TBD

Zaid and Zoe

```r
library(tidyverse)
library(tidymodels)
library(stringr)
library(stringi)
library(leaps)
library(MASS)
library(glmnet)
library(caret)
library(Matrix)
pass <- read.csv("data/pass.csv")
```

```r
uniqchars <- function(x) unique(strsplit(x, "")[[1]])

pass_more <- pass |>
  mutate(true_val = ifelse(time_unit == "years", 31536000*value,
                           ifelse(time_unit == "months", 2592000*value,
                          ifelse(time_unit == "days", 86400*value,
                          ifelse(time_unit == "hours", 3600*value,
                          ifelse(time_unit =="minutes", 60*value,value))))),
         true_val = jitter(true_val),
         pass_length = nchar(password),
         font_size = NULL,
         value = NULL,
         time_unit = NULL,
         rank_alt = NULL,
         num_digits = str_count(password, "[0-9]"),
         num_letters = str_count(password, "[a-z]"),
         num_vowels = str_count(password, "[a,e,i,o,u]"),
         num_unique = sapply(strsplit(password, ""),
                             function(x) length(unique(x)))) |>
  filter(!is.na(rank)) |>
  filter(strength < 11)

nrow(pass_more)
```

```
[1] 485
```

```
#this is just to organize categories for EDA

pass_ordered <- pass_more

pass_ordered$category <- with(pass_ordered,
                              reorder(category, strength, median, na.rm=T))
```

Introduction

Research Question and Motivation

In our increasingly technology-oriented world, data security is a pressing and essential topic. As cybercriminals' hacking tools have improved, data leaks at major companies such as Yahoo, Facebook, LinkedIn, Mariott International, Adobe, Bank of America, British Airways, and CVS have compromised billions of users' personal information. In 2022, IBM found that the average data breach in the U.S. cost companies an average of $9.44 million in lost business, crisis management efforts, and ransom payments. Data breaches can also allow hackers to access users' personal information such as names, addresses, credit card details, and Social Security numbers, which can be used for financial fraud or identity theft. One critical aspect of data security is password strength, which can reduce the risk of cybercriminals guessing users' passwords and acesssing personal information. Given our interset in datasecurity and the topcality of password strength as a key facet of this subject area, we wanted to explore password data for our project.

Our research question is: What characteristics yield strong passwords? We measure password strength in two ways: "strength" (which is calculated by an algorithm based on the password's length and complexity and is comparative to the generally bad passwords in the dataset) and the time the password takes to crack by online guessing. We decide which passwords are the strongest overall by looking at the characteristics that appear to impact both measures of password strength.

External research: https://www.keepersecurity.com/blog/2022/09/14/why-is-password-security-important/ https://www.bleepingcomputer.com/news/security/the-benefits-of-making-password-strength-more-transparent/ https://www.ibm.com/downloads/cas/3R8N1DZJ

Data source: https://github.com/rfordatascience/tidytuesday/tree/master/data/2020/2020-01-14 https://docs.google.com/spreadsheets/d/1cz7TDhm0ebVpySqbTvrHrD3WpxeyE4hLZtifWSnoNTQ/edit

Data Description

Variable Name	Type	Description
rank	numeric	Popularity in their database of released passwords
password	character	Actual text of password
category	categorical	Classification of type of password
true_val	double	Time to crack by online guessing standardized to seconds
offline_crack_sec	double	Time to crack offline in seconds
strength	ordinal	Quality of password where 10 is highest, 1 is lowest
pass_length	numeric	Length of the password
num_digits	numeric	Number of digits in the password
num_letters	numeric	Number of letters in the password
num_vowels	numeric	Number of vowels in the password
num_unique	numeric	Number of unique characters (letters or numbers in the password)

