Halloween Mini-Project

Job Rocha PID A59023124

Exploratory Analysis of Halloween Candy

Background

In this mini-project, you will explore FiveThirtyEight's Halloween Candy dataset. FiveThirtyEight, sometimes rendered as just 538, is an American website that focuses mostly on opinion poll analysis, politics, economics, and sports blogging. They recently ran a rather large poll to determine which candy their readers like best. From their website: "While we don't know who exactly voted, we do know this: 8,371 different IP addresses voted on about 269,000 randomly generated candy matchups".

So what is the top ranked snack-sized Halloween candy? What made some candies more desirable than others? Was it price? Maybe it was just sugar content? Were they chocolate? Did they contain peanuts or almonds? How about crisped rice or other biscuit-esque component, like a Kit Kat or malted milk ball? Was it fruit flavored? Was it made of hard candy, like a lollipop or a strawberry bon bon? Was there nougat? What even is nougat? I know I like nougat, but I still have no real clue what the damn thing is.

Your task is to explore their candy dataset to find out answers to these types of questions - but most of all your job is to have fun, learn by doing hands on data analysis, and hopefully make this type of analysis less frightining for the future! Let's get started.

1. Importing candy data.

First things first, let's get the data from the FiveThirtyEight GitHub repo. You can either read from the URL directely or download this candy-data.csv file and place it in your project directory. Either way we need to load it up with read.csv() and inspect the data to see exactly what we're dealing with.

```r
candy_file <- "https://raw.githubusercontent.com/fivethirtyeight/data/master/candy-power-r

candy = read.csv(candy_file, row.names=1)
head(candy)
```

```
             chocolate fruity caramel peanutyalmondy nougat crispedricewafer
100 Grand            1      0       1              0      0                 1
3 Musketeers         1      0       0              0      1                 0
One dime             0      0       0              0      0                 0
One quarter          0      0       0              0      0                 0
Air Heads            0      1       0              0      0                 0
Almond Joy           1      0       0              1      0                 0
             hard bar pluribus sugarpercent pricepercent winpercent
100 Grand       0   1        0        0.732        0.860   66.97173
3 Musketeers    0   1        0        0.604        0.511   67.60294
One dime        0   0        0        0.011        0.116   32.26109
One quarter     0   0        0        0.011        0.511   46.11650
Air Heads       0   0        0        0.906        0.511   52.34146
Almond Joy      0   1        0        0.465        0.767   50.34755
```

What is in the dataset?

The dataset includes all sorts of information about different kinds of candy. For example, is a candy chocolaty? Does it have nougat? How does its cost compare to other candies? How many people prefer one candy over another?

According to 538 the columns in the dataset include:

chocolate: Does it contain chocolate? fruity: Is it fruit flavored? caramel: Is there caramel in the candy? peanutyalmondy: Does it contain peanuts, peanut butter or almonds? nougat: Does it contain nougat? crispedricewafer: Does it contain crisped rice, wafers, or a cookie component? hard: Is it a hard candy? bar: Is it a candy bar? pluribus: Is it one of many candies in a bag or box? sugarpercent: The percentile of sugar it falls under within the data set. pricepercent: The unit price percentile compared to the rest of the set. winpercent: The overall win percentage according to 269,000 matchups (more on this in a moment).

We will take a whirlwind tour of this dataset and in the process answer the questions highlighted in red throught this page that aim to guide your exploration process. We will then wrap up by trying Principal Component Analysis (PCA) on this dataset to get yet more experience with this important multivariate method. It will yield a kind of "Map of Hallowen Candy Space". How cool is that! Let's explore...

Q1. How many different candy types are in this dataset? ***85 different candy types.***

```
nrow(candy)
```

```
[1] 85
```

Q2. How many fruity candy types are in the dataset? ***38 fruity candy types.***

```
sum(candy$fruity)
```

```
[1] 38
```

2. What is your favorate candy?

One of the most interesting variables in the dataset is winpercent. For a given candy this value is the percentage of people who prefer this candy over another randomly chosen candy from the dataset (what 538 term a matchup). Higher values indicate a more popular candy.

We can find the winpercent value for Twix by using its name to access the corresponding row of the dataset. This is because the dataset has each candy name as rownames (recall that we set this when we imported the original CSV file). For example the code for Twix is:

```
candy["Twix", ]$winpercent
```

```
[1] 81.64291
```

Q3. What is your favorite candy in the dataset and what is it's winpercent value? ***winpercent value equals to 55.37545.***

```
candy["Hershey's Kisses", ]$winpercent
```

```
[1] 55.37545
```

Q4. What is the winpercent value for "Kit Kat"? ***winpercent value equals to 76.7686.***

```
candy["Kit Kat", ]$winpercent
```

```
[1] 76.7686
```

Q5. What is the winpercent value for "Tootsie Roll Snack Bars"? ***winpercent value equals to 49.6535.***

```
candy["Tootsie Roll Snack Bars", ]$winpercent
```

```
[1] 49.6535
```

Side-note: the skimr::skim() function

There is a useful skim() function in the **skimr** package that can help give you a quick overview of a given dataset. Let's install this package and try it on our candy data.

```
#install.packages("skimr")
library("skimr")
skim(candy)
```

Table 1: Data summary

Name	candy
Number of rows	85
Number of columns	12
_____	
Column type frequency:	
numeric	12
_____	
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
chocolate	0	1	0.44	0.50	0.00	0.00	0.00	1.00	1.00	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
fruity	0	1	0.45	0.50	0.00	0.00	0.00	1.00	1.00	
caramel	0	1	0.16	0.37	0.00	0.00	0.00	0.00	1.00	
peanutyalmondy	0	1	0.16	0.37	0.00	0.00	0.00	0.00	1.00	
nougat	0	1	0.08	0.28	0.00	0.00	0.00	0.00	1.00	
crispedricewafer	0	1	0.08	0.28	0.00	0.00	0.00	0.00	1.00	
hard	0	1	0.18	0.38	0.00	0.00	0.00	0.00	1.00	
bar	0	1	0.25	0.43	0.00	0.00	0.00	0.00	1.00	
pluribus	0	1	0.52	0.50	0.00	0.00	1.00	1.00	1.00	
sugarpercent	0	1	0.48	0.28	0.01	0.22	0.47	0.73	0.99	
pricepercent	0	1	0.47	0.29	0.01	0.26	0.47	0.65	0.98	
winpercent	0	1	50.32	14.71	22.45	39.14	47.83	59.86	84.18	

From your use of the skim() function use the output to answer the following:

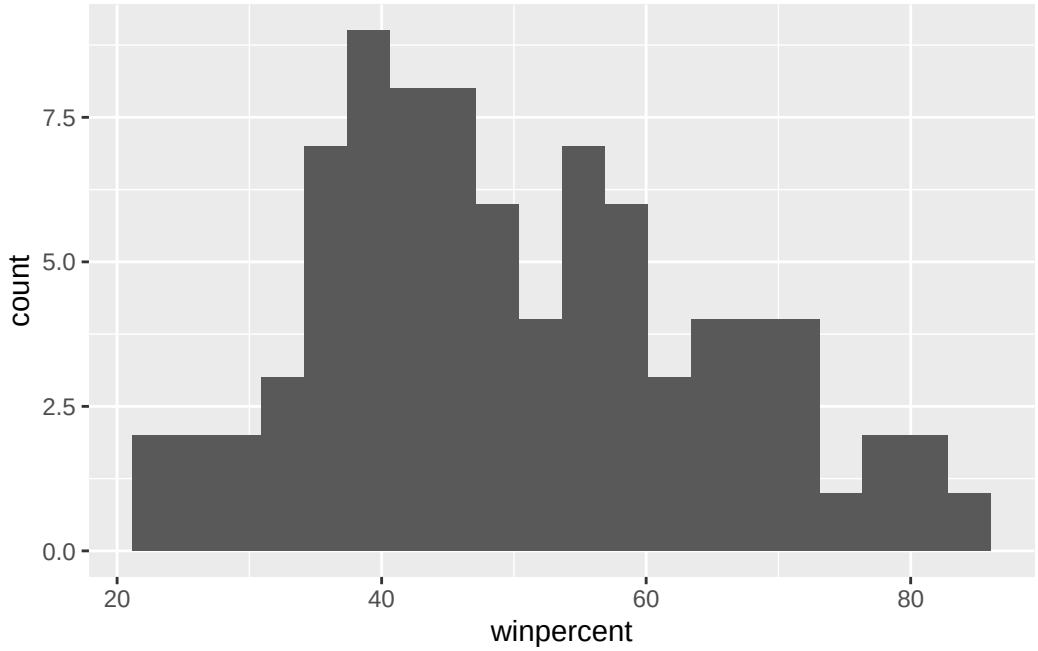
Q6. Is there any variable/column that looks to be on a different scale to the majority of the other columns in the dataset? ***Most of the columns seem to go from 0 to 1, except for winpercent which goes from ~22 to ~84.***

Q7. What do you think a zero and one represent for the candy$chocolate column? ***Zero means it doesn't contain chocolate and One means it has chocolate.***

A good place to start any exploratory analysis is with a histogram. You can do this most easily with the base R function hist(). Alternatively, you can use ggplot() with geom_hist(). Either works well in this case and (as always) its your choice.

Q8. Plot a histogram of winpercent values

```
library(ggplot2)

ggplot(candy, aes(x=winpercent)) +
  geom_histogram(bins = 20)
```



Q9. Is the distribution of winpercent values symmetrical? ***No it is not. Looks like it is skewed to the left side.***

Q10. Is the center of the distribution above or below 50%? ***It is skewed to the left, then the center of the distribution is below 50%.***

Q11. On average is chocolate candy higher or lower ranked than fruit candy? ***Chocolate is higher ranked than fruit candy. Mean Chocolate 60.92153 > Mean Fruit 44.11974.***

```
t.test( x = candy$winpercent[as.logical(candy$chocolate)],
        y = candy$winpercent[as.logical(candy$fruity)]
      )
```

```
    Welch Two Sample t-test

data:  candy$winpercent[as.logical(candy$chocolate)] and candy$winpercent[as.logical(candy$fi
t = 6.2582, df = 68.882, p-value = 2.871e-08
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 11.44563 22.15795
```

```
sample estimates:
mean of x mean of y
 60.92153  44.11974
```

Q12. Is this difference statistically significant? ***Yes, pvalue is 2.871e-08 < 0.05.***

3. Overall Candy Rankings

Let's use the base R order() function together with head() to sort the whole dataset by win-percent. Or if you have been getting into the tidyverse and the dplyr package you can use the arrange() function together with head() to do the same thing and answer the following questions:

Q13. What are the five least liked candy types in this set? ***Nik L Nip, Boston Baked Beans, Chiclets, Super Bubble and Jawbusters.***

```
library(dplyr)
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
candy %>% arrange(winpercent) %>% head(n=5)
```

	chocolate	fruity	caramel	peanutyalmondy	nougat
Nik L Nip	0	1	0	0	0
Boston Baked Beans	0	0	0	1	0
Chiclets	0	1	0	0	0
Super Bubble	0	1	0	0	0
Jawbusters	0	1	0	0	0

```
                  crispedricewafer hard bar pluribus sugarpercent pricepercent
Nik L Nip                       0    0   0        1        0.197        0.976
Boston Baked Beans              0    0   0        1        0.313        0.511
Chiclets                        0    0   0        1        0.046        0.325
Super Bubble                    0    0   0        0        0.162        0.116
Jawbusters                      0    1   0        1        0.093        0.511
                  winpercent
Nik L Nip           22.44534
Boston Baked Beans  23.41782
Chiclets            24.52499
Super Bubble        27.30386
Jawbusters          28.12744
```

Q14. What are the top 5 all time favorite candy types out of this set? *Reese's Peanut Butter cup, Reese's Miniatures, Twix, Kit Kat and Snickers.*

