# Lab 9 - 2.4.25 - Halloween Candy Project

Jessica Le (PID: A17321021)

## Table of contents



Today we will examind data from 538 common Halloween candy. In particular, we will use ggplot, dplyr, and PCA to make sense of this multivariate dataset.

### Importing candy data

```
candy_file <- "Lab 9 - candy-data.txt"

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
```

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

```
nrow(candy)
```

```
[1] 85
```

There are 85 different types of candy in the dataset.

Q2. How many fruity candy types are in this dataset?

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

```
[1] 38
```

There are 38 fruity candy types in the dataset.

How many chocolate candy are there in the dataset?

```
sum(candy$chocolate)
```

```
[1] 37
```

There are 27 chocolate candy in the dataset.

### What is your favorite candy?

`winpercent` provides the percentage of people who prefer a specific candy type over another randomly chosen candy from the dataset. A higher value indicates a more popular candy.

Q3. What is your favorite candy in the dataset and what is it's `winpercent` value?

```
candy["Reese's Peanut Butter cup", "winpercent"]
```

```
[1] 84.18029
```

```
candy["Reese's Peanut Butter cup", ]$winpercent
```

```
[1] 84.18029
```

When paired with another type of candy in the dataset, 84% of the time Reese's Peanut Butter cup, which is my favorite candy, will be picked as the favored one.

Q4. What is the `winpercent` value for "Kit Kat"?

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

```
[1] 76.7686
```

Q5. What is the `winpercent` value for "Tootsie Roll Snack Bars"?

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

```
[1] 49.6535
```

The `skim()` function in the skimr package that helps give an overview of a given dataset. First, the package needs to be installed using `install.packages("")`.

```
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 | |
| 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 | |

Q6. Is there any variable/column that looks to be on a different scale to the majority of the other columns in the dataset?

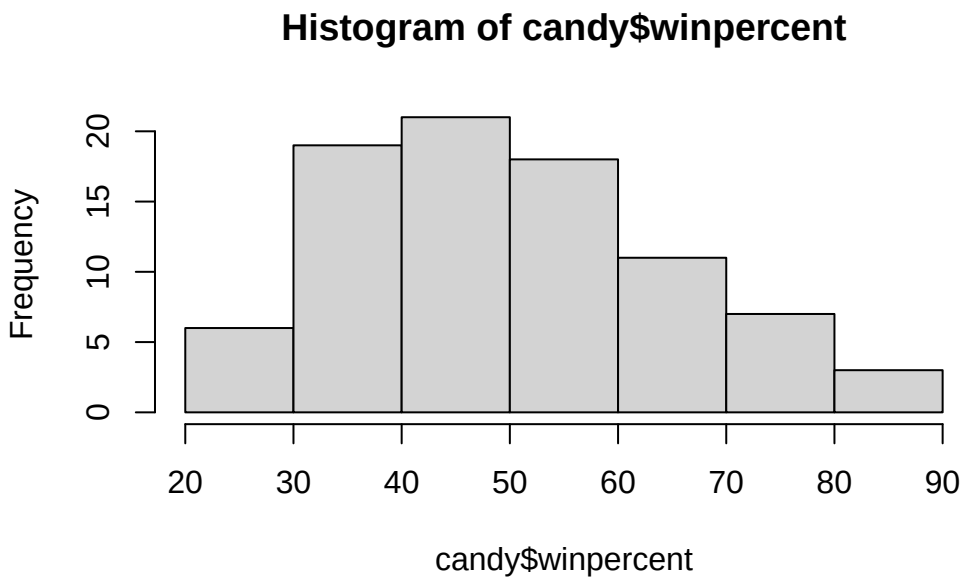
The winpercent variable is on a different scale to the other columns because all other variables go from 0-1 while this variable has values that range bewteen 14.71 to 50.32. The`winpercent` column goes from 0-100% rather than 0-1. We will need to scale this dataset before using it in analysis such as PCA.

Q7. What do you think a zero and one represent for the candy$chocolate column?

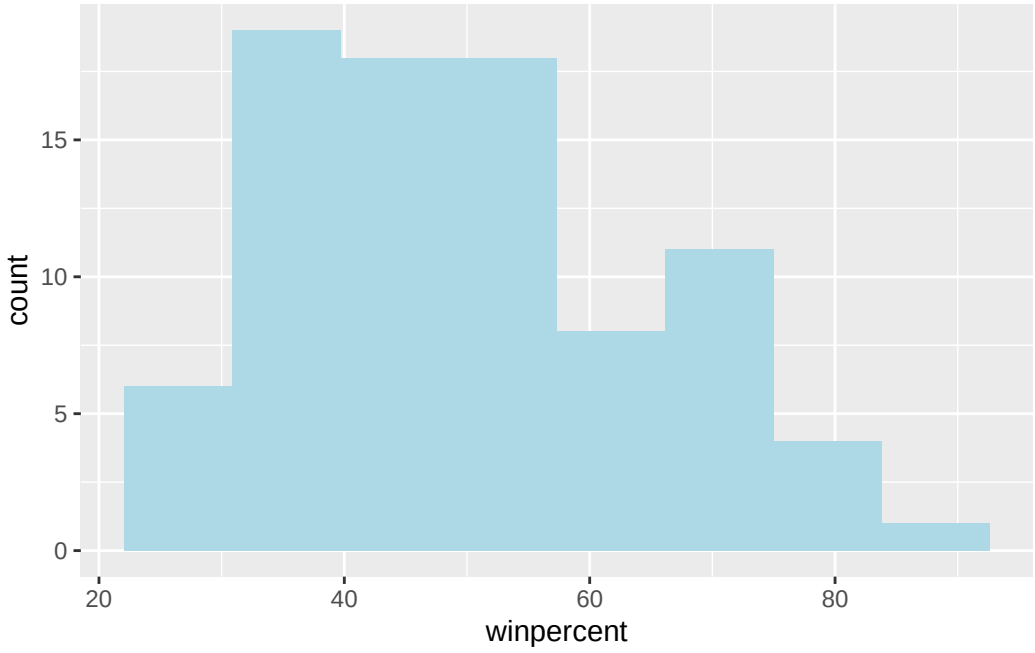
A zero indicates that the candy is not a chocolate type candy while a 1 indicates that the candy is a chocolate type.

Q8. Plot a histogram of `winpercent` values.