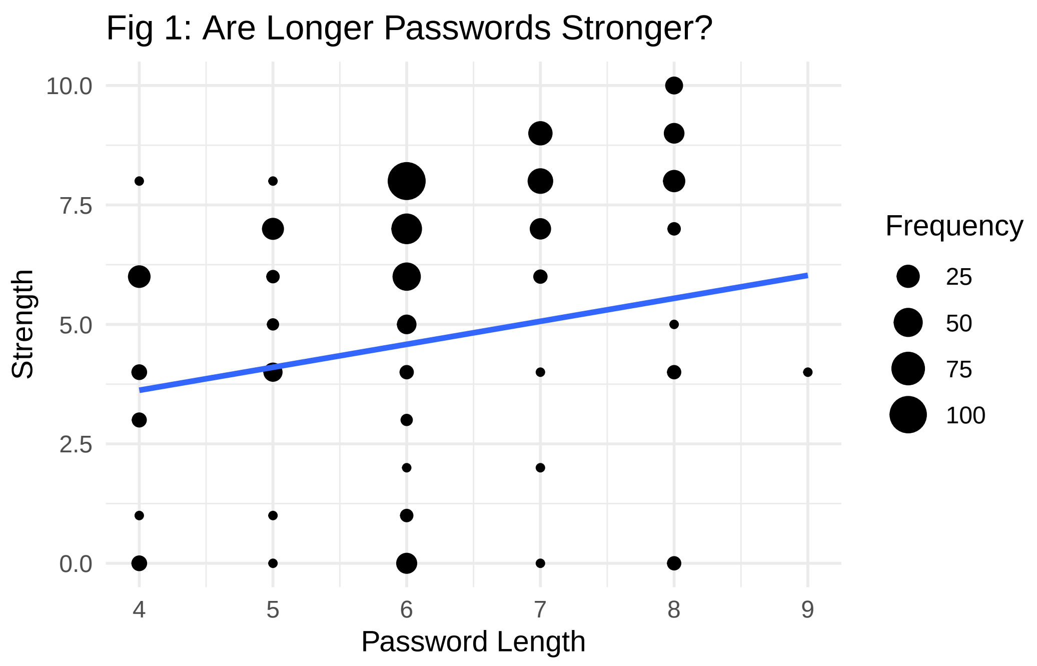
Our data come from Tidy Tuesday, originally sourced from Information is Beautiful, a design company that distills data into visualizations and infographics. Information is Beautiful acquired its data on passwords by deep-mining 20 separate data breaches in 2017, including breaches of Facebook, Sony, and Yahoo. The data only includes the 500 most popular passwords, which also tended to be low-strength. Therefore, the `strength` variable indicates password strength in relation to these generally weak passwords.

In the cleaning process, we removed the last seven observations, as all their values were "NA." We also removed several variables that appeared in the original dataset. First, we removed the variables that had a strength recorded over ten as those may have been miscalculations or strengths that were not standardized to values 1 through 10. Second, we removed the `rank_alt` variable because we wanted to focus on the passwords' first ranks of popularity, as opposed to their secondary ranks, because their first ranks were a clearer indicator of how common they were. Third, also removed the font_size variable, as the font sizes were chosen arbitrarily to display passwords in a graphic on the Knowledge is Beautiful webiste. Fourth and finally, we combined the `value` and `time_unit` variables into one time standardized to seconds called `true_val`. Previously, `value` referred to the time to crack by online guessing, and time unit was the time unit to match with that value (seconds, minutes, hours, days, months, or years). We added noise to our new `true_val` variable because the time to crack by online guessing only included discrete values (2.17 years, 0.00321 days, etc.). From there, we were left with 485 observations.

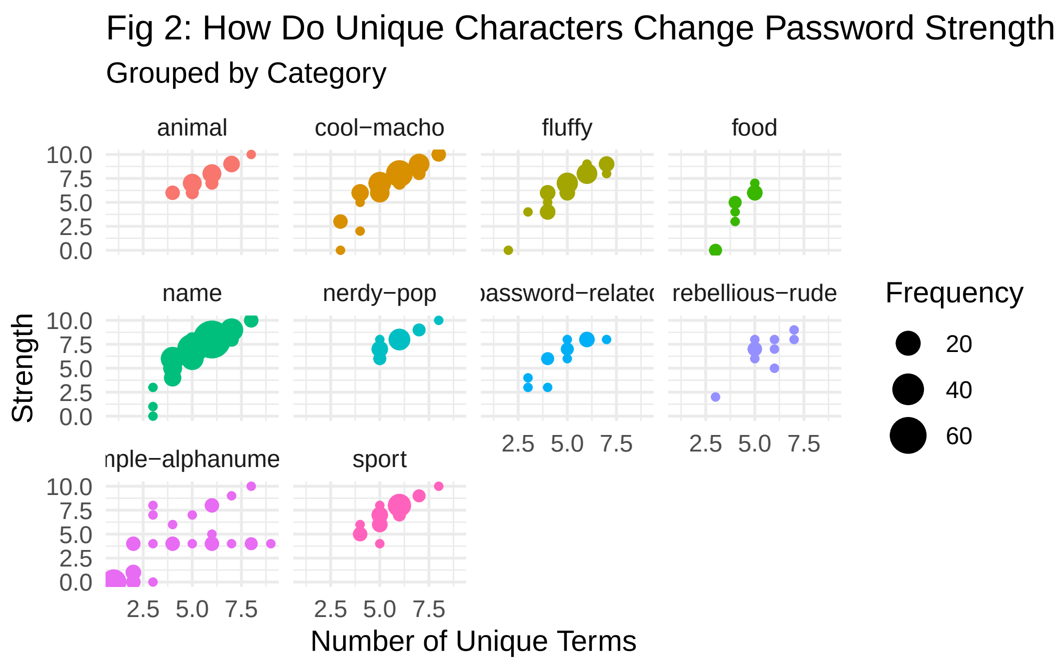
Finally, we added five new variables: pass_length, num_digits, num_letters, num_vowels, and num_unique. We added these variables because we believe that the length of the password, as well as its composition, could impact its strength.

