Homework 4
MSBA 400: Statistical Foundations for Data Analytics
Professor Rossi

Question: Prediction of Catalogue Orders

The dataset `cat_buy.rda` contains data on the response of customers to the mailing of spring catalogues. The variable `buytabw` is `1` if there is an order from this spring catalogue and `0` if not. This is the dependent or response variable (literally was there a "response" to or order from the direct mailing).

This spring catalogue was called a "tabloid" in the industry. The catalogue featured women's clothing and shoes. The independent variables represent information gathered from the internal `house file` of the past order activity of these 20,627 customers who received this catalogue.

In direct marketing, the predictor variables are typically of the "RFM" type: 1. Recency 2. Frequency and 3. Monetary value. This data set has both information on the volume of past orders as well as the recency of these orders.

The variables are: * tabordrs (total orders from past tabloids)
* divsords (total orders of shoes in past)
* divwords (total orders of women's clothes in past)
* spgtabord (total orders from past spring cats)
* moslsdvs (mos since last shoe order)
* moslsdvw (mos since last women's clothes order)
* moslstab (mos since last tabloid order)
* orders (total orders)

part A

Use the R `sample` command to randomly sample 1/2 of the data. The sample command will sample randomly from a list of numbers, e.g. 2, 3, 7, 6, 1 will select 5 from the numbers 1,2,3,4,5,6,7,8,9,10.

Use `sample` to select row numbers and then use these row numbers to divide your data into two parts. One part for estimation and one part for validation.

Hint: see code below (modify)

```
ind.est=sample(???,???)
est_sample = cat_buy[ind.est,]
holdout_sample = cat_buy[-ind.est,]
```

Solution:

```
# load data
load(file = 'cat_buy.rda')
# seed for reproducibility
set.seed(1)
# get rows for estimation and validation
size_est = as.integer(nrow(cat_buy) * 0.5) # 50% estimation (as.integer() since odd amt. samples)
size_val = nrow(cat_buy) - size_est # 50% validation
ind.est = sample(1: nrow(cat_buy), size = size_est)
ind.val = sample(1: nrow(cat_buy), size = size_val)
# split into estimation and validation sets
est_sample = cat_buy[unlist(ind.est),]
holdout_sample = cat_buy[unlist(-ind.est),]
```

part B

Fit a logistic regression model using the estimation sample produced in part A. Eliminate insignificant variables.

Discuss your final specification—do the signs of the coefficients make sense to you?

Should you worry about multi-collinearity in this dataset?

Solution:

```
# fit initial logistic regression
lr = glm(buytabw ~ ., data = est_sample, family = 'binomial')
summary(lr)
```

```
##
## Call:
## glm(formula = buytabw ~ ., family = "binomial", data = est_sample)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1835  -0.6370  -0.3769  -0.1311   3.0633
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.083313   0.093559 -11.579  < 2e-16 ***
## tabordrs     0.067046   0.013916   4.818 1.45e-06 ***
## divsords     0.023015   0.015977   1.441 0.149721
## divwords     0.107208   0.008277  12.953  < 2e-16 ***
## spgtabord    0.047141   0.019019   2.479 0.013189 *
## moslsdvs    -0.007613   0.002198  -3.464 0.000533 ***
## moslsdvw    -0.067523   0.005235 -12.899  < 2e-16 ***
## moslstab    -0.046271   0.004523 -10.231  < 2e-16 ***
## orders      -0.049407   0.005851  -8.444  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 9468.7  on 10312  degrees of freedom
## Residual deviance: 7481.2  on 10304  degrees of freedom
## AIC: 7499.2
##
## Number of Fisher Scoring iterations: 6
```

