BIO5702: Student homework for the #CandyPhenotyping module

Dan Chitwood

April 17, 2016

The following is unaltered homework exploring the multivariate nutrition, shape, and color information from 75 different candy types and 980 individual candy pieces. It was written by the students of Bio5702, a graduate course at Washington University in St. Louis, "Current Approaches in Plant Research".

So I have the tab delimited file of all the color and shape data called `Color_Data_all`

That file has the color information for each region of interest split into 3 rows; e.g. image 1's ROI 1 will have a Red row, Green row, and Blue row.

What I need to do is create a new dataframe that instead of having each ROI with those 3 rows, we want only one row for each ROI, and 3 columns each with the pertinent Red, Green, and Blue intensity.

- E.g. instead of:

```
ID     color   value
id01   red     00
id01   green   00
id01   blue    00
```

- We want this:

```
id   red   green   blue
01   00     00       00
```

R Reshape example:

```
library(reshape2)
```

1. First, melt the data, `newdata <- melt(data, id=c("RATING", "TIME_TO_MATURITY", "INDUSTRY_CODE", "BOND_TYPE"))`;
2. Second, cast the data based on your needs, for instance, to get the total amount of each industry, `cast(newdata, INDUSTRY_CODE ~ variable, sum)` returns you a `data.frame`

Read in my data properly

```
allcolordata <- read.table("./Color_Data_all_with_color.txt", header=TRUE)
```

Okay, so to apply that to our information, first I want to create a new `data.frame` with only the candy id and ROI id, the color column, and the color's intensity

```
subset_colors<- allcolordata[c(1:6, 11:15)]
```

Now I want to "unstack" my "long format" data into "wide format" meaning the 3 rows per ROI need to be collapsed into a single row, but with 3 columns for the mean values of each color.

Example of long format:

```
Subject Gender   Test Resul
1       1      M   Read     10
2       2      F  Write      4
3       1      M  Write      8
4       2      F Listen      6
5       2      F   Read      7
6       1      M Listen      7
```

Example of command to turn into wide format:

```
wide_format <- unstack(observations_long, Result ~ Test)
wide_format
Listen Read Write
1      6   10     4
2      7    7     8
wide_color <- unstack(mastercolor, Mean ~ Color)
```

Now look at my new table

```
wide_color
```

That didn't work at all.

Let's try another approach to do the same thing, convert long format into wide format, using `reshape()`

First download the "stats" command library

```
library(stats)
```

now create a "wide format" dataframe from my smaller dataframe that

```
color_to_wide <- reshape(subset_colors, timevar = "Color", idvar = c("Label", "ID"), direction = "wide"
```

Open that text file in Excel, open as space delimited, and fix the fact that the very top row is shifted left by one cell. Make sure that the columnname ID is over the actual candy IDs, not the number values added by R that span from 1 to 3518

Okay, the file we just made has color words added to a bunch of the column names that don't need to be there, so we'll manually delete those.

Also, manually delete all the duplicate columns, e.g. Solidity.Red, Solidity.Green, etc. (eveything except Mean.Red, Mean.Green, Mean.Blue)

Open that file that contains both the color and shape data

```
shape_color_wide <- read.table("./widedata.txt", header=T)
```

Open the nutrition information that has the ID's all capitalized

```
nutrition_facts <- read.table("./candy_nutrition_415.txt", header=T)
```

Now merge those two data.frames

```
merged_color_shape_nutrition <- merge(x=shape_color_wide, y=nutrition_facts, by.x = "ID", by.y = "ID", a
```

Now load all the libraries needed for the principle component analysis and graph making

```
library(ggplot2)
library(ggrepel)
library(ggdendro)
library(ape)
```

Now normalize all the nutrition information to the grams per serving size

```
merged_color_shape_nutrition$total_fat_per_serv <- merged_color_shape_nutrition$total_fat_g/merged_colo
merged_color_shape_nutrition$saturated_fat_per_serv <- merged_color_shape_nutrition$saturated_fat_g/merg
merged_color_shape_nutrition$cholesterol_per_serv <- merged_color_shape_nutrition$cholesterol_mg/merged_
merged_color_shape_nutrition$sodium_per_serv <- merged_color_shape_nutrition$sodium_mg/merged_color_shap
merged_color_shape_nutrition$total_carb_per_serv <- merged_color_shape_nutrition$total_carb_g/merged_col
merged_color_shape_nutrition$dietary_fiber_per_serv <- merged_color_shape_nutrition$dietary_fiber_g/merg
merged_color_shape_nutrition$sugars_per_serv <- merged_color_shape_nutrition$sugars_g/merged_color_shape
merged_color_shape_nutrition$protein_per_serv <- merged_color_shape_nutrition$protein_g/merged_color_sha
```

First we'll do a principle component analysis and make plots based on ALL the information in our dataframe.

Now we're going to center and scale our data, only the quantitative columns, excluding slice because that value is always 1, and excluding the non-normalized nutrition information.

```
pca <- prcomp(merged_color_shape_nutrition[c(4:5, 7:12, 28:35)], center = TRUE, scale. = TRUE)
```

Look at the PC analysis summary

```
summary(pca)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6
## Standard deviation     2.2973 1.7865 1.4712 1.00950 0.92506 0.88108
## Proportion of Variance 0.3299 0.1995 0.1353 0.06369 0.05348 0.04852
## Cumulative Proportion  0.3299 0.5293 0.6646 0.72830 0.78178 0.83030
##                           PC7    PC8     PC9    PC10    PC11    PC12
## Standard deviation     0.76773 0.6986 0.65256 0.52835 0.50151 0.47779
## Proportion of Variance 0.03684 0.0305 0.02661 0.01745 0.01572 0.01427
## Cumulative Proportion  0.86714 0.8976 0.92425 0.94170 0.95742 0.97169
##                           PC13    PC14    PC15    PC16
## Standard deviation     0.42589 0.40975 0.27009 0.17548
## Proportion of Variance 0.01134 0.01049 0.00456 0.00192
## Cumulative Proportion  0.98302 0.99352 0.99808 1.00000
```

