Kaggle Playground Series - Season 3, Episode 11

Jeremy Haakenson

2023-03-23

R Markdown

This file shows my approach to a Kaggle competition where the goal was to predict cost based on 16 variables.

Load packages.

```
library(splines)
library(gam)
```

```
## Loading required package: foreach
```

```
## Loaded gam 1.22-1
```

```
library(ranger)
library(gbm)
```

```
## Loaded gbm 2.1.8.1
```

```
library(e1071)
```

Load files.

```
train = read.csv('train.csv')
test = read.csv('test.csv')
```

Examine data.

```
summary(train)
```

```
##        id          store_sales.in.millions. unit_sales.in.millions.
##  Min.   :     0   Min.   : 0.510          Min.   :1.000
##  1st Qu.: 90084   1st Qu.: 3.720          1st Qu.:3.000
##  Median :180168   Median : 5.780          Median :3.000
##  Mean   :180168   Mean   : 6.337          Mean   :3.044
##  3rd Qu.:270251   3rd Qu.: 8.400          3rd Qu.:4.000
##  Max.   :360335   Max.   :22.920          Max.   :6.000
##  total_children  num_children_at_home avg_cars_at.home.approx..1
##  Min.   :0.000   Min.   :0.0000       Min.   :0.000
##  1st Qu.:1.000   1st Qu.:0.0000       1st Qu.:1.000
```

```
##  Median :2.000   Median :0.0000       Median :2.000
##  Mean   :2.456   Mean   :0.6894       Mean   :2.204
##  3rd Qu.:4.000   3rd Qu.:1.0000       3rd Qu.:3.000
##  Max.   :5.000   Max.   :5.0000       Max.   :4.000
##   gross_weight   recyclable_package    low_fat       units_per_case
##  Min.   : 6.00   Min.   :0.0000     Min.   :0.0000   Min.   : 1.00
##  1st Qu.: 9.71   1st Qu.:0.0000     1st Qu.:0.0000   1st Qu.:10.00
##  Median :13.60   Median :1.0000     Median :0.0000   Median :20.00
##  Mean   :13.82   Mean   :0.5681     Mean   :0.3278   Mean   :18.97
##  3rd Qu.:17.70   3rd Qu.:1.0000     3rd Qu.:1.0000   3rd Qu.:28.00
##  Max.   :21.90   Max.   :1.0000     Max.   :1.0000   Max.   :36.00
##    store_sqft      coffee_bar      video_store       salad_bar
##  Min.   :20319   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:23593   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :27694   Median :1.0000   Median :0.0000   Median :1.0000
##  Mean   :28180   Mean   :0.5648   Mean   :0.2774   Mean   :0.5048
##  3rd Qu.:33858   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :39696   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##  prepared_food       florist            cost
##  Min.   :0.0000   Min.   :0.0000   Min.   : 50.79
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 70.32
##  Median :1.0000   Median :1.0000   Median : 98.81
##  Mean   :0.5048   Mean   :0.5032   Mean   : 99.61
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:126.62
##  Max.   :1.0000   Max.   :1.0000   Max.   :149.75
```

```
str(train)
```

```
## 'data.frame':    360336 obs. of  17 variables:
##  $ id                        : int  0 1 2 3 4 5 6 7 8 9 ...
##  $ store_sales.in.millions.  : num  8.61 5 14.08 4.02 2.13 ...
##  $ unit_sales.in.millions.   : num  3 2 4 3 3 4 2 3 3 4 ...
##  $ total_children            : num  2 4 0 5 5 5 1 2 5 1 ...
##  $ num_children_at_home      : num  2 0 0 0 0 5 0 0 0 0 ...
##  $ avg_cars_at.home.approx..1: num  2 3 3 0 3 3 2 2 2 3 ...
##  $ gross_weight              : num  10.3 6.66 21.3 14.8 17 7.26 9.58 16.9 13.8 15.7 ...
##  $ recyclable_package        : num  1 1 1 0 1 0 0 1 1 1 ...
##  $ low_fat                   : num  0 0 0 1 1 1 0 0 0 1 ...
##  $ units_per_case            : num  32 1 26 36 20 5 6 2 6 9 ...
##  $ store_sqft                : num  36509 28206 21215 21215 27694 ...
##  $ coffee_bar                : num  0 1 1 1 1 1 1 1 0 1 ...
##  $ video_store               : num  0 0 0 0 1 0 1 1 0 1 ...
##  $ salad_bar                 : num  0 0 0 0 1 1 1 1 0 1 ...
##  $ prepared_food             : num  0 0 0 0 1 1 1 1 0 1 ...
##  $ florist                   : num  0 0 0 0 1 1 1 1 0 1 ...
##  $ cost                      : num  62.1 121.8 83.5 66.8 111.5 ...
```

