My M&M OCD

Yoni

13 04, 2025

Intro

Objective of Simulation

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 what is the chance of my finishing the package of M&M without mixing any color in one bite, eating them 2 by 2

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?

Method

I do not have an inner information of how does (M&M)[https://www.mms.com/en-us] make their delicious snacks nor we know how they make sure each package have fair amount of each color. Therefore, the method I chose is based of simulation of some M&M bags, according to the most common sizes of packages.

Each time we sample x lentils(units of M&M), name them by colors (V1,V2...), and see the results for many packages as a statistic data.

my hypothesis is that the probability of perfect package (aka a package with all colors number been even) is very small, at least for a standard 6 colors pack.

Parameters

Basic parameters:

```
#parameters
n<- 800             #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 packagenm,
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 the Sample

General Sample

In order to test the theoretical data, we need to simulate it using customize functions. here are there:

- Create_bag- function to create one snack package for chosen package size and number of colors.
- sample_MnM- function to create n bags from the Create_bag function.

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

```
##       1  2  3  4  5  6
## [1,] 18 15 17 25 12 14
```

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

```
##       Red Blue Green Orange Yellow Brown
## Bag_1   2    2     1      0      3     2
## Bag_2   2    1     3      2      1     1
## Bag_3   0    3     1      1      2     3
```

Preview Graph

Now will be creating n bugs of M&M

columns:

1. V1:V6- the number of lentils per color
2. even_count- how many evens colors there are
3. even_evens- are the uneven colors even
4. Variance- variance of lentils per color
5. low_col- sum true if one color's count is lower than $\frac{2}{3}$ of expected value
6. min- the lowest color in each row

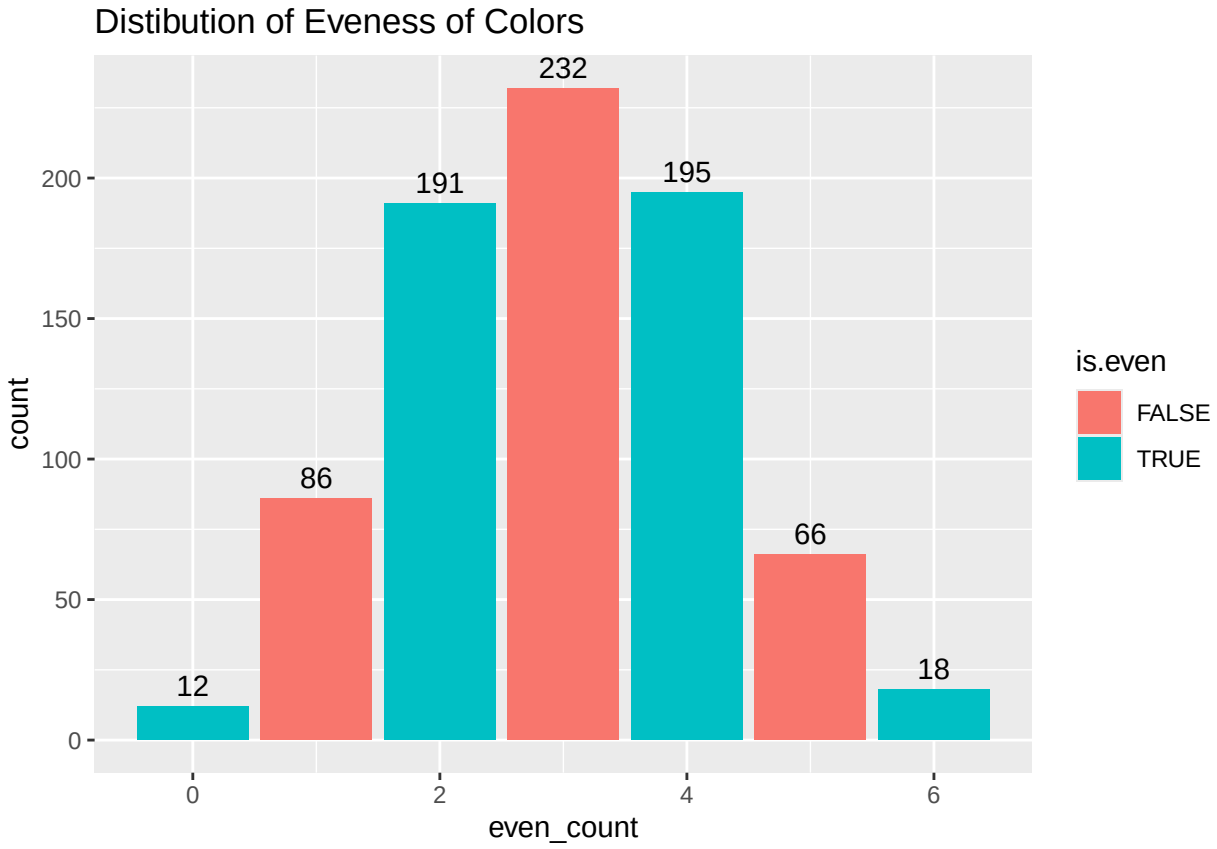
here are the first rows:

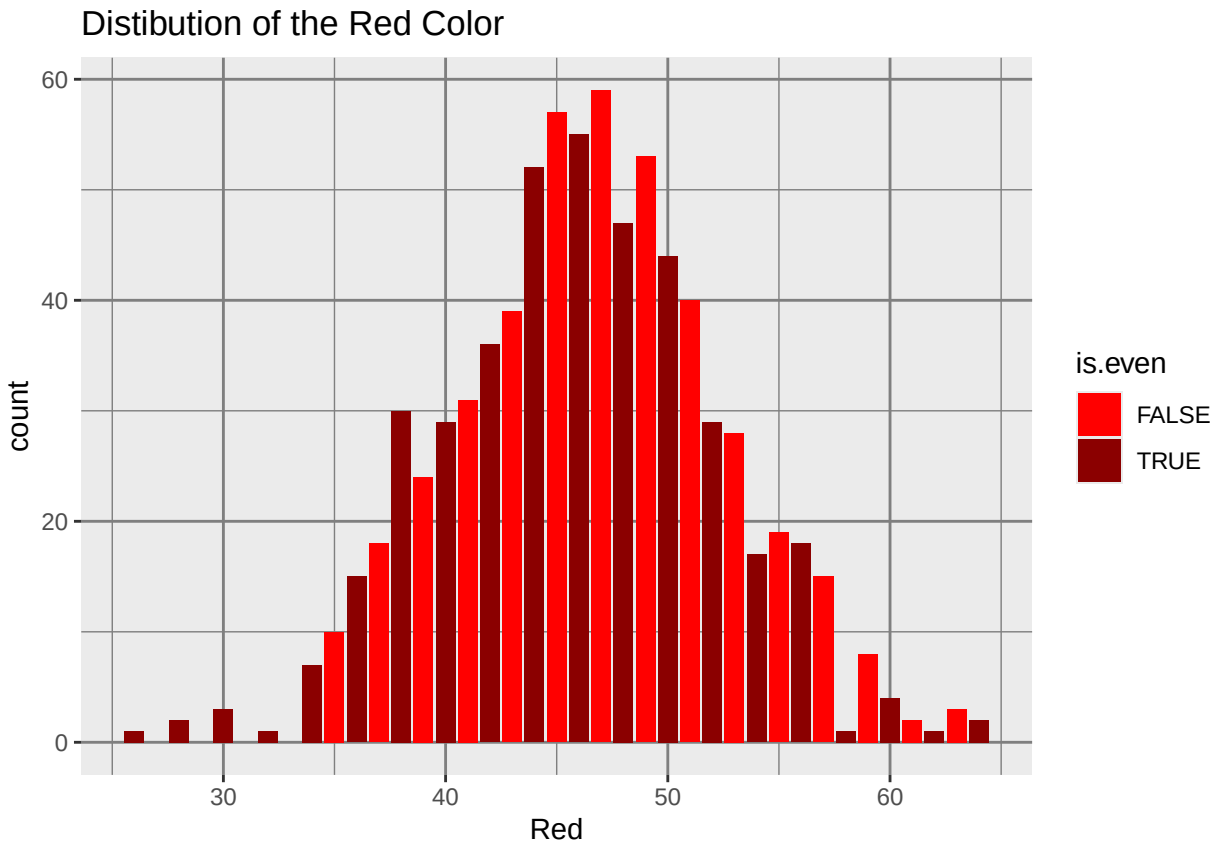
```
## # A tibble: 6 x 12
##     Red  Blue Green Orange Yellow Brown even_count even_evens low_col Variance
##   <int> <int> <int>  <int>  <int> <int>      <dbl> <lgl>        <dbl>    <dbl>
## 1    49    41    43     53     54    35          1 FALSE            0    55.4
## 2    47    46    49     48     48    36          4 TRUE             0    23.5
## 3    46    45    51     44     44    45          3 FALSE            0     6.97
## 4    37    40    34     53     57    53          2 TRUE             0    95.9
## 5    43    50    47     46     49    40          3 FALSE            0    14.2
## 6    41    54    42     48     40    49          4 TRUE             0    30.7
## # i 2 more variables: min <int>, all_even <lgl>
```

plot the M&M sample sample

