Generate the Unbalanced Data

The scenario is that we are dealing with 3 email campaigns that have different CTRs and we want to apply undersampling to normalize the CTR by the campaign so that to avoid any skewness and biased when we will build the Machine Learning model. The hypothetical dataset is the following:

- **Campaign A**: 5000 Observations with 10% CTR (approx)
- **Campaign B**: 10000 Observations with 20% CTR (approx)
- **Campaign C**: 1000 Observations with 30% CTR (approx)

Let's try to generate this random sample in R.

```
library(tidyverse)

set.seed(5)
df = rbind(data.frame(Campaign = "A", Click = rbinom(n=5000, size=1,
prob=0.1)),
           data.frame(Campaign = "B", Click = rbinom(n=10000, size=1,
prob=0.2)),
           data.frame(Campaign = "C", Click = rbinom(n=1000, size=1,
prob=0.3)))

head(df)
```

Output:

```
  Campaign Click
1        A     0
2        A     0
3        A     1
4        A     0
5        A     0
6        A     0
```

Let's get the CTR by Campaign

```
df%>%group_by(Campaign)%>%
    summarise(CTR=mean(Click))
```

Output:

```
# A tibble: 3 x 2
  Campaign   CTR

1 A        0.106
2 B        0.198
3 C        0.302
```

As we can see the A campaign has 10.6% CTR, the B 19.8% and the C 30.2%. Let's add also a random column called attribute which takes the values "X", "Y", "Z" since we will deal with datasets with more than two columns.

```
df$Attribute<-sample(c("X","Y", "Z"), size = dim(df)[1], replace =
TRUE, prob = c(0.2, 0.6, 0.2))
head(df)

  Campaign Click Attribute
1        A     0         Y
2        A     0         Y
3        A     1         Z
4        A     0         Y
5        A     0         Z
6        A     0         Z
```

Now, our goal is to apply undersampling so that each campaign will have around 50% CTR

Undersampling by Group

We will use the `map2` function from the `purrr` package which belongs to the `tidyverse` family:

```
campaign_summary <- df %>% group_by(Campaign)%>%
summarize(rr=sum(Click)/n(), pos= sum(Click))


df_neg_sample<- df %>% filter(Click==0) %>%
  group_by(Campaign) %>%
  nest() %>%              #group all data by campaign name
  ungroup() %>%
  inner_join(campaign_summary, by="Campaign")



sampled_df_neg<-df_neg_sample %>%
  mutate(samp = map2(data, pos, sample_n, replace = FALSE))  %>%#
sample based on the campaing summary
  select(-data) %>%  #remove original nested data
  unnest(samp) %>% select(c(-"rr",-"pos"))



df_pos <- df %>% filter(Click==1) #positive samples
new_df <- rbind(df_pos,sampled_df_neg) #balanced set positive negative
within each campaign

head(new_df)

  Campaign Click Attribute
1        A     1         Z
2        A     1         Y
3        A     1         Y
4        A     1         Y
5        A     1         Y
6        A     1         Y
```

```
tail(new_df)

     Campaign Click Attribute
5613        C     0         Y
5614        C     0         Y
5615        C     0         Z
5616        C     0         Y
5617        C     0         Y
5618        C     0         X
```

Let's check if the `new_df` is balanced by campaign. We will group by campaign and we will show the CTR and the number of observations:

```
new_df%>%group_by(Campaign)%>%summarise(CTR=mean(Click),
Observations=n())

  Campaign    CTR Observations

1 A           0.5         1064
2 B           0.5         3950
3 C           0.5          604
```

As we can see, we sacrificed a sample but we have a balanced number of classes for every campaign (50-50).

Undersampling by Group using the ROSE Package

We can use also the ROSE package. Below we will apply a for loop by campaign so that to get a balanced sample using the undersampling technique.

```
library(ROSE)


balanced_sample = NULL


for (c in unique(df$Campaign)) {
  tmp_df = df%>%filter(Campaign==c)
  tmp<-ovun.sample(Click ~ ., data = tmp_df, method = "under", p = 0.5,
seed = 5)$data
  balanced_sample<-rbind(balanced_sample, tmp)
 }
```

Let's check if the `balanced_sample` is actually balanced.

```
balanced_sample%>%group_by(Campaign)%>%summarise(CTR=mean(Click),
Observations=n())
```

Output:

```
  Campaign    CTR Observations
```

```
1 A         0.504         1056
2 B         0.496         3978
3 C         0.510          592
```

Awesome. We showed two different approaches of how you can apply undersampling by group.