class 10: Halloween Candy

Yinuo Song

1. Importing candy data

```r
candy_file <- "candy-data.csv"
candy = read.csv("candy-data.csv", 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?

```r
nrow(candy)
```

```
[1] 85
```

85 candy types.

Q2 How many fruity candy types are in the dataset?

```r
table(candy$fruity)
```

```
 0  1 
47 38 
```

38 fruity candy types are in the dataset.

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:

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

```r
candy["Haribo Gold Bears",]$winpercent
```

```
[1] 57.11974
```

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

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

```
[1] 76.7686
```

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

```r
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)
```

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 hist column looks like diagrams.

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

1 means chocolate is used in this candy type.0 means chocolate is not used in this candy type.

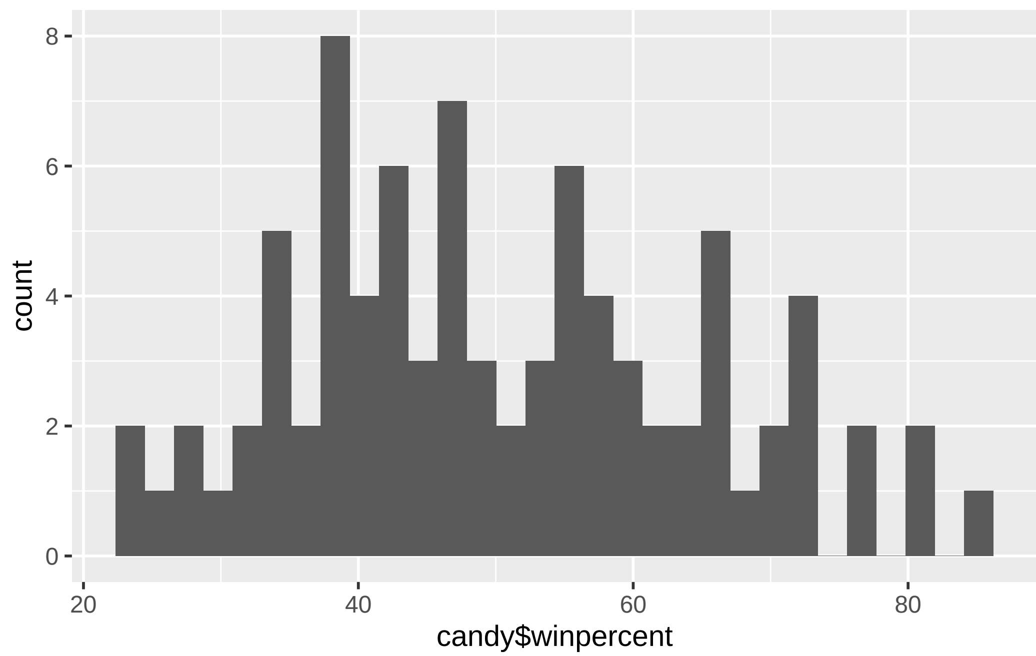
A good place to start any exploratory analysis is with a histogram. You can do this most easily with the base R function `hist()`. Alternatively, you can use `ggplot()` with `geom_hist()`. Either works well in this case and (as always) its your choice.

```
library(ggplot2)
```

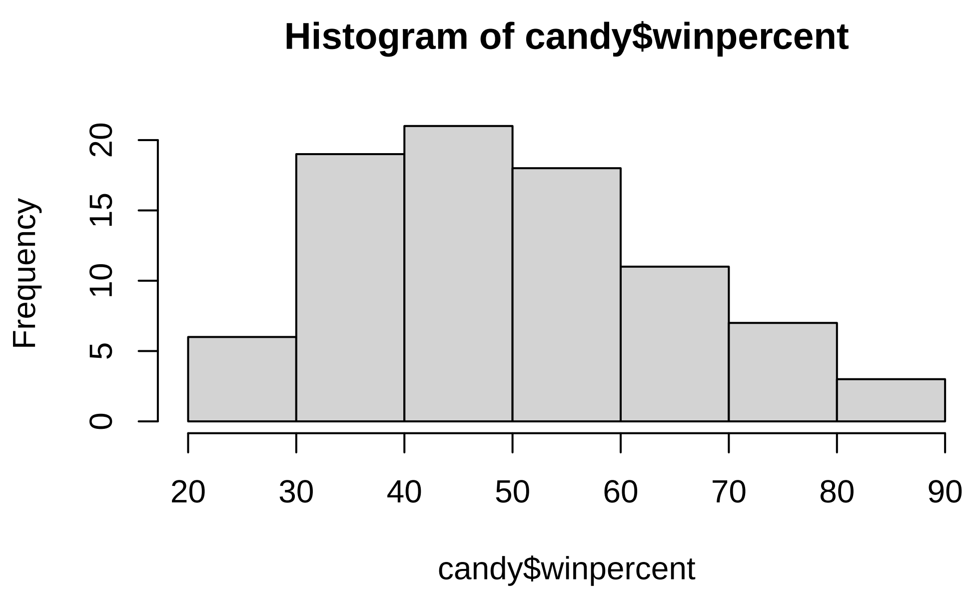
Q8 Plot a histogram of `winpercent` values

