7313 Hand-in - Assignment 3

Group 3

2021-12-06

#Data Import

```r
#Creating connection to MySQL database
con = dbConnect(MySQL(), dbname = "BnL",
                host = "mysql-1.cda.hhs.se", port = 3306,
                user = "bnl", password = "bnl@sse")
#Loading data into R
df <- dbGetQuery(con,"SELECT IF(SUM(returned)/COUNT(returned)>0,1,0) AS is_returned,
SUM(IF(amount>=0,amount,0))/SUM(IF(quantity>=0,quantity,0)) AS avg_amount,
SUM(IF(discount>=0,discount,0))/SUM(IF(quantity>=0,quantity,0)) AS avg_discount,
SUM(IF(quantity>=0,quantity,0)) AS total_quantity, MIN(dept_desc) AS category
FROM Transactions t LEFT JOIN Products p
USING(item)
GROUP BY(item)")
```

#Data Preparation: We reduced the threshold of SUM(returned)/COUNT(returned) to <0 (vs. <0.5) so that the number of items classified as "returned" increased from 124 to 3,600. We tested 3 different datasets: A balanced dataset (threshold <0.5), an unbalanced dataset (threshold <0), and another balanced dataset (threshold <0). We obtained the best test performance with the unbalanced dataset and a threshold of <0. Data balancing was performed with over- and undersampling (alternatively, we could have used the ROSE package). Data balancing may be reasonable if the test set was not representative, i.e., the proportion of items being classified as returned was much higher. Next to business acumen and judgement, we used lasso and ridge regression as well as variable importance for feature selection. We first started with a larger dataset that comprised more variables, incl. "division__desc" and "dept__desc" as categories, "sustainability__id__desc" as a binary variable (sustainable vs. non-sustainable), and the month in which the most transactions per item took place. We converted these variables into dummies using dummyVars to use them for the modeling part. However, based on the output of the lasso and ridge regressions as well as the variable importance, we figured out that "total_quantity", "avg_amount" and "avg_discount" are the best-performing/most important predictors. In addition, in the evaluation step, we figured out that the different models performed equally good with only the 3 predictors. Taking into account model complexity (bias-variance-tradeoff), we decided to train our final model only w/ the 3 predictors.

```r
#Handling missing values (mode imputation for characters + mean imputation for doubles)
mode_category <- df %>%
  filter(!is.na(category)) %>%
  count(category) %>%
  top_n(1, n) %>%
  select(category) %>%
  unlist(use.names = F)
df_clean <- df %>%
  mutate(avg_amount = ifelse(is.na(avg_amount), mean(avg_amount, na.rm = T), avg_amount),
```

```r
        avg_discount = ifelse(is.na(avg_discount), mean(avg_discount, na.rm = T), avg_discount),
        category = ifelse(is.na(category), mode_category, category))
#Excluding non-product-related observations (Receipt texts, Gift With Purchase, Marketing Material, Sal
df_clean <- df_clean %>%
  mutate(no_product = ifelse(category == "Receipt texts", 1,
                             ifelse(category == "Gift With Purchase", 1,
                                    ifelse(category == "Marketing Material", 1,
                                           ifelse(category == "Sales Kicks E-commerce", 1, 0)))))
df_clean <- df_clean %>%
  filter(no_product == 0) %>%
  select(-no_product, -category) #category only used to exclude non-product-related items
#Creating dummy variables (not necessary for 3 predictors; used for dataset w/ categorical variables)
#Ensuring correct data types
df_final <- df_clean %>%
  mutate(is_returned = as.factor(ifelse(is_returned == 1, "yes", "no")),
        total_quantity = as.integer(total_quantity))
#Partitioning dataset into train (70%) and test set(30%)
set.seed(7313)
smp_siz = round(0.7*nrow(df_final))
train_row = sample(seq_len(nrow(df_final)),size = smp_siz)
train = df_final[train_row,]
test = df_final[-train_row,]
#Handling data imbalance (not necessary for this dataset)
```

