Twitch Streamer Popularity

Jason Suerte
Arhum Khan
Chichen Chao

MATH 4322 - 16022
December 1st, 2023

Introduction (Arhum Khan)

In today's digital age, live streaming has taken center stage, with Twitch leading the charge as the go-to platform for gamers, content creators, and their audiences. Our group has chosen a data set from Kaggle.com, meticulously curated to capture the statistical landscape of the top 1000 Twitch streamers. With 1000 observations and 11 variables, our data set provides a comprehensive view of the streaming landscape. From audience watch time and streaming duration to follower metrics, viewer metrics, and content categorization, each variable offers valuable insights into the dynamic world of digital entertainers.

Through this project, we aim to explore the intricacies of Twitch streaming by analyzing our key metrics in multiple linear regression and random forest. As Twitch continues to shape online communities, our project aims to dive into the data, uncovering trends and patterns that define the success of these virtual entertainers. **What factors help Twitch Streamers become more popular?**

About the Data & Methods (Arhum Khan)

Using the data set of our Twitch streamers in 2022, we have constructed two important questions to help direct our investigation and extract meaningful information. Among the variables are:

- Channel: The name of the streamer channel and observation.
- Watch.time.minutes: The total audience watch time in minutes of this streamer. This directly ranks the streamer placement.
- Stream.time.minutes: The total duration of the streaming in minutes.
- Peak.viewers: The highest number of concurrent viewers during a streaming session.
- Average.viewers: The mean number of viewers throughout a streaming session.
- Followers: The total number of followers for a Twitch channel.
- Followers.gained: The number of new followers acquired during a specific time-period.
- Views.gained: The number of new views received during a specific time-period.
- Partnered: If the channel is partnered with Twitch (True/False).
- Mature: If the channel is for a mature audience or not (True/False).
- Language: The main language the channel broadcasts in (English, Japanese, etc).

In the multiple linear regression model, the aim is to predict the total watch time a streamer has accumulated in minutes (Watch.time.Minutes) using the remaining predictors. Because of the model's interpretability and simplicity, linear regression is a computationally efficient tool that works well in situations where relationships are roughly linear and normal. Its dependence on the linearity assumption, sensitivity to outliers, and restricted ability to manage non-linearities, however, are significant disadvantages. We chose this model because of its ability to predict the total streamer's watch time and to observe how the predictors affect our response variable.

For the Random Forest model, it combines the advantages of several decision trees to achieve exceptional predictive accuracy. Their ability to manage intricate relationships and non-linearities in the data enables them to capture complex patterns that may prove difficult for simpler models to accurately represent. Furthermore, Random Forests offer insights into the significance of features, which help identify the critical variables impacting the predictions. These benefits do, however, come with costs. Because the model is ensemble-based, its higher computational complexity might necessitate additional resources during training. It is also more difficult to accurately determine how each variable affects the predicted outcomes when using the Random Forest model than when using simpler models like linear regression. Additionally, there is a chance of overfitting, which can result in poorer generalization performance on unobserved data, especially when utilizing big or extremely complex ensembles. We chose this model because it covers intricate relationships in contrast to the linear regression. We wanted to also explore and predict the followers gained to provide insight into how a streamer can increase their audience.

Linear Regression Model (Jason Suerte & Arhum Khan)

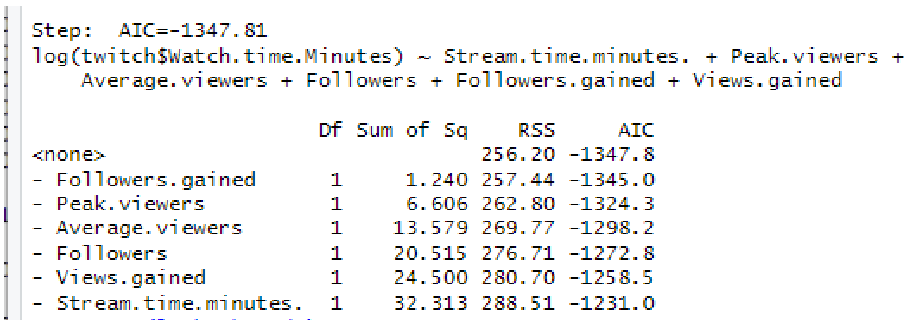
The purpose of this model is to infer the total watch time that a streamer has accumulated in minutes (Watch.time.Minutes), using multiple predictors as described above. However, due to our response variable ranging from millions to billions, we will use a logarithmic function to help make our data more normal and fit. If we are to use this linear regression, we must assume normality and a logarithmic transformation will meet this standard.

