# HW8 Key

2022-11-08

## HW8

**Question 1 (from 2020 midterm)**

Using a candy dataset (https://math.montana.edu/ahoegh/teaching/stat446/candy-data.csv), define and fit a regression model to understand the relationship between `winpercent` and `pricepercent, chocolate,` and `caramel`. More insight into the data is available at https://fivethirtyeight.com/videos/the-ultimate-halloween-candy-power-ranking/.

a. Write out the model and define all of the coefficients. (2 points)

$$win_i = \beta_0 + \beta_1 x_{i,I(chocolate)} + \beta_2 x_{i,I(caramel)} + \beta_3 x_{i,I(chocolate)} x_{i,I(caramel)} + \epsilon_i, \text{where } \epsilon_i \sim N(0, \sigma^2),$$

where $win_i$ is the winning percentage, $x_{i,I(chocolate)}$ and $x_{i,I(caramel)}$ are indicator variables for candy with chocolate and caramel, respectively. Then,

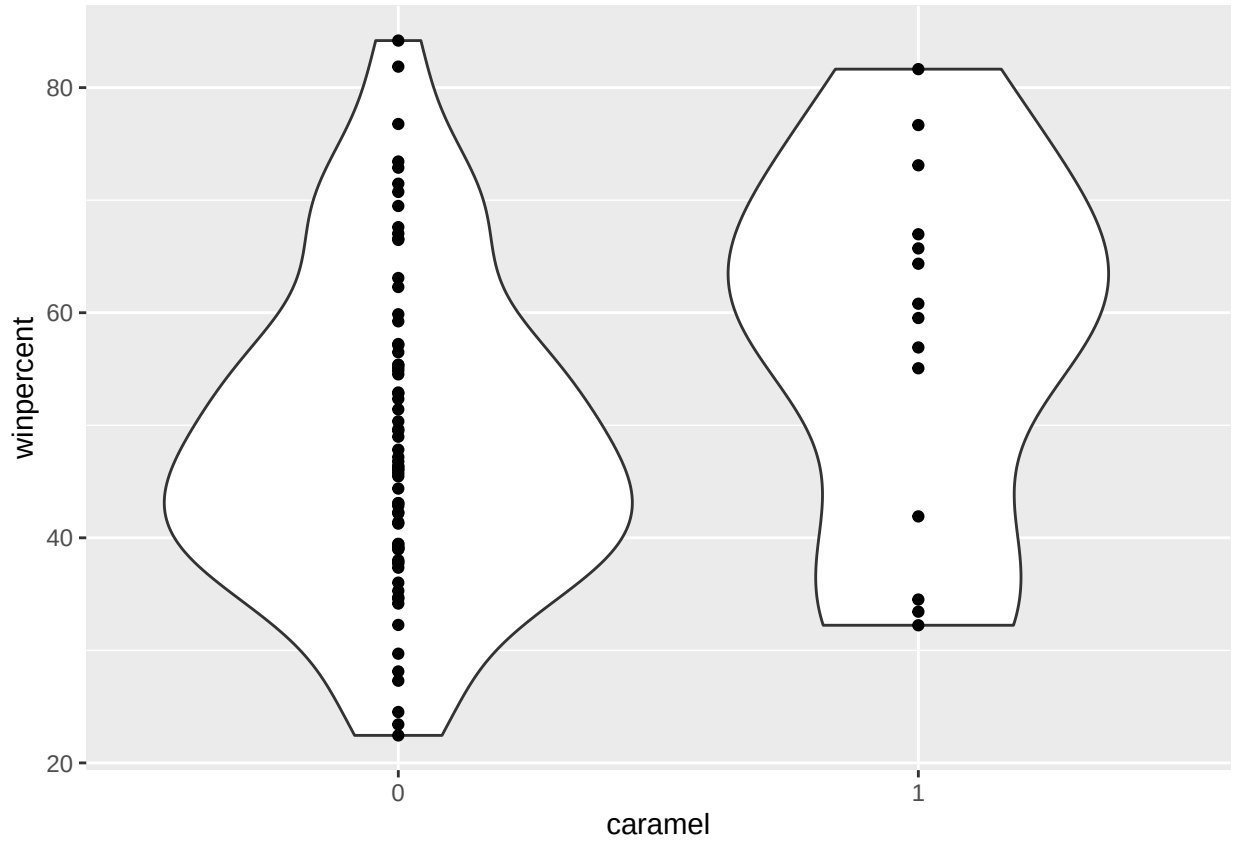
- $\beta_0$ is the expected win percentage for a candy without chocolate and caramel
- $\beta_1$ is the expected increase in win percentage between candy with chocolate and no caramel, from that of candy with no chocolate and no caramel
- $\beta_2$ is the expected increase in win percentage between candy with caramel and no chocolate, from that of candy with no caramel and no chocolate
- $\beta_3$ is an interaction term which can be interpreted as the difference in the increase in win percentage for a candy that has both chocolate *and* caramel. In other words, to estimate the expected win percentage for a candy with both chocolate and caramel, we would use $\beta_0 + \beta_1 + \beta_2 + \beta_3$

