Introduction

Google's Lift Measurement is a tool which offers a convenient, easy to set-up and almost real-time approach to measure the effect that our campaigns have on some of the most strategic marketing metrics, specifically, ones related to brand management.

Having said this, there's some key missing pieces of information that we as marketeers would normally want to have, as it is provided in most studies.

Google Brand Lift

About Lift Measurement

Lift Measurement is a tool within Google Ads Platform that's capable of measuring how your ads impact people's perception of your brand. These measures are extremely useful for us marketeers, since allows us to evaluate how a specific marketing campaign performs in terms of well-known key marketing metrics such as Brand Awareness, Recall and Consideration rather than the usual Digital Marketing metrics like clicks, impressions, frequency and reach.

Once enabled, the tool allow us to measure up to 3 different metrics which includes the already mentioned plus Favoravility and Purchase Intent. Additionally, like in traditional consumer tests, the tool requires that we provide some alternative brands from the category which typically includes direct competitors.

In terms of costs, even though it's advertised as free, in practice Google ask for a minimum investment in terms of ad spend of the campaign being tested which in case of YouTube Ads varies based on location but for videos anywhere else within their networks is USD $15,000.

How it works

Once the campaign is set up, metrics are selected and brands from category (typically competitors) are defined, you'll be given a preliminary minimum budget which one can assume is as a function of the statistical power.

Eventually when campaigns are running and ads are being served, the tool will randomly generate a test and control group. Specifically, groups will be generated based on the following criteria:

- **Test group:** People who have seen our ads
- **Control group:** People who were eligible by our campaign segmentation, but didn't saw our ads.

Surveys from both groups will be gathered and different metrics such as Brand Lift will be computed and shown in the dashboard as soon a "detectable" result is found, which Google defines this to be as a Lift >2.0%.

Results

Once enough survey responses are collected the tool will present its results in terms of expected values and their corresponding (confidence) intervals in the form of the following metrics:

- **Baseline PRR:** Percent of people in the control group who selected our brand as their preference amongst all other provided brands.
- **Exposed PRR:** Percent of people in the test group who selected our brand as their preference amongst all other provided brands.
- **Absolute Brand Lift:** Difference in PRR between Test and Control groups.
- **Relative Brand Lift:** Increase of Baseline PRR due to the tested campaign (Absolute Brand Lift / Baseline PRR).
- **Total Survey Responses:** Number of total surveys (Test + Control).

Other metrics engineered from the above, such as *Cost Per Lifted User*, *Exposed to Ads*, *Not Exposed to Ads* are also included in the report.

** PRR: Positive Response Rate*

Missing Information

To understand if a small observed lift might be due to just random chance, generate new intervals according to our preferences and generally speaking better communicating the results within our company we would need information that's currently not being provided in a straight forward way.

Additionally, this information might shed some light into a deeper understanding as for why we need to invest the minimum budget that's being asked which in many cases is done for the sole purpose of this study.

In particular:

- **Level of significance:** Even though numeric confidence intervals are included for every metric, the actual confidence level in terms of percentages is not given nor it is commonly addressed within the documentation. So far, I've been able to locate some official information that states that's "usually around 90%" </i>[ref] but whether this is one or two-tailed is unclear.
- **P-Value:** This would allow us to make a fast hypothesis check for different levels of confidence we might use.

Lastly, even though "*Total Survey Responses*" is clearly informed both in summary and within the report itself, actual sample size of test and control groups is not displayed by default in the summary page nor it is available anywhere else within the report itself, limiting our ability to conduct our own statistical tests for Age, Gender, Campaign and Video related results.

Finding the missing information

In order to find the missing information we need to perform a heuristic search of all possible combinations of Confidence Levels and Sample Sizes and then compare those results to check which has the better approximation to the results provided by Google.

In order to avoid a brute-forcing method which is time and resource consuming a custom algorithm was built that will try to learn 2 parameters: *Sample Sizes* and *PRR Interval* as defined by the difference between expected Upper and Lower Absolute Brand Lift values.

