Data Exploration Project

Data Exploration Project

In this project, I will be cleaning up and displaying data from a dataset on layoffs from companies around the world.

Cleaning Up

First, I will import the data set and start to explore the various pieces of information I have. Next, I want to make sure that my various pieces of information are the correct type–specifically, I would like for my integers/numbers to be integers/numbers instead of chr datatypes. This can make things easier for me to do work with my data.

Specifically, my goal is to look for any gaps in the data that might pose an issue. This includes:

- looking for duplicates
- standardize data, ensuring that similar cities are formatted the same way (NYC versus New York versus New York City)
- Analyze the blank/null values and ensure that they are properly assigned. If necessary, assign other values NULL
- Other oddities I find in the data

Read in the data, and change the types from chr -> int

```
library(readr)
library(dplyr)
```

```
Attaching package: 'dplyr'
```

```
The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```r
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2

-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

```r
library(moderndive)
library(broom)
layoffs_staging <- read_csv("/Users/rrahman/Downloads/layoffs.csv")
```

```
Rows: 2361 Columns: 9
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (9): company, location, industry, total_laid_off, percentage_laid_off, d...

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.
```

```r
# check to see how many unique values
# are in the dataset for the number of
# people laid off.
layoffs_staging  %>%
  count(total_laid_off) %>%
```

```r
  arrange(total_laid_off)
```

```
# A tibble: 286 x 2
   total_laid_off     n
   <chr>          <int>
 1 10                17
 2 100               97
 3 1000              10
 4 10000              2
 5 101                2
 6 104                2
 7 105                1
 8 108                1
 9 109                2
10 11                 6
# i 276 more rows
```

```r
# cast the total_laid_off and funds_raised_millions
# columns from chr to integer
layoffs_staging$total_laid_off <- as.integer(layoffs_staging$total_laid_off)
```

```
Warning: NAs introduced by coercion
```

```r
layoffs_staging$funds_raised_millions <- as.integer(layoffs_staging$funds_raised_millions)
```

```
Warning: NAs introduced by coercion
```

```r
#in the head, we can see that the type has changed
head(layoffs_staging)
```

```
# A tibble: 6 x 9
  company       location industry total_laid_off percentage_laid_off date  stage
  <chr>         <chr>    <chr>             <int> <chr>               <chr> <chr>
1 Atlassian     Sydney   Other               500 0.05                3/6/~ Post~
2 SiriusXM      New Yor~ Media               475 0.08                3/6/~ Post~
3 Alerzo        Ibadan   Retail              400 NULL                3/6/~ Seri~
4 UpGrad        Mumbai   Educati~            120 NULL                3/6/~ Unkn~
5 Loft          Sao Pau~ Real Es~            340 0.15                3/3/~ Unkn~
6 Embark Trucks SF Bay ~ Transpo~            230 0.7                 3/3/~ Post~
# i 2 more variables: country <chr>, funds_raised_millions <int>
```

Analyze duplicates

```r
# Let's see what the duplicates are in the dataset
layoffs_staging[duplicated(layoffs_staging), ]
```

```
# A tibble: 5 x 9
  company       location industry total_laid_off percentage_laid_off date  stage
  <chr>         <chr>    <chr>             <int> <chr>               <chr> <chr>
1 Cazoo         London   Transpo~            750 0.15                6/7/~ Post~
2 Yahoo         SF Bay ~ Consumer           1600 0.2                 2/9/~ Acqu~
3 Hibob         Tel Aviv HR                   70 0.3                 3/30~ Seri~
4 Casper        New Yor~ Retail               NA NULL                9/14~ Post~
5 Wildlife Stu~ Sao Pau~ Consumer            300 0.2                 11/2~ Unkn~
# i 2 more variables: country <chr>, funds_raised_millions <int>
```

```r
# remove duplicates
layoffs_staging <- layoffs_staging %>% distinct()

# Now, we want to determine if there are any
# misspelled or duplicate industry names.

layoffs_staging %>%
  count(industry)
```

```
# A tibble: 34 x 2
   industry            n
   <chr>           <int>
 1 Aerospace           6
 2 Construction       16
 3 Consumer          116
 4 Crypto             99
 5 Crypto Currency     2
 6 CryptoCurrency      1
 7 Data               79
 8 Education          93
 9 Energy             12
10 Fin-Tech            3
# i 24 more rows
```

```r
# First, we want to make all of the Crypto, Crypto Currency,
# and CryptoCurrency to be the same.
layoffs_staging <- layoffs_staging %>%
  mutate(industry =
    case_when(
      industry == "CryptoCurrency" ~ "Crypto",
      industry == "Crypto Currency" ~ "Crypto",
      .default = industry
    )
  )
```

Analyze the NULLs in company

It's important to now that when we imported the csv, all the NULL values became strings called "NULL" instead of the R equivalent, NA. We are lucky that there's no companies, industries, location, or other important entities called NULL that could mix up the data. If there were, we would have to analyze each instance of NULL separately.

```r
# We also want to examine the NULL company to see
# if we can somehow this with other categories,
# or understand why it is categorized as NULL.

layoffs_staging %>%
  filter(industry %in% c("NULL"))
```

```
# A tibble: 1 x 9
  company       location industry total_laid_off percentage_laid_off date  stage
  <chr>         <chr>    <chr>             <int> <chr>               <chr> <chr>
1 Bally's Inte~ Provide~ NULL                 NA 0.15                1/18~ Post~
# i 2 more variables: country <chr>, funds_raised_millions <int>
```

```r
# the company appears to be an online casino. It falls
# most appropriately under the category of "Other"

layoffs_staging <- layoffs_staging %>%
  mutate(
    industry =
      case_when(
        industry == "NULL" ~ "Other",
        .default = industry
```

```
    )
  )

# now we see that the single NULL company is gone.
layoffs_staging %>%
  filter(industry %in% c("NULL"))
```

```
# A tibble: 0 x 9
# i 9 variables: company <chr>, location <chr>, industry <chr>,
#   total_laid_off <int>, percentage_laid_off <chr>, date <chr>, stage <chr>,
#   country <chr>, funds_raised_millions <int>
```

Analyze the NULLs in industry

Upon observation, these companies do fall under industries already in our dataset. In fact, let's check to see if there are any companies that categorize themselves under multiple industries. The same company should not be under different industries.

