Assignment 3: Promotions Management

Group 11: Christina Wang, Kailin Fu, Shun Guan, Sylvia Lu, Yiran Huang

April 20, 2020

Contents

1 Promotional event planning

1. Evidence for strong seasonal demand There is a strong seasonal demand for this product. Per event summary, the product has a much higher demand around the time of event 2. The base case for event 2 is 1,360, a 87% increase from event 1, a 469% increase from event 3, a 203% increase from event 4, and a 308% increase from event 5. Similarly, the base demand around event 1 is also higher than other events except event 2. The higher base demands around the time of event 2 and event 1 show a strong seasonal demand.

2. Incremental sales response

```
eventsummary = data.frame(baseQ = c(728, 1360), baseP = c(2.31, 2.31), promoQ = c(1129, 2303), promoP =
eventsummary$baseRev = eventsummary$baseQ*eventsummary$baseP
eventsummary$promoRev = eventsummary$promoQ*eventsummary$promoP
eventsummary$incrRevFrac = (eventsummary$promoRev - eventsummary$baseRev)/eventsummary$baseRev
print(eventsummary)
```

```
  baseQ baseP promoQ promoP baseRev promoRev incrRevFrac
1   728  2.31   1129   1.99 1681.68  2246.71   0.3359914
2  1360  2.31   2303   2.31 3141.60  5319.93   0.6933824
```

Per summary Table, the incremental sales response for event 1 is 33.60% and for event 2 is 69.33%.

3. Profitability results From ROI per event summary, event 5 is the most profitable with a ROI of 53%. Event 2 is the second, with a ROI of 44%. Event 1 does not have a profitable result with the promotion, with a negative ROI of -2%. Event 3, and 4 are even worse with much more negative ROIs of -22% and -79%.

4. The profitability with forward buying

```
eventsummary2 = data.frame(incrContr = c(8019, 18874), VC = c(4740, 9674), FC = c(2500, 2500))
eventsummary2$ForwardBuyCost = c(962*2, 962*2)
eventsummary2$eventCost = eventsummary2$VC + eventsummary2$FC + eventsummary2$ForwardBuyCost
eventsummary2$grossContr = eventsummary2$incrContr - eventsummary2$eventCost
eventsummary2$ROI = eventsummary2$grossContr/eventsummary2$incrContr
print(eventsummary2)
```

```
  incrContr   VC   FC ForwardBuyCost eventCost grossContr        ROI
1      8019 4740 2500           1924      9164      -1145 -0.1427859
2     18874 9674 2500           1924     14098       4776  0.2530465
```

The profit for event 1 will be -1145 with a ROI of -14.28% and for event 2 will be 4776 with a ROI of 25.30%.

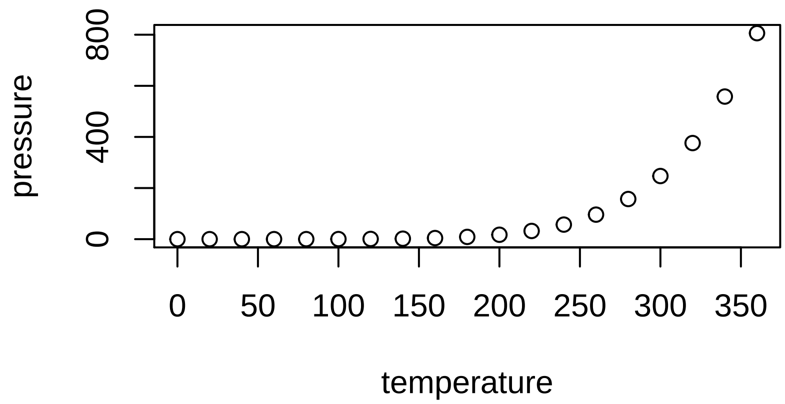
5. The approaches to calculate ROIs The Booz Allen Hamilton (BAH) approach and the one took in class are equally good. The BAH method is more applicable when considering each unit sale, while the one took in class is more generally applicable.

```
summary(cars)
```

```
     speed           dist       
 Min.   : 4.0   Min.   :  2.00  
 1st Qu.:12.0   1st Qu.: 26.00  
 Median :15.0   Median : 36.00  
 Mean   :15.4   Mean   : 42.98  
 3rd Qu.:19.0   3rd Qu.: 56.00  
 Max.   :25.0   Max.   :120.00  
```

