Predicting Tik-Tok User Data Based on Video Data

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

2023-12-01

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.

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).

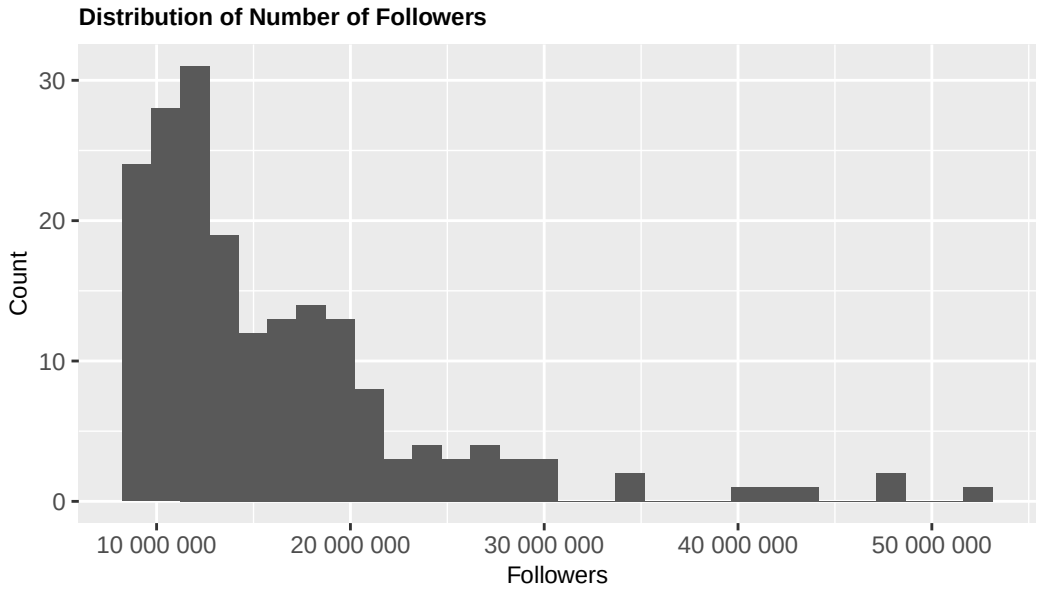
We see that the less relevant variables are create time and video ID. In addition, from looking at our data, hash tags and songs might not be useful. Most videos don't include a hashtag and there are too many unique instances of them for it to be valuable in our analysis. We could consider binning hashtag 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.

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.

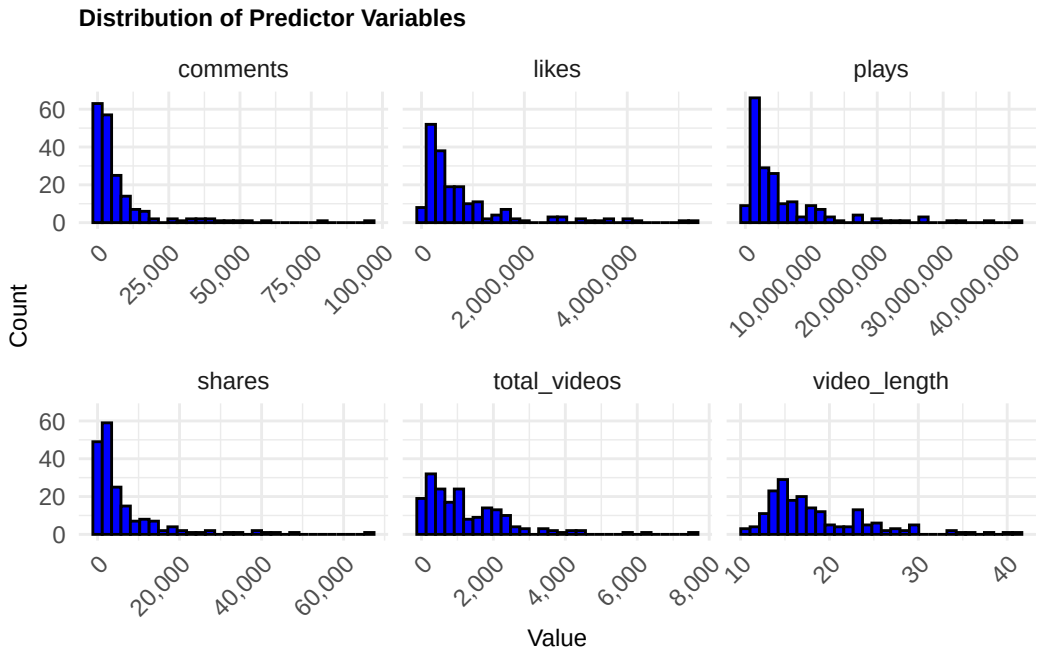
```
<Training/Testing/Total>
<190/64/254>
```

