Problem set 8: Worked solutions

Statistics and statistical programming
Northwestern University
MTS 525

Aaron Shaw

November 23, 2020

Contents

Import libraries

I'll start by loading some useful libraries...

```
library(openintro)
library(tidyverse)
library(ggfortify)
library(haven)
```

Part I: Mario kart replication/extension

Import data and setup

```
data(mariokart)

mariokart
```

```
## # A tibble: 143 x 12
```

```
##         id duration n_bids cond  start_pr ship_pr total_pr ship_sp seller_rate
##      <dbl>    <int>  <int> <fct>    <dbl>   <dbl>    <dbl> <fct>         <int>
##  1 1.50e11        3     20 new       0.99    4        51.6 standa~        1580
##  2 2.60e11        7     13 used      0.99    3.99     37.0 firstC~         365
##  3 3.20e11        3     16 new       0.99    3.5      45.5 firstC~         998
##  4 2.80e11        3     18 new       0.99    0        44   standa~           7
##  5 1.70e11        1     20 new       0.01    0        71   media            820
##  6 3.60e11        3     19 new       0.99    4        45   standa~      270144
##  7 1.20e11        1     13 used      0.01    0        37.0 standa~        7284
##  8 3.00e11        1     15 new       1       2.99     54.0 upsGro~        4858
##  9 2.00e11        3     29 used      0.99    4        47   priori~          27
## 10 3.30e11        7      8 used     20.0     4        50   firstC~         201
## # ... with 133 more rows, and 3 more variables: stock_photo <fct>,
## #   wheels <int>, title <fct>
```

To make things a bit easier to manage, I'll select the variables I want to use in the analysis and do some cleanup. Note that I convert the `cond_new` and `stock_photo` variables to logical `TRUE/FALSE` values first (using boolean comparisons) and then coerce them to be numeric values (using `as.numeric()`). This results in 1/0 values corresponding to the observations shown/described in Figure 9.13 and 9.14 on p. 365 of the textbook.

```
mariokart <- mariokart %>%
  select(
    price = total_pr, cond_new = cond, stock_photo, duration, wheels
  ) %>%
  mutate(
    cond_new = as.numeric(cond_new == "new"),
    stock_photo = as.numeric(stock_photo == "yes")
  )

mariokart
```

```
## # A tibble: 143 x 5
##    price cond_new stock_photo duration wheels
##    <dbl>    <dbl>       <dbl>    <int>  <int>
##  1  51.6        1           1        3      1
##  2  37.0        0           1        7      1
##  3  45.5        1           0        3      1
##  4  44          1           1        3      1
##  5  71          1           1        1      2
##  6  45          1           1        3      0
##  7  37.0        0           1        1      0
##  8  54.0        1           1        1      2
##  9  47          0           1        3      1
## 10  50          0           0        7      1
## # ... with 133 more rows
```

Now let's summarize and explore the variables in our model. I'll calculate summary statistics for each, univariate density plots for the continuous measures, boxplots for the dichotomous indicators, and some bivariate plots with each predictor and the outcome variable.

```
summary(mariokart)
```

```
##      price           cond_new       stock_photo        duration
##  Min.   : 28.98   Min.   :0.0000   Min.   :0.0000   Min.   : 1.000
##  1st Qu.: 41.17   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 1.000
##  Median : 46.50   Median :0.0000   Median :1.0000   Median : 3.000
```

```
##  Mean   : 49.88   Mean   :0.4126   Mean   :0.7343   Mean   : 3.769
##  3rd Qu.: 53.99   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.: 7.000
##  Max.   :326.51   Max.   :1.0000   Max.   :1.0000   Max.   :10.000
##      wheels
##  Min.   :0.000
##  1st Qu.:0.000
##  Median :1.000
##  Mean   :1.147
##  3rd Qu.:2.000
##  Max.   :4.000
```

```
sd(mariokart$price)
```

```
## [1] 25.68856
```

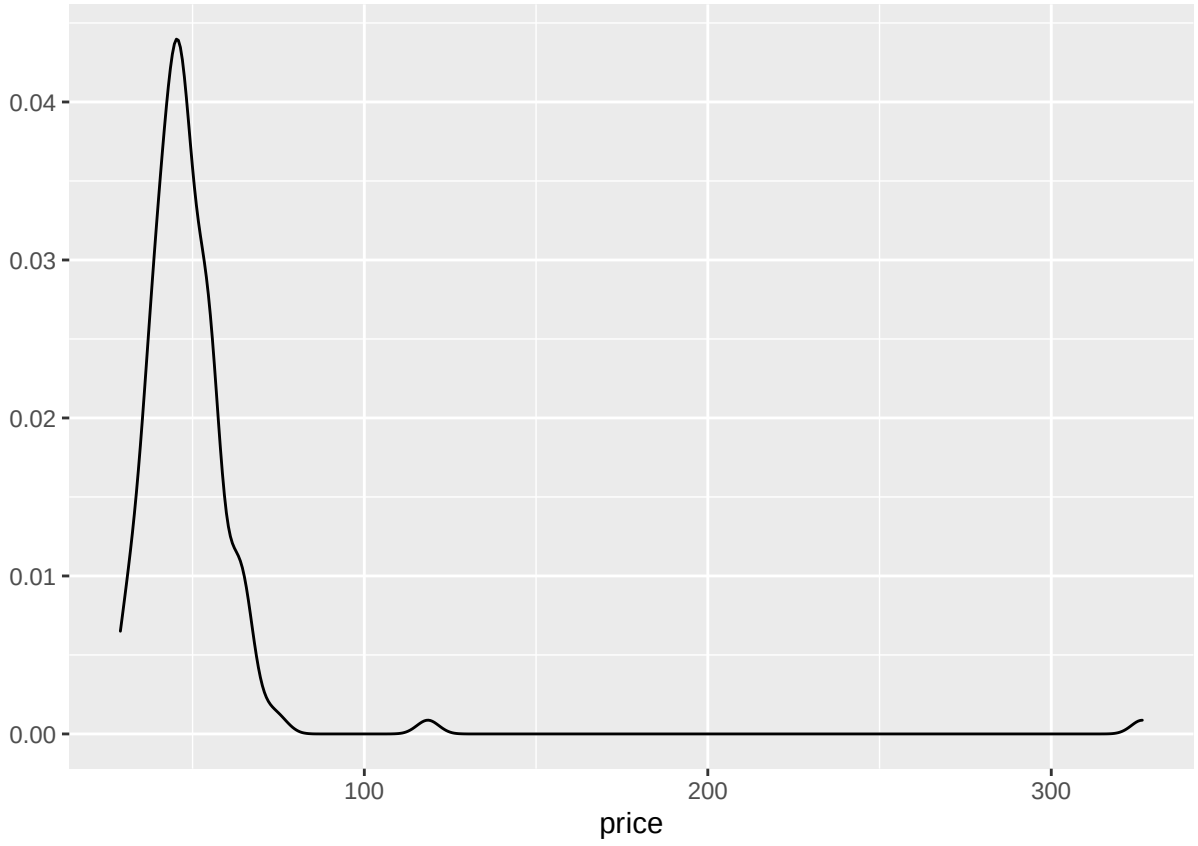
```
sd(mariokart$duration)
```

```
## [1] 2.585693
```

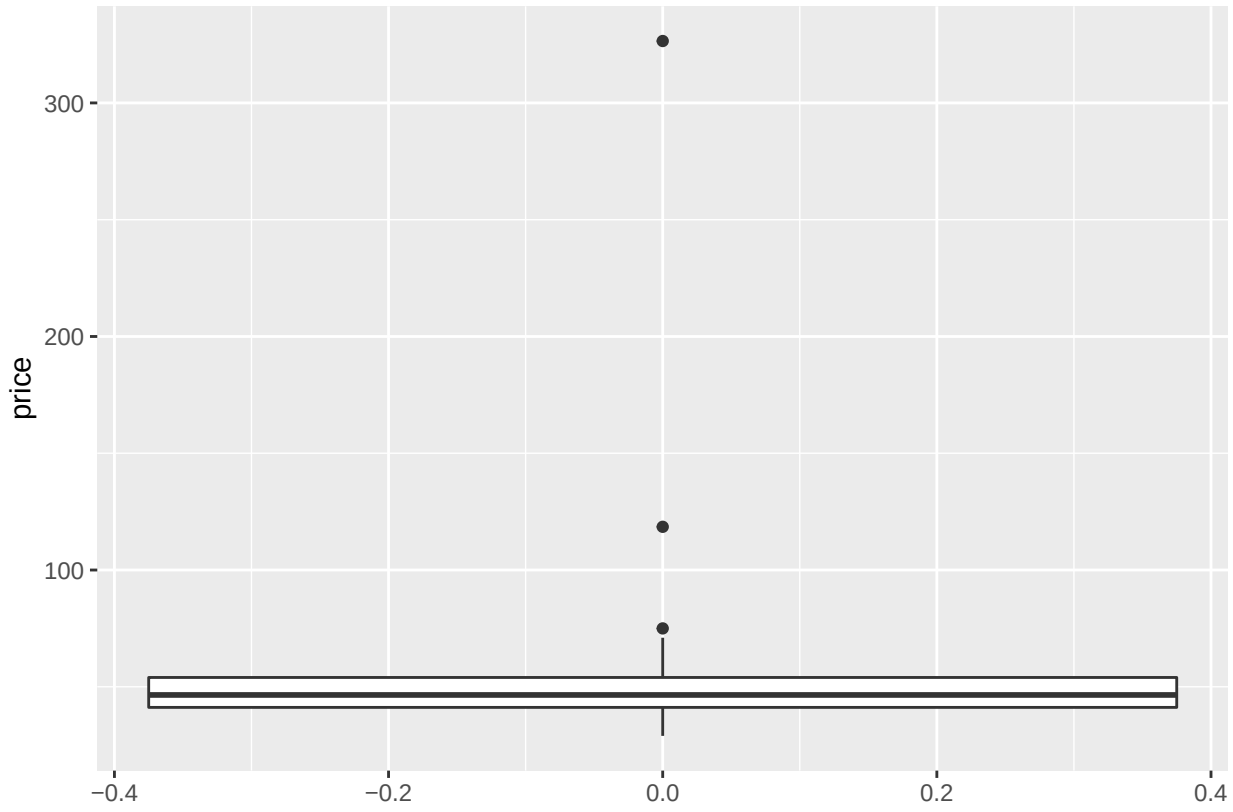
```
sd(mariokart$wheels)
```

```
## [1] 0.8471829
```