1.1 Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

2 Estimating lift factors and promotion ROI analysis

In this part of the assignment, we analyze the effectiveness and ROI of different promotions for Hellman's 32 oz Mayonnaise. The analysis is based on account level data at Jewel-Osco and Dominick's Finer Foods in Chicago. Use the table (data frame) hellmans_df in the file Hellmans.RData. hellmans_DF contains the following variables:

- account
- product
- week
- units
- dollars
- feature_pctacv
- display_pctacv

2.1 Question 1

Create a price variable for Hellman's 32oz mayo. Then, although not strictly necessary (because the estimated coefficients will scale in a linear regression), you should divide the feature and display columns (variables) by 100. Examine the feature and display variables. Provide summary statistics (number of observations, mean, standard deviation) and histograms of these variables, separately for both accounts. To what extent do these two promotional instruments differ? Calculate the correlations between feature_pctacv, display_pctacv, and price (use the cor function in R). Comment on your findings. Do the correlations indicate a potential problem for your regression analysis to be performed below?

```r
hellmans_df$price = hellmans_df$dollars / hellmans_df$units
hellmans_df$feature = hellmans_df$feature_pctacv / 100
hellmans_df$display = hellmans_df$display_pctacv / 100

my_summary <- function(df, account) {
  df_local = df[df$account == account,]
    list("Feature Summary" =
       list("count" = length(df_local$feature),
            "mean" = mean(df_local$feature),
            "sd" = sd(df_local$feature)),
       "Display Summary" =
       list("count" = length(df_local$display),
            "mean" = mean(df_local$display),
            "sd" = sd(df_local$display)))
}

D_summary = my_summary(hellmans_df, "Dominicks")
J_summary = my_summary(hellmans_df, "Jewel")
print(D_summary)
```

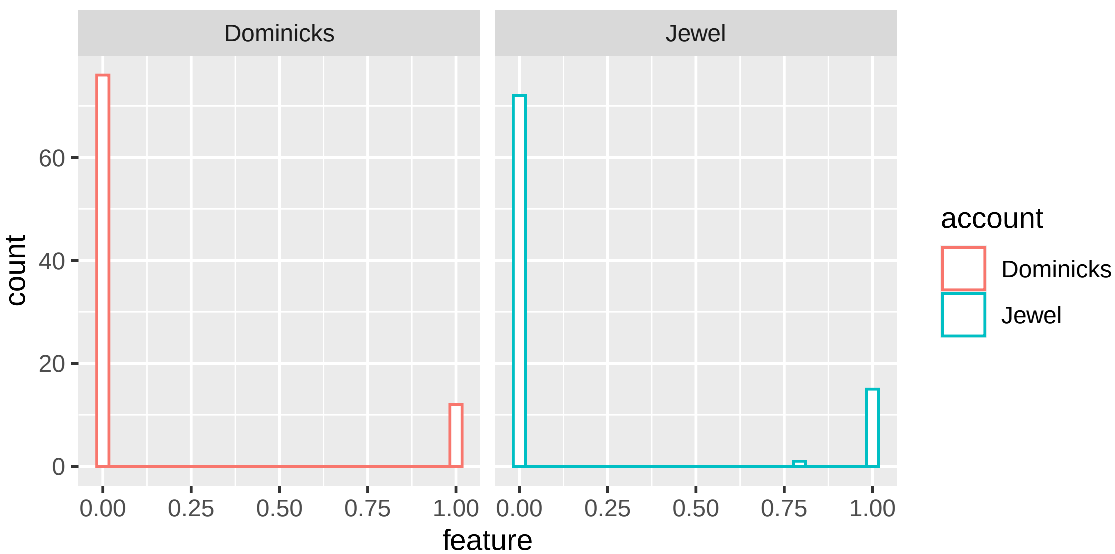
```
$`Feature Summary`
$`Feature Summary`$count
[1] 88

$`Feature Summary`$mean
[1] 0.1363636

$`Feature Summary`$sd
[1] 0.3451409
```

```
$`Display Summary`
$`Display Summary`$count
[1] 88

$`Display Summary`$mean
[1] 0.1206818

$`Display Summary`$sd
[1] 0.1762952
```