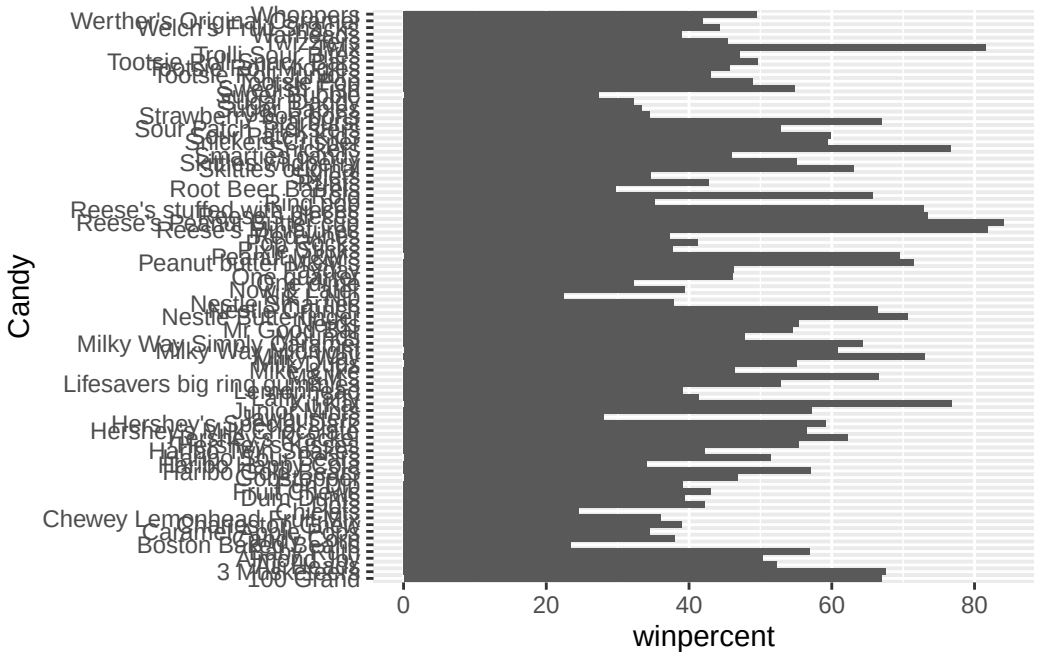
```
candy %>% arrange(desc(winpercent)) %>% head(n=5)
```

```
                        chocolate fruity caramel peanutyalmondy nougat
Reese's Peanut Butter cup         1      0       0              1      0
Reese's Miniatures                1      0       0              1      0
Twix                              1      0       1              0      0
Kit Kat                           1      0       0              0      0
Snickers                          1      0       1              1      1
                        crispedricewafer hard bar pluribus sugarpercent
Reese's Peanut Butter cup               0    0   0        0        0.720
Reese's Miniatures                      0    0   0        0        0.034
Twix                                    1    0   1        0        0.546
Kit Kat                                 1    0   1        0        0.313
Snickers                                0    0   1        0        0.546
                        pricepercent winpercent
Reese's Peanut Butter cup        0.651   84.18029
Reese's Miniatures               0.279   81.86626
Twix                             0.906   81.64291
Kit Kat                          0.511   76.76860
Snickers                         0.651   76.67378
```

To examine more of the dataset in this vain we can make a barplot to visualize the overall rankings. We will use an iterative approach to building a useful visulization by getting a rough starting plot and then refining and adding useful details in a stepwise process.

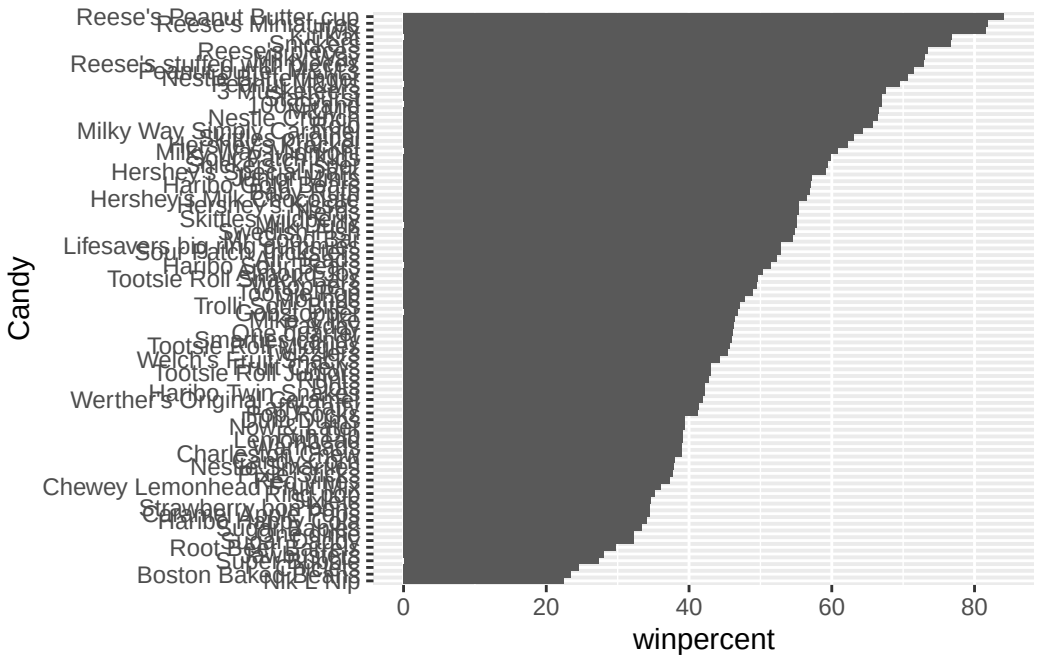
Q15. Make a first barplot of candy ranking based on winpercent values.

```
ggplot(candy) +
  aes(winpercent, rownames(candy)) +
  geom_col() +
  ylab("Candy")
```



Q16. This is quite ugly, use the reorder() function to get the bars sorted by winpercent?

```
ggplot(candy) +
  aes(winpercent, reorder(rownames(candy),winpercent)) +
  geom_col() +
  ylab("Candy")
```



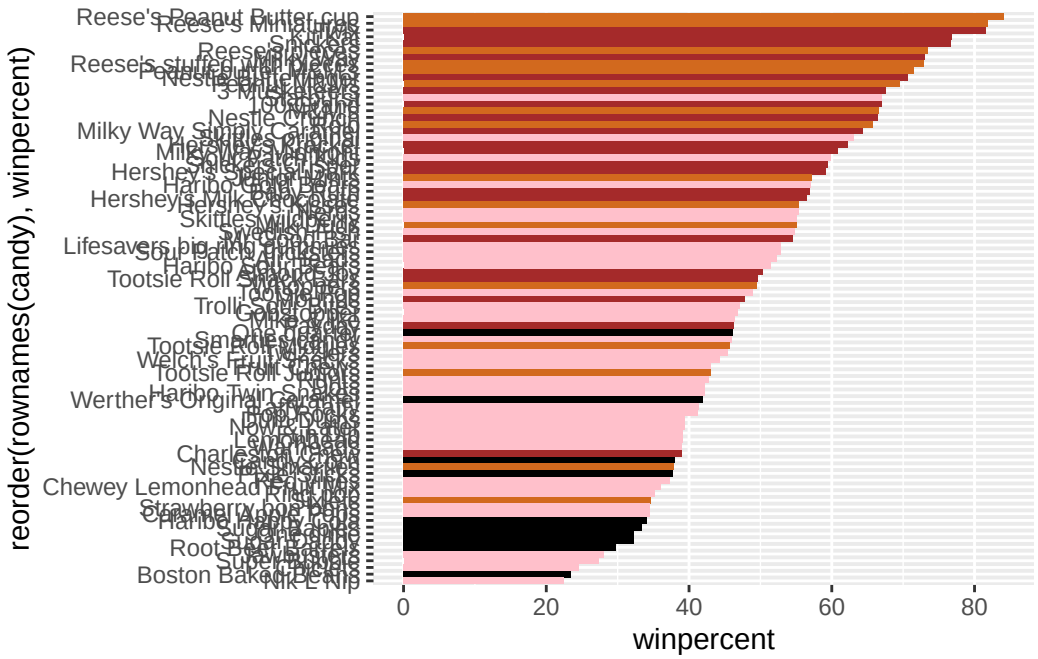
Time to add some useful color

Let's setup a color vector (that signifies candy type) that we can then use for some future plots. We start by making a vector of all black values (one for each candy). Then we overwrite chocolate (for chocolate candy), brown (for candy bars) and red (for fruity candy) values.

```
my_cols=rep("black", nrow(candy))
my_cols[as.logical(candy$chocolate)] = "chocolate"
my_cols[as.logical(candy$bar)] = "brown"
my_cols[as.logical(candy$fruity)] = "pink"
```

