My M&M OCD

Yoni

03 04, 2025

Intro

The goal of this simulation is to test the statistics of M&M and other stacks even Chocolate lentils by color, I wanted to know, if I eat m&m package 2 by 2, separated by color, what is the chance of my finishing the package without mixing any color in one bite.

In addition, here are some BI incite that needed to be checked:

1. What is the probability of M&M packages packaged fairly?
2. What is the probability of M&M packages packaged without one color?
3. How does the size of the package or number of colors affect this probability?

The method is based of simulation of some M&M bags, according to the most common sizes. Each time we sample x lentils, name them by colors (represented as factorial numbers), and see the results for many packages as a statistic data.

Parameters

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
##
## Attaching package: 'MASS'
##
##
## The following object is masked from 'package:dplyr':
##
##     select
##
##
##
```

```
## Attaching package: 'scales'
##
##
## The following object is masked from 'package:purrr':
##
##     discard
##
##
## The following object is masked from 'package:readr':
##
##     col_factor
```

The basic parameters (will be changed later):

```
#parameters
nn<- 500             #numbers of bags per sample
n_color<- 6          #unique colors of M&M
gram<- 0.91          #weight of one M&M
bag_g<- 250          #common weight of M&M package
n_unit<- bag_g/gram #M&M per package
av_per_color= n_unit/n_color
paste0("The avarage number of lentils per color is ", round(av_per_color,2))
```

```
## [1] "The avarage number of lentils per color is 45.79"
```

Creating of Sample

General Sample

create_bag is a function to create one snack package as matrix.

sample_MnM is a function to create n bags from the create_bag function.

