Predicting Tik-Tok User Data Based on Video Data

GGteam: Will Chen, Katelyn Cai, Hannah Choi, Weston Slayton

2001-11-09

Introduction and data

TikTok now has over 1 billion users globally, making it one of the fastest growing social platforms in the world. As it has risen to prominence, so has its ubiquitous algorithm, which is said to generally account for account factors (likes and comments) and video information (captions, sounds, hashtags). Given, that TikTok has been heavily criticized alongside other platforms for declining youth mental health outcomes and rising hate due to the addictive nature of its explore page, we decided to look at TikTok's data and how follower count (a huge driver of engagement) is impacted by other aspects of a user's account, like average number of videos, average number of likes, and average number of comments.

The dataset comes from the 'top_users_vids.csv' file (under folder 'Trending Videos Data Collection') of the Github repository found at: https://github.com/ivantran96/TikTok_famous/tree/main. The data was originally collected as part of the DataResolutions's Data Blog project exploring Tiktok's demographics and trending video analytics.

The original data curators collected the data using David Teather's open-source Unofficial Tiktok API (found at https://github.com/davidteather/TikTok-Api), which uses Python to scrape Tiktok data and fetch the most trending videos, specific user information, and much more. Using the list of top Tiktokers, the curators compiled a list of users with the getSuggestedUsersbyIDCrawler api method, which used the top TikTokers and collected the suggested users. Using the byUsername method, they collected video data of the 25 most recent posts of each user from the top TikTokers and the suggested list. The curators also used the API's bySound method to collect videos using some of the most famous songs on TikTok to get an idea of how the choice of music can impact the potential of a video to become a trending video.

EDA

We begin our EDA process by first examining the dataset. We have the following columns:

```
 [1] "id"            "create_time"   "user_name"     "hashtags"
 [5] "song"          "video_length"  "n_likes"       "n_shares"
 [9] "n_comments"    "n_plays"       "n_followers"   "n_total_likes"
[13] "n_total_vids"
```

Currently, our dataset tiktok has 13 columns and 12,559 observations. The columns cover attributes of a tiktok video such as video length, hashtags used, songs/sounds used, and number of likes, shares, comments, plays, and followers (and their total number of likes and videos). Variables id, create_time, video_length, n_likes, n_shares, n_comments, n_plays, n_followers, n_total_likes, and n_total_vids are numerical while the others are categorical.

However, from just our initial exploration, it's clear that some of the columns won't be useful for our analysis. It is also apparent that we should find a way to address the potential issue user_name might have with the other columns. There's a potential for severe multicollinearity if we choose to just drop user_name, since the number of plays or likes a video would have a strong relationship with the user that post it. Therefore, any analysis without user and its related features would have to consider the user's account as confounding variables. In addition, we'll be forced to drop valuable features directly related to a user such as user followers, user total likes and user total videos (n_followers, n_total_likes, n_total_vids).

We see that the less relevant variables are create time and video ID. In addition, from looking at our data, hash tags and songs might not be useful. Most videos don't include a hashtag and there are too many unique instances of them for it to be valuable in our analysis. We could consider binning hashtag into none and at least 1 hashtag(s), however that wouldn't be useful for our analysis since its rare for tiktok followers to actually look at the hashtags. The same is true for songs; one could consider grouping original songs into one bin and the rest into others. However, from our domain knowledge, its wouldn't be useful to categorize all original songs as similar since most of them could just be user-edited snippets of actual songs.