Custom Optimization Algorithm

A custom optimization algorithm was built which will allow us to obtain a good approximation of:

- Level of significance.
- Sample size of both test and control groups.
- Standard Error.
- p-Value

This information will allow us to conduct our own statistical tests, which is a stepping stone in generating confidence intervals that suit our needs.

In [47]:

```
LiftUncover <-function(Surveys,p1,p2,UpperInterval,LowerInterval,staticLR=.1,maxIterations=10000,pValue=NULL) {

  # Target parameter initialization
  n1 <- Surveys*0.5
  if(is.null(pValue)) {
    zScore <- 1
  } else {
      # False positive rate is provided. We dont need to predict Standard Score.
      zScore <- round(abs(qnorm(pValue/2)),2)
  }

  # Transform to percentages.
  RealInterval <- ((UpperInterval)-(LowerInterval))/200
  p1 <- p1/100
  p2 <- p2/100

  # For ADAGRAD
  gradientsN1 <- c()
  gradientszScore <- c()

  # Temporal variables.
  temp <- Inf

  # Iteration Loop!
  for(i in 1:maxIterations) {

    # n2 calculation
    n2 <- Surveys-n1

    # We compute a predicted interval for given values and parameters
    PredictedInterval <- (zScore*sqrt(((p1 * n1 + p2 * (Surveys - n1)) / (Surveys)) * ( 1 - ((p1 * n1 + p2 * (Surveys - n1)) / (Surveys)) ) * ((1/n1) + (1/(Surveys - n1))) ) )

    # We generate a differentiable cost function
    Cost <- (RealInterval-PredictedInterval)^2
    # Print current Cost every 1000 iterations
    if(i %% 1000==0) {
      print(paste("Iteration",i,":",sqrt(Cost)))
    }

    # N1 Gradient
    dn1dCost <- -(100 * (zScore * ((1 - (n1 * p1 + p2 * (Surveys - n1))/Surveys) * (1/(Surveys - n1)
^2 - 1/n1^2) * (n1 * p1 + p2 * (Surveys - n1)) + (1 - 2 * ((n1 * p1 + p2 * (Surveys - n1))/Surveys))
* (1/(Surveys - n1) + 1/n1) * (p1 - p2)) * (RealInterval - zScore * sqrt((1 - (n1 * p1 + p2 * (Surve
ys - n1))/Surveys) * (1/(Surveys - n1) + 1/n1) * (n1 * p1 + p2 * (Surveys - n1))/Surveys))/ (Surveys
* sqrt((1 - (n1 * p1 + p2 * (Surveys - n1))/Surveys) * (1/(Surveys - n1) + 1/n1) * (n1 * p1 + p2 * (
Surveys - n1))/Surveys))))

    # Sample size (n1) gradient adjustment
    gradientsN1 <- rbind(gradientsN1,dn1dCost)
    sumgradientsN1 <- sum(gradientsN1^2)
    learningRateN1 <- staticLR / sqrt(sumgradientsN1 + 10^-8)

    # Sample size Parameter update
    n1 <- n1 - (learningRateN1  * dn1dCost )

    if(is.null(pValue))  {
      # zScore Gradient
      dzScoredCost <- -(200 * ((RealInterval - zScore * sqrt((1 - (n1 * p1 + p2 * (Surveys -
n1))/Surveys) * (1/(Surveys - n1) + 1/n1) * (n1 * p1 + p2 * (Surveys - n1))/Surveys)) * sqrt((1 - (n
1 * p1 + p2 * (Surveys - n1))/Surveys) * (1/(Surveys - n1) + 1/n1) * (n1 * p1 + p2 * (Surveys - n1))
/Surveys)))

      # Standard Score gradient adjustment
      gradientszScore <- rbind(gradientszScore,dzScoredCost)
      sumGradientszScore  <- sum(gradientszScore^2)
      learningRatezScore <- staticLR / sqrt(sumGradientszScore + 10^-8)

      # zScore Parameter update
      zScore <- zScore - (learningRatezScore   * dzScoredCost)

    }

    # Results
    if(sqrt(Cost) >= temp || i == maxIterations) {
      cat(paste("Values found after",i,"iterations\n"))
      output <- list()
      if(is.null(pValue)) {

        # We calculate false positive rate
        errorRate <- round(2*pnorm(-abs(zScore)),2)
        cat(paste("   - Predicted Type I Error:",errorRate,"\n"))

        # Adjustment to most likely (minimum distance) error selected by Google
        # Difference might be due to rounded intervals provided
        DefaultErrors <- c(.01,.05,.1,.2,.25,.3,.45)
        Distance <- abs(DefaultErrors-errorRate)
        likelyError <- DefaultErrors[which(Distance==min(Distance))]
        cat(paste("   - Most likely Type I Error:",likelyError,"\n"))
        cat(paste("   - Most likely Level of Significance: ",(1-likelyError)*100,"% (two tailed)\n"
,sep=""))

        # Output values
        output$predicted <- errorRate
        output$likely <- likelyError

      } else {

        # Standard Error
        standardError = sqrt((1 - (n1 * p1 + n2 * p2)/Surveys) * (1/n1 + 1/n2) * (n1 * p1 + n2 * p2)
/Surveys)
        pValue <- round(2*pnorm(-abs((p1 - p2) / standardError)),6)

        cat(paste("   - Sample size 1 (n1):",round(n1),"surveys\n"))
        cat(paste("   - Sample size 2 (n2):",Surveys-round(n1),"surveys\n"))
        cat(paste("   - Standard Error:",standardError,"\n"))
        cat(paste("   - p-value:",pValue,"\n"))


        # Output values
        output$n1 <- round(n1)
        output$n2 <- Surveys-round(n1)
        output$standardError <- standardError
        output$pValue <- pValue
      }

      break();
    }
    temp <- round(sqrt(Cost),13)

  }
return(output)
}
```

