M&M Colors

Oliver de Pug

Import the Data

The data were collected using a Google Docs spreadsheet. The spreadsheet can be downloaded as an Excel file. This file can be imported into R, and then converted from frequency tables to individual observations.

```
  ### Import the first 150 lines of the Excel file
  counts <- read_xlsx("MandMsFall2019.xlsx", sheet="Sheet1", range = "A1:L150")
  ### Remove lines with missing identifiers
  counts <- counts[!is.na(counts$Type) & !is.na(counts$Initials) & !is.na(counts$Class),]
  ### Look at the frequency table
  head(counts)
```

```
## # A tibble: 6 x 12
##   Type    Red  Blue Green Orange Yellow Brown Total Initials Class Semester
##   <chr> <dbl> <dbl> <dbl>  <dbl>  <dbl> <dbl> <dbl> <chr>    <chr> <chr>
## 1 Plain     2     4     1      4      3     1    15 jlb      111-4 Fall
## 2 Pean~     1     1     1      3      0     0     6 jlb      111-5 Fall
## 3 Cara~     1     2     0      0      1     2     6 jlb      311-1 Fall
## 4 Cara~     0     2     0      2      2     0     6 TW       111-5 Fall
## 5 Plain     3     4     0      5      1     2    15 ED       111-5 Fall
## 6 Plain     1     7     3      2      0     2    15 cb       111-5 Fall
## # ... with 1 more variable: Year <dbl>
```

```
  ###

  ###### Reshape the data into a "long" format
  temp <- counts[,-c(8,11,12)] ### Remove "Total", "Semester", and "Year" variables
  ### melt seems to want a data.table.  We then melt using the given identifiers
  df.counts <- melt(data.table(temp), id = c("Type","Initials","Class"), na.rm = TRUE)
  ### Rename the column variables to make them more descriptive
  names(df.counts)
```

```
## [1] "Type"     "Initials" "Class"    "variable" "value"
```

```
  names(df.counts)[4:5] <- c("Color","Freq")
  names(df.counts)
```

```
## [1] "Type"     "Initials" "Class"    "Color"    "Freq"
```

```
  ### Look at the df.counts data.table
  head(df.counts)
```

```
##       Type Initials Class Color Freq
## 1:   Plain      jlb 111-4   Red    2
## 2:  Peanut      jlb 111-5   Red    1
## 3: Caramel      jlb 311-1   Red    1
## 4: Caramel       TW 111-5   Red    0
## 5:   Plain       ED 111-5   Red    3
## 6:   Plain       cb 111-5   Red    1
```

```
  ###### Now generate a data.frame with a single obs for each M&M
  countsToCases <- function(df, countcol = "Freq") {
    # Get the row indices to pull from df
```

```
    idx <- rep.int(seq_len(nrow(df)), df[[countcol]])

    # Drop count column
    df[[countcol]] <- NULL

    # Get the rows from df
    df[idx, ]
  }
  ### Run the function on the counts data from above
  df.cases <- countsToCases(df.counts)
  ### Look at df.cases
  head(df.cases)
```

```
##       Type Initials Class Color
## 1:   Plain      jlb 111-4   Red
## 2:   Plain      jlb 111-4   Red
## 3:  Peanut      jlb 111-5   Red
## 4: Caramel      jlb 311-1   Red
## 5:   Plain       ED 111-5   Red
## 6:   Plain       ED 111-5   Red
```

```
  ### Clean up
  rm(temp)

  ### Export the counts and cases
  write.csv(df.counts,"counts.csv")
  write.csv(df.cases,"cases.csv")
```

What's Up?

We can look at some tables to see if there relationships between the variables:

```
  ### Number of M&M's
  with(df.cases, table(Type, Color))
```

```
##          Color
## Type      Red Blue Green Orange Yellow Brown
##   caramel   2    1     0      1      1     1
##   Caramel   9    9     8      6     12    10
##   peanut    3    9     2      6      6     2
##   Peanut    6   21     9     17      9     6
##   plain     3   10     2     11      5     1
##   Plain    22   62    34     69     27    29
```

```
  ### Number of bags
  with(df.counts, table(Type, Color))
```

```
##          Color
## Type      Red Blue Green Orange Yellow Brown
##   caramel   1    1     1      1      1     1
##   Caramel   8    8     8      8      8     8
##   peanut    4    4     4      4      4     4
##   Peanut    9   10     9     10      9    10
##   plain     2    2     2      2      2     2
##   Plain    16   16    16     16     16    16
```

Oops. Some people didn't use the uppercase form of the Type of M&M. We can fix this.

```
  ### Replace lower case values
  df.counts[df.counts$Type == "caramel", "Type"] <- "Caramel"
  df.counts[df.counts$Type == "peanut", "Type"] <- "Peanut"
  df.counts[df.counts$Type == "plain", "Type"] <- "Plain"
  df.cases[df.cases$Type == "caramel", "Type"] <- "Caramel"
  df.cases[df.cases$Type == "peanut", "Type"] <- "Peanut"
  df.cases[df.cases$Type == "plain", "Type"] <- "Plain"

  ### Number of M&M's
  with(df.cases, table(Type, Color))
```

```
##          Color
## Type      Red Blue Green Orange Yellow Brown
##   Caramel  11   10     8      7     13    11
##   Peanut    9   30    11     23     15     8
##   Plain    25   72    36     80     32    30
```

```
  ### Number of bags
  with(df.counts, table(Type, Color))
```

```
##          Color
## Type      Red Blue Green Orange Yellow Brown
##   Caramel   9    9     9      9      9     9
##   Peanut   13   14    13     14     13    14
##   Plain    18   18    18     18     18    18
```

```
  ### Number of M&M's
  with(df.cases, table(Class, Color))
```

```
##        Color
## Class   Red Blue Green Orange Yellow Brown
##   111-4  18   44    18     43     21    15
##   111-5  16   52    29     50     20    21
##   311-1  11   16     8     17     19    13
```

```
  ### Number of bags
  with(df.counts, table(Class, Color))
```

```
##        Color
## Class   Red Blue Green Orange Yellow Brown
##   111-4  12   13    12     13     12    13
##   111-5  18   18    18     18     18    18
##   311-1  10   10    10     10     10    10
```

```
  ### Used missing (NA) instead of zero?
  counts[is.na(counts$Red),]
```