```
## [1] "summary of all colors Distibution:"
```

```
##         Min. 1st Qu. Median     Mean 3rd Qu. Max.      Var
## Red       26      42   46.0 46.29000      50   64 36.06348
## Blue      27      41   46.0 45.47625      49   66 37.33360
## Green     31      42   46.0 45.92500      50   66 36.37484
## Orange    25      41   46.0 45.46250      49   70 39.83088
## Yellow    28      42   45.0 45.57875      50   64 34.87740
## Brown     29      41   45.5 45.74750      50   63 35.76345
```





Statisics Checking of the Simullation

Test Expected Value

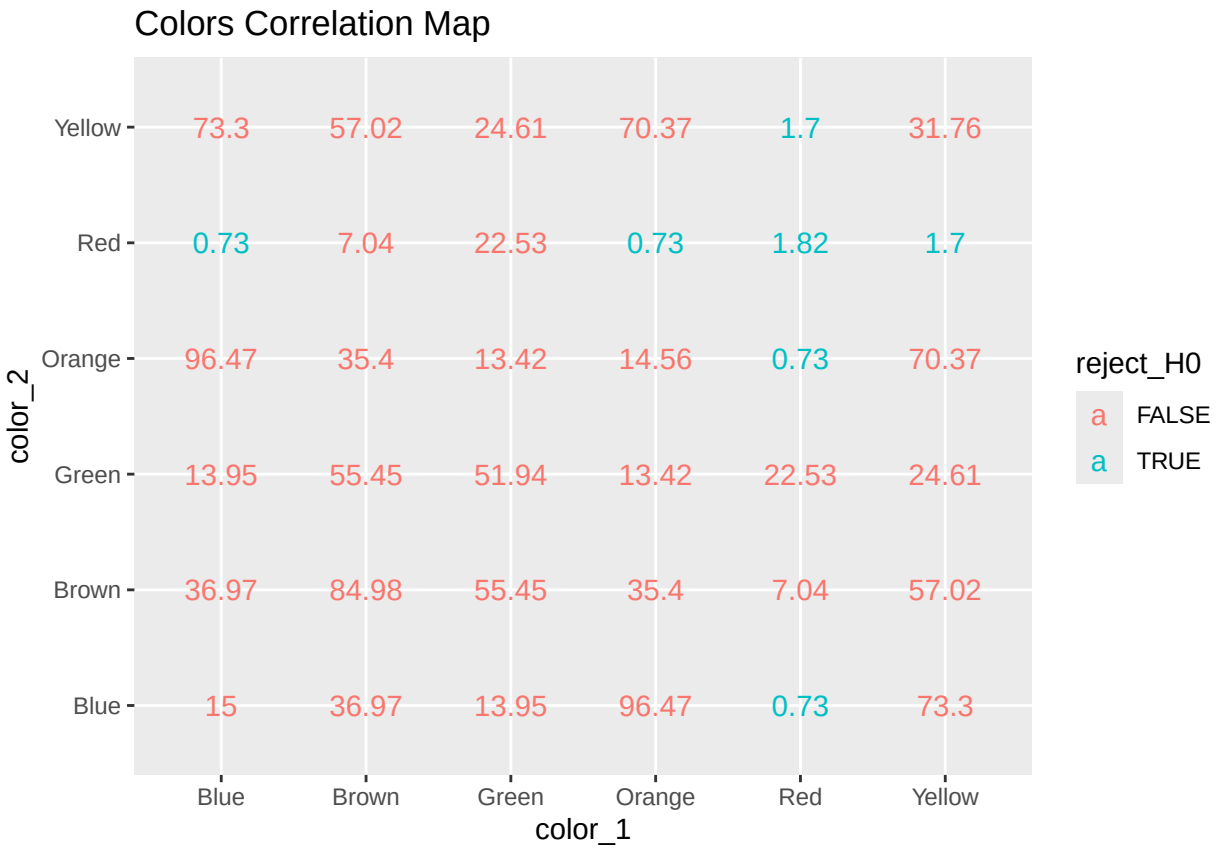
to see is the μ of the lentils per color are fair, we will test it per column with t.test for each color.

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