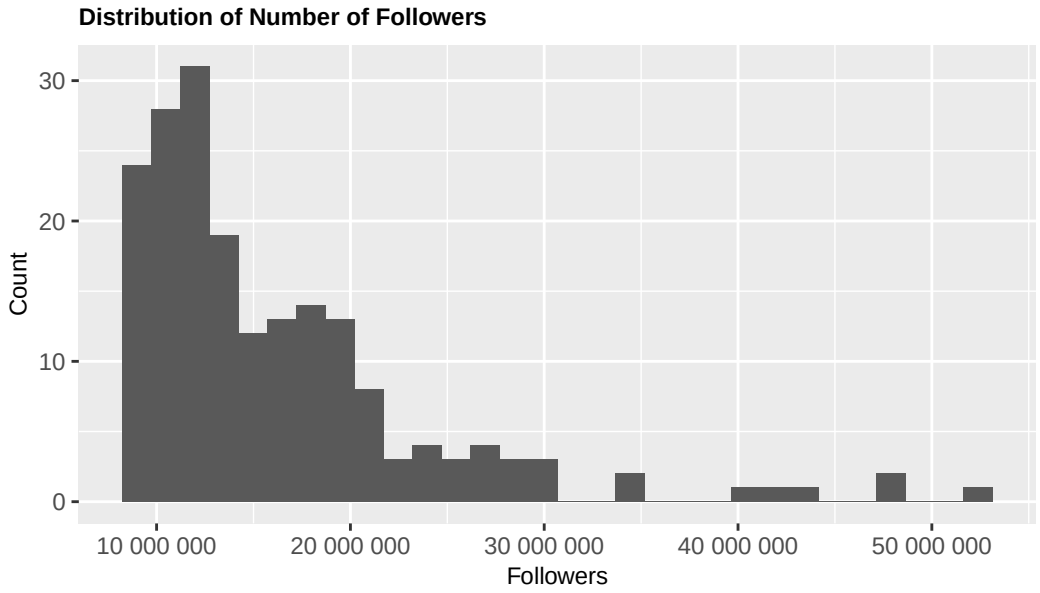
To address the issues mentioned above, we grouped the data by users and summarized all relevant predictor variables by taking their mean. Our modified dataset now has 8 columns and 254 observations.

```
# A tibble: 5 x 8
  user_name     likes shares comments  plays followers video_length total_videos
  <chr>         <dbl>  <dbl>    <dbl>  <dbl>     <dbl>        <dbl>        <dbl>
1 .kunno       335020  1067.    4521. 1.99e6  15300000         19.5         3442
2 _arishfakha~ 425868  5269.    2700. 3.96e6  28600000         14.5         2026
3 _saloniyaapa 166544  6122.     728. 1.85e6  12900000         14.6         2005
4 aashikabhat~ 194280  1335.    1053. 2.17e6  16000000         15.0         2720
5 abbyrartist~ 965586  5292.    7005. 3.97e6  13500000         16.5          811
```

Note that no data leakage is introduced in this process since we are just summarizing by the means of the predictor variable per user. When we split, it'll split based on observations, users. We'll now split our data into testing and training.

```
<Training/Testing/Total>
<190/64/254>
```

Here's a distribution of our response variable, user followers, from our training set.




```
[1] "Mean of followers: 16220526.3157895"
```

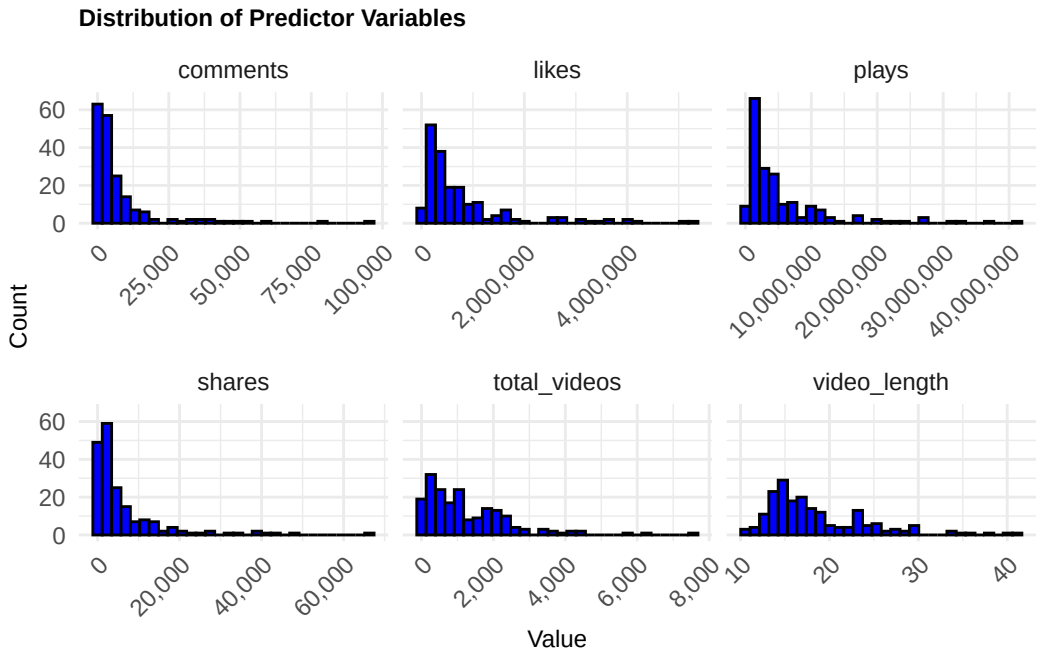
```
[1] "Standard deviation of followers: 7710869.791991"
```

```
[1] "Minimum number of followers: 8900000"
```