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

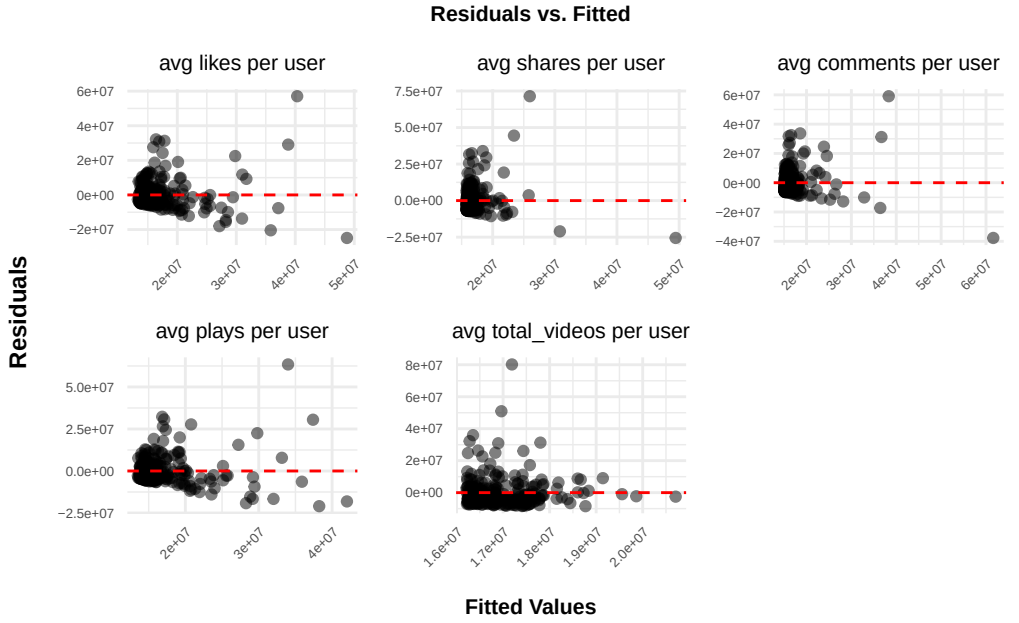


The distribution of our response variable follows, is unimodal and heavily right skewed. The mean is 16,220,526.3 and the standard deviation is 7,710,869.8. The minimum is 8,900,000 and the maximum is 52,300,000. Based on our standard deviation, there seems to be a lot of variation in our dataset; and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

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



juging by the number of outliers in our dataset, we are interesting in knowing how this might influence our model conditions. Hence, we have the following residual plots for each of them.