Look at it. The PC# column that has the largest value in the Proportion of Variance row is the PC that we want to use.

```
pca$rotation
```

```
##                                PC1         PC2          PC3         PC4
## cm2                     0.171721082 -0.32025207  0.008855066 -0.11012480
## Circ                   -0.003357819  0.48173318  0.031578251 -0.21656721
## AR                     -0.065668295 -0.42592154 -0.205927101  0.06121716
## Round                   0.085770081  0.47505573  0.161692940  0.04831627
## Solidity                0.059194321  0.39363103 -0.093175316 -0.29854478
## Mean.Green              0.081879539 -0.12930591  0.552199711 -0.20261162
## Mean.Blue               0.102984580 -0.16183742  0.411539488 -0.15134460
## Mean.Red                0.025560486 -0.15412701  0.475201921 -0.25043020
## total_fat_per_serv      0.417989129  0.03003611  0.031514837  0.10150232
## saturated_fat_per_serv  0.384833931  0.08088974  0.039039886  0.14405956
## cholesterol_per_serv    0.287382221 -0.01865257  0.111525848  0.46172353
## sodium_per_serv         0.194053643 -0.09045451 -0.188693780 -0.63314001
## total_carb_per_serv    -0.406909843  0.05029688  0.137018274  0.01451941
## dietary_fiber_per_serv  0.372031064  0.07700141  0.029554105  0.12670948
## sugars_per_serv        -0.232495620  0.10852171  0.387769026  0.23389261
## protein_per_serv        0.369147672 -0.02148365 -0.031829567 -0.05482351
##                                PC5           PC6         PC7         PC8
## cm2                     0.141623758 -0.6890913275  0.10290641  0.36134120
## Circ                   -0.070679366 -0.0232424224 -0.19463254 -0.21931496
## AR                     -0.009779902  0.0005725076 -0.20311728 -0.52622468
## Round                  -0.024970069  0.0890290958  0.29668002  0.06804219
## Solidity                0.164602750 -0.4207613169 -0.50690434 -0.08737201
## Mean.Green              0.067687693  0.2024223023 -0.17347123 -0.08175505
## Mean.Blue              -0.584732725  0.0811260993 -0.31543621  0.31345270
## Mean.Red                0.510184125 -0.0021999509  0.11212128 -0.32689503
## total_fat_per_serv     -0.064590092 -0.0444010606  0.14776663 -0.11376305
## saturated_fat_per_serv -0.087069254  0.0341117177  0.25798579 -0.24070658
## cholesterol_per_serv   -0.210902703 -0.2168673568 -0.33349941 -0.29666175
## sodium_per_serv        -0.404189645 -0.0946553082  0.34577730 -0.25852944
## total_carb_per_serv    -0.073054910 -0.1271603323  0.08365219 -0.04304734
## dietary_fiber_per_serv  0.104698316 -0.0754952314  0.06098377  0.03669027
## sugars_per_serv        -0.133767501 -0.4073374085  0.28823403 -0.10620361
## protein_per_serv        0.294206461  0.2125231298 -0.10082811  0.28311624
##                                PC9        PC10         PC11        PC12
## cm2                    -0.0183480177  0.15807228 -0.344048722  0.01196855
## Circ                    0.2737963765 -0.08131065 -0.530439279  0.40421361
## AR                      0.4847493902  0.15312452 -0.118176866 -0.14566243
## Round                  -0.0461545785  0.15148064 -0.078618425 -0.23208307
## Solidity               -0.0005836826  0.07459780  0.278074326 -0.37281455
## Mean.Green             -0.1050089175  0.71379496  0.041202452  0.12930726
## Mean.Blue               0.2271184043 -0.32380734 -0.004900792 -0.24055245
## Mean.Red               -0.1151934647 -0.52040091  0.001577512 -0.09203812
## total_fat_per_serv      0.0620541938  0.04609866 -0.079461241 -0.14476442
## saturated_fat_per_serv  0.0682081839  0.06875432 -0.200296766 -0.44730380
## cholesterol_per_serv   -0.4929003732 -0.12240849 -0.010601947  0.29058553
## sodium_per_serv        -0.1686637461 -0.01422877  0.173724749  0.23357517
## total_carb_per_serv     0.0218769339  0.04144857  0.097909607  0.04472440
## dietary_fiber_per_serv  0.4177584103 -0.01540826  0.619474584  0.28319266
```

```
## sugars_per_serv         0.3304394196  0.00115083  0.051054718  0.15240586
## protein_per_serv        0.2182084890 -0.08530292 -0.168526947  0.27180843
##                               PC13         PC14         PC15         PC16
## cm2                     -0.15545944 -0.202864533  0.074664630 -0.070360012
## Circ                     0.07532884 -0.308203290  0.016806813 -0.035073375
## AR                      -0.38615804  0.081600848  0.001184060  0.021666870
## Round                   -0.74022915  0.055779384 -0.032148061 -0.009162178
## Solidity                 0.07099997  0.217206488 -0.001520532  0.030279202
## Mean.Green               0.08584855 -0.027091158 -0.021378434 -0.027533447
## Mean.Blue               -0.08872384 -0.040655582  0.028081227 -0.007047344
## Mean.Red                -0.08766092 -0.083073313 -0.009774137  0.009688475
## total_fat_per_serv       0.17594260 -0.088967519 -0.152309769  0.822709397
## saturated_fat_per_serv   0.36408148  0.007802981  0.363224984 -0.419182348
## cholesterol_per_serv    -0.18467259  0.101416166  0.105523262 -0.051808571
## sodium_per_serv         -0.07122248  0.196540903  0.029978852 -0.031208578
## total_carb_per_serv     -0.02463391 -0.009891422  0.811076532  0.331992913
## dietary_fiber_per_serv -0.09279399 -0.373809802  0.113200395 -0.130932423
## sugars_per_serv          0.17715820  0.442665707 -0.290293124 -0.065490202
## protein_per_serv        -0.04293091  0.636353539  0.263951270  0.059776414
```