b. Fit the model with software of your choice and print the results. (2 points)

```
candy <- read_csv('https://math.montana.edu/ahoegh/teaching/stat446/candy-data.csv') %>%
  mutate(chocolate = factor(chocolate), caramel = factor(caramel))
```

```
## 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 %>% ggplot(aes(y = winpercent, x = caramel))  +
  geom_violin() + geom_point()
```




```
lm_candy <- lm(winpercent ~ chocolate * caramel , data = candy)
lm_candy %>% summary()
```

```
##
## Call:
## lm(formula = winpercent ~ chocolate * caramel, data = candy)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -24.290  -7.453  -1.005   8.591  25.168
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)           42.744      1.697  25.189  < 2e-16 ***
## chocolate1            16.268      2.752   5.912 7.69e-08 ***
## caramel1              -7.221      5.878  -1.228   0.2228
## chocolate1:caramel1   14.286      7.205   1.983   0.0508 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.26 on 81 degrees of freedom
## Multiple R-squared:  0.4357, Adjusted R-squared:  0.4148
```

```
## F-statistic: 20.85 on 3 and 81 DF,  p-value: 4.196e-10
```

c. Summarize your results from part 2, in a way that Willy Wonka could understand. (2 points)

Dear Mr. Wonka -

First of all, I'm a big fan of your candy, and the associated movie. At your request we explored the tastiness, using winpercent as a proxy, of candy as a function of `pricepercent`, `chocolate`, and `caramel`. Price doesn't seem to have a meaningful relationship with win percentage of the candy bar. As expected chocolate is good. Surprisingly, caramel itself, without chocolate, is not effective. However, both chocolate and caramel is better than one of those scenarios where the sum is greater than the parts. Candy with both caramel and chocolate are generally facored above those with just chocolate or just caramel - or heaven forbid, the absence of chocolate.

d. Using your model from part b, create the candy with the highest win percentage. Then specify the levels of the predictors and create a predictive distribution for an individual type of candy with those features. (2 points)

We can explore intervals for the 4 combinations of chocolate and caramel. Price percentage wasn't found to be a meaningful parameter in explaing win percentage.

```
new_candy <- tibble(chocolate = factor(c(0,0,1,1)), caramel = factor(c(1, 0, 1, 0)))

new_candy %>% bind_cols(predict(lm_candy, newdata = new_candy, interval = 'prediction') )
```

```
## # A tibble: 4 x 5
##   chocolate caramel   fit   lwr   upr
##   <fct>     <fct>   <dbl> <dbl> <dbl>
## 1 0         1        35.5  10.5  60.6
## 2 0         0        42.7  20.1  65.4
## 3 1         1        66.1  42.6  89.6
## 4 1         0        59.0  36.2  81.8
```

On average, candy with chocolate and caramel would be expected to win the most faceoffs. However, there is a fair amount of variablity for a single candy bar.