Exploratory Data Analysis

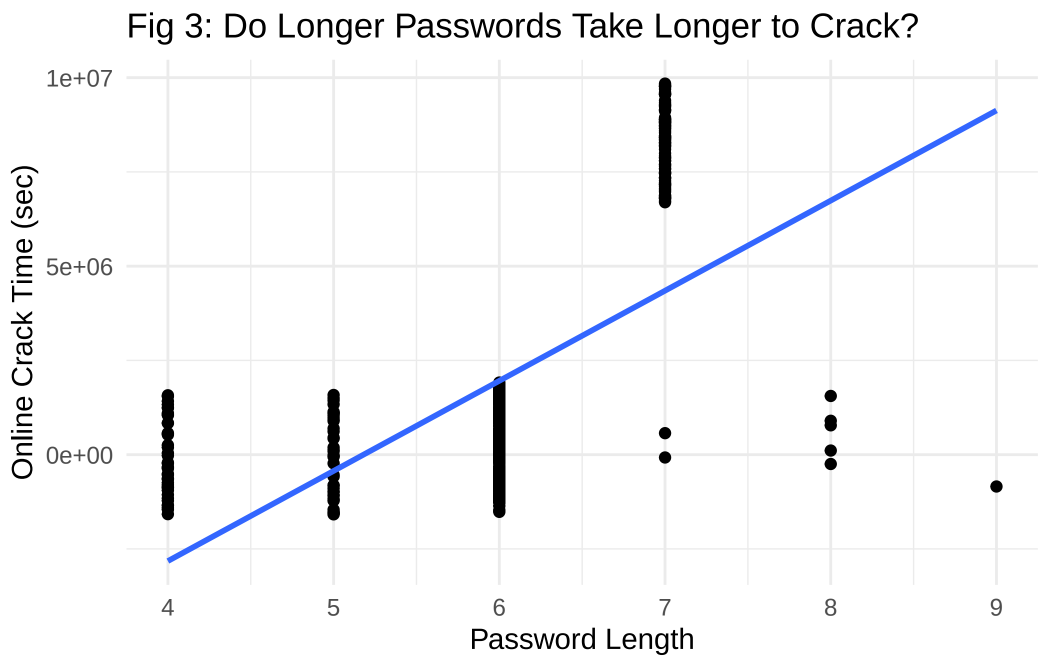
```
pass_more %>%
  group_by(strength, pass_length) %>%
  summarize(freq = n()) |>
  ggplot(aes(x = pass_length, y = strength)) +
  geom_point(aes(size = freq)) +
  geom_smooth(method = "lm", se = F) +
  theme_minimal() +
  labs(x = "Password Length", y = "Strength",
       title = "Fig 1: Are Longer Passwords Stronger?", size = "Frequency")
```