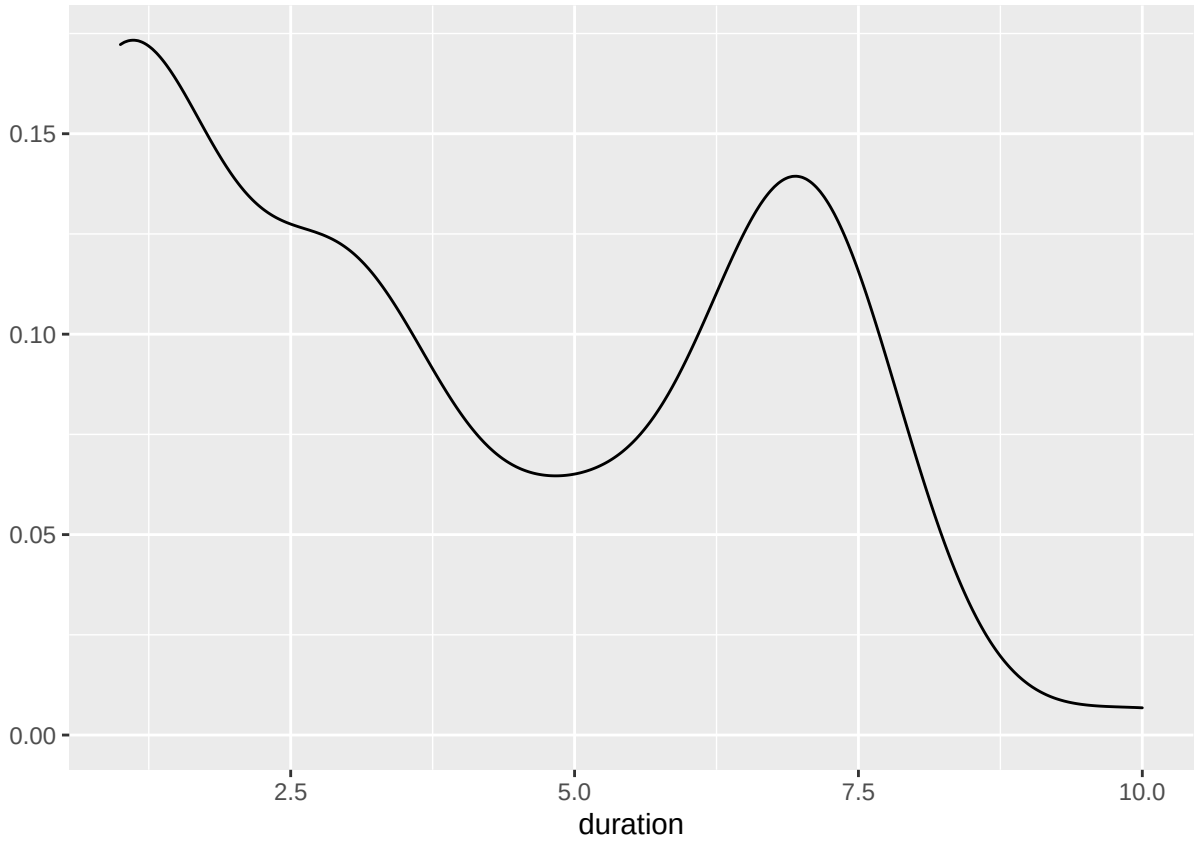
```
qplot(data = mariokart, x = price, geom = "density")
```



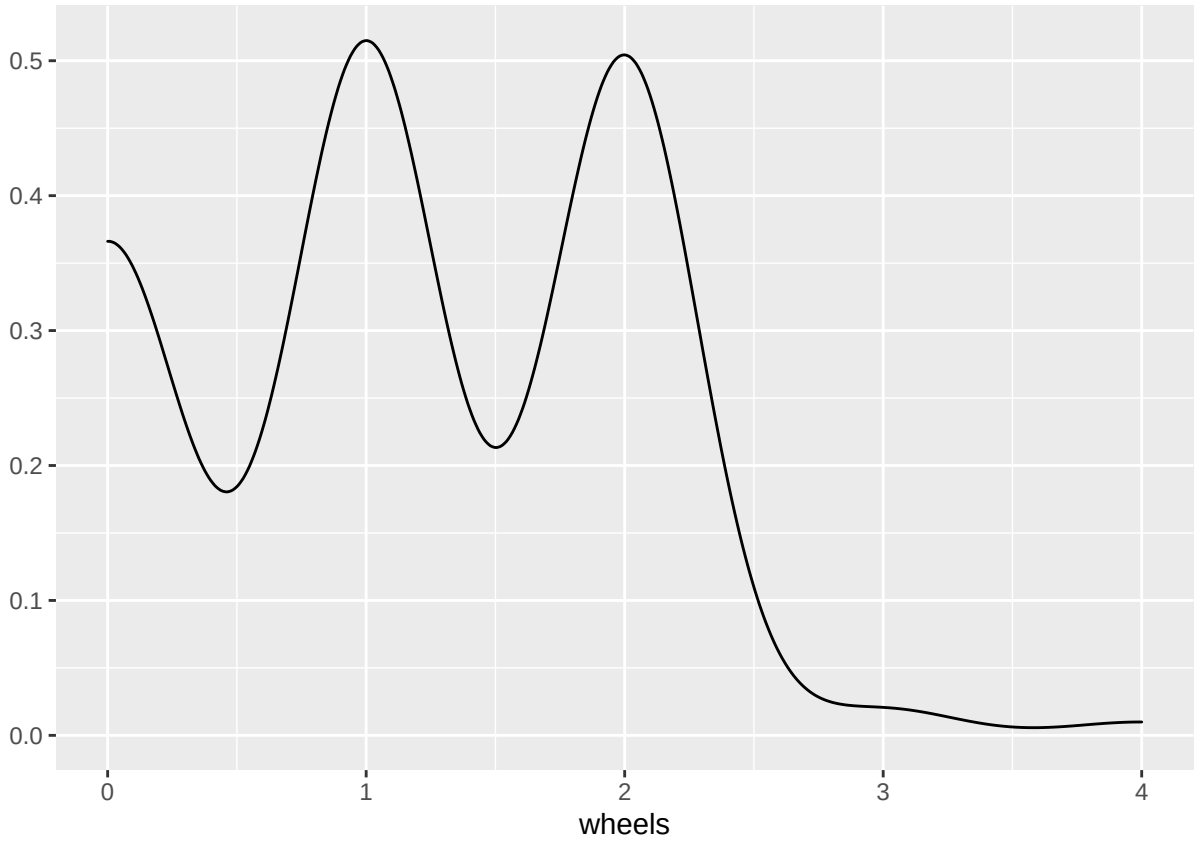

```
qplot(data = mariokart, y = price, geom = "boxplot") # check out the outliers!
```




```
qplot(data = mariokart, x = duration, geom = "density")
```

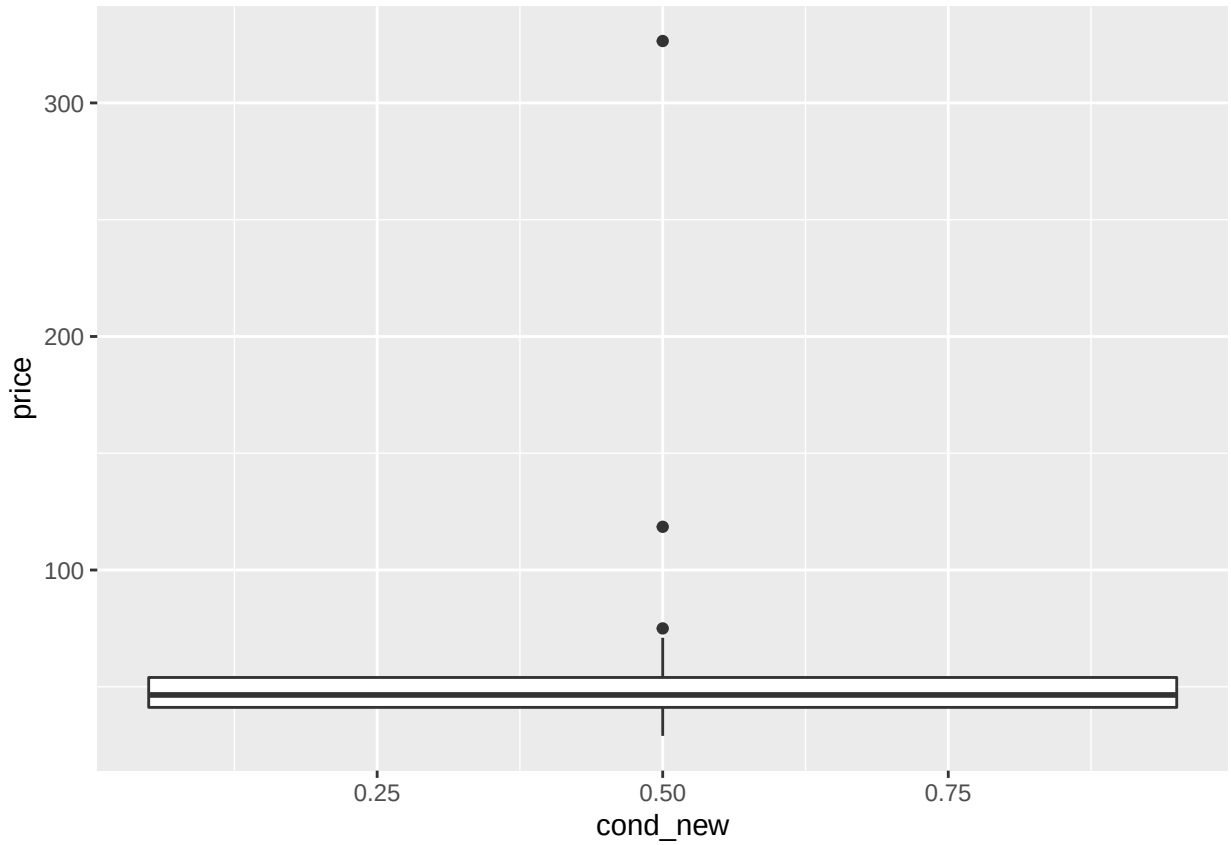


```
qplot(data = mariokart, x = wheels, geom = "density")
```