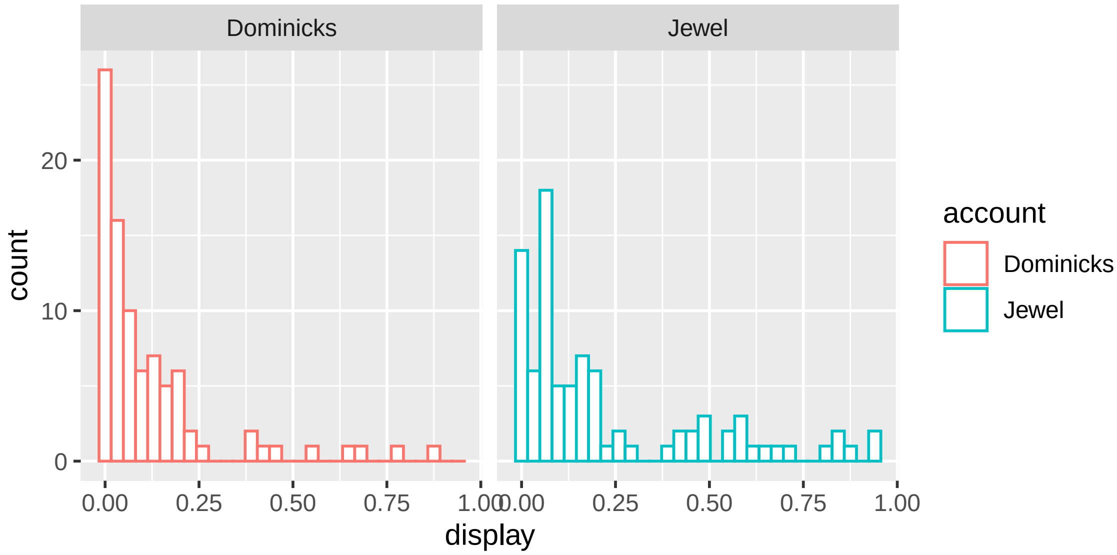
```r
print(J_summary)
```

```
$`Feature Summary`
$`Feature Summary`$count
[1] 88

$`Feature Summary`$mean
[1] 0.1794318

$`Feature Summary`$sd
[1] 0.3834338


$`Display Summary`
$`Display Summary`$count
[1] 88

$`Display Summary`$mean
[1] 0.2298864

$`Display Summary`$sd
[1] 0.2581655
```

```r
hellmans_df %>%
ggplot(data = ., aes(x=feature, color=account)) + geom_histogram(fill="white") + facet_grid(cols = vars
```



```r
hellmans_df %>%
ggplot(data = ., aes(x=display, color=account)) + geom_histogram(fill="white") + facet_grid(cols = vars
```




```r
#Correlations
cor(hellmans_df$feature_pctacv, hellmans_df$display_pctacv)
```

```
[1] 0.7599992
```

```r
cor(hellmans_df$feature_pctacv, hellmans_df$price)
```

```
[1] -0.5747241
```

```r
cor(hellmans_df$display_pctacv, hellmans_df$price)
```

```
[1] -0.6700056
```

2.2 Question 2

Estimate the log-linear demand model separately for each account, using price as the only explanatory variable. Then add the feature and display variables. Comment on the difference between the two regressions in terms of goodness of fit, and the price elasticity estimates. Is the change in price elasticity estimates as expected? What is the reason for this change? Are the coefficient estimates similar for both accounts?

```r
D_lm =
hellmans_df %>%
filter(account == "Dominicks") %>%
glm(log(units) ~ log(price), data = .)

summary(D_lm)
```

