**Background**

AUC is an important metric in machine learning for classification. It is often used as a measure of a model’s performance. In effect, AUC is a measure between 0 and 1 of a model’s performance that rank-orders predictions from a model. For a detailed explanation of AUC, see [this link](https://stats.stackexchange.com/questions/132777/what-does-auc-stand-for-and-what-is-it).

Since AUC is widely used, being able to get a confidence interval around this metric is valuable to both better demonstrate a model’s performance, as well as to better compare two or more models. For example, if model A has an AUC higher than model B, but the 95% confidence interval around each AUC value overlaps, then the models may not be statistically different in performance. We can get a confidence interval around AUC using [R’s](http://theautomatic.net/category/r/) **pROC** package, which uses bootstrapping to calculate the interval.

**Building a simple model to test**

To demonstrate how to get an AUC confidence interval, let’s build a model using a movies dataset from Kaggle ([you can get the data here](https://www.kaggle.com/carolzhangdc/imdb-5000-movie-dataset/downloads/imdb-5000-movie-dataset.zip/1)).

**Reading in the data**

# load packages

library(pROC)

library(dplyr)

library(randomForest)

# read in dataset

movies <- read.csv("movie\_metadata.csv")

# remove records with missing budget / gross data

movies <- movies %>% filter(![is.na](http://is.na)(budget) & ![is.na](http://is.na" \t "_blank)(gross))

**Split into train / test**

Next, let’s randomly select 70% of the records to be in the training set and leave the rest for testing.

# get random sample of rows

set.seed(0)

train\_rows <- sample(1:nrow(movies), .7 \* nrow(movies))

# split data into train / test

train\_data <- movies[train\_rows,]

test\_data <- movies[-train\_rows,]

# select only fields we need

train\_need <- train\_data %>% select(gross, duration, director\_facebook\_likes, budget, imdb\_score, content\_rating, movie\_title)

test\_need <- test\_data %>% select(gross, duration, director\_facebook\_likes, budget, imdb\_score, content\_rating, movie\_title)

**Create the label**

Lastly, we need to create our label i.e. what we’re trying to predict. Here, we’re going to predict if a movie’s gross beats its budget (1 if so, 0 if not).

train\_need$beat\_budget <- as.factor(ifelse(train\_need$gross > train\_need$budget, 1, 0))

test\_need$beat\_budget <- as.factor(ifelse(test\_need$gross > test\_need$budget, 1, 0))

**Train a random forest**

Now, let’s train a simple random forest model with just 50 trees.

# train a random forest

forest <- randomForest(beat\_budget ~ duration + director\_facebook\_likes + budget + imdb\_score + content\_rating,

train\_need, ntree = 50, na.omit = TRUE)

**Getting an AUC confidence interval**

Next, let’s use our model to get predictions on the test set.

test\_pred <- predict(forest, test\_need, type = "prob")[,2]

And now, we’re reading to get our confidence interval! We can do that in just one line of code using the *ci.auc* function from **pROC**. By default, this function uses 2000 bootstraps to calculate a 95% confidence interval. This means our 95% confidence interval for the AUC on the test set is between 0.6198 and 0.6822, as can be seen below.

ci.auc(test\_need$beat\_budget, test\_pred)

# 95% CI: 0.6198-0.6822 (DeLong)

We can adjust the confidence interval using the *conf.level* parameter:

ci.auc(test\_need$beat\_budget, test\_pred, conf.level = 0.9)

# 90% CI: 0.6248-0.6772 (DeLong)