```
summary(test)
```

```
##        id         store_sales.in.millions. unit_sales.in.millions.
##  Min.   :360336   Min.   : 0.510          Min.   :1.000
##  1st Qu.:420392   1st Qu.: 3.750          1st Qu.:3.000
##  Median :480448   Median : 5.800          Median :3.000
```

```
##  Mean   :480448   Mean   : 6.354          Mean   :3.044  
##  3rd Qu.:540503   3rd Qu.: 8.400          3rd Qu.:4.000  
##  Max.   :600559   Max.   :22.920          Max.   :6.000  
##  total_children  num_children_at_home avg_cars_at.home.approx..1
##  Min.   :0.000   Min.   :0.0000       Min.   :0.000             
##  1st Qu.:1.000   1st Qu.:0.0000       1st Qu.:1.000             
##  Median :2.000   Median :0.0000       Median :2.000             
##  Mean   :2.454   Mean   :0.6854       Mean   :2.198             
##  3rd Qu.:4.000   3rd Qu.:1.0000       3rd Qu.:3.000             
##  Max.   :5.000   Max.   :5.0000       Max.   :4.000             
##   gross_weight   recyclable_package    low_fat       units_per_case 
##  Min.   : 6.00   Min.   :0.0000     Min.   :0.0000   Min.   : 1.00  
##  1st Qu.: 9.71   1st Qu.:0.0000     1st Qu.:0.0000   1st Qu.:10.00  
##  Median :13.60   Median :1.0000     Median :0.0000   Median :20.00  
##  Mean   :13.83   Mean   :0.5657     Mean   :0.3269   Mean   :18.96  
##  3rd Qu.:17.80   3rd Qu.:1.0000     3rd Qu.:1.0000   3rd Qu.:28.00  
##  Max.   :21.90   Max.   :1.0000     Max.   :1.0000   Max.   :36.00  
##    store_sqft      coffee_bar      video_store       salad_bar     
##  Min.   :20319   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000  
##  1st Qu.:23593   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000  
##  Median :27694   Median :1.0000   Median :0.0000   Median :1.0000  
##  Mean   :28175   Mean   :0.5642   Mean   :0.2756   Mean   :0.5044  
##  3rd Qu.:33858   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000  
##  Max.   :39696   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000  
##  prepared_food       florist      
##  Min.   :0.0000   Min.   :0.0000  
##  1st Qu.:0.0000   1st Qu.:0.0000  
##  Median :1.0000   Median :1.0000  
##  Mean   :0.5044   Mean   :0.5018  
##  3rd Qu.:1.0000   3rd Qu.:1.0000  
##  Max.   :1.0000   Max.   :1.0000
```

```
str(test)
```

```
## 'data.frame':    240224 obs. of  16 variables:
##  $ id                        : int  360336 360337 360338 360339 360340 360341 360342 360343 360344 36
##  $ store_sales.in.millions.  : num  7.24 6.9 8.34 5.48 4.8 5.25 3.72 7.68 9.63 9.44 ...
##  $ unit_sales.in.millions.   : num  4 2 3 2 3 3 4 4 3 4 ...
##  $ total_children            : num  1 2 0 3 2 1 4 2 3 5 ...
##  $ num_children_at_home      : num  0 2 0 3 0 1 0 0 1 0 ...
##  $ avg_cars_at.home.approx..1: num  2 3 3 2 2 3 4 3 4 3 ...
##  $ gross_weight              : num  10.8 8.51 8.77 21.9 10.9 13.2 14.4 19.9 7.87 8.83 ...
##  $ recyclable_package        : num  0 1 0 1 1 1 1 0 1 1 ...
##  $ low_fat                   : num  1 0 1 0 0 0 0 0 0 1 ...
##  $ units_per_case            : num  7 4 14 9 11 9 4 20 6 29 ...
##  $ store_sqft                : num  20319 33858 39696 23688 27694 ...
##  $ coffee_bar                : num  0 1 0 1 1 1 1 1 1 0 ...
##  $ video_store               : num  0 0 0 1 1 1 0 1 0 0 ...
##  $ salad_bar                 : num  0 1 1 1 1 1 1 1 1 0 ...
##  $ prepared_food             : num  0 1 1 1 1 1 1 1 1 0 ...
##  $ florist                   : num  0 1 0 1 1 1 1 1 1 1 ...
```

