TBD

Zaid and Zoe

Introduction

Research Question and Motivation

In our increasingly technology-oriented world, data security is a pressing and essential topic. As cyber-criminals' 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#gid=21

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	numeric	Quality of password where 10 is highest, 1 is lowest
pass_length	numeric	Length of the password

Variable Name	Type	Description
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

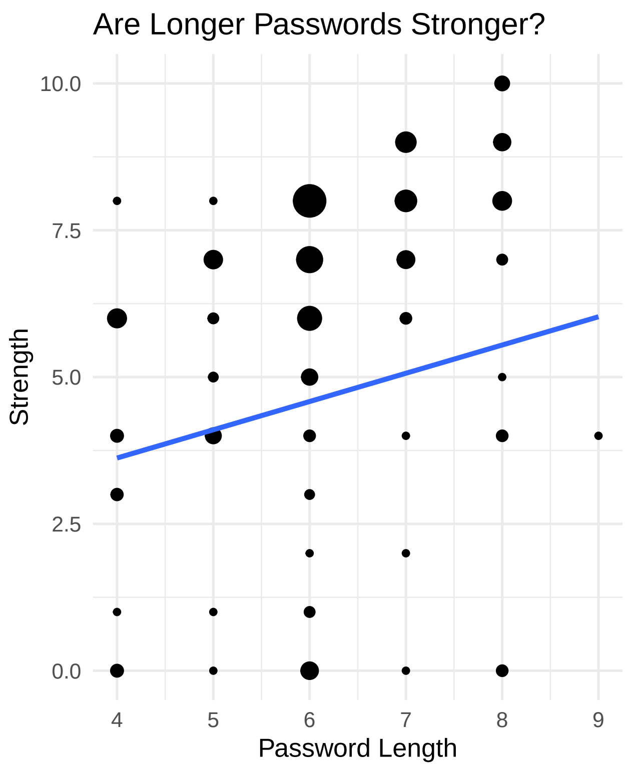

Figure 1

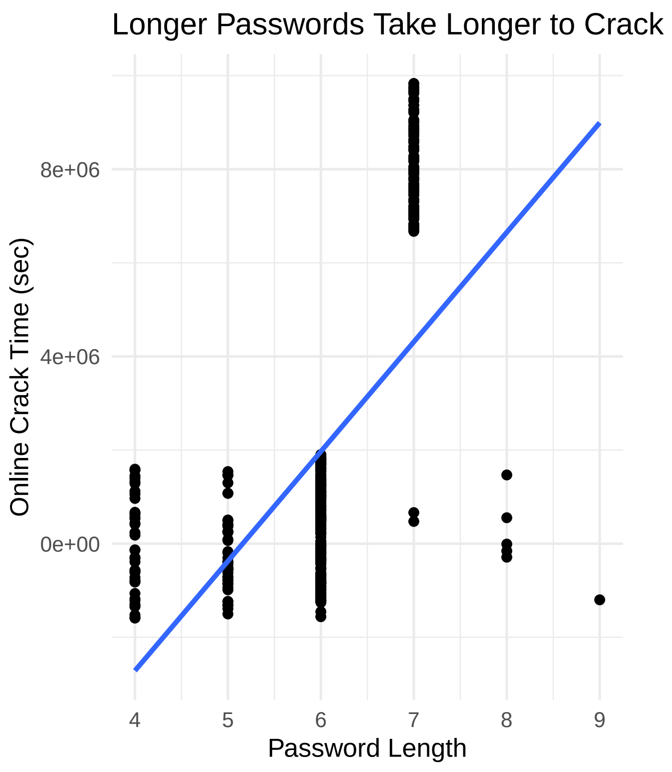

Figure 3

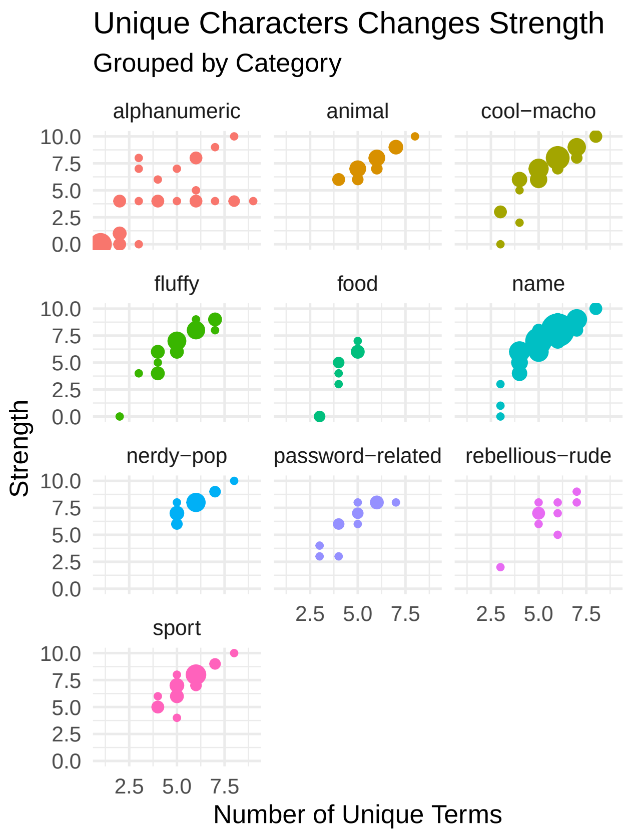

Figure 2

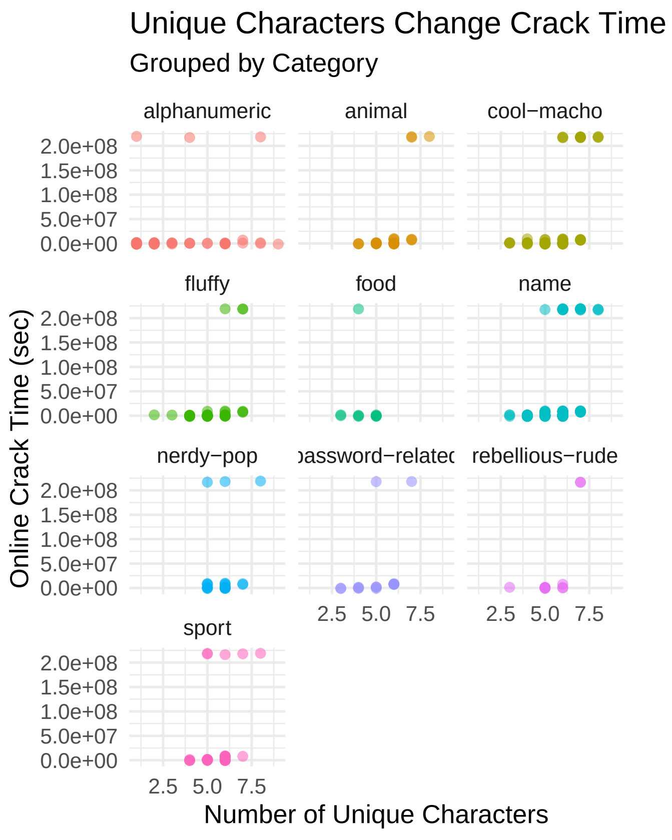

Figure 4

```
           Variable     Mean  Median       Sd        Min       Max
1 offline_crack_sec     0.27  0.0032     0.69 0.00000011       2.2
2          strength      6.6       7      2.3          0        10
3          true_val 26795164 1007127 69218453   -1589868 219440640
4 true_val_strength      8.6       9      2.1          1        11
5       pass_length      6.2       6      1.1          4         9
6        num_digits     0.46       0      1.6          0         9
7       num_letters      5.7       6      1.9          0         8
8        num_vowels      2.1       2     0.96          0         6
9        num_unique      5.2       5      1.5          1         9
```

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

In our analysis, we treat our two outcome variables, strength and online crack time, as ordinal outcomes. *do we tho*

The strength variable is a number 1-10, in order of increasing password strength, making ordinal a good fit. The strength variable meets the ordinal assumption of proportional odds, since it is reasonable to assume that one-unit changes in each predictor have the same conditional relationship with being in each strength category. For example, the strength variable is calculated in part based on password length, and each one character increase in password length has the same conditional relationship with being in each strength category.

Online crack time

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…

We treat strength as an ordinal variable…

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

```
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 - true_val_stre
                   data = pass_more)

y3 <- pass_more$true_val_strength
x3 <- model.matrix(true_val_strength ~ . - password - offline_crack_sec - strength - rank - true
                   data = pass_more)

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

```
[1] 0.03100214
```

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

```
16 x 1 sparse Matrix of class "dgCMatrix"
                                 s0
(Intercept)                .
categoryanimal             0.06375634
```

```
categorycool-macho        .
categoryfluffy            .
categoryfood             -1.08770643
categoryname              0.06735206
categorynerdy-pop         0.14709702
categorypassword-related  .
categoryrebellious-rude  -0.13606170
categorysport             .
true_val_strength         .
pass_length              -0.08152953
num_digits               -0.35613067
num_letters               .
num_vowels                .
num_unique                1.21377219
```

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