```
##      Red     Blue    Green   Orange   Yellow    Brown
##  "1.82%" "15.00%" "51.94%" "14.56%" "31.76%" "84.98%"
```

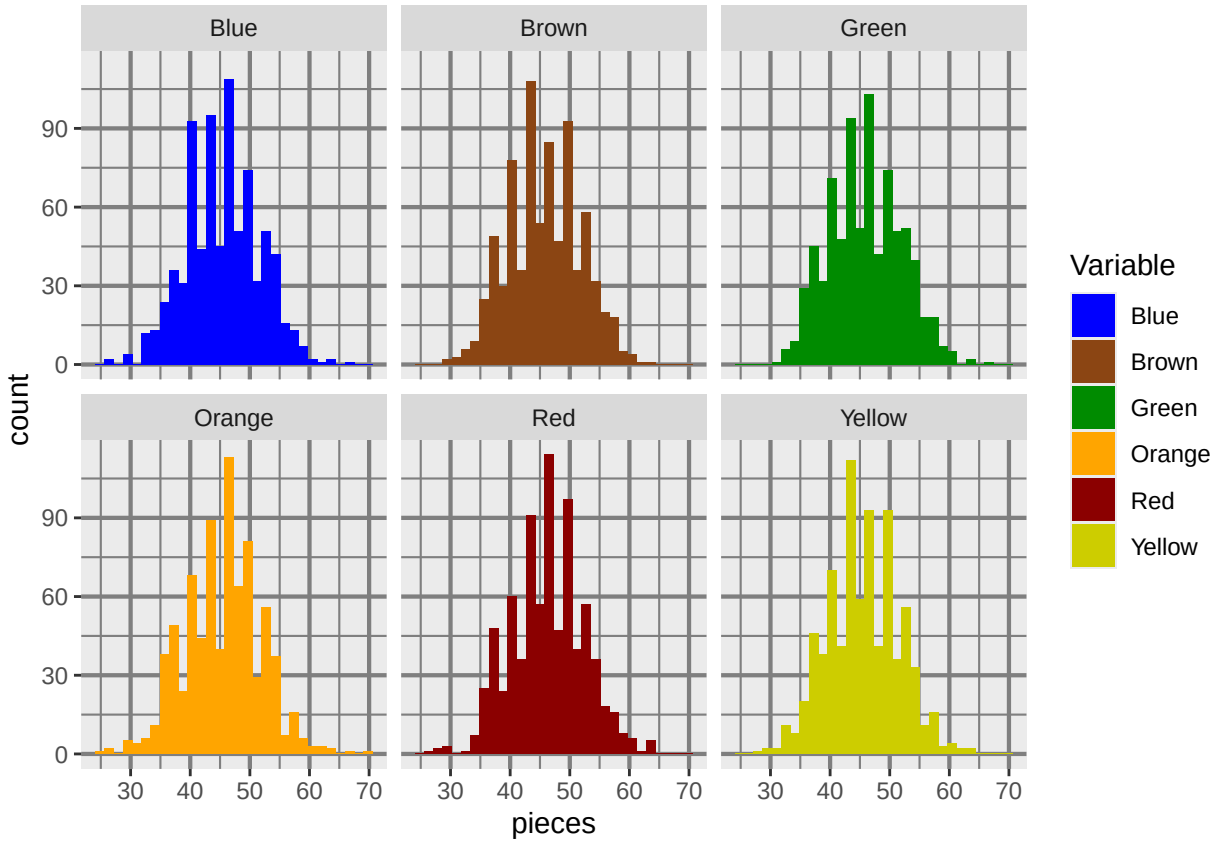
Now we will do the same checking for 2 samples, to see whether there is correlation between each 2 colors distribution.

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



Now here Is visualization of the actual data per color