recyclable_package, low_fat, coffee_bar, video_store, salad_bar, prepared_food, and florist are binary variables.

id should be omitted from any model.

Look for correlation between variables.

```
cor(train[, -1])
```

```
##                            store_sales.in.millions. unit_sales.in.millions.
## store_sales.in.millions.                1.000000000            0.4813757652
## unit_sales.in.millions.                 0.481375765            1.0000000000
## total_children                          0.069303226            0.1132311894
## num_children_at_home                    0.029261049            0.0467545583
## avg_cars_at.home.approx..1              0.006794259            0.0171189249
## gross_weight                            0.038158576            0.0004667436
## recyclable_package                      0.030389878            0.0010739744
## low_fat                                -0.008735475           -0.0036626600
## units_per_case                         -0.009893103            0.0011405670
## store_sqft                              0.021571915            0.0455398313
## coffee_bar                             -0.040039499           -0.0769673037
## video_store                             0.029564152            0.0537948590
## salad_bar                               0.044831915            0.0824451025
## prepared_food                           0.044854158            0.0824847913
## florist                                 0.046471926            0.0830621888
## cost                                   -0.012386967           -0.0265087663
##                            total_children num_children_at_home
## store_sales.in.millions.     0.0693032262         0.0292610494
## unit_sales.in.millions.      0.1132311894         0.0467545583
## total_children               1.0000000000         0.3592070613
## num_children_at_home         0.3592070613         1.0000000000
## avg_cars_at.home.approx..1   0.0785191739         0.1154756738
## gross_weight                -0.0009077197        -0.0006014972
## recyclable_package           0.0022356359         0.0061042640
## low_fat                     -0.0015974132        -0.0002076602
## units_per_case              -0.0002668987        -0.0041741829
## store_sqft                  -0.0089908507         0.0057847790
## coffee_bar                  -0.0064764368        -0.0232337747
## video_store                 -0.0133034887        -0.0207378304
## salad_bar                   -0.0235637940        -0.0311088116
## prepared_food               -0.0235604814        -0.0310503584
## florist                     -0.0125453620        -0.0178879497
## cost                        -0.0074816998        -0.0017271315
##                            avg_cars_at.home.approx..1  gross_weight
## store_sales.in.millions.                  0.006794259  0.0381585756
## unit_sales.in.millions.                   0.017118925  0.0004667436
## total_children                            0.078519174 -0.0009077197
## num_children_at_home                      0.115475674 -0.0006014972
## avg_cars_at.home.approx..1                1.000000000 -0.0022671343
## gross_weight                             -0.002267134  1.0000000000
## recyclable_package                        0.004020787  0.0590504265
## low_fat                                  -0.001912331 -0.0334207253
## units_per_case                            0.001190268 -0.0176090654
## store_sqft                               -0.020031687 -0.0004551021
## coffee_bar                               -0.000440755  0.0008351537
## video_store                               0.012702076 -0.0008447703
## salad_bar                                -0.013102435  0.0021163179
```

```
## prepared_food                             -0.013035045  0.0021547017
## florist                                   -0.004284011  0.0007635955
## cost                                       0.027097743 -0.0001161770
##                            recyclable_package       low_fat units_per_case
## store_sales.in.millions.         0.0303898784 -0.0087354751  -0.0098931034
## unit_sales.in.millions.          0.0010739744 -0.0036626600   0.0011405670
## total_children                   0.0022356359 -0.0015974132  -0.0002668987
## num_children_at_home             0.0061042640 -0.0002076602  -0.0041741829
## avg_cars_at.home.approx..1       0.0040207874 -0.0019123314   0.0011902682
## gross_weight                     0.0590504265 -0.0334207253  -0.0176090654
## recyclable_package               1.0000000000 -0.0300252967  -0.0030280744
## low_fat                         -0.0300252967  1.0000000000   0.0302257210
## units_per_case                  -0.0030280744  0.0302257210   1.0000000000
## store_sqft                      -0.0003428107  0.0019716790   0.0022966522
## coffee_bar                       0.0040368730  0.0026217321   0.0008282796
## video_store                      0.0041285191  0.0028927992   0.0005688030
## salad_bar                        0.0046735853  0.0056020164   0.0016977467
## prepared_food                    0.0046723238  0.0055671342   0.0016989680
## florist                          0.0048485629  0.0055401218   0.0005564609
## cost                            -0.0014548686 -0.0019750435   0.0001803538
##                               store_sqft    coffee_bar   video_store
## store_sales.in.millions.    0.0215719146 -0.0400394985  0.0295641521
## unit_sales.in.millions.     0.0455398313 -0.0769673037  0.0537948590
## total_children             -0.0089908507 -0.0064764368 -0.0133034887
## num_children_at_home        0.0057847790 -0.0232337747 -0.0207378304
## avg_cars_at.home.approx..1 -0.0200316872 -0.0004407550  0.0127020757
## gross_weight               -0.0004551021  0.0008351537 -0.0008447703
## recyclable_package         -0.0003428107  0.0040368730  0.0041285191
## low_fat                     0.0019716790  0.0026217321  0.0028927992
## units_per_case              0.0022966522  0.0008282796  0.0005688030
## store_sqft                  1.0000000000 -0.1982428101 -0.0838731679
## coffee_bar                 -0.1982428101  1.0000000000  0.5438258345
## video_store                -0.0838731679  0.5438258345  1.0000000000
## salad_bar                   0.3330547104  0.4812480973  0.6136401003
## prepared_food               0.3331023373  0.4812250020  0.6136094442
## florist                    -0.0741569709  0.5541826640  0.6154645610
## cost                       -0.0492006363 -0.0520856748 -0.1067861697
##                               salad_bar prepared_food      florist
## store_sales.in.millions.    0.044831915   0.044854158  0.0464719263
## unit_sales.in.millions.     0.082445102   0.082484791  0.0830621888
## total_children             -0.023563794  -0.023560481 -0.0125453620
## num_children_at_home       -0.031108812  -0.031050358 -0.0178879497
## avg_cars_at.home.approx..1 -0.013102435  -0.013035045 -0.0042840111
## gross_weight                0.002116318   0.002154702  0.0007635955
## recyclable_package          0.004673585   0.004672324  0.0048485629
## low_fat                     0.005602016   0.005567134  0.0055401218
## units_per_case              0.001697747   0.001698968  0.0005564609
## store_sqft                  0.333054710   0.333102337 -0.0741569709
## coffee_bar                  0.481248097   0.481225002  0.5541826640
## video_store                 0.613640100   0.613609444  0.6154645610
## salad_bar                   1.000000000   0.999839025  0.5986530671
## prepared_food               0.999839025   1.000000000  0.5986474858
## florist                     0.598653067   0.598647486  1.0000000000
## cost                       -0.098810123  -0.098843199 -0.1104140444
```

```
##                                   cost
## store_sales.in.millions.   -0.0123869670
## unit_sales.in.millions.    -0.0265087663
## total_children             -0.0074816998
## num_children_at_home       -0.0017271315
## avg_cars_at.home.approx..1  0.0270977425
## gross_weight               -0.0001161770
## recyclable_package         -0.0014548686
## low_fat                    -0.0019750435
## units_per_case              0.0001803538
## store_sqft                 -0.0492006363
## coffee_bar                 -0.0520856748
## video_store                -0.1067861697
## salad_bar                  -0.0988101229
## prepared_food              -0.0988431992
## florist                    -0.1104140444
## cost                        1.0000000000
```

salad_bar and prepared_food are highly correlated.