```
hist(candy$winpercent)
```




```
library(ggplot2)
ggplot(candy, aes(x=winpercent)) +
  geom_histogram(bins=8, fill="light blue")
```



Q9. Is the distribution of `winpercent` values symmetrical?

No, the distribution of `winpercent` values is not symmetrical based on the histogram.

Q10. Is the center of the distbrution above or below 50%?

```
summary (candy$winpercent)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  22.45   39.14   47.83   50.32   59.86   84.18 
```

The center of the distribution is slightly below 50%, where the median value is specifically 47.83%.

Q11. On average is chocolate candy higher or lower ranked than fruit candy?

- Step 1: Find all "chocolate" candy
- Step 2: Find their winpercent" values.
- Step 3: Summarize the values (through mean and median)
- Step 4: Find all "fruit" candy
- Step 2: Find their winpercent" values.
- Step 3: Summarize the values (through mean and median)
- Step 7: Compare the two summary values.

For the chocolate candies.

```
# Step 1 - Find all chocolate candy.
choc.inds <- candy$chocolate == 1

# Step 2 - Find their winpercent values.
choc.win <- candy[choc.inds, ]$winpercent

# Step 3 - Summarize the values
choc.mean <- mean(choc.win)
choc.mean
```

```
[1] 60.92153
```

For the fruity candies.

```
fruity.inds <- candy$fruit == 1
fruity.win <- candy[fruity.inds, ]$winpercent
fruity.mean <- mean(fruity.win)
fruity.mean
```

```
[1] 44.11974
```

On average, the chocolate candy is ranked higher than fruit candy because chocolate candy has an average winpercent value of 61% while it is only 44% for fruit candy.

```
choc.mean
```

```
[1] 60.92153
```

```
fruity.mean
```

```
[1] 44.11974
```

Q12. Is this difference statistically significant?

```
t.test(choc.win, fruity.win)
```

```
    Welch Two Sample t-test

data:  choc.win and fruity.win
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
```

Yes, the difference in the means comparing the rankings of the chocolate and fruity candy on average is statistically significant. People definitely prefer chocolate candy over fruity candy.

## Overall Candy Rankings

Q13. What are the five least liked candy types in this set?

```
# Not that useful - it just sorts the values
sort(candy$winpercent)
```

```
 [1] 22.44534 23.41782 24.52499 27.30386 28.12744 29.70369 32.23100 32.26109
 [9] 33.43755 34.15896 34.51768 34.57899 34.72200 35.29076 36.01763 37.34852
[17] 37.72234 37.88719 38.01096 38.97504 39.01190 39.14106 39.18550 39.44680
[25] 39.46056 41.26551 41.38956 41.90431 42.17877 42.27208 42.84914 43.06890
[33] 43.08892 44.37552 45.46628 45.73675 45.99583 46.11650 46.29660 46.41172
[41] 46.78335 47.17323 47.82975 48.98265 49.52411 49.65350 50.34755 51.41243
[49] 52.34146 52.82595 52.91139 54.52645 54.86111 55.06407 55.10370 55.35405
[57] 55.37545 56.49050 56.91455 57.11974 57.21925 59.23612 59.52925 59.86400
[65] 60.80070 62.28448 63.08514 64.35334 65.71629 66.47068 66.57458 66.97173
[73] 67.03763 67.60294 69.48379 70.73564 71.46505 72.88790 73.09956 73.43499
[81] 76.67378 76.76860 81.64291 81.86626 84.18029
```

```
x <- c(10,1,100)
sort(x)
```

```
[1]   1  10 100
```

```
order(x)
```

```
[1] 2 1 3
```

```
x[order (x)]
```

```
[1]   1  10 100
```

The `order()` function tells us how to arrange the elements of the input to make them sorted - i.e. how to order them.

We can determine the order of winpercent to make them sorted and use that order to arrange the whole dataset.

```
ord.inds <- order(candy$winpercent)
head (candy[ord.inds, ])
```

```
                  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
Root Beer Barrels         0      0       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
Root Beer Barrels                0    1   0        1        0.732        0.069
                  winpercent
Nik L Nip           22.44534
Boston Baked Beans  23.41782
Chiclets            24.52499
Super Bubble        27.30386
Jawbusters          28.12744
Root Beer Barrels   29.70369
```

Q14. What are the top 5 all time favorite candy types out of this set?

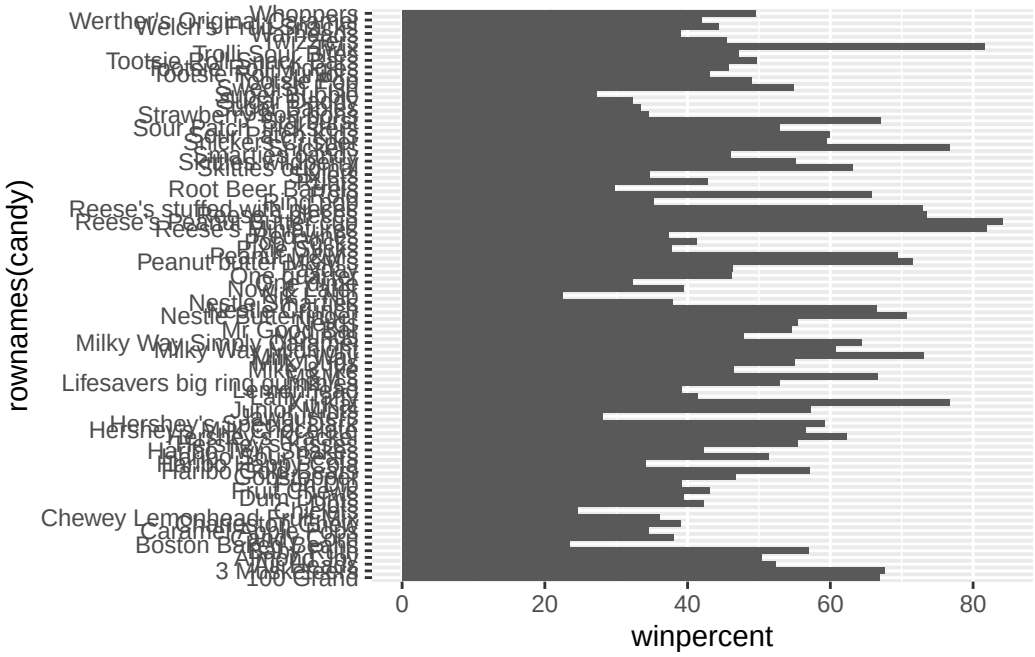
```
ord.inds <- order(candy$winpercent, decreasing=T)
head (candy[ord.inds, ])
```