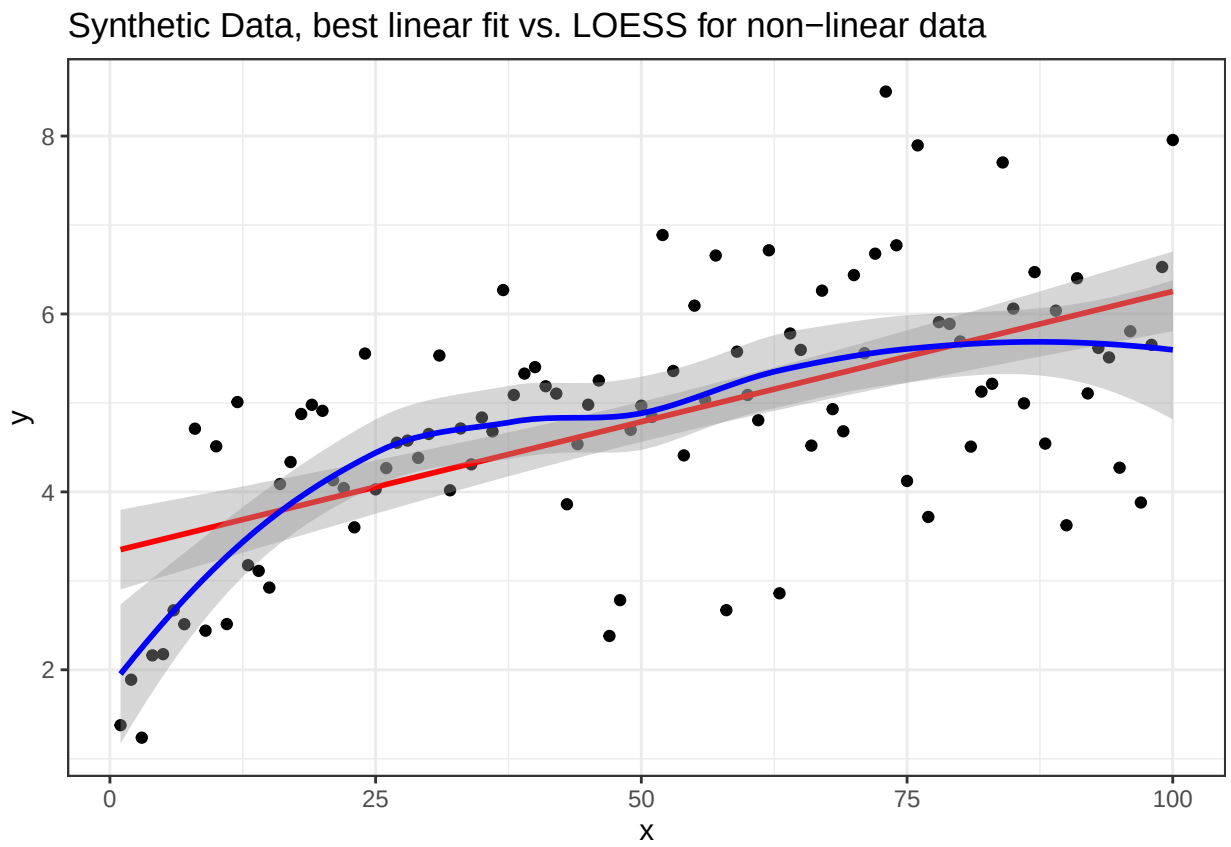
**Question 2 (from 2021 midterm)**

a. (2 points)

Consider the code below. Create a figure of x and y. What linear regression assumption does this data violate?

```
n <- 100
x <- seq(1,100, length.out = n)
sigma <- 1
beta <- c(1, 1)
x_star <- log(x)
y <- rnorm(n, beta[1] + beta[2] * x_star, sd = sigma)
combined <- tibble(y = y, x = x)
```

```r
combined %>% ggplot(aes(x = x, y = y)) +
  geom_point() +
  geom_smooth(method = 'lm', color =  'red', formula = 'y ~x') +
  geom_smooth(method = 'loess', color =  'blue', formula = 'y ~x') +
  theme_bw() +
  ggtitle('Synthetic Data, best linear fit vs. LOESS for non-linear data')
```




Scatterplot of synthetic data generated with non-linear relationship. The blue line is LOESS fit between x and y. The red line is best linear fit. The true relationship is linear between y and log x, which looks quite similar to the blue line.

Fitting this model, on this data, clearly violates the linearity assumption.