Initial linear regression.

```r
lmmod = lm(cost ~ . - id, data = train)
summary(lmmod)
```

```
##
## Call:
## lm(formula = cost ~ . - id, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -62.445 -26.422   0.094  26.098  58.070
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 1.115e+02  4.196e-01 265.615  < 2e-16 ***
## store_sales.in.millions.    1.651e-02  1.706e-02   0.968   0.3333
## unit_sales.in.millions.    -4.738e-01  7.311e-02  -6.481 9.11e-11 ***
## total_children             -1.925e-01  3.579e-02  -5.380 7.45e-08 ***
## num_children_at_home       -7.286e-02  4.378e-02  -1.664   0.0961 .
## avg_cars_at.home.approx..1  7.718e-01  4.592e-02  16.809  < 2e-16 ***
## gross_weight               -1.087e-03  1.074e-02  -0.101   0.9193
## recyclable_package         -6.174e-02  9.997e-02  -0.618   0.5369
## low_fat                    -8.444e-02  1.054e-01  -0.801   0.4229
## units_per_case              1.159e-03  4.839e-03   0.240   0.8106
## store_sqft                 -3.061e-04  1.028e-05 -29.764  < 2e-16 ***
## coffee_bar                  1.045e+00  1.344e-01   7.776 7.52e-15 ***
## video_store                -5.151e+00  1.608e-01 -32.036  < 2e-16 ***
## salad_bar                   6.628e+00  5.506e+00   1.204   0.2287
## prepared_food              -5.969e+00  5.506e+00  -1.084   0.2784
## florist                    -4.958e+00  1.421e-01 -34.888  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 29.65 on 360320 degrees of freedom
## Multiple R-squared:  0.01953,    Adjusted R-squared:  0.01949
## F-statistic: 478.5 on 15 and 360320 DF,  p-value: < 2.2e-16
```

salad_bar has a higher absolute coefficient, so I will exclude prepared_food from future models.

Look for interactions.

```
lmmod2 = lm(cost ~ (unit_sales.in.millions. +
                      total_children + avg_cars_at.home.approx..1 +
                      store_sqft + coffee_bar + video_store +
                      florist)^2, data = train)
summary(lmmod2)
```

```
##
## Call:
## lm(formula = cost ~ (unit_sales.in.millions. + total_children +
##     avg_cars_at.home.approx..1 + store_sqft + coffee_bar + video_store +
##     florist)^2, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -61.148 -25.007   0.818  26.145  70.951
##
## Coefficients: (1 not defined because of singularities)
##                                                     Estimate Std. Error
## (Intercept)                                        1.073e+02  1.382e+00
## unit_sales.in.millions.                           -8.868e-01  3.922e-01
## total_children                                     3.089e-01  2.278e-01
## avg_cars_at.home.approx..1                        -2.209e-01  3.031e-01
## store_sqft                                        -1.154e-04  4.139e-05
## coffee_bar                                        -8.066e+00  1.134e+00
## video_store                                       -1.800e+01  8.012e+00
## florist                                            2.242e+01  1.292e+00
## unit_sales.in.millions.:total_children            -6.934e-02  4.280e-02
## unit_sales.in.millions.:avg_cars_at.home.approx..1  2.936e-02  6.024e-02
## unit_sales.in.millions.:store_sqft                 1.566e-05  1.148e-05
## unit_sales.in.millions.:coffee_bar                -2.370e-01  1.587e-01
## unit_sales.in.millions.:video_store               -5.543e-01  2.005e-01
## unit_sales.in.millions.:florist                    1.137e+00  1.721e-01
## total_children:avg_cars_at.home.approx..1          1.135e-01  3.086e-02
## total_children:store_sqft                         -1.820e-05  5.705e-06
## total_children:coffee_bar                          5.290e-02  8.828e-02
## total_children:video_store                         1.114e+00  1.010e-01
## total_children:florist                            -7.127e-01  9.186e-02
## avg_cars_at.home.approx..1:store_sqft              4.690e-06  7.850e-06
## avg_cars_at.home.approx..1:coffee_bar              7.849e-01  1.191e-01
## avg_cars_at.home.approx..1:video_store            -2.247e+00  1.356e-01
## avg_cars_at.home.approx..1:florist                 1.275e+00  1.230e-01
## store_sqft:coffee_bar                              3.869e-04  3.853e-05
## store_sqft:video_store                            -2.711e-04  4.073e-05
## store_sqft:florist                                -1.316e-03  4.495e-05
## coffee_bar:video_store                                    NA         NA
## coffee_bar:florist                                 5.264e+00  3.438e-01
```

```
## video_store:florist                                  2.155e+01  7.898e+00
##                                                     t value Pr(>|t|)
## (Intercept)                                          77.645  < 2e-16 ***
## unit_sales.in.millions.                              -2.261 0.023762 *
## total_children                                        1.356 0.175236
## avg_cars_at.home.approx..1                           -0.729 0.466143
## store_sqft                                           -2.788 0.005309 **
## coffee_bar                                           -7.114 1.13e-12 ***
## video_store                                          -2.246 0.024682 *
## florist                                              17.354  < 2e-16 ***
## unit_sales.in.millions.:total_children               -1.620 0.105262
## unit_sales.in.millions.:avg_cars_at.home.approx..1    0.487 0.626003
## unit_sales.in.millions.:store_sqft                    1.364 0.172450
## unit_sales.in.millions.:coffee_bar                   -1.494 0.135275
## unit_sales.in.millions.:video_store                  -2.765 0.005699 **
## unit_sales.in.millions.:florist                       6.609 3.87e-11 ***
## total_children:avg_cars_at.home.approx..1             3.677 0.000236 ***
## total_children:store_sqft                            -3.190 0.001423 **
## total_children:coffee_bar                             0.599 0.549064
## total_children:video_store                           11.035  < 2e-16 ***
## total_children:florist                               -7.758 8.63e-15 ***
## avg_cars_at.home.approx..1:store_sqft                 0.597 0.550214
## avg_cars_at.home.approx..1:coffee_bar                 6.591 4.38e-11 ***
## avg_cars_at.home.approx..1:video_store              -16.563  < 2e-16 ***
## avg_cars_at.home.approx..1:florist                   10.369  < 2e-16 ***
## store_sqft:coffee_bar                                10.042  < 2e-16 ***
## store_sqft:video_store                               -6.656 2.83e-11 ***
## store_sqft:florist                                  -29.280  < 2e-16 ***
## coffee_bar:video_store                                   NA       NA
## coffee_bar:florist                                   15.313  < 2e-16 ***
## video_store:florist                                   2.728 0.006367 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 29.53 on 360308 degrees of freedom
## Multiple R-squared:  0.02751,    Adjusted R-squared:  0.02743
## F-statistic: 377.4 on 27 and 360308 DF,  p-value: < 2.2e-16
```