```
[1] 1081112
```

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

```
15 x 1 sparse Matrix of class "dgCMatrix"
                               s0
(Intercept)                     .
categoryanimal            -2703505
categorycool-macho              .
categoryfluffy                  .
categoryfood              -4712006
categoryname              -3299695
categorynerdy-pop        -12840050
categorypassword-related        .
categoryrebellious-rude         .
categorysport             -7416644
pass_length               39017162
num_digits                      .
num_letters                5760250
num_vowels                      .
num_unique                -3935805
```

```
m_lasso_true_strength <- cv.glmnet(x3, y3, alpha = 1)
best_lambda <- m_lasso_true_strength$lambda.min
best_lambda
```

```
[1] 0.04011265
```

```r
m_best <- glmnet(x3, y3, alpha = 1, lambda = best_lambda)
m_best$beta
```

```
15 x 1 sparse Matrix of class "dgCMatrix"
                                 s0
(Intercept)                .
categoryanimal             .
categorycool-macho         .
categoryfluffy             .
categoryfood               0.01992769
categoryname               .
categorynerdy-pop          .
categorypassword-related .
categoryrebellious-rude  0.05406715
categorysport              .
pass_length                0.91386153
num_digits                 .
num_letters                0.62757414
num_vowels                 0.02838280
num_unique                 .
```

```r
m1 <- polr(factor(strength) ~ . - password - true_val - true_val_strength - offline_crack_sec -
summary(m1)
```

```
Call:
polr(formula = factor(strength) ~ . - password - true_val - true_val_strength -
    offline_crack_sec - rank - num_letters - num_vowels, data = pass_more)

Coefficients:
                           Value Std. Error t value
categoryanimal           -0.5422     0.7617 -0.7119
categorycool-macho       -0.8514     0.7015 -1.2137
categoryfluffy           -0.8548     0.7243 -1.1802
categoryfood             -3.0098     0.9140 -3.2929
categoryname             -0.6723     0.6809 -0.9873
categorynerdy-pop        -0.2455     0.7982 -0.3076
categorypassword-related -0.7074     0.8632 -0.8195
categoryrebellious-rude  -1.5309     0.9026 -1.6960
categorysport            -0.8894     0.7473 -1.1901
pass_length              -0.3116     0.1418 -2.1972
num_digits               -1.1207     0.2267 -4.9428
num_unique                3.6946     0.2155 17.1474

Intercepts:
     Value   Std. Error t value
0|1   4.4379  1.1224     3.9538
1|2   6.3768  1.1000     5.7971
2|3   7.2025  1.0380     6.9389
```

```
3|4   8.5633  1.0033     8.5348
4|5  11.4748  1.0175    11.2772
5|6  12.3062  1.0212    12.0507
6|7  15.0371  1.1057    13.5991
7|8  17.6989  1.1877    14.9012
8|9  22.0490  1.3242    16.6502
9|10 25.7696  1.4916    17.2765

Residual Deviance: 906.9067
AIC: 950.9067
```

```
m2 <- polr(factor(true_val_strength) ~ . - password - num_vowels - num_letters - offline_crack_s
summary(m2)
```

```
Call:
polr(formula = factor(true_val_strength) ~ . - password - num_vowels -
    num_letters - offline_crack_sec - strength - rank - true_val,
    data = pass_more)

Coefficients:
                           Value Std. Error t value
categoryanimal           -2.1008     1.7570 -1.1956
categorycool-macho       -2.2598     1.3952 -1.6197
categoryfluffy           -2.1499     1.4265 -1.5071
categoryfood             -1.3163     3.5464 -0.3712
categoryname             -2.0787     1.2922 -1.6087
categorynerdy-pop        -0.8051     3.3502 -0.2403
categorypassword-related -2.5701     1.9118 -1.3443
categoryrebellious-rude   3.1858     1.8991  1.6776
categorysport            -1.7996     2.0268 -0.8879
pass_length              12.2155     1.1713 10.4294
num_digits               -4.0217     0.4257 -9.4473
num_unique               -0.1844     0.1566 -1.1776

Intercepts:
      Value   Std. Error t value
1|2   30.0988  3.3669     8.9397
2|3   37.1905  4.2517     8.7471
3|4   41.7130  4.4530     9.3673
4|5   47.8291  4.6903    10.1974
5|6   53.0538  5.3539     9.9093
6|7   55.1551  5.6692     9.7289
7|8   62.3844  6.1357    10.1675
8|9   65.2460  6.6060     9.8768
9|10  77.7581  7.8402     9.9178
10|11 89.2016  8.9607     9.9548

Residual Deviance: 146.3203
AIC: 190.3203
```

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