Predicting Tik-Tok User Data Based on Video Data

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

2023-12-01

Introduction and data

With over 1 billion users globally, TikTok is one of the fastest growing social platforms in the world. Understanding ubiquitous algorithm, which is said to generally account for account factors (likes and comments) and video information (captions, sounds, hashtags), is critical to understanding the app's many critiques, from declining youth mental health outcomes and its addictive nature of its explore page. To better understand TikTok's social impact, we decided to explore 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 expanded the list of users by collecting suggested users with the API's getSuggestedUsersbyIDCrawler method. They then collected video data of the 25 most recent posts of each user using the byUsername method. They also used the 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 start "trending."

EDA

We begin our EDA process by first examining the dataset.

Currently, our dataset tiktok has 13 columns and 12,559 observations. Each row is a video. The columns cover attributes of each video such as video length, hashtags used, songs/sounds used, and statistics (number of likes, shares, comments, plays, followers, and total number of likes and videos across the account). 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.

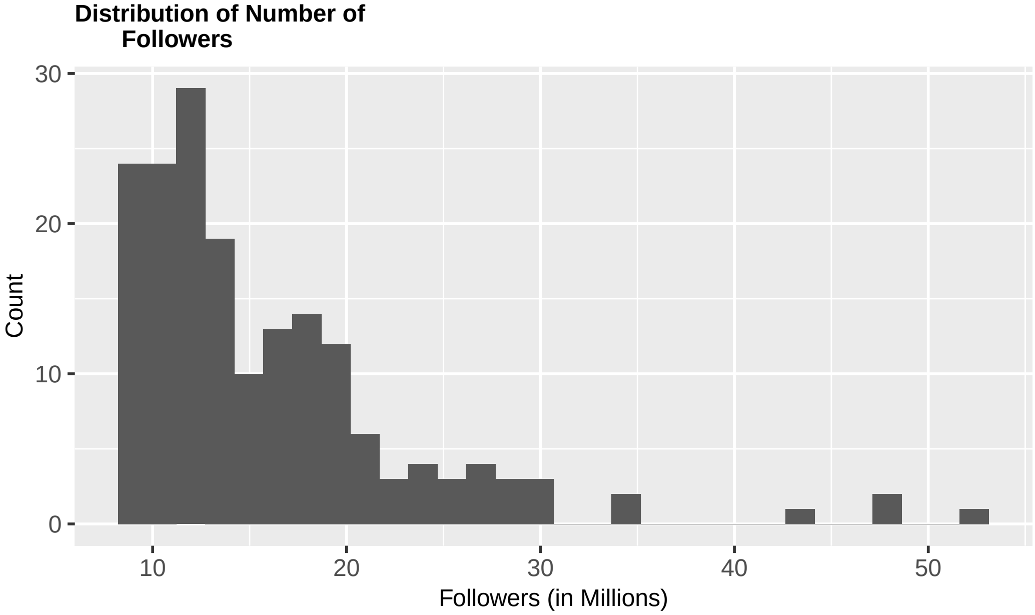
However, it's clear that some of the columns won't be useful for predicting number of followers. It is also apparent that we must address the potential issue user_name might have with the other columns. There's a potential for severe multicollinearity if we choose to just drop user_name, since the number of plays or likes a video would have a strong relationship with the user who posted 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).

The less relevant variables are create time and video ID. In addition, hashtags 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 mind the number of 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 relevant predictor variables by taking their mean. Our modified dataset has 8 columns and 254 observations, with each row being a user.