```
ggplot(candy,aes(candy$winpercent))+geom_histogram()
```

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




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



Q9 Is the distribution of `winpercent` values symmetrical?

No

Q10 Is the center of the distribution above or below 50%?

below 50%

We want to conpare chocolated and fruity candy

```
chocaolate_candy=candy$winpercetnt[as.logical(candy$chocolate)]
table
```

```
function (..., exclude = if (useNA == "no") c(NA, NaN), useNA = c("no",
    "ifany", "always"), dnn = list.names(...), deparse.level = 1)
{
    list.names <- function(...) {
        l <- as.list(substitute(list(...)))[-1L]
        if (length(l) == 1L && is.list(..1) && !is.null(nm <- names(..1)))
            return(nm)
        nm <- names(l)
        fixup <- if (is.null(nm))
            seq_along(l)
        else nm == ""
        dep <- vapply(l[fixup], function(x) switch(deparse.level +
            1, "", if (is.symbol(x)) as.character(x) else "",
            deparse(x, nlines = 1)[1L]), "")
        if (is.null(nm))
            dep
        else {
            nm[fixup] <- dep
            nm
        }
    }
    miss.use <- missing(useNA)
    miss.exc <- missing(exclude)
    useNA <- if (miss.use && !miss.exc && !match(NA, exclude,
        nomatch = 0L))
        "ifany"
    else match.arg(useNA)
    doNA <- useNA != "no"
    if (!miss.use && !miss.exc && doNA && match(NA, exclude,
        nomatch = 0L))
        warning("'exclude' containing NA and 'useNA' != \"no\"' are a bit contradicting")
```

```
args <- list(...)
if (length(args) == 1L && is.list(args[[1L]])) {
    args <- args[[1L]]
    if (length(dnn) != length(args))
        dnn <- paste(dnn[1L], seq_along(args), sep = ".")
}
if (!length(args))
    stop("nothing to tabulate")
bin <- 0L
lens <- NULL
dims <- integer()
pd <- 1L
dn <- NULL
for (a in args) {
    if (is.null(lens))
        lens <- length(a)
    else if (length(a) != lens)
        stop("all arguments must have the same length")
    fact.a <- is.factor(a)
    if (doNA)
        aNA <- anyNA(a)
    if (!fact.a) {
        a0 <- a
        op <- options(warn = 2)
        a <- factor(a, exclude = exclude)
        options(op)
    }
    add.na <- doNA
    if (add.na) {
        ifany <- (useNA == "ifany")
        anNAc <- anyNA(a)
        add.na <- if (!ifany || anNAc) {
            ll <- levels(a)
            if (add.ll <- !anyNA(ll)) {
              ll <- c(ll, NA)
              TRUE
            }
            else if (!ifany && !anNAc)
              FALSE
            else TRUE
        }
        else FALSE
    }
```

