Yao Yao
MSDS 6370 Section 403
Class Project

Estimating the Initial Mean Number of Views for Videos to be on Youtube's Trending List

Introduction

From Kaggle, the metadata of top trending YouTube videos from United States, Canada, Great Britain, Germany, and France are collected daily from November 14, 2017 to March 20, 2018 (127 days) via the YouTube API [1]. The YouTube algorithm for videos to start trending is derived from social interactions of users who view, share, comment, and like certain videos. From the dataset, the chance for a YouTube video to start trending during this time frame could span from a few hours since its publish time or from as old as 2010. Once the video starts trending on YouTube, there can be multiple days in which the video stays on the trending list while gaining more views. We are aware that time of day that the data is collected matters and there may be other algorithmic factors of social media interactions on YouTube and on other platforms that YouTube considers for a video to start trending [2]. For a rough estimate of how many views a video must garner to be on the YouTube trending list, we decided to focus our estimation approach without the extended interactions. The number of shares, comments, and likes are directly correlated as a result from how many views a video has and the video publisher, such as for Apple commercials, could disable embedding, comments, and likes for certain videos, which can skew their respective resulting dataset even further. Only the number of views a video has is the true initial indicator to where all subsequent user interaction could result and is the primary focal number of how a YouTube video can be on the top trending list.

Data Cleaning

From the scope of our analysis, we are only concerned about estimating the initial number of views a video must acquire before being featured on YouTube's top trending list. Therefore, for all the duplicate days that a video is featured, only the initial number of views would be counted towards the analysis. Due to the limited time frame of the data collection, older videos that may have higher view counts may have gotten their initial boost in views from a time when they were trending outside of the 127-day data collection period. Therefore, we decided to only keep the YouTube data from the videos that are within 1.5 times the inter-quartile range of the whole dataset for a more accurate model of estimating the mean number of views a video takes to get on the trending list (Figure 1).

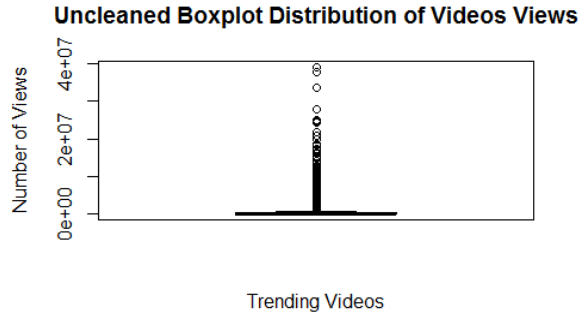

Figure 1a: Uncleaned Box Plot Distribution of Video Views (N = 49841)

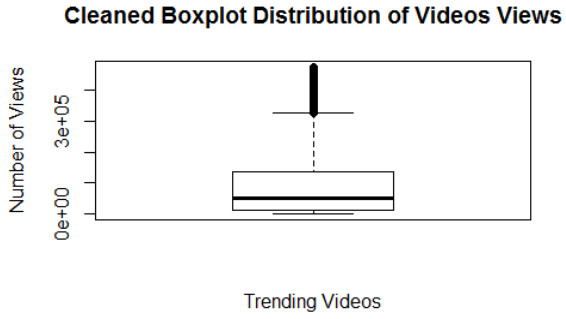

Figure 1b: Cleaned Box Plot Distribution of Video Views (N = 44506)

Figure 2 shows a numerical representation of how drastically the numbers change once cleaned, where the true population mean decreases from 224,403 views to 93,981 views, the standard deviation decreases from 731,176 views to 109607 views, and the standard error decreases from 3275.13 view to 519.55 views for that of uncleaned and cleaned datasets, respectively, where the statistics from the cleaned dataset is used for the further estimation procedures.