We will then find our initial multiple linear regression that includes all predictors- excluding Partnered since every observation is true.

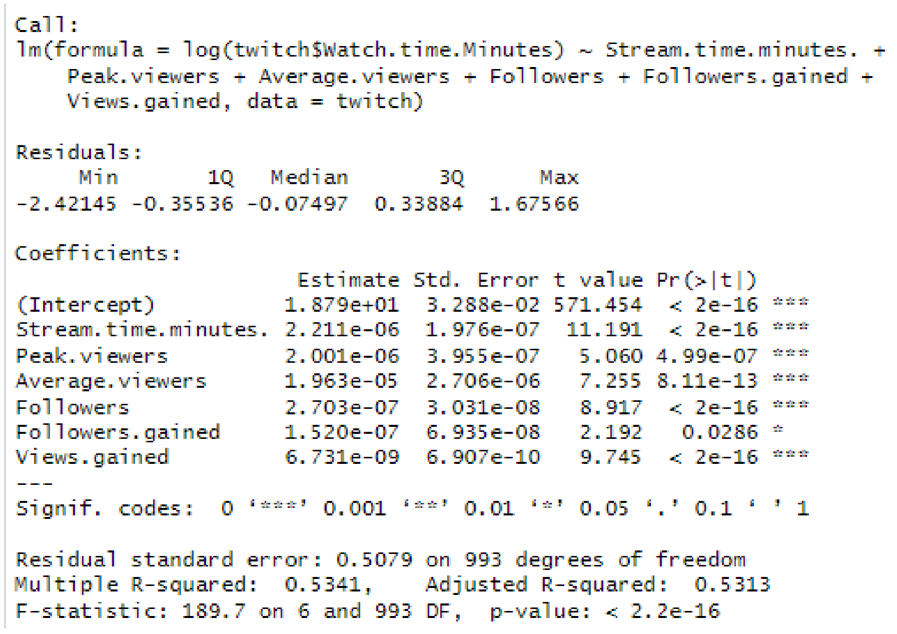
```
lm.1 <- lm(log(twitch$Watch.time.Minutes)~.-Channel-Watch.time.Minutes.-Partnered, data =
        twitch)
summary(lm.1)
```

The summary will produce many insignificant variables where the p-values are >0.05. We can get rid of these using a backward stepwise function.

```
lm.2 <- step(lm.1, direction = "backward")
summary(lm.2)
```



```
Step:  AIC=-1347.81
log(twitch$Watch.time.Minutes) ~ Stream.time.minutes. + Peak.viewers +
    Average.viewers + Followers + Followers.gained + Views.gained

                       Df Sum of Sq    RSS     AIC
<none>                              256.20 -1347.8
- Followers.gained      1     1.240 257.44 -1345.0
- Peak.viewers          1     6.606 262.80 -1324.3
- Average.viewers       1    13.579 269.77 -1298.2
- Followers             1    20.515 276.71 -1272.8
- Views.gained          1    24.500 280.70 -1258.5
- Stream.time.minutes.  1    32.313 288.51 -1231.0
```



```
Call:
lm(formula = log(twitch$Watch.time.Minutes) ~ Stream.time.minutes. +
    Peak.viewers + Average.viewers + Followers + Followers.gained +
    Views.gained, data = twitch)

Residuals:
     Min       1Q   Median       3Q      Max
-2.42145 -0.35536 -0.07497  0.33884  1.67566

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)          1.879e+01  3.288e-02 571.454  < 2e-16 ***
Stream.time.minutes. 2.211e-06  1.976e-07  11.191  < 2e-16 ***
Peak.viewers         2.001e-06  3.955e-07   5.060 4.99e-07 ***
Average.viewers      1.963e-05  2.706e-06   7.255 8.11e-13 ***
Followers            2.703e-07  3.031e-08   8.917  < 2e-16 ***
Followers.gained     1.520e-07  6.935e-08   2.192   0.0286 *
Views.gained         6.731e-09  6.907e-10   9.745  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5079 on 993 degrees of freedom
Multiple R-squared:  0.5341,    Adjusted R-squared:  0.5313
F-statistic: 189.7 on 6 and 993 DF,  p-value: < 2.2e-16
```