```
        if (add.na)
            a <- factor(a, levels = ll, exclude = NULL)
        else ll <- levels(a)
        a <- as.integer(a)
        if (fact.a && !miss.exc) {
            ll <- ll[keep <- which(match(ll, exclude, nomatch = 0L) ==
                0L)]
            a <- match(a, keep)
        }
        else if (!fact.a && add.na) {
            if (ifany && !aNA && add.ll) {
                ll <- ll[!is.na(ll)]
                is.na(a) <- match(a0, c(exclude, NA), nomatch = 0L) >
                  0L
            }
            else {
                is.na(a) <- match(a0, exclude, nomatch = 0L) >
                  0L
            }
        }
        nl <- length(ll)
        dims <- c(dims, nl)
        if (prod(dims) > .Machine$integer.max)
            stop("attempt to make a table with >= 2^31 elements")
        dn <- c(dn, list(ll))
        bin <- bin + pd * (a - 1L)
        pd <- pd * nl
    }
    names(dn) <- dnn
    bin <- bin[!is.na(bin)]
    if (length(bin))
        bin <- bin + 1L
    y <- array(tabulate(bin, pd), dims, dimnames = dn)
    class(y) <- "table"
    y
}
<bytecode: 0x7fb2d61653e0>
<environment: namespace:base>
```

```
winpercent_chocolate <- candy$winpercent[as.logical(candy$chocolate)]
```

```r
mean(winpercent_chocolate)
```

```
[1] 60.92153
```

```r
winpercent_fruity <- candy$winpercent[as.logical(candy$fruity)]
mean(winpercent_fruity)
```

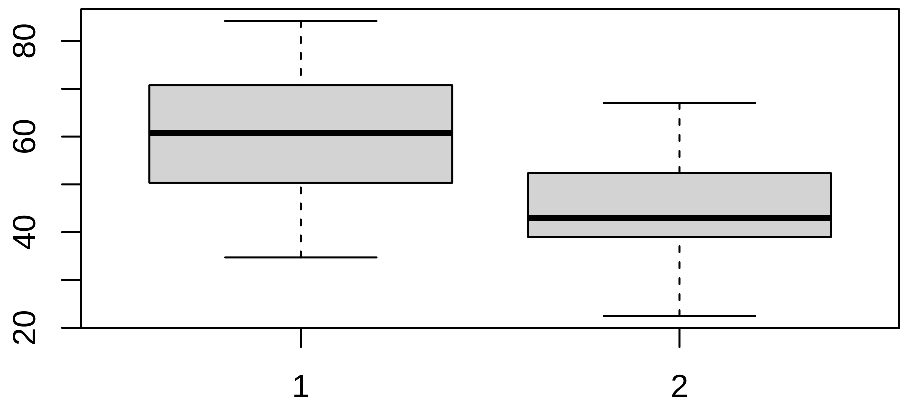
```
[1] 44.11974
```

Statistical test

```r
t.test(winpercent_chocolate,winpercent_fruity)
```

```

    Welch Two Sample t-test

data:  winpercent_chocolate and winpercent_fruity
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 
```

```r
boxplot(winpercent_chocolate,winpercent_fruity)
```



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

chocolate candy is higher ranked than fruit candy on average.

Q12 Is this difference statistically significant?

Yes, with a p-value <0.05.

3. Overall Candy Rankings

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

```
head(candy[order(candy$winpercent),], 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
```

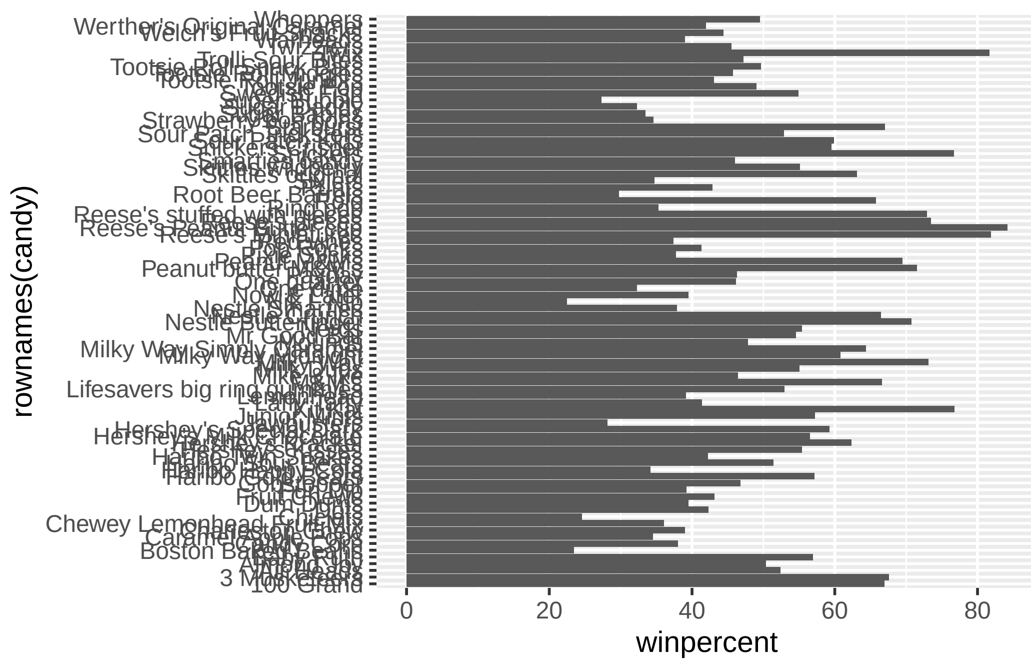
What are the top 5 all time favorite candy types out of this set?