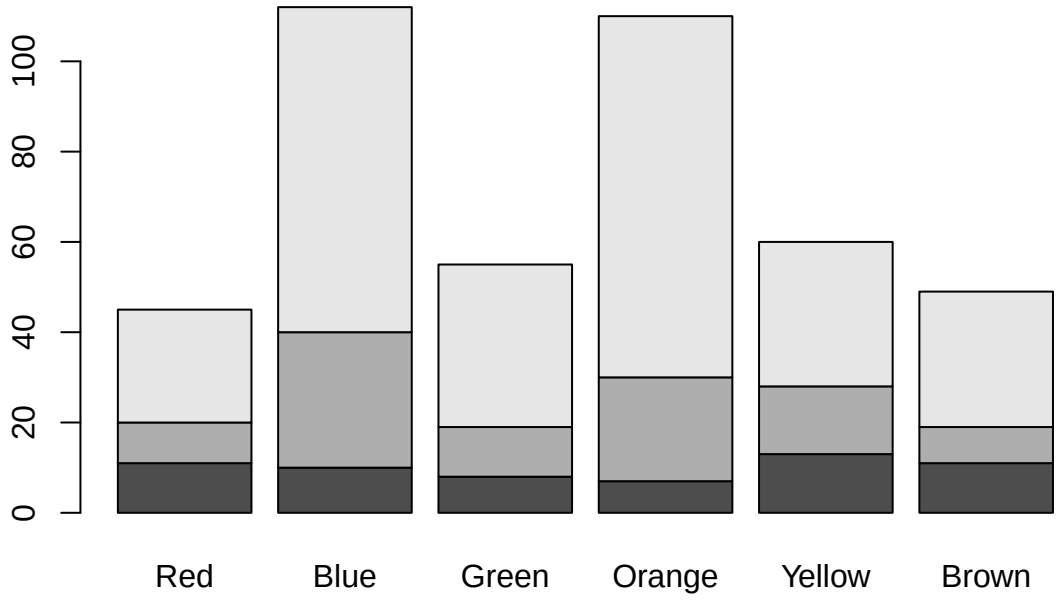
```
## # A tibble: 1 x 12
##   Type    Red  Blue Green Orange Yellow Brown Total Initials Class Semester
##   <chr> <dbl> <dbl> <dbl>  <dbl>  <dbl> <dbl> <dbl> <chr>    <chr> <chr>
## 1 Pean~    NA     5    NA      2     NA     1    NA KL       111-4 Fall
## # ... with 1 more variable: Year <dbl>
```

```
  ### Export the corrected counts and cases
  write.csv(df.counts,"corrcounts.csv")
  write.csv(df.cases,"corrcases.csv")
```

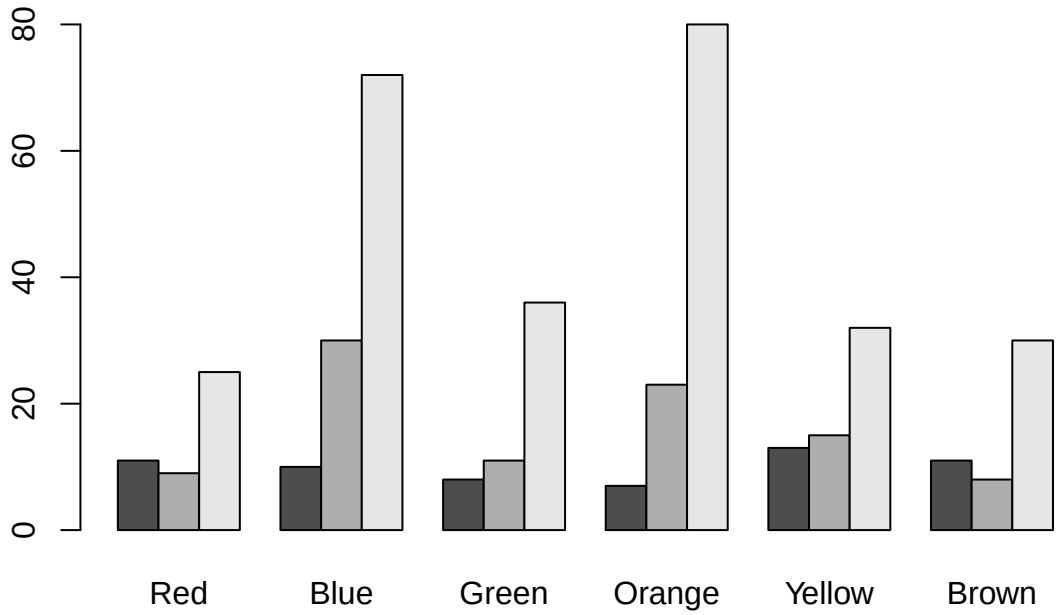
That's better. Although, it looks like someone (KL?) in the 111-4 class may have not entered 0's for Red, Green, and Yellow. R is smart and treats missing values (NA's) differently than it treats zeros (0's).

Now we can look at the way that the colors are distributed. Maybe a few plots will help. Which questions do the different plots address?

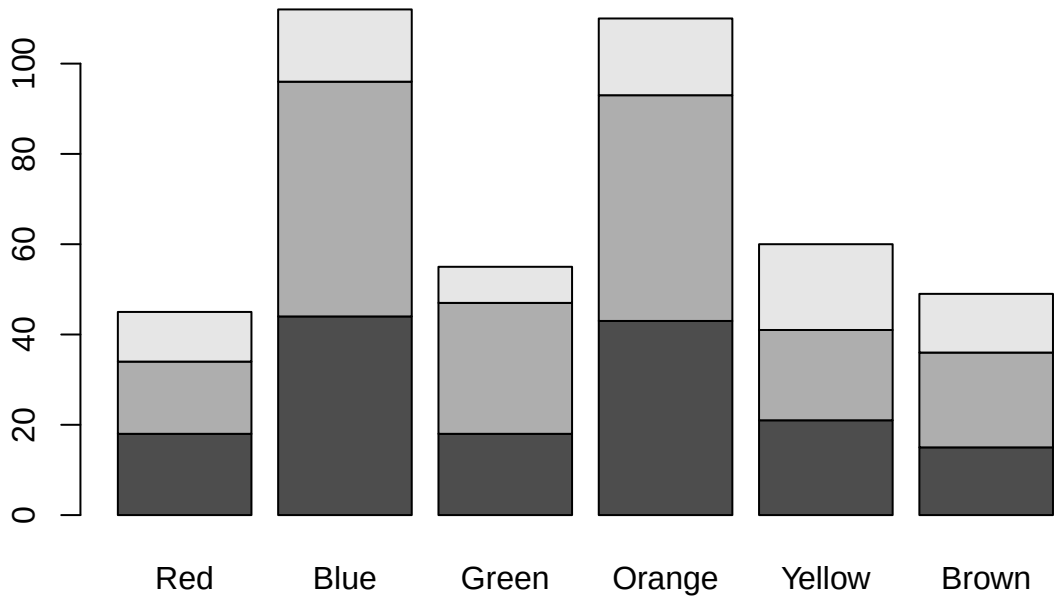
```
df.tc <- table(df.cases$Type, df.cases$Color)
barplot(df.tc,)
```