The stepwise function removed these predictors: Mature and Language. We are left with all significant predictors with a p-value less than 0.05. This is our model formula:

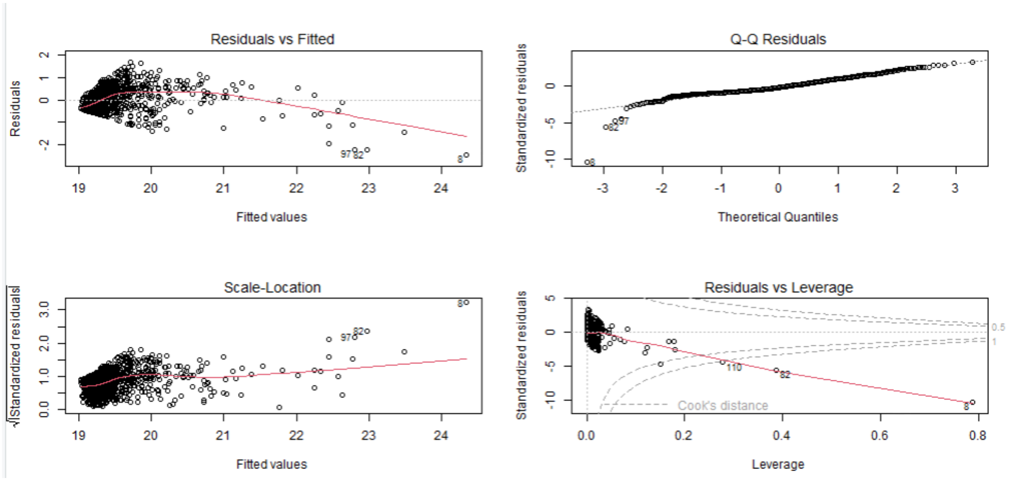
$$log(Watch.time.Minutes) = \beta0 + \beta1(Stream.time.minutes) + \beta2(Peak.viewers) + \beta3(Average.viewers) + \beta4(Followers) + \beta5(Followers.gained) + \beta6(Views.gained)$$

Because our F-statistic p-value = 2.2 e-16 and is less than 0.05, we can confidently reject the null hypothesis and say that our predictors can be used to predict our watch time. Additionally, our adjusted R^2 is 0.5313. This means that 53.13% of the watch time variance is explained by our model equation. The fit is not the best and can be improved.

```
AIC(lm.2)
BIC(lm.2)

# 1492.068
# 1531.33
```

Both AIC and BIC are pretty high, meaning that our models are not well-fitted. We can now look at our residual plots and individual scatterplots.