The MEANS Procedure

Analysis Variable : views							
Minimum	Lower Quartile	Mean	Median	Upper Quartile	Maximum	Std Error	Std Dev
223.0000000	14840.00	224403.96	61786.00	198996.00	39118664.00	3275.13	731176.26

Figure 2a: Uncleaned Quartile Distribution of Video Views (N = 49841)

The MEANS Procedure

Analysis Variable : views							
Minimum	Lower Quartile	Mean	Median	Upper Quartile	Maximum	Std Error	Std Dev
223.0000000	12030.00	93891.70	48041.50	137346.00	475127.00	519.5546202	109607.56

Figure 2b: Cleaned Quartile Distribution of Video Views (N = 44506)

Task 1: Simple Random Sample

Using the 95% confidence interval threshold, where margin of error is 5000 views for the cleaned dataset, we take a simple random sample of 1847 observations. The FPC adjustment is ignored because the cleaned dataset is 89.29% of the original population, which is less than 10%.

$$n_{0,srs} = \frac{(Z_{\alpha/2}S)^2}{(moe)^2} = \frac{(1.96 * 109607.56)^2}{5000^2} = 1846.09 \approx 1847 \text{ samples}$$

For simple random sample where sample size is 1847 and set seed is 1847, the mean estimate is 91,932.82 views with a standard error of 2,490.11 views for 95% confidence interval, where the true mean of 93,891.70 views is within the standard error range of the survey mean estimate.

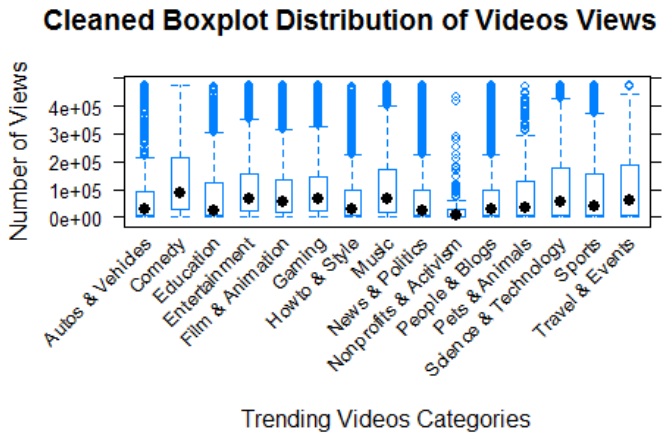

Figure 3a: Stratified YouTube Views Box Plot Distribution by Category

The MEANS Procedure

Analysis Variable : views									
category_id2	N Obs	Minimum	Lower Quartile	Mean	Median	Upper Quartile	Maximum	Std Error	Std Dev
Autos & Vehicles	898	873.0000000	5651.00	76604.95	26299.50	90376.00	473212.00	3821.88	114528.81
Comedy	2732	385.0000000	30755.50	134680.54	87519.00	212030.00	474626.00	2458.08	128480.41
Education	1165	640.0000000	5426.00	79568.58	22477.00	125144.00	468531.00	3098.66	105763.67
Entertainment	13136	223.0000000	23581.00	106526.37	65077.00	154283.50	475127.00	959.6016099	109982.23
Film & Animation	2307	887.0000000	16135.00	94314.14	55111.00	135336.00	474065.00	2175.29	104482.00
Gaming	1806	528.0000000	23217.00	99459.75	64837.00	145387.00	474042.00	2369.87	100712.42
Howto & Style	2824	748.0000000	8047.50	71889.18	27638.50	95472.00	471266.00	1811.41	96260.96
Music	2467	920.0000000	20544.00	114123.08	68182.00	171849.00	474925.00	2392.70	118843.03
News & Politics	4991	274.0000000	8411.00	69044.84	25610.00	95900.00	473026.00	1297.23	91645.27
Nonprofits & Activ	201	1420.00	4922.00	32267.94	9450.00	30007.00	429677.00	4337.28	61491.50
People & Blogs	6522	543.0000000	8346.00	72868.97	28451.50	95555.00	474846.00	1215.92	98196.09
Pets & Animals	399	743.0000000	7382.00	82815.97	32043.00	128316.00	467416.00	5259.74	105063.17
Science & Technolo	1043	594.0000000	8082.00	113837.62	55731.00	175988.00	473531.00	4135.82	133568.37
Sports	3740	374.0000000	7780.50	96654.91	40939.00	154509.50	474375.00	1944.12	118893.54
Travel & Events	275	789.0000000	8009.00	104031.37	59758.00	187612.00	472072.00	6561.75	108814.33