```
# let's take a look at all the companies with
# NA/blank values for their industry
layoffs_staging %>%
  filter(is.na(industry))
```

```
# A tibble: 3 x 9
  company location    industry total_laid_off percentage_laid_off date     stage
  <chr>   <chr>       <chr>             <int> <chr>               <chr>    <chr>
1 Airbnb  SF Bay Area <NA>                 30 NULL                3/3/2023 Post~
2 Juul    SF Bay Area <NA>                400 0.3                 11/10/2~ Unkn~
3 Carvana Phoenix     <NA>               2500 0.12                5/10/20~ Post~
# i 2 more variables: country <chr>, funds_raised_millions <int>
```

```
companies <- unique(layoffs_staging$company)
multipleInd <- c()
for (com in companies){
  industries <- layoffs_staging %>%
    filter(company == com) %>%
    select(industry)
  uniqueInd <- unique(industries)
    if (dim(uniqueInd)[1] != 1){
```

```
    multipleInd <- append(multipleInd,com)
  }
}
multipleInd
```

```
 [1] "Airbnb"      "Bolt"        "PeerStreet"  "LinkedIn"    "Noom"
 [6] "Hubilo"      "RingCentral" "ShareChat"   "Carvana"     "Carta"
[11] "100 Thieves" "Wonder"      "Juul"        "Code42"      "Nuri"
[16] "Sea"         "Pollen"      "Glossier"    "Clearco"     "People.ai"
[21] "OneTrust"    "Domio"
```

It appears that to remedy this issue, we would need to adjust the company names by entry. If we wanted to populate them automatically, we would fall into an issue. Let's say Airbnb is categorized under "Travel" and "Hospitality"; what is most appropriate? We need to make individual decisions to make sure this is accurate. An easy fix would be to simply use the industry of the first instance of the company in the dataset.

```
layoffs_staging %>%
  filter(company %in% multipleInd)
```

```
# A tibble: 53 x 9
   company     location  industry total_laid_off percentage_laid_off date  stage
   <chr>       <chr>     <chr>             <int> <chr>               <chr> <chr>
 1 Airbnb      SF Bay A~ <NA>                 30 NULL                3/3/~ Post~
 2 Bolt        Lagos     Transpo~             17 NULL                2/21~ Seri~
 3 PeerStreet  Los Ange~ Real Es~             NA NULL                2/21~ Seri~
 4 LinkedIn    SF Bay A~ HR                   NA NULL                2/13~ Acqu~
 5 Noom        New York~ Fitness              NA NULL                1/25~ Seri~
 6 Bolt        SF Bay A~ Finance              50 0.1                 1/24~ Seri~
 7 Hubilo      SF Bay A~ Other               115 0.35                1/19~ Seri~
 8 RingCentral SF Bay A~ Other                30 NULL                1/17~ Post~
 9 ShareChat   Bengaluru Consumer            500 0.2                 1/16~ Seri~
10 Carvana     Phoenix   Transpo~             NA NULL                1/13~ Post~
# i 43 more rows
# i 2 more variables: country <chr>, funds_raised_millions <int>
```

```
layoffs_staging <- layoffs_staging %>%
  mutate(industry =
           case_when(
```

```r
                company == "Airbnb" ~ "Travel",
                company %in% c("100 Thieves", "Glossier", "Juul", "ShareChat") ~ "Consumer",
                company %in% c("Carta", "Clearco") ~ "Finance",
                company == "Carvana" ~ "Transportation",
                company == "Code42" ~ "Security",
                company %in% c("Pollen", "Domio") ~ "Travel",
                company == "Hubilo" ~ "Marketing",
                company == "LinkedIn" ~ "Recruiting",
                company == "Noom" ~ "Fitness",
                company == "Nuri" ~ "Crypto",
                company == "OneTrust" ~ "Security",
                company == "PeerStreet" ~ "Real Estate",
                company == "People.ai" ~ "Sales",
                company == "RingCentral" ~ "Support",
                company == "Wonder" ~ "Food",
                .default = industry
              ))
```

Consistency in country names.

Let's look at the different countries represented in our dataset. Note that there is "United States" and "United States.", which is not convenient. Let's change this so that "United States" is the country name, not "United States."

```r
# Let's look at the different countries represented in our dataset.
layoffs_staging %>%
  count(country)
```

```
# A tibble: 60 x 2
   country       n
   <chr>     <int>
 1 Argentina     6
 2 Australia    54
 3 Austria       3
 4 Bahrain       1
 5 Belgium       1
 6 Brazil       76
 7 Bulgaria      1
 8 Canada       99
 9 Chile         3
10 China        18
```

```
# i 50 more rows
```

```r
layoffs_staging <- layoffs_staging %>%
  mutate(country =
           case_when(
             country == "United States." ~ "United States",
             .default = country
           ))

# As expected, there are no longer multiple categories for "United States"
layoffs_staging %>%
  filter(country == "United States.")
```

```
# A tibble: 0 x 9
# i 9 variables: company <chr>, location <chr>, industry <chr>,
#   total_laid_off <int>, percentage_laid_off <chr>, date <chr>, stage <chr>,
#   country <chr>, funds_raised_millions <int>
```

Change date from a string to date datatype

To make working with our data easier, we should change our date category from text to a date datatype.

I wanted to check and see if there were any NA values. This may indicate that there was a NULL value for the original csv file.

```r
layoffs_staging[1,]
```

```
# A tibble: 1 x 9
  company   location industry total_laid_off percentage_laid_off date     stage
  <chr>     <chr>    <chr>             <int> <chr>               <chr>    <chr>
1 Atlassian Sydney   Other               500 0.05                3/6/2023 Post-~
# i 2 more variables: country <chr>, funds_raised_millions <int>
```

```r
layoffs_staging$date <- as.Date(layoffs_staging$date, format = "%m/%d/%Y")
sapply(layoffs_staging, class)
```

```
               company              location              industry
           "character"           "character"           "character"
        total_laid_off   percentage_laid_off                  date
             "integer"           "character"                "Date"
                 stage               country funds_raised_millions
           "character"           "character"             "integer"
```

```
layoffs_staging %>%
  filter(is.na(date))
```

```
# A tibble: 1 x 9
  company  location industry total_laid_off percentage_laid_off date   stage
  <chr>    <chr>    <chr>             <int> <chr>               <date> <chr>
1 Blackba~ Charles~ Other               500 0.14                NA     Post-IPO
# i 2 more variables: country <chr>, funds_raised_millions <int>
```

```
# If we don't have information on the date of being laid off,
# then we might choose to not include this data
# layoffs_staging <- layoffs_staging[(!layoffs_staging$company == "Blackbaud"), ]
# And now, the company is gone!
# layoffs_staging %>%
# filter(company == "Blackbaud")
```

Adjusting the stage category so that NULL, Other, and Unknown are all NA.

I'm not interested in the distinction between not knowing the stage (Unknown), the stage data not being collected in the first place (NULL), or the stage not fitting into specific categories, which is why I've put them all to NA.