Now let's try our barplot with these colors. Note that we use fill=my_cols for geom_col(). Experement to see what happens if you use col=mycols.

```
ggplot(candy) +
  aes(winpercent, reorder(rownames(candy),winpercent)) +
  geom_col(fill=my_cols)
```



Now, for the first time, using this plot we can answer questions like:

Q17. What is the worst ranked chocolate candy? *Sixlets.*

Q18. What is the best ranked fruity candy? *Starburst.*

4. Taking a look at pricepercent

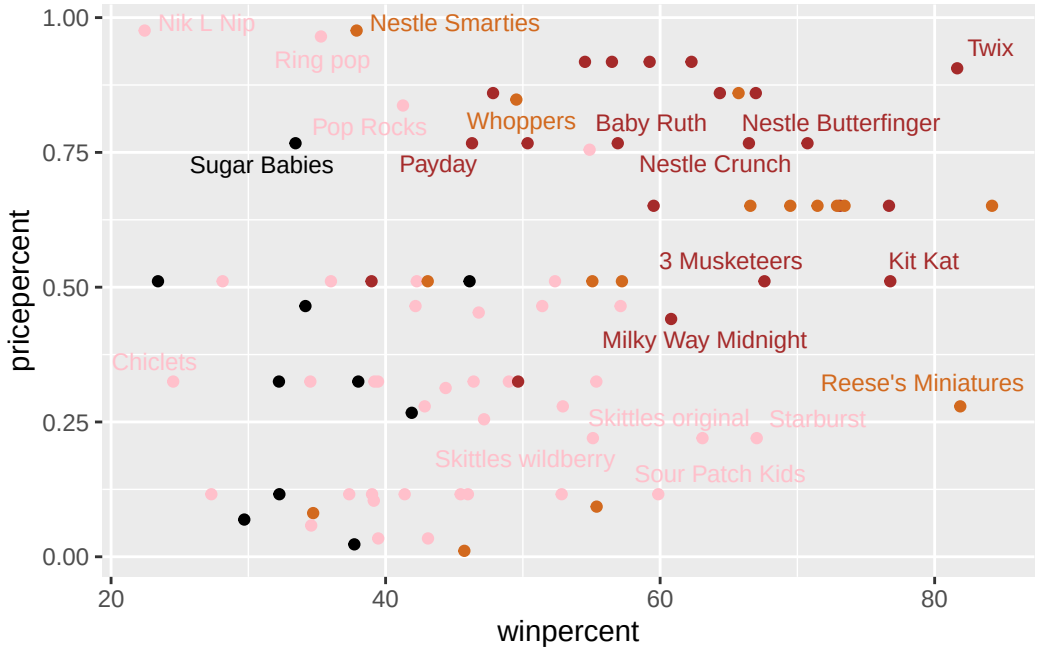
What about value for money? What is the the best candy for the least money? One way to get at this would be to make a plot of winpercent vs the pricepercent variable. The pricepercent variable records the percentile rank of the candy's price against all the other candies in the dataset. Lower vales are less expensive and high values more expensive.

To this plot we will add text labels so we can more easily identify a given candy. There is a regular geom_label() that comes with ggplot2. However, as there are quite a few candys in our dataset lots of these labels will be overlapping and hard to read. To help with this we can use the geom_text_repel() function from the ggrepel package.

```
#install.packages("ggrepel")
library(ggrepel)
```

```
# How about a plot of price vs win
ggplot(candy) +
  aes(winpercent, pricepercent, label=rownames(candy)) +
  geom_point(col=my_cols) +
  geom_text_repel(col=my_cols, size=3.3, max.overlaps = 5)
```

```
Warning: ggrepel: 65 unlabeled data points (too many overlaps). Consider
increasing max.overlaps
```



Q19. Which candy type is the highest ranked in terms of winpercent for the least money - i.e. offers the most bang for your buck? ***Chocolate Candy Reese's Miniatures.***

```
candy[ order(candy$winpercent, decreasing = TRUE), c(11,12)]
```

```
                           pricepercent winpercent
Reese's Peanut Butter cup         0.651   84.18029
Reese's Miniatures                0.279   81.86626
Twix                              0.906   81.64291
Kit Kat                           0.511   76.76860
```

Snickers	0.651	76.67378
Reese's pieces	0.651	73.43499
Milky Way	0.651	73.09956
Reese's stuffed with pieces	0.651	72.88790
Peanut butter M&M's	0.651	71.46505
Nestle Butterfinger	0.767	70.73564
Peanut M&Ms	0.651	69.48379
3 Musketeers	0.511	67.60294
Starburst	0.220	67.03763
100 Grand	0.860	66.97173
M&M's	0.651	66.57458
Nestle Crunch	0.767	66.47068
Rolo	0.860	65.71629
Milky Way Simply Caramel	0.860	64.35334
Skittles original	0.220	63.08514
Hershey's Krackel	0.918	62.28448
Milky Way Midnight	0.441	60.80070
Sour Patch Kids	0.116	59.86400
Snickers Crisper	0.651	59.52925
Hershey's Special Dark	0.918	59.23612
Junior Mints	0.511	57.21925
Haribo Gold Bears	0.465	57.11974
Baby Ruth	0.767	56.91455
Hershey's Milk Chocolate	0.918	56.49050
Hershey's Kisses	0.093	55.37545
Nerds	0.325	55.35405
Skittles wildberry	0.220	55.10370
Milk Duds	0.511	55.06407
Swedish Fish	0.755	54.86111
Mr Good Bar	0.918	54.52645
Lifesavers big ring gummies	0.279	52.91139
Sour Patch Tricksters	0.116	52.82595
Air Heads	0.511	52.34146
Haribo Sour Bears	0.465	51.41243
Almond Joy	0.767	50.34755
Tootsie Roll Snack Bars	0.325	49.65350
Whoppers	0.848	49.52411
Tootsie Pop	0.325	48.98265
Mounds	0.860	47.82975
Trolli Sour Bites	0.255	47.17323
Gobstopper	0.453	46.78335
Mike & Ike	0.325	46.41172
Payday	0.767	46.29660