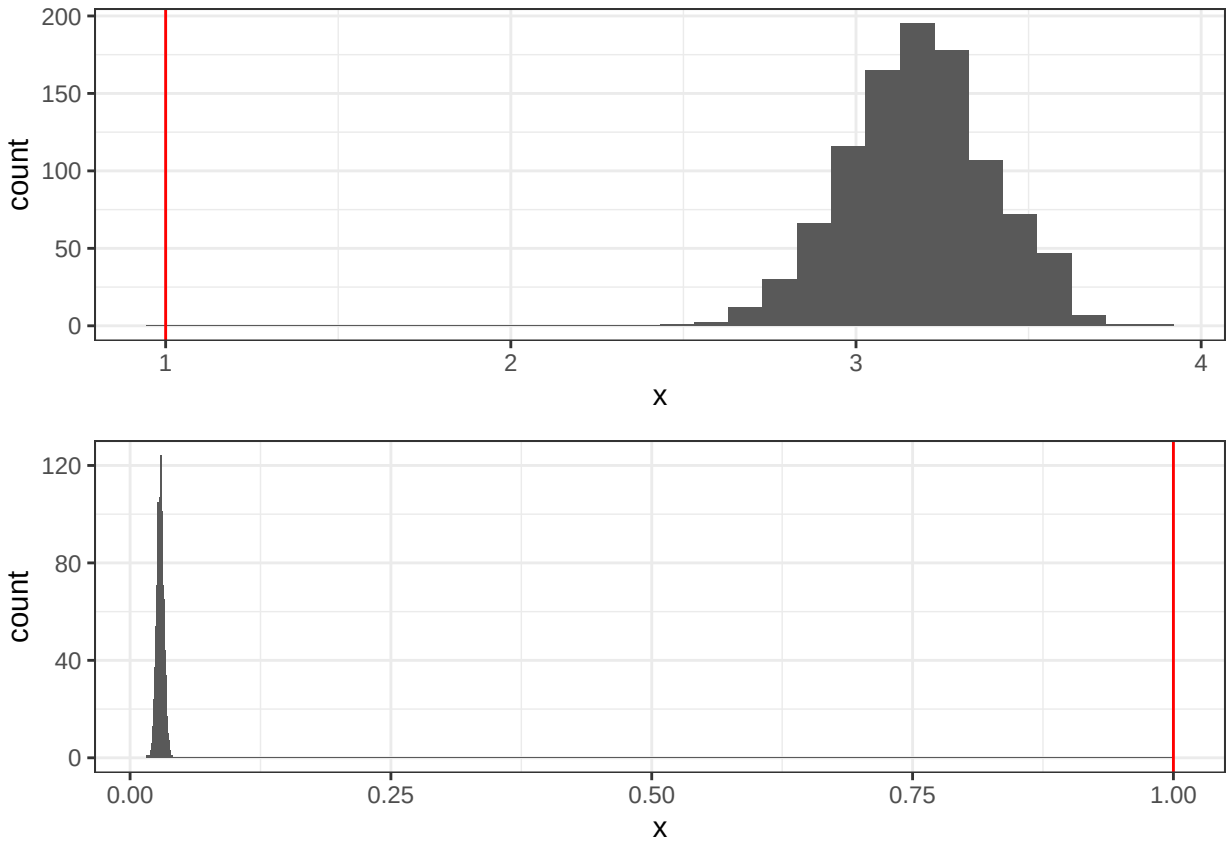
b. (4 points)

How well does a linear regression model (`y ~ x`) recover the point estimates of beta? Justify your answer by using several (~1000) replications of simulated data.

```r
num_reps <- 1000
point_estimates <- confint_beta0 <- confint_beta1 <-  matrix(0, nrow = num_reps, ncol = 2)

for (i in 1:num_reps){
  # simulate synthetic data
  n <- 100
  x <- seq(1,100, length.out = n)
  sigma <- 1
```

```
  beta <- c(1, 1)
  x_star <- log(x)
  y <- rnorm(n, beta[1] + beta[2] * x_star, sd = sigma)
  combined <- tibble(y = y, x = x)

  # fit model & return confidence interval

  lm_out <- lm(y ~ x, data = combined)

  # return point estimates and confidence intervals

  point_estimates[i,] = lm_out$coefficients
  confint_beta0[i,] <- confint(lm_out)[1,]
  confint_beta1[i,] <- confint(lm_out)[2,]
}

f1 <- tibble(x = point_estimates[,1]) %>%
  ggplot(aes(x=x)) +
  geom_histogram() +
  theme_bw() +
  geom_vline(xintercept = 1, color = 'red')

f2 <- tibble(x = point_estimates[,2]) %>%
  ggplot(aes(x=x)) +
  geom_histogram(bins = 1000) +
  theme_bw() +
  geom_vline(xintercept = 1, color = 'red')

grid.arrange(f1, f2)
```

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




The point estimates are a long way from the true simulated values. Furthermore, 0 and 0 out of 1000 of the confidence intervals contain the true parameter values.

```
mean(confint_beta0[,1] < 1 & confint_beta0[,2] > 1)
```

```
## [1] 0
```

