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.

```
             id create_time     user_name hashtags                         song
1  6.892505e+18  1604786417 charlidamelio       [] Adderall (Corvette Corvette)
2  6.892162e+18  1604706644 charlidamelio       []                original sound
3  6.892157e+18  1604705486 charlidamelio       []                original sound
4  6.891688e+18  1604596107 charlidamelio       []                original sound
5  6.891016e+18  1604439653 charlidamelio       []                original sound
6  6.890973e+18  1604429723 charlidamelio       []      Lemonade Internet Money
7  6.890944e+18  1604423084 charlidamelio       []                original sound
8  6.890783e+18  1604385576 charlidamelio       []                    All I Want
9  6.890716e+18  1604369764 charlidamelio       []                original sound
10 6.890712e+18  1604369025 charlidamelio       []                   hammer time
   video_length n_likes n_shares n_comments  n_plays n_followers n_total_likes
1            15  480800     9256      51300  1900000    97400000       7.6e+09
2             9 3100000    17200     105700 13300000    97400000       7.6e+09
3             4 2400000    17800      69200 10100000    97400000       7.6e+09
4            15 3200000    12700      64100 14600000    97400000       7.6e+09
5            13 7500000    31100     290300 34700000    97400000       7.6e+09
6             7 7100000    43000      82000 33300000    97400000       7.6e+09
7            13 3200000     8610      43600 17800000    97400000       7.6e+09
8            38 4100000    25000      55500 19400000    97400000       7.6e+09
9            13 4000000    39100      63100 20100000    97400000       7.6e+09
10            7 5800000    26800      61400 26600000    97400000       7.6e+09
   n_total_vids
1          1642
2          1642
3          1642
4          1642
5          1642
6          1642
7          1642
8          1642
9          1642
10         1642
```

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.

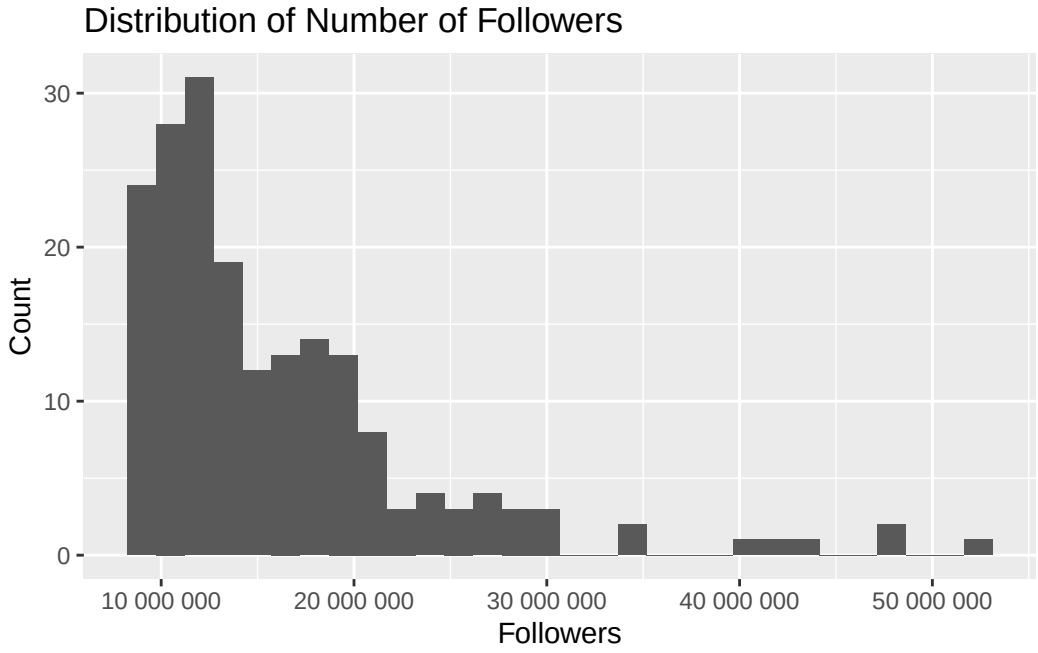
To address the issues mentioned above, we grouped the data by users and summarized all relevant predictor variables by taking their mean. Our modified dataset now has 8 columns and 254 observations.

```
# A tibble: 10 x 8
   user_name    likes shares comments  plays followers video_length total_videos
   <chr>        <dbl>  <dbl>    <dbl>  <dbl>     <dbl>        <dbl>        <dbl>
 1 .kunno      3.35e5  1067.    4521. 1.99e6  15300000         19.5         3442
 2 _arishfakh~ 4.26e5  5269.    2700. 3.96e6  28600000         14.5         2026
 3 _saloniyaa~ 1.67e5  6122.     728. 1.85e6  12900000         14.6         2005
 4 aashikabha~ 1.94e5  1335.    1053. 2.17e6  16000000         15.0         2720
 5 abbyrartis~ 9.66e5  5292.    7005. 3.97e6  13500000         16.5          811
 6 abrazkhan91 2.69e5  2074.     847. 2.08e6  15900000         26           2102
 7 addisonre   5.72e6 50692.   96540  4.01e7  67900000         12           1261
 8 afshanrooh  2.44e5  4673.     689. 2.96e6   9300000         14.9         1377
 9 alex.stemp  8.11e5  2636.    3352. 4.35e6  10700000         40.3          184
10 alexcasasvz 2.60e5   143.    2954. 9.31e5  13100000         22.7         1612
```