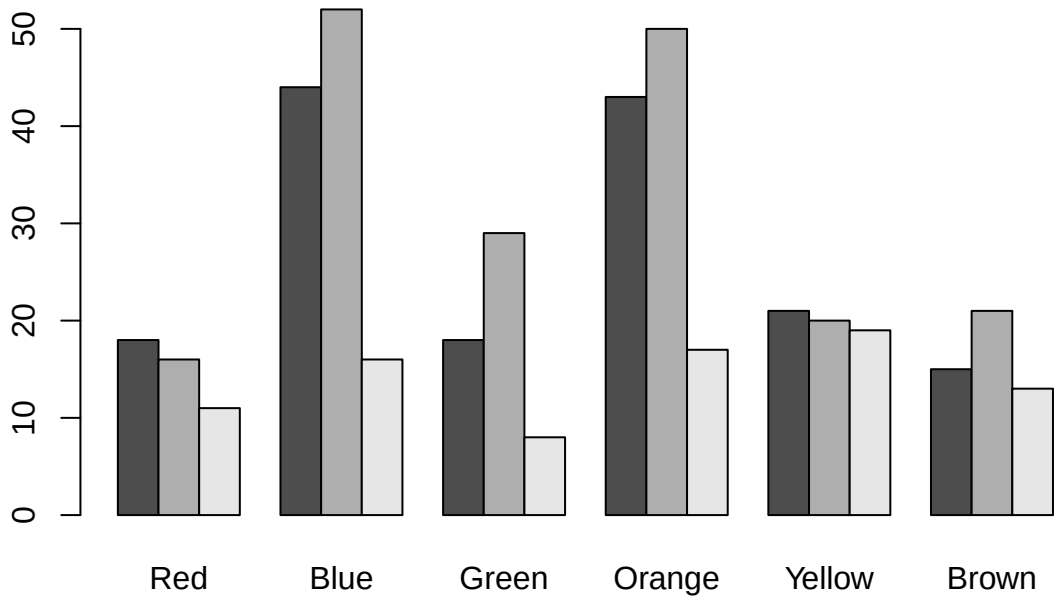

```
barplot(df.tc, beside = TRUE)
```



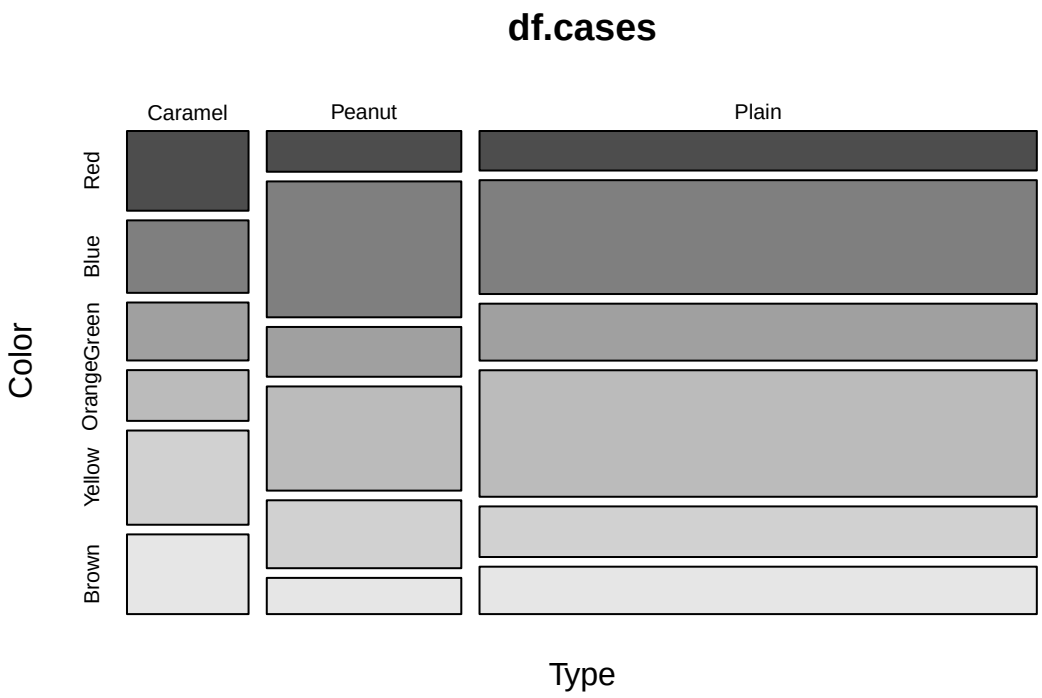

```
df.cc <- table(df.cases$Class, df.cases$Color)
barplot(df.cc)
```