```
layoffs_staging <- layoffs_staging %>%
  mutate(
    stage =
      case_when(
        stage == "NULL" ~ NA,
        stage == "Unknown" ~ NA,
        stage == "Other" ~ NA,
        .default = stage
      )
  )
```

What if we don't have information on being laid off?

It's possible that one of the rows in our dataset has neither `total_laid_off` and `percentage_laid_off`, which means that we aren't getting productive information from this row of data. The objective of this dataset is to inform us of the relationship between layoffs and various factors in the dataset, like date or location.

So, let's remove any rows that have neither of the two columns filled in. To do so, I will make a subset of the dataset to include only those rows with `NA` for the `total_laid_off` column and another subset of the dataset to include only those rows with "NULL" for the `percentage_laid_off` column. I didn't yet cast this percentage to string, which is why I'm checking for "NULL" instead of `NA` . I will find the intersection of these two sets, and remove that intersection from the dataset.

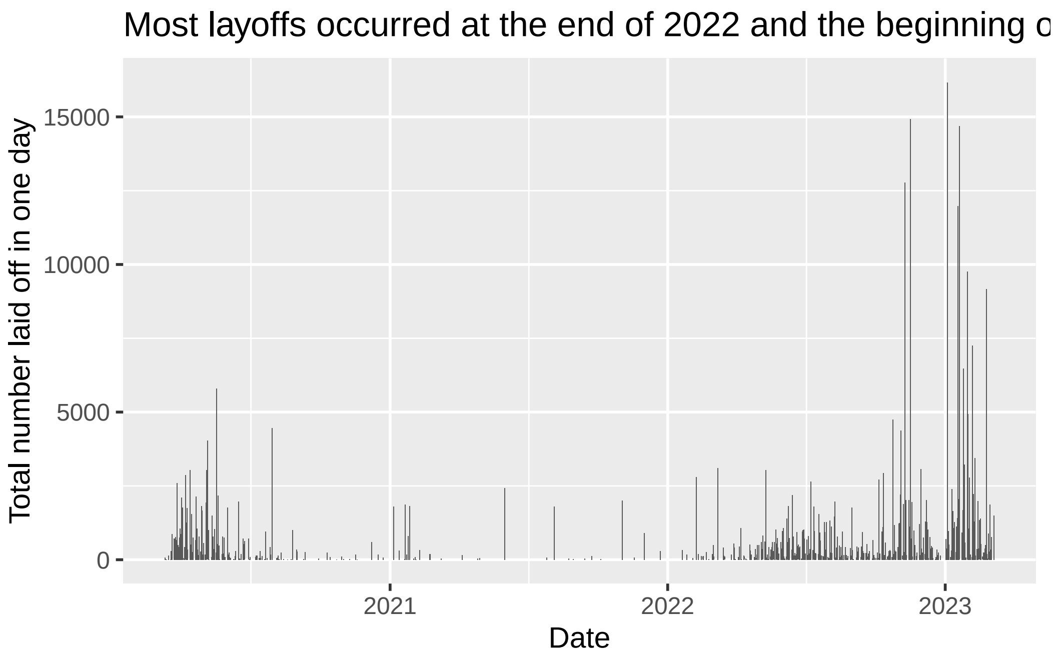
```r
# find overlap of NA and null before double casting
total_laid_off_na <- subset(layoffs_staging,is.na(total_laid_off))
percentage_laid_off_na <- subset(layoffs_staging, percentage_laid_off == "NULL")
both_na <- intersect(total_laid_off_na, percentage_laid_off_na)
layoffs_staging <- setdiff(layoffs_staging, both_na)

# cast from str to double for the percentage
layoffs_staging$percentage_laid_off <- as.double(layoffs_staging$percentage_laid_off)
```

```
Warning: NAs introduced by coercion
```

Exploratory graphics

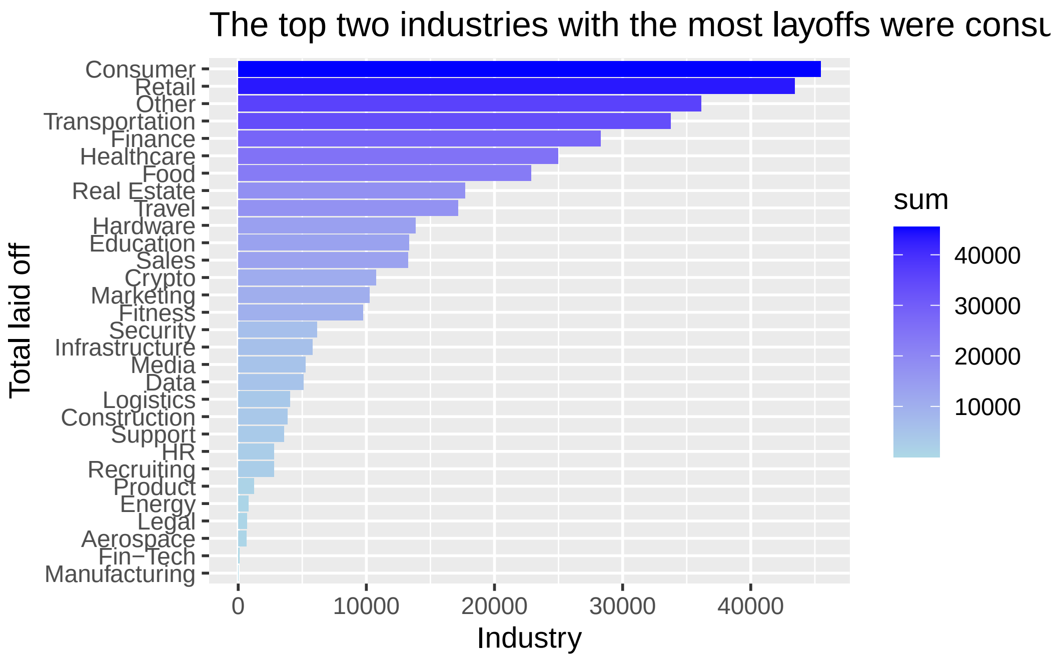
```r
layoffs_staging %>%
  group_by(date) %>%
  drop_na(total_laid_off) %>%
  summarize(sum = sum(total_laid_off)) %>%
  ggplot(aes(x = date, y =sum ))+
  geom_bar(stat = "identity") +
  labs(title = "Most layoffs occurred at the end of 2022 and the beginning of 2023",
       y = "Total number laid off in one day",
       x = "Date")
```

```
Warning: Removed 1 row containing missing values or values outside the scale range
(`geom_bar()`).
```




```
layoffs_staging %>%
  group_by(industry) %>%
  drop_na(total_laid_off) %>%
  summarize(sum = sum(total_laid_off)) %>%
  ggplot(aes(y= reorder(industry, sum), x =sum, fill = sum))+
  geom_col(stat = "identity") +
  scale_fill_gradient(high="blue",low="lightblue") +
  labs(title = "The top two industries with the most layoffs were consumer and retail",
       y = "Total laid off",
       x = "Industry")
```