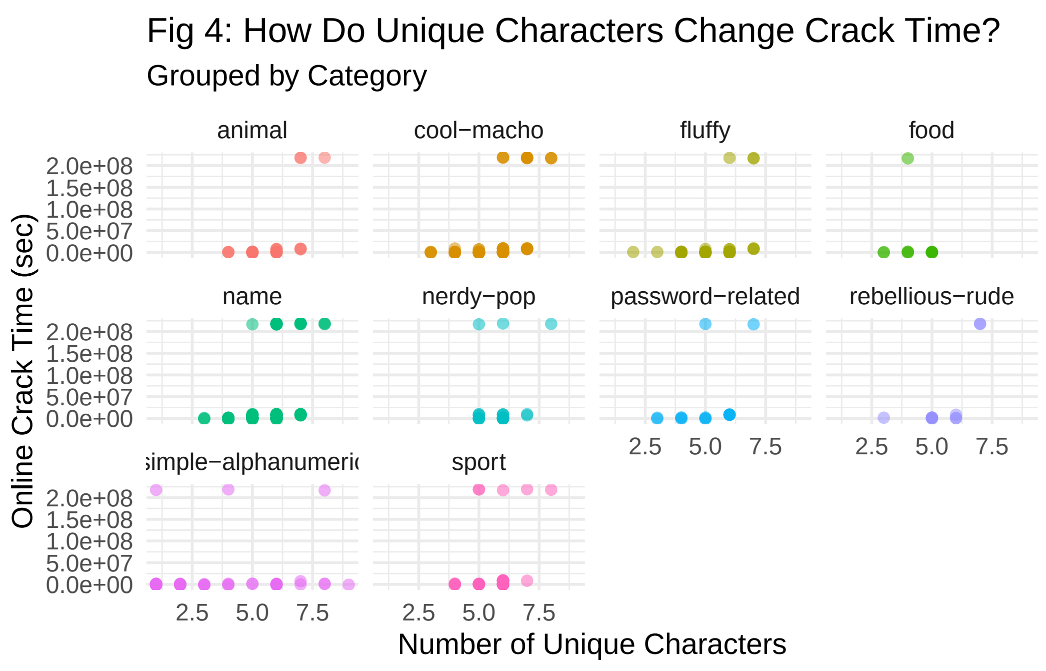
```
pass_more %>%
  group_by(num_unique, strength, category) %>%
  summarize(freq=n()) |>
  ggplot(aes(x = num_unique, y = strength, color = category, size = freq)) +
  facet_wrap(~category) +
  geom_point() +
  theme_minimal() +
  scale_fill_viridis_d() +
  guides(size=guide_legend("Frequency"), color = "none") +
  labs(x = "Number of Unique Terms", y = "Strength",
       title = "Fig 2: How Do Unique Characters Change Password Strength?",
       subtitle = "Grouped by Category")
```



```
pass_more |>
  filter(true_val < 100000000) %>%
  ggplot(aes(x = pass_length, y = true_val)) +
  geom_point() +
  geom_smooth(method = "lm", se = F) +
  theme_minimal() +
  labs(x = "Password Length", y = "Online Crack Time (sec)",
       title = "Fig 3: Do Longer Passwords Take Longer to Crack?")
```



```
pass_more %>%
  ggplot(aes(x = num_unique, y = true_val, color = category, alpha = 0.05)) +
  facet_wrap(~category) +
  geom_point() +
  theme_minimal() +
  scale_fill_viridis_d() +
  guides(color = "none", alpha = "none") +
  labs(x = "Number of Unique Characters", y = "Online Crack Time (sec)",
       title = "Fig 4: How Do Unique Characters Change Crack Time?",
       subtitle = "Grouped by Category")
```



```
# pass_more %>%
#  filter(true_val < 100000000) %>%
#  ggplot(aes(x = num_unique, y = true_val, color = category, alpha = 0.05)) +
#  facet_wrap(~category) +
#  geom_point() +
#  theme_minimal() +
#  scale_fill_viridis_d() +
#  guides(color = "none", alpha = "none") +
#  labs(x = "Number of Unique Terms", y = "Online Crack Time (sec)",
#       title = "Fig 4: How Do Unique Characters Change Crack Time?",
#       subtitle = "Grouped by Category")

pass_more %>%
  summarize(mean(strength))
```

```
  mean(strength)
1       6.606186
```

```
pass_more %>%
  summarize(mean(true_val))
  mean(true_val)
1       26769084
```

```
pass_more %>%
  summarize(mean(num_digits))
```

```
  mean(num_digits)
1        0.4639175
```

```
pass_more %>%
  summarize(mean(num_letters))
```

```
  mean(num_letters)
1          5.717526
```

```
pass_more %>%
  summarize(mean(num_unique))
```

```
  mean(num_unique)
1         5.191753
```

```
pass_more %>%
  summarize(mean(num_vowels))
```

```
  mean(num_vowels)
1         2.078351
```

```
pass_more %>%
  summarize(mean(pass_length))
```

```
  mean(pass_length)
1          6.181443
```

Given our prior knowledge of what makes passwords stronger, we chose to focus our exploratory data analysis on the predictors password length and number of unique characters, along with their relationships with other variables in the dataset.

Figure 1 demonstrates that there appears to be a positive relationship between password length and the strength variable. The line of best fit shows that as password length increases, the strength of the password also tends to increase. We can also see this from the data themselves based on how the size of the points change as strength increases. For passwords of length 6, 7, and 8, most passwords have strengths of above 5. For passwords of length 4 and 5, there are some passwords with strengths above 5, but there appear to be a similar number of passwords with these lengths with strengths below 5, as well. This graph also helps visualize the composition of the data itself. The large size of several points associated with password length 6 indicates that, by far, most passwords in this dataset have length 6. Passwords of length 7 appear to be the next-most frequent, passwords of length 4, 5, and 8 appear to be about equally frequent, and passwords of length 9 are very infrequent. It is helpful to understand for the following data analysis that our sample size of long passwords is very small, which may limit the conclusions we can draw for that group.