```
                        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
Reese's pieces                    1      0       0              1      0
                        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
Reese's pieces                         0    0    0        1        0.406
                          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
```

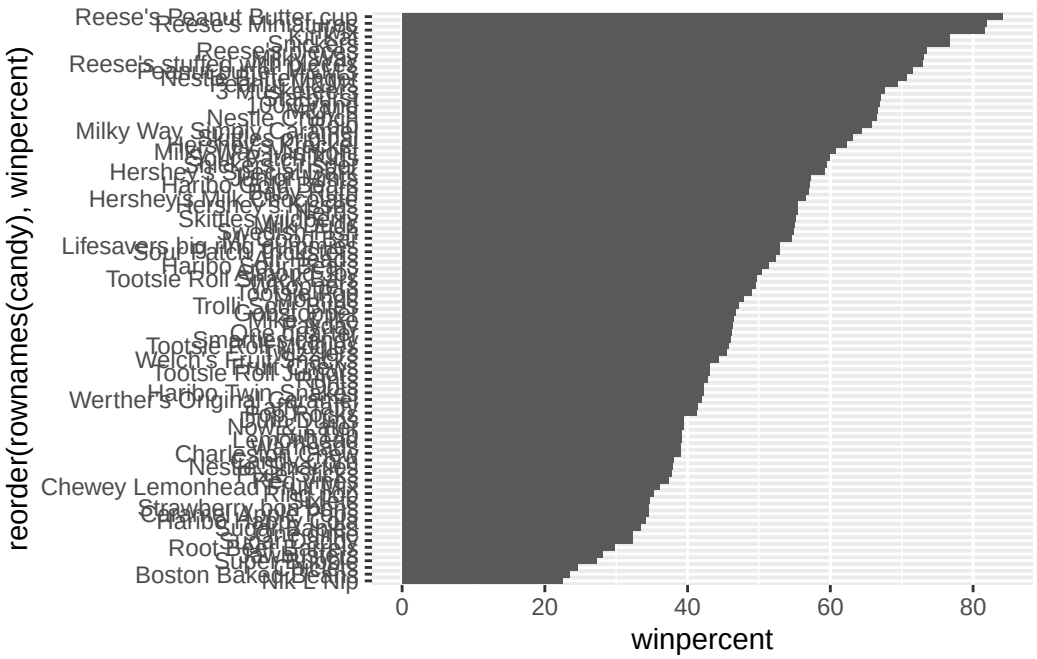
Q15. Make a first barplot of candy ranking based on winpercent values.

```
ggplot(candy) +
  aes(winpercent, rownames(candy)) +
  geom_col()
```



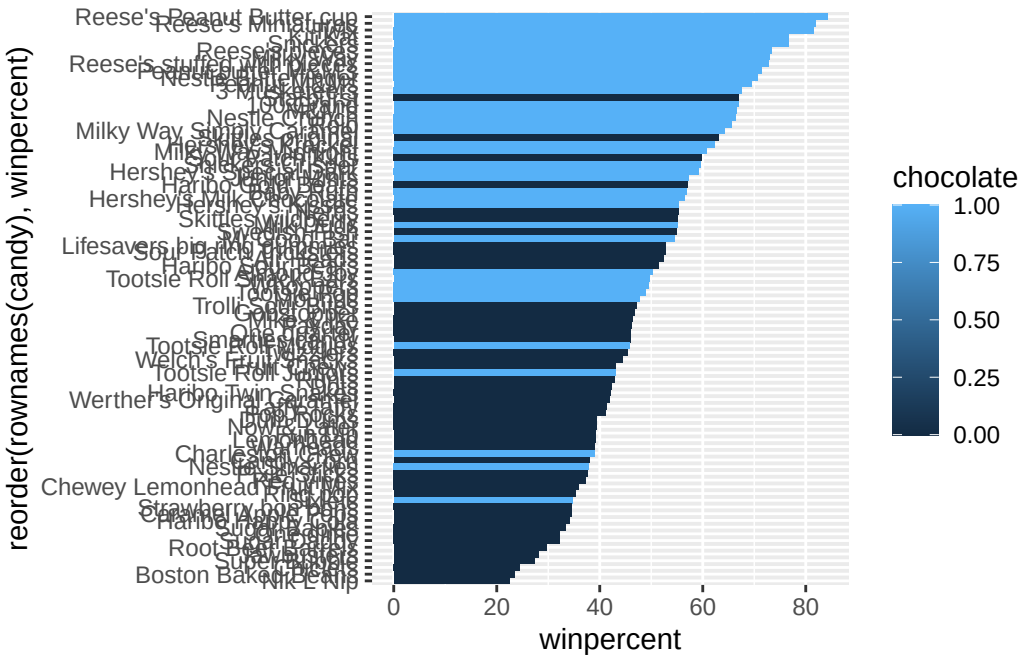

Q16. This is quite ugly, use the reorder() function to get the bars sorted by winpercent?

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




**Time to Add Some Useful Color**

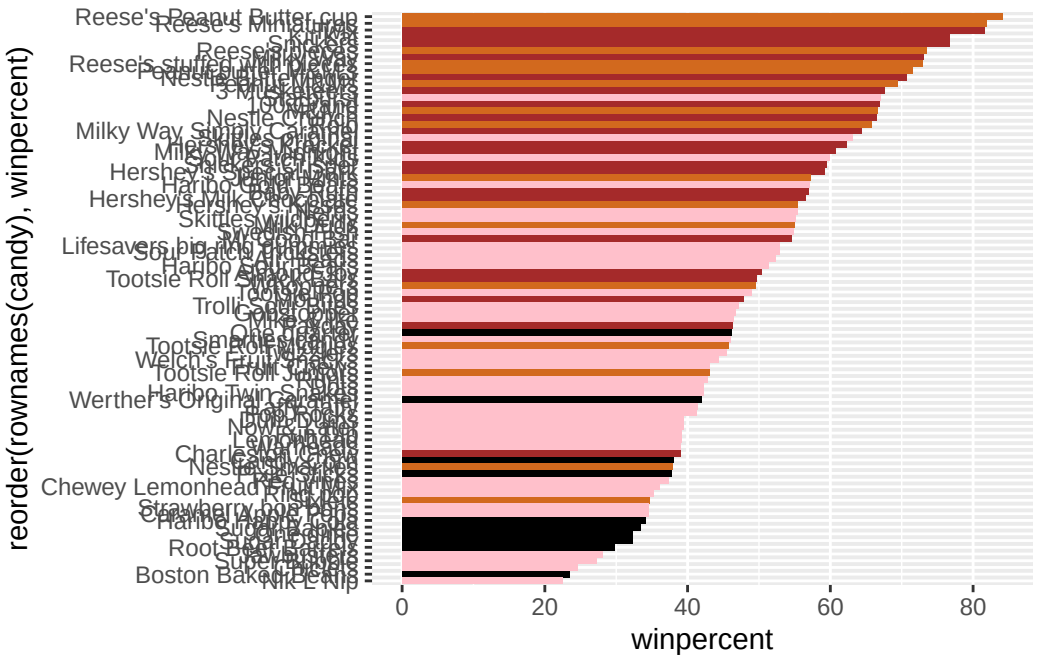
```
ggplot(candy) +
  aes(winpercent, reorder(rownames(candy), winpercent), fill=chocolate) +
  geom_col()
```




We need to make our own separate color vector where we can specify the type of color for each candy type.