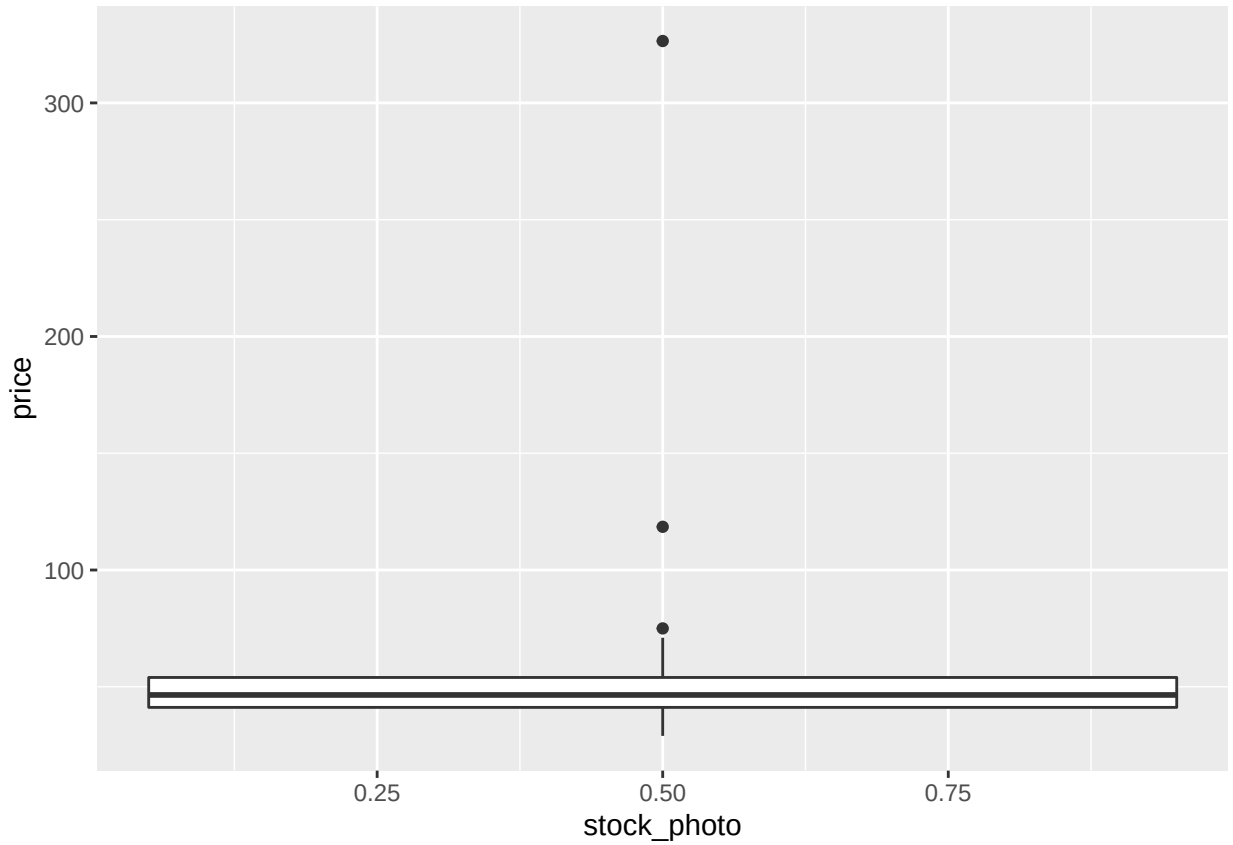

```
ggplot(data = mariokart, aes(cond_new, price)) +
  geom_boxplot()
```

```
## Warning: Continuous x aesthetic -- did you forget aes(group=...)?
```

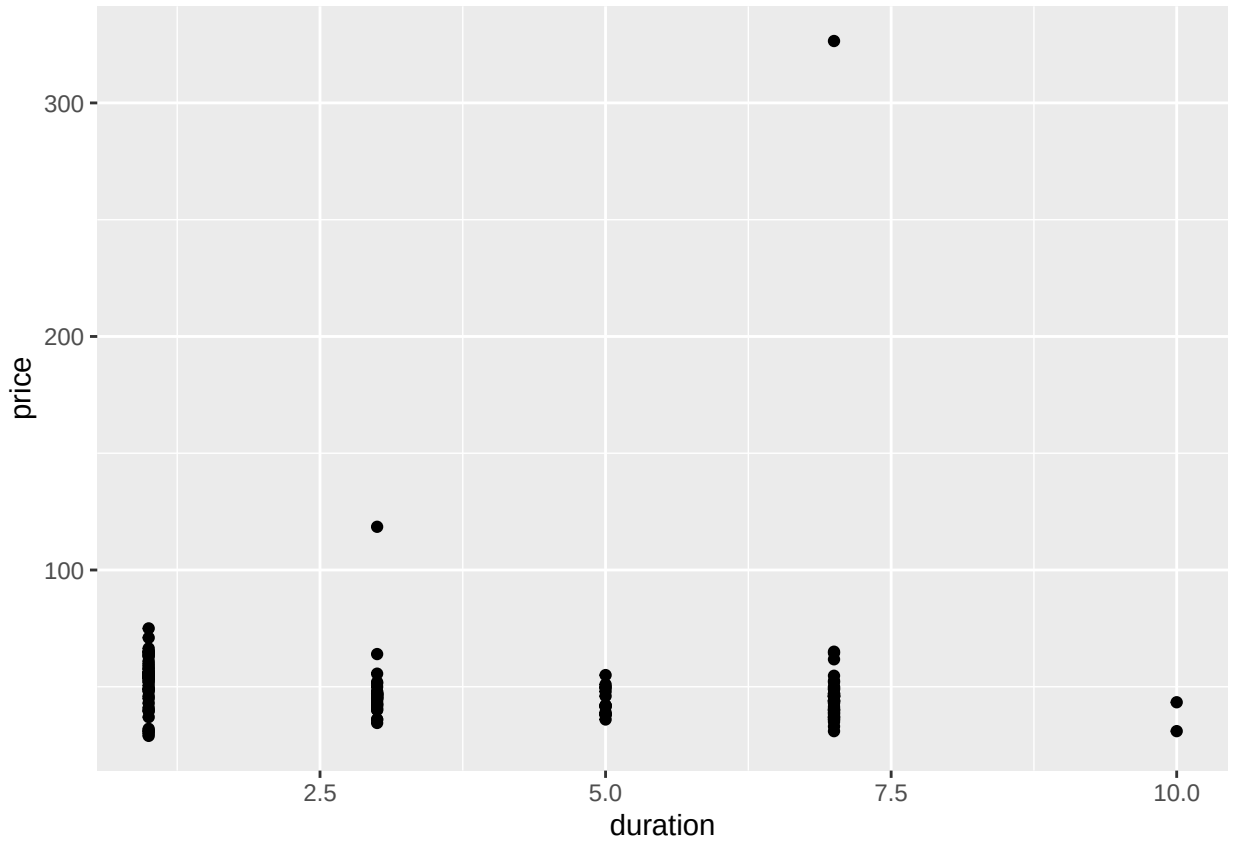



```
ggplot(data = mariokart, aes(stock_photo, price)) +
  geom_boxplot()
```

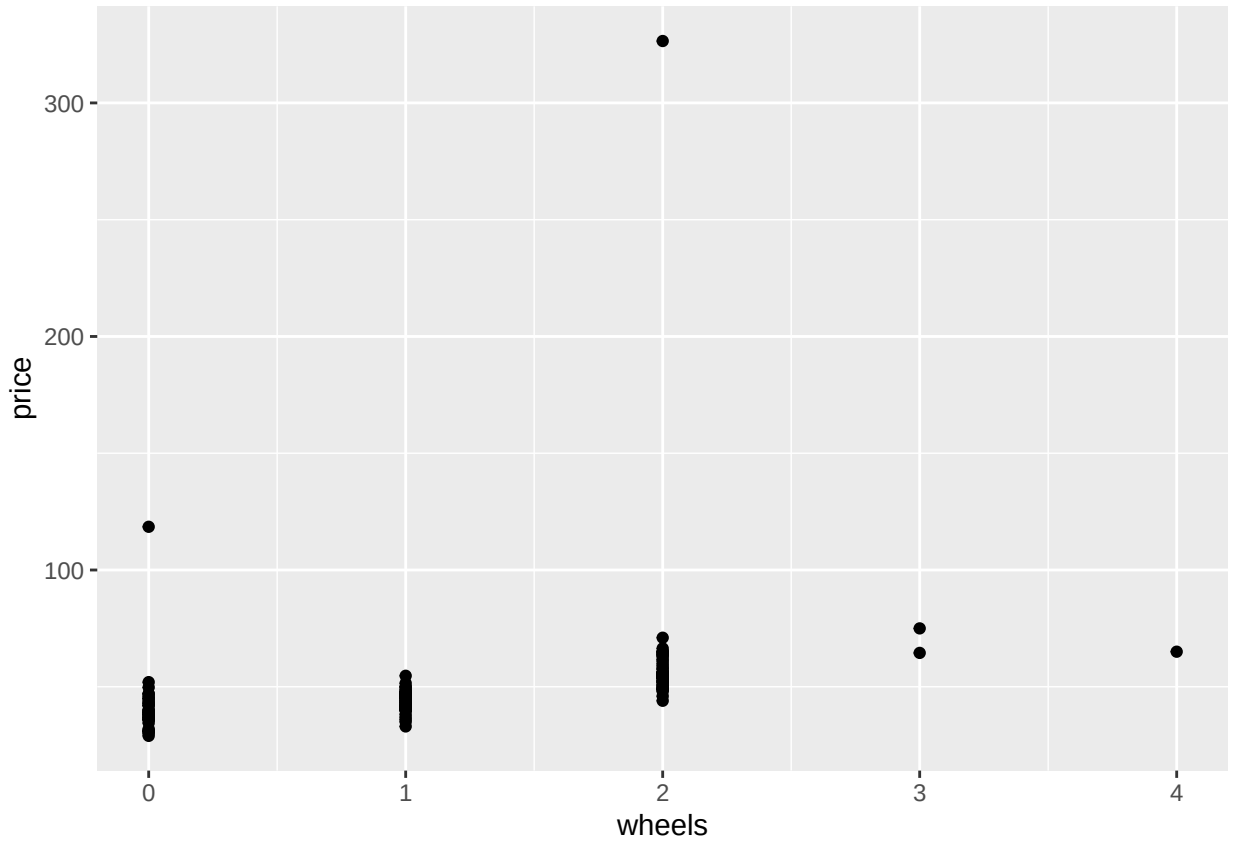
```
## Warning: Continuous x aesthetic -- did you forget aes(group=...)?
```




```
ggplot(data = mariokart, aes(duration, price)) +
  geom_point()
```




```
ggplot(data = mariokart, aes(wheels, price)) +
  geom_point()
```



