# Module 3 - Assignment 1

## Pierce, Michaela

### Data Visualization

In this assignment, I will be using datasets containing candy rankings and production. These datasets can be found on the Canvas course website.

library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.4 ✔ readr 2.1.5  
## ✔ forcats 1.0.0 ✔ stringr 1.5.1  
## ✔ ggplot2 3.5.0 ✔ tibble 3.2.1  
## ✔ lubridate 1.9.3 ✔ tidyr 1.3.1  
## ✔ purrr 1.0.2   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

candy\_data <- read\_csv("candy\_data.csv")

## Rows: 85 Columns: 13  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## chr (1): competitorname  
## dbl (12): chocolate, fruity, caramel, peanutyalmondy, nougat, crispedricewaf...  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

candy\_production <- read\_csv("candy\_production.csv")

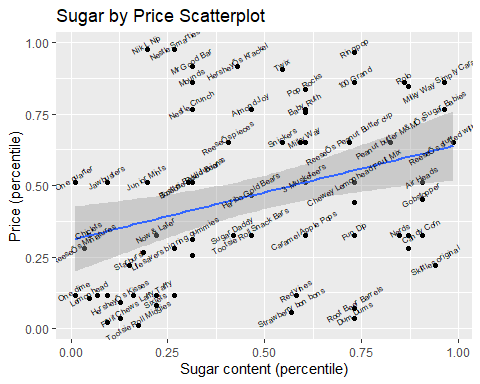
## Rows: 548 Columns: 2  
## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## dbl (1): IPG3113N  
## date (1): observation\_date  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

#### **Visualization with Scatterplots (geom\_point)**

# Create scatterplot (geom\_point)  
ggplot(candy\_data, aes(x = sugarpercent, label= competitorname, y = pricepercent)) +  
 geom\_point()+  
 geom\_smooth(method = "lm") + # adds a fitted line   
   
labs(title = "Sugar by Price Scatterplot", # plot title   
 x = "Sugar content (percentile)", # x axis label   
 y = "Price (percentile)" # y axis label   
) +  
  
geom\_text(check\_overlap = T,# automatically reduce overlap (deletes some labels)   
 vjust = "bottom", # adjust the vertical orientation   
 nudge\_y = 0.01, # move the text up a bit so it doesn't touch the points   
 angle = 30,# tilt the text 30 degrees   
 size = 2 # make the text smaller (to reduce overlap more)   
) # and then add labels to the points

## `geom\_smooth()` using formula = 'y ~ x'

## Warning: The following aesthetics were dropped during statistical transformation: label.  
## ℹ This can happen when ggplot fails to infer the correct grouping structure in  
## the data.  
## ℹ Did you forget to specify a `group` aesthetic or to convert a numerical  
## variable into a factor?



The candy with the most sugar and the lowest price is ReeseÕs stuffed with pieces with a price of 0.651 % and a sugar content of 0.988 %. The most expensive candy with the highest sugar content is 100 Grand with a price of 0.86 % and a sugar content of 0.732 %.

Candy with the most sugar: ReeseÕs stuffed with pieces  
Candy with the least sugar: One dime  
Candy with the lowest price: Tootsie Roll Midgies  
Candy with the highest price: Nik L Nip

I wanted to get the exact values so it would be accurate. All the sources and code I used are below.

* Learned how to use cat to print, which converts and prints all arguments as a character string.  
  <https://www.geeksforgeeks.org/printing-output-of-an-r-program/>
* Learned how to print a new line in a string. Found this is similar/same as printing new lines in Python.  
  <https://www.geeksforgeeks.org/r-program-to-print-a-new-line-in-string/>
* For Max / Refresher/Guide  
  <https://www.geeksforgeeks.org/return-the-index-of-the-first-maximum-value-of-a-numeric-vector-in-r-programming-which-max-function/>
* For Min / Refresher/ Guide  
  <https://www.geeksforgeeks.org/return-the-index-of-the-first-minimum-value-of-a-numeric-vector-in-r-programming-which-min-function/>

# Find the index of the candy with the highest price  
max\_price\_index <- which.max(candy\_data$pricepercent)  
  
# Find the index of the candy with the highest sugar content  
max\_sugar\_index <- which.max(candy\_data$sugarpercent)  
  
# Find the index of the candy with the highest price among candies with the highest sugar content  
max\_price\_highest\_sugar\_index <- which.max(candy\_data$pricepercent[candy\_data$sugarpercent == candy\_data$sugarpercent[max\_sugar\_index]])  
  