```
head(candy[order(candy$winpercent,decreasing=TRUE),], 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
```

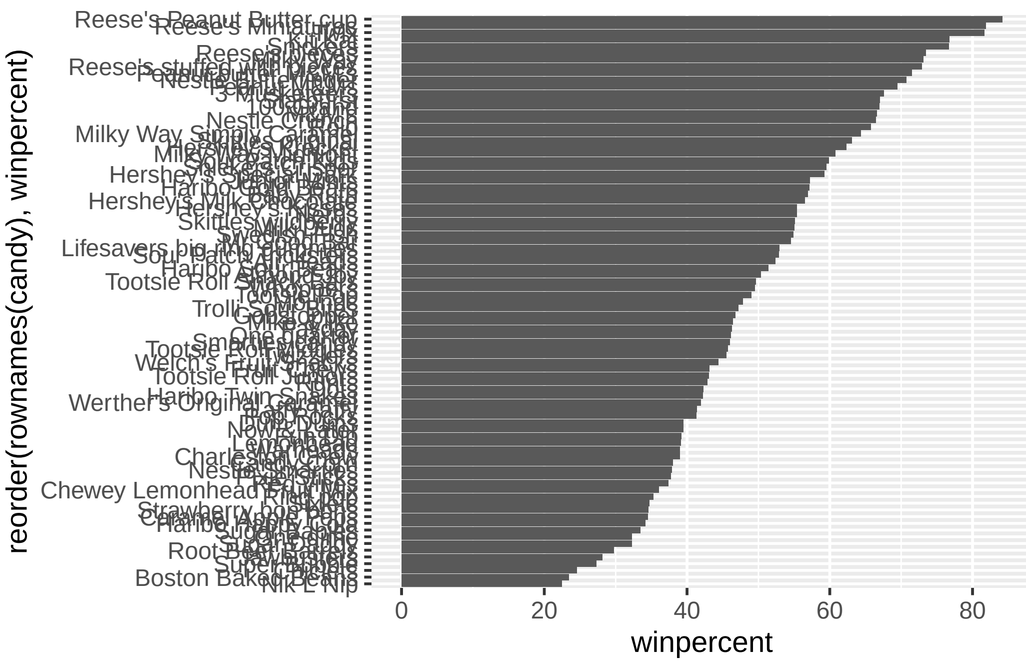
Q15 Make a first barplot of candy ranking based on `winpercent` values.