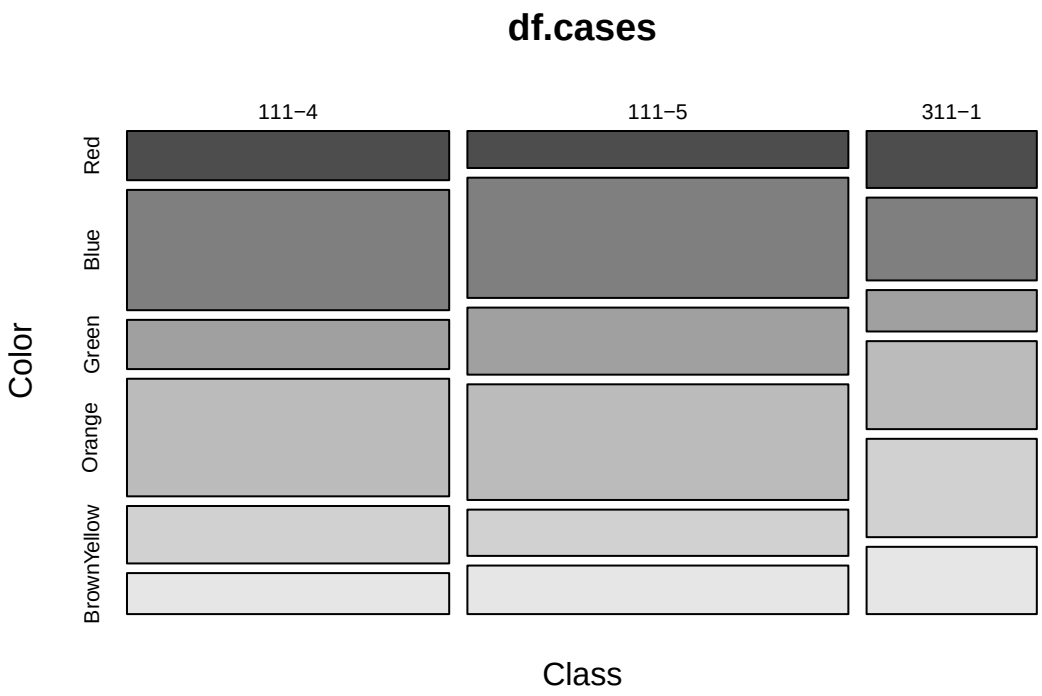

```
barplot(df.cc, beside = TRUE)
```



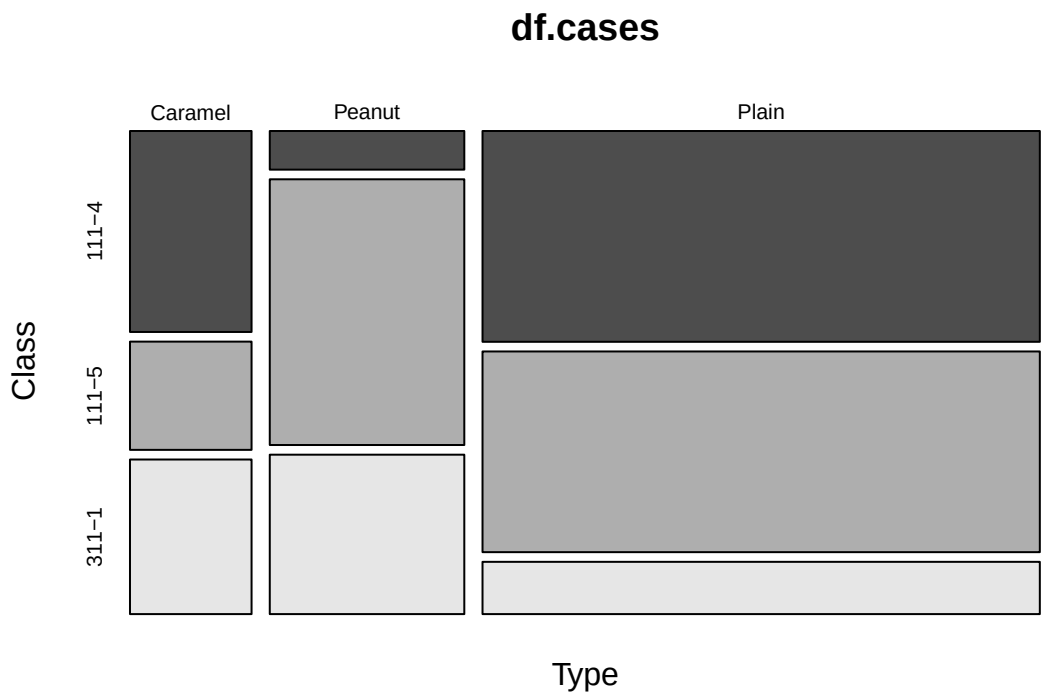

```
mosaicplot(~ Type + Color, data = df.cases, color = TRUE)
```



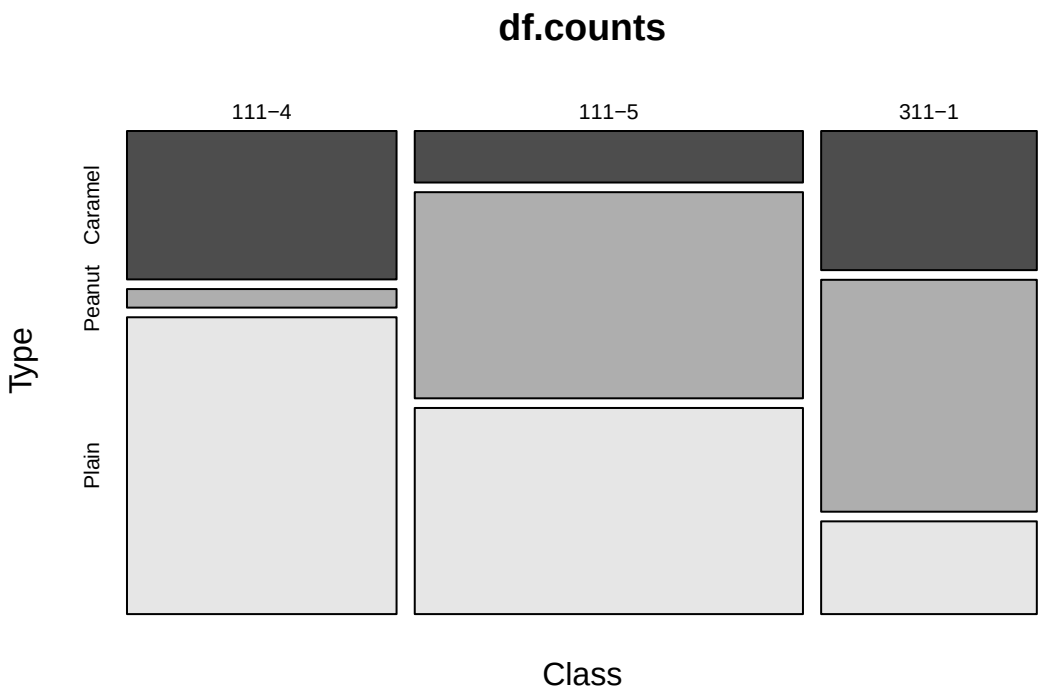

```
mosaicplot(~ Class + Color, data = df.cases, color = TRUE)
```