One quarter	0.511	46.11650
Smarties candy	0.116	45.99583
Tootsie Roll Midgies	0.011	45.73675
Twizzlers	0.116	45.46628
Welch's Fruit Snacks	0.313	44.37552
Fruit Chews	0.034	43.08892
Tootsie Roll Juniors	0.511	43.06890
Runts	0.279	42.84914
Dots	0.511	42.27208
Haribo Twin Snakes	0.465	42.17877
Werther's Original Caramel	0.267	41.90431
Laffy Taffy	0.116	41.38956
Pop Rocks	0.837	41.26551
Dum Dums	0.034	39.46056
Now & Later	0.325	39.44680
Fun Dip	0.325	39.18550
Lemonhead	0.104	39.14106
Warheads	0.116	39.01190
Charleston Chew	0.511	38.97504
Candy Corn	0.325	38.01096
Nestle Smarties	0.976	37.88719
Pixie Sticks	0.023	37.72234
Red vines	0.116	37.34852
Chewey Lemonhead Fruit Mix	0.511	36.01763
Ring pop	0.965	35.29076
Sixlets	0.081	34.72200
Strawberry bon bons	0.058	34.57899
Caramel Apple Pops	0.325	34.51768
Haribo Happy Cola	0.465	34.15896
Sugar Babies	0.767	33.43755
One dime	0.116	32.26109
Sugar Daddy	0.325	32.23100
Root Beer Barrels	0.069	29.70369
Jawbusters	0.511	28.12744
Super Bubble	0.116	27.30386
Chiclets	0.325	24.52499
Boston Baked Beans	0.511	23.41782
Nik L Nip	0.976	22.44534

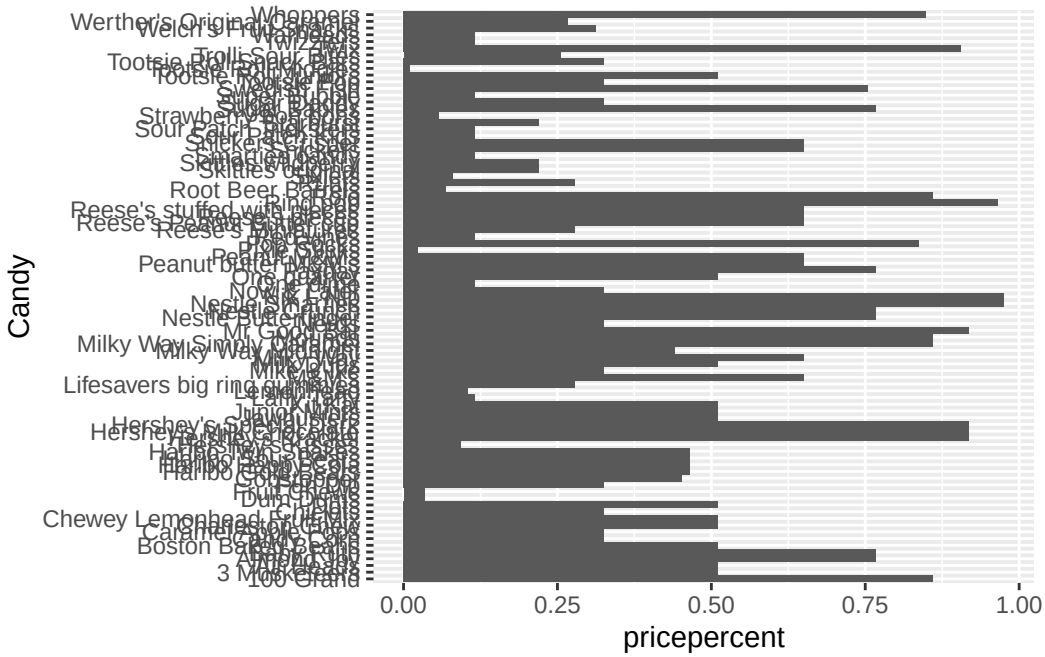
Q20. What are the top 5 most expensive candy types in the dataset and of these which is the least popular? ***The top 5 more expensive are: Nik L Nip, Nestle Smarties, Ring pop, Hershey's Krackel, Hershey's Milk Chocolate. The least favorite among these is Nik L Nip.***

```
head(
  candy[ order(candy$pricepercent, decreasing = TRUE), c(11,12)],
  n = 5
  )
```

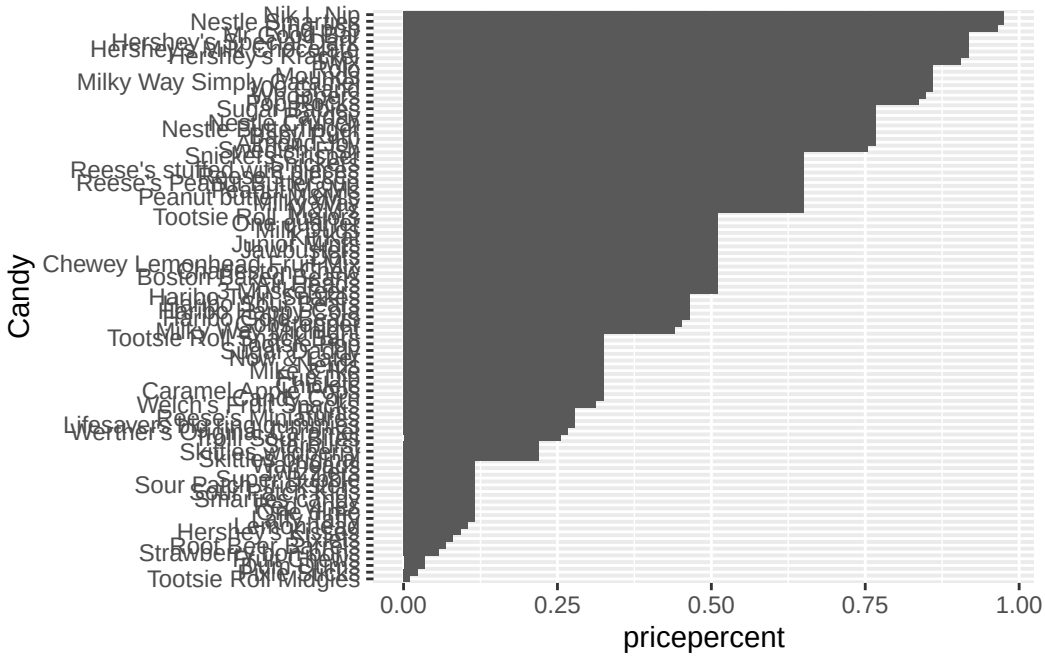
```
                          pricepercent winpercent
Nik L Nip                        0.976   22.44534
Nestle Smarties                  0.976   37.88719
Ring pop                         0.965   35.29076
Hershey's Krackel                0.918   62.28448
Hershey's Milk Chocolate         0.918   56.49050
```

Q21. Make a barplot again with geom_col() this time using pricepercent and then improve this step by step, first ordering the x-axis by value and finally making a so called "dot chat" or "lollipop" chart by swapping geom_col() for geom_point() + geom_segment().