Algorithm in use

(Inputs provided by Google Brand Lift Results)

In [59]:

```
ExposedPRR <- 40.22
BaselinePRR <- 37.67
TotalSurveys <- 5623
LiftUpperInterval <- 4.3
LiftLowerInterval <- 0.9
```

In [60]:

```
# Some metrics
Lift <- (ExposedPRR-BaselinePRR)/100
cat("Expected Lift: ",Lift*100,"%",sep="")
```

```
Expected Lift: 2.55%
```

Confidence Level Calculation

We will try to obtain the confidence level used by Google in the report generated by the tool.

Note: We have two outputs (*Predicted* and *Most Likely*). This is because Google provides us approximate (rounded) values, which affects this calculation. We will later on use only “

Most Likely
” value, which is an approximation to commonly used intervals.

In [50]:

```
ErrorPrediction <-
LiftUncover(TotalSurveys,ExposedPRR,BaselinePRR,LiftUpperInterval,LiftLowerInterval)
```

```
Values found after 98 iterations
   - Predicted Type I Error: 0.19
   - Most likely Type I Error: 0.2
   - Most likely Level of Significance: 80% (two tailed)
```

Sample Sizes, Standard Error and p-Values

Values are calculated using the “Most likely Type I Error” calculated above. Note that these values will be the best approximation possible to given parameters.

In [56]:

```
likelyError <- ErrorPrediction$likely
samplePrediction <-
LiftUncover(TotalSurveys,ExposedPRR,BaselinePRR,LiftUpperInterval,LiftLowerInterval,staticLR=100000
,pValue=likelyError)
```

```
Values found after 86 iterations
Values found after 60 iterations
   - Sample size 1 (n1): 3366 surveys
   - Sample size 2 (n2): 2257 surveys
   - Standard Error: 0.0132812499999766
   - p-value: 0.054858
```

As seen in the results above, we obtained a **p-Value of 0.054858** so as we will later confirm, we can anticipate results will **not be significant for intervals of 95% or more**, though it's close at 95%.

Found the missing pieces

We can now proceed to define values for our custom intervals.