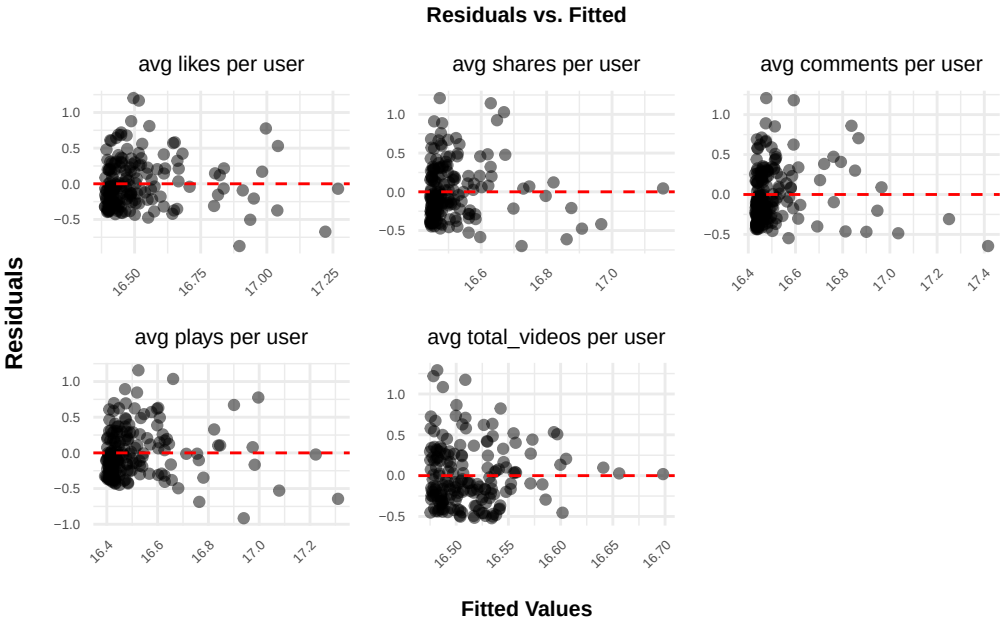


It's clear that all our predictors variables violates constant variance. There's a clear out-

ward 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.

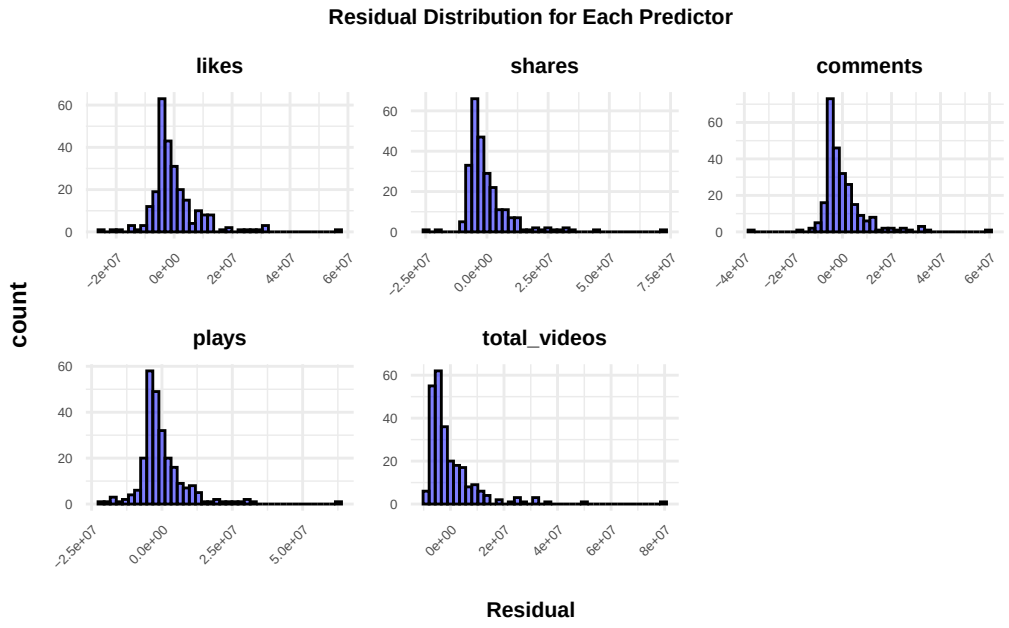
In order to deal with constant variance for the other terms, we log transformed the rest of the predictor variables.