Make final linear regression model.

```r
lmmod3 = lm(cost ~ unit_sales.in.millions. + total_children + avg_cars_at.home.approx..1 +
              store_sqft + coffee_bar + video_store +
              florist + unit_sales.in.millions.:video_store +
              unit_sales.in.millions.:florist + total_children:avg_cars_at.home.approx..1 +
              total_children:store_sqft + total_children:video_store +
              total_children:florist + avg_cars_at.home.approx..1:coffee_bar +
              avg_cars_at.home.approx..1:video_store +
              avg_cars_at.home.approx..1:florist + store_sqft:coffee_bar +
              store_sqft:video_store + store_sqft:florist + coffee_bar:florist +
              video_store:florist, data = train)
summary(lmmod3)
```

```
##
```

```
## Call:
## lm(formula = cost ~ unit_sales.in.millions. + total_children +
##     avg_cars_at.home.approx..1 + store_sqft + coffee_bar + video_store +
##     florist + unit_sales.in.millions.:video_store + unit_sales.in.millions.:florist +
##     total_children:avg_cars_at.home.approx..1 + total_children:store_sqft +
##     total_children:video_store + total_children:florist + avg_cars_at.home.approx..1:coffee_bar +
##     avg_cars_at.home.approx..1:video_store + avg_cars_at.home.approx..1:florist +
##     store_sqft:coffee_bar + store_sqft:video_store + store_sqft:florist +
##     coffee_bar:florist + video_store:florist, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -61.294 -25.034   0.813  26.135  71.230
##
## Coefficients:
##                                             Estimate Std. Error t value
## (Intercept)                                1.063e+02  6.489e-01 163.808
## unit_sales.in.millions.                   -6.655e-01  8.834e-02  -7.533
## total_children                             1.134e-01  1.780e-01   0.637
## avg_cars_at.home.approx..1                 3.228e-03  1.048e-01   0.031
## store_sqft                                -5.664e-05  1.749e-05  -3.238
## coffee_bar                                -8.724e+00  9.509e-01  -9.175
## video_store                               -1.786e+01  8.012e+00  -2.229
## florist                                    2.273e+01  1.261e+00  18.022
## unit_sales.in.millions.:video_store       -6.492e-01  1.926e-01  -3.371
## unit_sales.in.millions.:florist            1.091e+00  1.677e-01   6.502
## total_children:avg_cars_at.home.approx..1  1.135e-01  3.063e-02   3.707
## total_children:store_sqft                 -1.795e-05  5.519e-06  -3.253
## total_children:video_store                 1.133e+00  9.588e-02  11.821
## total_children:florist                    -7.072e-01  8.636e-02  -8.189
## avg_cars_at.home.approx..1:coffee_bar      7.654e-01  1.160e-01   6.597
## avg_cars_at.home.approx..1:video_store    -2.245e+00  1.354e-01 -16.584
## avg_cars_at.home.approx..1:florist         1.287e+00  1.221e-01  10.548
## store_sqft:coffee_bar                      3.948e-04  3.798e-05  10.394
## store_sqft:video_store                    -2.707e-04  4.072e-05  -6.647
## store_sqft:florist                        -1.324e-03  4.450e-05 -29.760
## coffee_bar:florist                         5.167e+00  3.403e-01  15.185
## video_store:florist                        2.164e+01  7.898e+00   2.740
##                                           Pr(>|t|)
## (Intercept)                                < 2e-16 ***
## unit_sales.in.millions.                   4.96e-14 ***
## total_children                             0.52417
## avg_cars_at.home.approx..1                 0.97543
## store_sqft                                 0.00120 **
## coffee_bar                                 < 2e-16 ***
## video_store                                0.02581 *
## florist                                    < 2e-16 ***
## unit_sales.in.millions.:video_store        0.00075 ***
## unit_sales.in.millions.:florist           7.93e-11 ***
## total_children:avg_cars_at.home.approx..1  0.00021 ***
## total_children:store_sqft                  0.00114 **
## total_children:video_store                 < 2e-16 ***
## total_children:florist                    2.65e-16 ***
## avg_cars_at.home.approx..1:coffee_bar     4.20e-11 ***
```

```
## avg_cars_at.home.approx..1:video_store     < 2e-16 ***
## avg_cars_at.home.approx..1:florist         < 2e-16 ***
## store_sqft:coffee_bar                      < 2e-16 ***
## store_sqft:video_store                    3.00e-11 ***
## store_sqft:florist                         < 2e-16 ***
## coffee_bar:florist                         < 2e-16 ***
## video_store:florist                        0.00614 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 29.53 on 360314 degrees of freedom
## Multiple R-squared:  0.02748,    Adjusted R-squared:  0.02742
## F-statistic: 484.8 on 21 and 360314 DF,  p-value: < 2.2e-16
```