I'm also going to calculate correlation coefficients for all of the variables. We can discuss what to make of this in class:

```
cor(mariokart)
```

```
##                   price   cond_new stock_photo    duration      wheels
## price        1.00000000  0.1273624 -0.08987876 -0.04123545  0.32998375
## cond_new     0.12736236  1.0000000  0.37554707 -0.48174393  0.42629801
## stock_photo -0.08987876  0.3755471  1.00000000 -0.36722999  0.06714198
## duration    -0.04123545 -0.4817439 -0.36722999  1.00000000 -0.29947345
## wheels       0.32998375  0.4262980  0.06714198 -0.29947345  1.00000000
```

Replicate model results

Based on the information in the textbook, I'm assuming that the model is an ordinary least squares regression.

```
model <- lm(price ~ cond_new + stock_photo + duration + wheels, data = mariokart)
summary(model)
```

```
##
## Call:
## lm(formula = price ~ cond_new + stock_photo + duration + wheels,
##     data = mariokart)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -19.485  -6.511  -2.530   1.836 263.025
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  40.9385     7.3607   5.562 1.34e-07 ***
## cond_new      2.5816     5.2272   0.494 0.622183
## stock_photo  -6.7542     5.1729  -1.306 0.193836
## duration      0.3788     0.9388   0.403 0.687206
## wheels        9.9476     2.7184   3.659 0.000359 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 24.4 on 138 degrees of freedom
## Multiple R-squared:  0.1235, Adjusted R-squared:  0.09808
## F-statistic:  4.86 on 4 and 138 DF,  p-value: 0.001069
```

Huh, that doesn't quite look like what's in the textbook Figure 9.15...

I'll go ahead and calculate a confidence interval around the only parameter for which the model rejects the null hypothesis (`wheels`):

```
confint(model, "wheels")
```

```
##           2.5 %   97.5 %
## wheels 4.572473 15.32278
```

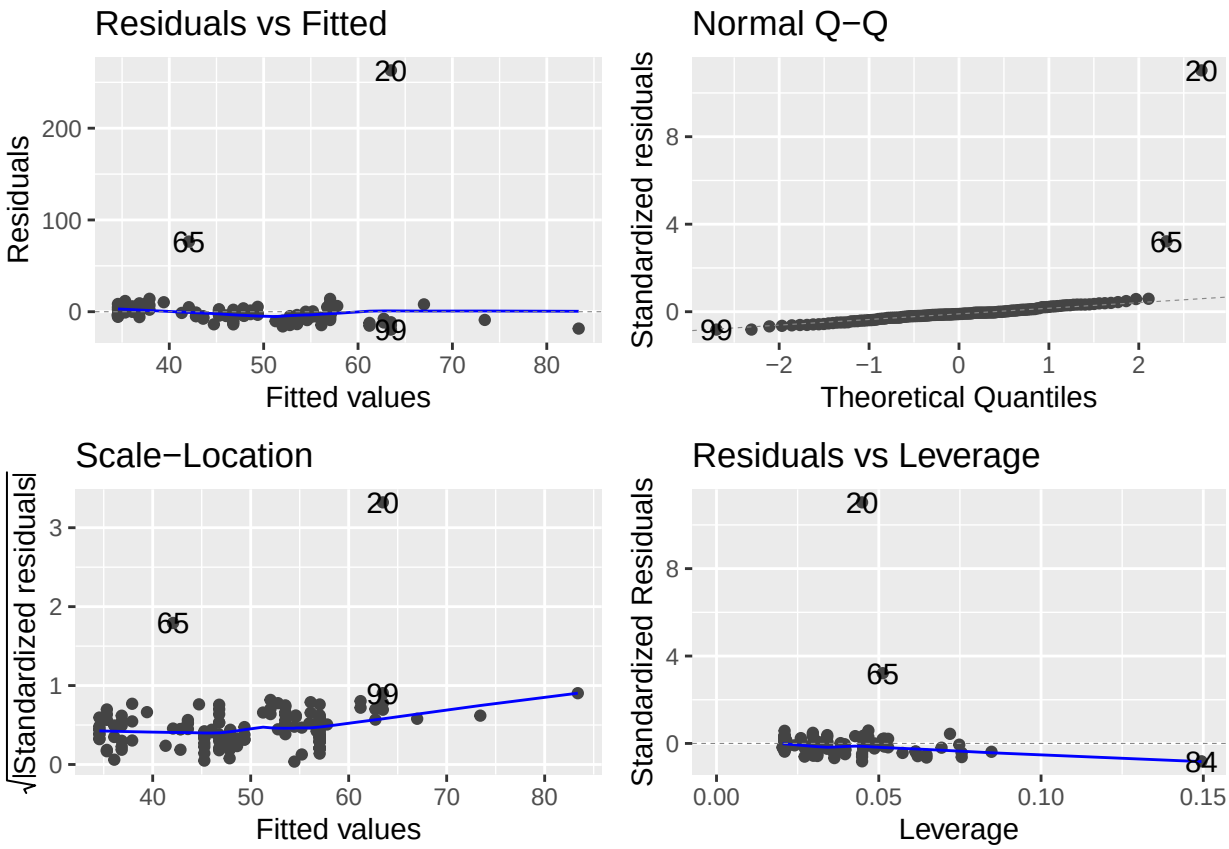
Without more information from the authors of the textbook it's very hard to determine exactly where or why the differences emerge. My guess at this point is that it might have something to do with those outlying `price` values (take a look at that boxplot and density plot again). Maybe they did something to transform the variable? Remove the outliers? I have no idea.

Assess model fit/assumptions

I've already generated a bunch of univariate and bivariate summaries and plots. Let's inspect the residuals more closely to see what else we can learn. I'll use the `autoplot()` function from the `ggfortify` package.

```
autoplot(model)
```

```
## Warning: `arrange_()` is deprecated as of dplyr 0.7.0.
## Please use `arrange()` instead.
## See vignette('programming') for more help
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```



There are those outliers again. At this point, there are a number of issues with this model fit that I'd want to mention/consider: * The distribution of the dependent variable (`price`) is skewed, with two extreme outliers. I'd recommend trying some transformations to see if it would look more appropriate for a linear regression and/or inspecting the cases that produced the outlying values more closely to understand what's happening there and identify reasons why they're so different. * The plots of the residuals reveal those same two outlying points are also outliers with respect to the line of best fit. That said, they are not exerting huge amounts of leverage on the estimates, so it's possible that the estimates from the fitted model wouldn't change *too much* without those two points. Indeed, based on the degrees of freedom reported in Figure 9.15 (136) vs. the number reported in our version of the model (138) my best guess at this point is that the textbook authors silently dropped those two outlying observations from their model.

More out of curiosity than anything, I'll create a version of the model that drops the two largest values of price. From the plots, I can see that those two are the only ones above $100, so I'll use that information here:

```
summary(
  lm(price ~ cond_new + stock_photo + duration + wheels,
    data = mariokart[mariokart$price < 100, ]
  )
)
```

```
##
## Call:
## lm(formula = price ~ cond_new + stock_photo + duration + wheels,
##     data = mariokart[mariokart$price < 100, ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.3788  -2.9854  -0.9654   2.6915  14.0346
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 36.21097    1.51401  23.917  < 2e-16 ***
## cond_new     5.13056    1.05112   4.881 2.91e-06 ***
## stock_photo  1.08031    1.05682   1.022    0.308
## duration    -0.02681    0.19041  -0.141    0.888
## wheels       7.28518    0.55469  13.134  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.901 on 136 degrees of freedom
## Multiple R-squared:  0.719,  Adjusted R-squared:  0.7108
## F-statistic: 87.01 on 4 and 136 DF,  p-value: < 2.2e-16
```

What do you know. That was it. The difference in R^2 is huge!

Interpret some results

The issues above notwithstanding, we can march ahead and interpret the results of the original model that I fit. Here are some general comments and some specifically focused on the `cond_new` and `stock_photo` variables: * Overall, the linear model regressing total auction price on condition, stock photo, duration, and number of Wii wheels shows evidence of a positive, significant relationship between number of wheels and price. According to this model fit, an increase of 1 wheel in a listing is associated with a total auction price increase of $10 on average (95% confindence interval: $4.57-$15.32). * The point estimate for selling a new condition game is positive, but with a large standard error. The model fails to reject the null of no association and provides no evidence of any relationship between the game condition and auction price. * The point estimate for including a stock photo is negative, but again, the standard error is very large and the model fails to reject the null hypothesis. There is no evidence of any relationship between including a stock photo and the final auction price.

Recommendations

Based on this model result, I'd recommend the prospective vendor of a **used** copy of the game not worry about it too much unless they can get their hands on some extra Wii wheels, since it seems like the number of Wii wheels explains variations in the auction price outcome more than anything else they can control (such as whether or not a stock photo is included).

Part II: Hypothetical study

Import, explore, summarize

I'll start off by just importing things and summarizing the different variables we care about here:

```
grads <- readRDS(url("https://communitydata.science/~ads/teaching/2020/stats/data/week_11/grads.rds"))

summary(grads)
```

```
##        id              cohort         gpa              income
##  Min.   :   1.0   cohort_01:142   Min.   : 0.01512   Min.   :15.56
##  1st Qu.: 462.2   cohort_02:142   1st Qu.:22.56107   1st Qu.:41.07
##  Median : 923.5   cohort_03:142   Median :47.59445   Median :52.59
##  Mean   : 923.5   cohort_04:142   Mean   :47.83510   Mean   :54.27
##  3rd Qu.:1384.8   cohort_05:142   3rd Qu.:71.81078   3rd Qu.:67.28
##  Max.   :1846.0   cohort_06:142   Max.   :99.69468   Max.   :98.29
```