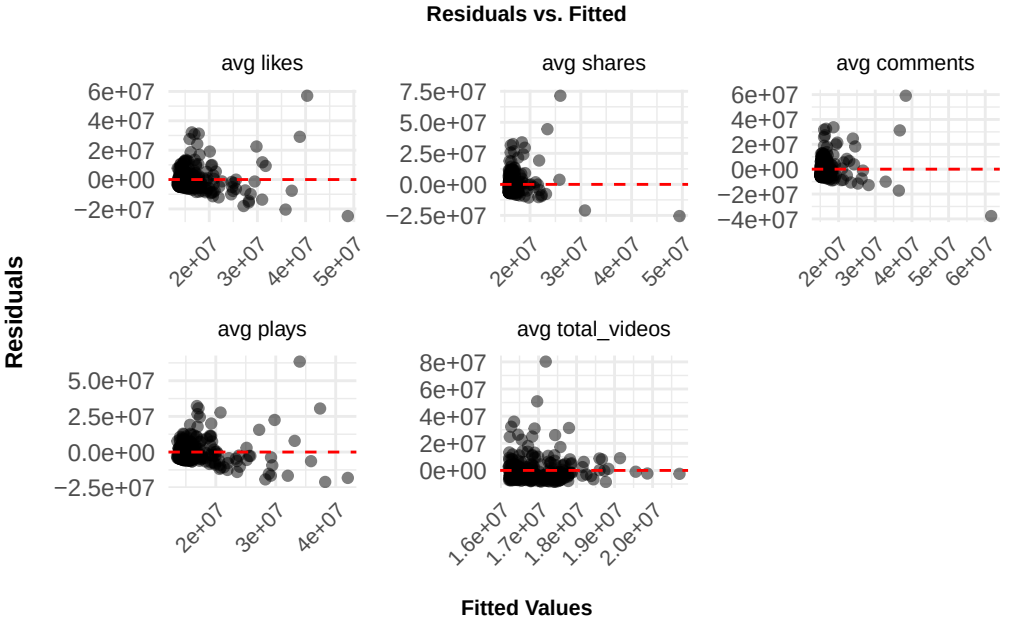
```
[1] "Maximum number of followers: 52300000"
```

The distribution of our response variable follows, is unimodal and heavily right skewed. The mean is 16,220,526.3 and the standard deviation is 7,710,869.8. The minimum is 8,900,000 and the maximum is 52,300,000. Based on our standard deviation, there seems to be a lot of variation in our dataset; and from our plot, there are major outliers.

Here are the distributions for the predictor variables we are interested in:



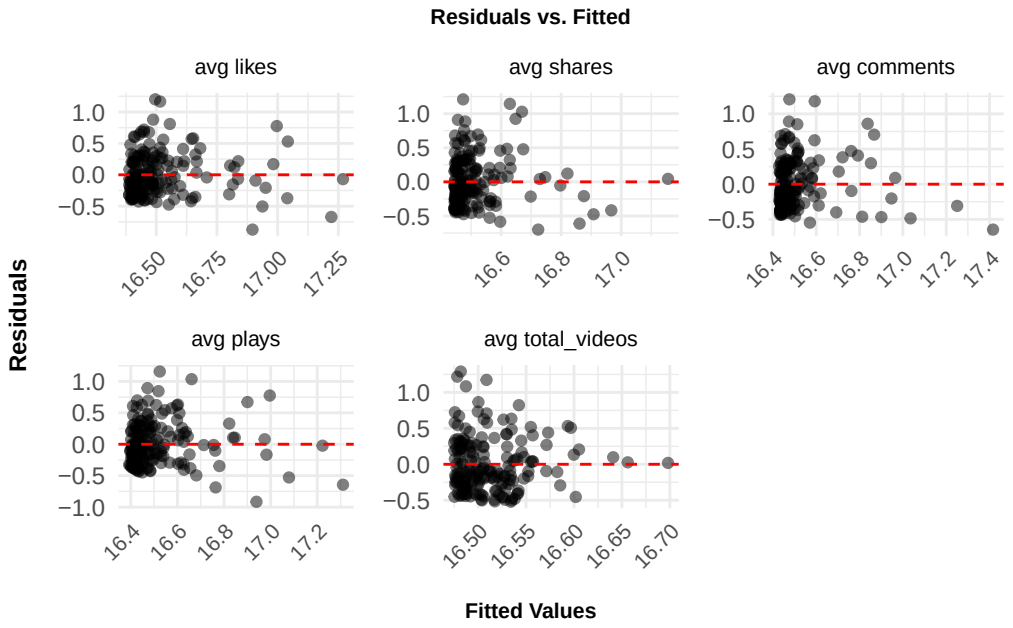
We would like to know if any of our predictor variables violate any conditions. Hence, we have the following residual plots for each of them.