```
library(ggplot2)
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()
```

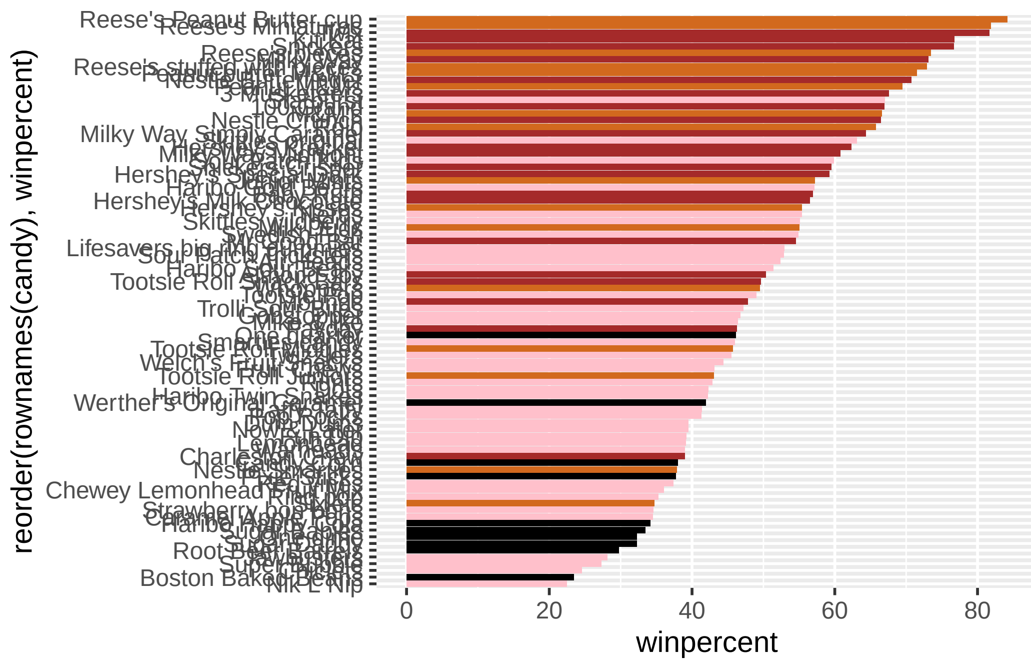


Let's setup a color vector (that signifies candy type) that we can then use for some future plots. We start by making a vector of all black values (one for each candy). Then we overwrite chocolate (for chocolate candy), brown (for candy bars) and red (for fruity candy) values.

```
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)
```



Q17 What is the worst ranked chocolate candy?

Reese's Peanut Butter cup

Q18 What is the best ranked fruity candy?

Starbust

4. Taking a look at pricepercent

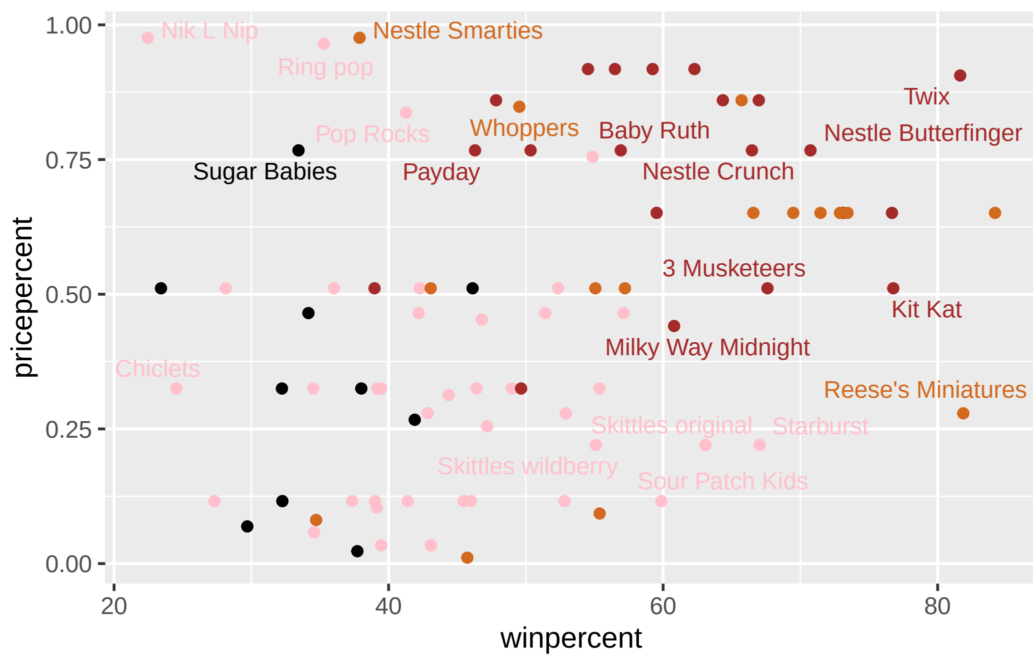
The `pricepercent` variable records the percentile rank of the candy's price against all the other candies in the dataset. Lower vales are less expensive and high values more expensive.

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.

```
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?