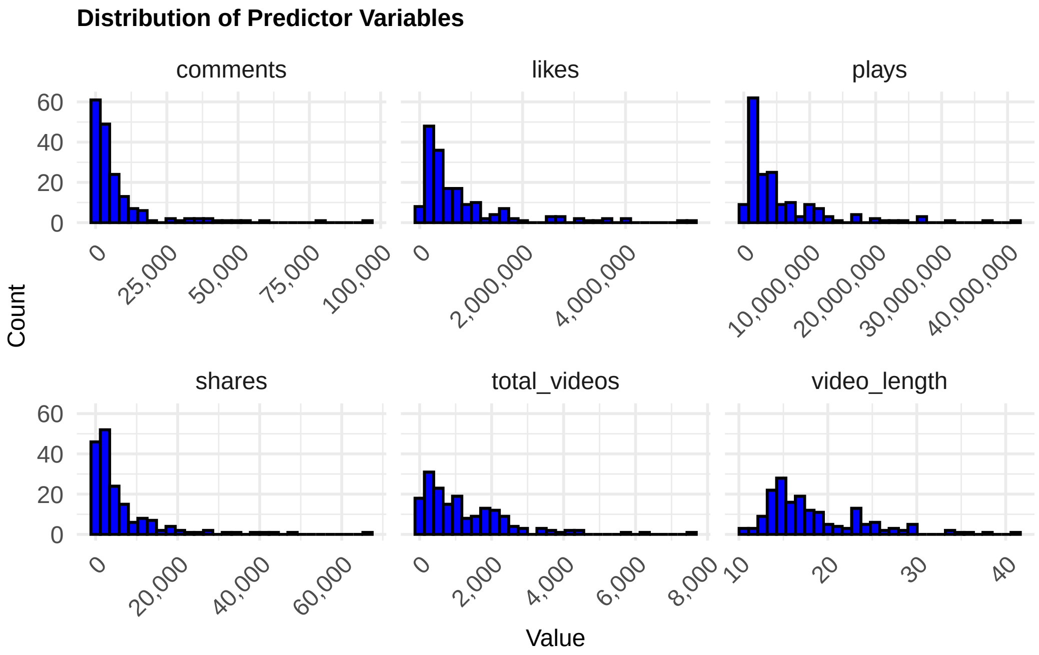
Note that no data leakage is introduced in this process since we are just summarizing by the means of the predictor variable per user. When we split, it'll split based on observations, which are users.

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, we can see major outliers.

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



Methodology

We want to use multiple linear regression to predict the number of followers a user has. We choose multiple linear regression rather than logistic regression because followers is a quantiative response variable. We start off with an initial model containing the predictors likes, shares, comments, plays, video_length (factor with 3 levels), total_videos, and followers, our response variable. Because Tiktok videos are commonly divided into 15-second, 1 minute, or 3 minute videos, we bin average video length into 3 levels, corresponding to “short”, “medium” and “long.” We also mean-center all our numerical variables to make our intercept meaningful. Here is a tidy table of our initial model:

term	estimate	std.error	statistic	p.value
(Intercept)	15546821.6420	854032.1390	18.2040	0.0000
likes	1.7238	1.7323	0.9951	0.3211
shares	-276.8025	94.9198	-2.9162	0.0040
comments	104.1125	58.8750	1.7684	0.0788
plays	0.4979	0.2225	2.2373	0.0266
video_lengthbin2	554321.5366	1222113.2347	0.4536	0.6507
video_lengthbin3	976399.9780	1209972.1453	0.8070	0.4208
total_videos	1524.8024	434.0951	3.5126	0.0006

Detecting Multicollinearity & Model Comparison

Upon conducting a VIF test, we found that likes and plays had the highest vif values (11.614 and 9.82 respectively):

	GVIF	Df	GVIF^(1/(2*Df))
likes	11.614	1	3.408
shares	3.537	1	1.881
comments	2.682	1	1.638
plays	9.820	1	3.134
video_length	1.079	2	1.019
total_videos	1.182	1	1.087

Therefore, we wanted to assess which model would perform better: a model without likes or a model without plays. To do this, we performed 5-fold cross validation and extracted the resulting AIC, BIC, adj.r-squared, and RMSE values for the two models:

Model 1: (without likes):

```
# A tibble: 1 x 4
  mean_rmse mean_adj_rsq mean_aic mean_bic
      <dbl>        <dbl>    <dbl>    <dbl>
1  6774943.        0.259    4846.    4867.
```

Model 2 (without plays):

```
# A tibble: 1 x 4
  mean_rmse mean_adj_rsq mean_aic mean_bic
      <dbl>        <dbl>    <dbl>    <dbl>
1  6626034.        0.239    4850.    4870.
```

The difference between the model's evaluations aren't large. Model 1 has a higher RMSE, while it has a lower AIC and BIC, and a higher adjusted r-squared. In this case, we would consider model 2 (the model without plays) to be a better model, because it has a lower RMSE, which is gathered from the assessment set and is used to assess prediction. The goal of our model is to predict followers, so we want to choose the model with better predictive power (Model 2). Therefore, we remove plays from our model.