It's clear that all our predictors variables violates constant variance. There's a clear outward spread for likes, shares, comments, plays and video_length, while an inward spread for total_videos.

For interpretability, it makes sense to process average bin video length into levels, corresponding to "short", "medium" and "long." This also allows us to search for interesting interactions effects video_length might have with other predictors such as likes. Therefore, we'll add step_discretize() into the recipe for video_length.

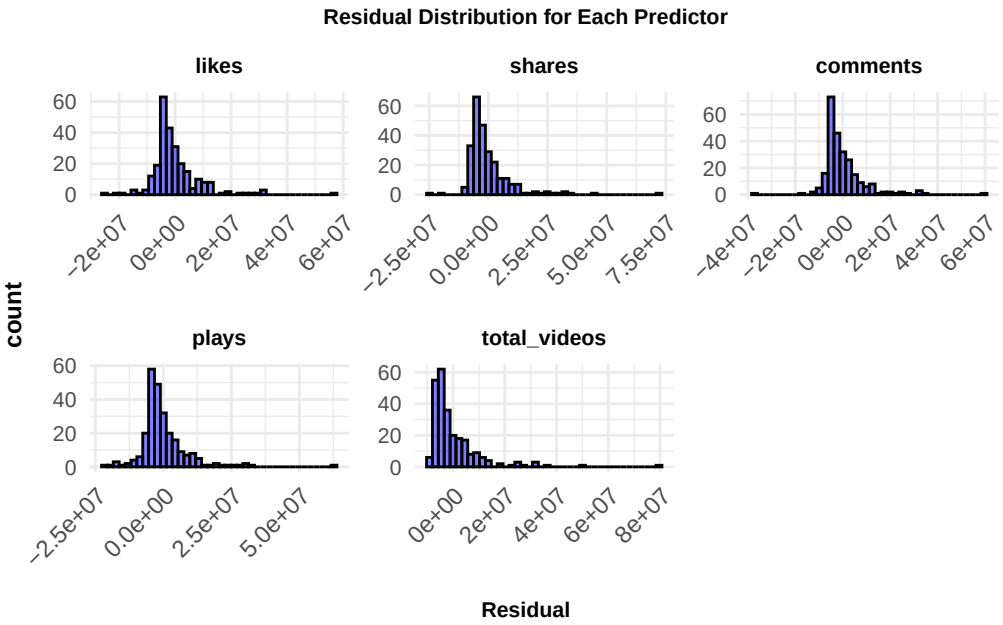
In order to deal with constant variance for the other terms, we log transformed the rest of the predictor variables.



The residual plots don't seem to suggest any underlying patterns about linearity. As such, we conclude the predictors satisfy linearity and constant variance.

We can also assume independence is met. Each of the videos are by individual creators, therefore the videos are independent of each other.

We continue with normality.



Normality seems to be satisfied for each of the predictors except total_videos and possibly shares. However, even though total_videos and shares do not have completely normal distributions, because we have more than 30 observations in the dataset, we can conclude that normality is satisfied regardless of the distribution.

Methodology

Before constructing our model, we chose to log transformed all our variables in the dataset 'transformed_tiktok_users' because, prior to the transformation, the variables were not to scale and returned coefficients of 0.000.

