Anahuac ML: Problem Set 2

Prof. Jonathan Hersh

Question 1: What Predicts Blockbuster Movies?

a) Run the code below to clean the movies data frame and generate test and training sets of the clean data. Note we have created a variable `top_director` if a director is in the top 10% of film directors by number of films produced. We have also used the `left_join()` command to merge this variable back into our main dataset. We will explore this type of feature generation later in the course, but for now please you can take the code as given and run the code without fully understanding it.

```
library('tidyverse')
options(scipen = 50)
set.seed(1861)
movies <- read.csv(here::here("datasets","movie_metadata.csv"))
movies <- movies %>% filter(budget < 400000000) %>%
  filter(content_rating != "",
         content_rating != "Not Rated",
         !is.na(gross))
movies <- movies %>%
  mutate(genre_main = unlist(map(strsplit(as.character(movies$genres),"\\|"),1)),
                             grossM = gross / 1000000,
                             budgetM = budget / 1000000,
                             profitM = grossM - budgetM,

                             blockbuster = ifelse(grossM > 200,1,0))
movies <- movies %>% mutate(genre_main = fct_lump(genre_main,5),
                             content_rating = fct_lump(content_rating,3),
                             country = fct_lump(country,2),
                             cast_total_facebook_likes000s =
           cast_total_facebook_likes / 1000,) %>%
  drop_na()
```

```
top_director <- movies %>%
  group_by(director_name) %>%
  summarize(num_films = n()) %>%
  top_frac(.1) %>%
  mutate(top_director = 1) %>%
  select(-num_films)

movies <- movies %>%
  left_join(top_director, by = "director_name") %>%
  mutate(top_director = replace_na(top_director,0))

train_idx <- sample(1:nrow(movies),size = floor(0.75*nrow(movies)))
movies_train <- movies %>% slice(train_idx)
movies_test <- movies %>% slice(-train_idx)
```

b) One concern we might have is that the distributions of the outcome we want to predict – in this case `blockbuster` – is different in the test and training sets. This can arise through random chance, but can affect our outcome. Use the `mean()` command to calculate the average number of blockbusters in the test and training sets. To test for whether this difference in means is statistically meaningful, use the `t.test()` command to calculate a statistical test where the null hypothesis is that the means are similar in both datasets. What is the p-value you calculate? What do you conclude from this p-value?

c) Estimate a logit model with `blockbuster` as your dependent variable, and include `budgetM`, `top_director`, `cast_total_facebook_likes000s`, `content_rating`, and `genre_main` as your predictor variables. Use the summary command to display the output of your model.

d) Interpret the coefficients on `content_ratingR`, `genre_mainAdventure`, and `top_director`.

e) You are worried that the predictions from the model might be overfit if you only rely on the in-sample predictions. And you're concerned that the test set might have high variance. Therefore you decide to implement leave-one-out-cross validation to estimate the predicted probabilities (scores). Using a `for` loop, generate LOOCV predicted probabilities for each observation in the training set, being sure to estimate the model on every observation in the training set except that observation. Store these predictions in the vector `preds_LOOCV`.

f) Use the fitted model to generate predictions into the test and training sets.

g) Plot the three different ROC curves for the test predictions, the in-sample training predictions, and the LOOCV predictions. Describe the three ROC curves in relation to each other.

h) Calculate the AUC for the three models. Order the models in terms of performance. Why do you suppose they are ordered as they are?

Question 2: Can we parsimoniously predict bike share usage?

a) We'll use the Bike Sharing Dataset at the UCI Machine Learning website. Follow the code below to download and clean the `day.csv` dataset.

```
set.seed(1861)
options(scipen = 10)
library("readr")
Bike_DF <- read_csv(here::here("datasets", "day.csv"))

library("tidyverse")
Bike_DF <- Bike_DF %>% mutate(weathersit = factor(weathersit),
    season = factor(season), yr = factor(yr), month = factor(mnth),
    holiday = factor(holiday), weekday = factor(weekday), workingday = factor(workingday))


Bike_DF <- Bike_DF %>% mutate(temp_sq = temp * temp, atemp_sq = atemp *
    atemp, hum_sq = hum * hum, windspeed_sq = windspeed * windspeed)

train_idx <- sample(1:nrow(Bike_DF), size = 0.8 * floor(nrow(Bike_DF)))
bike_train <- Bike_DF %>% slice(train_idx)
bike_test <- Bike_DF %>% slice(-train_idx)
```

b) Estimate a Lasso model predicting `cnt` as a function of `season`, `holiday`,`workingday`,`weathersit`, `temp`,`temp_sq`,`hum`,`hum_sq`,`windspeed`,`windspeed_sq`, `mnth` and `yr`. Store this as an object `lasso_fit`.

c) Use the `plot` command over the fitted object. Describe the plot, being sure to label the two dashed vertical lines, and the numbers at the top of the plot.

d) Use the `coef()` function to print the coefficients for the lasso model, being sure to use the `round` function to round to three digits.

e) Print the coefficients for the `lambda.min` and `lambda.1se` versions of the model using the `coef` function, again rounding to the 3rd digit.

f) Create a grid of alphas between 0 and 1 of length 11 using the `seq` function. Use the `cva.glmnet` function to estimate an elasticNet model using this alpha grid, the `bike_train` dataset and the outcome and predictors used in the Lasso model. Use the `print` function over this stored object to show model summary information.

g) Use the `mindlossplot` function over the elasticNet model. Describe the plot. What do you conclude from the model?

h) In your own words, describe how a Lasso model is different from a regular OLS model.

i) Estimate a regular OLS model over the bike train dataset using the same outcome and predictors as before. Use the `summary` command over the model. Note the R^2 value.