Predict on test data.

```
cost = predict(lmmod3, test)
head(cost)
```

```
##         1         2         3         4         5         6
## 102.45754  97.35886 102.05310  99.66160  93.91672  98.67901
```

```
summary(train$cost)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   50.79   70.32   98.81   99.61  126.62  149.75
```

```
lm.guess = cbind.data.frame(test$id, cost)
head(lm.guess)
```

```
##   test$id      cost
## 1  360336 102.45754
## 2  360337  97.35886
## 3  360338 102.05310
## 4  360339  99.66160
## 5  360340  93.91672
## 6  360341  98.67901
```

```
write.csv(lm.guess, 'lm.csv')
```

Linear regression model Kaggle score = .315. (Lower is better.)

Make spline model.

```
spline1 = lm(cost ~ ns(unit_sales.in.millions.) + ns(total_children) + ns(avg_cars_at.home.approx..1) +
               ns(store_sqft) + ns(coffee_bar) + ns(video_store) +
               ns(florist), data = train)
summary(spline1)
```

```
##
## Call:
```

```
## lm(formula = cost ~ ns(unit_sales.in.millions.) + ns(total_children) +
##     ns(avg_cars_at.home.approx..1) + ns(store_sqft) + ns(coffee_bar) +
##     ns(video_store) + ns(florist), data = train)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -56.63 -26.40   0.08  25.98  57.70
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    104.6394     0.2070 505.535  < 2e-16 ***
## ns(unit_sales.in.millions.)     -2.6496     0.4017  -6.596 4.23e-11 ***
## ns(total_children)              -1.3465     0.2089  -6.445 1.16e-10 ***
## ns(avg_cars_at.home.approx..1)   3.8019     0.2280  16.672  < 2e-16 ***
## ns(store_sqft)                  -6.8329     0.2043 -33.438  < 2e-16 ***
## ns(coffee_bar)                   1.4846     0.1615   9.191  < 2e-16 ***
## ns(video_store)                 -6.1002     0.1839 -33.178  < 2e-16 ***
## ns(florist)                     -5.9443     0.1669 -35.618  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 29.65 on 360328 degrees of freedom
## Multiple R-squared:  0.01947,    Adjusted R-squared:  0.01945
## F-statistic:  1022 on 7 and 360328 DF,  p-value: < 2.2e-16
```

spline1 R-squared = .0194

Predict using spline model.

```
cost = predict(spline1, test)
spline.guess = cbind.data.frame(test$id, cost)
write.csv(spline.guess, 'spline.csv')
```

Kaggle score for spline model = .316

Make GAM model.

```
gam.mod = gam(cost ~ s(unit_sales.in.millions.) + s(total_children) + s(avg_cars_at.home.approx..1) +
                s(store_sqft) + coffee_bar + video_store +
                florist, data = train)
summary(gam.mod)
```

```
##
## Call: gam(formula = cost ~ s(unit_sales.in.millions.) + s(total_children) +
##     s(avg_cars_at.home.approx..1) + s(store_sqft) + coffee_bar +
##     video_store + florist, data = train)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -59.7851 -25.9954   0.7405  25.0822  60.1984
##
## (Dispersion Parameter for gaussian family taken to be 868.053)
##
##     Null Deviance: 322993390 on 360335 degrees of freedom
## Residual Deviance: 312773384 on 360316 degrees of freedom
```

```
## AIC: 3460736
##
## Number of Local Scoring Iterations: NA
##
## Anova for Parametric Effects
##                                   Df    Sum Sq Mean Sq   F value  Pr(>F)
## s(unit_sales.in.millions.)         1    302402  302402  348.3679 < 2e-16 ***
## s(total_children)                  1      4009    4009    4.6181 0.03164 *
## s(avg_cars_at.home.approx..1)      1    257498  257498  296.6391 < 2e-16 ***
## s(store_sqft)                      1    688068  688068  792.6564 < 2e-16 ***
## coffee_bar                         1    400708  400708  461.6174 < 2e-16 ***
## video_store                        1    743678  743678  856.7198 < 2e-16 ***
## florist                            1   1031405 1031405 1188.1824 < 2e-16 ***
## Residuals                     360316 312773384     868
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##                               Npar Df  Npar F     Pr(F)
## (Intercept)
## s(unit_sales.in.millions.)          3   15.78 2.939e-10 ***
## s(total_children)                   3   71.21 < 2.2e-16 ***
## s(avg_cars_at.home.approx..1)       3   56.94 < 2.2e-16 ***
## s(store_sqft)                       3 1379.90 < 2.2e-16 ***
## coffee_bar
## video_store
## florist
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Predict using GAM model.

```
cost = predict(gam.mod, test)
gam.guess = cbind.data.frame(test$id, cost)
write.csv(gam.guess, 'gam.csv')
summary(cost)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   87.29   96.85   99.66   99.63  103.33  111.92
```

Kaggle score = .314

Make random forest model.

