Predicting NANSE Store Traffic Through Generalized Linear Models (GLM)

Creating the confusion matrix

```r
# RStudio Options
options(scipen = 1000) #Prevent display in scientific notation
# Run this reusable confusion matrix function (https://en.wikipedia.org/wiki/Confusion_matrix)
my_confusion_matrix <- function(cf_table) {
  true_positive <- cf_table[4]
  true_negative <- cf_table[1]
  false_positive <- cf_table[2]
  false_negative <- cf_table[3]
  accuracy <- (true_positive + true_negative) / (true_positive + true_negative + false_positive + false_
  sensitivity_recall <- true_positive / (true_positive + false_negative)
  specificity_selectivity <- true_negative / (true_negative + false_positive)
  precision <- true_positive / (true_positive + false_positive)
  neg_pred_value <- true_negative/(true_negative + false_negative)
  print(cf_table)
  my_list <- list(sprintf("%1.0f = True Positive (TP), Hit", true_positive),
                  sprintf("%1.0f = True Negative (TN), Rejection", true_negative),
                  sprintf("%1.0f = False Positive (FP), Type 1 Error", false_positive),
                  sprintf("%1.0f = False Negative (FN), Type 2 Error", false_negative),
                  sprintf("%1.4f = Accuracy (TP+TN/(TP+TN+FP+FN))", accuracy),
                  sprintf("%1.4f = Sensitivity, Recall, Hit Rate, True Positive Rate (How many positives
                  sprintf("%1.4f = Specificity, Selectivity, True Negative Rate (How many negatives did
                  sprintf("%1.4f = Precision, Positive Predictive Value (How good are the model's positi
                  sprintf("%1.4f = Negative Predictive Value (How good are the model's negative predicti
  )
  return(my_list)
}
```

Installing and Loading Packages

```r
#install.packages('tidyverse')
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
# Load data
df <- read_rds("mod6HE_logit.rds")

# Explore the data and discuss in PowerPoint
summary(df)
```

```
##      store            week         high_med_gp      high_med_rev   
##  186    :  52   Min.   : 1.00   Min.   :0.0000   Min.   :0.0000  
##  227    :  52   1st Qu.:13.00   1st Qu.:0.0000   1st Qu.:0.0000  
##  233    :  52   Median :26.00   Median :0.0000   Median :0.0000  
##  236    :  52   Mean   :26.47   Mean   :0.4997   Mean   :0.4997  
##  272    :  52   3rd Qu.:39.00   3rd Qu.:1.0000   3rd Qu.:1.0000  
##  291    :  52   Max.   :52.00   Max.   :1.0000   Max.   :1.0000  
##  (Other):9759                                                    
##  high_med_units    high_med_gpm         size             region    
##  Min.   :0.0000   Min.   :0.0000   Min.   : 890.0   ONTARIO :3120  
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 916.0   WEST    :4776  
##  Median :0.0000   Median :0.0000   Median : 943.0   QUEBEC  :1863  
##  Mean   :0.4995   Mean   :0.4639   Mean   : 949.6   ATLANTIC: 312  
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.: 972.0                  
##  Max.   :1.0000   Max.   :1.0000   Max.   :1163.0                  
##                                                                    
##  promo_units_per  altbev_units_per confect_units_per salty_units_per 
##  Min.   :0.1053   Min.   :0.2030   Min.   :0.0000    Min.   :0.0000  
##  1st Qu.:0.3046   1st Qu.:0.3575   1st Qu.:0.2349    1st Qu.:0.1520  
##  Median :0.3451   Median :0.4019   Median :0.2645    Median :0.1716  
##  Mean   :0.3506   Mean   :0.4055   Mean   :0.2706    Mean   :0.1737  
##  3rd Qu.:0.3929   3rd Qu.:0.4491   3rd Qu.:0.3012    3rd Qu.:0.1928  
##  Max.   :0.5797   Max.   :0.7250   Max.   :0.4741    Max.   :0.3684  
##                                                                      
##  velocityA_units_per velocityB_units_per velocityC_units_per
##  Min.   :0.4429      Min.   :0.1410      Min.   :0.00000    
##  1st Qu.:0.5744      1st Qu.:0.2032      1st Qu.:0.07078    
##  Median :0.6107      Median :0.2162      Median :0.07994    
##  Mean   :0.6086      Mean   :0.2169      Mean   :0.08009    
##  3rd Qu.:0.6428      3rd Qu.:0.2300      3rd Qu.:0.08929    
##  Max.   :0.7500      Max.   :0.3135      Max.   :0.14865    
##                                                             
##  velocityD_units_per velocityNEW_units_per
##  Min.   :0.00000     Min.   :0.000000     
##  1st Qu.:0.05533     1st Qu.:0.000000     
##  Median :0.07045     Median :0.000000     
##  Mean   :0.07918     Mean   :0.004608     
##  3rd Qu.:0.10265     3rd Qu.:0.007963     
##  Max.   :0.21637     Max.   :0.055623     
## 
```