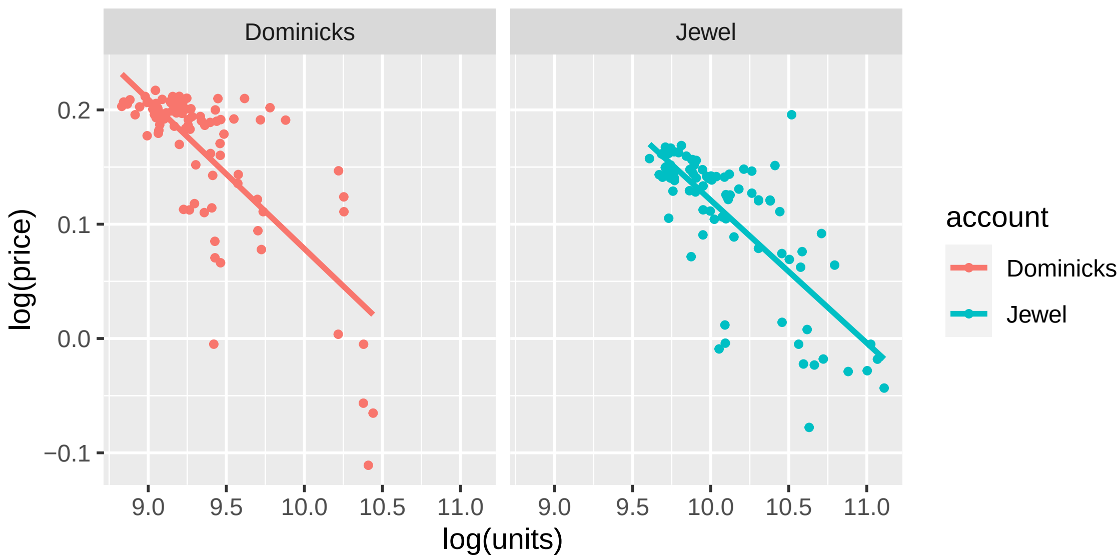
```
Call:
glm(formula = log(units) ~ log(price), data = .)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-0.63760  -0.17214  -0.01628   0.10558   0.79349

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  10.0368     0.0719  139.60  < 2e-16 ***
log(price)   -4.1665     0.4107  -10.15  2.3e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 0.06617719)

    Null deviance: 12.5037  on 87  degrees of freedom
Residual deviance:  5.6912  on 86  degrees of freedom
AIC: 14.753

Number of Fisher Scoring iterations: 2
```

```r
J_lm =
hellmans_df %>%
filter(account == "Jewel") %>%
glm(log(units) ~ log(price), data = .)

summary(J_lm)
```

```
Call:
glm(formula = log(units) ~ log(price), data = .)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-0.59230  -0.16883  -0.03486   0.15152   0.81131

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 10.60443    0.05259  201.66   <2e-16 ***
log(price)  -4.58359    0.42660  -10.74   <2e-16 ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 0.06161774)

    Null deviance: 12.4124  on 87  degrees of freedom
Residual deviance:  5.2991  on 86  degrees of freedom
AIC: 8.4712

Number of Fisher Scoring iterations: 2
```

```
#Two linear model demand-price graph
hellmans_df %>%
ggplot(data = ., aes(x= log(units), y = log(price), color=account)) + geom_point(size = 1, alpha = 1) +
  facet_grid(cols = vars(account)) + geom_smooth(method = "lm", se = FALSE)
```




```
#Compare with the feature add model

D_d_lm =
hellmans_df %>%
filter(account == "Dominicks") %>%
glm(log(units) ~ log(price) + display, data = .)
summary(D_d_lm)
```

```
Call:
glm(formula = log(units) ~ log(price) + display, data = .)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-0.43297  -0.14369  -0.02460   0.09584   0.59909

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  9.61572    0.08888 108.190  < 2e-16 ***
log(price)  -2.36500    0.44187  -5.352 7.25e-07 ***
display      1.07331    0.16833   6.376 9.04e-09 ***
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 0.04529276)

    Null deviance: 12.5037  on 87  degrees of freedom
Residual deviance:  3.8499  on 85  degrees of freedom
AIC: -17.645

Number of Fisher Scoring iterations: 2
```

```
J_d_lm =
hellmans_df %>%
filter(account == "Jewel") %>%
glm(log(units) ~ log(price) + display, data = .)
summary(J_d_lm)
```

```
Call:
glm(formula = log(units) ~ log(price) + display, data = .)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-0.36125  -0.10576  -0.03313   0.09383   0.51352

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 10.09791    0.06263 161.233  < 2e-16 ***
log(price)  -1.89014    0.39966  -4.729 8.86e-06 ***
display      0.95534    0.09657   9.892 8.47e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 0.02897969)

    Null deviance: 12.4124  on 87  degrees of freedom
Residual deviance:  2.4633  on 85  degrees of freedom
AIC: -56.941

Number of Fisher Scoring iterations: 2
```

```
D_d_f_lm =
hellmans_df %>%
filter(account == "Dominicks") %>%
glm(log(units) ~ log(price) + display + feature, data = .)
summary(D_d_f_lm)
```

```
Call:
glm(formula = log(units) ~ log(price) + display + feature, data = .)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-0.34144  -0.13771  -0.02137   0.11067   0.61078

Coefficients:
```

```
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  9.52123    0.08944 106.451  < 2e-16 ***
log(price)  -1.84318    0.45032  -4.093 9.74e-05 ***
display      0.83410    0.17653   4.725 9.14e-06 ***
feature      0.28531    0.08925   3.197  0.00196 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 0.04086078)

    Null deviance: 12.5037  on 87  degrees of freedom
Residual deviance:  3.4323  on 84  degrees of freedom
AIC: -25.748

Number of Fisher Scoring iterations: 2
```

```r
J_d_f_lm =
hellmans_df %>%
filter(account == "Jewel") %>%
glm(log(units) ~ log(price) + display + feature, data = .)
summary(J_d_f_lm)
```

```
Call:
glm(formula = log(units) ~ log(price) + display + feature, data = .)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-0.36769  -0.12020  -0.02219   0.08526   0.49093

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 10.08881    0.06327 159.450  < 2e-16 ***
log(price)  -1.89735    0.39969  -4.747 8.39e-06 ***
display      1.06947    0.14891   7.182 2.56e-10 ***
feature     -0.09124    0.09062  -1.007    0.317
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 0.02897501)

    Null deviance: 12.4124  on 87  degrees of freedom
Residual deviance:  2.4339  on 84  degrees of freedom
AIC: -55.997

Number of Fisher Scoring iterations: 2
```

```r
# add display improve the model, reduce deviance. However, add feature just improve it a little bit, si
```

