Generalized Linear Models

February 8, 2019

1 Generalized Linear Models

```
In [1]: getwd()
```

```
'/Users/harsh/code/adbi/3-regression/1-glm'
```

1.0.1 Load the Data

```
In [2]: library('readxl')
        df <- read_excel('eBayAuctions.xls')
        colnames(df)[which(names(df) == "Competitive?")] <- "competitive"
        names(df) <- tolower(names(df))
        summary(df)
        dim(df)
        head(df)
```

```
readxl works best with a newer version of the tibble package.
You currently have tibble v1.4.2.
Falling back to column name repair from tibble <= v1.4.2.
Message displays once per session.
```

```
   category            currency          sellerrating       duration     
 Length:1972        Length:1972        Min.   :    0   Min.   : 1.000  
 Class :character   Class :character   1st Qu.:  595   1st Qu.: 5.000  
 Mode  :character   Mode  :character   Median : 1853   Median : 7.000  
                                       Mean   : 3560   Mean   : 6.486  
                                       3rd Qu.: 3380   3rd Qu.: 7.000  
                                       Max.   :37727   Max.   :10.000  
    endday            closeprice         openprice        competitive    
 Length:1972        Min.   :  0.010   Min.   :  0.01   Min.   :0.0000  
 Class :character   1st Qu.:  4.907   1st Qu.:  1.23   1st Qu.:0.0000  
 Mode  :character   Median :  9.995   Median :  4.50   Median :1.0000  
                    Mean   : 36.449   Mean   : 12.93   Mean   :0.5406  
                    3rd Qu.: 28.000   3rd Qu.:  9.99   3rd Qu.:1.0000  
                    Max.   :999.000   Max.   :999.00   Max.   :1.0000  
```

1. 1972 2. 8

category	currency	sellerrating	duration	endday	closeprice	openprice	competitive
Music/Movie/Game	US	3249	5	Mon	0.01	0.01	0
Music/Movie/Game	US	3249	5	Mon	0.01	0.01	0
Music/Movie/Game	US	3249	5	Mon	0.01	0.01	0
Music/Movie/Game	US	3249	5	Mon	0.01	0.01	0
Music/Movie/Game	US	3249	5	Mon	0.01	0.01	0
Music/Movie/Game	US	3249	5	Mon	0.01	0.01	0

1.0.2 Create Pivot Table and Dummy Columns

```
In [3]: library(reshape)

        # Function to generate Pivot table for a colum
        generatePivotTable <- function(df, col) {
            # Melt the column we want
            df_melt = melt(df, id.vars = c(col), measure.vars = 'competitive')
            # Cast to pivot form
            p_table <- cast(df_melt, paste(paste(col), "~", "variable"), mean)
            # Duplicate the first column so we can merge
            p_table['merge'] <- p_table[1]
            # Number of rows in the table
            len <- dim(p_table[1])
            # Threshold of ratio/mean to use for merging the categorical varaibles
            threshold <- 0.05
            # Merge
            for (i in 1:(len-1)) {
                for (j in (i+1):len){
                    if (abs(p_table[i,2] - p_table[j,2]) < threshold) {
                        p_table[j,3] = p_table[i,3]
                    }
                }
            }
            return (p_table)
        }

        createDummy <- function(x, col) {
            for (level in unique(x[,col])) {
                x[paste('d', col, level, sep = "_")] <- ifelse(x[,col] == level, 1, 0)
            }
            return(x)
        }

        # Columns to check and merge
        columns <- c('category', 'currency', 'endday', 'duration')

        for (col in columns) {
            # Generate Pivot table for col
```

```
    p_table <- generatePivotTable(df, col)
    print(p_table)
    # Merge Rows
    rows <- dim(p_table[1])
    for (i in 1:rows) {
      df[df[paste(col)] == p_table[i,1], paste(col)] = p_table[i,3]
    }
    # Create dummy columns
    df <- createDummy(df, col)
    # Drop the column
    df[, paste(col)] <- NULL
}

head(df)
```

```
Warning message in 1:(len - 1):
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first used
```

```
               category competitive               merge
1     Antique/Art/Craft   0.5649718   Antique/Art/Craft
2            Automotive   0.3539326          Automotive
3                 Books   0.5000000               Books
4   Business/Industrial   0.6666667 Business/Industrial
5  Clothing/Accessories   0.5042017               Books
6          Coins/Stamps   0.2972973        Coins/Stamps
7          Collectibles   0.5774059   Antique/Art/Craft
8              Computer   0.6666667 Business/Industrial
9           Electronics   0.8000000         Electronics
10       EverythingElse   0.2352941      EverythingElse
11        Health/Beauty   0.1718750       Health/Beauty
12          Home/Garden   0.6568627 Business/Industrial
```

```
13            Jewelry   0.3658537         Automotive
14     Music/Movie/Game   0.6029777   Antique/Art/Craft
15        Photography   0.8461538        Electronics
16       Pottery/Glass   0.3500000         Automotive
17       SportingGoods   0.7258065      SportingGoods
18         Toys/Hobbies   0.5299145   Antique/Art/Craft
```

```
Warning message in 1:rows:
numerical expression has 2 elements: only the first usedWarning message in 1:(len - 1):
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first used
```

```
  currency competitive merge
1      EUR   0.5515947   EUR
2      GBP   0.6870748   GBP
3       US   0.5193498   EUR
```

```
Warning message in 1:rows:
numerical expression has 2 elements: only the first usedWarning message in 1:(len - 1):
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first used
```

```
  endday competitive merge
1    Fri   0.4668990   Fri
2    Mon   0.6733577   Mon
3    Sat   0.4273504   Fri
4    Sun   0.4852071   Fri
5    Thu   0.6039604   Thu
6    Tue   0.5321637   Fri
7    Wed   0.4800000   Fri
```

```
Warning message in 1:rows:
numerical expression has 2 elements: only the first usedWarning message in 1:(len - 1):
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first usedWarning message in (i + 1):len:
numerical expression has 2 elements: only the first used
```

```
  duration competitive merge
1        1   0.5217391     1
```

```
2        3   0.4507042     3
3        5   0.6866953     5
4        7   0.4891417     3
5       10   0.5445545     1


Warning message in 1:rows:
numerical expression has 2 elements: only the first used
```