We can see that the variable `divsords` has a relatively large p-value, so we can eliminate it from the model. We fit a new logistic regression:

```
# fitting revised logistic regression
lr_new = glm(buytabw ~ . - divsords, data = est_sample, family = 'binomial')
summary(lr_new)
```

```
##
## Call:
## glm(formula = buytabw ~ . - divsords, family = "binomial", data = est_sample)
##
```

```
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1748  -0.6372  -0.3761  -0.1311   3.0630
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.038283   0.087950 -11.805  < 2e-16 ***
## tabordrs     0.066833   0.013899   4.809 1.52e-06 ***
## divwords     0.106777   0.008259  12.928  < 2e-16 ***
## spgtabord    0.048072   0.018983   2.532   0.0113 *
## moslsdvs    -0.009403   0.001809  -5.197 2.03e-07 ***
## moslsdvw    -0.067400   0.005233 -12.881  < 2e-16 ***
## moslstab    -0.046102   0.004523 -10.194  < 2e-16 ***
## orders      -0.047026   0.005594  -8.407  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 9468.7  on 10312  degrees of freedom
## Residual deviance: 7483.2  on 10305  degrees of freedom
## AIC: 7499.2
##
## Number of Fisher Scoring iterations: 6
```

We see that the AIC has remained the same, thus justifying the removal of the variable `divsords`.

We can see that `tabordrs`, `divwords`, and `spgtabord` have positive coefficients. This makes sense as they are all indicators of a positive purchase history in the past. Accordingly, it would make sense that more purchases in the past would correspond to an increased likelihood of future purchases.

We can also see that `moslsdvs`, `moslsdvw`, and `moslstab` have negative coefficients. This makes a lot of sense as a longer time since a customer's last purchase would logically imply that they are less likely to purchase in the future. The negative sign for `orders`, however, seems counterintuitive and does not make much sense as we would expect a higher number of past orders to make a user more likely to purchase again.

We compute variance-inflation factors to check for multi-collinearity:

```
library(car)
```

```
## Loading required package: carData
```

```
vif(lm(buytabw ~ . - divsords, data = est_sample))
```

```
##  tabordrs  divwords spgtabord  moslsdvs  moslsdvw  moslstab    orders
##  6.778159  2.915672  5.030645  1.177720  1.316083  1.327025  3.549035
```

Based on the variance-inflation factors, we can see that there is not a substantial concern about multi-collinearity (assuming that multi-collinearity is a potential issue when we have one or more VIFs ≥ 10).