```
Warning in geom_col(stat = "identity"): Ignoring unknown parameters: `stat`
```

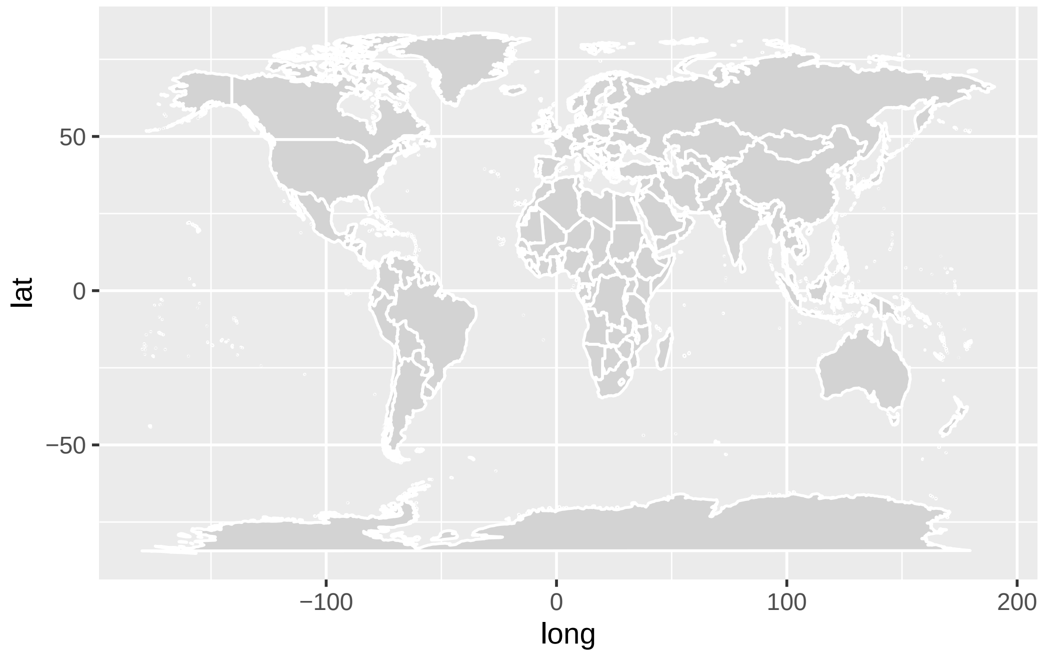


Attempt to show geographically with a map

Previous attempt to make a map of layoffs colored based on the total number of those laid off.

```
world_map <- map_data("world")
layoffs_staging %>%
  group_by(location) %>%
  drop_na(total_laid_off) %>%
  summarize(sum = sum(total_laid_off)) %>%
  ggplot(world_map, aes(x = long, y = lat, group = group, fill=sum)) +
  geom_polygon()
```

```
# need to alter data to work here
#layoffs.exp.map <- left_join(layoffs_staging, world_map, by = "region")
world_map <- map_data("world")
  ggplot(world_map, aes(x = long, y = lat, group = group)) +
  geom_polygon(fill = "lightgray", colour = "white")
```



Attempt to conduct regression analysis

There are 4 assumptions to conduct linear regression; without fulfilling these assumptions, linear regression will not be effective. This includes:

1. Linearity of the data

   We can determine this by examining a plot of the predictor and response variables

2. Constant variability of the response variable as the predictor changes

   We can determine this by examining a residual plot of the y-axis and the predicted values on the x-axis

3. Independent observations

   We can assume that the 3rd condition is met, although this is a little naive. It's possible that previous layoffs can impact future layoffs, but I cannot determine this from the data we have at the moment. This condition is dependent on the nature of your data.

4. Normality of the residuals.

   Create a normal probability plot (also known as a Q-Q plot) of the residuals.

Since there are more rows which have `total_laid_off` than `percentage_laid_off`, I'll try to use regression to predict the `total_laid_off`.

```r
layoffs_staging %>%
  drop_na(total_laid_off) %>%
  nrow()
```

```
[1] 1617
```

```r
layoffs_staging %>%
  drop_na(percentage_laid_off) %>%
  nrow()
```

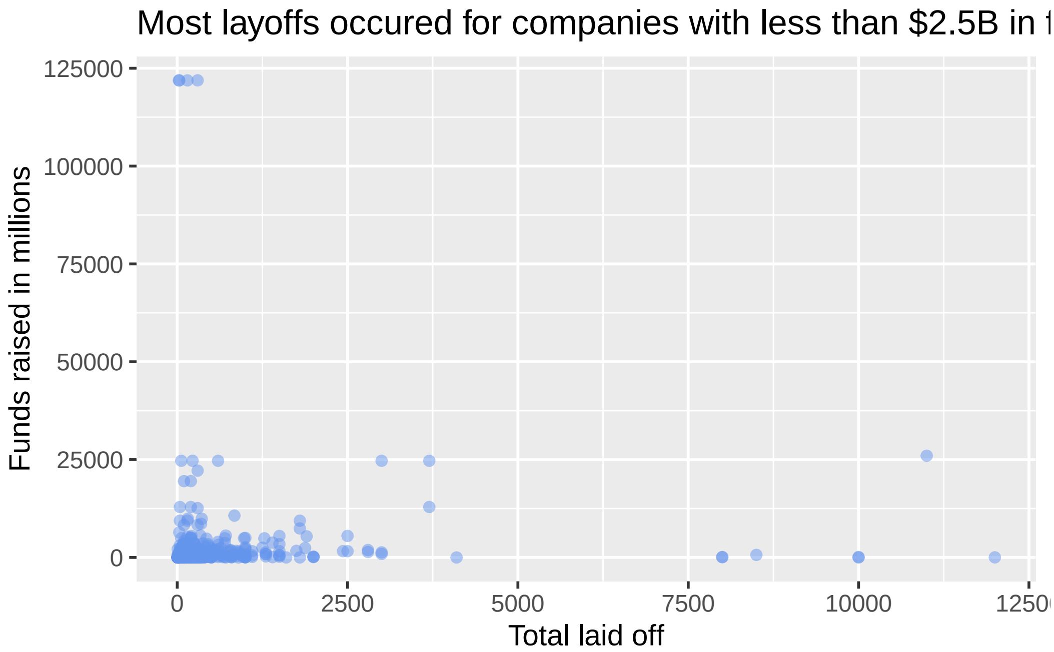
```
[1] 1572
```

Of the one quantitative variable I can conduct regression on (`funds_raised_millions`), the correlation is very low, and there doesn't appear to be any clear pattern in the data. Our condition of linearity isn't met. So, this isn't a promising variable to make predictions on.

```r
layoffs_staging %>%
  drop_na(total_laid_off, funds_raised_millions) %>%
  summarize(cor = cor(total_laid_off, funds_raised_millions))
```