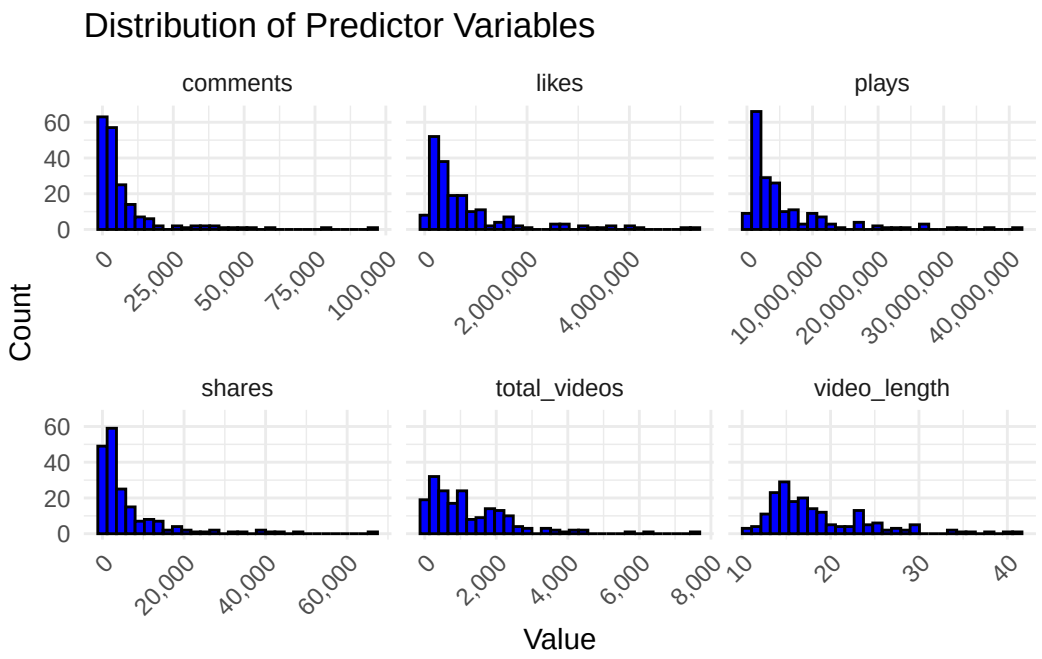
Note that no data leakage is introduced in this process since we are just summarizing by the means of the predictor variable per user. When we split, it'll split based on observations, users. We'll now split our data into testing and training.