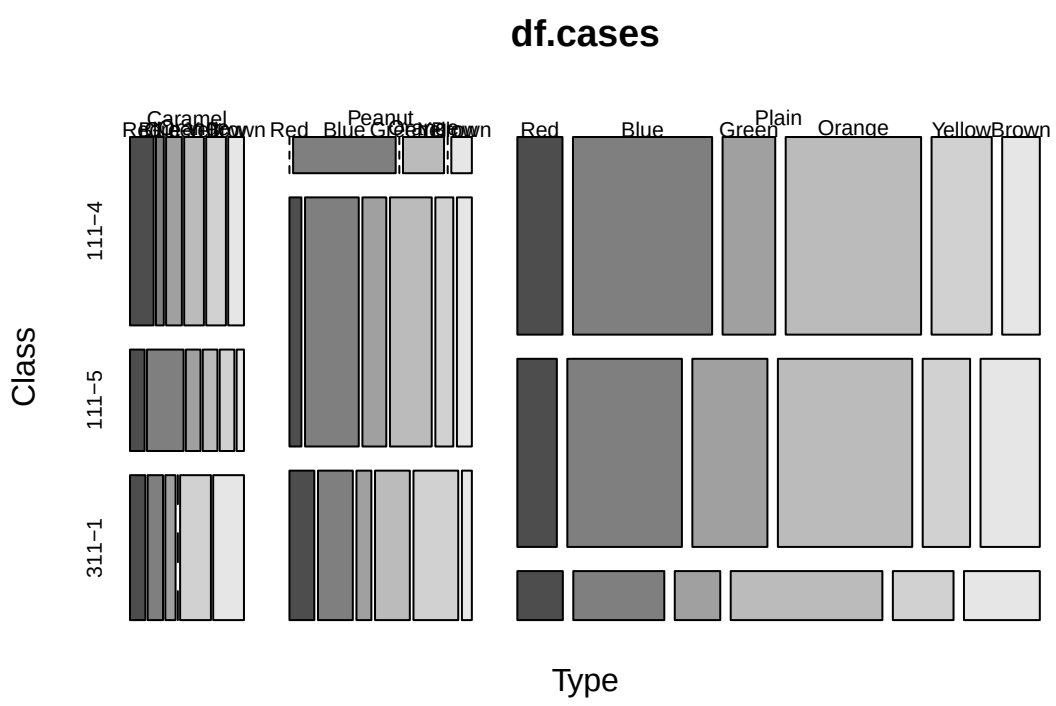

```
mosaicplot(~ Type + Class, data = df.cases, color = TRUE)
```




```
mosaicplot(~ Class + Type, data = df.counts, color = TRUE)
```




```
mosaicplot(~ Type + Class + Color, data = df.cases, color = TRUE)
```