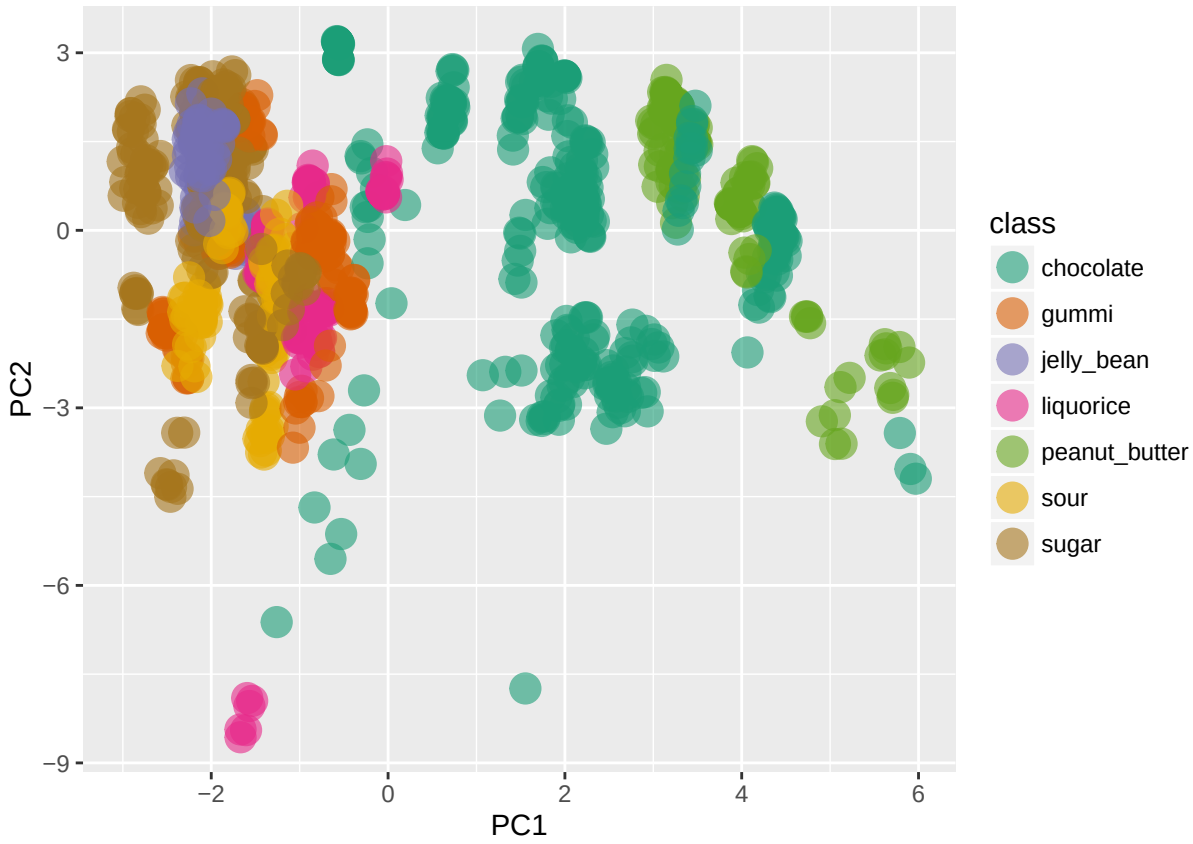
This is neat to look at but we don't really need it, because the PC1 and PC2 are going to be our x and y axes, at least for our first graphs.

Now we want to get the PC scores, look at them, and bind those PC scores with the original data

```
scores <- as.data.frame(pca$x)
pca_scores <- cbind(merged_color_shape_nutrition, scores)
```

