Homework Assignment 5+7

Panagiotis (Panos) Toulis– Chicago Booth
BUS 41100 Applied Regression Analysis

1 Customer churn

Use the following code to load the customer churn data.

```
rm(list=ls())
library(tidyverse)
load_data = function() {
  telco = read.csv("Telco-Customer-Churn.csv", stringsAsFactors = T)
  telco$customerID = NULL # not useful
  telco$PaymentMethod = NULL
  telco = telco %>% filter(InternetService != "No" & PhoneService != "No")
  telco$PhoneService = NULL
  telco$TotalCharges = NULL
  telco = droplevels(telco)
  return(telco)
}

> telco = load_data()
```

(a) Using the dataset calculate the average monthly charges for customers who have churned. Then multiply this with an assumed 12.5% profit margin.[1] The resulting number (say `Cost_1`) represents the expected cost of losing one customer (as monthly revenue loss).

(b) Calculate the average monthly charges on customers who haven't churned. Then, multiply this number with (12.5%-5%) = 7.5%. The resulting number (say `Cost_2`) represents the average cost of retaining a customer through a retention strategy that gives a 5% discount to existing customers.

(c) Fit a full model as follows:

```
fit = glm(Churn ~ ., data=telco, family="binomial")
```

Define the classifier predictions as

```
ypred = as.numeric(fit$fitted.values > 0.5)
```

Using the true `telco$Churn`, calculate `FN` = total number of false negatives, and `FP` = total number of false positives in the model fit. Then, argue why

$$\texttt{Loss} = \texttt{Cost\_1} * \texttt{FN} + \texttt{Cost\_2} * \texttt{FP}$$

is a reasonable loss function that the company would like to minimize.

(d) What is the optimal cutoff value for the `Loss` in (c)? Plot the shape of the `Loss` with respect to the threshold when you present your result.

[1] https://www.investopedia.com/ask/answers/060215/what-average-profit-margin-company-telecommunications-sector.asp

2 Titanic dataset

The dataset `titanic.csv` contains actual survival data from the sinking of the "Titanic". Variable `Parch` is #parents/children in the ship and `SibSp` is #siblings/spouse in the ship. The variable `Pclass` refers to ticket class.

(a) Use the `boxplot` command to figure out which class is the most expensive one.

(b) Use the `table` command to create a confusion matrix for `Sex` and `Survived` variables. What we are looking for is a breakdown of the Titanic passengers in terms of whether they survived or not and their recorded sex. Do the same for `Pclass` and `Survived`. Then, explain why Jack [2] didn't stand a chance.

(c) Transform `Pclass` into a factor variable. Then run a logistic regression model of `Survived` on all other variables. Interpret briefly the coefficients for `Pclass` in terms of odds-ratios (i.e., use `exp(coeff)`).

(d) Sample 75% of the data at random to be the training dataset, and hold out the rest to be the testing set. Fit a logistic regression model of `Survived` over all other variables on the training dataset, and then predict the survival probabilities on the testing dataset. Using the 0.5-threshold rule classify all people in the testing set in terms of survival. Create a confusion matrix of your classification and the true survival status, and report your sensitivity and specificity. Report the specificity/sensitivity of your classifier.

3 Cross-validated ROC curve

Here, we will extend the example with the `Spotify` data to calculate a cross-validated ROC curve. Instead of the smart model in class we will fit a full model with interactions.

The following skeleton code performs the following tasks:

(i) Splits the data into 10 folds.

(ii) **For every fold:**

- It fits the `full` and `full_x` models on the training folds data.
- It predicts $P(Y = 1)$ on the testing fold data.
- It uses `calculate_roc` to calculate the ROC curve based on those predictions and the Y in test fold data, and stores the result.

(iii) It averages across all ROC curves. This is the cross-validated ROC curve for every model.

In the code that follows, fill in the above blanks. Also, plot the cross-validated ROC curves, and perform model selection.