Figure 3b: Stratified YouTube Views Quartile Distribution by Category

Task 1: Proportional Allocation of Stratified Sample

We stratified the videos by category, to see if there is a difference in mean and distributions for how a YouTube video becomes trending on YouTube (Figure 3). 'Entertainment' and 'People & Blogs' have the most observations while 'Nonprofits & Activism' and 'Travel & Events' have the least observations for the dataset collected. 'Nonprofits & Activism' and 'News and Politics' require the least mean views for a video to be on trending while 'Music' and 'Comedy' require the most mean views for a video to be on trending. 'Entertainment' and 'People & Blogs' have the lowest standard error for their mean views to be on trending while 'Pets & Animals' and 'Travel & Events' have the highest standard error for their mean views to be on trending.

For proportional allocation of sample of size 1847 for the stratified YouTube trending views by category, the categories with the least sample size ('Nonprofits & Activism' and 'Travel & Events') were rounded up for the sampled total to be 1847 (Figure 4).

Stratum	Observ	N_h/N	$1847*N_h/N$	Sample Size	$1893*N_h/N$	Sample Size
Autos & Vehicles	898	0.020177	37.26702	37	38.19516	38
Comedy	2732	0.061385	113.3781	113	116.2018	116
Education	1165	0.026176	48.34753	48	49.55163	50
Entertainment	13136	0.295151	545.1443	545	558.7213	559
Film & Animation	2307	0.051836	95.74055	96	98.12499	98
Gaming	1806	0.040579	74.94904	75	76.81567	77
Howto & Style	2824	0.063452	117.1961	117	120.1149	120
Music	2467	0.055431	102.3806	102	104.9304	105
News & Politics	4991	0.112142	207.1266	207	212.2852	212
Nonprofits & Activism	201	0.004516	8.341505	9	8.549252	9

People & Blogs	6522	0.146542	270.6631	271	277.4041	277
Pets & Animals	399	0.008965	16.55851	17	16.9709	17
Science & Technology	1043	0.023435	43.28452	43	44.36254	44
Sports	3740	0.084034	155.2101	155	159.0756	159
Travel & Events	275	0.006179	11.41251	12	11.69674	12
Total	**44506**			**1847**		**1893**

Figure 4: Proportional Allocation of Stratified Views by Category for Sample Size 1847 and 1893 After Design Effect

For proportional allocation of stratified sample where sample size is 1847 and set seed is 1847, the mean estimate is 96,607.95 views with a standard error of 2,551.78 views for 95% confidence interval, where the true mean of 93,891.70 views is not within the standard error range of the survey mean estimate.

$$Deff_{complex} = \frac{V(\bar{y}_{complex})}{V(\bar{y}_{srs})} = \frac{2551.78}{2490.11} = 1.024767$$

$$n_{0,complex} = n_{0,srs} * Deff_{complex} = 1847 * 1.024767 = 1892.74 \approx 1893 \text{ samples}$$

To account for design effect, the standard errors of simple random sampling and proportional allocation for stratified sampling show that the estimate for simple random sample is more precise. The new sample size for proportional allocation is 1893 views (Figure 4).

For proportional allocation of stratified sample after design effect where sample size is 1893 and set seed is 1893, the mean estimate is 95,055.87 views with a standard error of 2,484.97 views for 95% confidence interval, where the true mean of 93,891.70 views is within the standard error range of the survey mean estimate.

Task 1: Comparisons of Sampling Procedures x1

Sample Procedure	Sample Size	Mean Estimate	Standard Error	Lower CI	Upper CI	True Mean Within CI?	Abs Diff From True Mean
Simple Random	1847	91932.82	2490.11	89442.71	94422.93	Yes	1958.88
Proportional Allocation	1847	96607.95	2551.78	94056.17	99159.73	No	2716.25
Proportional Allocation After Design Effect	1893	95055.87	2484.97	92570.9	97540.84	Yes	1164.17

Figure 5: Comparisons of Mean Estimate for Task 1 Sampling Procedures for True Mean of 93891.7

For sample size 1847, simple random sample performed better than proportional allocation where the true mean was within confidence interval for simple random and not for that of proportional allocation. Once corrected for design effect for a sample size of 1893, the proportional allocation mean estimate is closer to the true mean than that of simple random sample (Figure 5). The standard error is the smallest for that of proportional allocation after design effect.