```
my_cols <- rep("black", nrow(candy))
my_cols[candy$chocolate == 1] <- "chocolate"
my_cols[candy$bar == 1 ] <- "brown"
my_cols [candy$fruity == 1] <- "pink"
```

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




Q17. What is the worst ranked chocolate candy?

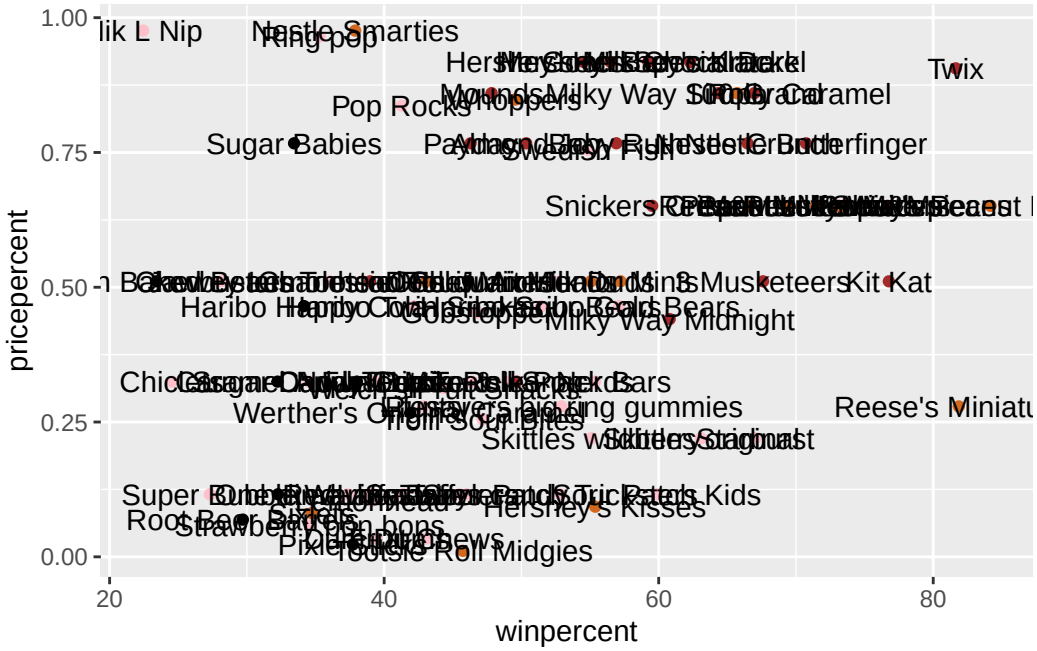
The worst ranked chocolate candy is Sixlets.

Q18. What is the best ranked fruity candy?

The best ranked fruity candy is Starburst.

### Taking a look at pricepercent

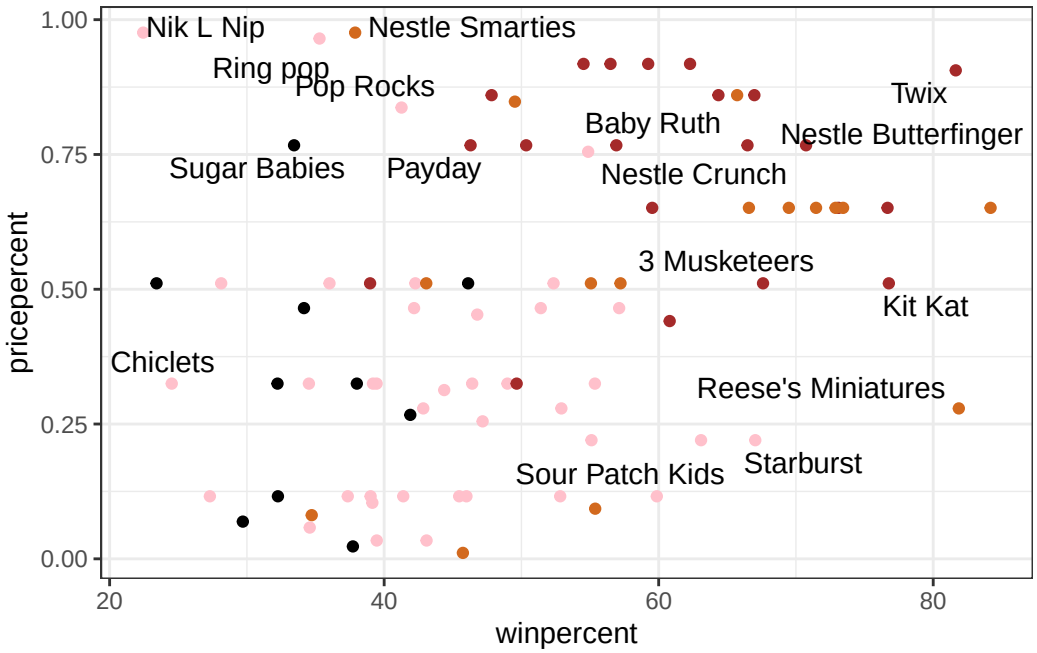
```
# Make a plot of winpercent (x-axis) vs. pricepercent (y-axis).
ggplot(candy) +
  aes(winpercent, pricepercent, label=rownames(candy)) +
  geom_point(col=my_cols) +
  geom_text()
```




To avoid overplotting of the text labels, we can use the add on package ggrepel