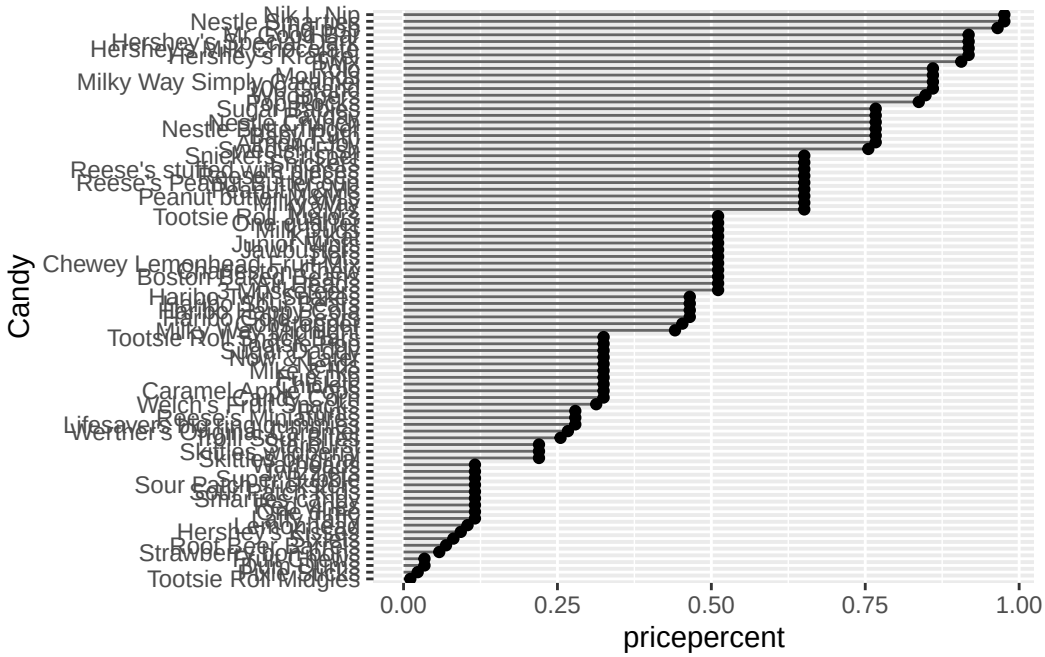
```
ggplot(candy, aes(x=pricepercent, y=rownames(candy))) +
  geom_col() +
  ylab("Candy")
```




```
ggplot(candy,
       aes(x=pricepercent, y=reorder(rownames(candy),pricepercent))
       ) +
  geom_col() +
  ylab("Candy")
```




```
ggplot(candy,
       aes(x=pricepercent, y=reorder(rownames(candy),pricepercent))
       ) +
  geom_segment(aes(yend = reorder(rownames(candy), pricepercent), xend = 0), col="gray40") +
  geom_point() +
  ylab("Candy")
```



One of the most interesting aspects of this chart is that a lot of the candies share the same ranking, so it looks like quite a few of them are the same price.

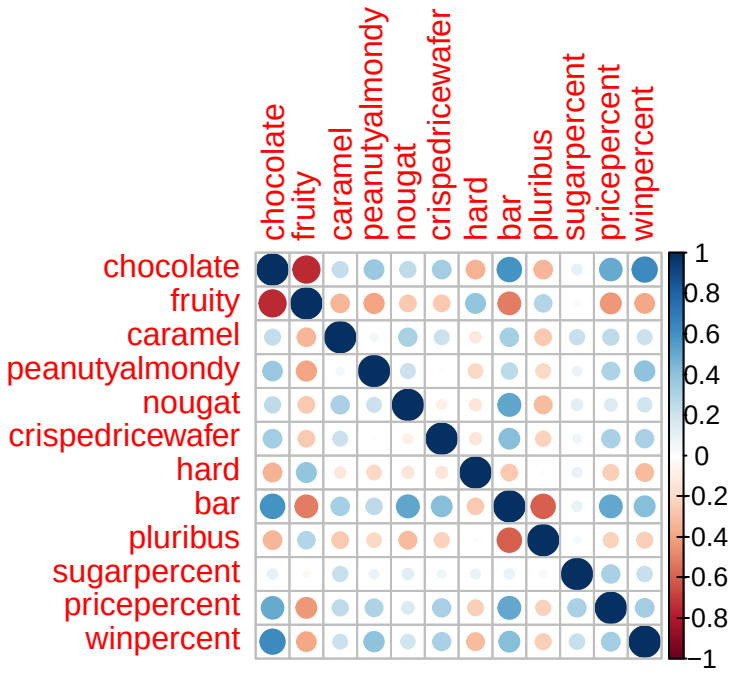
5 Exploring the correlation structure.

Now that we've explored the dataset a little, we'll see how the variables interact with one another. We'll use correlation and view the results with the corrplot package to plot a correlation matrix.

```
#install.packages("corrplot")
library(corrplot)
```

```
corrplot 0.92 loaded
```

```
cij <- cor(candy)
corrplot(cij)
```



Q22. Examining this plot what two variables are anti-correlated (i.e. have minus values)? ***Fruity and Chocolate are anti-correlated.***

Q23. Similarly, what two variables are most positively correlated? ***Winpercent and Chocolate.***

6. Principal Component Analysis

Let's apply PCA using the prcom() function to our candy dataset remembering to set the scale=TRUE argument.

```
pca <- prcomp(candy, scale=TRUE)
summary(pca)
```

```
Importance of components:
                          PC1    PC2    PC3     PC4    PC5     PC6     PC7
Standard deviation     2.0788 1.1378 1.1092 1.07533 0.9518 0.81923 0.81530
Proportion of Variance 0.3601 0.1079 0.1025 0.09636 0.0755 0.05593 0.05539
Cumulative Proportion  0.3601 0.4680 0.5705 0.66688 0.7424 0.79830 0.85369
                           PC8     PC9    PC10    PC11    PC12
Standard deviation     0.74530 0.67824 0.62349 0.43974 0.39760
```

```
Proportion of Variance 0.04629 0.03833 0.03239 0.01611 0.01317
Cumulative Proportion  0.89998 0.93832 0.97071 0.98683 1.00000
```