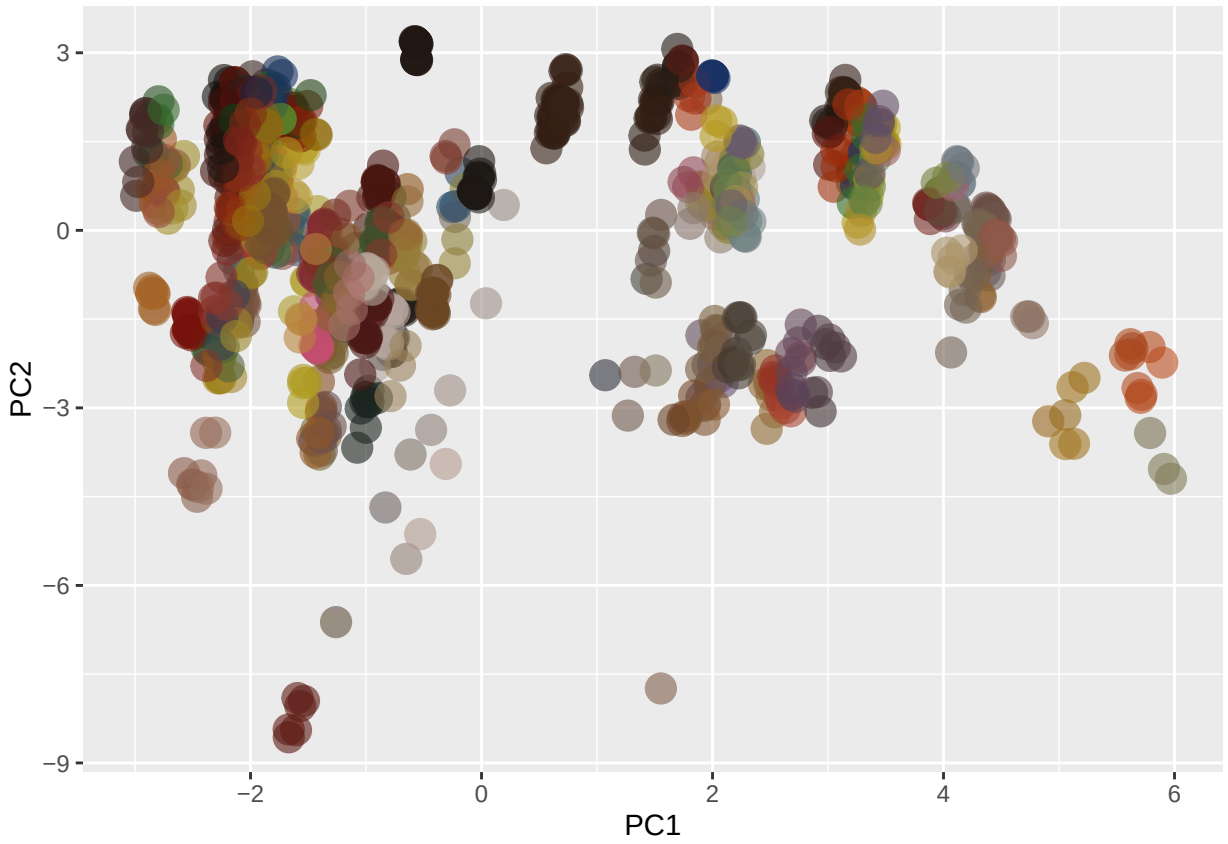
Now plot our candy information using PC1 and PC2

```
p <- ggplot(pca_scores, aes(PC1, PC2, colour=class))
p + geom_point(size=5, alpha=0.6) + scale_colour_brewer(type="qual", palette = 2)
```



Now we'll make the point the mean colors of the actual candies

```
p <- ggplot(pca_scores, aes(PC1, PC2, colour=rgb(Mean.Red, Mean.Green, Mean.Blue, maxColorValue = 255)))
p + geom_point(size=5, alpha=0.6) + scale_color_identity()
```



Second, we'll do a principle component analysis based only on the shape information.

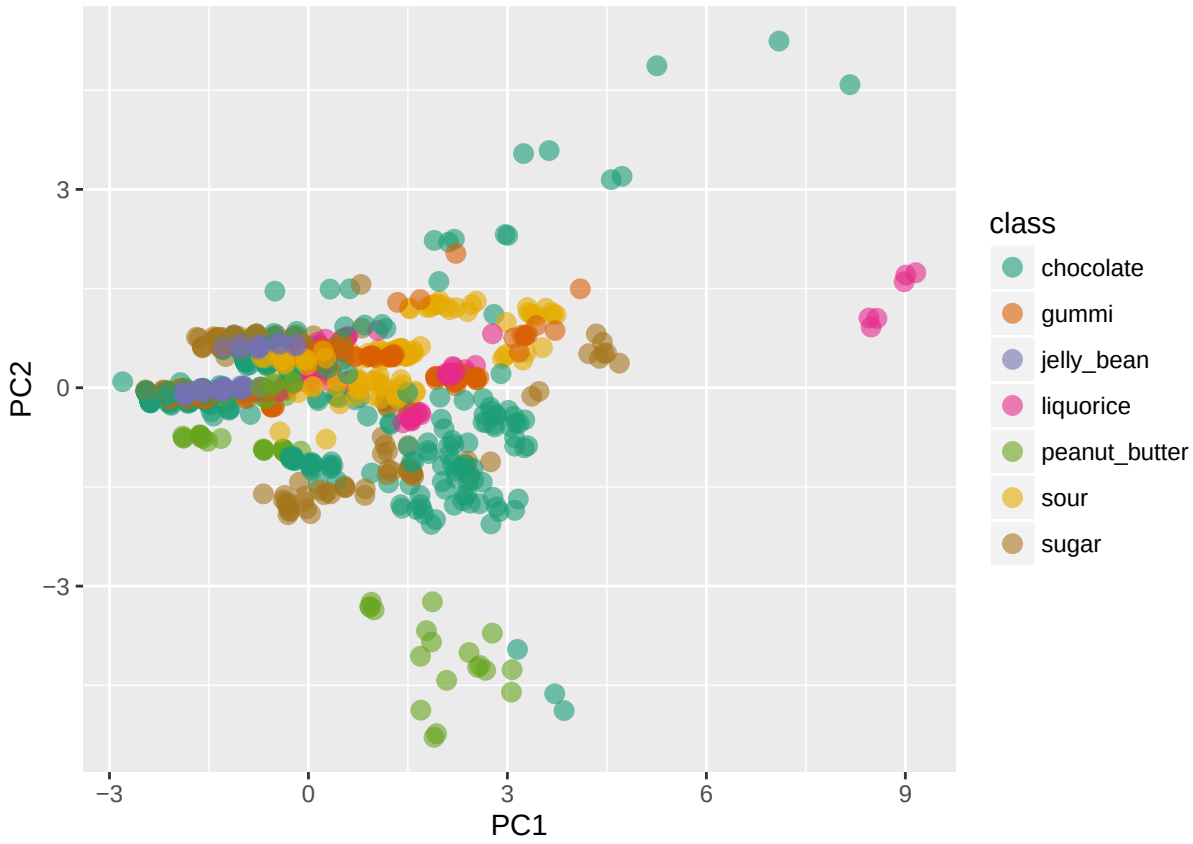
```
shape_pca <- prcomp(merged_color_shape_nutrition[c(4:5,7:9)], center = TRUE, scale. = TRUE)

summary(shape_pca)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5
## Standard deviation     1.7258 0.9317 0.8064 0.52541 0.47659
## Proportion of Variance 0.5957 0.1736 0.1301 0.05521 0.04543
## Cumulative Proportion  0.5957 0.7693 0.8994 0.95457 1.00000
```

```
shape_pca$rotation
```

```
##                 PC1          PC2        PC3        PC4         PC5
## cm2       0.3182575 -0.850962662  0.2421031  0.3278451 -0.09207804
## Circ     -0.5072872 -0.004143061 -0.3381955  0.5795874 -0.54068924
## AR        0.4676279  0.141779777 -0.5902492  0.4722855  0.43563163
## Round    -0.5027325  0.011938292  0.4052726  0.4020241  0.64903641
## Solidity -0.4122574 -0.505570326 -0.5606869 -0.4146312  0.29690671
```

```
shape_scores <- as.data.frame(shape_pca$x)
shape_pca_scores <- cbind(merged_color_shape_nutrition, shape_scores)

p <- ggplot(shape_pca_scores, aes(PC1, PC2, colour=class))
p + geom_point(size=3, alpha=0.6) + scale_colour_brewer(type="qual", palette = 2)
```