Question 3 Can Tree Models Predict Movie Profit?

a) Let's apply the cleaning code to our data to generate data which we can use for modeling.

```
library("tidyverse")
options(scipen = 50)
set.seed(1861)
movies <- read.csv(here::here("datasets", "movie_metadata.csv"))
movies <- movies %>% filter(budget < 400000000) %>% filter(content_rating !=
    "", content_rating != "Not Rated", plot_keywords != "", !is.na(gross))
movies <- movies %>% mutate(genre_main = unlist(map(strsplit(as.character(movies$genres),
    "\\|"), 1)), plot_main = unlist(map(strsplit(as.character(movies$plot_keywords),
    "\\|"), 1)), grossM = gross/1000000, budgetM = budget/1000000)
movies <- movies %>% mutate(genre_main = fct_lump(genre_main,
    7), plot_first = fct_lump(plot_main, 20), content_rating = fct_lump(content_rating,
    4), country = fct_lump(country, 8), language = fct_lump(language,
    4), cast_total_facebook_likes000s = cast_total_facebook_likes/1000,
    ) %>% drop_na()

top_director <- movies %>% group_by(director_name) %>% summarize(num_films = n()) %>%
    top_frac(0.1) %>% mutate(top_director = 1) %>% select(-num_films)

movies <- movies %>% left_join(top_director, by = "director_name") %>%
    mutate(top_director = replace_na(top_director, 0)) %>% select(-c(director_name,
    actor_2_name, gross, genres, actor_1_name, movie_title, actor_3_name,
    plot_keywords, movie_imdb_link, budget, color, aspect_ratio,
    plot_main, actor_3_facebook_likes, actor_2_facebook_likes,
    color, num_critic_for_reviews, num_voted_users, num_user_for_reviews,
    actor_2_facebook_likes))

sapply(movies %>% select_if(is.factor), table)
## $language
##
##  English   French Mandarin  Spanish    Other
##     3576       32       13       22       70
```

```
##
## $country
##
## Australia    Canada     China    France   Germany Hong Kong
##        39        57        13        97        79        13
##     Spain        UK       USA     Other
##        19       315      2974       107
##
## $content_rating
##
##     G    PG PG-13     R Other
##    87   565  1306  1694    61
##
## $genre_main
##
##    Action Adventure Biography    Comedy     Crime     Drama
##       952       367       204       979       250       654
##    Horror     Other
##       163       144
##
## $plot_first
##
##            1950s            1970s            actor
##               18               18               24
## african american            alien        apartment
##               24               69               19
##             army         assassin             baby
##               20               26               22
##             bank              bar       basketball
##               19               18               18
##           battle            beach      best friend
##               26               19               32
##  box office flop              boy        christmas
##               28               36               18
##              cia          college            death
##               19               22               40
##           friend            Other
##               21             3157
```

```
train_idx <- sample(1:nrow(movies), size = floor(0.75 * nrow(movies)))
movies_train <- movies %>% slice(train_idx)
movies_test <- movies %>% slice(-train_idx)
```

b) Fit a random forest model of 500 trees against the `movies_train` data. Use as `mtry` the square root of the number of variables – *that is the number of columns of the final data used for estimation.*

c) What value did you set `mtry` to in the problem above? Why is it not sqrt(16)?

d) How does our model improve as we increase the number of trees? Use `plot` over the function to determine how many trees should be used.

e) Use the function `plot_min_depth_distribution` in the `randomForestExplainer` package to explore mininum depth by variable. Write 3-5 sentences explaining the plot and its conclusions as you would explain to a client who has no background in machine learning. (Note, this step might take a while. On my laptop this takes about 1-2 minutes. Set 'catch=TRUE' as an option on the chunk and every time you knit your document unless you change the code you won't need to re-run this chunk.)

f) Use the function `plot_predict_interaction()` to produce two plots: one exploring the interactions between `budgetM` and `imdb_score` and another exploring interactions between `budgetM` and `title_year`. Write 3-5 sentences explaining the plots as you would to a non-expert, including the conclusion we would draw from such a plot.

g) Generate test predictions, out of bag predictions (read the help here to learn how to generate them) and in-bag predictions. Compare the performance metrics across the three different types of predictions. *Comment on why we see this pattern.*