Reese's Miniatures

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
```

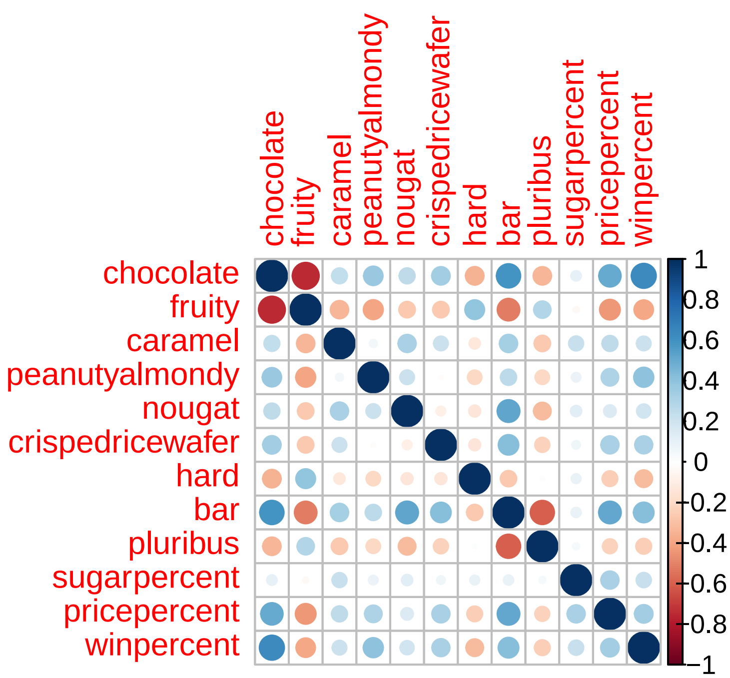
5 Exploring the correlation structure

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.

```
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)?

A red circle means two variables are anti-correlated.

Q23 Similarly, what two variables are most positively correlated?

"bar" and "chocolate".

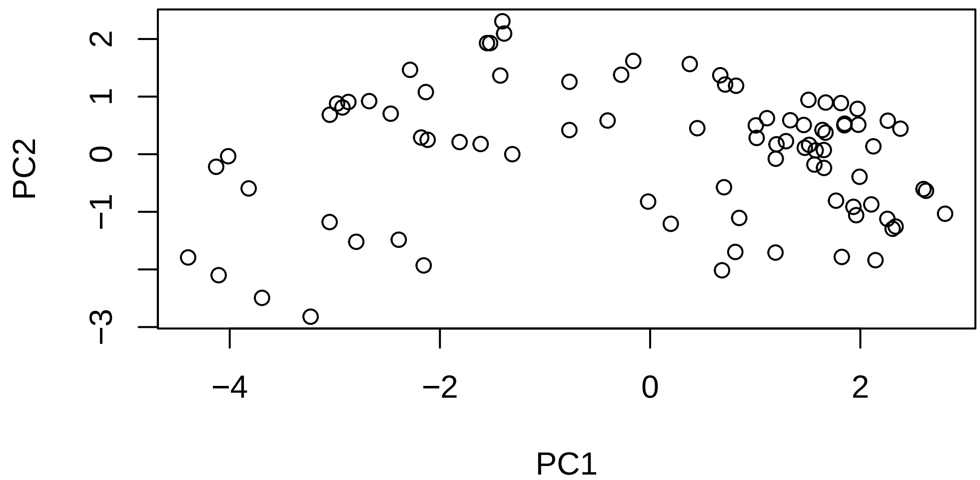
6. Principal Component Analysis

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)
```

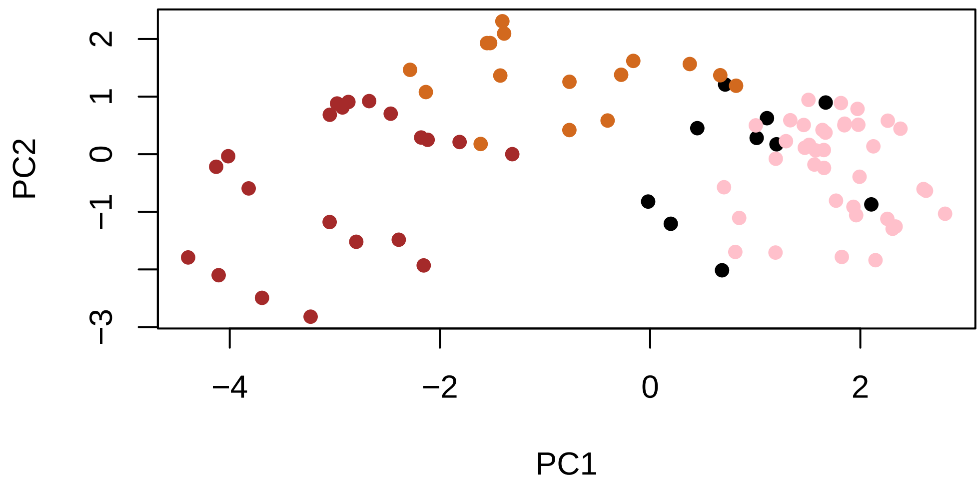
```
Importance of components:
                          PC1    PC2    PC3     PC4    PC5     PC6     PC7
Standard deviation     2.0788 1.1378 1.1092 1.07533 0.9518 0.81923 0.81530
Proportion of Variance 0.3601 0.1079 0.1025 0.09636 0.0755 0.05593 0.05539
Cumulative Proportion  0.3601 0.4680 0.5705 0.66688 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
```