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



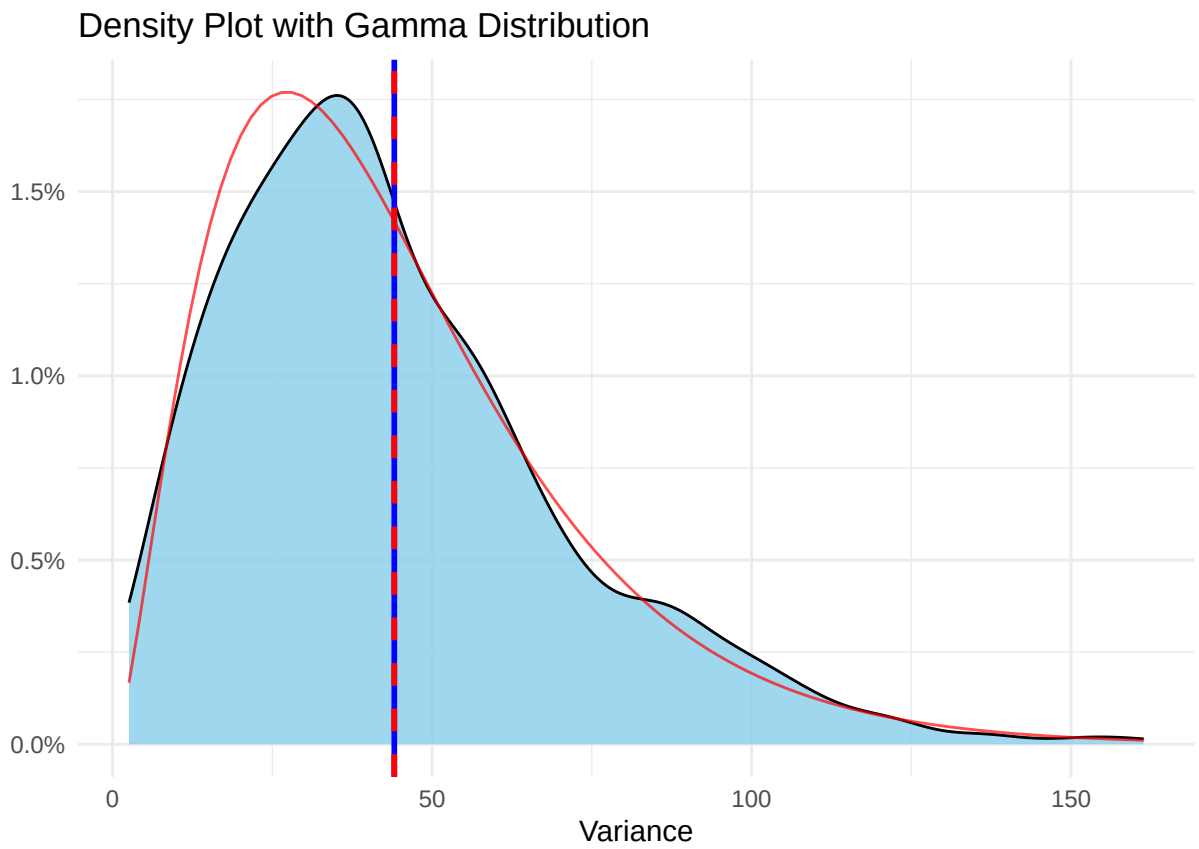
Variance Distribution Checking

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

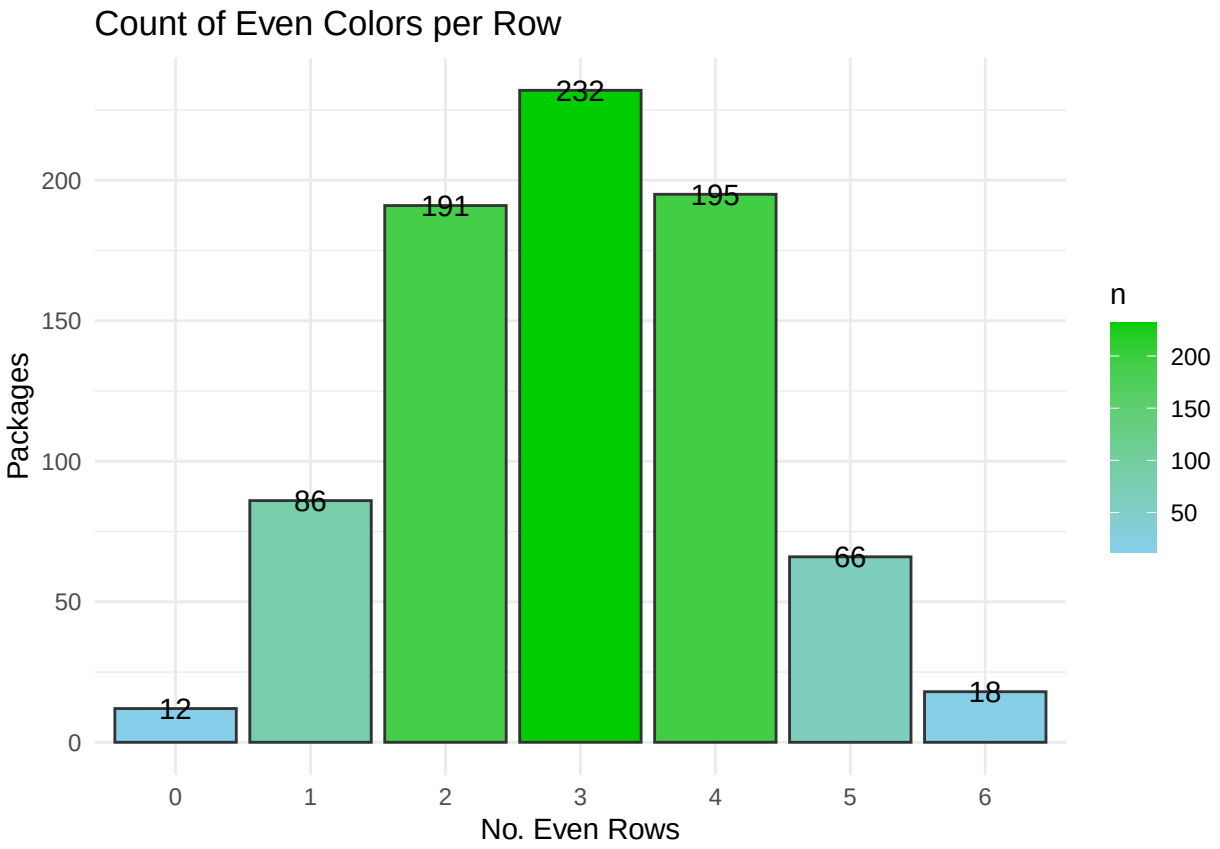
$$f(x) = \frac{1}{(\Gamma(\alpha)\theta^{\alpha})} x^{\alpha-1} e^{-x/\theta}$$

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 2.62 and rate 0.059"
```



