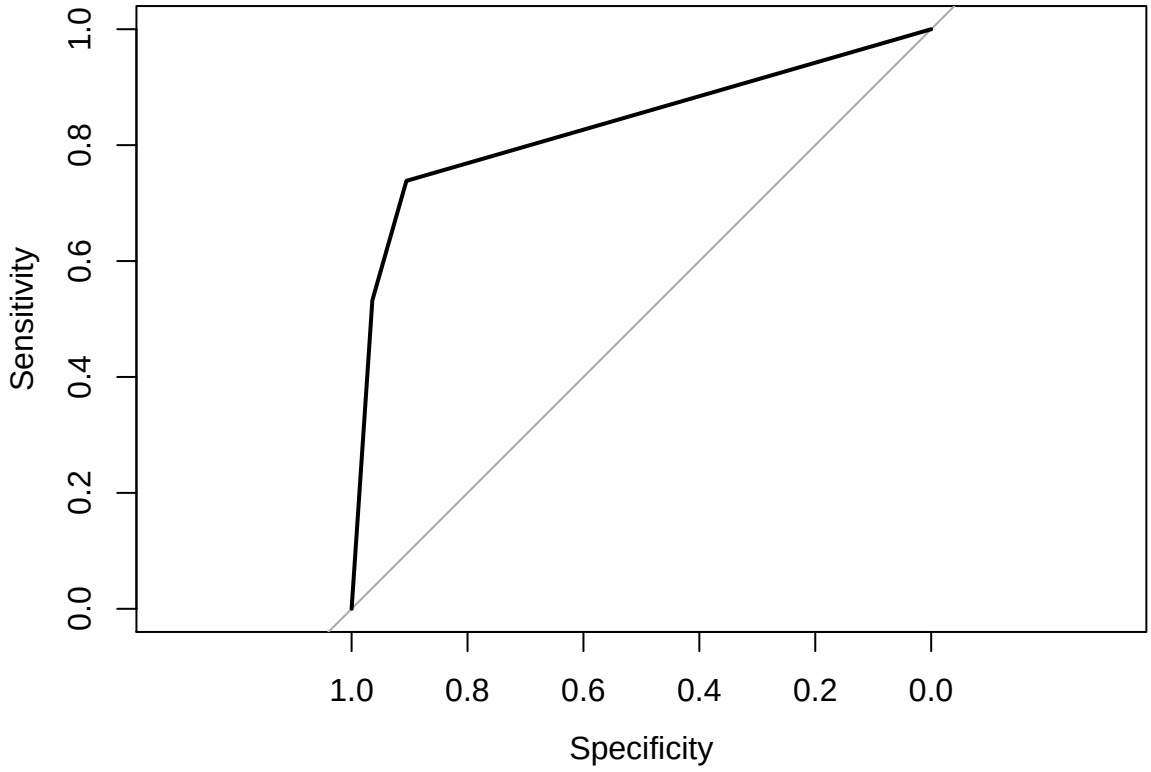
```
# We want only the probabilities for those categorized as yes, so we specify [,2].
pred3<-predict(wtloss_1, lose5_overagefw)[,2]
```

```
# Now, we use the predicted responses to build an ROC curve with subsample 3.
roc_test3 <- roc(lose5_overagefw$outcome5new, pred3)
```

```
# We then compute the area under the ROC curve.
a_test3 <- auc(roc_test3)
a_test3
```

```
## Area under the curve: 0.8339
```

```
# We also can plot the area under the ROC curve.
plot(roc_test3)
```




```
##
## Call:
## roc.default(response = lose5_overagefw$outcome5new, predictor = pred3)
##
## Data: pred3 in 275402 controls (lose5_overagefw$outcome5new no) < 48573 cases (lose5_overagefw$outcon
## Area under the curve: 0.8339
```