```
plot(pca$x[,1:2])
```



We can change the plotting character and add some color:

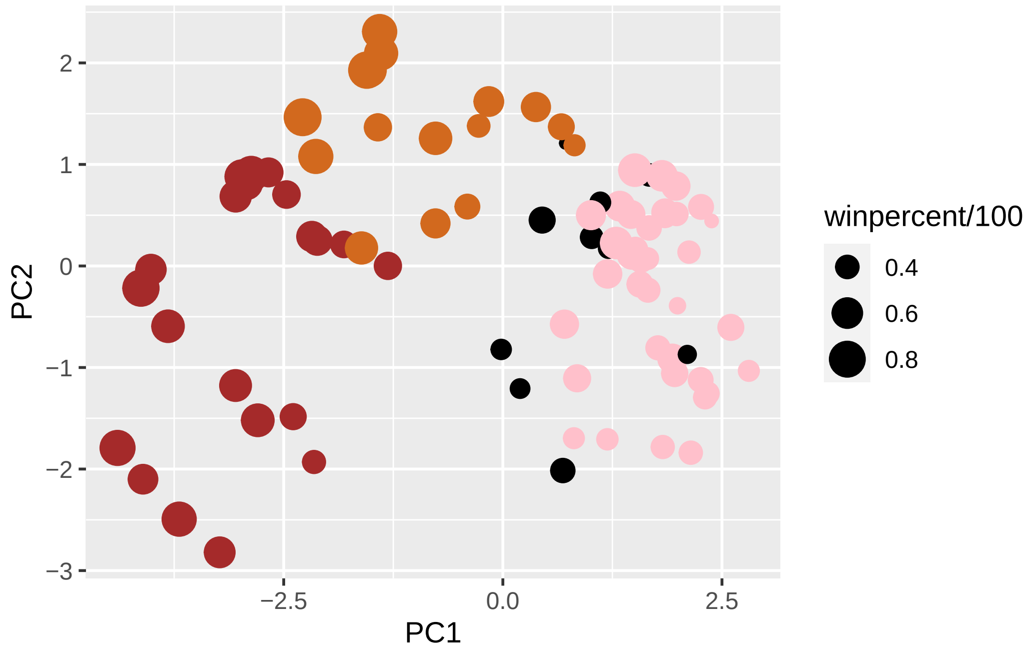
```
plot(pca$x[,1:2], col=my_cols, pch=16)
```



we make a new data.frame here that contains our PCA results with all the rest of our candy data.

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

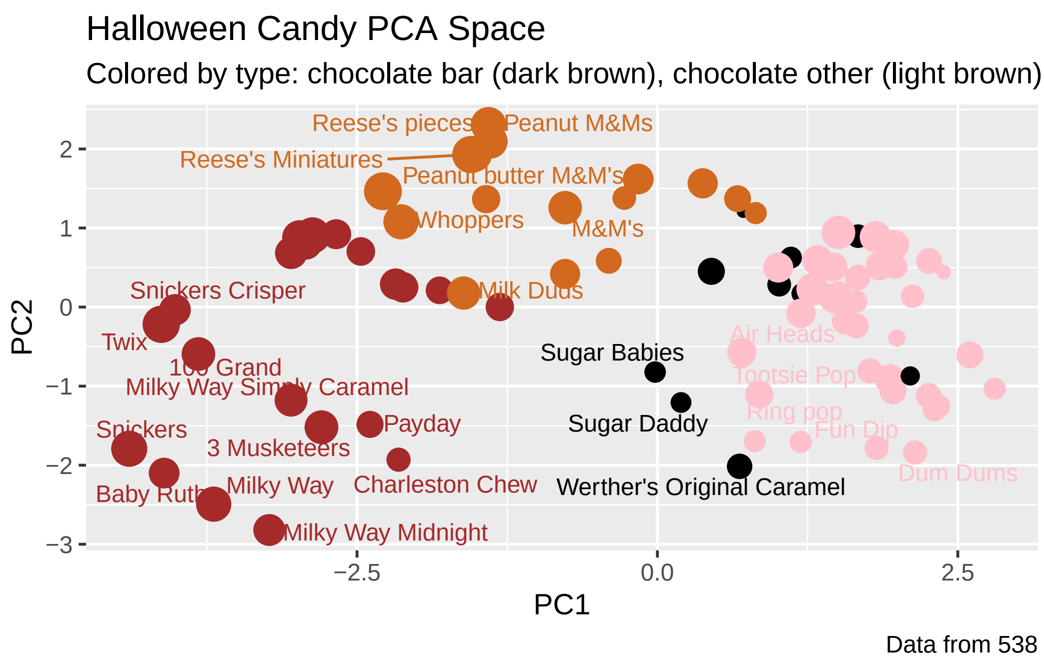
```
p <- ggplot(my_data) +
        aes(x=PC1, y=PC2,
            size=winpercent/100,
            text=rownames(my_data),
            label=rownames(my_data)) +
        geom_point(col=my_cols)