Afterwards, we constructed two linear regression model fitting variable 'followers' with predictor variables 'likes', 'shares', 'comments', 'plays', 'video_length_bin', and 'total_videos.' Our first model m1 had all the previously indicated predictor variables excluding variable 'likes' while our second model m2 had those predictor variables excluding variable 'plays.' We compared these two models because likes and plays had the highest vif values in our VIF test (14.431 and 12.253 respectively), meaning they have the highest likelihood for multicollinearity. We chose the model without plays, m2, because it had a lower AIC and BIC value, indicating that it was a better fit. Therefore, we choose to remove plays from the model and leave likes in the model to deal with the multicollinearity.

In our recipe, we fit the dataset 'tiktok_user,' after which we added steps omitting all NA values from our predictors and log-transforming all our predictor variables and followers. We then conducted a cross-validation test on tiktok_train.

Detecting Multicollinearity & Model Comparison

```
           likes           shares         comments            plays
       11.613679         3.536988         2.681733         9.820479
video_length_bin2 video_length_bin3      total_videos
         1.394816         1.367240         1.181881
```

term	estimate	std.error	statistic	p.value
(Intercept)	16.512	0.024	675.614	0.000
shares	-0.119	0.046	-2.582	0.011
comments	0.105	0.035	2.982	0.003
plays	0.224	0.046	4.885	0.000
total_videos	0.113	0.027	4.243	0.000
video_length_bin2	0.001	0.029	0.027	0.979
video_length_bin3	0.028	0.029	0.973	0.332

term	estimate	std.error	statistic	p.value
(Intercept)	16.512	0.025	673.395	0.000
likes	0.238	0.050	4.753	0.000
shares	-0.098	0.044	-2.247	0.026
comments	0.060	0.039	1.533	0.127
total_videos	0.110	0.027	4.142	0.000
video_length_bin2	-0.001	0.029	-0.043	0.966
video_length_bin3	0.024	0.029	0.831	0.407

Model Comparison with 5-fold CV

Model 1: (without likes):

RMSE:

```
# A tibble: 1 x 1
   mean
  <dbl>
1 0.340
```

Adj. r-sq, AIC, BIC:

```
# A tibble: 1 x 3
  mean_adj_rsq mean_aic mean_bic
         <dbl>    <dbl>    <dbl>
1        0.266     90.7     114.
```

Model 2 (without plays):

RMSE:

```
# A tibble: 1 x 1
   mean
  <dbl>
1 0.333
```

Adj. r-sq, AIC, BIC:

```
# A tibble: 1 x 3
  mean_adj_rsq mean_aic mean_bic
         <dbl>    <dbl>    <dbl>
1        0.259     92.1     116.
```

While model 1 has a higher RMSE, it also has a lower AIC and BIC, and a higher adjusted r-squared. Because of this, we can conclude that model 1 (the model without likes) is a better fit. Therefore, we choose to remove likes from the model and leave plays in the model to deal with the multicollinearity.

Determining whether video_length_bin are necessary

We can see from our previous tidy table that the p-values associated with the video length bins are high, indicating that the variables aren't significant. Because of this, we can do a cross validation test to see how a model without video_length performs

RMSE:

```
# A tibble: 1 x 1
   mean
  <dbl>
1 0.339
```

Adj. r-sq, AIC, BIC:

```
# A tibble: 1 x 3
  mean_adj_rsq mean_aic mean_bic
         <dbl>    <dbl>    <dbl>
1        0.269     88.2     106.
```

We can see that when we remove video_length, AIC and BIC both significantly decrease and adjusted r-squared increases. However, RMSE also increases. This means that a model without video_length may fit our data better, however it also performs worse when it comes to prediction (there is more distance between predicted and observed values). Because of this, we should keep video_length in our model.

Determining whether interaction terms are needed

We also want to determine whether there are any significant interaction effects among our predictor variables. To do this, we can start by adding all of the interaction terms associated with video_length bins and observing our tidy table for significant coefficients.

This is a table with all of our interaction terms:

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	16.509	0.025	671.654	0.000	16.460	16.557
plays	0.095	0.078	1.221	0.224	-0.059	0.249
shares	-0.118	0.050	-2.376	0.019	-0.216	-0.020
comments	0.069	0.043	1.603	0.111	-0.016	0.154
total_videos	0.129	0.029	4.458	0.000	0.072	0.186
video_length_bin2	0.005	0.029	0.160	0.873	-0.052	0.061
video_length_bin3	0.022	0.029	0.755	0.451	-0.035	0.078
likes	0.162	0.085	1.914	0.057	-0.005	0.329
video_length_bin2:likes	-0.028	0.055	-0.510	0.611	-0.137	0.081
shares:video_length_bin2	-0.002	0.056	-0.040	0.968	-0.113	0.108
total_videos:video_length_bin2	-0.009	0.036	-0.235	0.814	-0.081	0.063
video_length_bin3:likes	-0.158	0.062	-2.529	0.012	-0.281	-0.035
shares:video_length_bin3	0.132	0.060	2.202	0.029	0.014	0.250
total_videos:video_length_bin3	-0.041	0.028	-1.473	0.143	-0.095	0.014