Determining whether video_length_bin are necessary

We saw from our initial tidy table that the p-values associated with video length bins are high, indicating that the variables may not be significant. Because of this, we can once again perform cross validation to test how a model without video_length compares to our current model (Model 1):

Model 3 (without plays and video length):

```
# A tibble: 1 x 4
  mean_rmse mean_adj_rsq mean_aic mean_bic
      <dbl>        <dbl>    <dbl>    <dbl>
1  6663499.        0.234    4850.    4867.
```

We can see that when we remove video_length, RMSE is slightly higher than it was for Model 2, while AIC remains about the same and BIC slightly decreases. We also see that adjusted r-squared remains about the same. Therefore, despite BIC slightly decreasing, we prefer the model with a lower RMSE (better prediction), so we don't want to remove video_length from our model.

Determining whether interaction terms are needed

Our only categorical variable in our model is video_length. Therefore, we can include all possible interaction terms with video_length and assess which combinations look significant:

term	estimate	std.error	statistic	p.value
(Intercept)	15541087.342	807753.873	19.240	0.000
likes	8.648	2.855	3.030	0.003
shares	-482.841	196.014	-2.463	0.015
comments	66.733	191.688	0.348	0.728
total_videos	2258.583	667.260	3.385	0.001
video_lengthbin2	851635.204	1133531.524	0.751	0.454
video_lengthbin3	2195535.102	1174580.262	1.869	0.063
likes:video_lengthbin2	-0.637	3.145	-0.203	0.840
likes:video_lengthbin3	-11.138	3.289	-3.386	0.001
shares:video_lengthbin2	75.107	230.064	0.326	0.744
shares:video_lengthbin3	521.285	248.735	2.096	0.038
comments:video_lengthbin2	-117.733	201.728	-0.584	0.560
comments:video_lengthbin3	744.228	254.445	2.925	0.004
total_videos:video_lengthbin2	-204.878	1174.317	-0.174	0.862
total_videos:video_lengthbin3	-1376.528	891.946	-1.543	0.125

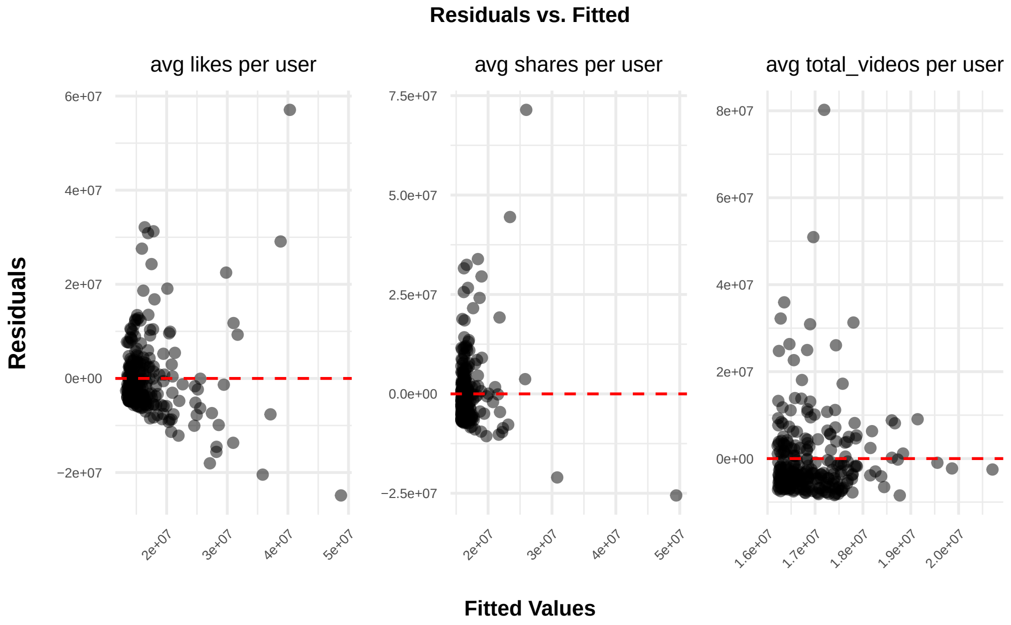
We can see from the table that all variables are significant when interacting with video_lengthbin3 (p-value is less than significance level of 0.05) except for total_videos. Because of this, we know that we won't need to include the interaction term between total_videos and video_length. Also, given that comments has a high p-value in this new model, we can try removing comments from our model as well. We can use cross validation to test how a model without comments and with video_length interacting with shares and likes performs compared to our current model (Model 2):

```
# A tibble: 1 x 4
  mean_rmse mean_adj_rsq mean_aic mean_bic
      <dbl>        <dbl>    <dbl>    <dbl>
1  6446434.        0.287    4841.    4865.
```