Task 2: Comparisons of Sampling Procedures x5

Sample Procedure	Sample Size	Mean Estimate	Standard Error	Lower CI	Upper CI	True Mean Within CI?	Abs Diff From True Mean
Simple Random	1847	91494.1	2511.972	88982.13	94006.07	Yes	2397.6
Simple Random	1847	95733.22	2562.7	93170.52	98295.92	Yes	1841.52
Simple Random	1847	94591.21	2575.722	92015.49	97166.93	Yes	699.51
Simple Random	1847	93210.51	2568.106	90642.4	95778.62	Yes	681.19
Simple Random	1847	96893.45	2609.723	94283.73	99503.17	No	3001.75
Proportional Allocation	1847	98122.43	2600.976	95521.45	100723.4	No	4230.73
Proportional Allocation	1847	92762.59	2533.739	90228.85	95296.33	Yes	1129.11
Proportional Allocation	1847	95200.77	2577.481	92623.29	97778.25	Yes	1309.07
Proportional Allocation	1847	91645.58	2441.759	89203.82	94087.34	Yes	2246.12
Proportional Allocation	1847	96482.08	2465.507	94016.57	98947.59	No	2590.38
Proportional Allocation After Design Effect	1893	97117.92	2547.008	94570.91	99664.93	No	3226.22
Proportional Allocation After Design Effect	1893	92751.74	2451.187	90300.55	95202.93	Yes	1139.96
Proportional Allocation After Design Effect	1893	91844.2	2463.036	89381.16	94307.24	Yes	2047.5
Proportional Allocation After Design Effect	1893	93422.21	2506.443	90915.77	95928.65	Yes	469.49
Proportional Allocation After Design Effect	1893	95806.36	2549.604	93256.76	98355.96	Yes	1914.66

Figure 6a: Comparisons of Mean Estimate for Task 2 Sampling Procedures for True Mean of 93891.7

After conducting the Task 1 sampling procedures for 5 independent sample seeds, every different sampling procedure has at least one result where the true mean is not within the 95% confidence interval, where proportional allocation has two (Figure 6).

Average Sample Procedure	Sample Size	Mean Estimate	Standard Error	Lower CI	Upper CI	%True Mean Within CI	Abs Diff From True Mean
Simple Random	1847	94384.5	2565.645	91818.86	96950.15	80%	492.8
Proportional Allocation	1847	94842.69	2523.892	92318.8	97366.58	60%	950.99
Proportional Allocation After Design Effect	1893	94188.49	2503.456	91685.03	96691.95	80%	296.79

Figure 6b: Average Comparisons of Mean Estimate for Task 2 Sampling Procedures for True Mean of 93891.7

By averaging the samples by procedure for sample size of 1847, simple random sample performed better than proportional allocation where the estimated mean is closer than the true mean. Averaging the proportional allocation after design effect for sample size of 1893, the proportional allocation mean estimate is closer to the true mean than that of simple random sample. The average standard error is the smallest for that of proportional allocation after design effect. Proportional allocation after design effect is the best method to estimate the mean views of 94188 for a video to get on YouTube Trending.

References

[1] "Trending YouTube Video Statistics: Daily statistics for trending YouTube videos," Kaggle, 2018. [Online]. Available: https://www.kaggle.com/datasnaek/youtube-new [Accessed 23-Mar-2018]

[2] "Trending on YouTube," Google, 2018. [Online]. Available: https://support.google.com/youtube/answer/7239739 [Accessed 23-Mar-2018]