```
rf1 = ranger(cost ~ . - id, data = train, importance = 'permutation')
```

```
## Growing trees.. Progress: 51%. Estimated remaining time: 29 seconds.
## Computing permutation importance.. Progress: 49%. Estimated remaining time: 32 seconds.
```

```
summary(rf1)
```

```
##                            Length Class         Mode
```

```
## predictions               360336 -none-        numeric
## num.trees                      1 -none-        numeric
## num.independent.variables      1 -none-        numeric
## mtry                           1 -none-        numeric
## min.node.size                  1 -none-        numeric
## variable.importance           15 -none-        numeric
## prediction.error               1 -none-        numeric
## forest                         7 ranger.forest list
## splitrule                      1 -none-        character
## treetype                       1 -none-        character
## r.squared                      1 -none-        numeric
## call                           4 -none-        call
## importance.mode                1 -none-        character
## num.samples                    1 -none-        numeric
## replace                        1 -none-        logical
```

```
rf1$variable.importance
```

```
##   store_sales.in.millions.    unit_sales.in.millions.
##                  10.703273                  13.061615
##             total_children       num_children_at_home
##                  51.926407                  35.171591
## avg_cars_at.home.approx..1               gross_weight
##                  50.896392                   2.061862
##         recyclable_package                    low_fat
##                   1.064264                   1.174548
##             units_per_case                 store_sqft
##                   1.907494                 235.792467
##                 coffee_bar                video_store
##                 109.183614                  75.263146
##                  salad_bar              prepared_food
##                 184.223360                 166.930836
##                    florist
##                 243.906487
```

```
rf1$r.squared
```

```
## [1] 0.1059016
```

R-squared = .106

Make new RF model.

```
set.seed(21)
rf2 = ranger(cost ~ . -id -recyclable_package -units_per_case -gross_weight
             -low_fat, data = train)
```

```
## Growing trees.. Progress: 68%. Estimated remaining time: 14 seconds.
```

```
rf2$r.squared
```

```
## [1] 0.1136328
```

R-squared = .114

Predict using new RF model.

```
cost = predict(rf2, test)
rf.guess = cbind.data.frame(test$id, cost)
write.csv(rf.guess, 'rf.csv')
```

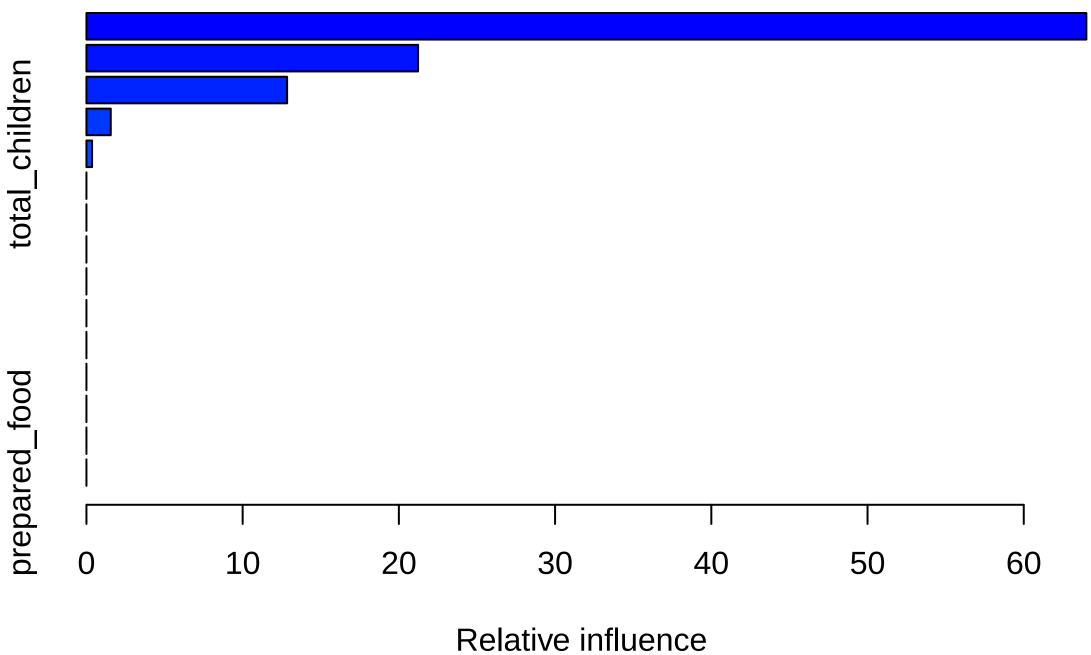
rf2 Kaggle score = .301

Make GBM models.