```
##                   (Other)   :994
```

```
table(grads$cohort)
```

```
##
## cohort_01 cohort_02 cohort_03 cohort_04 cohort_05 cohort_06 cohort_07 cohort_08
##       142       142       142       142       142       142       142       142
## cohort_09 cohort_10 cohort_11 cohort_12 cohort_13
##       142       142       142       142       142
```

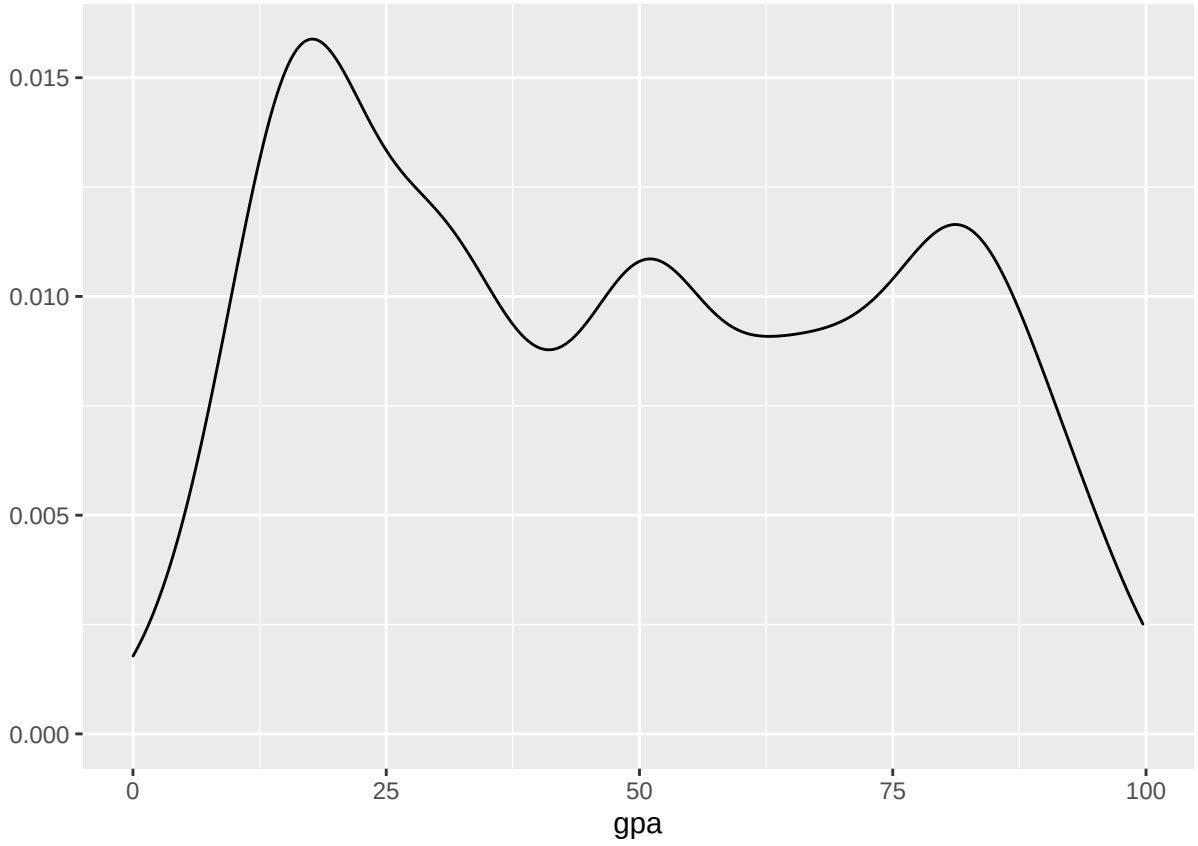
```
sd(grads$gpa)
```

```
## [1] 26.84777
```

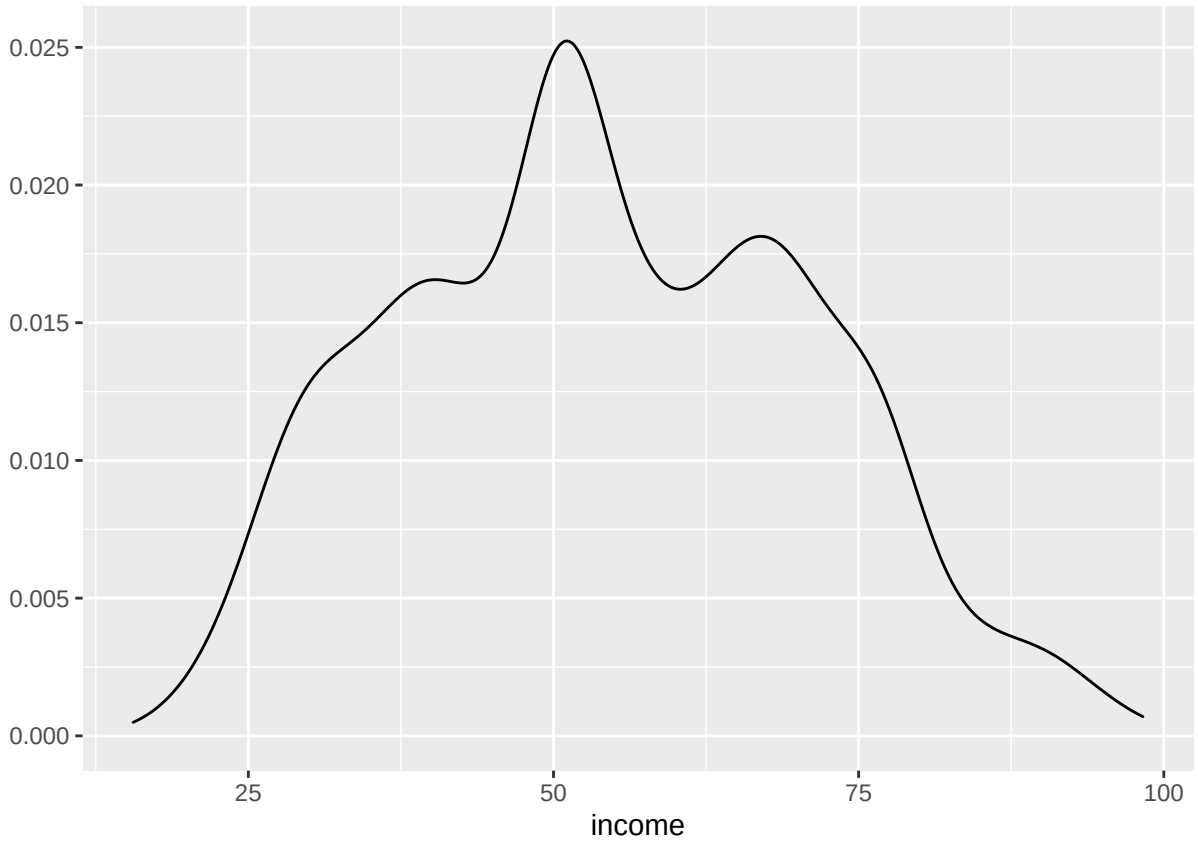
```
sd(grads$income)
```

```
## [1] 16.713
```

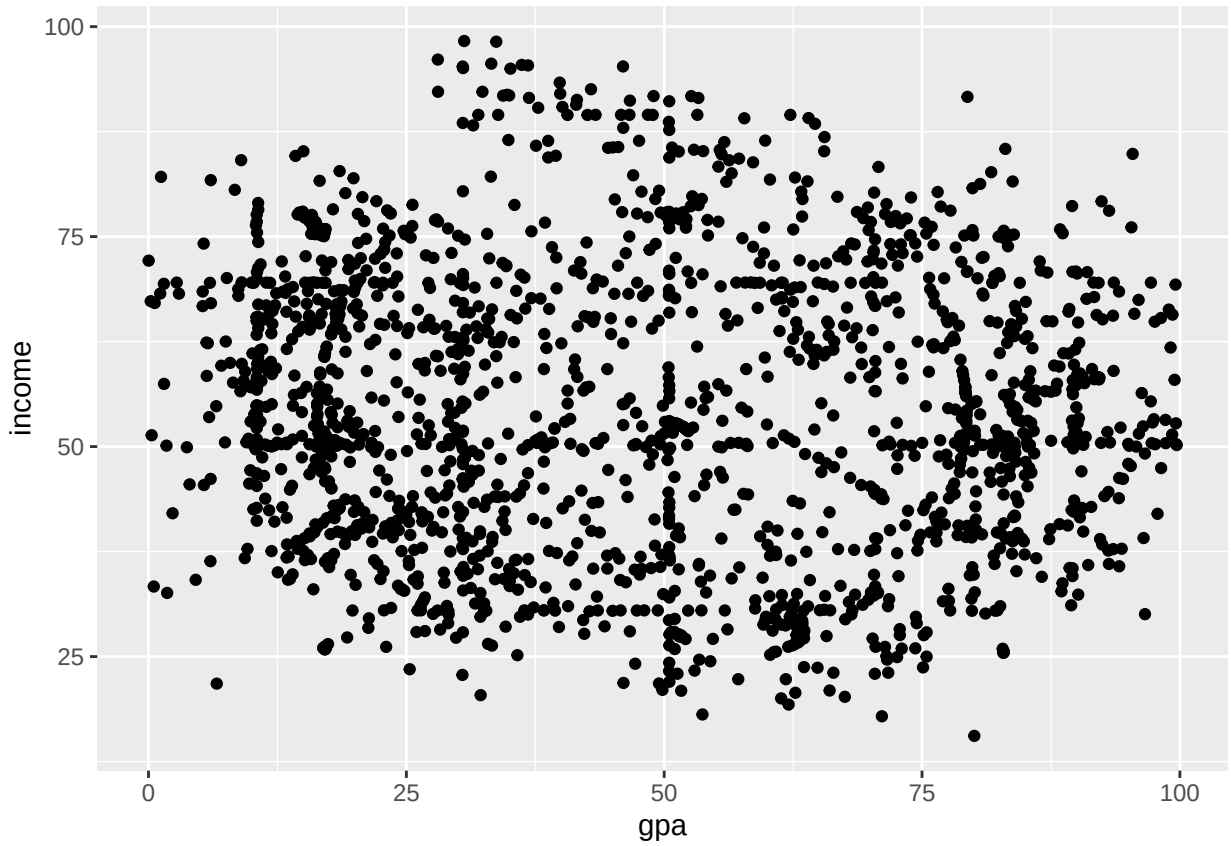
```
qplot(data = grads, x = gpa, geom = "density")
```




```
qplot(data = grads, x = income, geom = "density")
```




```
ggplot(data = grads, aes(x = gpa, y = income)) +
  geom_point()
```



I'll also calculate some summary statistics and visual comparisons within districts (cohorts):

```
tapply(grads$income, grads$cohort, summary)
```

```
## $cohort_01
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   27.02   41.03   56.53   54.27   68.71   86.44
##
## $cohort_02
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   25.44   50.36   50.98   54.26   75.20   77.95
##
## $cohort_03
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   20.21   42.81   54.26   54.27   64.49   95.26
##
## $cohort_04
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   22.00   42.29   53.07   54.26   66.77   98.29
##
## $cohort_05
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   30.45   49.96   50.36   54.27   69.50   89.50
##
## $cohort_06
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   17.89   41.54   54.17   54.27   63.95   96.08
##
```