```
# A tibble: 1 x 1
     cor
   <dbl>
1 0.0772
```

```r
layoffs_staging %>%
  drop_na(total_laid_off, funds_raised_millions) %>%
  ggplot(aes(total_laid_off,funds_raised_millions)) +
  geom_point(alpha =  0.5, col = "cornflowerblue") +
  labs(
    title = "Most layoffs occured for companies with less than $2.5B in funds raised.",
    x = "Total laid off",
    y = "Funds raised in millions"
  )
```

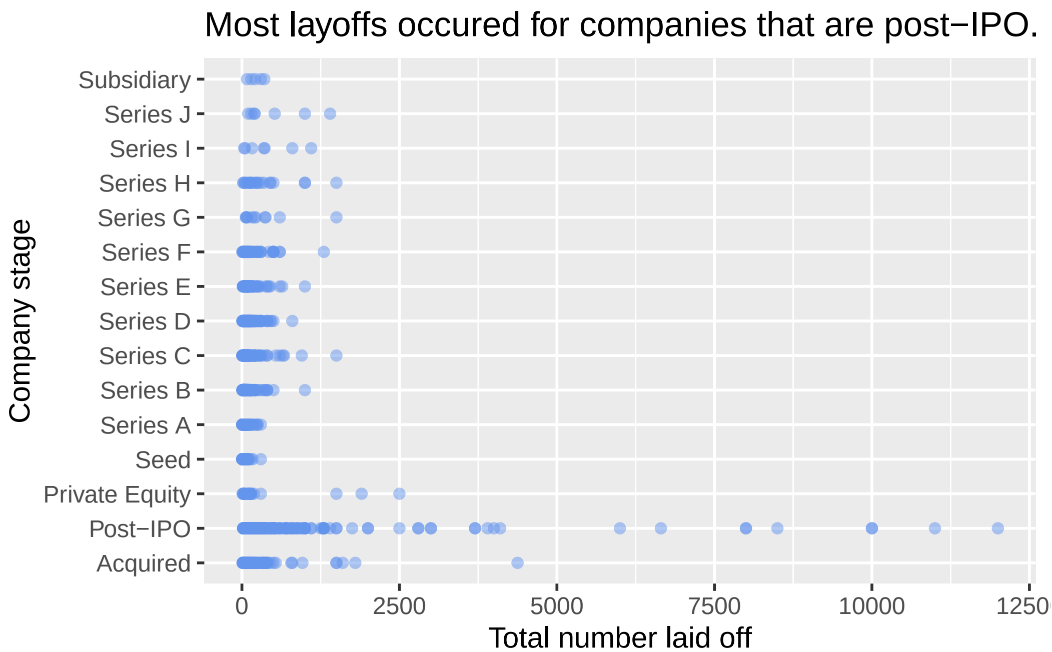


It seems that across the different stages a company can be in, some stages have more laid off than others, which can be a good start to analyzing this data.

```
layoffs_staging %>%
  drop_na(total_laid_off, funds_raised_millions) %>%
  summarize(cor = cor(total_laid_off, funds_raised_millions))
```

```
# A tibble: 1 x 1
     cor
   <dbl>
1 0.0772
```

```
layoffs_staging %>%
  drop_na(total_laid_off, stage) %>%
  ggplot(aes(total_laid_off, stage)) +
  geom_point(alpha =  0.5, col = "cornflowerblue") +
  labs(
    title = "Most layoffs occured for companies that are post-IPO.",
    x = "Total number laid off",
    y = "Company stage",
  )
```



This model attempts to explain the number of those laid off based on the stage of a company. We can interpret the coefficients and information provided:

- The intercept represents the number of people that would be laid off if a company was not in any of these stage categories.
- For a given category x, the β coefficient represents the average increase in the number of those laid off if a company is in that category.

So, `stage: Post-IPO` means that the average number of those laid off from a company that is in the Post-IPO stage is approximately 433 more than the average number of those laid off from a company that isn't classified under any business stage (which is approximately 240).

To understand how much of the variation in the response variable (or, the trends in the number of people laid off) is explained by our linear regression model, we can calculate the R^2 value, which is calculated by taking

$$\frac{\text{Var}(\widehat{\text{predicted number of those laid off}})}{\text{Var}(\text{number of those laid off})}$$

We can also use the adjusted R^2 value, which is calculated similar to the R^2 value, but has an additional penalty term which ensures that the measure will not increase if the predictor (in this case, business stage) does not contribute much to explaining the variation in the response (number of those laid off).

We see here that the R^2 value is 0.08 and the adjusted R^2 value is 0.07. This means that between 7-8% of the variation in the number of those laid off is explained by their business stage.

We can also interpret the p-value for this R^2 value. It is the probability that we would receive this R^2 value at random for the model at hand. Since the p-value is incredibly small, this means that it is unlikely we would receive this R^2 value at random for the model and data that we have available.

But this doesn't sound great–we would like to have a model that's more effective. Let's investigate one possible reason why this model did not work effectively.

```
model <- lm(total_laid_off ~ stage, data = layoffs_staging)
get_regression_table(model)
```

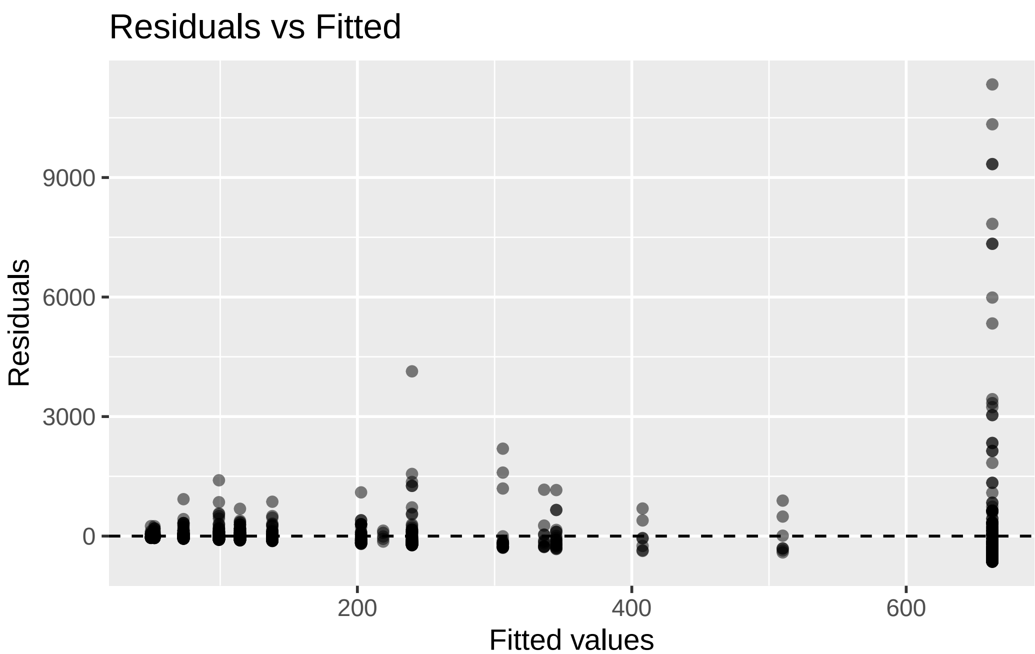
```
# A tibble: 15 x 7
   term                   estimate std_error statistic p_value lower_ci upper_ci
   <chr>                     <dbl>     <dbl>     <dbl>   <dbl>    <dbl>    <dbl>
 1 intercept                 240.       74.1     3.24    0.001     94.5    385.
 2 stage: Post-IPO           423.       86.8     4.87    0         253.    593.
 3 stage: Private Equity      66.2     172.      0.384   0.701    -272.    405.
 4 stage: Seed              -190.      157.     -1.21    0.225    -498.    117.
 5 stage: Series A          -188.      106.     -1.77    0.077    -396.     20.6
 6 stage: Series B          -167.       92.2    -1.81    0.071    -347.     14.4
 7 stage: Series C          -141.       92.8    -1.52    0.13     -323.     41.3
 8 stage: Series D          -125.       96.1    -1.30    0.192    -314.     63.2
 9 stage: Series E          -102.      111.     -0.916   0.36     -320.    116.
10 stage: Series F           -37.1     136.     -0.274   0.784    -303.    229.
11 stage: Series G            96.3     251.      0.384   0.701    -395.    588.
12 stage: Series H           105.      188.      0.558   0.577    -265.    475.
13 stage: Series I           168.      309.      0.544   0.587    -439.    775.
14 stage: Series J           270.      309.      0.874   0.382    -336.    877.
15 stage: Subsidiary         -21.0     363.     -0.058   0.954    -733.    691.
```