sellerrating	closeprice	openprice	competitive	d_category_Antique/Art/Craft	d_category_Automo
3249	0.01	0.01	0	1	1
3249	0.01	0.01	0	0	0
3249	0.01	0.01	0	0	0
3249	0.01	0.01	0	0	0
3249	0.01	0.01	0	0	0
3249	0.01	0.01	0	0	0

1.0.3 Split Data into train/test

```
In [4]: ## 60% of the sample size
        smp_size <- floor(0.60 * nrow(df))

        ## set the seed to make your partition reproducible
        set.seed(123)
        train_ind <- sample(seq_len(nrow(df)), size = smp_size)

        train <- df[train_ind, ]
        test <- df[-train_ind, ]
```

1.0.4 Train fit.all

```
In [5]: fit.all <- glm(`competitive` ~., family = binomial(link = 'logit'), data = train)

Warning message:
glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
In [6]: summary(fit.all)


Call:
glm(formula = competitive ~ ., family = binomial(link = "logit"),
    data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-4.7429  -0.9375   0.0001   0.9968   2.4957

Coefficients: (13 not defined because of singularities)
                                   Estimate Std. Error z value Pr(>|z|)
```

```
(Intercept)                      -2.220e-01  1.349e-01  -1.645   0.0999 .
sellerrating                     -2.305e-05  1.330e-05  -1.733   0.0831 .
closeprice                        1.355e-01  1.310e-02  10.341   <2e-16 ***
openprice                        -1.512e-01  1.378e-02 -10.969   <2e-16 ***
`d_category_Antique/Art/Craft`   -2.976e-02  2.193e-01  -0.136   0.8921
d_category_Automotive                    NA         NA      NA       NA
d_category_SportingGoods                 NA         NA      NA       NA
`d_category_Business/Industrial`         NA         NA      NA       NA
d_category_Books                         NA         NA      NA       NA
d_category_Electronics                   NA         NA      NA       NA
d_category_EverythingElse                NA         NA      NA       NA
`d_category_Coins/Stamps`                NA         NA      NA       NA
`d_category_Health/Beauty`               NA         NA      NA       NA
d_currency_EUR                   -2.153e-01  1.357e-01  -1.587   0.1126
d_currency_GBP                           NA         NA      NA       NA
d_endday_Mon                     -1.813e-02  1.542e-01  -0.118   0.9064
d_endday_Fri                             NA         NA      NA       NA
d_endday_Thu                             NA         NA      NA       NA
d_duration_5                     -1.237e-01  1.585e-01  -0.780   0.4353
d_duration_3                             NA         NA      NA       NA
d_duration_1                             NA         NA      NA       NA
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1636.4  on 1182  degrees of freedom
Residual deviance: 1258.3  on 1175  degrees of freedom
AIC: 1274.3

Number of Fisher Scoring iterations: 8
```

1.0.5 Predict and check accuracy

```
In [30]: predicted <- predict(fit.all, test, type = 'response')
         predicted <- ifelse(predicted > 0.5, 1, 0)
         accuracy <- mean(test$competitive == predicted)
         print(accuracy)
```

```
Warning message in predict.lm(object, newdata, se.fit, scale = 1, type = ifelse(type == :
prediction from a rank-deficient fit may be misleading

[1] 0.7756654
```