```
## $cohort_07
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   31.11   40.09   47.14   54.26   71.86   85.45
##
## $cohort_08
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   22.31   44.10   53.33   54.26   64.74   98.21
##
## $cohort_09
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   15.56   39.72   53.34   54.27   69.15   91.64
##
## $cohort_10
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   27.44   35.52   64.55   54.27   67.45   77.92
##
## $cohort_11
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   19.29   41.63   53.84   54.27   64.80   91.74
##
## $cohort_12
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   21.86   43.38   54.02   54.27   64.97   85.66
##
## $cohort_13
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.11   42.89   53.14   54.27   64.47   95.59
```

```
tapply(grads$income, grads$cohort, sd)
```

```
## cohort_01 cohort_02 cohort_03 cohort_04 cohort_05 cohort_06 cohort_07 cohort_08
##  16.76896  16.76774  16.76885  16.76590  16.76996  16.76670  16.76996  16.76514
## cohort_09 cohort_10 cohort_11 cohort_12 cohort_13
##  16.76982  16.77000  16.76924  16.76001  16.76676
```

```
tapply(grads$gpa, grads$cohort, summary)
```

```
## $cohort_01
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   14.37   20.37   50.11   47.84   63.55   92.21
##
## $cohort_02
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   15.77   17.11   51.30   47.84   82.88   94.25
##
## $cohort_03
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.646  24.756  45.292  47.831  70.856  99.580
##
## $cohort_04
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   10.46   30.48   50.47   47.83   70.35   90.46
##
## $cohort_05
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

```
##   2.735  22.753  47.114  47.837  65.845  99.695
##
## $cohort_06
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   14.91   22.92   32.50   47.84   75.94   87.15
##
## $cohort_07
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.578  23.471  39.876  47.840  73.610  97.838
##
## $cohort_08
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.949  25.288  46.026  47.832  68.526  99.487
##
## $cohort_09
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##  0.01512 24.62589 47.53527 47.83472 71.80315 97.47577
##
## $cohort_10
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.217  24.347  46.279  47.832  67.568  99.284
##
## $cohort_11
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   9.692  26.245  47.383  47.831  72.533  85.876
##
## $cohort_12
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   16.33   18.35   51.03   47.84   77.78   85.58
##
## $cohort_13
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3039 27.8409 46.4013 47.8359 68.4394 99.6442
```

```
tapply(grads$gpa, grads$cohort, sd)
```

```
## cohort_01 cohort_02 cohort_03 cohort_04 cohort_05 cohort_06 cohort_07 cohort_08
##  26.93027  26.93019  26.93861  26.93988  26.93768  26.94000  26.93000  26.93540
## cohort_09 cohort_10 cohort_11 cohort_12 cohort_13
##  26.93974  26.93790  26.93573  26.93004  26.93610
```

Note that you could also do this pretty easily with a call to `group_by` and `summarize` in the tidyverse:

```
grads %>%
  group_by(cohort) %>%
  summarize(
    n = n(),
    min = min(income),
    mean = mean(income),
    max = max(income),
    sd = sd(income)
  )
```

```
## # A tibble: 13 x 6
##    cohort        n   min  mean   max    sd
##    <fct>     <int> <dbl> <dbl> <dbl> <dbl>
```

```
##  1 cohort_01   142  27.0  54.3  86.4  16.8
##  2 cohort_02   142  25.4  54.3  78.0  16.8
##  3 cohort_03   142  20.2  54.3  95.3  16.8
##  4 cohort_04   142  22.0  54.3  98.3  16.8
##  5 cohort_05   142  30.4  54.3  89.5  16.8
##  6 cohort_06   142  17.9  54.3  96.1  16.8
##  7 cohort_07   142  31.1  54.3  85.4  16.8
##  8 cohort_08   142  22.3  54.3  98.2  16.8
##  9 cohort_09   142  15.6  54.3  91.6  16.8
## 10 cohort_10   142  27.4  54.3  77.9  16.8
## 11 cohort_11   142  19.3  54.3  91.7  16.8
## 12 cohort_12   142  21.9  54.3  85.7  16.8
## 13 cohort_13   142  18.1  54.3  95.6  16.8
```

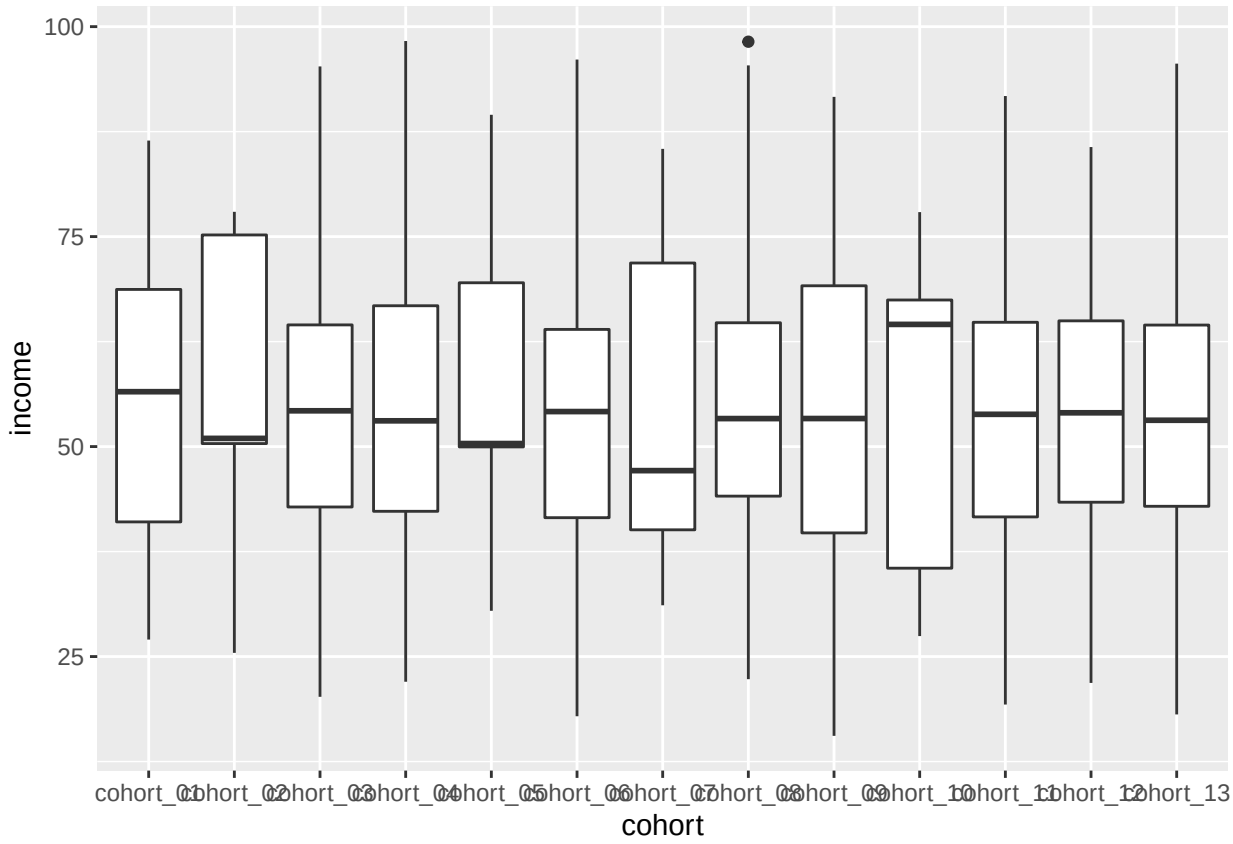
```
grads %>%
  group_by(cohort) %>%
  summarize(
    n = n(),
    min = min(gpa),
    mean = mean(gpa),
    max = max(gpa),
    sd = sd(gpa)
  )
```

```
## # A tibble: 13 x 6
##    cohort        n     min  mean   max    sd
##    <fct>     <int>   <dbl> <dbl> <dbl> <dbl>
##  1 cohort_01   142 14.4     47.8  92.2  26.9
##  2 cohort_02   142 15.8     47.8  94.2  26.9
##  3 cohort_03   142  5.65    47.8  99.6  26.9
##  4 cohort_04   142 10.5     47.8  90.5  26.9
##  5 cohort_05   142  2.73    47.8  99.7  26.9
##  6 cohort_06   142 14.9     47.8  87.2  26.9
##  7 cohort_07   142  4.58    47.8  97.8  26.9
##  8 cohort_08   142  2.95    47.8  99.5  26.9
##  9 cohort_09   142  0.0151  47.8  97.5  26.9
## 10 cohort_10   142  0.217   47.8  99.3  26.9
## 11 cohort_11   142  9.69    47.8  85.9  26.9
## 12 cohort_12   142 16.3     47.8  85.6  26.9
## 13 cohort_13   142  0.304   47.8  99.6  26.9
```