```
## [1] "One bag:"
```

```
##       1  2  3  4  5  6
## [1,] 18 14 11 20 22 16
```

```
## [1] "3 bags:"
```

```
##       1 2 3 4 5 6
## Bag_1 0 1 3 4 2 1
## Bag_2 4 0 3 3 1 0
## Bag_3 1 1 1 2 3 3
```

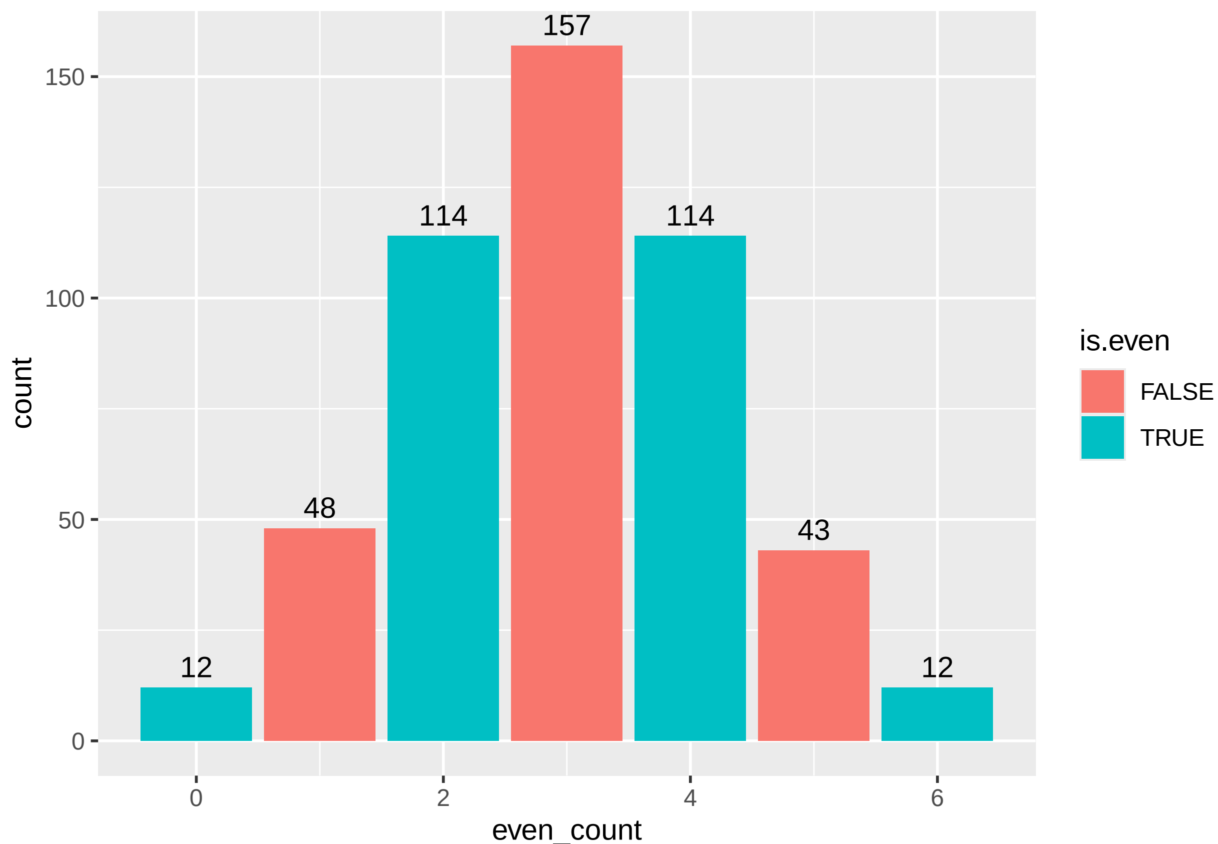
Preview Graph

Now will be creating nn bugs of M&M

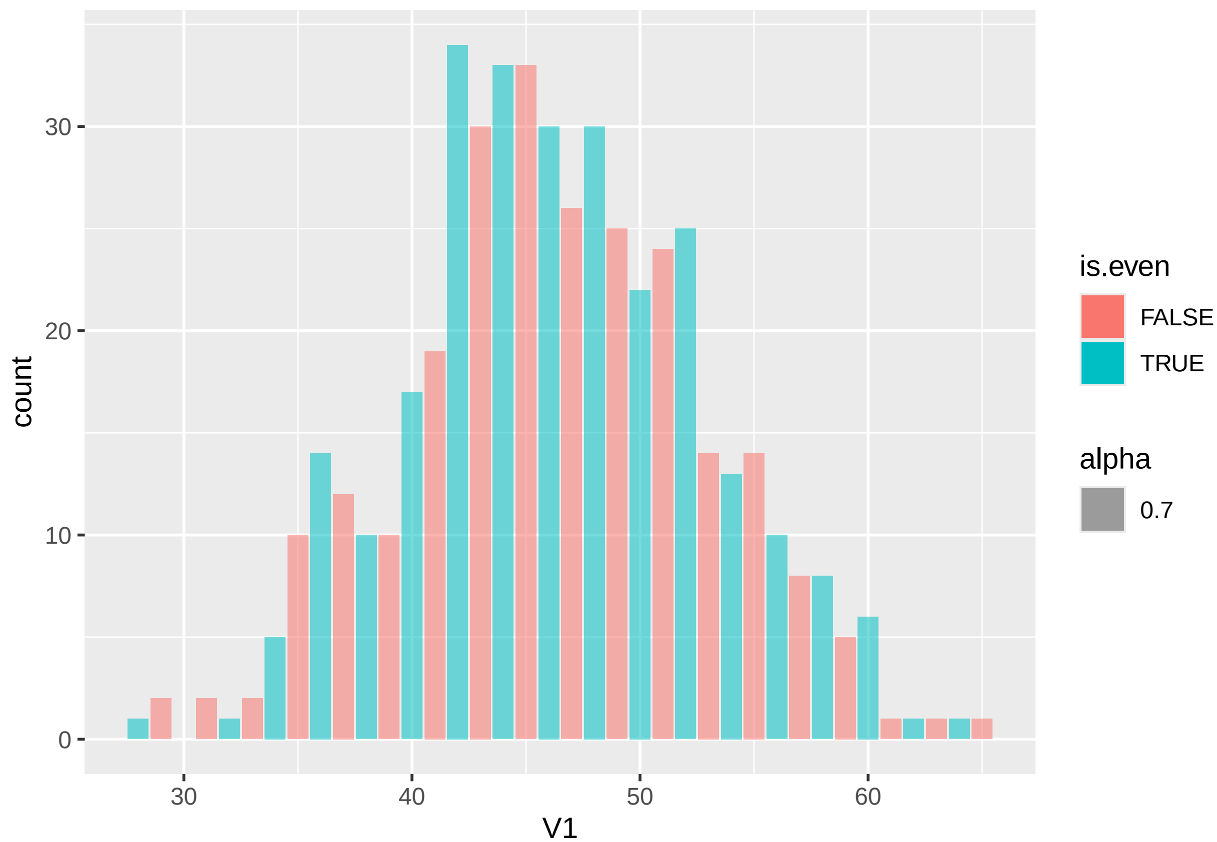
```
MnM_sample<- sample_MnM(nn,n_unit,n_color)

MnM_sample<-
  MnM_sample %>% as_data_frame() %>%
  mutate(even_count= rowSums(across(everything()  , ~ .x %% 2 == 0)), #how many evens colors there are
         even_evens= rowSums(across(c(1:6), ~ .x %% 2 == 1)) %% 2 ==0, #are the uneven colors even
         Variance=    apply(across(c(1:6)), 1, var), #var of candy per color/ type
         low_col=    rowSums(across(c(1:6), ~ .x <= 0.6*av_per_color)),
         min=        apply(across(c(1:6)),1,FUN = min)) # lowest value in color
```

plot the MnM sample sample