Note: these values are just (good) approximations of real ones. If there's a chance you already have real sample sizes or additional information, try using those and compare the results.

In [52]:

```r
n1 <- samplePrediction$n1
n2 <- samplePrediction$n2
standardError <- samplePrediction$standardError
pValue <- samplePrediction$pValue
```

Original Brand Lift Values

Now we can calculate the original report with all missing information included. This should be a good approximation to the actual values provided by Google

In [53]:

```r
# Original detected Brand Lift Error
Error <- likelyError

# Brand Lift confidence Interval
ConfidenceInterval <- (1-Error)*100
zScore <- round(abs(qnorm(Error/2)),2)
cat(paste(ConfidenceInterval,"% confidence interval:\n",sep=""))
Upper <- round(Lift + (zScore*standardError),5)*100
Lower <- round(Lift - (zScore*standardError),5)*100
cat(paste(Lower,"%  -  ",Upper,"%\n\n",sep=""))
```

```
80% confidence interval:
0.85%  -  4.25%
```

Confidence interval at 90%

In [54]:

```r
# 10 percent probability of type 1 error
Error <- 0.1

# Brand Lift confidence Interval
ConfidenceInterval <- (1-Error)*100
zScore <- round(abs(qnorm(Error/2)),2)
cat(paste(ConfidenceInterval,"% confidence interval:\n",sep=""))
Upper <- round(Lift + (zScore*standardError),5)*100
Lower <- round(Lift - (zScore*standardError),5)*100
cat(paste(Lower,"%  -  ",Upper,"%\n\n",sep=""))
```

```
90% confidence interval:
0.372%  -  4.728%
```

Confidence interval at 95%

In [55]:

```
# 5 percent probability of type 1 error
Error <- 0.05

# Brand Lift confidence Interval
ConfidenceInterval <- (1-Error)*100
zScore <- round(abs(qnorm(Error/2)),2)
cat(paste(ConfidenceInterval,"% confidence interval:\n",sep=""))
Upper <- round(Lift + (zScore*standardError),5)*100
Lower <- round(Lift - (zScore*standardError),5)*100
cat(paste(Lower,"%  -  ",Upper,"%\n\n",sep=""))
```

```
95% confidence interval:
-0.053%  -  5.153%
```

As predicted above, at this point, expected interval includes the **chance of lift being zero** which is consistent with our p-value found before.

Confidence interval at 99%

In [10]:

```
# 1 percent probability of type 1 error
Error <- 0.01

# Brand Lift confidence Interval
ConfidenceInterval <- (1-Error)*100
zScore <- round(abs(qnorm(Error/2)),2)
cat(paste(ConfidenceInterval,"% confidence interval:\n",sep=""))
Upper <- round(Lift + (zScore*standardError),5)*100
Lower <- round(Lift - (zScore*standardError),5)*100
cat(paste(Lower,"%  -  ",Upper,"%\n\n",sep=""))
```

```
99% confidence interval:
-0.827%  -  6.027%
```

Final thoughts

As seen by the report at 95% confidence interval we cannot reject the null hypothesis, therefore, we do not observe a significant lift in Positive Response Rate if we use a level of confidence of 95%. Contrary to this, we initially concluded there was a significant increment by observing the default output from Lift Measurement Tool.

Opinions may vary as to which confidence interval is acceptable for a Brand Lift Study or any other Marketing Study, which most likely, will ultimately depend on our goal.

Marketers will most likely be concerned about the actual effect the campaign had in lift values rather than the sole determination if this study is significant or not. In other words, we would like to know if we should continue investing on a specific campaign since we will most likely create budget scenarios in order to achieve a specific goal of lifted users. For this, we need a (known) level of certainty that can enable us to create budget scenarios which will include *Costs per lifted users* and *New Lifted Users* amongst others.

In most studies, this situation is usually addressed by providing the p-value which allows the reader to define its tolerance on accepting or rejecting the hypothesis.

Feedback

What's your thought on this? What levels of significance do you use in such marketing studies? Is it something you consider relevant?

https://www.linkedin.com/in/crisandrews/