We can model the effects of **Type** and **Class** upon the distribution of **Color**. Multinomial logistic regression is one method for fitting a model. The response in this case is the log-odds where the baseline for the odds is the "ref" level — in this case "Plain".

```
  df.cases$Type2 <- relevel(as.factor(df.cases$Type), ref = "Plain")
  df.cases$Class2 <- relevel(as.factor(df.cases$Class), ref = "311-1")
  df.cases.logistic <- multinom(Color ~  Type2 + Class2, data = df.cases)
```

```
## # weights:  36 (25 variable)
## initial  value 772.248331
## iter  10 value 725.863562
## iter  20 value 724.663754
## final  value 724.660942
## converged
```

```
  df.cases.logistic
```

```
## Call:
## multinom(formula = Color ~ Type2 + Class2, data = df.cases)
##
## Coefficients:
##        (Intercept) Type2Caramel Type2Peanut Class2111-4 Class2111-5
## Blue    0.49990225   -1.0357394   0.2656340   0.5768709  0.68230404
## Green  -0.09212262   -0.5316848  -0.1920117   0.2225996  0.79756893
## Orange  0.82914924   -1.5281958  -0.1759303   0.2977140  0.47468572
## Yellow  0.53696291   -0.1567771   0.1580659  -0.3494844 -0.34068911
## Brown   0.37757252   -0.1897356  -0.4775237  -0.4937109  0.03285833
##
```

```
## Residual Deviance: 1449.322
## AIC: 1499.322
```

```
  summary(df.cases.logistic)
```

```
## Call:
## multinom(formula = Color ~ Type2 + Class2, data = df.cases)
##
## Coefficients:
##        (Intercept) Type2Caramel Type2Peanut Class2111-4 Class2111-5
## Blue    0.49990225   -1.0357394   0.2656340   0.5768709  0.68230404
## Green  -0.09212262   -0.5316848  -0.1920117   0.2225996  0.79756893
## Orange  0.82914924   -1.5281958  -0.1759303   0.2977140  0.47468572
## Yellow  0.53696291   -0.1567771   0.1580659  -0.3494844 -0.34068911
## Brown   0.37757252   -0.1897356  -0.4775237  -0.4937109  0.03285833
##
## Std. Errors:
##        (Intercept) Type2Caramel Type2Peanut Class2111-4 Class2111-5
## Blue     0.4718325    0.5036408   0.4772183   0.5162091   0.4993299
## Green    0.5436741    0.5447558   0.5500142   0.6007979   0.5725391
## Orange   0.4641847    0.5431572   0.4853996   0.5131378   0.4972598
## Yellow   0.4764822    0.5012458   0.5323822   0.5287140   0.5194745
## Brown    0.5009398    0.5200331   0.5847121   0.5666570   0.5431069
##
## Residual Deviance: 1449.322
## AIC: 1499.322
```

```
  z <- summary(df.cases.logistic)$coefficients/summary(df.cases.logistic)$standard.errors
  z
```

```
##        (Intercept) Type2Caramel Type2Peanut Class2111-4 Class2111-5
## Blue     1.0594908   -2.0565044   0.5566300   1.1175140  1.36643931
## Green   -0.1694446   -0.9760057  -0.3491031   0.3705066  1.39303828
## Orange   1.7862486   -2.8135421  -0.3624442   0.5801833  0.95460310
## Yellow   1.1269317   -0.3127748   0.2969030  -0.6610084 -0.65583411
## Brown    0.7537283   -0.3648529  -0.8166817  -0.8712694  0.06050066
```

```
  p <- (1 - pnorm(abs(z), 0, 1)) * 2
  p
```

```
##        (Intercept) Type2Caramel Type2Peanut Class2111-4 Class2111-5
## Blue    0.28937630  0.039733927   0.5777803   0.2637746   0.1718011
## Green   0.86544696  0.329061643   0.7270119   0.7110050   0.1636082
## Orange  0.07405901  0.004899897   0.7170201   0.5617910   0.3397784
## Yellow  0.25977135  0.754451736   0.7665406   0.5086069   0.5119309
## Brown   0.45101241  0.715221214   0.4141104   0.3836071   0.9517569
```

```
  df.cases.logistic <- multinom(Color ~  Type2, data = df.cases)
```

```
## # weights:  24 (15 variable)
## initial  value 772.248331
## iter  10 value 731.832956
## iter  20 value 730.609028
## final  value 730.608778
## converged
```

```
  df.cases.logistic
```

```
## Call:
## multinom(formula = Color ~ Type2, data = df.cases)
##
## Coefficients:
##        (Intercept) Type2Caramel Type2Peanut
## Blue     1.0577896  -1.15308250   0.1461676
## Green    0.3646376  -0.68304163  -0.1639976
## Orange   1.1631553  -1.61512201  -0.2249155
## Yellow   0.2468538  -0.07980641   0.2639450
## Brown    0.1823215  -0.18227624  -0.3001975
##
## Residual Deviance: 1461.218
## AIC: 1491.218
```

```
  summary(df.cases.logistic)
```

```
## Call:
## multinom(formula = Color ~ Type2, data = df.cases)
##
## Coefficients:
##        (Intercept) Type2Caramel Type2Peanut
## Blue     1.0577896  -1.15308250   0.1461676
## Green    0.3646376  -0.68304163  -0.1639976
## Orange   1.1631553  -1.61512201  -0.2249155
## Yellow   0.2468538  -0.07980641   0.2639450
## Brown    0.1823215  -0.18227624  -0.3001975
##
## Std. Errors:
##        (Intercept) Type2Caramel Type2Peanut
## Blue     0.2321398    0.4947724   0.4453424
## Green    0.2603419    0.5326207   0.5194183
## Orange   0.2291286    0.5350401   0.4550674
## Yellow   0.2669273    0.4889642   0.4990239
## Brown    0.2708012    0.5051245   0.5562819
##
## Residual Deviance: 1461.218
## AIC: 1491.218
```

```
  z <- summary(df.cases.logistic)$coefficients/summary(df.cases.logistic)$standard.errors
  z
```

```
##        (Intercept) Type2Caramel Type2Peanut
## Blue     4.5566926   -2.3305310   0.3282140
## Green    1.4006105   -1.2824167  -0.3157332
## Orange   5.0764294   -3.0186936  -0.4942465
## Yellow   0.9247981   -0.1632152   0.5289226
## Brown    0.6732667   -0.3608541  -0.5396499
```

```
  p <- (1 - pnorm(abs(z), 0, 1)) * 2
  p
```

```
##         (Intercept) Type2Caramel Type2Peanut
## Blue   5.196540e-06  0.019778106   0.7427499
## Green  1.613306e-01  0.199696519   0.7522050
```

```
## Orange 3.845937e-07  0.002538671   0.6211321
## Yellow 3.550709e-01  0.870348960   0.5968591
## Brown  5.007776e-01  0.718208514   0.5894385
```

```
  ### Odds estimates with "Plain" as baseline
  exp(coef(df.cases.logistic))[,-1]  ### Remove intercepts as they are not interesting
```

```
##        Type2Caramel Type2Peanut
## Blue      0.3156622   1.1573901
## Green     0.5050784   0.8487441
## Orange    0.1988664   0.7985837
## Yellow    0.9232951   1.3020566
## Brown     0.8333711   0.7406719
```

```
  ### Look for CIs that do not include 1:1 odds
  exp(confint(df.cases.logistic))[-1,,] ### Remove intercepts as they are not interesting
```

```
## , , Blue
##
##                  2.5 %    97.5 %
## Type2Caramel 0.1196938 0.8324798
## Type2Peanut  0.4835080 2.7704859
##
## , , Green
##
##                  2.5 %   97.5 %
## Type2Caramel 0.1778244 1.434585
## Type2Peanut  0.3066529 2.349126
##
## , , Orange
##
##                   2.5 %    97.5 %
## Type2Caramel 0.06968423 0.5675294
## Type2Peanut  0.32731534 1.9483839
##
## , , Yellow
##
##                  2.5 %   97.5 %
## Type2Caramel 0.3541061 2.407396
## Type2Peanut  0.4896209 3.462580
##
## , , Brown
##
##                  2.5 %   97.5 %
## Type2Caramel 0.3096533 2.242855
## Type2Peanut  0.2489533 2.203606
```

```
    ### Compute reciprocals so that Caramel and Peanut are now baseline
  exp(-coef(df.cases.logistic))[,-1] ### Remove intercepts as they are not interesting
```

```
##        Type2Caramel Type2Peanut
## Blue       3.167943   0.8640129
## Green      1.979891   1.1782115
## Orange     5.028501   1.2522169
## Yellow     1.083077   0.7680158
## Brown      1.199946   1.3501254
```

```
  exp(-confint(df.cases.logistic))[-1,,] ### Remove intercepts as they are not interesting
```

```
## , , Blue
##
##                  2.5 %    97.5 %
## Type2Caramel 8.354653 1.2012304
## Type2Peanut  2.068218 0.3609475
##
## , , Green
##
##                  2.5 %    97.5 %
## Type2Caramel 5.623527 0.6970656
## Type2Peanut  3.261016 0.4256902
##
## , , Orange
##
##                   2.5 %    97.5 %
## Type2Caramel 14.350449 1.7620233
## Type2Peanut   3.055158 0.5132459
##
## , , Yellow
##
##                  2.5 %    97.5 %
## Type2Caramel 2.824012 0.4153865
## Type2Peanut  2.042396 0.2888020
##
## , , Brown
##
##                  2.5 %    97.5 %
## Type2Caramel 3.229418 0.4458603
## Type2Peanut  4.016818 0.4538017
```

```
  ### Compute fitted values for observed data
  head(pp <- fitted(df.cases.logistic))
```

```
##          Red      Blue     Green    Orange    Yellow      Brown
## 1 0.09090912 0.2618181 0.1309084 0.2909105 0.1163629 0.10909094
## 2 0.09090912 0.2618181 0.1309084 0.2909105 0.1163629 0.10909094
## 3 0.09375257 0.3125037 0.1145830 0.2395828 0.1562501 0.08332787
## 4 0.18332994 0.1666665 0.1333375 0.1166667 0.2166612 0.18333823
## 5 0.09090912 0.2618181 0.1309084 0.2909105 0.1163629 0.10909094
## 6 0.09090912 0.2618181 0.1309084 0.2909105 0.1163629 0.10909094
```

```
  types <- data.frame(Type2 = c("Plain", "Peanut","Caramel"))
  cbind(types,predict(df.cases.logistic, newdata = types, "probs"))
```

```
##     Type2        Red      Blue     Green    Orange    Yellow      Brown
## 1   Plain 0.09090912 0.2618181 0.1309084 0.2909105 0.1163629 0.10909094
## 2  Peanut 0.09375257 0.3125037 0.1145830 0.2395828 0.1562501 0.08332787
## 3 Caramel 0.18332994 0.1666665 0.1333375 0.1166667 0.2166612 0.18333823
```