```
MnM_sample %>%
  mutate(is.even = even_count %%2 ==0) %>%
  ggplot(aes(x= even_count, fill= is.even))+
  geom_bar()+
    geom_text(
    stat = "count",
    aes(label = ..count..),vjust = -0.5
  )
```




```
MnM_sample %>%
  mutate(is.even = V1 %%2 ==0) %>%
  ggplot(aes(x= V1, fill= is.even, alpha= 0.7))+
  geom_bar()
```




```
#summary of all colors
rbind(
MnM_sample$V1 %>% summary(),
MnM_sample$V2 %>% summary(),
MnM_sample$V3 %>% summary(),
MnM_sample$V4 %>% summary(),
MnM_sample$V5 %>% summary(),
MnM_sample$V6 %>% summary()
) %>% as.data.frame() %>% cbind(sapply(MnM_sample[,1:6],var,na.rm=1)) %>%
  rename("Var" ="sapply(MnM_sample[, 1:6], var, na.rm = 1)")
```

```
##    Min. 1st Qu. Median   Mean 3rd Qu. Max.      Var
## V1   28      42   46.0 46.306      51   65 41.69976
## V2   31      41   45.0 45.650      50   65 35.45842
## V3   30      41   45.0 45.354      49   67 38.99868
## V4   24      41   46.0 45.862      50   66 40.17531
## V5   27      41   45.5 45.552      49   65 39.98326
## V6   25      42   46.0 45.772      50   67 34.98198
```

Test Expected Value

to see is the mu of the lentils per color are fair, we will test it per column with t.test for each color.

Here is the result, none of them bellow 5% P. value

```
##      V1      V2      V3      V4      V5      V6 
##  "7.3%" "60.6%" "12.1%" "79.3%" "40.5%" "95.3%"
```

Now we will do the same checking for 2 samples, to see whether there is correlation between each 2 colors distribution.

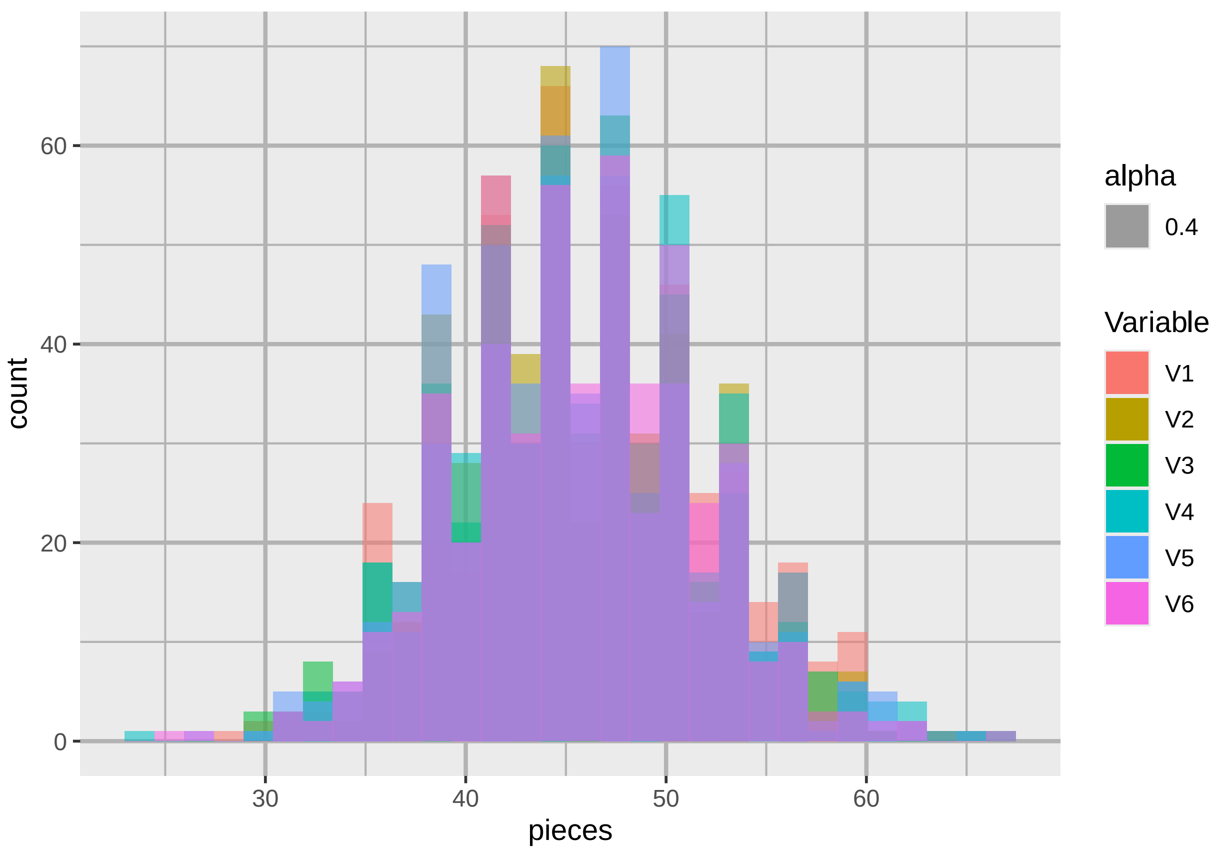
for each row i and column j, 1) if i==j, this it the check from before of the expected value to n_unit/n_color 2) if i!=j, this is two samples test of same expected value hypothesis