Now we'll do a principle component analysis based only on the color information

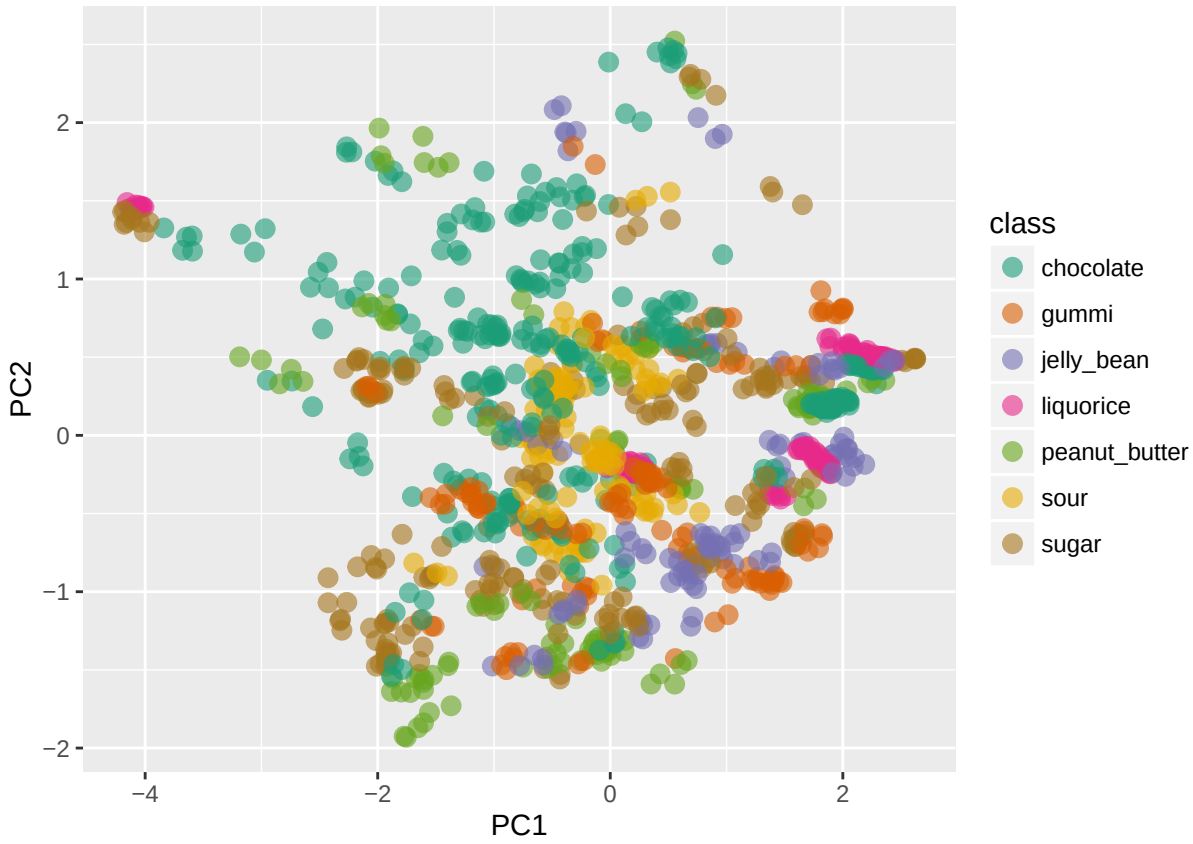
```
color_pca <- prcomp(merged_color_shape_nutrition[c(10:12)], center = TRUE, scale. = TRUE)

summary(color_pca)
```

```
## Importance of components:
##                           PC1    PC2     PC3
## Standard deviation     1.4062 0.8548 0.54041
## Proportion of Variance 0.6591 0.2435 0.09735
## Cumulative Proportion  0.6591 0.9026 1.00000
```

```
color_pca$rotation
```