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

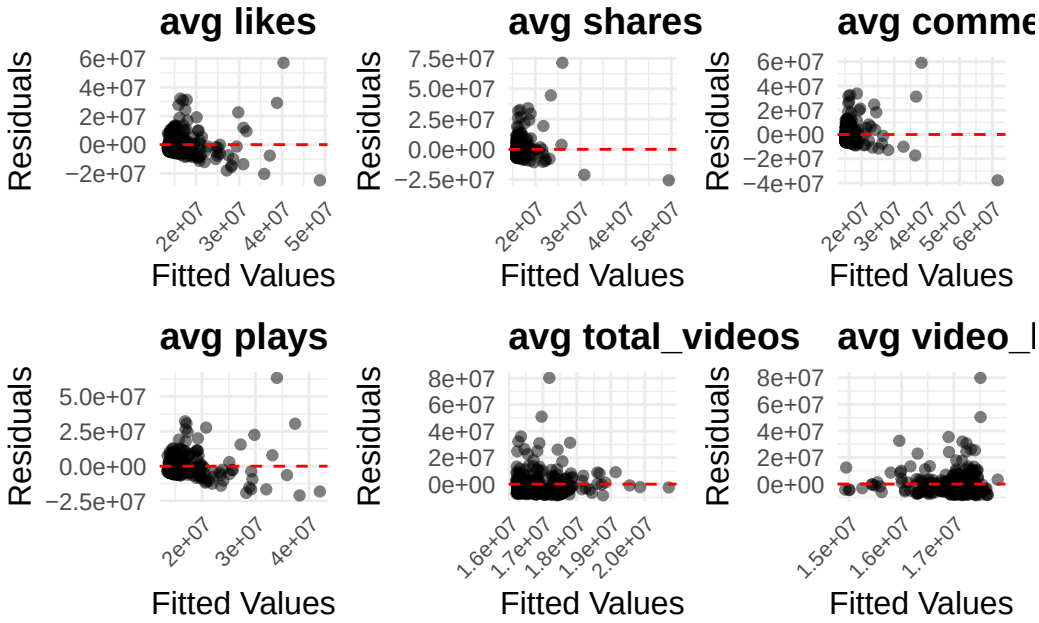
Here's a distribution of our response variable, user followers, from our training set.



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.

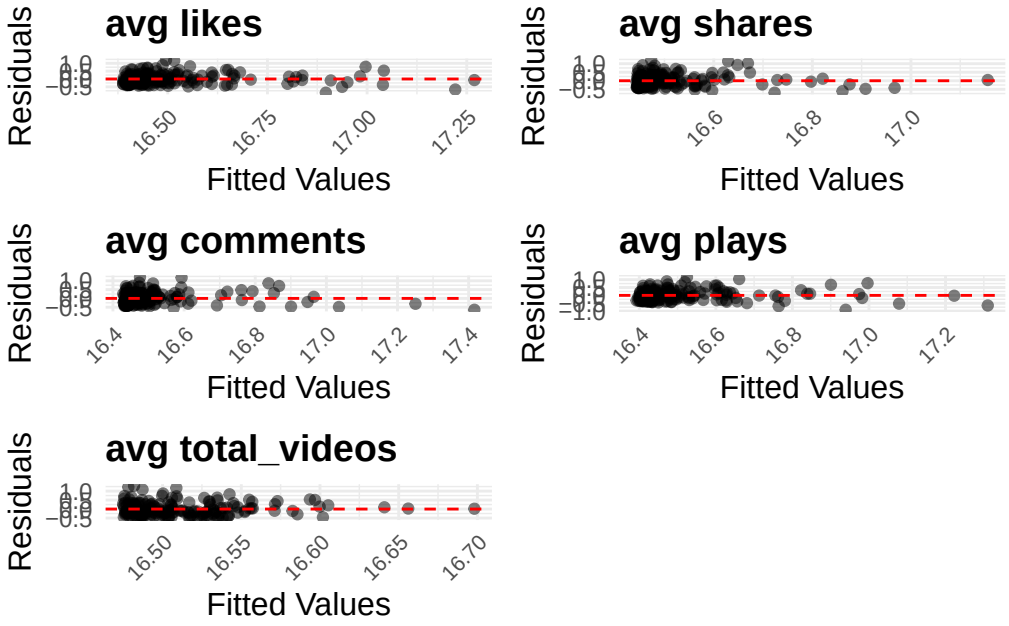


It's clear that all our predictors variables violates constant variance. There's a clear out-

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.