Preparing the data for the logistic regression algorithm

```r
# Not for the model (for use later)
ColumnsNotUsed <- df %>%
  select(store, week, high_med_rev, high_med_gpm,high_med_gp)
```

For use in the model
```
# For use in the model
logit1 <- df %>%
  select(high_med_units,size, region, promo_units_per,
         altbev_units_per, confect_units_per, salty_units_per,
         velocityA_units_per, velocityB_units_per, velocityC_units_per, velocityD_units_per, velocityNEW
```

Partitioning the data into testing and training datasets

```
#install.packages('caret') (don't install twice)
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
set.seed(42)
partition <- caret::createDataPartition(y=logit1$high_med_units, p=.75, list=FALSE) #gives matrix of ro
data_train <- logit1[partition, ] #keeps the rows indicated in `partition` and all columns from `logit1
data_test <- logit1[-partition, ] #keeps the rows not indicated in `partition` and all columns from `lo
```

Training the multivariate model

These are the instructions part of machine learning

```
model_train <- glm(high_med_units ~ ., family=binomial, data=data_train)
summary(model_train)
```

```
##
## Call:
## glm(formula = high_med_units ~ ., family = binomial, data = data_train)
##
## Coefficients:
##                        Estimate  Std. Error z value            Pr(>|z|)
## (Intercept)         -33.7554575   3.4503718  -9.783 < 0.0000000000000002 ***
## size                  0.0198825   0.0008586  23.156 < 0.0000000000000002 ***
## regionWEST            1.2372452   0.0747641  16.549 < 0.0000000000000002 ***
## regionQUEBEC          1.6486085   0.0943744  17.469 < 0.0000000000000002 ***
## regionATLANTIC        0.5126349   0.1818002   2.820             0.004806 **
## promo_units_per      -7.0364858   0.5411102 -13.004 < 0.0000000000000002 ***
## altbev_units_per      7.2688615   0.9837629   7.389    0.000000000000148 ***
## confect_units_per    -0.8353451   1.1030602  -0.757             0.448871
## salty_units_per      26.0703631   1.4808940  17.604 < 0.0000000000000002 ***
## velocityA_units_per  11.4092559   3.1533882   3.618             0.000297 ***
## velocityB_units_per   5.1015780   3.2684432   1.561             0.118557
## velocityC_units_per  14.4423422   3.7934750   3.807             0.000141 ***
## velocityD_units_per   2.3609835   3.7725451   0.626             0.531424
## velocityNEW_units_per -33.5103228  5.0935739  -6.579    0.000000000047381 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 10471.8  on 7553  degrees of freedom
## Residual deviance:  7671.1  on 7540  degrees of freedom
## AIC: 7699.1
##
## Number of Fisher Scoring iterations: 5
```

Predicting the response variable on the test data using the training data

```r
predict_test <- predict(model_train, newdata=data_test, type='response')
```

Forming table to look at the accuracy of the model

```r
table2 <- table(predict_test>.5, data_test$high_med_units) #prediction on left and truth on top
my_confusion_matrix(table2)
```

```
##
##           0   1
##   FALSE 972 289
##   TRUE  314 942
```

```
## [[1]]
## [1] "942 = True Positive (TP), Hit"
##
## [[2]]
## [1] "972 = True Negative (TN), Rejection"
##
## [[3]]
## [1] "314 = False Positive (FP), Type 1 Error"
##
## [[4]]
## [1] "289 = False Negative (FN), Type 2 Error"
##
## [[5]]
## [1] "0.7604 = Accuracy (TP+TN/(TP+TN+FP+FN))"
##
## [[6]]
## [1] "0.7652 = Sensitivity, Recall, Hit Rate, True Positive Rate (How many positives did the model get
##
## [[7]]
## [1] "0.7558 = Specificity, Selectivity, True Negative Rate (How many negatives did the model get righ
##
## [[8]]
## [1] "0.7500 = Precision, Positive Predictive Value (How good are the model's positive predictions? T
##
## [[9]]
## [1] "0.7708 = Negative Predictive Value (How good are the model's negative predictions? TN/(TN+FN)"
```

Using the predictions above to help the business.