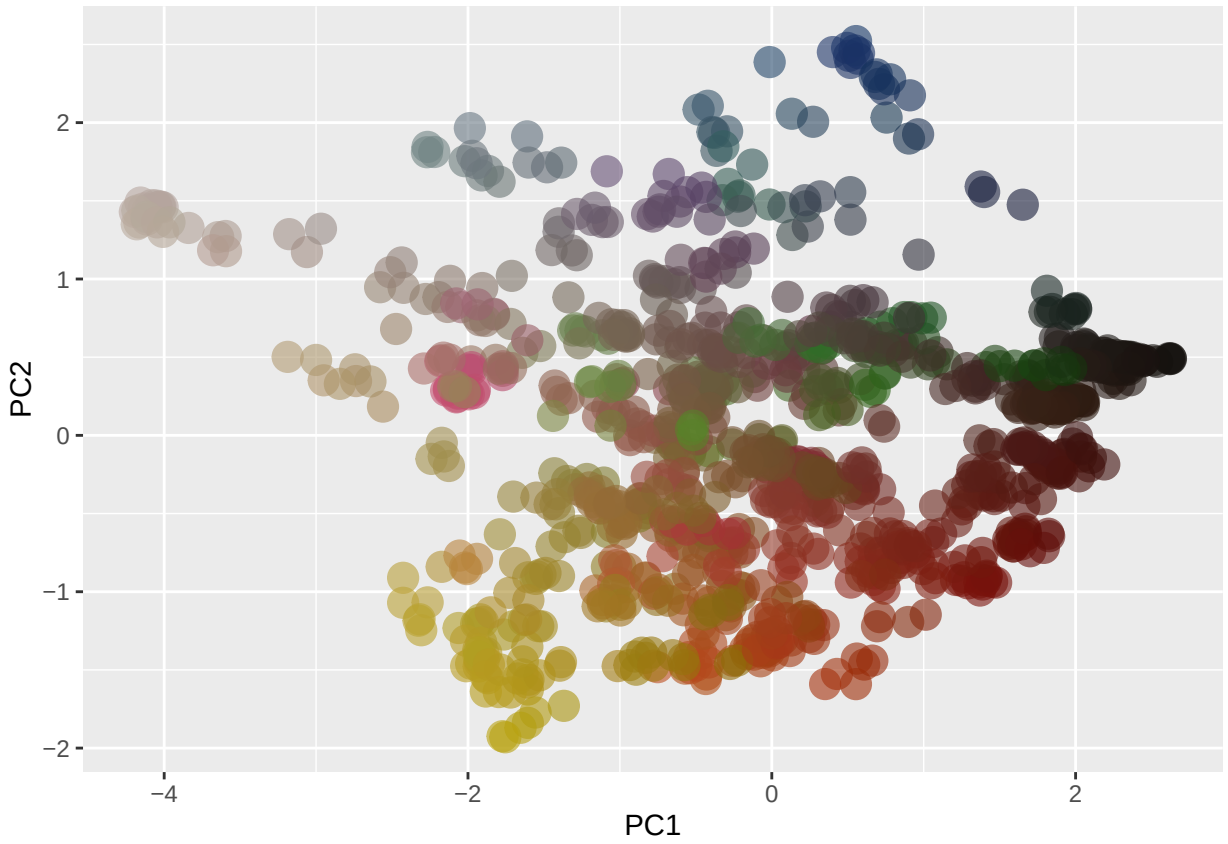
```
##                   PC1         PC2        PC3
## Mean.Green -0.6476034 -0.05127971  0.7602501
## Mean.Blue  -0.5195724  0.75952945 -0.3913560
## Mean.Red   -0.5573637 -0.64844846 -0.5185174
```

```
shape_scores <- as.data.frame(color_pca$x)
color_pca_scores <- cbind(merged_color_shape_nutrition, shape_scores)

p <- ggplot(color_pca_scores, aes(PC1, PC2, colour=class))
p + geom_point(size=3, alpha=0.6) + scale_colour_brewer(type="qual", palette = 2)
```



This would look really cool using the actual candy colors.

```
p <- ggplot(color_pca_scores, aes(PC1, PC2, colour=rgb(Mean.Red, Mean.Green, Mean.Blue, maxColorValue =
p + geom_point(size=5, alpha=0.6) + scale_color_identity()
```



Now do a principle component analysis on the nutrition information

```
nutrition_pca <- prcomp(merged_color_shape_nutrition[c(28:35)], center = TRUE, scale. = TRUE)

summary(nutrition_pca)
```