[2] `http://jamescameronstitanic.wikia.com/wiki/Jack_Dawson`

```
music = read.csv("spotify.csv")
music$song_title = NULL; music$artist = NULL  # remove some columns

# ROC curve calculation. Returns 5000 x 2 matrix. Column 1=specificity, Column 2= sensitivity
num_cuts = 5000
calculate_roc = function(preds, y_true) {
  all_cuts = seq(0, 1, length.out=num_cuts)
  roc = matrix(0, nrow=0, ncol=2)
  colnames(roc) = c("specificity", "sensitivity")
  for(cutoff in all_cuts) {
    y_class = as.numeric(preds > cutoff)    # Classify with cutoff
    M1 = table(y_true, y_class) #   Create confusion matrix
    if(length(diag(M1)) == 2) {
      roc = rbind(roc, diag(M1) / rowSums(M1))  #  Check whether the confusion matrix is 2x2
    } else { roc = rbind(roc, c(0, 1)) }
  }
  return(roc)
}

set.seed(41100)   # keep this here. It's important.
num_folds = 10    # (i)
I = seq(1, nrow(music))
folds = as.numeric(cut(I, num_folds))  #  = 1 1 1 1 1 2 2 2 2 3 3 3 3....
folds = sample(folds)  # Shuffle. Now folds[i] tells us which fold datapoint i is in.
cv_roc_full = matrix(0, nrow=num_cuts, ncol=2) # CV ROC for full model
cv_roc_full_x = matrix(0, nrow=num_cuts, ncol=2). # CV ROC for full model with interactions
colnames(cv_roc_full) = c("sensitivity", "specificity")
colnames(cv_roc_full_x) = c("sensitivity", "specificity")

X = music[,-ncol(music)] # remove Y, keep only Xs.

for(k in 1:num_folds) {
  # Fold k will be for testing.  All else will be for training.
  index_train = which(folds != k)
  index_test = which(folds == k)

  data_train = music[index_train,]
  Xtest = X[index_test,]
  Ytest = music$like[...]

  # (ii)a Train on training data.
  full_k = ...
  full_x_k = ...

  # (ii)b Predict on testing data.
  yhat_full = predict(..., newdata=...)
  yhat_full_x = predict(..., newdata=...)

  # (ii)c Calculate ROC curves
  curve_full = calculate_roc(..., ...)
  curve_full_x = calculate_roc(..., ...)

  # Update CV roc curve. Keep adding for now. Will average later.
  cv_roc_full = cv_roc_full + curve_full
  cv_roc_full_x = cv_roc_full_x + curve_full_x
}
# (iii) Average ROC curves.
cv_roc_full = cv_roc_full / ...
cv_roc_full_x = cv_roc_full_x / ...
```

4 Learning to read

The file `letters.csv` contains writing examples of English letters. Variable `lettr` has the letter label of each example, and the other variables correspond to geometrical aspects of letter writing. For more information you may take a look here `https://archive.ics.uci.edu/ml/machine-learning-databases/letter-recognition/letter-recognition.names`.

(a) Split the data in a training and a testing set. The training set should be a random 75% of the original data, and the testing set should be the rest. Fit a multinomial regression model on the training set using `lettr` as the response variable.

(b) Report the accuracy of your model in recognizing letters in the training set (see *Help* below).

(c) Report the accuracy of your model in recognizing letters in the testing set.

(d) Now fit the model on the full data set. Use the fitted model to decode the message hidden in `topsecret.csv` (*Hint: It's a Booth course!*)

Help for this Problem: Suppose you fit a multinomial model on some `train` data:

```
fit = multinom(letter ~., data=train)
```

You can get what letters the model predicts for that data by doing:

```
train_pred = predict(fit, newdata=train, type="class")
> head(train_pred)
[1] P R N P M Q
Levels: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
```

Then the confusion matrix (26 x 26) is simply:

```
M = table(train$lettr, train_pred)
```

For classification accuracy we can calculate what is the proportion of correct classifications over all classifications made by the model:

```
> sum(diag(M)) / sum(M)
[1] 0.7338
```

You can follow the same procedure for `test` data as well.