```
##        V1     V2     V3     V4     V5     V6
## V1 0.0732 0.0952 0.0180 0.2728 0.0624 0.1730
## V2 0.0952 0.6057 0.4432 0.5858 0.8009 0.7452
## V3 0.0180 0.4432 0.1212 0.2020 0.6185 0.2774
## V4 0.2728 0.5858 0.2020 0.7929 0.4390 0.8165
## V5 0.0624 0.8009 0.6185 0.4390 0.4053 0.5700
## V6 0.1730 0.7452 0.2774 0.8165 0.5700 0.9532
```

and lets see it visually:

```
MnM_sample %>%
    pivot_longer(cols = 1:6, names_to = "Variable", values_to = "pieces") %>%
    ggplot(aes(fill=Variable ,x= pieces, alpha= 0.4))+
  geom_histogram(position = "identity")+
  theme(panel.grid = element_line(color = "grey70",
                                  size = 0.75))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



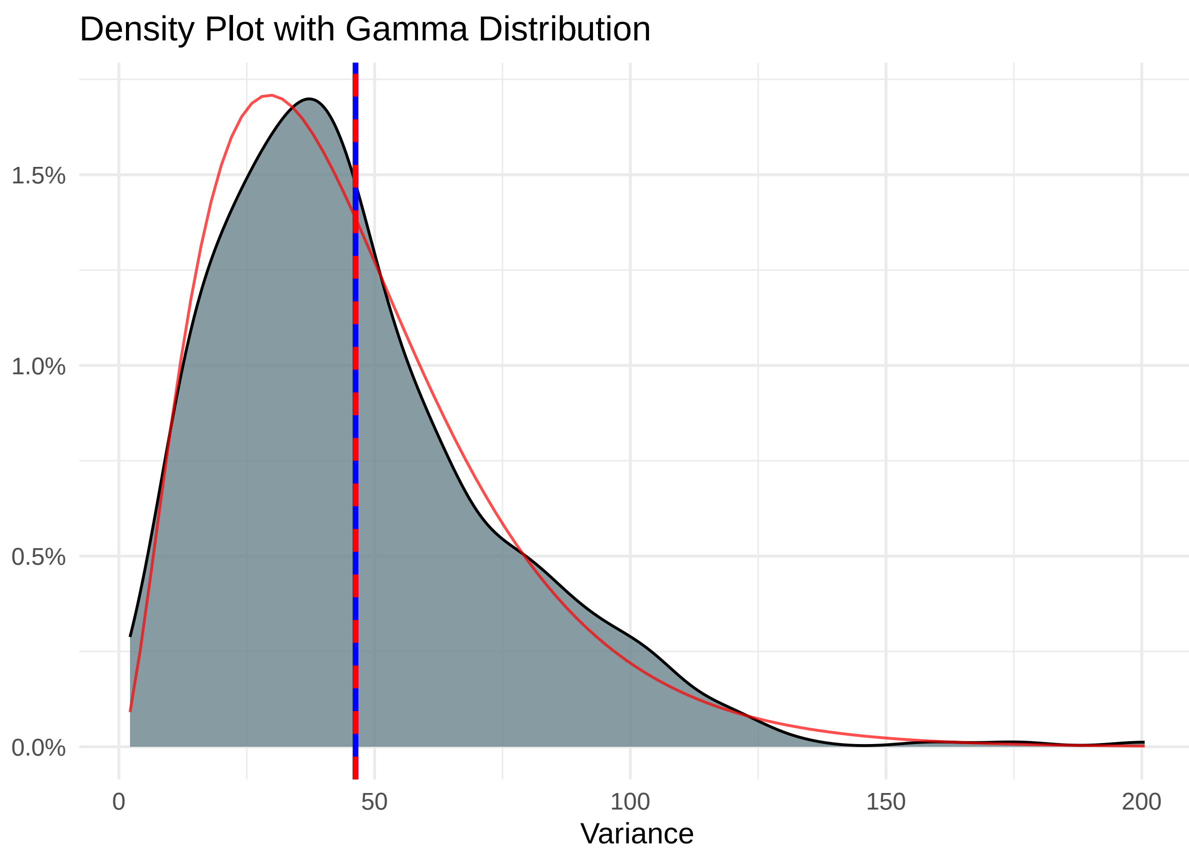
Variance Distribution Checking

W know that the distribution of variance is approximately Gamma distribution:

$$f(x) = \frac{1}{(\Gamma(\alpha)\theta^{\alpha})} x^{\alpha-1} e^{-x/\theta}$$

We can see that the variance distribution is Gamma like with shape and rate as seen below

```
## [1] "The parameters of the gamma shaped variance is shape 2.744 and rate 0.059"
```