Although there's still slight signs of fanning going on, it's a lot less severe than before. For now, this concludes our EDA/data cleaning process and we move onto our model selection process.

Methodology

This section includes a brief description of your modeling process. Explain the reasoning for the type of model you're fitting, predictor variables considered for the model including any interactions. Additionally, show how you arrived at the final model by describing the model selection process, any variable transformations (if needed), and any other relevant considerations that were part of the model fitting process.

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

```
# A tibble: 2 x 6
  .metric .estimator  mean     n std_err .config
  <chr>   <chr>      <dbl> <int>   <dbl> <chr>
1 rmse    standard   0.340     5  0.0255 Preprocessor1_Model1
2 rsq     standard   0.263     5  0.0800 Preprocessor1_Model1
```

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

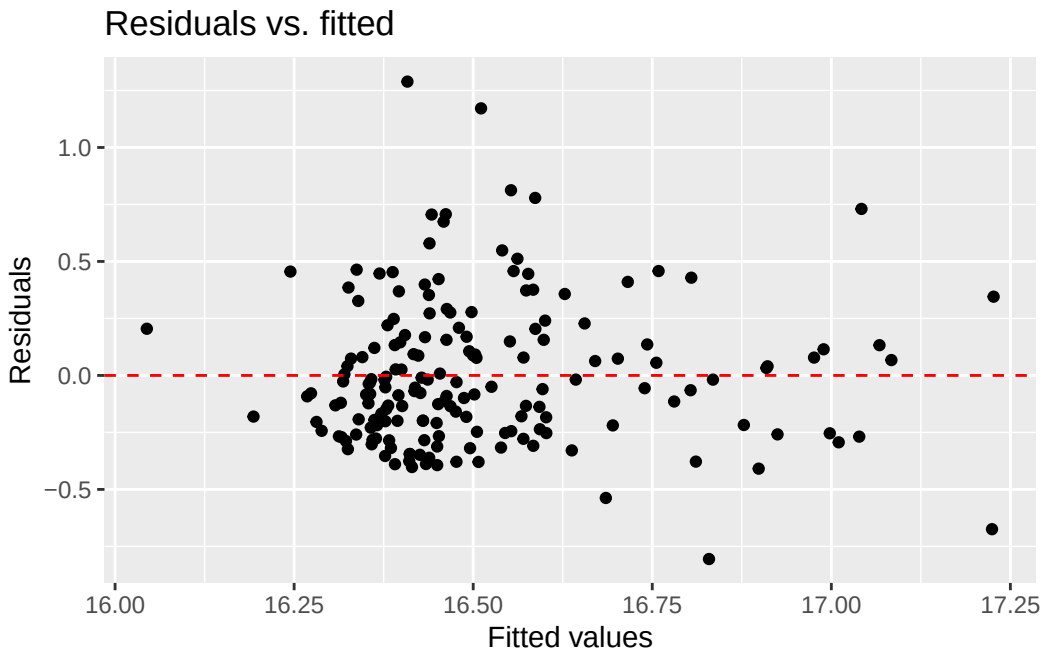
```
# A tibble: 2 x 6
  .metric .estimator  mean     n std_err .config
  <chr>   <chr>      <dbl> <int>   <dbl> <chr>
1 rmse    standard   0.333     5  0.0287 Preprocessor1_Model1
2 rsq     standard   0.248     5  0.0765 Preprocessor1_Model1
```