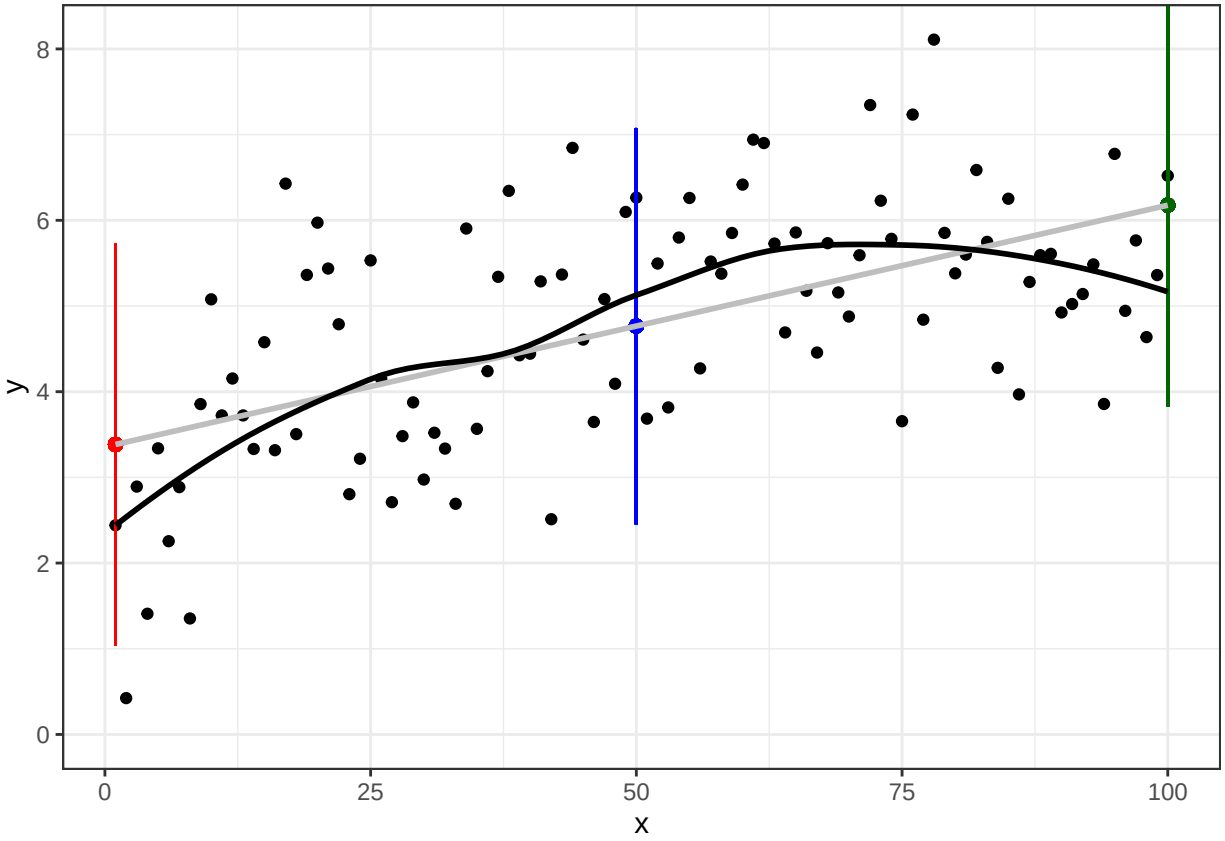
c. (4 points)

How well does a linear regression model capture the uncertainty in a predictions for y conditional on

- x = 1
- x = 50
- x = 100

```
lm_qc <- lm(y ~ x, data = combined)
preds1 <- predict(lm_qc, newdata = tibble(x=1), interval = 'prediction')
preds50 <- predict(lm_qc, newdata = tibble(x=50), interval = 'prediction')
preds100 <- predict(lm_qc, newdata = tibble(x=100), interval = 'prediction')
combined %>% ggplot(aes(y = y, x=x)) +
  geom_point() + theme_bw() +
  geom_segment(x = 1, xend = 1, y = preds1[2], yend = preds1[3], color = 'red') +
  geom_point(x = 1, y = preds1[1], color = 'red', size = 2) +
  geom_segment(x = 50, xend = 50, y = preds50[2], yend = preds50[3], color = 'blue') +
  geom_point(x = 50, y = preds50[1], color = 'blue', size = 2) +
```

```
geom_segment(x = 100, xend = 100, y = preds100[2], yend = preds100[3], color = 'darkgreen') +
geom_point(x = 100, y = preds100[1], color = 'darkgreen', size = 2) +
geom_smooth(formula = 'y~x', method = 'lm', se = F, color = 'grey') +
geom_smooth(formula = 'y~x', method = 'loess', se = F, color = 'black') +
ylim(0,NA)
```




The figure contains prediction intervals, in different colors, along with a linear fit in grey and the loess curve in black. The prediction interval at `x=50` is fairly reasonable - centered near the correct value with reasonable uncertainty. The intervals are less effective at `x=1` and `x=100` with noticeable bias. The interval at `x=1` could easily miss points.