Since **Class** did not appear to be important, it was removed. It looks like the color distribution in "Peanut" is not different from "Plain." However, "Blue" and "Orange" seem to appear at lower rates within "Caramel" than they do in "Plain". Some might find this interesting. By the way, the lower proportion of blue and orange in caramel can be seen in the mosaic plot above.

For more on multinomial logistic regression, you can take a look at https://stats.idre.ucla.edu/r/dae/multinomial-logistic-regression/ . This site has a pretty good quick introduction to the topic.

By the Bag

We might be interested in if the **Type** of M&M's in a bag affects the number of M&M's that one gets. We now ignore **Color** and deal with the **Total** number of M&M's. This data is in the original **counts** data.frame.

```
  with(counts, table(Total,Type))
```

```
##      Type
## Total caramel Caramel peanut Peanut plain Plain
##    0        0       0      0      0     0     1
##    6        1       4      0      4     0     0
##    7        0       2      3      2     0     0
##    8        0       2      1      2     0     0
##    13       0       0      0      0     0     1
##    14       0       0      0      0     0     2
##    15       0       0      0      0     1     6
##    16       0       0      0      0     0     5
##    17       0       0      0      0     1     1
```

As before, we need to clean up. The empty bag is interesting.

```
  bags <- counts
  bags[bags$Total == 0,]
```

```
## # A tibble: 3 x 12
##   Type    Red  Blue Green Orange Yellow Brown Total Initials Class Semester
##   <chr> <dbl> <dbl> <dbl>  <dbl>  <dbl> <dbl> <dbl> <chr>    <chr> <chr>
## 1 <NA>     NA    NA    NA     NA     NA    NA    NA <NA>     <NA>  <NA>
## 2 Plain     0     6     1      7      0     1     0 ET       111-4 Fall
## 3 <NA>     NA    NA    NA     NA     NA    NA    NA <NA>     <NA>  <NA>
## # ... with 1 more variable: Year <dbl>
```

```
  bags$Total = bags$Red + bags$Blue + bags$Green + bags$Orange + bags$Yellow + bags$Brown
  head(bags)
```

```
## # A tibble: 6 x 12
##   Type    Red  Blue Green Orange Yellow Brown Total Initials Class Semester
##   <chr> <dbl> <dbl> <dbl>  <dbl>  <dbl> <dbl> <dbl> <chr>    <chr> <chr>
## 1 Plain     2     4     1      4      3     1    15 jlb      111-4 Fall
## 2 Pean~     1     1     1      3      0     0     6 jlb      111-5 Fall
## 3 Cara~     1     2     0      0      1     2     6 jlb      311-1 Fall
## 4 Cara~     0     2     0      2      2     0     6 TW       111-5 Fall
## 5 Plain     3     4     0      5      1     2    15 ED       111-5 Fall
## 6 Plain     1     7     3      2      0     2    15 cb       111-5 Fall
## # ... with 1 more variable: Year <dbl>
```

```
  bags[bags$Total == 0,]
```

```
## # A tibble: 1 x 12
##   Type    Red  Blue Green Orange Yellow Brown Total Initials Class Semester
##   <chr> <dbl> <dbl> <dbl>  <dbl>  <dbl> <dbl> <dbl> <chr>    <chr> <chr>
## 1 <NA>     NA    NA    NA     NA     NA    NA    NA <NA>     <NA>  <NA>
## # ... with 1 more variable: Year <dbl>
```

```
  bags[is.na(bags$Total),]
```

```
## # A tibble: 1 x 12
##   Type    Red  Blue Green Orange Yellow Brown Total Initials Class Semester
##   <chr> <dbl> <dbl> <dbl>  <dbl>  <dbl> <dbl> <dbl> <chr>    <chr> <chr>
## 1 Pean~    NA     5    NA      2     NA     1    NA KL       111-4 Fall
## # ... with 1 more variable: Year <dbl>
```

```
  bags <- bags[!is.na(bags$Total),]
  bags[is.na(bags$Total),]
```

```
## # A tibble: 0 x 12
## # ... with 12 variables: Type <chr>, Red <dbl>, Blue <dbl>, Green <dbl>,
## #   Orange <dbl>, Yellow <dbl>, Brown <dbl>, Total <dbl>, Initials <chr>,
## #   Class <chr>, Semester <chr>, Year <dbl>
```