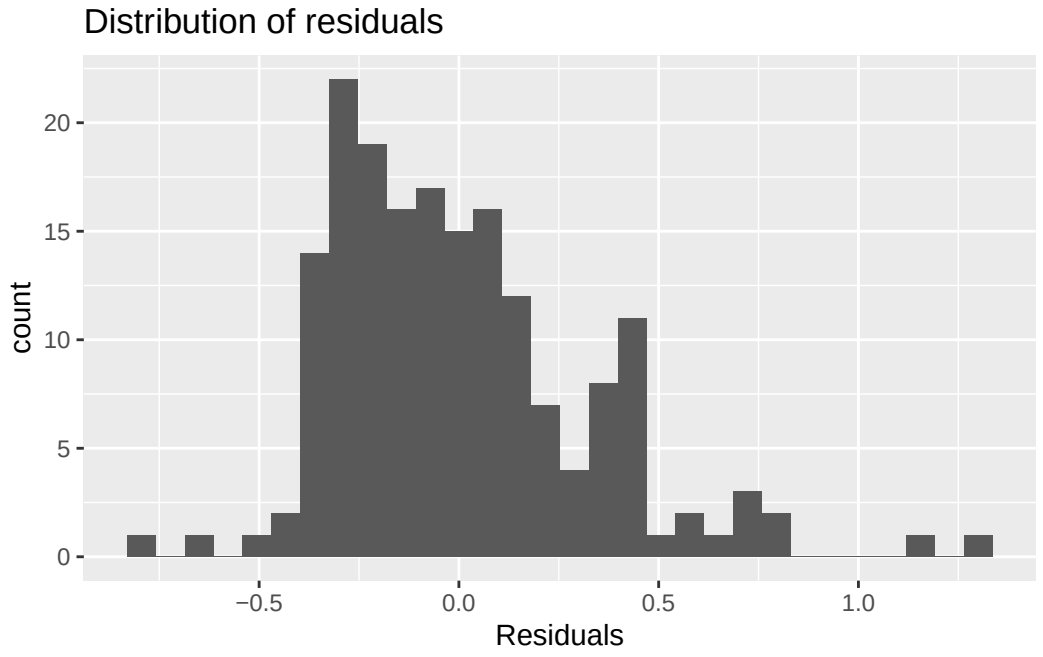
```
# A tibble: 1 x 3
  mean_adj_rsq mean_aic mean_bic
         <dbl>    <dbl>    <dbl>
1        0.259     92.1     116.
```

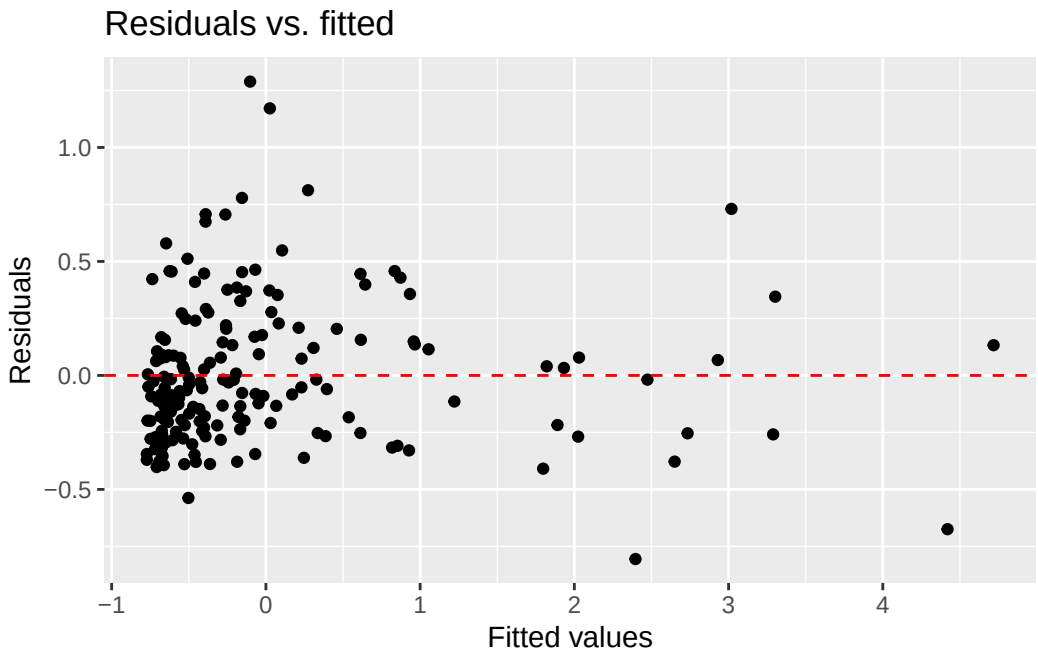
```
# A tibble: 1 x 3
  adj.r.squared   AIC   BIC
          <dbl> <dbl> <dbl>
1         0.261  113.  139.
```