```
glance(model)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic  p.value    df  logLik    AIC    BIC
      <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>   <dbl>  <dbl>  <dbl>
1    0.0807        0.0711  794.      8.44 1.02e-17    14 -11020. 22071. 22155.
# i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

As we can see from the residuals, the variance is not constant. The vertical spread of the points is not constant; closer to the 600 mark on the x axis, we can see that there is a wide spread in variability. So, second condition (of constant variability) for using this model fails.

```
library(gglm)
ggplot(model) +
  stat_fitted_resid(alhpa = 0.4)
```



We can continue to make many types of models from the factors that we have in our data set. Unfortunately, we will find that linear regression and multiple linear regression are all ineffective ways to model our data (as evaluated by the R^2 and adjusted R^2 values). In the following code blocks, I will provide the linear regression model, the R^2 and adjusted R^2 values, and a graphic captioned with the reasoning why this model failed.

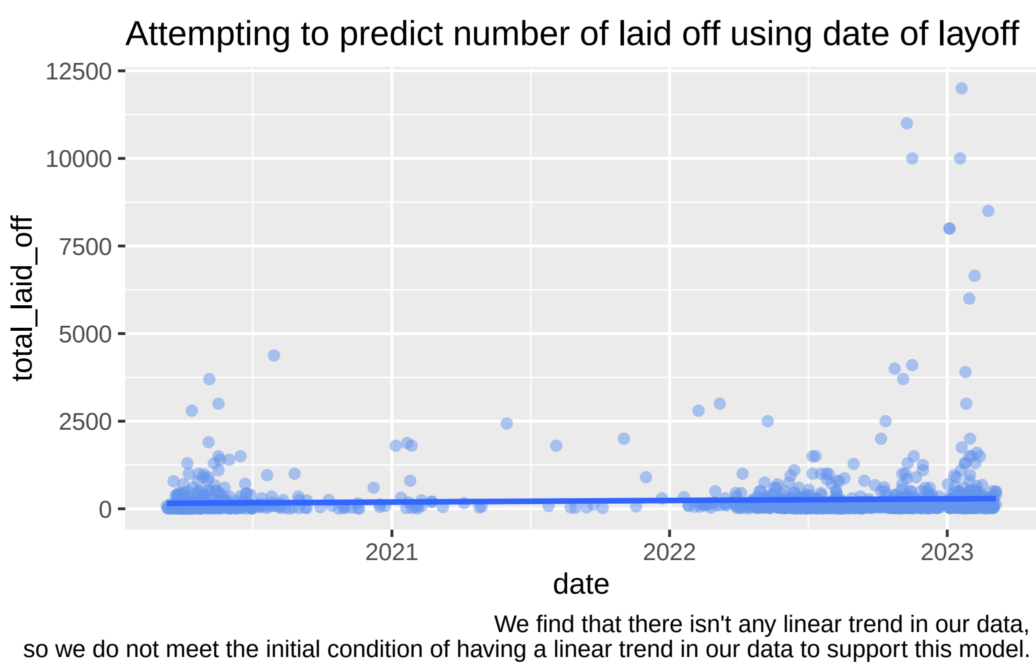
```
model4 <- lm(total_laid_off ~ date, data = layoffs_staging)
get_regression_table(model4)
```

```
# A tibble: 2 x 7
  term      estimate std_error statistic p_value  lower_ci upper_ci
  <chr>        <dbl>     <dbl>     <dbl>   <dbl>     <dbl>    <dbl>
1 intercept -2114.      885.       -2.39   0.017 -3850.    -378.
2 date          0.124     0.047      2.66   0.008     0.032    0.215
```

```r
glance(model4)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic p.value    df  logLik    AIC    BIC
      <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>   <dbl>  <dbl>  <dbl>
1   0.00435       0.00374  769.      7.06 0.00797     1 -13030. 26065. 26081.
# i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```r
layoffs_staging %>%
  drop_na(total_laid_off, date) %>%
  ggplot(aes( date, total_laid_off)) +
  geom_point(alpha =  0.5, col = "cornflowerblue") +
  stat_smooth(method = lm, formula = y ~ x)+
  labs(
    title = "Attempting to predict number of laid off using date of layoff",
    caption = "We find that there isn't any linear trend in our data, \nso we do not meet
  )
```



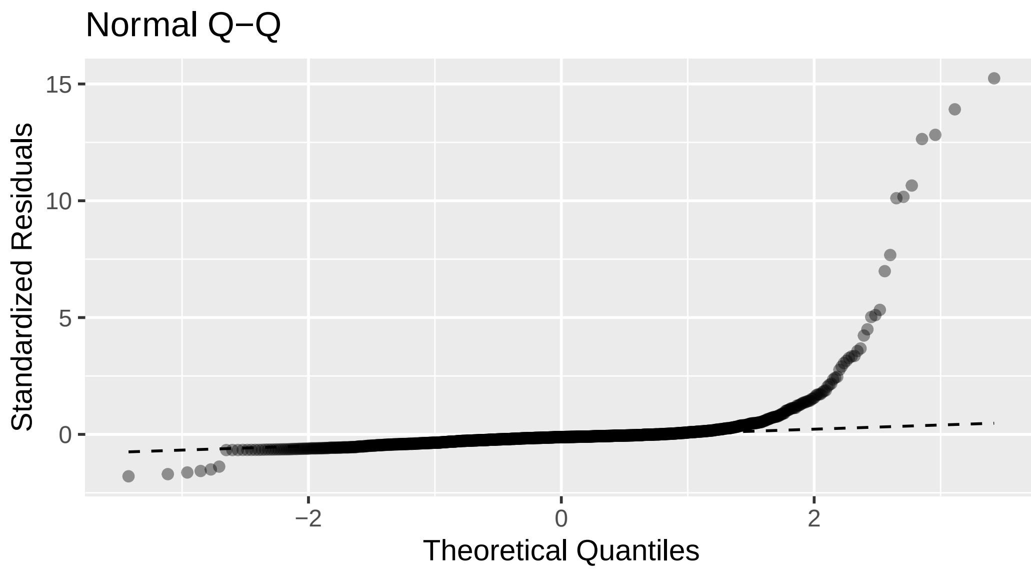

```r
model5 <- lm(total_laid_off ~ industry, data = layoffs_staging)
get_regression_table(model5)
```

```
# A tibble: 30 x 7
   term                    estimate std_error statistic p_value lower_ci upper_ci
   <chr>                      <dbl>     <dbl>     <dbl>   <dbl>    <dbl>    <dbl>
 1 intercept                 165.        380.     0.434   0.664    -581.     911.
 2 industry: Construction    186.        444.     0.419   0.676    -685.    1057.
 3 industry: Consumer        345.        389.     0.888   0.374    -417.    1108.
 4 industry: Crypto            7.94      392.     0.02    0.984    -762.     778.
 5 industry: Data            -61.0       396.    -0.154   0.878    -837.     715.
 6 industry: Education        33.8       392.     0.086   0.931    -734.     802.
 7 industry: Energy          -31.6       491.    -0.064   0.949    -995.     932.
 8 industry: Fin-Tech       -120.        659.    -0.183   0.855   -1413.    1172.
 9 industry: Finance         -22.3       384.    -0.058   0.954    -776.     731.
10 industry: Fitness         258.        412.     0.627   0.531    -550.    1067.
# i 20 more rows
```

```
glance(model5)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic   p.value    df  logLik    AIC    BIC
      <dbl>         <dbl> <dbl>     <dbl>     <dbl> <dbl>   <dbl>  <dbl>  <dbl>
1    0.0409        0.0234  761.      2.33 0.0000811    29 -13007. 26076. 26243.
# i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```
ggplot(model5) +
  stat_normal_qq(alpha = 0.4) +
  labs(caption = "In the right tail, we see that there are severe deviations at the end of
```