```
library(ggrepel)

ggplot(candy) +
  aes(winpercent, pricepercent, label=rownames(candy)) +
  geom_point(col=my_cols) +
  geom_text_repel(max.overlaps=6) +
  theme_bw()
```

```
Warning: ggrepel: 69 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?

Reese's Miniatures is the highest ranked candy in terms of winpercent for the least money.

Q20. What are the top 5 most expensive candy types in the dataset and of these which is the least popular?

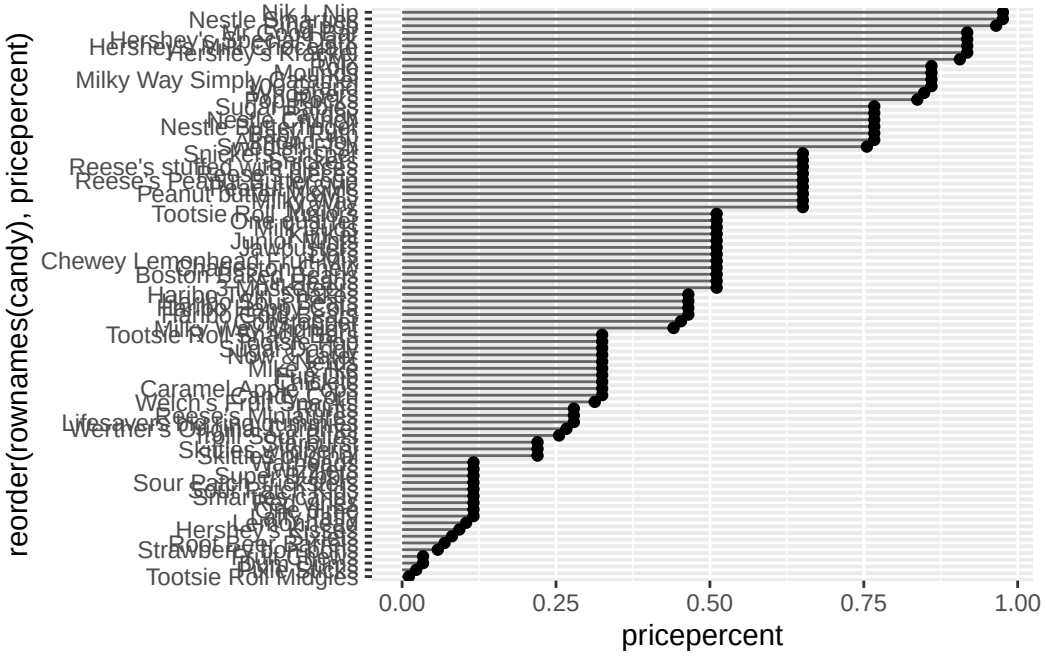
```
ord <- order(candy$pricepercent, decreasing = TRUE)
head(candy[ord, c(11,12)], n=5)
```

```
                      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
```

The top five most expensive candies that are the least popular are Nik L Nip, Nestle Smarties, Ring Pop, Hersheys Krackel, Hersheys Milk Chocolate.