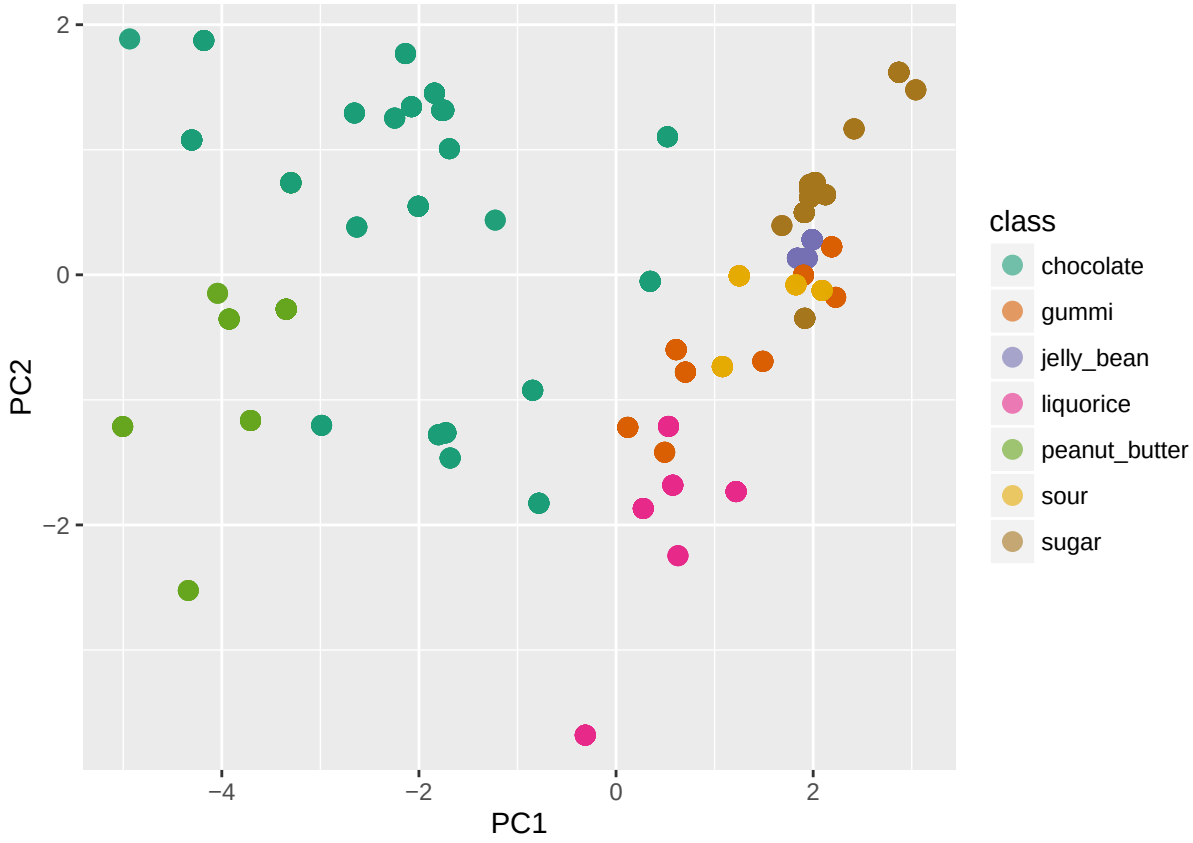
```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6
## Standard deviation     2.2385 1.0962 0.82961 0.71756 0.51022 0.45949
## Proportion of Variance 0.6264 0.1502 0.08603 0.06436 0.03254 0.02639
## Cumulative Proportion  0.6264 0.7766 0.86261 0.92698 0.95952 0.98591
##                           PC7     PC8
## Standard deviation     0.28062 0.18436
## Proportion of Variance 0.00984 0.00425
## Cumulative Proportion  0.99575 1.00000
```

```
nutrition_pca$rotation
```

```
##                                PC1         PC2         PC3         PC4
## total_fat_per_serv      -0.4258544  0.15230008  0.16678991  0.08750172
## saturated_fat_per_serv -0.3944010  0.19234821  0.21817337  0.14084474
## cholesterol_per_serv   -0.2903991  0.44901914  0.12870489 -0.76038133
## sodium_per_serv        -0.2021186 -0.58361848  0.72868972 -0.01256169
## total_carb_per_serv     0.4250225  0.12582020  0.16144698  0.06762863
## dietary_fiber_per_serv -0.3791893  0.20707763 -0.02021696  0.39933066
## sugars_per_serv         0.2623695  0.57791402  0.44406256  0.38496156
```

```
## protein_per_serv       -0.3803234 -0.06986875 -0.39161518  0.28624225
##                                PC5          PC6         PC7         PC8
## total_fat_per_serv     -0.23343218  0.025030924 -0.21178334 -0.81240874
## saturated_fat_per_serv -0.67253884  0.002228528  0.35266521  0.40413732
## cholesterol_per_serv    0.29027798 -0.105963657  0.14346969  0.05685352
## sodium_per_serv         0.25308010 -0.124502755  0.07374769  0.04900847
## total_carb_per_serv     0.04282573  0.066248098  0.79109931 -0.37526708
## dietary_fiber_per_serv  0.49659213  0.614583472  0.12001742  0.12140788
## sugars_per_serv         0.16029882 -0.367005906 -0.27810544  0.11857060
## protein_per_serv        0.26600110 -0.675162398  0.29524203 -0.03750560
```

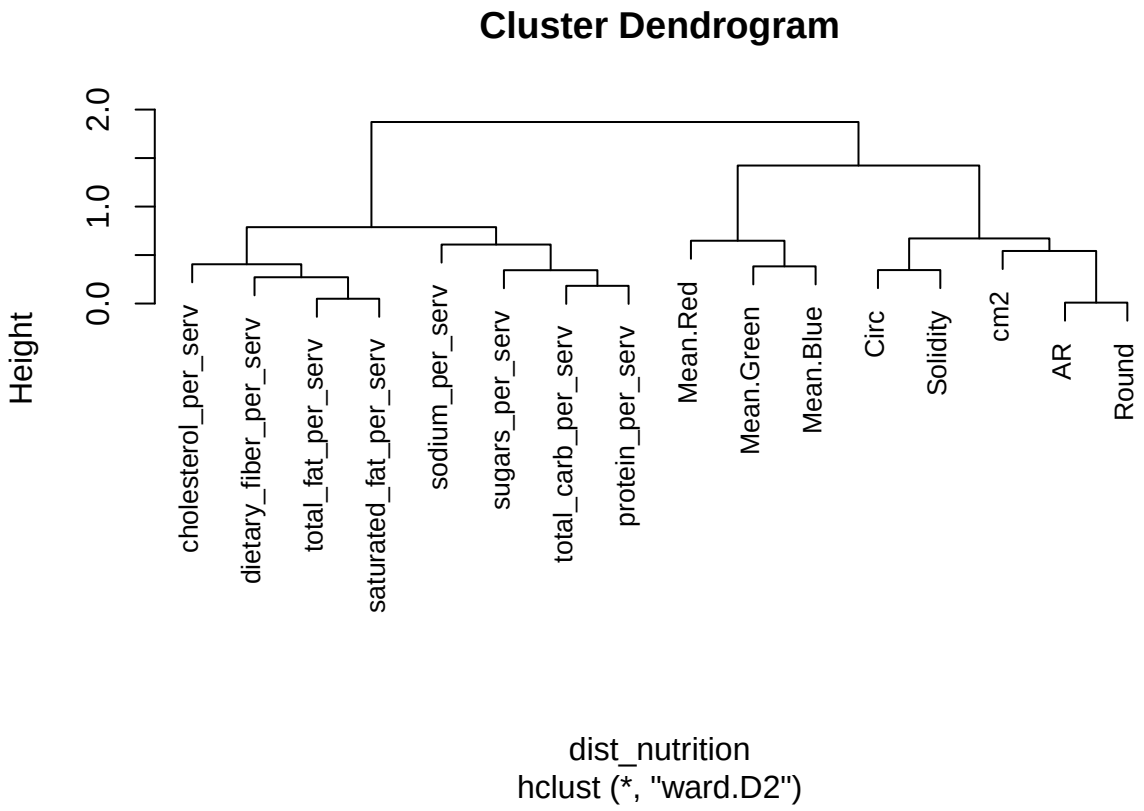
```
shape_scores <- as.data.frame(nutrition_pca$x)
nutrition_pca_scores <- cbind(merged_color_shape_nutrition, shape_scores)

p <- ggplot(nutrition_pca_scores, aes(PC1, PC2, colour=class))
p + geom_point(size=3, alpha=0.6) + scale_colour_brewer(type="qual", palette = 2)
```