Appendix: R Code

```
cat("\014")
options(warn=1)
require(survey)
require(dplyr)
require(lattice)

#read in, remove columns, from https://www.kaggle.com/datasnaek/youtube-new/data
setwd("C:/Users/Yao/Desktop/you")
youtubeRawUS <- read.csv(file="USvideos.csv", header=TRUE, sep=",")
youtubeRawUS$country <- rep("US",nrow(youtubeRawUS))
youtubeRawUS <- youtubeRawUS[-c(3:4,7,12:16)]
youtubeRawCA <- read.csv(file="CAvideos.csv", header=TRUE, sep=",")
youtubeRawCA$country <- rep("CA",nrow(youtubeRawCA))
youtubeRawCA <- youtubeRawCA[-c(3:4,7,12:16)]
youtubeRawDE <- read.csv(file="DEvideos.csv", header=TRUE, sep=",")
youtubeRawDE$country <- rep("DE",nrow(youtubeRawDE))
youtubeRawDE <- youtubeRawDE[-c(3:4,7,12:16)]
youtubeRawFR <- read.csv(file="FRvideos.csv", header=TRUE, sep=",")
youtubeRawFR$country <- rep("FR",nrow(youtubeRawFR))
youtubeRawFR <- youtubeRawFR[-c(3:4,7,12:16)]
youtubeRawGB <- read.csv(file="GBvideos.csv", header=TRUE, sep=",")
youtubeRawGB$country <- rep("GB",nrow(youtubeRawGB))
youtubeRawGB <- youtubeRawGB[-c(3:4,7,12:16)]

youtubeRaw<- rbind(youtubeRawUS, youtubeRawCA)
youtubeRaw<- rbind(youtubeRaw, youtubeRawDE)
youtubeRaw<- rbind(youtubeRaw, youtubeRawFR)
youtubeRaw<- rbind(youtubeRaw, youtubeRawGB)

head(youtubeRaw)
youtubeRaw2 <- youtubeRaw
#remove duplicates because they can be trending in multiple months, keep the least views to get on
  trending
youtubeRaw2 = youtubeRaw2[order(youtubeRaw2[,'video_id'],youtubeRaw2[,'views']),]
youtubeRaw2 = youtubeRaw2[!duplicated(youtubeRaw2$video_id),]
head(youtubeRaw2)
write.csv(youtubeRaw2, file = "youtubeRaw2.csv")
```

```
#replace strata categories into real names
youtubeRaw2$category_id2[youtubeRaw2$category_id == '1'] <- 'Film & Animation'
youtubeRaw2$category_id2[youtubeRaw2$category_id == '2'] <- 'Autos & Vehicles'
youtubeRaw2$category_id2[youtubeRaw2$category_id == '10'] <- 'Music'
youtubeRaw2$category_id2[youtubeRaw2$category_id == '15'] <- 'Pets & Animals'
youtubeRaw2$category_id2[youtubeRaw2$category_id == '17'] <- 'Sports'
youtubeRaw2$category_id2[youtubeRaw2$category_id == '19'] <- 'Travel & Events'
youtubeRaw2$category_id2[youtubeRaw2$category_id == '20'] <- 'Gaming'
youtubeRaw2$category_id2[youtubeRaw2$category_id == '22'] <- 'People & Blogs'
youtubeRaw2$category_id2[youtubeRaw2$category_id == '23'] <- 'Comedy'
youtubeRaw2$category_id2[youtubeRaw2$category_id == '24'] <- 'Entertainment'
youtubeRaw2$category_id2[youtubeRaw2$category_id == '25'] <- 'News & Politics'
youtubeRaw2$category_id2[youtubeRaw2$category_id == '26'] <- 'Howto & Style'
youtubeRaw2$category_id2[youtubeRaw2$category_id == '27'] <- 'Education'
youtubeRaw2$category_id2[youtubeRaw2$category_id == '28'] <- 'Science & Technology'
youtubeRaw2$category_id2[youtubeRaw2$category_id == '29'] <- 'Nonprofits & Activism'
youtubeRaw2$category_id2[youtubeRaw2$category_id == '43'] <- 'Science & Technology'
youtubeRaw2$category_id[youtubeRaw2$category_id == '43'] <- '28'

#sanity check, remove column names
head(youtubeRaw2)
rownames(youtubeRaw2) <- c()
youtubeRaw3 <- youtubeRaw2[order(youtubeRaw2$category_id2),]
head(youtubeRaw3)
write.csv(youtubeRaw3, file = "youtubeRaw3.csv", row.names=FALSE)

#use sas for more youtubeRaw3.csv dataset stats

#check initial distribution and number of rows
boxplot(youtubeRaw3$views, main="Uncleaned Boxplot Distribution of Videos Views", xlab="Trending
   Videos", ylab="Number of Views")