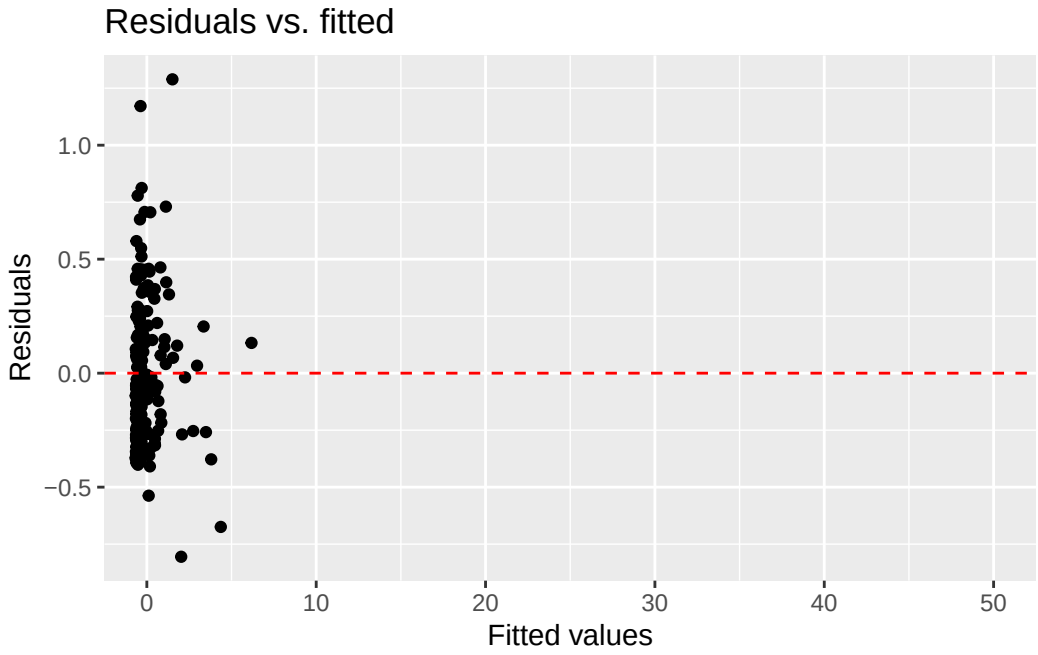
```
# A tibble: 1 x 3
  adj.r.squared   AIC   BIC
          <dbl> <dbl> <dbl>
1         0.257  115.  140.
```

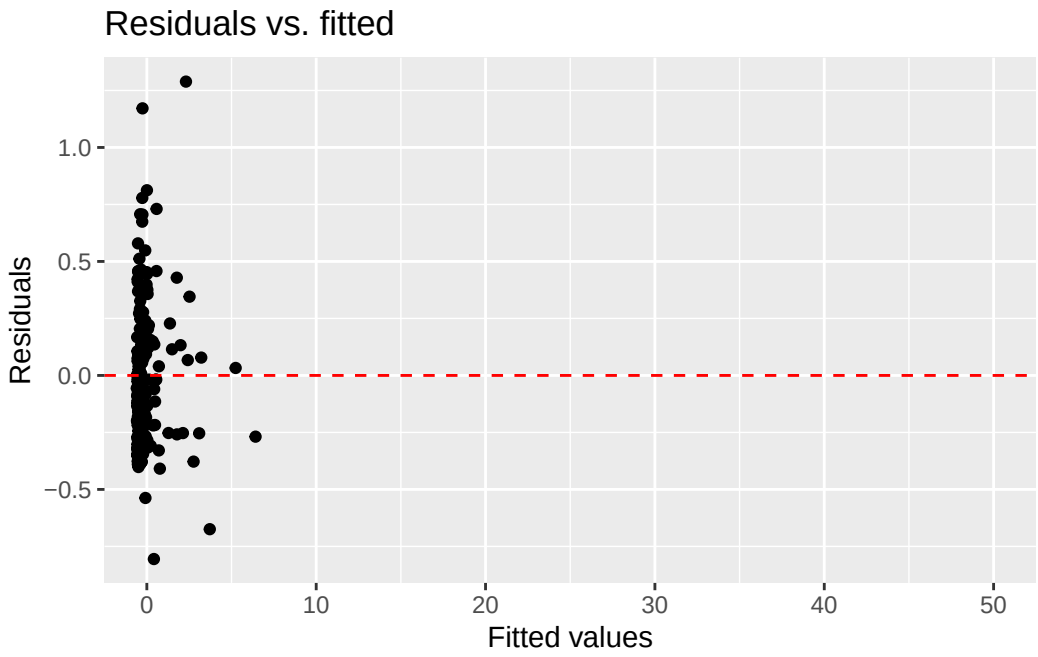
Based on AIC and BIC, model 2 (the model without plays) is a better fit. Therefore, we choose to remove plays from the model and leave likes in the model to deal with the multicollinearity.