We can see from the p-values in the table that likes:video_length_bin3 and shares:video_length_bin3 seem to be the only statistically significant interaction terms. Therefore, those terms should certainly be in our model. All three of the interaction terms associated with video_length_bin2 have extremely high p-values and low coefficients, meaning that video_length_bin2 is insignificant. The p-value for total_videos:video_length_bin3 is not less than our significance level of

0.05, but it is still low. We can now perform two CV tests: one with likes:video_length_bin3 and shares:video_length_bin3, and one with total_videos:video_length_bin3, as well.

Model 1 (with two interaction terms):

RMSE:

```
# A tibble: 1 x 1
   mean
  <dbl>
1 0.339
```

Given that the mean RMSE (0.3389576) is barely larger than our original model without likes and with video_length_bin (0.3398198), we choose the more parsimonious model and select our original model without likes, with video_length_bin, and no interaction terms.

Results

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 rmse    standard       0.624
```

```
# A tibble: 1 x 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 rsq     standard       0.205
```

RMSE of 1.867 (exponentially scaled) and RSQ of 0.205 indicates that the model does poorly in predicting the number of followers. Only about 20.5% of the variability in followers can be explained by our predictor variables. On average, the estimated follower count is about 1.867 followers from the observed values.

```
# A tibble: 77 x 2
   .pred followers
   <dbl>     <dbl>
 1    17        17
 2    18        18
 3    16        16
 4    16        16
 5    17        16
```

```
 6    16        16
 7    17        17
 8    17        17
 9    16        16
10    16        16
# i 67 more rows
```

```
[1] 1.866729
```

term	estimate	std.error	statistic	p.value
(Intercept)	16.512	0.024	675.614	0.000
shares	-0.119	0.046	-2.582	0.011
comments	0.105	0.035	2.982	0.003
plays	0.224	0.046	4.885	0.000
total_videos	0.113	0.027	4.243	0.000
video_length_bin2	0.001	0.029	0.027	0.979
video_length_bin3	0.028	0.029	0.973	0.332

We can see from this model above that there are several terms that are significant when determining the number of followers a tik tok user has. Likes, for example, always had the strongest correlation with followers throughout our modeling process. This makes sense, as likes represent how much the users are enjoying a creator's content. Total videos, as well, seems to have a clear positive relationship with follower count. This also would align with our expectations, as the more videos you make, the more engagement your profile is likely to have. Lastly, we can look at the video length bin variable, which separates a user's average video length into three bins. We can see from the model, that the middle video length bin (2) has statistically significant difference from the other two video length bins, as well as a statistically significant interaction term with total videos. This shows that not only do medium length videos generate the most followers, but medium length videos combined with a higher number of total videos significantly increase follower count as well. This is certainly an interesting finding from our analysis, as it isn't the most expected result.

Conclusion

We originally decided to look at TikTok's data and how follower count (a huge driver of engagement) is impacted by other aspects of a user's account. We learned that it is extremely difficult to correctly predict follower count, given our model only captures 20.5% of the variability in the dataset. More complex models seemed to only worsen performance, and we chose to prioritize parsimony for this reason - however, even the simple models did not predict well.

Our dataset was extremely difficult to work with, given that it did not meet the conditions for linear regression (linearity and constant variance), and contained multicollinearity. The variables were also extremely large, and needed to be scaled down to have meaningful coefficients - which made late interpretation significantly more difficult. A more complex model was likely needed, that was beyond the scope of our knowledge, given how poorly our model performed at the end. There also may be underlying relationships between follower count, and other portions of the TikTok algorithm that are not contained in the dataset, which our model might have also failed to capture. In order to improve our analysis, it would be helpful to comb TikTok for a dataset that potentially contains more variables.

> **!** Important
>
> Before you submit, make sure your code chunks are turned off with `echo: false` and there are no warnings or messages with `warning: false` and `message: false` in the YAML.