part C

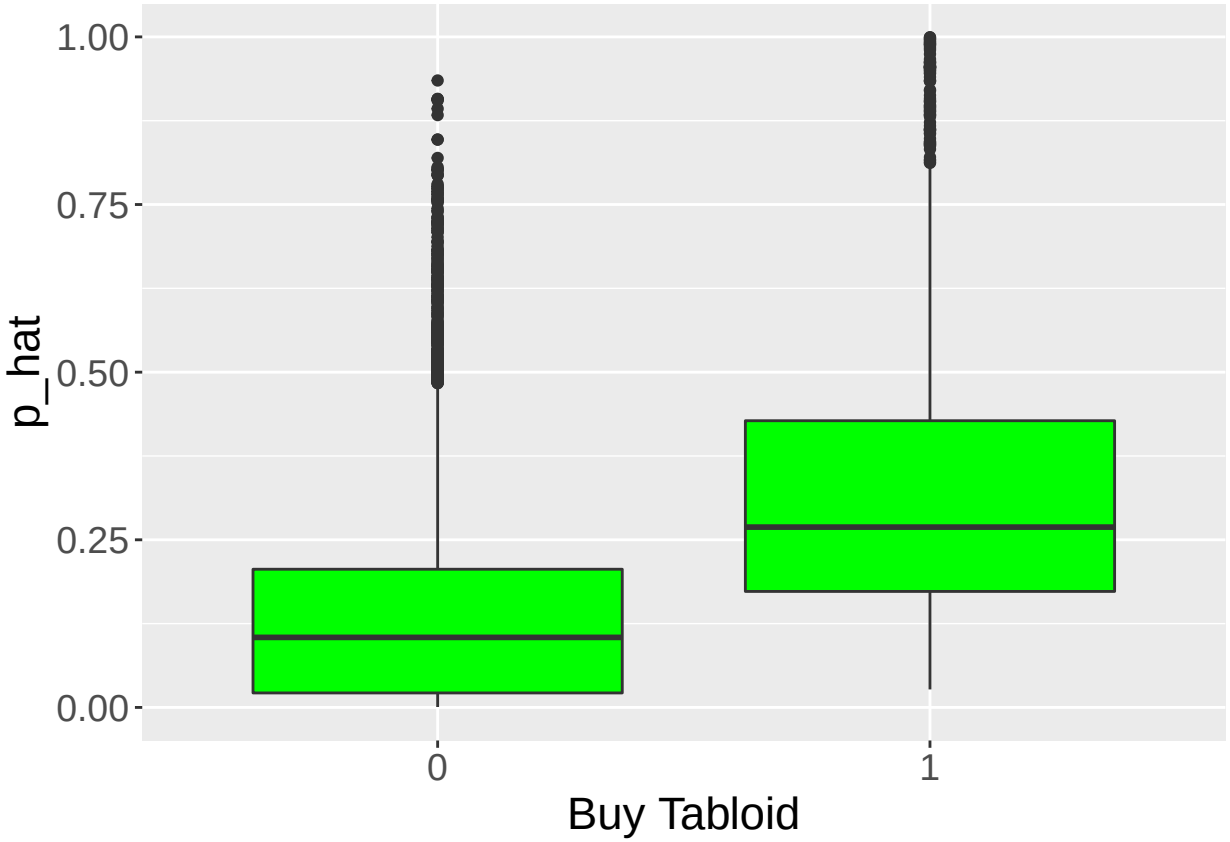
Use the best-fit from part B to predict using the holdout sample.

Plot boxplots of the fitted probabilities for each value of `buytabw` for the holdout sample (see code snippets from Chapter 7 for an example)

Compute a "lift" table as done in Chapter 7 code snippets.

Solution:

```
# predicting on the holdout sample with optimal model
p_hat = predict(lr_new, holdout_sample, type = 'response')
# creating boxplots
library(ggplot2)
qplot(factor(holdout_sample$buytabw),
      p_hat, geom = "boxplot", fill = I("green"),
      xlab = "Buy Tabloid") +
  theme(axis.title = element_text(size = rel(1.5)),
        axis.text = element_text(size = rel(1.25)))
```



We can see from the boxplots that there is a substantial degree of separation between the `p_hat` values for the different classes. This indicates a relatively good model performance.

Computing a lift table:

```
# create deciles
deciles = cut(p_hat, breaks = quantile(p_hat, probs = c(seq(from = 0, to = 1, by = .1))),
              include.lowest = TRUE)
deciles = as.numeric(deciles)
# construct data frame, aggregate to lift table
df = data.frame(deciles = deciles, phat = p_hat, default = holdout_sample$buytabw)
lift = aggregate(df, by = list(deciles), FUN="mean", data=df)
lift=lift[,c(2,4)]
lift[,3]=lift[,2]/mean(holdout_sample$buytabw)
names(lift) = c("decile","Mean Response","Lift Factor")
lift
```

```
##    decile Mean Response Lift Factor
## 1       1   0.000000000   0.0000000
## 2       2   0.000000000   0.0000000
## 3       3   0.006789525   0.0389256
## 4       4   0.070736434   0.4055451
## 5       5   0.158098933   0.9064104
## 6       6   0.188166828   1.0787953
## 7       7   0.236434109   1.3555205
## 8       8   0.237633366   1.3623961
## 9       9   0.336566440   1.9295977
## 10     10   0.509689922   2.9221467
```

We can see that the lift factors increase monotonically as we move up the deciles. This indicates a good model performance.