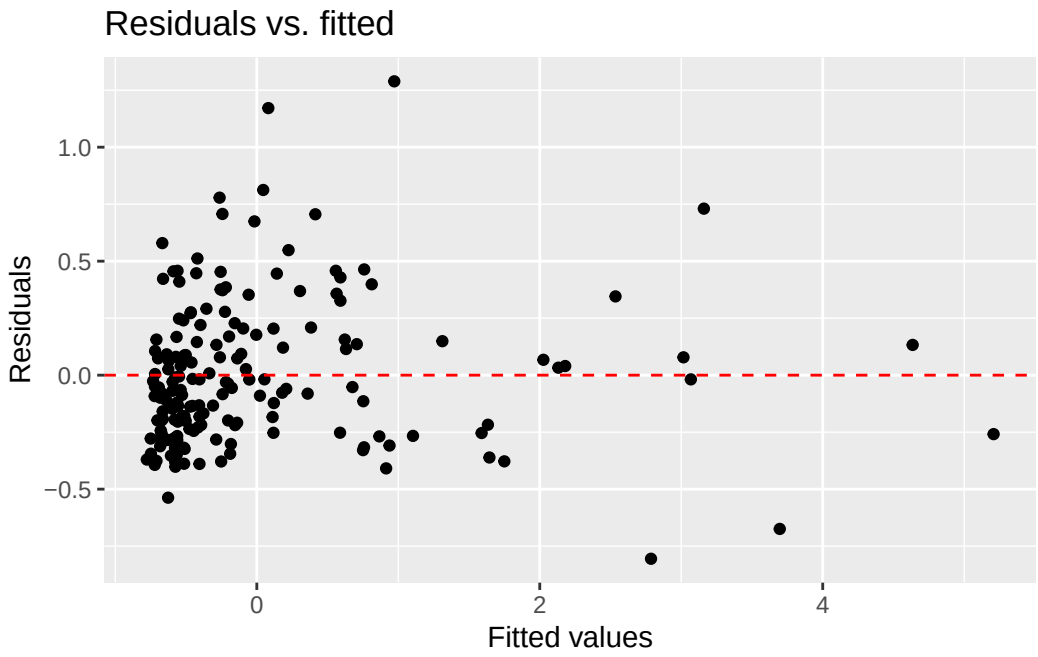

Residuals vs. fitted
Residuals
1.0
0.5
0.0
−0.5
16.00
16.25
16.50
16.75
17.00
17.25
Fitted values