Are All Even in the Sample?



n*m types of snacks

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

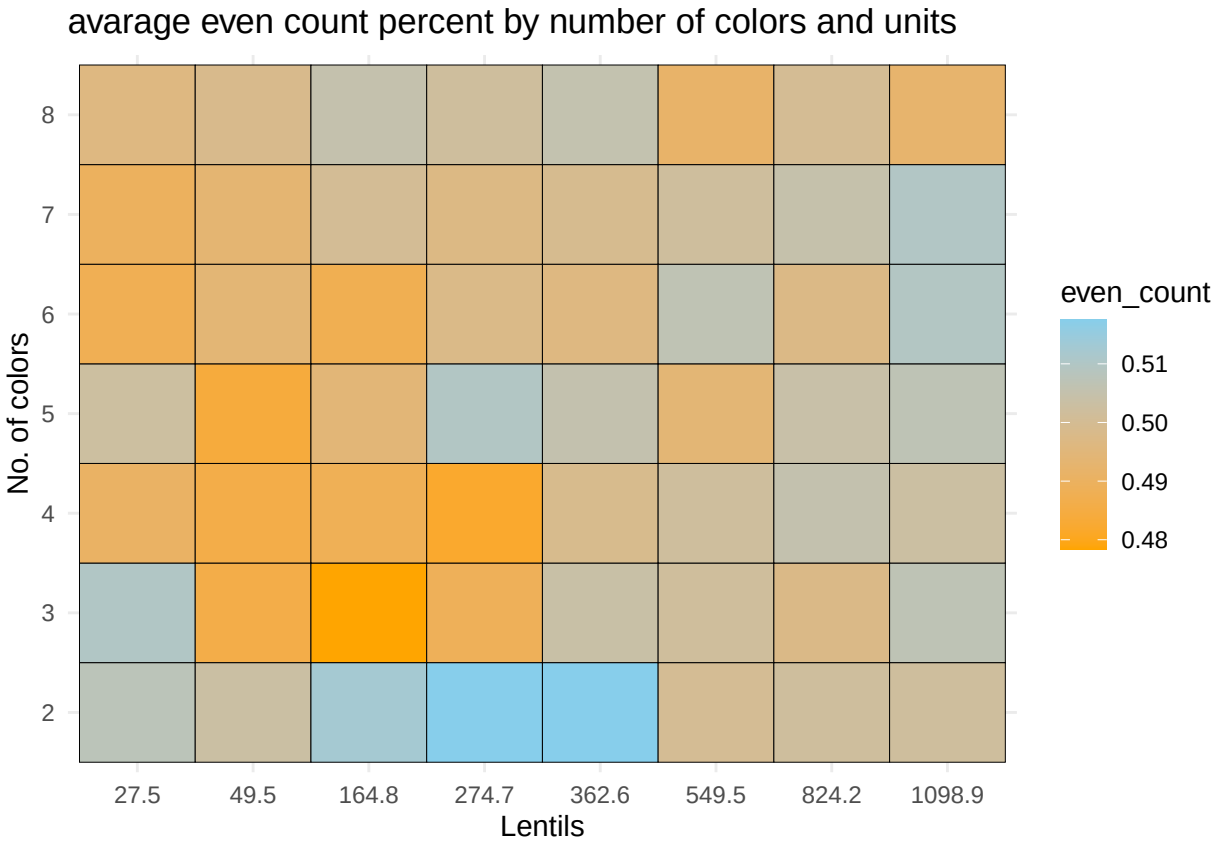
```
color_op<- 2:8
grams_op<- c(25,45,150,250,330,500,750,1000)
n_unit_op<- grams_op/gram
nn=800
```