nrow(youtubeRaw3)
bwplot(views ~ category_id2 , data = youtubeRaw3, scales=list(x=list(rot=45)), main="Uncleaned Boxplot
   Distribution of Videos Views", xlab="Trending Videos Categories", ylab="Number of Views")

#solve for view count without cleaning data
sum(as.numeric(youtubeRaw3$views))
max(youtubeRaw3$views)
min(youtubeRaw3$views)
mean(youtubeRaw3$views)

#remove outliers more than 1.5 quant, save into new clean dataset
remove_outliers <- function(x, na.rm = TRUE, ...) {
 qnt <- quantile(x, probs=c(.25, .75), na.rm = na.rm, ...)
 H <- 1.5 * IQR(x, na.rm = na.rm)
 y <- x
 y[x < (qnt[1] - H)] <- NA
 y[x > (qnt[2] + H)] <- NA
```

```
  y
}
youtubeClean <- youtubeRaw3
youtubeClean$views <- remove_outliers(youtubeRaw3$views)

#check new distribution, only keep data within distribution
boxplot(youtubeClean$views, main="Cleaned Boxplot Distribution of Videos Views", xlab="Trending
    Videos", ylab="Number of Views")
bwplot(views ~ category_id2 , data = youtubeClean, scales=list(x=list(rot=45)), main="Cleaned Boxplot
    Distribution of Videos Views", xlab="Trending Videos Categories", ylab="Number of Views")
youtubeClean2 <- youtubeClean[complete.cases(youtubeClean), ]
write.csv(youtubeClean2, file = "youtubeClean2.csv", row.names=FALSE)

#solve for actual target average view count and number of rows for strata
sum(as.numeric(youtubeClean2$views))
nrow(youtubeClean2)
max(youtubeClean2$views)
min(youtubeClean2$views)
mean(youtubeClean2$views)
sd(youtubeClean2$views)

#average amount of views for the outliers removed
(sum(as.numeric(youtubeRaw3$views)) - sum(as.numeric(youtubeClean2$views))) /
    (nrow(youtubeRaw3) - nrow(youtubeClean2))
max(youtubeRaw3$views)-max(youtubeClean2$views)
min(youtubeRaw3$views)-min(youtubeClean2$views)
mean(youtubeRaw3$views)-mean(youtubeClean2$views)
nrow(youtubeClean2)/nrow(youtubeRaw3)

#use sas for more youtubeClean2.csv dataset stats

#how do we choose MOE? currently, MOE = 5000 views
#do we remove outliers per strata or for the whole dataset? currently, we remove for whole dataset
#after removing outliers, 88% of the dataset is kept...do we ignore fpc adjustment? currently we ignore
    b/c more than 10%

n0srs <- ceiling((1.96^2*sd(youtubeClean2$views)^2)/(5000^2))
n0srs

#SRS function

SrsMeanEstimate<-function(Seed, SampSize, printOutput= TRUE){
  set.seed(Seed)

  youtubeClean2.SRSSampled = sample_n(youtubeClean2,SampSize)

  if(printOutput == TRUE){
    print(nrow(youtubeClean2.SRSSampled))
```

```
  print(bwplot(views ~ category_id2, data = youtubeClean2.SRSSampled, scales=list(x=list(rot=45)),
   main="SRS Boxplot Distribution of Videos Views", xlab="Trending Videos Categories", ylab="Number
   of Views"))
 }

 mydesign <- svydesign(id = ~1, data = youtubeClean2.SRSSampled)

 srsMean = svymean(~views, design = mydesign)
 srsSE = SE(srsMean)
 srsCI = confint(srsMean)

 rm(youtubeClean2.SRSSampled)
 rm(mydesign)

 return(list(as.numeric(srsMean[1]),
         as.numeric(srsSE),
         as.numeric(srsCI[1]),
         as.numeric(srsCI[2])))
}

srsMean <- SrsMeanEstimate(n0srs, n0srs)
print(paste('The Mean Estimate =', srsMean[[1]]))
print(paste('The Standard Error =', srsMean[[2]]))
mean(youtubeClean2$views)

#Proportional Strata

PropMeanEstimate<-function(Seed, SampSize, printOutput= TRUE){

 set.seed(Seed)

 # Identify Frequency of category_id2 Stratum
 PropFreq <- as.data.frame(table(youtubeClean2[,c("category_id2")]))