#Modeling: For modeling, we tested: logistic regression, ridge & lasso logistic regression, stepwise logistic regression, decision trees, random forest, and boosted trees (XGBoost). It was not possible to train Neural Networks since the keras interface did not work on the MacBook (as mentioned in our first lecture). We used 10-fold repeated cross-validation to minimize the risk of overfitting (bias-variance-tradeoff).

```r
#Training a logistic regression
train.control <- trainControl(method = "repeatedcv",
                              number = 10,
                              repeats = 10,
                              verboseIter = F,
                              classProbs = T,
                              summaryFunction = prSummary)
set.seed(7313)
simple.logistic.regression <- train(is_returned ~ .,
                                    data = train,
                                    method = "glm",
                                    metric = "AUC",
                                    trControl = train.control)
#Training performance
#simple.logistic.regression
#summary(simple.logistic.regression)
#confusionMatrix(simple.logistic.regression)
```

#Model Evaluation: We selected AUC and accuracy as our primary performance metrics to evaluate the model performance since we wanted to obtain a model with an overall good performance that reduces both FNs and FPs. Based on these 2 metrics, the simple logistic regression with the 3 predictors was among the best performing models (Accuracy: 0.88; AUC: 0.96). In addition, we used the same dataset to train and tune a XGBoost model (tuning parameters: nrounds: 100, max_depth: 2, eta: 0.1, gamma: 0, colsample_bytree; 1, min_child_weight: 1, subsample: 1) and achieved a slightly better performance

(Accuracy: 0.89; AUC: 0.96). In our business case, we want to identify the drivers of product returns, which is why the interpretability of the model is slightly more important to weight than performance. Therefore, taking into account the performance-interpretability-tradeoff and Occam's Razor principle, we opted for the simple logistic regression model. Lastly, the training and test performance differed only marginally, pointing to a good bias-variance-tradeoff.

```
#Making predictions on the test set
test.predictions <- data.frame(pred = predict(simple.logistic.regression, newdata = test),
                               obs = test$is_returned)
prob.predictions <- predict(simple.logistic.regression, newdata = test, type = "prob")
test.predictions <- bind_cols(test.predictions, prob.predictions)
test.predictions <- test.predictions %>%
  mutate(obs = as.factor(obs)) %>%
  mutate(obs = factor(obs, levels = c("no", "yes")))
#Test performance
confusionMatrix(test.predictions$pred,
                test.predictions$obs)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   no  yes
##        no  7063  905
##        yes   81  138
##
##                Accuracy : 0.8796
##                  95% CI : (0.8723, 0.8865)
##     No Information Rate : 0.8726
##     P-Value [Acc > NIR] : 0.02982
##
##                   Kappa : 0.1826
##
##  Mcnemar's Test P-Value : < 2e-16
##
##             Sensitivity : 0.9887
##             Specificity : 0.1323
##          Pos Pred Value : 0.8864
##          Neg Pred Value : 0.6301
##              Prevalence : 0.8726
##          Detection Rate : 0.8627
##    Detection Prevalence : 0.9733
##       Balanced Accuracy : 0.5605
##
##        'Positive' Class : no
##
```

```
#twoClassSummary(test.predictions, lev = c("no", "yes"))
prSummary(test.predictions, lev = c("no", "yes"))
```

```
##       AUC Precision    Recall         F
## 0.9605462 0.8864207 0.9886618 0.9347538
```

#Business Case: The average amount of an item and the total amount an item was purchased have a significantly (***) positive effect on whether a product is returned, whereas the average discount of an item has a negative effect. Although the coefficients of avg_amount and avg_discount a considerably lower, one must keep in mind that the average values for both variables are higher (Mean: 2278 and 638.8) than for total_quantity (19.57). Nevertheless, the total quantity has the greatest effect on whether an item is returned or not.