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(pricepercent, reorder(rownames(candy), pricepercent)) +
  geom_segment(aes(yend = reorder(rownames(candy), pricepercent),
                   xend = 0), col="gray40") +
    geom_point()
```




## Exploring the correlation structure

Now that we have explored thed ataset a little, we will see how the variables interact with one another.

First we will use correlation and view the results with the **corrplot** package to plot a correlation matrix.

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

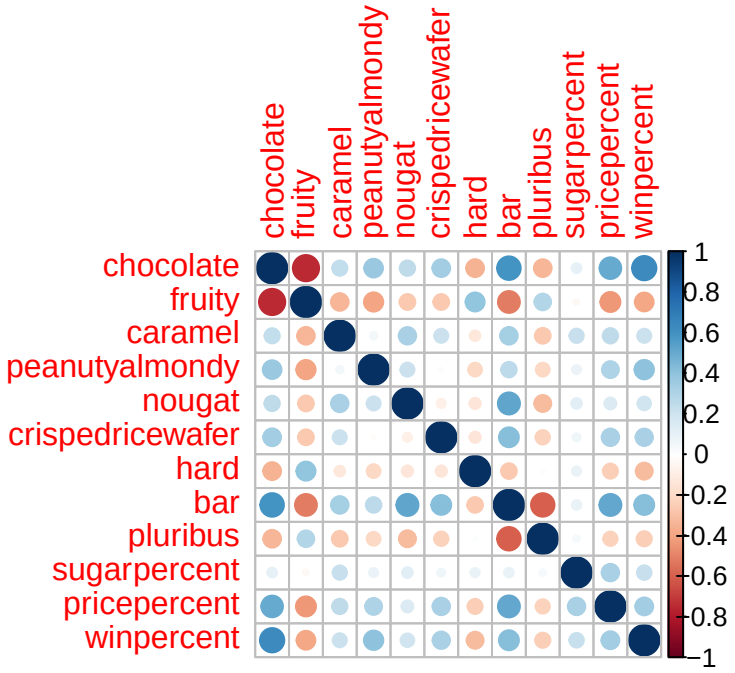
```
                  chocolate      fruity     caramel peanutyalmondy      nougat
chocolate         1.0000000 -0.74172106  0.24987535     0.37782357  0.25489183
fruity           -0.7417211  1.00000000 -0.33548538    -0.39928014 -0.26936712
caramel           0.2498753 -0.33548538  1.00000000     0.05935614  0.32849280
peanutyalmondy    0.3778236 -0.39928014  0.05935614     1.00000000  0.21311310
nougat            0.2548918 -0.26936712  0.32849280     0.21311310  1.00000000
crispedricewafer  0.3412098 -0.26936712  0.21311310    -0.01764631 -0.08974359
hard             -0.3441769  0.39067750 -0.12235513    -0.20555661 -0.13867505
bar               0.5974211 -0.51506558  0.33396002     0.26041960  0.52297636
pluribus         -0.3396752  0.29972522 -0.26958501    -0.20610932 -0.31033884
sugarpercent      0.1041691 -0.03439296  0.22193335     0.08788927  0.12308135
pricepercent      0.5046754 -0.43096853  0.25432709     0.30915323  0.15319643
winpercent        0.6365167 -0.38093814  0.21341630     0.40619220  0.19937530
                  crispedricewafer        hard         bar    pluribus
chocolate              0.34120978 -0.34417691  0.59742114 -0.33967519
fruity                -0.26936712  0.39067750 -0.51506558  0.29972522
caramel                0.21311310 -0.12235513  0.33396002 -0.26958501
peanutyalmondy        -0.01764631 -0.20555661  0.26041960 -0.20610932
nougat                -0.08974359 -0.13867505  0.52297636 -0.31033884
crispedricewafer       1.00000000 -0.13867505  0.42375093 -0.22469338
hard                  -0.13867505  1.00000000 -0.26516504  0.01453172
bar                    0.42375093 -0.26516504  1.00000000 -0.59340892
pluribus              -0.22469338  0.01453172 -0.59340892  1.00000000
sugarpercent           0.06994969  0.09180975  0.09998516  0.04552282
pricepercent           0.32826539 -0.24436534  0.51840654 -0.22079363
winpercent             0.32467965 -0.31038158  0.42992933 -0.24744787
                  sugarpercent pricepercent winpercent
chocolate           0.10416906    0.5046754  0.6365167
fruity             -0.03439296   -0.4309685 -0.3809381
caramel             0.22193335    0.2543271  0.2134163
peanutyalmondy      0.08788927    0.3091532  0.4061922
nougat              0.12308135    0.1531964  0.1993753
crispedricewafer    0.06994969    0.3282654  0.3246797
hard                0.09180975   -0.2443653 -0.3103816
bar                 0.09998516    0.5184065  0.4299293
pluribus            0.04552282   -0.2207936 -0.2474479
sugarpercent        1.00000000    0.3297064  0.2291507
pricepercent        0.32970639    1.0000000  0.3453254
winpercent          0.22915066    0.3453254  1.0000000
```

To visually view the correlation between the variables, use `corrplot`.

```
library (corrplot)
```

```
corrplot 0.95 loaded
```

```
corrplot(cij)
```




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

The chocolate and fruity variables are anti-correlated. This tells us that chocolate candies often don't have a fruity component to them.

Q23. Similarly, what two variables are most positively correlated?

The chocolate and winpercent variables are most positively correlated. This tells us that chocolate candies are the mainly favored type of candy.

## 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
```

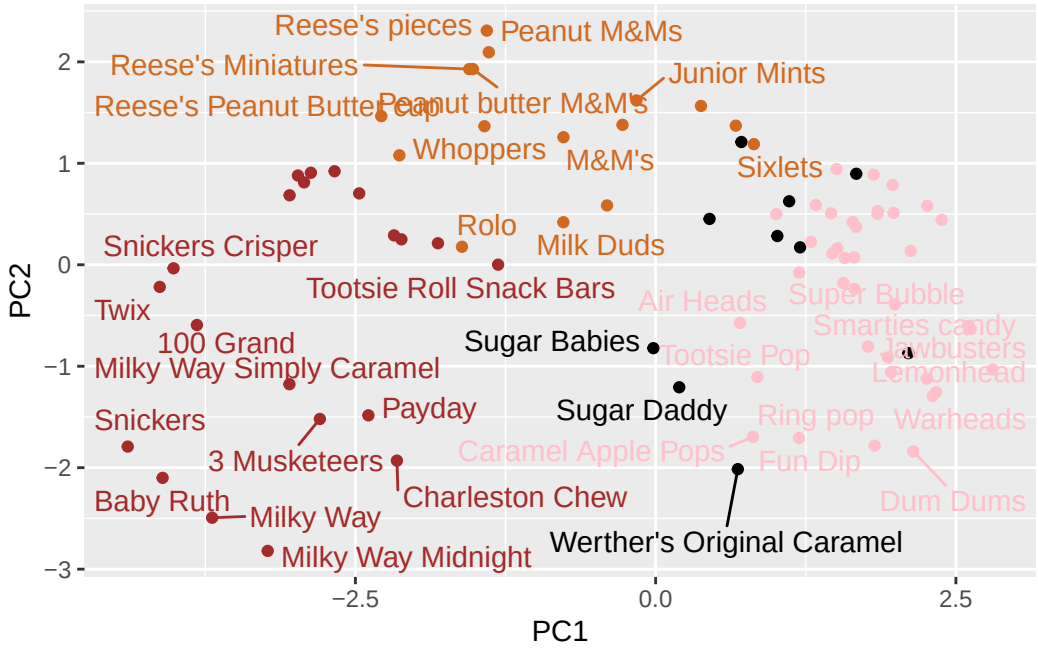
```
attributes(pca)
```

```
$names
[1] "sdev"     "rotation" "center"   "scale"    "x"

$class
[1] "prcomp"
```