1.0.6 find predictor with highest absolute coefficient value

```
In [22]: coef = fit.all$coefficients
         print(coef)
```

```
                   (Intercept)                     sellerrating
                 -2.220110e-01                    -2.304733e-05
                    closeprice                        openprice
                  1.354970e-01                    -1.511612e-01
`d_category_Antique/Art/Craft`            d_category_Automotive
                 -2.975724e-02                               NA
      d_category_SportingGoods `d_category_Business/Industrial`
                            NA                               NA
              d_category_Books           d_category_Electronics
                            NA                               NA
     d_category_EverythingElse         `d_category_Coins/Stamps`
                            NA                               NA
    `d_category_Health/Beauty`                   d_currency_EUR
                            NA                    -2.153414e-01
                d_currency_GBP                     d_endday_Mon
                            NA                    -1.812749e-02
                  d_endday_Fri                     d_endday_Thu
                            NA                               NA
                  d_duration_5                     d_duration_3
                 -1.236763e-01                               NA
                  d_duration_1
                            NA
```

```
In [23]: sort(abs(coef))
```

sellerrating 2.30473328886959e-05 **d_endday_Mon** 0.0181274874325235
‘d_category_Antique/Art/Craft‘ 0.029757244212542 **d_duration_5** 0.123676280673723
closeprice 0.135496990839086 **openprice** 0.151161171550078 **d_currency_EUR**
0.215341437564903 **(Intercept)** 0.222011004707867

1.0.7 Train fit.single

```
In [24]: max = 0
         name = names(coef)[1]
         index = 1;
         for (i in 2:length(coef)) {
             val = abs(as.numeric(coef[i]))
             if (!is.na(val) && val > abs(as.numeric(max))) {
                 name = names(coef)[i]
                 if (name != '(Intercept)') {
                     max = coef[i]
                     index = i
                 }
```

```
            }
        }

        print(max)
        print(name)

        subset = c("competitive", name)
        fit.single = glm(competitive ~., family = binomial(link='logit'), data = train[subset]
```

```
d_currency_EUR
    -0.2153414
[1] "d_currency_EUR"
```

```
In [25]: summary(fit.single)
```

```
Call:
glm(formula = competitive ~ ., family = binomial(link = "logit"),
    data = train[subset])

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.261  -1.189   1.096   1.166   1.166

Coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)     0.19549    0.08308   2.353   0.0186 *
d_currency_EUR -0.16873    0.11659  -1.447   0.1478
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1636.4  on 1182  degrees of freedom
Residual deviance: 1634.3  on 1181  degrees of freedom
AIC: 1638.3

Number of Fisher Scoring iterations: 3
```

1.0.8 Find Significant predictors

```
In [26]: significance_level = 0.05

         coefs = summary(fit.all)$coefficients
         significant_predictors = coefs[coefs[,4] < significance_level,]
         print(significant_predictors)
```

```
             Estimate Std. Error   z value     Pr(>|z|)
closeprice  0.1354970 0.01310350  10.34052 4.620204e-25
openprice  -0.1511612 0.01378103 -10.96879 5.399195e-28
```

1.0.9 Train fit.reduced

```
In [27]: subset = names(significant_predictors[,1])
         subset = c('competitive', subset)
         fit.reduced = glm(competitive ~., family = binomial(link='logit'), data = train[subse
         summary(fit.reduced)
```

```
Warning message:
glm.fit: fitted probabilities numerically 0 or 1 occurred


Call:
glm(formula = competitive ~ ., family = binomial(link = "logit"),
    data = train[subset])

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-4.7397  -0.9455   0.0001   1.0102   2.5705

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.46376    0.08590  -5.399 6.72e-08 ***
closeprice   0.13790    0.01317  10.467  < 2e-16 ***
openprice   -0.15364    0.01386 -11.083  < 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: 1636.4  on 1182  degrees of freedom
Residual deviance: 1264.7  on 1180  degrees of freedom
AIC: 1270.7

Number of Fisher Scoring iterations: 8
```

1.0.10 Anova

```
In [28]: anova(fit.reduced, fit.all, test='Chisq')
```

Resid. Df	Resid. Dev	Df	Deviance	Pr(>Chi)
1180	1264.706	NA	NA	NA
1175	1258.281	5	6.425043	0.2670283

1.0.11 Over-Dispersion test

```
In [29]: library(qcc)
         s = rep(length(train$competitive), length(train$competitive))
         qcc.overdispersion.test(train$competitive, size = s, type="binomial")
```

```
Package 'qcc' version 2.7
Type 'citation("qcc")' for citing this R package in publications.
```

	Obs.Var/Theor.Var	Statistic	p-value
binomial data	0.4731382	559.2494	1