#check too low color (under 10%) and sample by n number

#use statistics to sample better low chance cases

n*m types of snacks

We will create a function that create sample for each number of colors and package size we want, and then calculate some interesting parameters

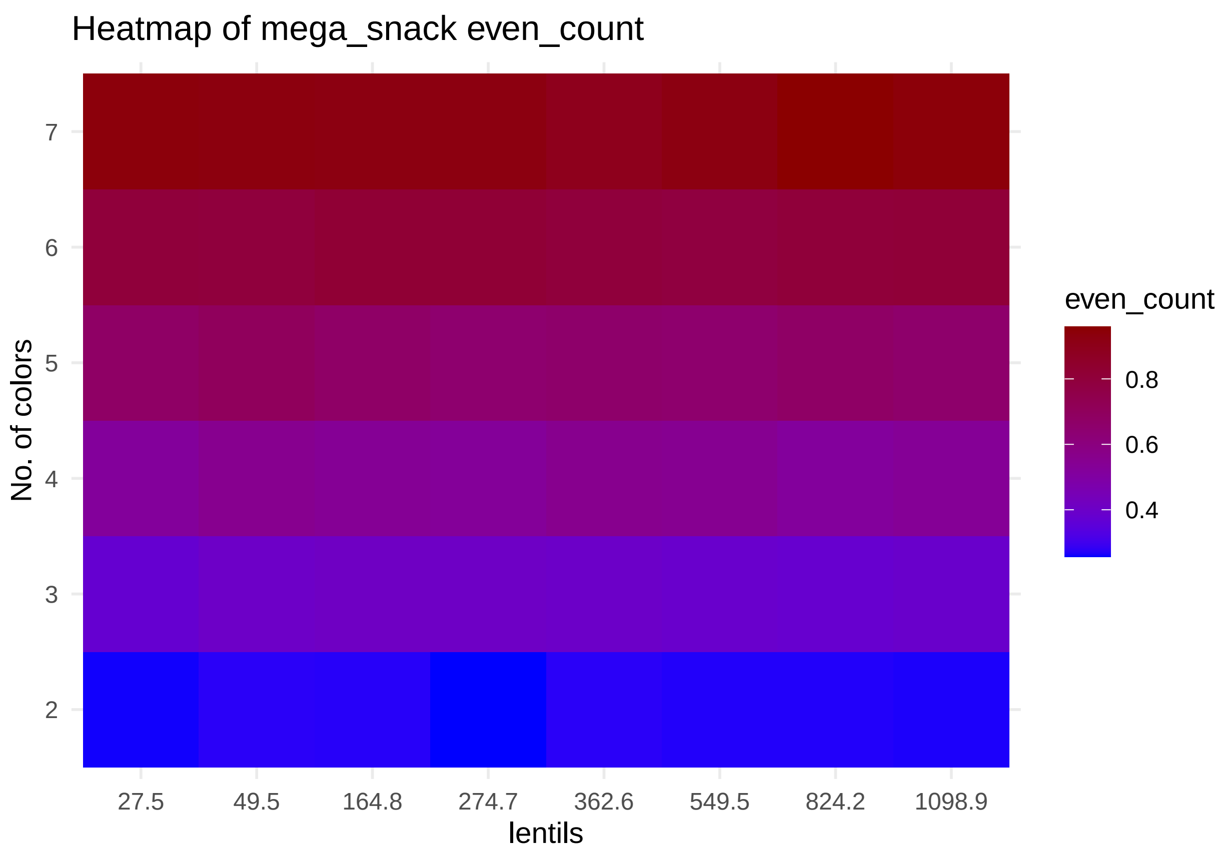
```
mega_snack<- function(nn,n_unit,n_color)
  {
  m_sample<- length(n_unit)*length(n_color)
  nul_mat= matrix(,nrow = m_sample, ncol = 5)
  res<- cbind(rep(n_unit,length(n_color)),sort(rep(n_color,length(n_unit))),
                                               nul_mat)
```

```r
  for (i in 1:(dim(res)[1]))
  {
    #print(c(res[i,1],res[i,2]))
    low_color<- 0.666*res[i,1]/(res[i,2])
    small_sample<- sample_MnM(nn,res[i,1],res[i,2])
    small_sample<-
      small_sample %>% as_data_frame() %>%
      mutate(even_count= rowSums(across(everything()  , ~ .x %% 2 == 0))/n_color, #how many evens colors
             even_evens= (rowSums(across(c(1:n_color)  , ~ .x %% 2 == 1)) %% 2 ==0)/n_color, #are the u
             var_col=    apply(across(c(1:n_color)), 1, var), #var of candy per color/ type
             all_even=   rowSums(across(c(1:n_color)  , ~ .x %% 2 == 0))== n_color,
             low_col=   rowSums(across(c(1:n_color), ~ .x <= low_color ))>=1)
    res[i,3]<- mean(small_sample$even_count)
    res[i,4]<- mean(small_sample$even_evens)
    res[i,5]<- mean(small_sample$var_col)
    res[i,6]<- mean(small_sample$all_even)
    res[i,7]<- mean(small_sample$low_col)
      }
  colnames(res)<- c("n_unit", "n_color", "even_count", "even_evens", "var_col","all_even","low_color")
  res
  }
```