Figure 2 demonstrates that the number of unique characters appears to have a positive relationship with password strength. Within each password category, as the number of unique characters increases, the strength of the password also tends to increase. This relationship holds for all categories, except simple-alphanumeric. Although there appears to be a positive relationship between number of unique terms and password strengths for some passwords in this category, the horizontal line in this plot also shows that some passwords with varying numbers of unique terms have the same password strength. Additionally, the size of the points in the plot demonstrates that some categories of passwords were more popular in our data, especially name, cool-macho, fluffy, and sport.

Figure 3 shows a positive relationship between password length and online crack time, as demonstrated by the positive slope of the line of best fit. To create this visualization, we excluded passwords with online crack times of above 10^{8} seconds, since this expanded the visualization and made individual points more difficult to visualize. Interestingly, only passwords of length 8 had online crack times above 10^{8} seconds, which seems to confirm the trend that longer passwords have higher strength scores.

Figure 4 demonstrates that there appears to be a positive relationship between number of unique terms and online crack time. Although the data is squished at the top and bottom of the graph, making individual points difficult to visualize, the transparency of the points allows us to visualize their density. As the number of unique terms increases, the online crack time remains similar for many passwords. However, those passwords with the greatest online crack seconds (> 10^{8}) are typically those with at least 5 characters. There are, however, some exceptions with some passwords with few unique characters in the simple-alphanumeric and food categories still acheiving high online crack times.

To better understand our data, we also calculated summary statistics to determine which kinds of passwords were the most common. For our data, the average number of digits in a password are 0.464, the average nubmer of letters is 5.718, the average number of unique characters is 5.192, the average number of vowels is 2.08, and the average password length is 6.181. In general, the most popular passwords in the data leaks (the source of our data) used all unique letters and rarely used numbers.

Methodology

Based on our research and prior knowledge, the variables we believe will be important to include are `pass_length`, `num_digits`, `num_unique`, and XX…

- *do we think category is important?* - since online cracking is partially done by guessing common passwords…

```r
y1 <- pass_more$strength
x1 <- model.matrix(strength ~ . - password - true_val - offline_crack_sec - rank,
            data = pass_more)

y2 <- pass_more$true_val
x2 <- model.matrix(true_val ~ . - password - offline_crack_sec - strength - rank,
            data = pass_more)

m_lasso_strength <- cv.glmnet(x1, y1, alpha = 1)
best_lambda <- m_lasso_strength$lambda.min
best_lambda
```

```
[1] 0.02345198
```

```r
m_best <- glmnet(x1, y1, alpha = 1, lambda = best_lambda)
m_best$beta
```

```
15 x 1 sparse Matrix of class "dgCMatrix"
                                  s0
(Intercept)                .
categorycool-macho         .
categoryfluffy             .
categoryfood             -1.17744127
categoryname              0.04378974
categorynerdy-pop         0.15385968
```

```
categorypassword-related     .
categoryrebellious-rude     -0.21682159
categorysimple-alphanumeric -0.38005274
categorysport                .
pass_length                 -0.09581339
num_digits                  -0.30570217
num_letters                  .
num_vowels                   .
num_unique                   1.20915166
```

```
m_lasso_strength <- cv.glmnet(x2, y2, alpha = 1)
best_lambda <- m_lasso_strength$lambda.min
best_lambda
```

```
[1] 1568801
```

```
m_best <- glmnet(x2, y2, alpha = 1, lambda = best_lambda)
m_best$beta
```

```
15 x 1 sparse Matrix of class "dgCMatrix"
                                    s0
(Intercept)                        .
categorycool-macho            410861.9
categoryfluffy                     .
categoryfood                       .
categoryname                       .
categorynerdy-pop           -7488028.5
categorypassword-related           .
categoryrebellious-rude            .
categorysimple-alphanumeric 24116065.4
categorysport               -1986082.2
pass_length                 34897386.7
num_digits                         .
num_letters                  8170847.2
num_vowels                         .
num_unique                  -1729138.1
```

to-dos - LASSO for online crack sec (and offline if we decide to do that) - hypothesis test for idk yet - should we plug in our LASSO variables into a regression (e.g. ordinal for password strength?, OLS for crack time?) or should we just use the LASSO coefficients themselves - would it be too much to see whether these strong passwords are the most common?

Results

Discussion