Let's plot our main results as our PCA "score plot".

```
ggplot(pca$x) +
  aes(PC1, PC2, label=rownames(pca$x)) +
  geom_point(col=my_cols) +
  geom_text_repel(col=my_cols)
```

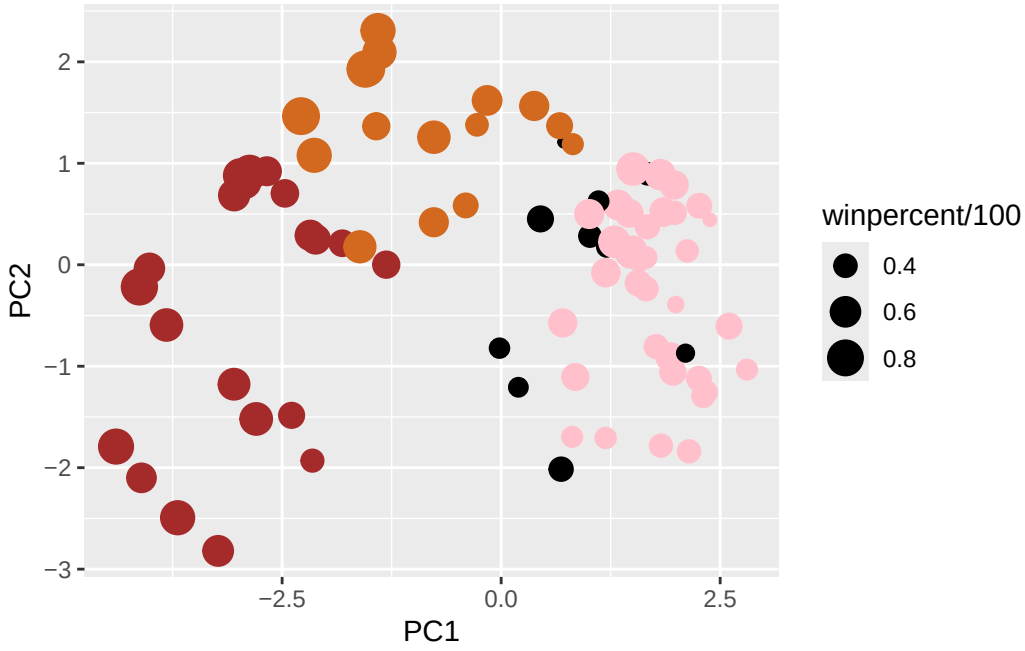
```
Warning: ggrepel: 48 unlabeled data points (too many overlaps). Consider
increasing max.overlaps
```




The plot shows us that there is a separation between the chocolate, bar, and fruity type candies. It also suggests that if an individual likes Peanut M&M's, there's a chance they also like Reese's Pieces.

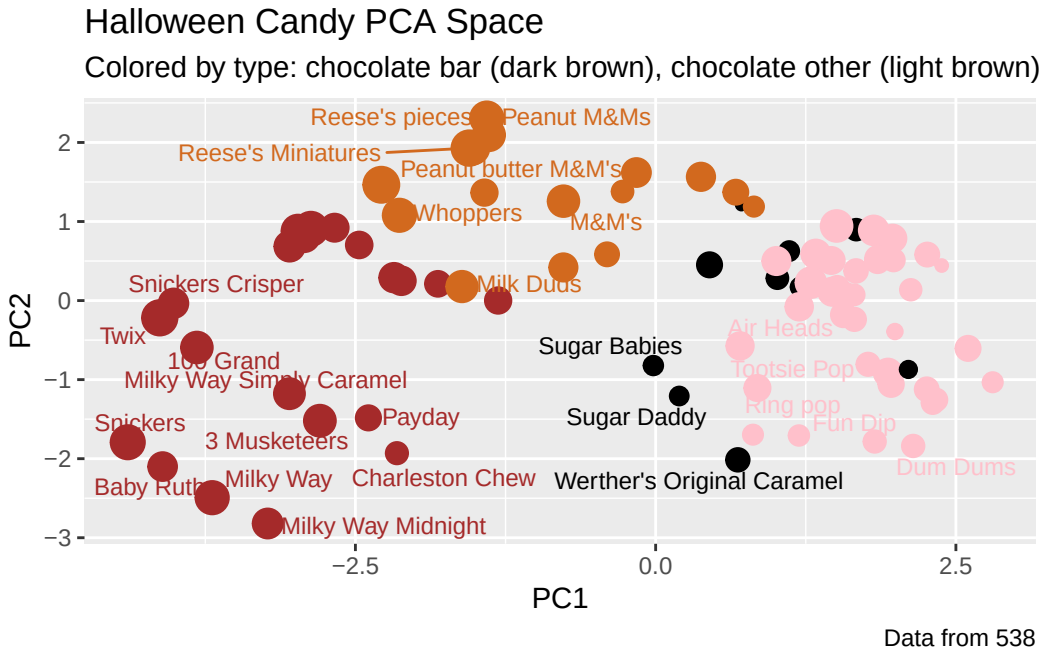
```
# Make a new data-frame with our PCA results and candy data
my_data <- cbind(candy, pca$x[,1:3])
```

```
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
```




```
library(ggrepel)

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
```




Use `plotly` to generate an interactive plot so we can mouse over to see labels instead of having to change the max.overlaps value.

**Note the plot was made in Rstudio, but since an interactive plot can't be generated in pdf format it was removed from the Quarto document.

Finally, let's look at how the original variables contribute to the PCs. Let's start with PC1.