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:

```
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
```



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", repos = "http://cran.us.r-project.org")
```

```
The downloaded binary packages are in
	/var/folders/8z/wdc4st_x2lb2j29hz7q7jbnh0000gn/T//RtmpUa1hCp/downloaded_packages
```

```
library(plotly)
```

```
Attaching package: 'plotly'
```

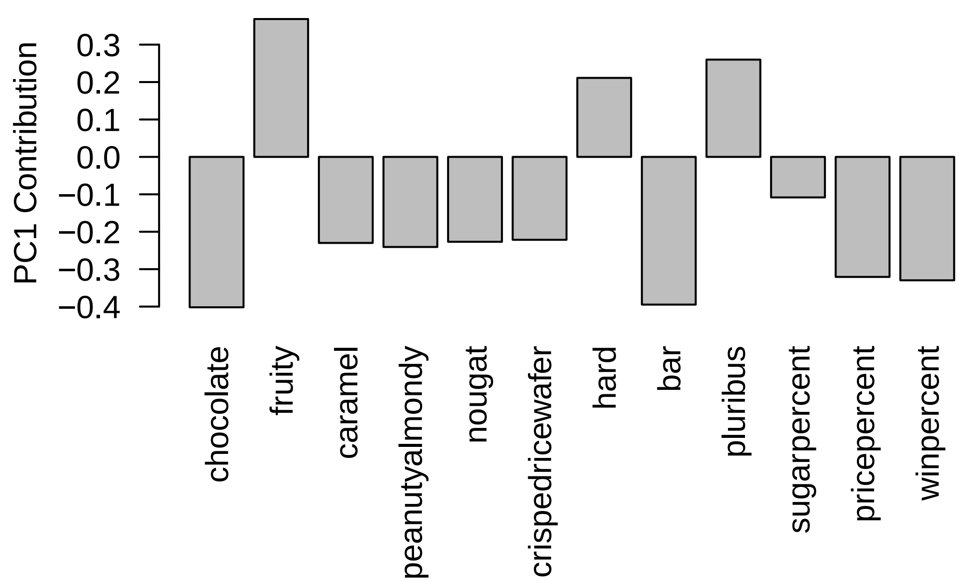
```
The following object is masked from 'package:ggplot2':

    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?

"fruity","hard" and "pluribus".

```r
pca$rotation[,1]
```

```
      chocolate          fruity         caramel   peanutyalmondy
     -0.4019466       0.3683883      -0.2299709       -0.2407155
         nougat crispedricewafer          hard              bar
     -0.2268102      -0.2215182       0.2111587       -0.3947433
       pluribus     sugarpercent    pricepercent       winpercent
      0.2600041      -0.1083088      -0.3207361       -0.3298035
```

They make sense. There is positive correlation between "fruity" and "hard". And there are positive correlation between "fruity" and "pluribus".