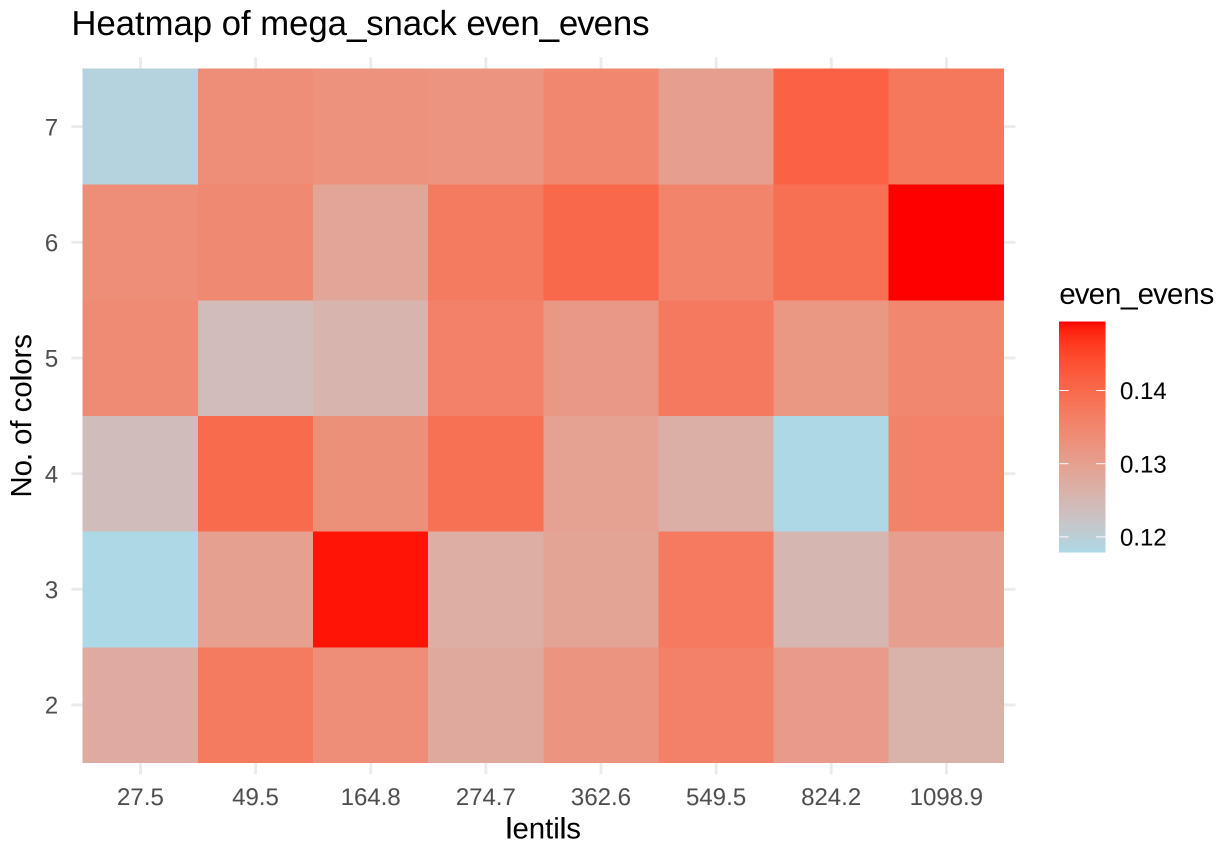
```r
color_op<- 2:7
grams_op<- c(25,45,150,250,330,500,750,1000)
n_unit_op<- grams_op/gram

mega_snack_1<-
  mega_snack(500,n_unit_op,color_op) %>% as.data.frame() %>%
  mutate(n_unit= round(n_unit,1))

mega_snack_1 %>% head(6)
```

```
##   n_unit n_color even_count even_evens var_col all_even low_color
## 1   27.5       2  0.2617952  0.1278476  14.119    0.036     0.106
## 2   49.5       2  0.2759857  0.1370857  22.828    0.050     0.014
## 3  164.8       2  0.2734286  0.1334905  81.972    0.034     0.000
## 4  274.7       2  0.2576095  0.1280429 143.465    0.032     0.000
## 5  362.6       2  0.2763476  0.1321333 165.056    0.032     0.000
## 6  549.5       2  0.2701000  0.1359429 281.674    0.038     0.000
```

```r
mega_snack_1 %>%
  ggplot(aes(x = factor((round( n_unit,1) )), y = factor(n_color ), fill = even_count )) +
  geom_tile() +
  scale_fill_gradient(low = "blue", high = "red4")+
  labs(title = "Heatmap of mega_snack even_count",
       x = "lentils",
       y = "No. of colors",
       fill = "even_count") +
  theme_minimal()
```