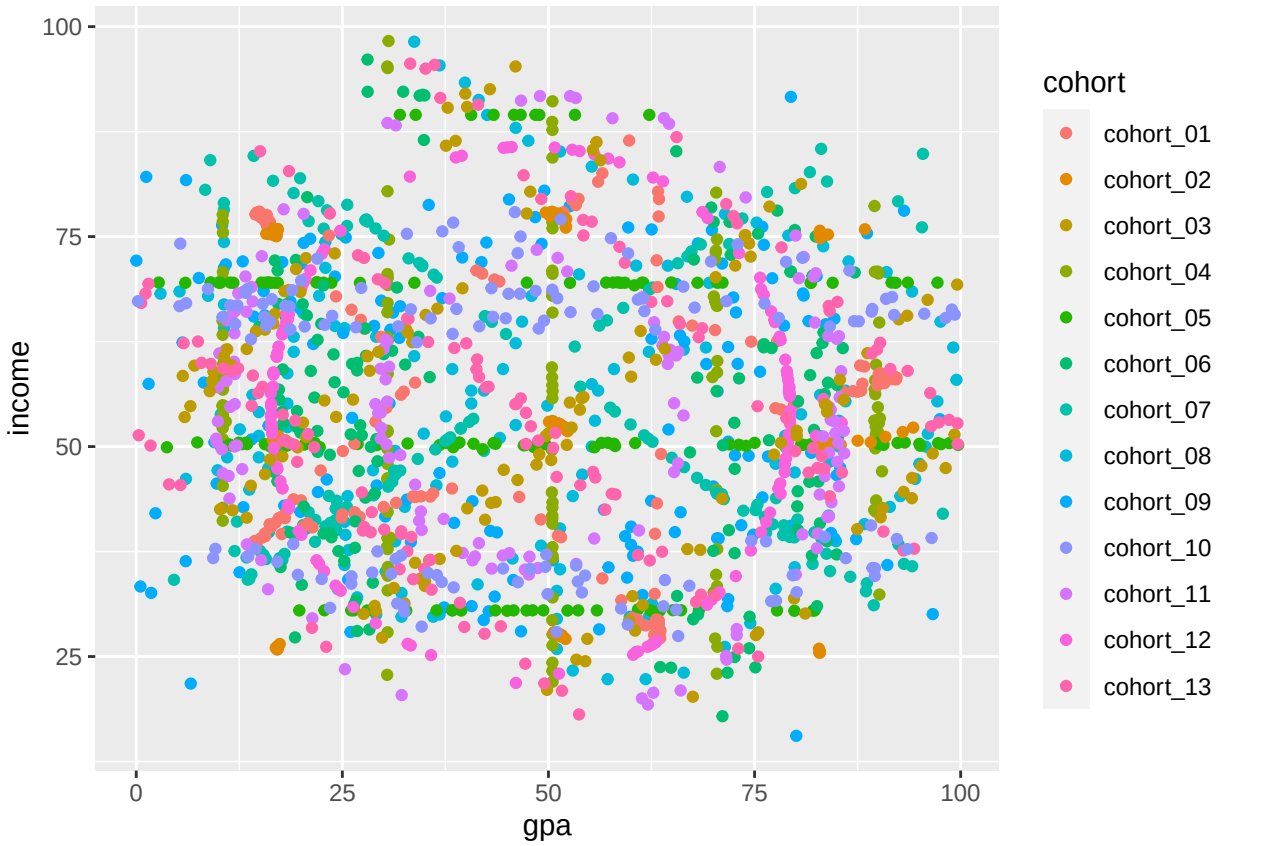
Huh. Those are remarkably similar values for the group means and the group standard deviations... weird.

Onwards to plotting:

```
ggplot(data = grads, aes(x = cohort, y = income)) +
  geom_boxplot()
```

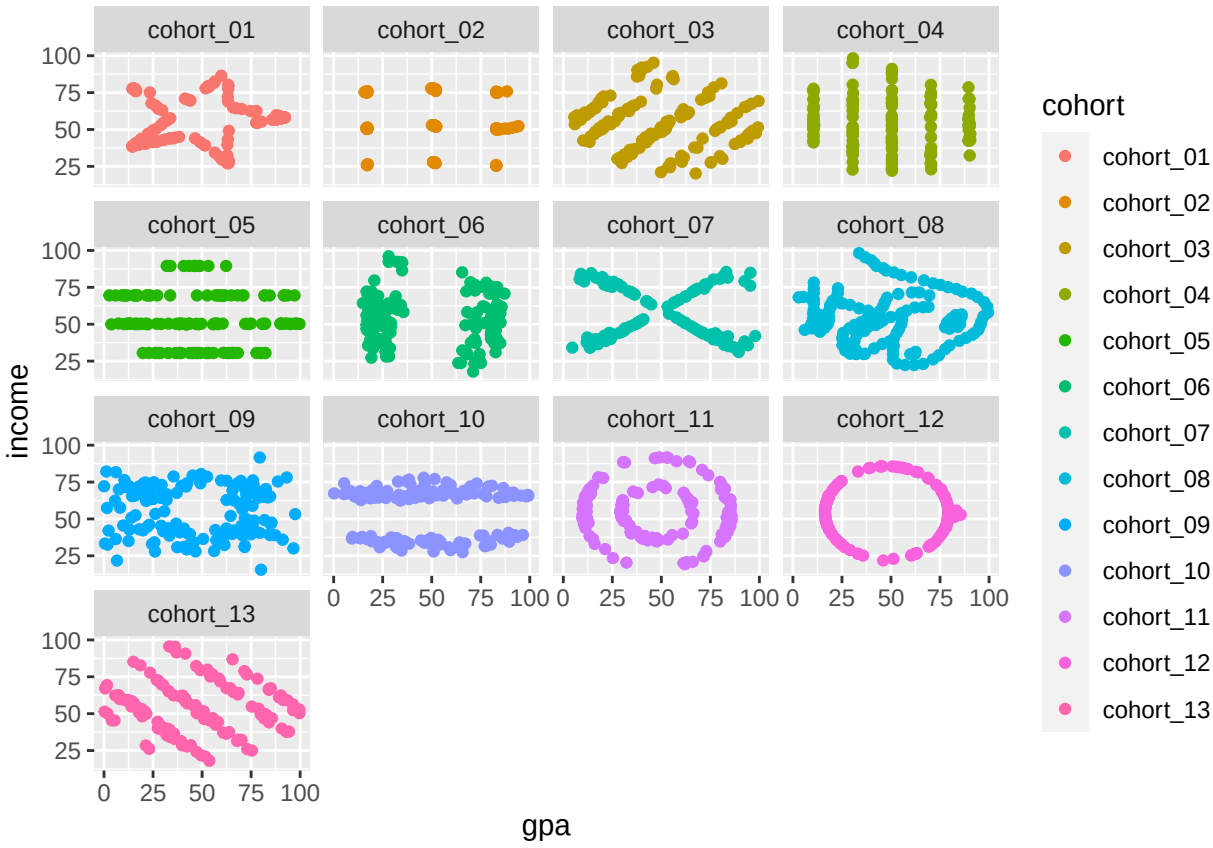



```
ggplot(data = grads, aes(x = gpa, y = income, color = cohort)) +
  geom_point()
```



Those plots are also a little strange. Even though this is just a simulated analysis, it still seems weird that overall the scatterplot just looks like a big mass of points, but when I color the points by district, I can see some regularities within groups. At this point, I might want to facet the scatterplots by district to see any patterns more clearly.

```
ggplot(data = grads, aes(x = gpa, y = income, color = cohort)) +
  geom_point() +
  facet_wrap(vars(cohort))
```



Okay, that's... a joke (in particular, cohort 8 looks like a sideways dinosaur). At this point, if I were really working as a consultant on this project, I would write to the client and start asking some probing questions about data quality (who collected this data? how did it get recorded/stored/etc.? what quality controls were in place?). I would also feel obligated to tell them that I suspect there's just no way the data correspond to the variables they think are here. If you did that and the client was honest, they might tell you where the data actually came from.

In the event that you marched ahead with the analysis and are curious about what that could have looked like, I've provided some example code below. That said, **this is a situation where the assumptions and conditions necessary to identify ANOVA, t-tests, or regression are all pretty broken** because the data was generated programmatically in ways that undermine the kinds of interpretation you've been asked to make. The best response here (IMHO) is to abandon these kinds of analysis once you discover that there's something systematically weird going on like this. While the experience of discovering a scatterplot dinosaur in your data is... unlikely outside of the context of a problem set, there are many situations in which careful data exploration will bring a realization that you just don't understand some important things about the sources or qualities of your data. You have to learn to identify these moments and develop strategies for dealing with them! Often, the statistical procedures will "work" in the sense that they will return a result without any apparent errors, but because those results aren't even close to meaningful, any relationships you do observe in the data reflect something different than the sorts of relationships the statistical procedures were designed to identify.

Fake analysis for fake data

Okay, if you wanted example code to look at for this, here it is. Please just keep in mind that the results are not informative!

```
summary(aov(income ~ cohort, data = grads)) # no global differences of means across groups
```

```
##               Df Sum Sq Mean Sq F value Pr(>F)
## cohort        12      0     0.0       0      1
## Residuals   1833 515354   281.1
```

```
summary(grads.model <- lm(income ~ gpa + cohort, data = grads)) # gpa percentile has a small, negative
```

```
##
## Call:
## lm(formula = income ~ gpa + cohort, data = grads)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -37.381 -13.259  -1.536  13.000  43.362
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)      5.623e+01  1.567e+00  35.894  < 2e-16 ***
## gpa             -4.110e-02  1.451e-02  -2.832  0.00468 **
## cohortcohort_02 -7.026e-03  1.986e+00  -0.004  0.99718
## cohortcohort_03 -1.790e-03  1.986e+00  -0.001  0.99928
## cohortcohort_04 -6.281e-03  1.986e+00  -0.003  0.99748
## cohortcohort_05  2.481e-03  1.986e+00   0.001  0.99900
## cohortcohort_06  1.296e-03  1.986e+00   0.001  0.99948
## cohortcohort_07 -7.184e-03  1.986e+00  -0.004  0.99711
## cohortcohort_08 -4.368e-03  1.986e+00  -0.002  0.99825
## cohortcohort_09 -1.440e-03  1.986e+00  -0.001  0.99942
## cohortcohort_10 -7.512e-04  1.986e+00   0.000  0.99970
## cohortcohort_11  1.030e-03  1.986e+00   0.001  0.99959
## cohortcohort_12 -9.652e-05  1.986e+00   0.000  0.99996
## cohortcohort_13  3.578e-04  1.986e+00   0.000  0.99986
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 16.74 on 1832 degrees of freedom
## Multiple R-squared:  0.004359,   Adjusted R-squared:  -0.002707
## F-statistic: 0.6169 on 13 and 1832 DF,  p-value: 0.8419
```

```
confint(grads.model, "gpa") # 95% confidence interval
```

```
##           2.5 %      97.5 %
## gpa -0.06955974 -0.01263512
```

Note that the failure to reject the null of any association between district and income in the ANOVA would not (even in the event of more realistic data) provide very compelling evidence that the relationship between GPA and income does not vary by cohort. There were several things you might have done here. One is to calculate correlation coefficients within groups. Here's some tidyverse code that does that:

```
grads %>%
  group_by(cohort) %>%
  summarize(
    correlation = cor(income, gpa)
  )
```

```
## # A tibble: 13 x 2
##    cohort    correlation
##    <fct>           <dbl>
##  1 cohort_01     -0.0630
```

```
##  2 cohort_02     -0.0603
##  3 cohort_03     -0.0686
##  4 cohort_04     -0.0617
##  5 cohort_05     -0.0694
##  6 cohort_06     -0.0685
##  7 cohort_07     -0.0656
##  8 cohort_08     -0.0645
##  9 cohort_09     -0.0641
## 10 cohort_10     -0.0666
## 11 cohort_11     -0.0686
## 12 cohort_12     -0.0683
## 13 cohort_13     -0.0690
```

Because these correlation coefficients are nearly identical, I would likely end an analysis here and conclude that the correlation between gpa and income appears to be consistently small and negative. If you wanted to go further, you could theoretically calculate an interaction term in the model (by including `I(gpa*cohort)` in the model formula), but the analysis up to this point gives no indication that you'd be likely to find much of anything (and we haven't really talked about interactions yet). On top of that, there's a literal dinosaur lurking in your data...just give up!