In the right tail, we see that there are severe deviations at the end of the tails and that this deviation starts around 1.2 on the x−axis. This means that our residuals don't follow a normal distribution and do not fulfill our fourth condition.

```
model6 <- lm(total_laid_off ~ location, data = layoffs_staging)
get_regression_table(model6)
```

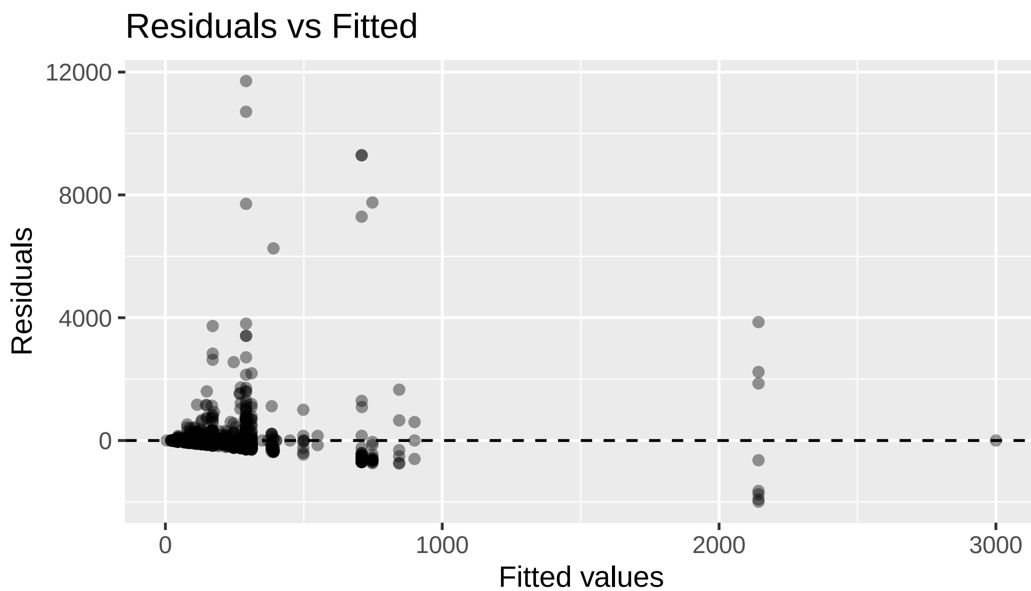
```
# A tibble: 146 x 7
   term                 estimate std_error statistic p_value lower_ci upper_ci
   <chr>                   <dbl>     <dbl>     <dbl>   <dbl>    <dbl>    <dbl>
 1 intercept                500       776.     0.644   0.519  -1022.     2022.
 2 location: Albany        -129      1097.    -0.118   0.906  -2282.     2024.
 3 location: Amsterdam     1642.      823.     2.00    0.046     28.1    3257.
 4 location: Ann Arbor     -380       950.    -0.4     0.689  -2244.     1484.
 5 location: Atlanta       -329.      810.    -0.406   0.685  -1919.     1260.
 6 location: Auckland      -455      1097.    -0.415   0.678  -2608.     1698.
 7 location: Austin        -110.      793.    -0.138   0.89   -1664.     1445.
 8 location: Baltimore     -467       950.    -0.491   0.623  -2331.     1397.
 9 location: Bangkok       -445      1097.    -0.406   0.685  -2598.     1708.
10 location: Barcelona     -250      1097.    -0.228   0.82   -2403.     1903.
# i 136 more rows
```

```
glance(model6)
```

```
# A tibble: 1 x 12
```

```
  r.squared adj.r.squared sigma statistic p.value    df  logLik    AIC    BIC
      <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>   <dbl>  <dbl>  <dbl>
1    0.0752       -0.0160  776.     0.825   0.932   145 -12978. 26249. 27041.
# i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```
ggplot(model6) +
  stat_fitted_resid(alpha = 0.4) +
  labs(caption = "We can clearly see that the residuals do not have constant variability a
```



We can clearly see that the residuals do not have constant variability along the x axis, indicating that we do not have our second condition for the model fulfilled.

```
model7 <- lm(total_laid_off ~ country, data = layoffs_staging)
get_regression_table(model7)
```

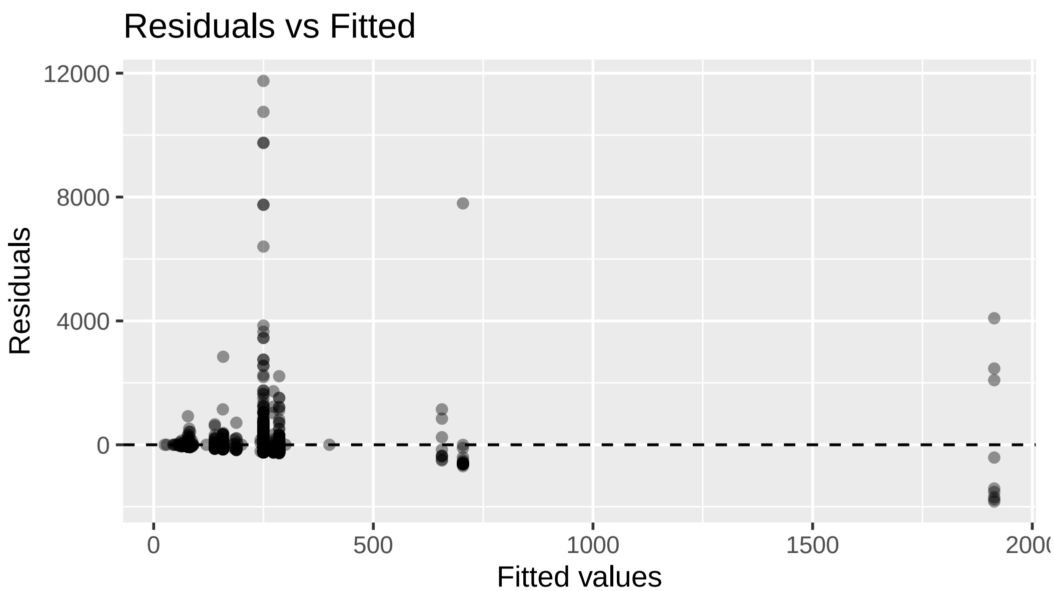
```
# A tibble: 44 x 7
   term               estimate std_error statistic p_value lower_ci upper_ci
   <chr>                 <dbl>     <dbl>     <dbl>   <dbl>    <dbl>    <dbl>
 1 intercept              64.6      342.     0.189   0.85     -606.     736.
 2 country: Australia     18.4      371.     0.05    0.96     -710.     747.
 3 country: Austria      125.       559.     0.224   0.822    -970.    1221.
 4 country: Brazil        92.8      355.     0.262   0.794    -603.     789.
 5 country: Bulgaria      55.4      838.     0.066   0.947   -1588.    1699.
 6 country: Canada        13.4      352.     0.038   0.97     -678.     705.
```

```
 7 country: Chile        -34.6       838.    -0.041   0.967   -1678.    1609.
 8 country: China        592.        427.     1.39    0.166    -245.    1428.
 9 country: Colombia       0.4       640.     0.001   1        -1255.    1256.
10 country: Denmark       -4.6       513.    -0.009   0.993   -1011.    1002.
# i 34 more rows
```

```r
glance(model7)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic p.value    df  logLik    AIC    BIC
      <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>   <dbl>  <dbl>  <dbl>
1    0.0389        0.0126  765.      1.48  0.0240    43 -13009. 26107. 26350.
# i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