# Print the candy with the highest price and the most sugar  
cat("Candy with the highest price and most sugar:", candy\_data$competitorname[max\_price\_highest\_sugar\_index], "\n")

## Candy with the highest price and most sugar: 100 Grand

cat("Price:", candy\_data$pricepercent[max\_price\_highest\_sugar\_index], "%\n")

## Price: 0.86 %

cat("Sugar content:", candy\_data$sugarpercent[max\_price\_highest\_sugar\_index], "%\n")

## Sugar content: 0.732 %

# Print the candy with the most sugar and lowest price  
cat("Candy with the most sugar and lowest price:", candy\_data$competitorname[max\_sugar\_index], "\n")

## Candy with the most sugar and lowest price: ReeseÕs stuffed with pieces

cat("Price:", candy\_data$pricepercent[max\_sugar\_index], "%\n")

## Price: 0.651 %

cat("Sugar content:", candy\_data$sugarpercent[max\_sugar\_index], "%\n")

## Sugar content: 0.988 %

# Find the index of the candy with the most sugar  
most\_sugar\_index <- which.max(candy\_data$sugarpercent)  
# Find the index of the candy with the least sugar  
least\_sugar\_index <- which.min(candy\_data$sugarpercent)  
# Find the index of the candy with the lowest price  
lowest\_price\_index <- which.min(candy\_data$pricepercent)  
# Find the index of the candy with the highest price  
highest\_price\_index <- which.max(candy\_data$pricepercent)  
  
# Print the candies with the most and least sugar  
cat("Candy with the most sugar:", candy\_data$competitorname[most\_sugar\_index], "\n")

## Candy with the most sugar: ReeseÕs stuffed with pieces

cat("Candy with the least sugar:", candy\_data$competitorname[least\_sugar\_index], "\n")

## Candy with the least sugar: One dime

# Print the candies with the lowest and highest price  
cat("Candy with the lowest price:", candy\_data$competitorname[lowest\_price\_index], "\n")

## Candy with the lowest price: Tootsie Roll Midgies

cat("Candy with the highest price:", candy\_data$competitorname[highest\_price\_index], "\n")

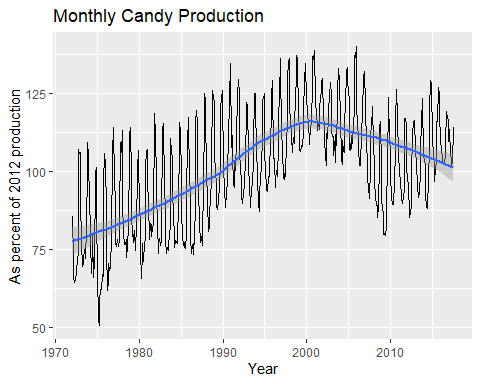
## Candy with the highest price: Nik L Nip

#### **Line Chart of Candy Production**

I am using the candy production dataset that will display a specific date and how production during that month is compared to 2012.

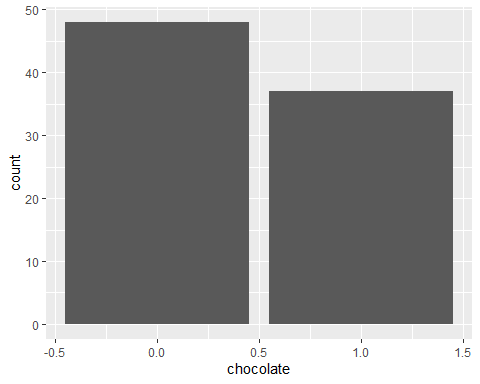
# Create line chart with fitted line  
ggplot(candy\_production, aes(x = observation\_date, y = IPG3113N)) +  
 geom\_line() +  
 geom\_smooth() +  
 labs(title = "Monthly Candy Production",  
 x = "Year",  
 y = "As percent of 2012 production")

## `geom\_smooth()` using method = 'loess' and formula = 'y ~ x'



#### **Bar Chart of Ingredients**

# Create bar chart for chocolate variable  
ggplot(candy\_data, aes(x = chocolate)) +  
 geom\_bar()



# select out the columns that have the features of the candy (chocolate, caramel, etc.)   
candyFeatures <- candy\_data %>% select(2:10)   
# make sure that these are booleans (logical)   
candyFeatures[] <- lapply(candyFeatures, as.logical)  
  
  
# Create bar chart for candyFeatures dataset  
ggplot(candyFeatures, aes(x = chocolate)) +  
 geom\_bar()