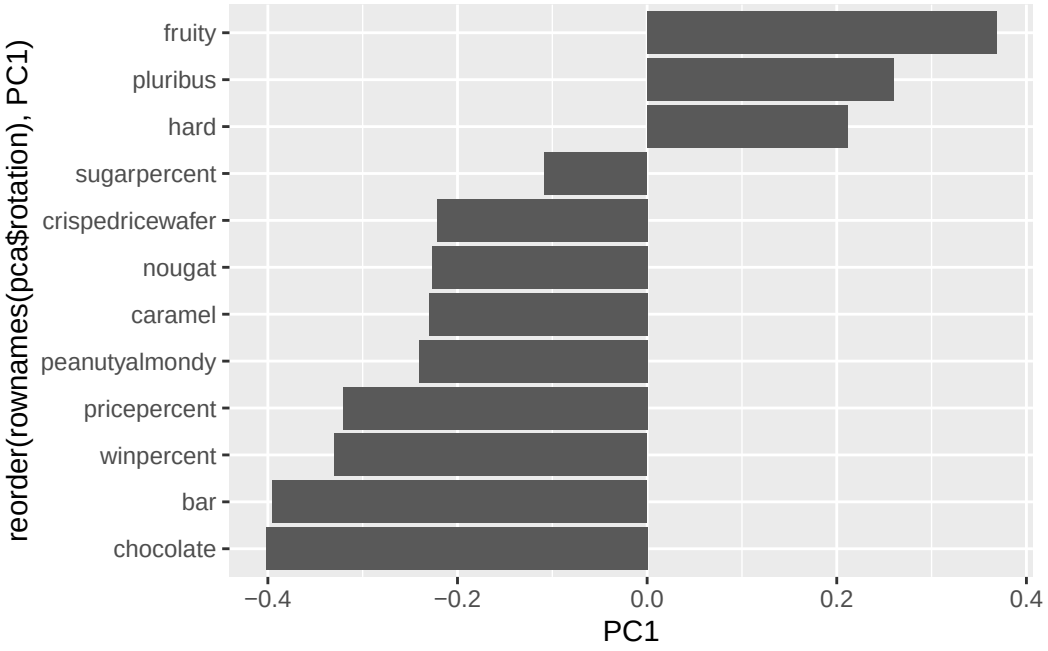
```
pca$rotation
```

```
                     PC1         PC2         PC3         PC4          PC5
chocolate      -0.4019466  0.21404160  0.01601358 -0.016673032  0.066035846
fruity          0.3683883 -0.18304666 -0.13765612 -0.004479829  0.143535325
caramel        -0.2299709 -0.40349894 -0.13294166 -0.024889542 -0.507301501
peanutyalmondy -0.2407155  0.22446919  0.18272802  0.466784287  0.399930245
nougat         -0.2268102 -0.47016599  0.33970244  0.299581403 -0.188852418
crispedricewafer -0.2215182  0.09719527 -0.36485542 -0.605594730  0.034652316
hard            0.2111587 -0.43262603 -0.20295368 -0.032249660  0.574557816
bar            -0.3947433 -0.22255618  0.10696092 -0.186914549  0.077794806
pluribus        0.2600041  0.36920922 -0.26813772  0.287246604 -0.392796479
sugarpercent   -0.1083088 -0.23647379 -0.65509692  0.433896248  0.007469103
pricepercent   -0.3207361  0.05883628 -0.33048843  0.063557149  0.043358887
winpercent     -0.3298035  0.21115347 -0.13531766  0.117930997  0.168755073
                     PC6         PC7         PC8         PC9          PC10
```

```
chocolate        -0.09018950 -0.08360642 -0.49084856 -0.151651568  0.107661356
fruity           -0.04266105  0.46147889  0.39805802 -0.001248306  0.362062502
caramel          -0.40346502 -0.44274741  0.26963447  0.019186442  0.229799010
peanutyalmondy   -0.09416259 -0.25710489  0.45771445  0.381068550 -0.145912362
nougat            0.09012643  0.36663902 -0.18793955  0.385278987  0.011323453
crispedricewafer -0.09007640  0.13077042  0.13567736  0.511634999 -0.264810144
hard             -0.12767365 -0.31933477 -0.38881683  0.258154433  0.220779142
bar               0.25307332  0.24192992 -0.02982691  0.091872886 -0.003232321
pluribus          0.03184932  0.04066352 -0.28652547  0.529954405  0.199303452
sugarpercent      0.02737834  0.14721840 -0.04114076 -0.217685759 -0.488103337
pricepercent      0.62908570 -0.14308215  0.16722078 -0.048991557  0.507716043
winpercent       -0.56947283  0.40260385 -0.02936405 -0.124440117  0.358431235
                        PC11        PC12
chocolate         0.10045278  0.69784924
fruity            0.17494902  0.50624242
caramel           0.13515820  0.07548984
peanutyalmondy    0.11244275  0.12972756
nougat           -0.38954473  0.09223698
crispedricewafer -0.22615618  0.11727369
hard              0.01342330 -0.10430092
bar               0.74956878 -0.22010569
pluribus          0.27971527 -0.06169246
sugarpercent      0.05373286  0.04733985
pricepercent     -0.26396582 -0.06698291
winpercent       -0.11251626 -0.37693153
```

This tells us how much each of the columns in the dataset contribute to the overall analysis and information gathered from the dataset.

```
ggplot(pca$rotation) +
  aes(PC1, reorder(rownames(pca$rotation), PC1)) +
  geom_col()
```




Q24. What original variables are picked up strongly by PC1 in the positive direction? Do these make sense to you?

The fruity component contributes to the positive direction while the chocolate components contributes to the negative direction. This makes sense because this data aligns with the information provided on the PC2 vs. PC1 plot. The characteristics generally associated with a fruity candy, which are pluribus (comes in a bag or multiple boxes of candy) and hard, are also contributing to the positive direction in the PC1 plot while the characteristics of a chocolate candy are also contributing to the negative direction in the PC1 plot.