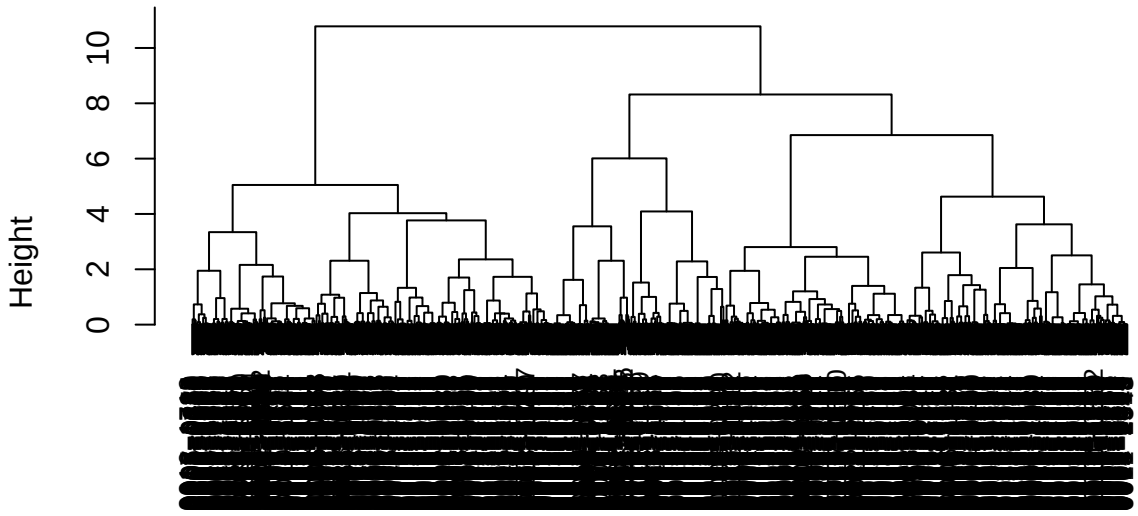
HIERARCHICAL CLUSTERING TIME

```
scaled_everything <- scale(merged_color_shape_nutrition[c(4:5, 7:12, 28:35)])
scaled_candies <- scale(t(scaled_everything))
colnames(scaled_candies) <- as.matrix(merged_color_shape_nutrition[2])

corell_nutrition <- cor(scaled_everything, method="spearman")
```

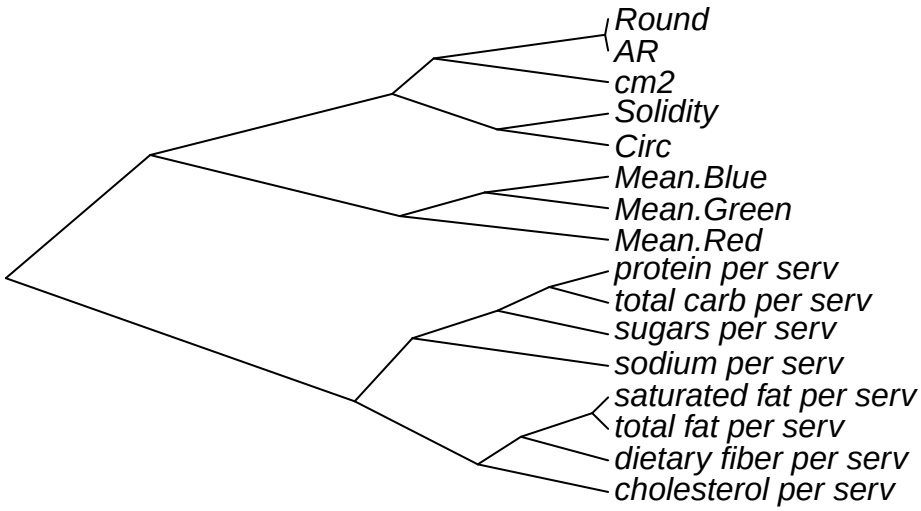
```r
corell_candies <- cor(scaled_candies, method="spearman")

dist_nutrition <- as.dist(1-abs(corell_nutrition))
dist_candies <- as.dist(1-abs(corell_candies))

hc_nutrition <- hclust(dist_nutrition, method="ward.D2")
hc_candies <- hclust(dist_candies, method="ward.D2")

plot(hc_nutrition, cex=0.8)
```




```r
plot(hc_candies, cex=0.8)
```




```
plot(as.phylo(hc_nutrition), type="cladogram", label.offset=0.01)
```




```
plot(as.phylo(hc_candies), type="cladogram", label.offset=0.01, cex=0.5)
```