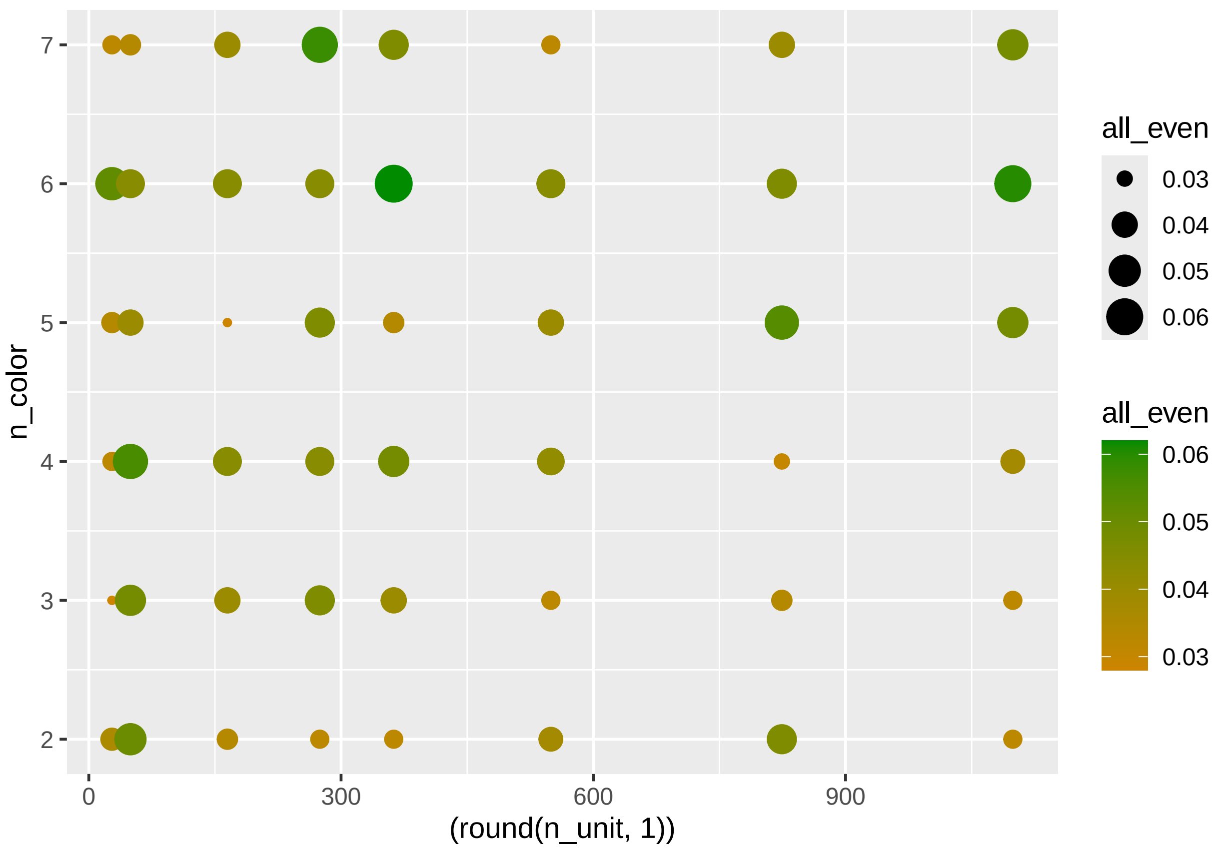



```r
mega_snack_1 %>%
  ggplot(aes(x = factor((round( n_unit,1) )), y = factor(n_color ), fill = even_evens )) +
  geom_tile() +
  scale_fill_gradient(low = "lightblue", high = "red")+
  labs(title = "Heatmap of mega_snack even_evens",
       x = "lentils",
       y = "No. of colors",
       fill = "even_evens") +
  theme_minimal()
```



now let us see the probability of all even, and whether there is pattern.