```
set.seed(21)
gbm1 = gbm(cost ~ . - id, data = train)
```

```
## Distribution not specified, assuming gaussian ...
```

```
summary(gbm1)
```



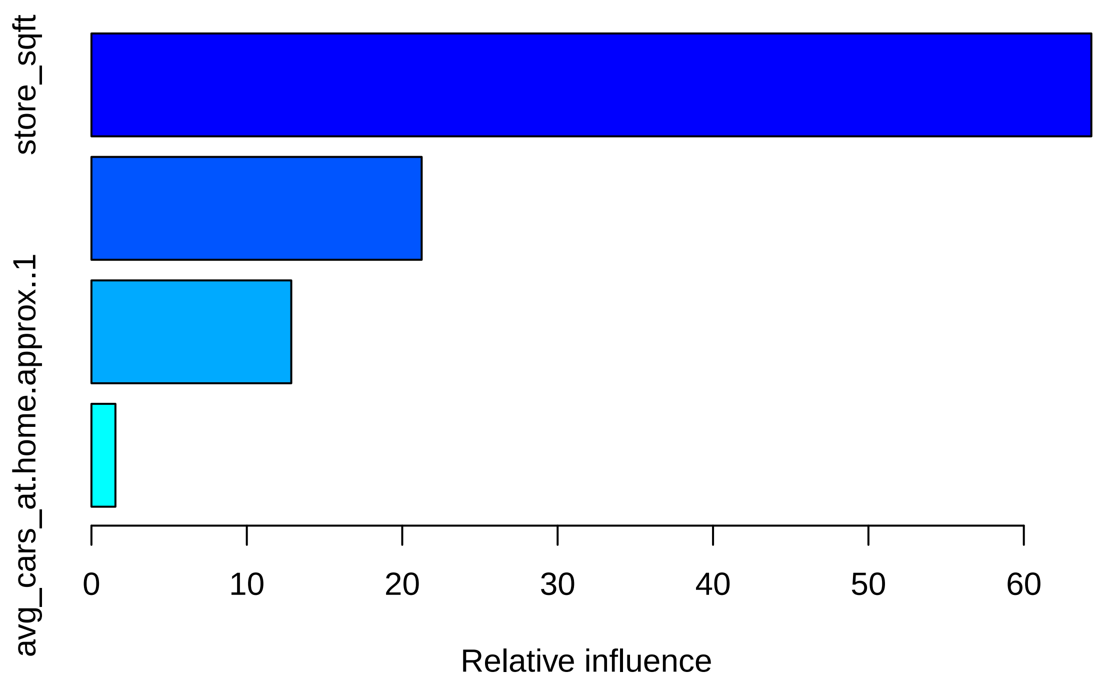

```
##                                                   var    rel.inf
## store_sqft                                 store_sqft 64.0082788
## florist                                       florist 21.2292966
## video_store                               video_store 12.8461256
## avg_cars_at.home.approx..1 avg_cars_at.home.approx..1  1.5563887
## total_children                         total_children  0.3599102
```

```
## store_sales.in.millions.     store_sales.in.millions.  0.0000000
## unit_sales.in.millions.       unit_sales.in.millions.  0.0000000
## num_children_at_home             num_children_at_home  0.0000000
## gross_weight                             gross_weight  0.0000000
## recyclable_package                 recyclable_package  0.0000000
## low_fat                                       low_fat  0.0000000
## units_per_case                         units_per_case  0.0000000
## coffee_bar                                 coffee_bar  0.0000000
## salad_bar                                   salad_bar  0.0000000
## prepared_food                           prepared_food  0.0000000
```

```r
set.seed(21)
gbm2 = gbm(cost ~ store_sqft + florist + video_store +
             avg_cars_at.home.approx..1, data = train)
```

```
## Distribution not specified, assuming gaussian ...
```

```r
summary(gbm2)
```




```
##                                                   var   rel.inf
## store_sqft                                 store_sqft 64.345276
## florist                                       florist 21.251004
## video_store                               video_store 12.859261
## avg_cars_at.home.approx..1 avg_cars_at.home.approx..1  1.544458
```

Predict using GBM model.

```
cost = predict(gbm2, test)
```

```
## Using 100 trees...
```

```
gbm.guess = cbind.data.frame(test$id, cost)
write.csv(gbm.guess, 'gbm.csv')
```

Kaggle score for GBM model = .314

Make SVM model. Default kernel is radial.

```
train.svm = train[1:2402,]

svm1 = svm(cost ~ . - id, data = train.svm, kernel = 'linear')
```

Predict using SVM model.

```
cost = predict(svm1, test)
svm.guess = cbind.data.frame(test$id, cost)
write.csv(svm.guess, 'svm.csv')
```

SVM Kaggle score = .319

Optimize the random forest model, which has been the best model so far.

```
set.seed(23)
rf3 = ranger(cost ~ . -id -recyclable_package -units_per_case -gross_weight
             -low_fat, data = train, num.trees = 1000)
```

```
## Growing trees.. Progress: 34%. Estimated remaining time: 1 minute, 0 seconds.
## Growing trees.. Progress: 69%. Estimated remaining time: 28 seconds.
```

```
rf3$r.squared
```

```
## [1] 0.1132479
```

R-squared = .113. Adding more trees didn't help.

Try different values of mtry.

```
set.seed(23)
rf4 = ranger(cost ~ . -id -recyclable_package -units_per_case -gross_weight
             -low_fat, data = train, mtry = 1)
rf4$r.squared
```

```
## [1] 0.03702256
```

R-squared of rf4 = .038

```r
set.seed(23)
rf5 = ranger(cost ~ . -id -recyclable_package -units_per_case -gross_weight
             -low_fat, data = train, mtry = 2)
rf5$r.squared
```

```
## [1] 0.0788579
```

rf5 R-squared = .079

```r
set.seed(23)
rf6 = ranger(cost ~ . -id -recyclable_package -units_per_case -gross_weight
             -low_fat, data = train, mtry = 4)
```

```
## Growing trees.. Progress: 47%. Estimated remaining time: 35 seconds.
## Growing trees.. Progress: 95%. Estimated remaining time: 3 seconds.
```

```r
rf6$r.squared
```

```
## [1] 0.1283611
```

rf6 R-squared = .128

```r
set.seed(23)
rf7 = ranger(cost ~ . -id -recyclable_package -units_per_case -gross_weight
             -low_fat, data = train, mtry = 8)
```

```
## Growing trees.. Progress: 20%. Estimated remaining time: 2 minutes, 7 seconds.
## Growing trees.. Progress: 41%. Estimated remaining time: 1 minute, 30 seconds.
## Growing trees.. Progress: 62%. Estimated remaining time: 57 seconds.
## Growing trees.. Progress: 83%. Estimated remaining time: 25 seconds.
```

```r
rf7$r.squared
```

```
## [1] 0.05681667
```

rf7 R-squared = .057

The best R-squared was achieved with an mtry of 4.

Use rf6 to predict.

```r
cost = predict(rf6, test)
final.guess = cbind.data.frame(test$id, cost)
write.csv(final.guess, 'rf6.csv')
```

rf6 Kaggle score = .298. (Lower is better.) As of March 23rd, 2023, this puts me in 151st place out of 386.