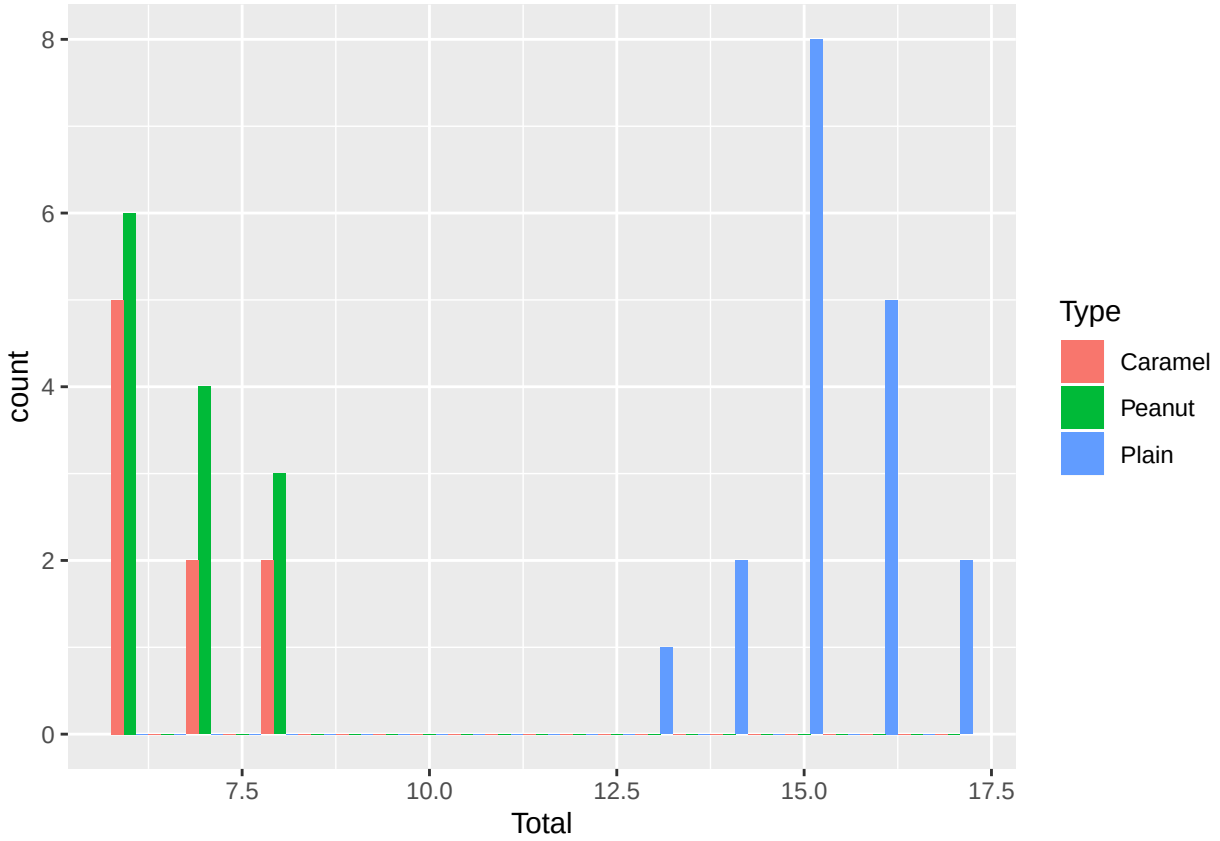
```
  ### Replace lower case values
  bags[bags$Type == "caramel", "Type"] <- "Caramel"
  bags[bags$Type == "peanut", "Type"] <- "Peanut"
  bags[bags$Type == "plain", "Type"] <- "Plain"

  ### Number of M&M's
  with(bags, table(Total, Type))
```

```
##      Type
## Total Caramel Peanut Plain
##    6        5      6     0
##    7        2      4     0
##    8        2      3     0
##    13       0      0     1
##    14       0      0     2
##    15       0      0     8
##    16       0      0     5
##    17       0      0     2
```

The data now appear to be clean. A quick plot might help us figure out what is going on.

```
  ggplot(bags, aes(Total, fill = Type)) +
    geom_histogram(binwidth=.5, position="dodge")
```



It looks like we get more "Plain" M&M's per bag.

We can now model the number of M&M's per bag as a function of **Type**. Because this is counts data, we use Poisson regression.

```
bags$Type2 <- relevel(as.factor(bags$Type), ref = "Plain")
bags$Class2 <- relevel(as.factor(bags$Class), ref = "311-1")
summary(bags.pois <- glm(Total ~ Type2 + Class2, family="poisson", data=bags))
```

```
##
## Call:
## glm(formula = Total ~ Type2 + Class2, family = "poisson", data = bags)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.65242  -0.28497  -0.02409   0.12727   0.53457
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.717393   0.129633  20.962  < 2e-16 ***
## Type2Caramel -0.829967   0.145546  -5.702 1.18e-08 ***
## Type2Peanut  -0.803081   0.133694  -6.007 1.89e-09 ***
## Class2111-4   0.023209   0.147321   0.158    0.875
## Class2111-5  -0.003129   0.134945  -0.023    0.981
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 71.7198  on 39  degrees of freedom
## Residual deviance:  3.1961  on 35  degrees of freedom
## AIC: 178.28
##
## Number of Fisher Scoring iterations: 4
```

```
summary(bags.pois <- glm(Total ~ Type2, family="poisson", data=bags))
```

```
##
## Call:
## glm(formula = Total ~ Type2, family = "poisson", data = bags)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.59821  -0.26269  -0.07128   0.18335   0.50048
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    2.7264     0.0603  45.212  < 2e-16 ***
## Type2Caramel  -0.8293     0.1425  -5.820 5.89e-09 ***
## Type2Peanut   -0.8140     0.1225  -6.646 3.00e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 71.7198  on 39  degrees of freedom
## Residual deviance:  3.2504  on 37  degrees of freedom
## AIC: 174.33
##
## Number of Fisher Scoring iterations: 4
```

Cameron and Trivedi (2009) recommend using robust standard errors for the parameter estimates to control for any mild violations of the distribution assumption that the variance equals the mean. We use R package ***sandwich*** below to obtain the robust standard errors and calculate the p-values accordingly. Together with the p-values, we also calculate 95% confidence intervals using the parameter estimates and their robust standard errors.

```
bags.cov <- vcovHC(bags.pois, type="HC0")
std.err <- sqrt(diag(bags.cov))
r.est <- cbind(Estimate= coef(bags.pois), "Robust SE" = std.err,
          "Pr(>|z|)" = 2 * pnorm(abs(coef(bags.pois)/std.err),
                                 lower.tail=FALSE),
LL = coef(bags.pois) - 1.96 * std.err,
UL = coef(bags.pois) + 1.96 * std.err)

r.est
```

```
##                Estimate  Robust SE      Pr(>|z|)         LL         UL
## (Intercept)   2.7263993 0.01526022  0.000000e+00  2.6964893  2.7563094
## Type2Caramel -0.8292794 0.04358372  1.013950e-80 -0.9147035 -0.7438553
## Type2Peanut  -0.8140119 0.03613399 2.227183e-112 -0.8848345 -0.7431893
```

Clearly, both "Caramel" and "Peanut" bags contain fewer M&M's than do "Plain" bags. This supports what we obeserved in the plot above.

For more information on Poisson regression look at the site https://stats.idre.ucla.edu/r/dae/poisson-regression/ . The example that they use includes more explanatory variables and looks at how to interpret more interesting models.