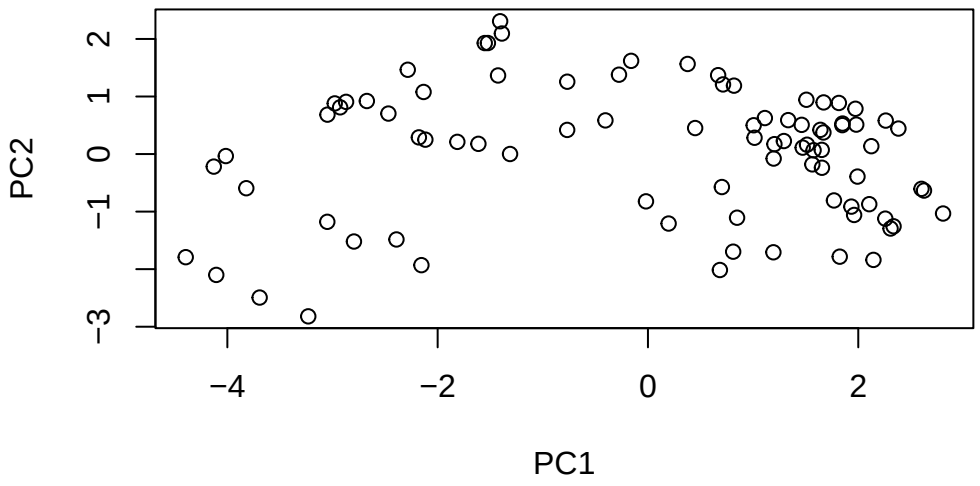
Side-note: Feel free to examine what happens if you leave this argument out (i.e. use the default scale=FALSE). Then examine the summary(pca) and pca$rotation[,1] component and see that it is dominated by winpercent (which is after all measured on a very different scale than the other variables).

```
pca <- prcomp(candy, scale=FALSE)
summary(pca)
```

```
Importance of components:
                          PC1     PC2     PC3     PC4     PC5     PC6     PC7
Standard deviation     14.7231 0.70241 0.47762 0.37292 0.34641 0.33614 0.30748
Proportion of Variance  0.9935 0.00226 0.00105 0.00064 0.00055 0.00052 0.00043
Cumulative Proportion   0.9935 0.99574 0.99678 0.99742 0.99797 0.99849 0.99892
                           PC8     PC9    PC10    PC11    PC12
Standard deviation     0.27417 0.23826 0.21435 0.18434 0.15331
Proportion of Variance 0.00034 0.00026 0.00021 0.00016 0.00011
Cumulative Proportion  0.99927 0.99953 0.99974 0.99989 1.00000
```

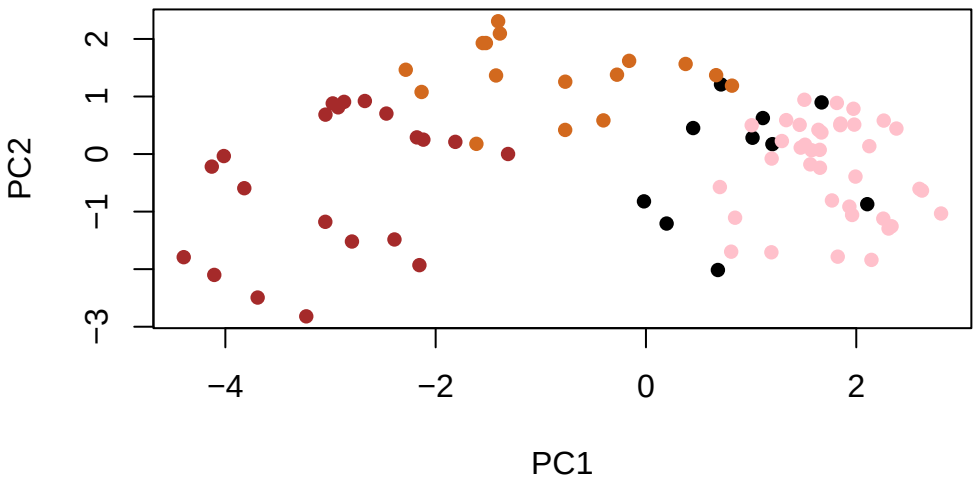
Now we can plot our main PCA score plot of PC1 vs PC2.

```
pca <- prcomp(candy, scale=TRUE)
plot(pca$x[, 1:2])
```



We can change the plotting character and add some color:

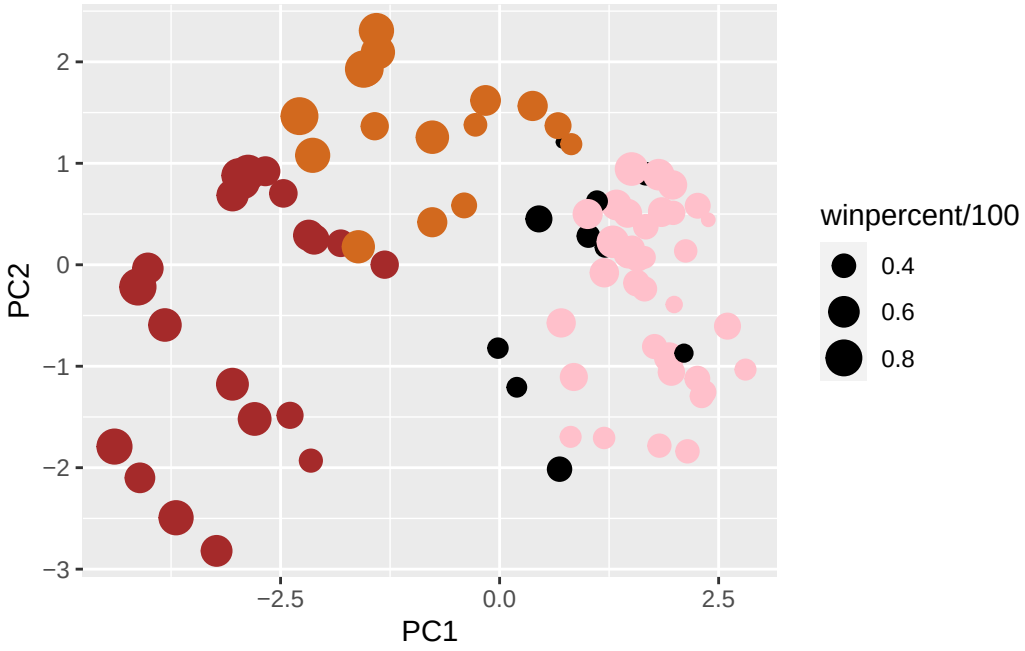
```
plot(pca$x[,1:2], col=my_cols, pch=16)
```