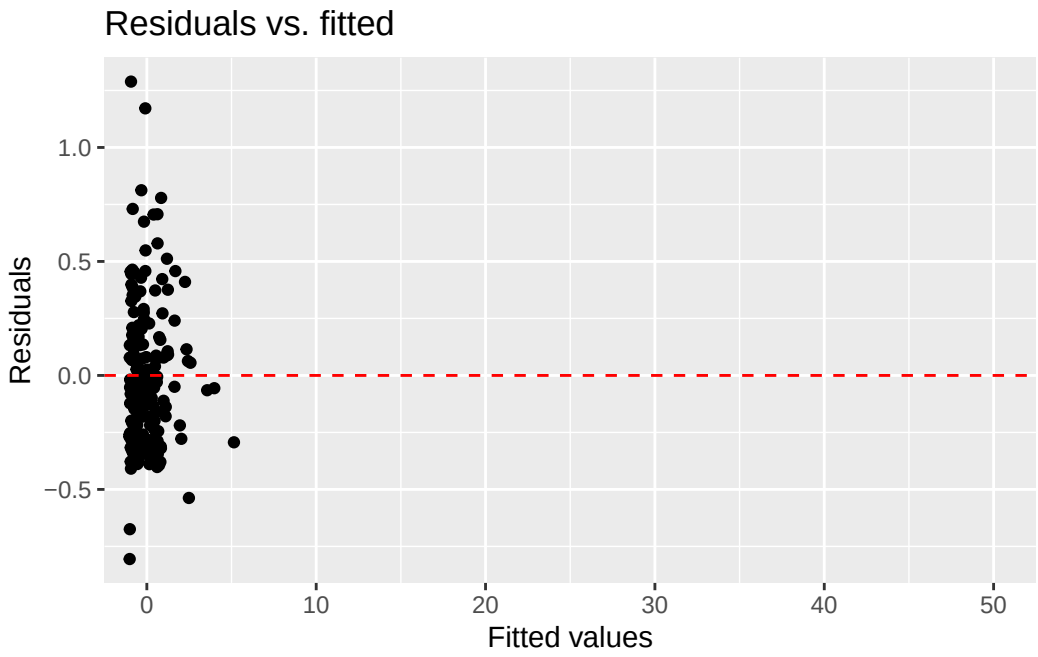

Distribution of residuals
count
20
15
10
5
0
−0.5
0.0
0.5
1.0
Residuals


Residuals vs. fitted
Residuals
1.0
0.5
0.0
−0.5
−1
0
1
2
3
4
Fitted values


Residuals vs. fitted
Residuals
1.0
0.5
0.0
−0.5
0
10
20
30
40
50
Fitted values


Residuals vs. fitted
Residuals
1.0
0.5
0.0
−0.5
0
10
20
30
40
50
Fitted values


Residuals vs. fitted
Residuals
1.0
0.5
0.0
−0.5
0
2
4
Fitted values



Determining whether interaction terms are needed

term	estimate	std.error	statistic	p.value
(Intercept)	16.507	0.025	671.234	0.000
likes	0.243	0.053	4.627	0.000
shares	-0.099	0.047	-2.098	0.037
comments	0.054	0.041	1.303	0.194
total_videos	0.125	0.029	4.350	0.000
video_length_bin2	0.004	0.029	0.143	0.887
video_length_bin3	0.020	0.029	0.706	0.481
likes:video_length_bin2	-0.023	0.055	-0.412	0.681
shares:video_length_bin2	-0.002	0.056	-0.029	0.977
total_videos:video_length_bin2	-0.009	0.036	-0.252	0.801
likes:video_length_bin3	-0.166	0.062	-2.677	0.008
shares:video_length_bin3	0.143	0.059	2.411	0.017
total_videos:video_length_bin3	-0.041	0.028	-1.485	0.139

term	estimate	std.error	statistic	p.value
(Intercept)	16.517	0.025	666.512	0.000

term	estimate	std.error	statistic	p.value
likes	0.232	0.050	4.639	0.000
shares	-0.092	0.044	-2.097	0.037
comments	0.075	0.040	1.861	0.064
total_videos	0.114	0.029	3.906	0.000
video_length_bin2	-0.012	0.025	-0.490	0.625
shares:video_length_bin2	-0.037	0.026	-1.449	0.149
total_videos:video_length_bin2	0.006	0.033	0.182	0.856

```
# A tibble: 1 x 3
  adj.r.squared   AIC   BIC
          <dbl> <dbl> <dbl>
1         0.257  115.  140.
```

```
# A tibble: 1 x 3
  adj.r.squared   AIC   BIC
          <dbl> <dbl> <dbl>
1         0.260  115.  143.
```

Results

In this section, you will output the final model and include a brief discussion of the model assumptions, diagnostics, and any relevant model fit statistics.

This section also includes initial interpretations and conclusions drawn from the model.

RMSE of 1.399284 indicates the average prediction error in terms of the log.

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 vidoes, 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.

! 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.