 names(PropFreq)[1] = 'category_id2'
 PropFreq

 PropFreq$N = nrow(youtubeClean2)
 PropFreq$p = PropFreq$Freq/PropFreq$N
 PropFreq$SampSizeh = (PropFreq$p * SampSize)
 PropFreq$SampSizehRounded = round(PropFreq$SampSizeh)

 youtubeClean2.PropSampled <- NULL

 for (i in 1:nrow(PropFreq)){
  youtubeClean2.PropSampled<-rbind(youtubeClean2.PropSampled,
                  sample_n(youtubeClean2[(youtubeClean2$category_id2 ==
   PropFreq[i,"category_id2"]),]
                           ,PropFreq[i,"SampSizehRounded"]))
```

```
 }

 if(printOutput == TRUE){
  print(PropFreq)
  print(nrow(youtubeClean2.PropSampled))
  print(bwplot(views ~ category_id2, data = youtubeClean2.PropSampled, scales=list(x=list(rot=45)),
   main="Prop Boxplot Distribution of Videos Views", xlab="Trending Videos Categories", ylab="Number
   of Views"))
 }

 mydesign <- svydesign(id = ~1, strata = ~category_id2, data = youtubeClean2.PropSampled)

 propMean = svymean(~views, design = mydesign)
 propSE = SE(propMean)
 propCI = confint(propMean)

 rm(youtubeClean2.PropSampled)
 rm(mydesign)
 propCI = confint(propMean)
 return(list(as.numeric(propMean[1]),
         as.numeric(propSE),
         as.numeric(propCI[1]),
         as.numeric(propCI[2])))
}

#adjusting the sample size calculation?

propMean <- PropMeanEstimate(n0srs, n0srs)
print(paste('The Mean Estimate =', propMean[[1]]))
print(paste('The Standard Error =', propMean[[2]]))
mean(youtubeClean2$views)

#deff = se_complex/se_srs
deffProp = as.numeric(propMean[[2]]/srsMean[[2]])
deffProp

n0prop = ceiling(n0srs*deffProp)
n0prop

#prop adjusted for deff

propMean <- PropMeanEstimate(n0srs, n0prop)
print(paste('The Mean Estimate =', propMean[[1]]))
print(paste('The Standard Error =', propMean[[2]]))

#task 2

SeedList <- c(10000, 20000, 30000, 40000, 50000)
```

```
df<- NULL

#SRS Seed Executions
for (seed in SeedList){
  srsEstimate <- SrsMeanEstimate(seed, n0srs, FALSE)
  srsEstimate <- data.frame('SRS', seed, srsEstimate)
  names(srsEstimate) <- c("EstimateType","SeedValue", "MeanEstimate", "SE", "LowerCI", "UpperCI")
  df<- rbind(df,srsEstimate)
}

#Prop Seed Executions
for (seed in SeedList){
  PropEstimate <- PropMeanEstimate(seed, n0srs, FALSE)
  PropEstimate <- data.frame('Prop', seed, PropEstimate)
  names(PropEstimate) <- c("EstimateType","SeedValue", "MeanEstimate", "SE", "LowerCI", "UpperCI")
  df<- rbind(df,PropEstimate)
}

#Prop Seed Executions
for (seed in SeedList){
  PropEstimate <- PropMeanEstimate(seed, n0prop, FALSE)
  PropEstimate <- data.frame('Prop DE', seed, PropEstimate)
  names(PropEstimate) <- c("EstimateType","SeedValue", "MeanEstimate", "SE", "LowerCI", "UpperCI")
  df<- rbind(df,PropEstimate)
}

#Add True Mean Value, in-line with estimates
df$TrueMeanValue <- mean(youtubeClean2$views)

#Add Bool Value for whether the Conf Limit contains the True Mean Value
df$WithinConfLimit <- df$LowerCI <= df$TrueMeanValue & df$UpperCI >= df$TrueMeanValue

#Print Results
print(df)

winner = aggregate(df[, 3:7], list(df$EstimateType), mean)
winner
#Prop wins slightly
#What is the percentage that the actual value is in the 95% confidence intervals for each design?
#abs(94384.50-93891.7)/2565.645 = 19.20734%
#how do you phrase it? true value is within 19.20734% of standard error for SRS estimation?

winner$PercentFromTrueMean <- abs(winner$TrueMeanValue -
   winner$MeanEstimate)/winner$SE*100
print(winner)
```