Part III: Trick or treating again

Import and update data

Revisit the text and worked solutions for problem set 7 for more details about the study design, data collection and more.

```
## reminder that the "read_dta()" function requires the "haven" library

df <- read_dta(url("https://communitydata.science/~ads/teaching/2020/stats/data/week_07/Halloween2012-20

df <- df %>%
  mutate(
    obama = as.logical(obama),
    fruit = as.logical(fruit),
    year = as.factor(year),
    male = as.logical(male),
    age = age - mean(age, na.rm = T)
  )

df
```

```
## # A tibble: 1,223 x 7
##    obama fruit year     age male   neob treat_year
##    <lgl> <lgl> <fct>  <dbl> <lgl> <dbl>      <dbl>
##  1 FALSE FALSE 2014  -2.52  FALSE     1          4
##  2 FALSE TRUE  2014  -3.52  FALSE     1          4
##  3 FALSE FALSE 2014   0.480 TRUE      1          4
##  4 FALSE FALSE 2014  -3.52  TRUE      1          4
##  5 FALSE FALSE 2014  -1.52  FALSE     1          4
##  6 FALSE FALSE 2014   0.480 FALSE     1          4
##  7 FALSE FALSE 2014   1.48  TRUE      1          4
##  8 FALSE TRUE  2014  -3.52  FALSE     1          4
##  9 FALSE FALSE 2014  -0.520 TRUE      1          4
## 10 FALSE TRUE  2014  -1.52  FALSE     1          4
```

```
## # ... with 1,213 more rows
```

That looks consistent with what we want here. Let's fit and summarize the model:

```
f <- formula(fruit ~ obama + age + male + year)

fit <- glm(f, data = df, family = binomial("logit"))

summary(fit)
```

```
##
## Call:
## glm(formula = f, family = binomial("logit"), data = df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9079  -0.7853  -0.7319   1.4685   1.8134
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.2509167  0.2180417  -5.737 9.63e-09 ***
## obamaTRUE    0.2355872  0.1385821   1.700   0.0891 .
## age          0.0109508  0.0214673   0.510   0.6100
## maleTRUE    -0.1401293  0.1329172  -1.054   0.2918
## year2014     0.0002101  0.2215401   0.001   0.9992
## year2015     0.2600857  0.2107893   1.234   0.2173
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1374.9  on 1220  degrees of freedom
## Residual deviance: 1367.2  on 1215  degrees of freedom
##   (2 observations deleted due to missingness)
## AIC: 1379.2
##
## Number of Fisher Scoring iterations: 4
```

Interesting. Looks like adjusting for these other variables in a regression setting allows us to uncover some different the results.

Onwards to generating more interpretable results. You might recall that the big problem with interpreting logistic regression is that the results are given to you in "log-odds." Not only is it difficult to have intuitions about odds, but intuitions about the natural logarithms of odds are just intractable (for most of us).

To make things easier, the typical first step is to calculare odds-ratios instead of log-odds. This is done by exponentiating the coefficients (as well as the corresponding 95% confidence intervals):

```
## Odds ratios (exponentiated log-odds!)
exp(coef(fit))
```

```
## (Intercept)   obamaTRUE         age    maleTRUE    year2014    year2015
##   0.2862423   1.2656518   1.0110110   0.8692458   1.0002101   1.2970413
```

```
exp(confint(fit))
```

```
##                 2.5 %    97.5 %
## (Intercept) 0.1846712 0.4348622
```

```
## obamaTRUE   0.9635277 1.6593989
## age         0.9691402 1.0543000
## maleTRUE    0.6697587 1.1280707
## year2014    0.6518923 1.5564945
## year2015    0.8649706 1.9799543
```

You can use these to construct statements about the change in odds of the dependent variable flipping from 0 to 1 (or `FALSE` to `TRUE`) predicted by a 1-unit change in the corresponding predictor (where an odds ratio of 1 corresponds to unchanged odds). We'll interpret the `obamaTRUE` odds ratio below.

Now, model-predicted probabilities for prototypical observations. Recall that it's necessary to create synthetic ("fake"), hypothetical individuals to generate predicted probabilities like these. In this case, I'll create two versions of each fake kid: one assigned to the treatment condition and one assigned to the control. Then I'll use the `predict()` function to generate fitted values for each of the fake kids.

```
fake.kids <- data.frame(
  obama = c(FALSE, FALSE, TRUE, TRUE),
  year = factor(rep(c("2015", "2012"), 2)),
  age = rep(c(9, 7), 2),
  male = rep(c(FALSE, TRUE), 2)
)

fake.kids.pred <- cbind(fake.kids, pred.prob = predict(fit, fake.kids, type = "response"))

fake.kids.pred
```

```
##   obama year age  male pred.prob
## 1 FALSE 2015   9 FALSE 0.2906409
## 2 FALSE 2012   7  TRUE 0.2117531
## 3  TRUE 2015   9 FALSE 0.3414844
## 4  TRUE 2012   7  TRUE 0.2537326
```

Note that this UCLA logit regression tutorial also contains example code to help extract standard errors and confidence intervals around these predicted probabilities. You were not asked to create them here, but if you'd like an example here you go. The workhorse is the `plogis()` function, which essentially does the inverse logit transformation detailed in the textbook chapter 8:

```
fake.kids.more.pred <- cbind(
  fake.kids,
  predict(fit, fake.kids, type = "link", se = TRUE)
)

within(fake.kids.more.pred, {
  pred.prob <- plogis(fit)
  lower <- plogis(fit - (1.96 * se.fit))
  upper <- plogis(fit + (1.96 * se.fit))
})
```

```
##   obama year age  male        fit    se.fit residual.scale     upper     lower
## 1 FALSE 2015   9 FALSE -0.8922736 0.2243091              1 0.3887361 0.2088421
## 2 FALSE 2012   7  TRUE -1.3143903 0.2682218              1 0.3124531 0.1370389
## 3  TRUE 2015   9 FALSE -0.6566864 0.2363758              1 0.4518027 0.2460144
## 4  TRUE 2012   7  TRUE -1.0788031 0.2594135              1 0.3611555 0.1697706
##   pred.prob
## 1 0.2906409
## 2 0.2117531
## 3 0.3414844
```

```
## 4 0.2537326
```

Sub-group analysis

To do this, we'll need to create a slightly new model formula that drops the `year` term since we're going to restrict each subset of the data along that dimension.

Once I fit the models, I'll use the `stargazer` package to create a reasonably pretty regression table that incorporates all three summaries.

```
f2 <- formula(fruit ~ obama + age + male)

m2012 <- glm(f2, data = df[df$year == "2012", ], family = binomial("logit"))
m2014 <- glm(f2, data = df[df$year == "2014", ], family = binomial("logit"))
m2015 <- glm(f2, data = df[df$year == "2015", ], family = binomial("logit"))


## I can make a pretty table out of that:
library(stargazer)
stargazer(m2012, m2014, m2015, column.labels = c("2012", "2014", "2015"), type = "text")
```

```
##
## ================================================
##                         Dependent variable:
##                     ---------------------------
##                                 fruit
##                       2012      2014      2015
##                        (1)       (2)       (3)
## ------------------------------------------------
## obama                 0.326     0.073    0.317*
##                      (0.387)   (0.242)   (0.190)
##
## age                   0.085     0.012    0.0005
##                      (0.067)   (0.037)   (0.029)
##
## male                  0.322    -0.240    -0.179
##                      (0.388)   (0.235)   (0.179)
##
## Constant            -1.538*** -1.134*** -0.997***
##                      (0.377)   (0.198)   (0.138)
##
## ------------------------------------------------
## Observations           164       422       635
## Log Likelihood      -87.451  -224.427  -370.002
## Akaike Inf. Crit.   182.902   456.853   748.005
## ================================================
## Note:                *p<0.1; **p<0.05; ***p<0.01
```

Interpret and discuss

Well, for starters, the model providing a "pooled" estimate of treatment effects while adjusting for age, gender, and study year suggests that the point estimate is "marginally" statistically significant ($p < 0.1$) indicating some evidence that the data support the alternative hypothesis (being shown a picture of Michelle Obama causes trick-or-treaters to be more likely to pick up fruit than the control condition). In more concrete terms, the trick-or-treaters shown the Obama picture were, on average, about 26% more likely to pick up fruit

than those exposed to the control (95% CI: -4% $-$ $+66\%$).[1] In even more concrete terms, the estimated probability that a 9 year-old girl in 2015 and a 7 year-old boy in 2012 would take fruit increase about 17% and 19% respectively on average (from 29% to 34% in the case of the 9 year-old and from 21% to 25% in the case of the 7 year-old). These findings are sort of remarkable given the simplicity of the intervention and the fairly strong norm that Halloween is all about candy.

All of that said, the t-test results from Problem set 5 and the "unpooled" results reported in the sub-group analysis point to some potential concerns and limitations. For starters, the fact that the experiment was run iteratively over multiple years and that the sample size grew each year raises some concerns that the study design may not have anticipated the small effect sizes eventually observed and/or was adapted on the fly. This would undermine confidence in some of the test statistics and procedures. Furthermore, because the experiment occurred in sequential years, there's a very real possibility that the significance of a picture of Michelle Obama shifted during that time period and/or the house in question developed a reputation for being "that weird place where they show you pictures of Michelle Obama and offer you fruit." Whatever the case, my confidence in the findings here is not so great and I have some lingering suspicions that the results might not replicate.

On a more nuanced/advanced statistical note, I also have some concerns about the standard errors. This goes beyond the content of our course, but basically, a randomized controlled trial introduces clustering into the data by-design (you can think of it as analogous to the observations coming from the treatment "cluster" and the control "cluster"). In this regard, the normal standard error formulas can be biased. Luckily, there's a fix for this: compute "robust" standard errors as a result and re-calculate the corresponding confidence intervals. Indeed, robust standard errors are often considered to be the best choice even when you don't know about potential latent clustering or heteroskedastic error structures in your data. This a short pdf provides a little more explanation, citations, as well as example R code for how you might calculate robust standard errors.

[1] Remember when I said we would use those odds ratios to interpret the parameter on `obamaTRUE`? Here we are. The parameter value is approximately 1.26, which means that the odds of picking fruit are, on average, 1.26 times as large for a trick-or-treater exposed to the picture of Michelle Obama versus a trick-or-treater in the control condition. In other words, the odds go up by about 26% ($= \frac{1.26-1}{1}$).