2.3 Question 3

Consider the following three promotions:
(a) 15% TPR
(b) 15% TPR, 70% display
(c) 15% TPR, 70% display, 100% feature
Calculate the lift factors for each promotion for both accounts, based on the regression estimates in 2. Set estimates that are not statistically significant = 0.

```r
lift_factor <- function(model, TPR = 0, DIS = 0, FEA = 0) {
  alpha  = if(summary(model)$coef[1,4] < 0.1) {summary(model)$coef[1,1]} else {0}

  beta_lp  = if(summary(model)$coef[2,4] < 0.1) {summary(model)$coef[2,1]} else {0}
#  print(beta_lp)

  beta_d  =
if(length(summary(model)$coef[,1]) >= 3 ){
  if(summary(model)$coef[3,4] < 0.1)
  {summary(model)$coef[3,1]} else {
      0}
} else {
  0
}
#  print(beta_d)

  beta_f  =
if(length(summary(model)$coef[,1]) >= 4 ){
  if(summary(model)$coef[4,4] < 0.1)
  {summary(model)$coef[4,1]} else {
      0}
} else {
  0
}

  lf = exp(beta_lp*log(1 - TPR) + beta_d*DIS + beta_f*FEA)
}


#(a)
#For Dominicks
print(lift_factor(D_d_f_lm, 0.15))
```

```
[1] 1.349254
```

```r
#For Jewel
print(lift_factor(J_d_f_lm, 0.15))
```

```
[1] 1.361185
```

```r
#(b)
#For Dominicks
print(lift_factor(D_d_f_lm, 0.15, 0.7))
```

```
[1] 2.419158
```

```r
#For Jewel
print(lift_factor(J_d_f_lm, 0.15, 0.7))
```

```
[1] 2.877672
```

```r
#(c)
#For Dominicks
print(lift_factor(D_d_f_lm, 0.15, 0.7, 1))
```

```
[1] 3.217896
```

```r
#For Jewel
print(lift_factor(J_d_f_lm, 0.15, 0.7, 1))
```

```
[1] 2.877672
```

2.4 Question 4

Perform an ROI analysis of the three promotions, (a), (b), and (c), separately for the two retail accounts, Dominick's and Jewel-Osco. The promotions last for one week. Your analysis should follow the approach that we took in class, not the version of this approach taken by Booz Allen Hamilton in the first part of the assignment.

Note. Perform the analysis using units, not cases of Hellman's mayo. You will need the following data for your analysis:

- The regular price of the product at both accounts is $1.20.
- The VCM for Hellman's is $0.55 per unit.
- The manufacturer fully pays for the shelf price reduction. E.g., if the shelf price is reduced from $1.20 to $1.00, the manufacturer pays for this TPR through a $0.20 per unit (off-invoice) allowance.
- The fixed cost (MDF) for the promotion involving display only is $3,000 at Dominick's and $5,000 at Jewel-Osco. The fixed cost for the promotion including feature and display is $4,500 at Dominick's and $6,800 at Jewel-Osco.