Appendix: SAS Code

```
FILENAME REFFILE '/home/yaoy890/youtubeClean2.csv';

PROC IMPORT DATAFILE=REFFILE
   DBMS=CSV
   OUT=youtubeClean2;
   GETNAMES=YES;
RUN;

FILENAME REFFILE '/home/yaoy890/youtubeRaw3.csv';

PROC IMPORT DATAFILE=REFFILE
   DBMS=CSV
   OUT=youtubeRaw3;
   GETNAMES=YES;
RUN;

proc means data=youtubeRaw3 min q1 mean median q3 max stderr stddev;
var views;
run;

proc means data=youtubeClean2 min q1 mean median q3 max stderr stddev;
var views;
run;

proc means data=youtubeClean2 min q1 mean median q3 max stderr stddev;
class category_id2;
var views;
run;


PROC SURVEYSELECT DATA= youtubeClean2 METHOD= srs OUT=
srssample SAMPSIZE= 1847 SEED = 1847 STATS;

proc surveymeans data=srssample total=44506 mean stderr CLM;
  var views;
run;

*===;

PROC SURVEYSELECT DATA= youtubeClean2 METHOD= srs OUT=
srssample1 SAMPSIZE= 1847 SEED = 1000 STATS;

PROC SURVEYSELECT DATA= youtubeClean2 METHOD= srs OUT=
srssample2 SAMPSIZE= 1847 SEED = 2000 STATS;

PROC SURVEYSELECT DATA= youtubeClean2 METHOD= srs OUT=
srssample3 SAMPSIZE= 1847 SEED = 3000 STATS;
```

```
PROC SURVEYSELECT DATA= youtubeClean2 METHOD= srs OUT=
srssample4 SAMPSIZE= 1847 SEED = 4000 STATS;

PROC SURVEYSELECT DATA= youtubeClean2 METHOD= srs OUT=
srssample5 SAMPSIZE= 1847 SEED = 5000 STATS;

proc surveymeans data=srssample1 total=44506 mean stderr CLM stacking;
  var views;
  ods output statistics=stacking;
run;

proc surveymeans data=srssample2 total=44506 mean stderr CLM stacking;
  var views;
  ods output statistics=stacking;
run;

proc surveymeans data=srssample3 total=44506 mean stderr CLM stacking;
  var views;
  ods output statistics=stacking;
run;

proc surveymeans data=srssample4 total=44506 mean stderr CLM stacking;
  var views;
  ods output statistics=stacking;
run;

proc surveymeans data=srssample5 total=44506 mean stderr CLM stacking;
  var views;
  ods output statistics=stacking;
run;

*===;

proc sort data = youtubeClean2;
by category_id;
run;

proc means data = youtubeClean2;
class category_id;
var views;
run;

proc boxplot data=youtubeClean2;
plot views*category_id;
run;

proc means data=youtubeClean2 min q1 mean median q3 max stderr stddev;
```

```
class category_id;
var views;
run;

data strsizes;
input category_id _total_;
datalines;
1 898
2 2732
10 1165
15 13136
17 2307
19 1806
20 2824
22 2467
23 4991
24 201
25 6522
26 399
27 1043
28 3740
29 275
;

proc surveyselect data=youtubeClean2 method = srs out = propsample sampsize = (37, 113, 48, 545, 96,
   75, 117, 102, 207, 9, 271, 17, 43, 155, 12) seed=1847;
strata category_id;
title "Proportional allocation";
run;

proc surveymeans data = propsample sum clsum total = strsizes mean sum CLM CLSUM;
var views;
weight SamplingWeight;
strata category_id;
title "Proportional allocation";
run;
```