We can make a much nicer plot with the ggplot2 package but it is important to note that ggplot works best when you supply an input data.frame that includes a separate column for each of the aesthetics you would like displayed in your final plot. To accomplish this we make a new data.frame here that contains our PCA results with all the rest of our candy data. We will then use this for making plots below

```
# Make a new data-frame with our PCA results and candy data
my_data <- cbind(candy, pca$x[,1:3])
```

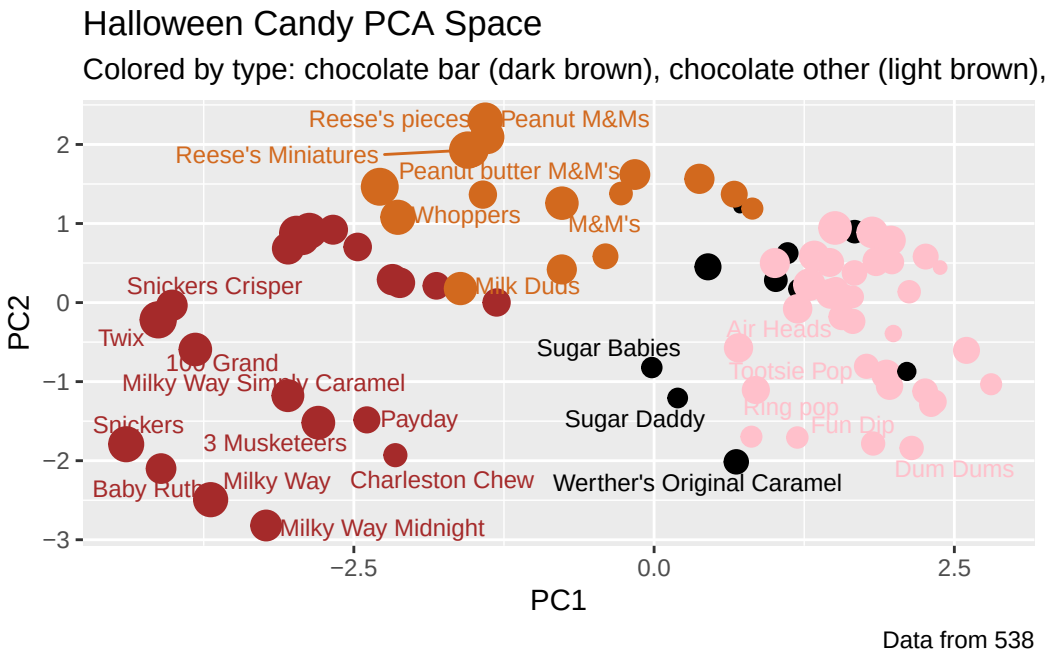
```
p <- ggplot(my_data) +
        aes(x=PC1, y=PC2,
            size=winpercent/100,
            text=rownames(my_data),
            label=rownames(my_data)) +
        geom_point(col=my_cols)

p
```



Again we can use the ggrepel package and the function ggrepel::geom_text_repel() to label up the plot with non overlapping candy names like. We will also add a title and subtitle like so:

```
p + geom_text_repel(size=3.3, col=my_cols, max.overlaps = 7)  +
  theme(legend.position = "none") +
  labs(title="Halloween Candy PCA Space",
       subtitle="Colored by type: chocolate bar (dark brown), chocolate other (light brown
       caption="Data from 538")
```

```
Warning: ggrepel: 59 unlabeled data points (too many overlaps). Consider
increasing max.overlaps
```



more candy labels you can change the max.overlaps value to allow more overlapping labels or pass the ggplot object p to plotly like so to generate an interactive plot that you can mouse over to see labels:

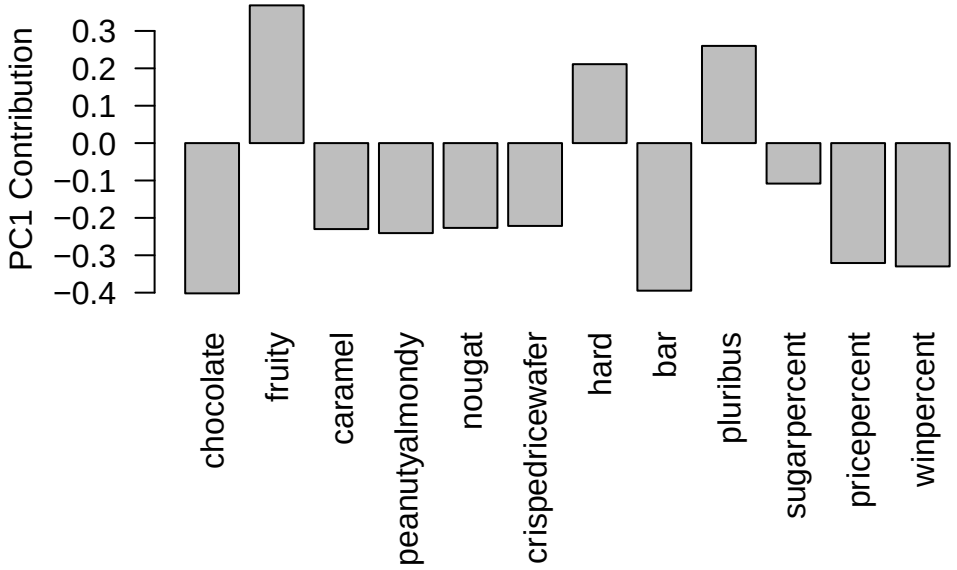
```
#install.packages("plotly")
library(plotly)
```

```
Attaching package: 'plotly'

The following object is masked from 'package:ggplot2':

    last_plot

The following object is masked from 'package:stats':

    filter

The following object is masked from 'package:graphics':

    layout
```

```
#ggplotly(p)
```

Let's finish by taking a quick look at PCA our loadings. Do these make sense to you? Notice the opposite effects of chocolate and fruity and the similar effects of chocolate and bar (i.e. we already know they are correlated).

```
par(mar=c(8,4,2,2))
barplot(pca$rotation[,1], las=2, ylab="PC1 Contribution")
```



Q24. What original variables are picked up strongly by PC1 in the positive direction? Do these make sense to you? ***The variables strongly picked by PC1 in the positive direction are Fruity and Pluribus. Yes, they make sense because as seen before they are the ones contributing the most to the variance.***