In order to estimate baseline sales, use the regression estimates and the regular price, and predict sales for display and feature = 0.

Using these data, and the lift factors found in 3, you can then fill in the cells in the blueprint of a spreadsheet below, for each of the three promotions at both accounts.

Consider both:

- No stockpiling (purchase acceleration)
- The case where 20 percent of the incremental units as predicted by the event lift are due to stockpiling (purchase acceleration), and hence not truly incremental

```
ROI_Summary <- function(model, TPR = 0, DIS = 0, FEA = 0, fixed_payment_cost = 0, regular_price = 1.2, 

baseline_units = exp(predict(D_d_f_lm, data.frame(price = regular_price, display = 0, feature = 0),
   type = "response"))

total_units = lift_factor(D_d_f_lm, TPR, DIS, FEA) * baseline_units

incremental_units = (total_units - baseline_units)

incremental_units_Stockpiling = incremental_units * Stockpiling

incremental_units_net = incremental_units - incremental_units_Stockpiling

promoted_price = (1 - TPR)*regular_price

promoted_margine = promoted_price - (regular_price - regular_margin)

incremental_contribution = promoted_margine * incremental_units_net

variable_cost = TPR * regular_price * baseline_units

event_cost = variable_cost + fixed_payment_cost

gross_contribution = incremental_contribution - event_cost

ROI = gross_contribution/event_cost

list("Baseline units" = baseline_units,
     "Incremental units" = incremental_units,
     "Total units" = total_units,
```

```r
      "Precent with pa" = Stockpiling,
      "Incremental units with pa" = incremental_units_Stockpiling,
      "Incremental units net" = incremental_units_net,
      "Incremental contribution" = incremental_contribution,
      "Variable cost" = variable_cost,
      "Fixed payment cost" = fixed_payment_cost,
      "Event cost" = event_cost,
      "Event gross contribution" = gross_contribution,
      "ROI" = ROI)
}




#For Dominicks
#(a)
df1 = data.frame(ROI_Summary(D_d_f_lm, 0.15))
#(b)
df2 = data.frame(ROI_Summary(D_d_f_lm, 0.15, 0.7, fixed_payment_cost = 3000))
#(c)
df3 = data.frame(ROI_Summary(D_d_f_lm, 0.15, 0.7, 1, fixed_payment_cost = 4500))

D_df = cbind(data.frame(t(df1)), data.frame(t(df2)), data.frame(t(df3)))
colnames(D_df) = c("Dominicks(a)", "Dominicks(b)", "Dominicks(c)")
print(D_df)
```

```
                           Dominicks(a) Dominicks(b) Dominicks(c)
Baseline.units               9751.5563384 9.751556e+03 9.751556e+03
Incremental.units            3405.7695478 1.383900e+04 2.162793e+04
Total.units                 13157.3258862 2.359055e+04 3.137949e+04
Precent.with.pa                 0.0000000 0.000000e+00 0.000000e+00
Incremental.units.with.pa       0.0000000 0.000000e+00 0.000000e+00
Incremental.units.net        3405.7695478 1.383900e+04 2.162793e+04
Incremental.contribution     1260.1347327 5.120429e+03 8.002336e+03
Variable.cost                1755.2801409 1.755280e+03 1.755280e+03
Fixed.payment.cost              0.0000000 3.000000e+03 4.500000e+03
Event.cost                   1755.2801409 4.755280e+03 6.255280e+03
Event.gross.contribution     -495.1454082 3.651484e+02 1.747056e+03
ROI                            -0.2820891 7.678799e-02 2.792929e-01
```

```r
#Consider stockpiling is 20%
#(a)
df1 = data.frame(ROI_Summary(D_d_f_lm, 0.15, Stockpiling = 0.2))
#(b)
df2 = data.frame(ROI_Summary(D_d_f_lm, 0.15, 0.7, fixed_payment_cost = 3000, Stockpiling = 0.2))
#(c)
df3 = data.frame(ROI_Summary(D_d_f_lm, 0.15, 0.7, 1, fixed_payment_cost = 4500, Stockpiling = 0.2))

D_20_df = cbind(data.frame(t(df1)), data.frame(t(df2)), data.frame(t(df3)))