Fanning is a lot less noticeable now. The residual plots don't seem to suggest any underlying patterns. 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

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

In our model selection process, we are interested in detecting any multicollinearity. We found that likes and plays had the highest vif values in our VIF test (11.614 and 9.82 respectively).

```
           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
```

Therefore, 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 meaning they have the highest likelihood for multicollinearity.

Model Comparison with 5-fold CV

To determine which between m1 and m2, we preform 5-fold cross validation and extract the resulting BIC and AIC scores along with additional evaluations.

Model 1: (without likes):

RMSE:

```
# A tibble: 1 x 4
  mean_rmse mean_adj_rsq mean_aic mean_bic
      <dbl>        <dbl>    <dbl>    <dbl>
1     0.340        0.266     90.7     114.
```

Model 2 (without plays):

RMSE:

```
# A tibble: 1 x 4
  mean_rmse mean_adj_rsq mean_aic mean_bic
      <dbl>        <dbl>    <dbl>    <dbl>
1     0.333        0.259     92.1     116.
```

The difference between the model's evaluations aren't large. Model 1 has a slightly higher RMSE, but it has a lower AIC and BIC, and a higher adjusted r-squared. In thise case, we would consider model 1 (the model without likes) to be a better model since it's able to explain more of the variance while also maintaining lower AIC and BIC scores. 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 once again do cross validation to test to see how a model without video_length and likes performs against m1 (our chosen model without likes but contains video length).

RMSE:

Adj. r-sq, AIC, BIC:

```
# A tibble: 1 x 4
  mean_rmse mean_adj_rsq mean_aic mean_bic
      <dbl>        <dbl>    <dbl>    <dbl>
1     0.339        0.269     88.2     106.
```

We can see that when we remove video_length, it makes sense that AIC and BIC both decrease. However, we also see that adjusted r-squared only slightly increased. We wish to further evaluation video_length and potentially tease out some of its importance for our model. Therefore, we explore its role as an interaction term.

Determining whether interaction terms are needed

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. We will be incorporating both likes and plays into this model for a fuller evaluation of video_length as an interaction term. The end goal is to determine whether the parsimonious model of not including likes and video_length bin (model 3) is a better predictor than a model could potentially fully incorporate video_length into inself.

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
likes	0.162	0.085	1.914	0.057	-0.005	0.329
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:video_length_bin2	-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
likes:video_length_bin3	-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 preform CV on a model with likes:video_length_bin and shares:video_length_bin.

Model 4 (with plays:video_length and likes:video_length interaction terms):

RMSE:

```
# A tibble: 1 x 4
  mean_rmse mean_adj_rsq mean_aic mean_bic
      <dbl>        <dbl>    <dbl>    <dbl>
1     0.338        0.284     91.9     130.
```

The mean RMSE (0.3378707) is barely smaller than model3, one without likes and video_length_bin (0.3389576). However given we are have included more terms, our adj r-square shows a slight increase (0.2839 vs. 0.26883, the latter is the parsimonious model), and BIC shows a noticeable decrease (130.3332 vs 105.9608). There's also a difference in AIC (91.944 vs 88.243). Becuase of the noticeable decrease in AIC and BIC going from model 4 to model 3 and because our the increase in our adj r-square isn't significant enough to consider, we choose the more parsimonious model, model 3.

Results

Model 3 performance on test:

```
# A tibble: 2 x 3
  .metric .estimator .estimate
  <chr>   <chr>          <dbl>
1 rmse    standard       0.407
2 rsq     standard       0.362
```

Note that we log transformed our response variable. In order to evaluate the meaning of our RMSE of 0.4067, we take $\exp(0.4067) \sim 1.502$. This value is the multiplicative square difference. For example, if have log followers of 16.04552, our model will be more or less off by $16.04552 \pm \sqrt{.4067} \implies \exp(16.04552 \pm .0.6378) \implies 4914595 < 9,299,954 < 17,601,945$. This means our model does a fairly poor at predicting a tiktok user's followers. We also have an RSQ of 0.3615, indicating only 36.2% of the variability in followers can be explained by our predictor variables.

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.