Putting the data back together for future use

```r
# Put the prediction back into the test data
data_test$prediction <- predict_test

# Create a variable that shows if the prediction was correct
# (We have to do the classification--in `round(prediction)`--since logistic regression gives us a proba
data_test <- data_test %>%
  mutate(correct_prediction = if_else(round(prediction) == high_med_units, 'correct', 'WRONG!'))

# Add back the original data
temp1 <- ColumnsNotUsed[-partition, ]
full_test <- bind_cols(temp1, data_test)

# For viewing in class
full_test <- full_test %>%
  select(store, week, high_med_units, prediction, correct_prediction,
         size, region, promo_units_per, salty_units_per)
slice_sample(full_test, n=10)
```

```
## # A tibble: 10 x 9
##    store  week high_med_units prediction correct_prediction  size region
##    <fct> <dbl>          <dbl>      <dbl> <chr>              <int> <fct>
##  1 63966    15              0     0.661  WRONG!               955 WEST
##  2 91361    23              1     0.692  correct              972 WEST
##  3 69921     3              0     0.405  correct              958 WEST
##  4 872       3              0     0.202  correct              937 ONTARIO
##  5 16110    21              0     0.528  WRONG!               915 QUEBEC
##  6 35155     9              0     0.0481 correct              930 ONTARIO
##  7 91226    14              1     0.893  correct              985 WEST
##  8 92576    49              1     0.910  correct             1015 WEST
##  9 37641     1              1     0.654  correct              967 WEST
## 10 77809     2              1     0.896  correct              982 WEST
## # i 2 more variables: promo_units_per <dbl>, salty_units_per <dbl>
```

##Summary of Findings

Feature/variable with the largest positive coefficient and statistically significant in the trained model summary

salty_units_per has the largest positive coefficient of 26.0703631 and is statistically significant because has a P-Value less than 0.05

Effect of selling a higher proportion of alternative beverages on the chance of having above median units sold

Selling a higher proportion of alternative beverages (variable "altbev_units_per") **increases** the chance of having above median units sold. We know this because "altbev_units_per" has a coefficient of 7.2688615, which means that 1 unit increase in the "altbev_units_per" variable, will increase the median units sold by around 7.2688615.

Effect of selling a higher proportion of velocity B units on the chance of having above median units sold

Neither increase nor decrease the chance of having the above median units sold. We know this because the coefficient of "velocityB_units_per" is 5.1015780 but the associate p-value of 0.118557 is greater 0.05, which indicates that the coefficient for the said variable is not statistically significance. In addition, this indicates that there is not enough evidence to conclude that selling a higher proportion of velocity B units will significantly affect, whether increase or decrease, the chance of having above median units sold.

Examining the accuracy of the predictions on the test data by answering whether there are more true positives or more true negatives.

There are more True Negatives (972) than True Positives (942).

First store in the 'full_test' dataset that has a "WRONG!" prediction

Store 186 located in ONTARIO region. This is true when stores are sorted by the 'store' feature in an ascending manner (lowest number first).

Why training data is used for the model training step

We use the **data-training** because it checks the quality and performance on how well the model generalizes new and unseen data.

Level of the variable is not present but accounted for in the intercept term

The level on the variable not present is but is accounted for in the intercept term is **ONTARIO**.

The feature 'region' has changed in the summary of the trained model. Further, only three regions show up in the summary of the model. The reasoning for this is that the 'glm()' function automatically recognizes that 'region' is a categorical variable (specifically a factor in R). Thus, the 'glm()' function has created "dummy variables" for the levels of 'region'.

Interpreting the confusion matrix using the test/holdout data - hgihest value

The **Negative Predictive Value (NPV)** has the highest value of 0.7708. This means that the model is good at making predictions that a predicted negative outcome is actually negative. The model is 77.08% accurate when it comes to predicitng negative outcome.

Interpreting the confusion matrix - lowest value

The **Precision, Positive Predictive Value** has the lowest value of 0.7500. This means that the model is not as good at making predictions that a predicted positive outcome is actually positive. The model is 75% accurate when it comes to predicting positive outcome.

Interpreting the confusion matrix - highest concern for NANSE

In NANSE's business setting, the measure that NANSE care about the most is **Sensitivity**, which represents the proportion of weeks and stores with above-median sales taht are correctly identified on the model. NANSE would care about **Sensitivity** because the store would want to ensure that the model will correctly capture instances of higher traffic and lessen the risk of false negatives. Meaning, NANSE will want to ensure that they minimize instances that theiir model fails to correctly predict above median sales when they occur, which could adversely impact business decisions in critical areas such as inventory management and resource allocation (e.g., marketing costs).