colnames(D_20_df) = c("Dominicks(a)", "Dominicks(b)", "Dominicks(c)")
print(D_20_df)
```

```
                           Dominicks(a)  Dominicks(b) Dominicks(c)
Baseline.units               9751.5563384  9751.5563384 9.751556e+03
```

```
Incremental.units          3405.7695478 13838.9961068 2.162793e+04
Total.units               13157.3258862 23590.5524453 3.137949e+04
Precent.with.pa              0.2000000     0.2000000 2.000000e-01
Incremental.units.with.pa   681.1539096  2767.7992214 4.325587e+03
Incremental.units.net      2724.6156382 11071.1968855 1.730235e+04
Incremental.contribution   1008.1077861  4096.3428476 6.401869e+03
Variable.cost              1755.2801409  1755.2801409 1.755280e+03
Fixed.payment.cost            0.0000000  3000.0000000 4.500000e+03
Event.cost                 1755.2801409  4755.2801409 6.255280e+03
Event.gross.contribution   -747.1723548  -658.9372933 1.465884e+02
ROI                          -0.4256713    -0.1385696 2.343434e-02
```

```r
#For Jewel
#(a)
df1 = data.frame(ROI_Summary(J_d_f_lm, 0.15))
#(b)
df2 = data.frame(ROI_Summary(J_d_f_lm, 0.15, 0.7, fixed_payment_cost = 5000))
#(c)
df3 = data.frame(ROI_Summary(J_d_f_lm, 0.15, 0.7, 1, fixed_payment_cost = 6800))

J_20_df = cbind(data.frame(t(df1)), data.frame(t(df2)), data.frame(t(df3)))
colnames(J_20_df) = c("Jewel(a)", "Jewel(b)", "Jewel(c)")
print(J_20_df)
```

```
                              Jewel(a)      Jewel(b)      Jewel(c)
Baseline.units             9751.5563384  9751.5563384  9.751556e+03
Incremental.units          3405.7695478 13838.9961068  2.162793e+04
Total.units               13157.3258862 23590.5524453  3.137949e+04
Precent.with.pa               0.0000000     0.0000000  0.000000e+00
Incremental.units.with.pa     0.0000000     0.0000000  0.000000e+00
Incremental.units.net      3405.7695478 13838.9961068  2.162793e+04
Incremental.contribution   1260.1347327  5120.4285595  8.002336e+03
Variable.cost              1755.2801409  1755.2801409  1.755280e+03
Fixed.payment.cost            0.0000000  5000.0000000  6.800000e+03
Event.cost                 1755.2801409  6755.2801409  8.555280e+03
Event.gross.contribution   -495.1454082 -1634.8515814 -5.529445e+02
ROI                          -0.2820891    -0.2420109 -6.463196e-02
```

```r
#Consider stockpiling is 20%
#(a)
df1 = data.frame(ROI_Summary(J_d_f_lm, 0.15, Stockpiling = 0.2))
#(b)
df2 = data.frame(ROI_Summary(J_d_f_lm, 0.15, 0.7, Stockpiling = 0.2, fixed_payment_cost = 5000))
#(c)
df3 = data.frame(ROI_Summary(J_d_f_lm, 0.15, 0.7, 1, Stockpiling = 0.2, fixed_payment_cost = 6800))

J_20_df = cbind(data.frame(t(df1)), data.frame(t(df2)), data.frame(t(df3)))
colnames(J_20_df) = c("Jewel(a)", "Jewel(b)", "Jewel(c)")
print(J_20_df)
```

```
                              Jewel(a)      Jewel(b)      Jewel(c)
Baseline.units             9751.5563384  9751.5563384  9751.5563384
Incremental.units          3405.7695478 13838.9961068 21627.9341851
Total.units               13157.3258862 23590.5524453 31379.4905235
Precent.with.pa               0.2000000     0.2000000     0.2000000
```

```
Incremental.units.with.pa    681.1539096  2767.7992214  4325.5868370
Incremental.units.net       2724.6156382 11071.1968855 17302.3473481
Incremental.contribution    1008.1077861  4096.3428476  6401.8685188
Variable.cost               1755.2801409  1755.2801409  1755.2801409
Fixed.payment.cost             0.0000000  5000.0000000  6800.0000000
Event.cost                  1755.2801409  6755.2801409  8555.2801409
Event.gross.contribution    -747.1723548 -2658.9372933 -2153.4116221
ROI                           -0.4256713    -0.3936087    -0.2517056
```