```r
ggplot(model7) +
  stat_fitted_resid(alpha = 0.4) +
  labs(caption = "We can see that the residuals do not have constant variability,\n meanin
```



We can see that the residuals do not have constant variability,
meaning that the second condition isn't fulfilled.

```r
tibble(layoffs_staging)
```

```
# A tibble: 1,995 x 9
   company location industry total_laid_off percentage_laid_off date       stage
   <chr>   <chr>    <chr>             <int>               <dbl> <date>     <chr>
 1 Atlass~ Sydney   Other               500                0.05 2023-03-06 Post~
 2 Sirius~ New Yor~ Media               475                0.08 2023-03-06 Post~
 3 Alerzo  Ibadan   Retail              400               NA    2023-03-06 Seri~
 4 UpGrad  Mumbai   Educati~            120               NA    2023-03-06 <NA>
 5 Loft    Sao Pau~ Real Es~            340                0.15 2023-03-03 <NA>
 6 Embark~ SF Bay ~ Transpo~            230                0.7  2023-03-03 Post~
 7 Lendi   Sydney   Real Es~            100               NA    2023-03-03 <NA>
 8 UserTe~ SF Bay ~ Marketi~             63               NA    2023-03-03 Acqu~
 9 Airbnb  SF Bay ~ Travel               30               NA    2023-03-03 Post~
10 Zscaler SF Bay ~ Security            177                0.03 2023-03-02 Post~
# i 1,985 more rows
# i 2 more variables: country <chr>, funds_raised_millions <int>
```

I will also provide an interpretation for some of the coefficients in the multiple linear regression models. I tried both equal slopes and varying slopes with a few factors, but given that our initial assumptions for linear regression weren't met, this is more meant as an exercise in interpretation. I do not expect (as I find in my models) that MLR is effective in predicting the number of those laid off.

In the following model, which is an equal slopes model, the

- intercept means that for a layoff for a company that is not in a stage or a country, the number of those laid off is, on average, approximately 212.
- β value for `stage: Post-IPO` means that being a company with a business stage of Post-IPO leads to an average 447 increase in the size of the layoff, keeping the country constant.
- β value for `country: China` means that being a company located in China leads to an average 338 increase in the size of the layoff, keeping business stage constant.

An equal slopes model assumes that the relationship between the total number of those laid off and the business stage does not depend on country, and the total number of those laid off and the country does not depend on the business stage.

```r
model2 <- lm(total_laid_off ~ stage + country, data = layoffs_staging)
get_regression_table(model2)
```

```
# A tibble: 52 x 7
   term                  estimate std_error statistic p_value lower_ci upper_ci
   <chr>                    <dbl>     <dbl>     <dbl>   <dbl>    <dbl>    <dbl>
```

```
 1 intercept                 212.        467.       0.453   0.65      -705.   1128.
 2 stage: Post-IPO           448.         88.5      5.06    0          274.    621.
 3 stage: Private Equity     106.        175.       0.604   0.546     -238.    449.
 4 stage: Seed              -128.        163.      -0.789   0.43      -447.    191.
 5 stage: Series A          -156.        109.      -1.43    0.154     -369.     58.2
 6 stage: Series B          -116.         94.6     -1.23    0.219     -302.     69.3
 7 stage: Series C           -86.6        94.8     -0.913   0.361     -273.     99.5
 8 stage: Series D           -91.1        97.8     -0.932   0.352     -283.    101.
 9 stage: Series E           -69.5       113.      -0.618   0.537     -290.    151.
10 stage: Series F            -9.46      137.      -0.069   0.945     -278.    259.
# i 42 more rows
```

```r
glance(model2)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic  p.value    df  logLik    AIC    BIC
      <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>   <dbl>  <dbl>  <dbl>
1     0.110        0.0757  792.      3.19 1.39e-12    51 -10997. 22101. 22377.
# i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

This, on the other hand, is a varying slopes model. This assumes that the relationship between the total number of those laid off and business stage differs based on the country; similarly, the relationship between the total number of those laid off and country differs based on the business stage.

The β value for `stage: Post-IPO, country: Australia` means that a business in the Post-IPO stage is associated with a 441 decrease in the size of layoff for a business in Australia compared to a business in the Post-IPO stage that is not in Australia.

```r
model3 <- lm(total_laid_off ~ stage * country, data = layoffs_staging)
get_regression_table(model3)
```

```
# A tibble: 570 x 7
   term                  estimate std_error statistic p_value lower_ci upper_ci
   <chr>                    <dbl>     <dbl>     <dbl>   <dbl>    <dbl>    <dbl>
 1 intercept              129.        824.      0.157   0.876   -1488.    1746.
 2 stage: Post-IPO        502.        106.      4.72    0         293.     711.
 3 stage: Private Equity    8.03      209.      0.038   0.969    -402.     418.
 4 stage: Seed           -145.        230.     -0.632   0.527    -597.     306.
 5 stage: Series A        -39.1      1006.     -0.039   0.969   -2012.    1934.
 6 stage: Series B       -107.        118.     -0.903   0.367    -339.     125.
```

```
 7 stage: Series C          -91.6       117.    -0.785   0.433    -321.     137.
 8 stage: Series D          -76.1       122.    -0.626   0.532    -315.     162.
 9 stage: Series E          -31.9       136.    -0.234   0.815    -299.     235.
10 stage: Series F          -14.4       165.    -0.087   0.93     -337.     308.
# i 560 more rows
```

```r
glance(model3)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic p.value    df  logLik    AIC    BIC
      <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>   <dbl>  <dbl>  <dbl>
1     0.130        0.0220  815.      1.20  0.0567   150 -10982. 22269. 23062.
# i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```