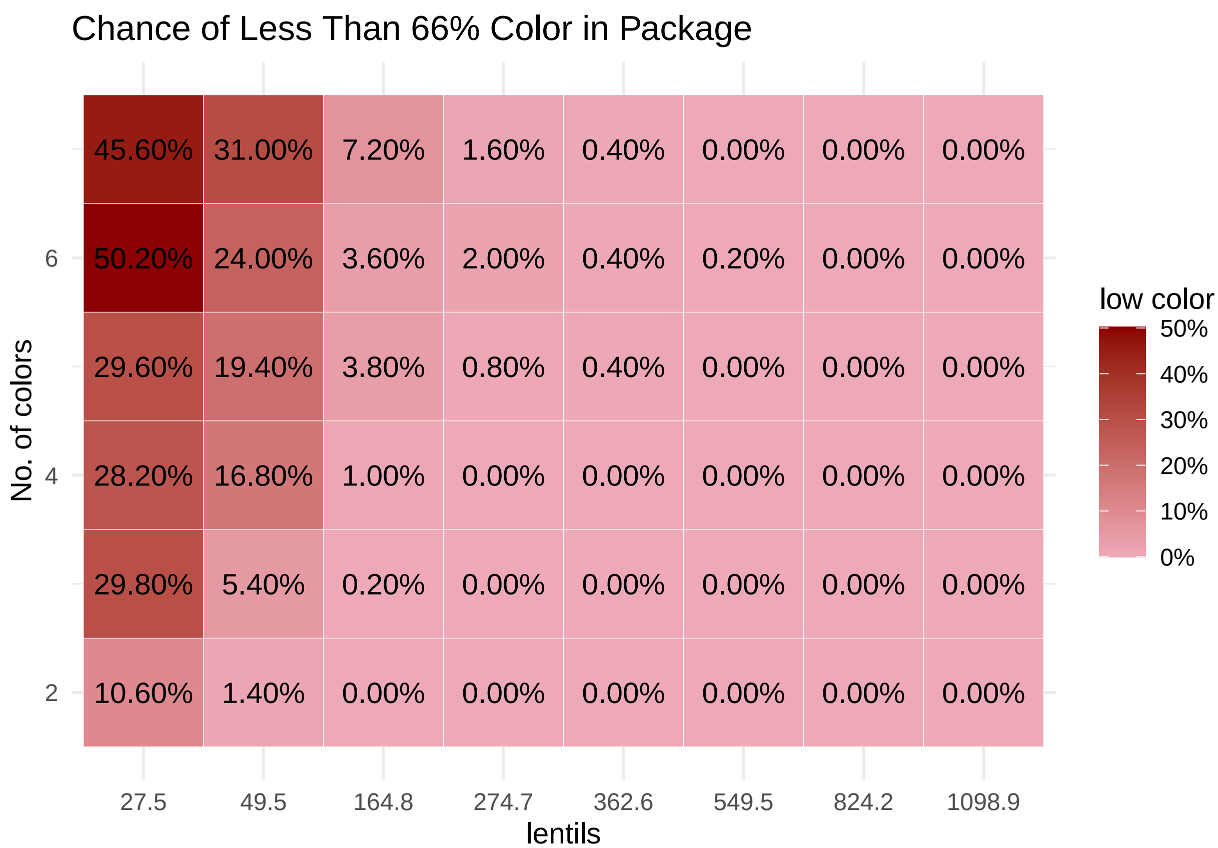
```
mega_snack_1 %>%
  ggplot(aes(x = (round( n_unit,1) ), y = n_color , color = all_even,size = all_even )) +
  geom_point() +
  scale_color_gradient(low = "orange3", high = "green4")
```




```
  labs(title = "Heatmap of mega_snack all evens",
       x = "lentils",
       y = "No. of colors",
       color = "all_even") +
  theme_minimal()
```

```
## NULL
```

```
mega_snack_1 %>%
  ggplot(aes(x = factor(round(n_unit, 1)), y = n_color, fill = low_color))+
  geom_tile(color = "white") +
    geom_text(aes(label = sprintf("%.2f%%", low_color * 100)), color = "black", size = 4)+
  scale_fill_gradient(low = "pink2", high = "red4",labels = scales::percent)+
  labs(title = "Chance of Less Than 66% Color in Package",
       x = "lentils",
       y = "No. of colors",
       fill = "low color")+
  theme_minimal()
```



As we can see, only the small package (less than 50 lentils) have high probability of at least one color to appear severely lower.

Therefore, splitting package by color on the big ones should be relatively even.