We will make the multiple sample. Here is some random rows:

```
##   n_unit n_color even_count even_evens   var_col all_even low_color
## 1  164.8       4  0.4881250 0.12343750  41.13813  0.06750   0.00625
## 2  362.6       8  0.5054688 0.06328125  43.35938  0.00625   0.00625
## 3 1098.9       6  0.5095833 0.08041667 174.88875  0.02000   0.00000
## 4  164.8       5  0.4942500 0.10025000  34.95687  0.03625   0.01375
## 5   49.5       5  0.4840000 0.09700000  10.57250  0.01875   0.21500
##   smallest_col
## 1           22
## 2           28
## 3          142
## 4           17
## 5            2
```

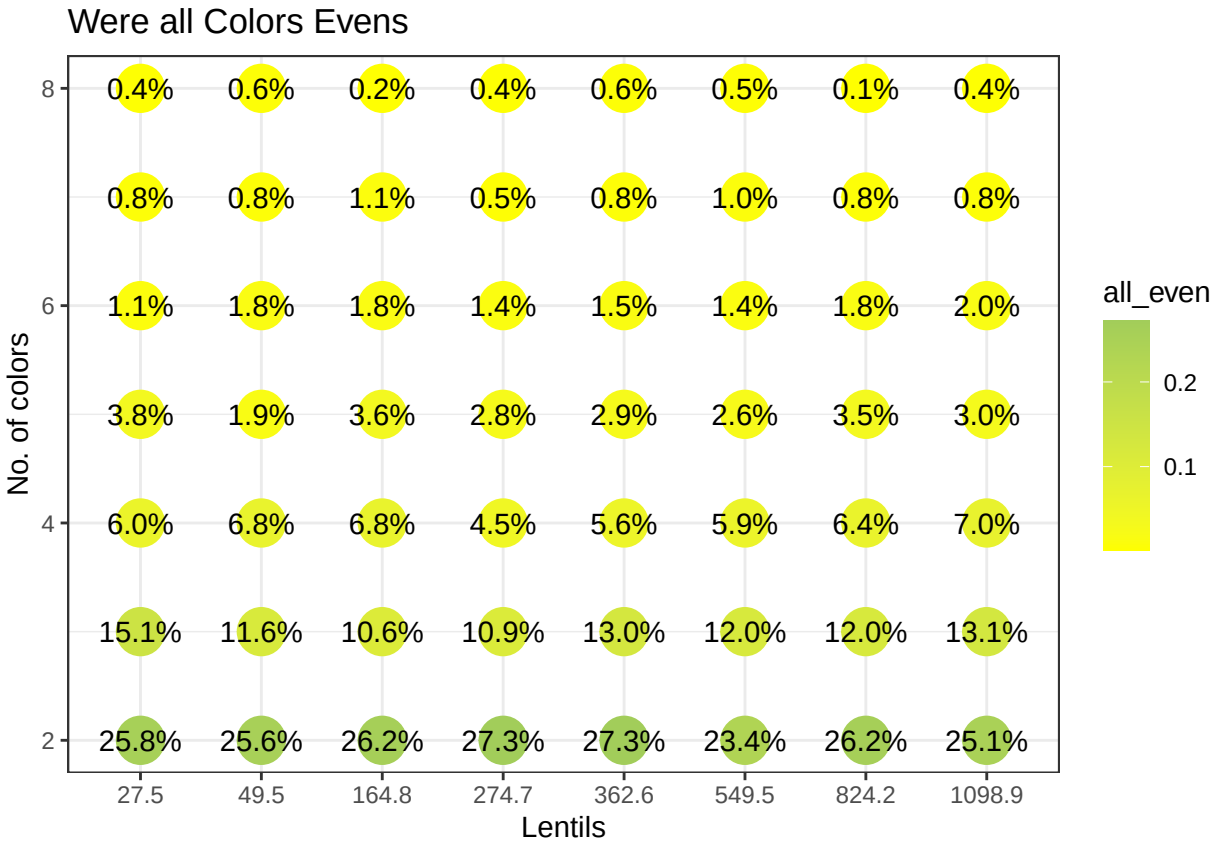
Deep Insight on the Data

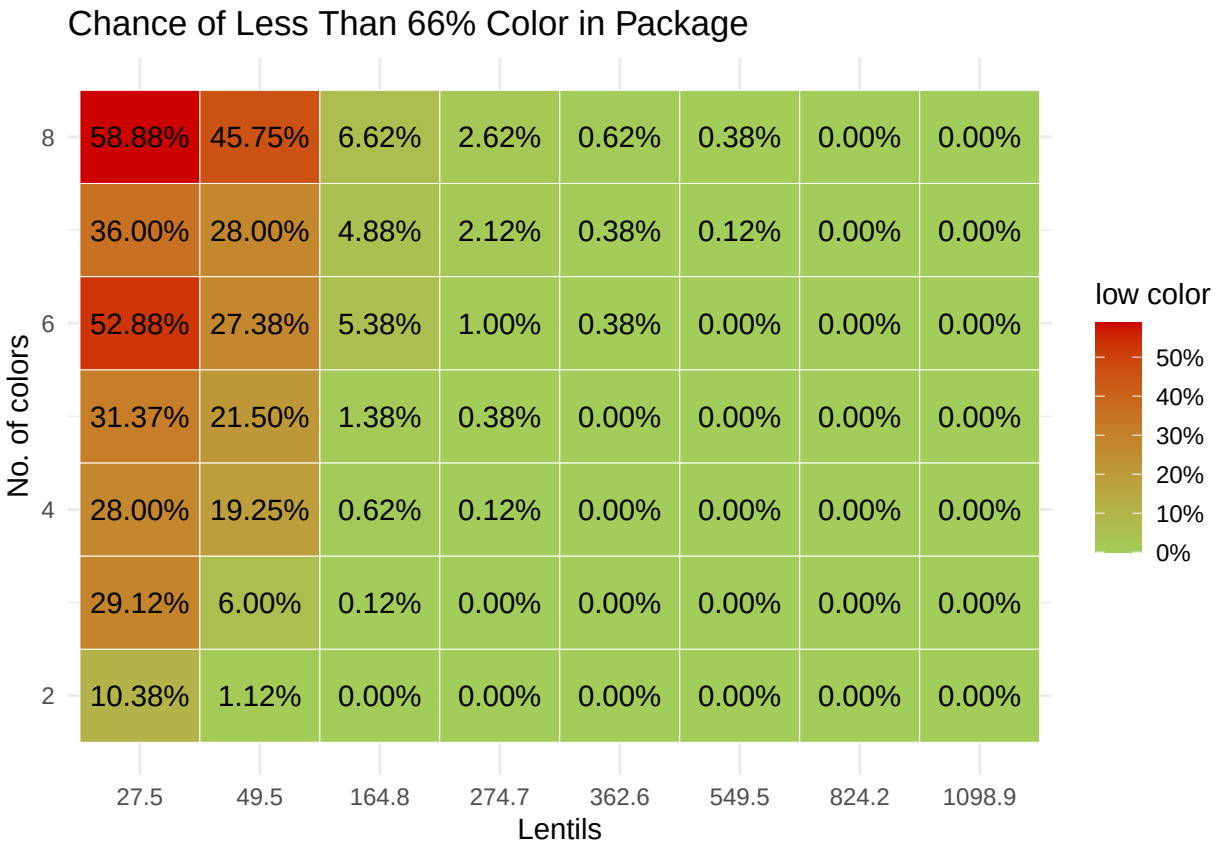
here are some insights:

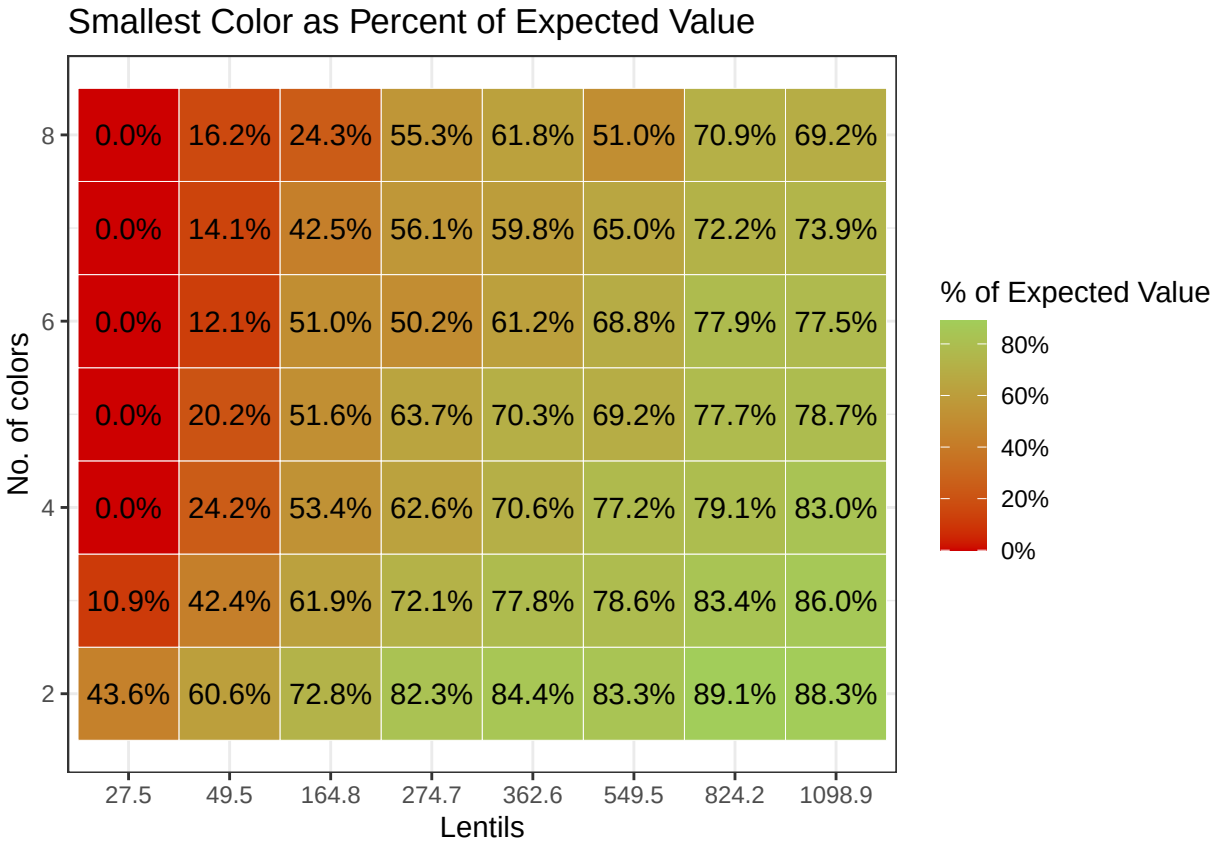


Here is probability of all even, and whether there is pattern.

```
## <Guides[1] ggproto object>
##
## colour : "none"
```


Were all Colors Evens
No. of colors
8
6
4
2
0.4% 0.6% 0.2% 0.4% 0.6% 0.5% 0.1% 0.4%
0.8% 0.8% 1.1% 0.5% 0.8% 1.0% 0.8% 0.8%
1.1% 1.8% 1.8% 1.4% 1.5% 1.4% 1.8% 2.0%
3.8% 1.9% 3.6% 2.8% 2.9% 2.6% 3.5% 3.0%
6.0% 6.8% 6.8% 4.5% 5.6% 5.9% 6.4% 7.0%
15.1% 11.6% 10.6% 10.9% 13.0% 12.0% 12.0% 13.1%
25.8% 25.6% 26.2% 27.3% 27.3% 23.4% 26.2% 25.1%
27.5
49.5
164.8
274.7
362.6
549.5
824.2
1098.9
Lentils
all_even
0.2
0.1


Chance of Less Than 66% Color in Package
No. of colors
8
6
4
2
58.88%
45.75%
6.62%
2.62%
0.62%
0.38%
0.00%
0.00%
36.00%
28.00%
4.88%
2.12%
0.38%
0.12%
0.00%
0.00%
52.88%
27.38%
5.38%
1.00%
0.38%
0.00%
0.00%
0.00%
31.37%
21.50%
1.38%
0.38%
0.00%
0.00%
0.00%
0.00%
28.00%
19.25%
0.62%
0.12%
0.00%
0.00%
0.00%
0.00%
29.12%
6.00%
0.12%
0.00%
0.00%
0.00%
0.00%
0.00%
10.38%
1.12%
0.00%
0.00%
0.00%
0.00%
0.00%
0.00%
27.5
49.5
164.8
274.7
362.6
549.5
824.2
1098.9
Lentils
low color
50%
40%
30%
20%
10%
0%



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.

using regression for correlation check

```
##
## Call:
## lm(formula = mega_snack_2$even_count ~ mega_snack_2$n_color +
##     mega_snack_2$n_unit)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.51776 -0.15963  0.00218  0.15624  0.50835
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)            4.963e-01  2.470e-02  20.094   <2e-16 ***
## mega_snack_2$n_color -1.316e-03  4.205e-03  -0.313    0.754
## mega_snack_2$n_unit   2.191e-05  2.353e-05   0.931    0.352
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2438 on 837 degrees of freedom
## Multiple R-squared:  0.001152,   Adjusted R-squared:  -0.001235
```

```
## F-statistic: 0.4826 on 2 and 837 DF,  p-value: 0.6174
```

```
##
## Call:
## lm(formula = mega_snack_2$all_even ~ mega_snack_2$n_color + mega_snack_2$n_unit +
##     mega_snack_2$color_No2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.16455 -0.06633 -0.03162  0.01858  0.85571
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 1.552e-01  1.985e-02   7.823 1.56e-14 ***
## mega_snack_2$n_color       -2.946e-02  3.145e-03  -9.370  < 2e-16 ***
## mega_snack_2$n_unit        -1.891e-05  1.760e-05  -1.075    0.283
## mega_snack_2$color_No2TRUE  6.875e-02  1.271e-02   5.410 8.25e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1823 on 836 degrees of freedom
## Multiple R-squared:  0.1239, Adjusted R-squared:  0.1207
## F-statistic:  39.4 on 3 and 836 DF,  p-value: < 2.2e-16
```

Conclusions

Data Structure

The simulation created

Main Q: Eating M&M by Two

Although there is no clear pattern to the right M&M package for all the colors to have even count, maybe different approach can find a clear reason for more or less couples of M&M.

The general probability of all colors to be even is 4% for small 50g package 2.8% for big 1000g package, and overall 2.8%, which is less than I expected.

Summery