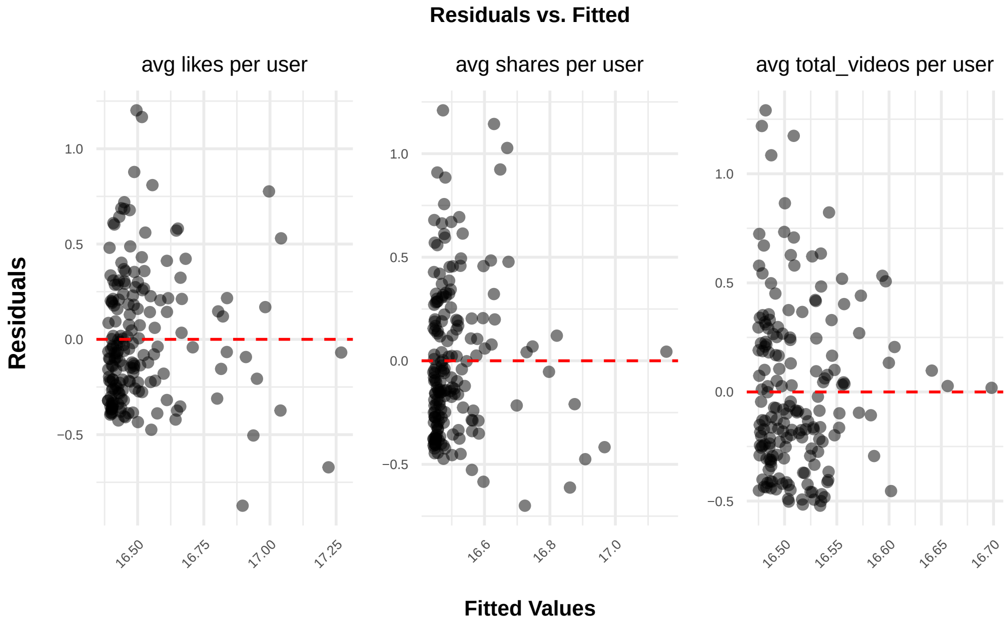
We can see that RMSE significantly decreased from about 6.6 million in Model 2 to 6.4 million in Model 4. We also see that adjusted r-squared increased, AIC decreased, and BIC decreased. All of these signs point to Model 4 being a better model in both fit and prediction. Therefore, we will remove comments, and add interactions between video_length and both shares and likes.

Conditions for Inference

In assessing linearity and constant variance, it is important to look at Residual vs. Fitted plots for quantitative predictor variables (in our current model that's likes, shares, and total videos) and look for patterns and fanning:



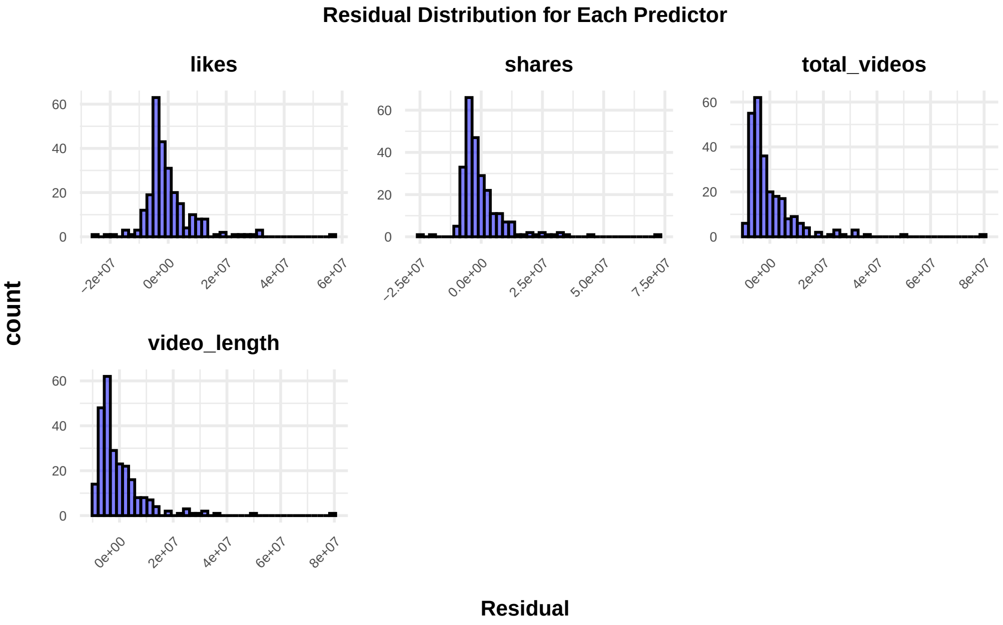
We can see from the residual plots that there doesn't appear to be any non-random patterns that violate linearity. Therefore, we can conclude that the linearity condition is satisfied. However, there does appear to be a clear outward fanning spread for each each predictor, meaning that constant variance is not satisfied. To solve this, we can log-transform our response variable (followers) and see if the fanning is minimized:



We can see that after log-transorming followers, the scale of our y axis decreases significantly. Additionally, there is no clear outward fanning, but rahter a lower density of points as you move to higher values on the x-axis As such, we conclude the predictors satisfy constant variance.

When assessing independence, we know that each of the videos are by individual creators, therefore the videos were produced independently of each other. There is no reason to believe that one TikTok user's video performance would directly affect another's.

Finally, we assess normality by looking at the residual histograms for each predictor:



Normality doesn't seem to be satisfied for each of the predictors, however, because we have more than 30 observations in the dataset, we can conclude that normality is satisfied regardless of the distribution.

Results

After removing plays and comments and adding interaction terms between video_length and both likes and shares, we arrive at our final model:

term	estimate	std.error	statistic	p.value
(Intercept)	16.4959	0.0434	380.0932	0.0000
shares	0.0000	0.0000	-1.7409	0.0835
likes	0.0000	0.0000	3.9769	0.0001
total_videos	0.0001	0.0000	4.4614	0.0000
video_lengthbin2	0.0063	0.0608	0.1031	0.9180
video_lengthbin3	0.0317	0.0604	0.5246	0.6006
likes:video_lengthbin2	0.0000	0.0000	-0.1557	0.8765
likes:video_lengthbin3	0.0000	0.0000	-2.6748	0.0082
shares:video_lengthbin2	0.0000	0.0000	-0.1003	0.9202
shares:video_lengthbin3	0.0000	0.0000	2.3846	0.0182

Final Model performance on testing set:

.metric	.estimator	.estimate
rmse	standard	23434875.554
rsq	standard	0.187

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) ~ 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 $16.04552 \pm .4067^2 \implies \exp(16.04552 \pm 0.1654) \implies 7882219 < 9,299,954 < 10,972,689$. 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.

There are several terms that are significant when determining the number of followers a tik tok user has. The number of total videos, comments, and plays 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 and more followers you may gain. However, shares have a negative relationship with follower count, which initially seemed counter-intuitive. While it is impossible for the model to determine causality or explain why exactly a relationship exists, we hypothesize that users may share a video because they dislike it, resulting in them not following the user.

When observing the video length bin variable, 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.

Discussion + 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 36.2% 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 the real world, users have reported that TikTok enforces policies differently from user-to-user, and uses different algorithms from region to region.

In order to improve our analysis, it would be helpful to comb TikTok for a dataset that potentially contains more variables. Three potential options we considered included: finding a meaningful way to capture hashtags (which may require manually looking at TikTok videos), finding a meaningful way to capture whether a user typically utilizes trending music, and using the demographic statistics for users (to account for human decisionmaking).

Appendix

First 5 data points before transformation

```
            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
  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
  n_total_vids
1         1642
2         1642
3         1642
4         1642
5         1642
6         1642
```

After transformation

```
# 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
```

Split results:

<a name=“appendix”></a>

```
<Training/Testing/Total>
<177/77/254>
```

> ❗ 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.