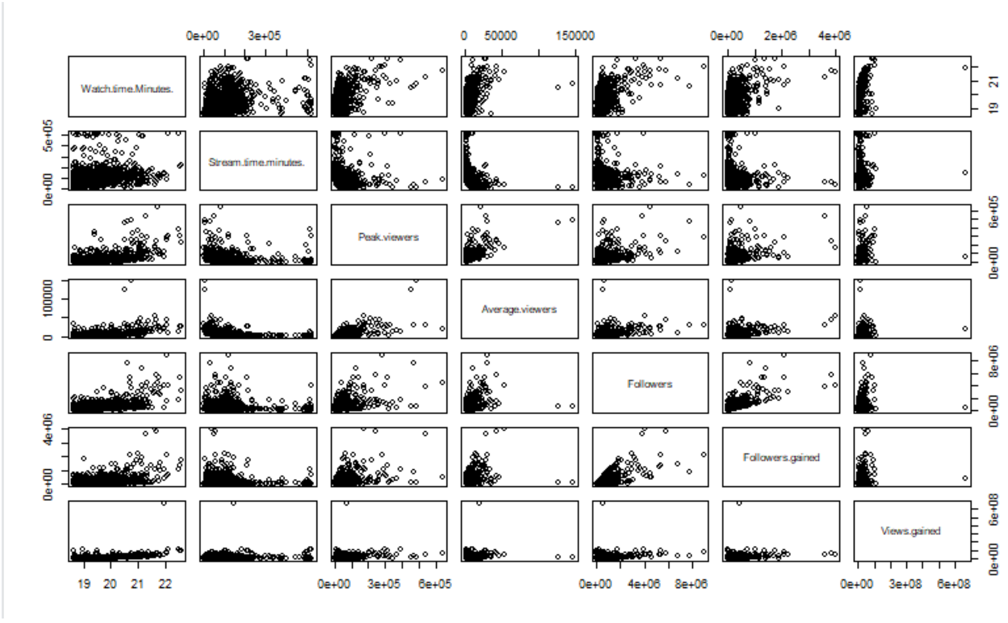
```
par(mfrow=c(2,2))
plot(lm.2)
library(car)
vif(lm.2)
```



```
Stream.time.minutes.          Peak.viewers      Average.viewers            Followers     Followers.gained          Views.gained
            1.101693              2.203173             2.025742             2.302419             2.151671              1.145817
```

Our model has a residual standard error of 0.5079. This high error can be seen in this residual plot. We can see that our model is not normal, and has unequal variance and extreme values. Additionally, we have no problems with collinearity.

```
attributes_to_exclude <- c("Channel", "Partnered", "Mature", "Language")
attributes_to_include <- setdiff(names(twitch), attributes_to_exclude)
twitch.pairs <- twitch[, attributes_to_include]
twitch.pairs$Watch.time.Minutes. = log(twitch.pairs$Watch.time.Minutes.)
pairs(twitch.pairs)
```



We can proceed to randomly divide our data into 80:20 (training : testing) datasets. Afterward, we will compute a cross-validate k-fold to calculate our mean squared error (MSE).

```
twitch.new = twitch[, attributes_to_include]
twitch.new$Watch.time.Minutes. = log(twitch.new$Watch.time.Minutes.)
MSE = rep(0,10)
for (i in 1:10) {
  set.seed(i)
  train = sample(1:nrow(twitch.new), 0.8*nrow(twitch.new))
  test = twitch.new[-train,]
  twitch.lm = lm(twitch.new$Watch.time.Minutes ~., data=twitch.new,
                subset=train)
  yhat = predict(twitch.lm, newdata=test)
  MSE[i] = mean((yhat - twitch.new$Watch.time.Minutes) * (yhat -
twitch.new$Watch.time.Minutes))
}
MSE
mean(MSE)
```

```
> MSE
 [1] 0.7794956 0.8137910 0.7166798 0.8237797 0.8139236 0.8011114 0.7937541 0.9408114 0.8237353 0.8331077
> mean(MSE)
[1] 0.814019
> 
```

Our average test MSE is 0.814019 throughout 10 sets of training and testing the data. This means that our predictions are on average 0.814 log(minutes) in difference to the actual observation. However, we want this MSE to be closer to 0, and our model is decently accurate in predicting the log(minutes) that are watched by a streamer.

To summarize, followers gained, peak viewers, average viewers, followers, views gained, and stream time in minutes are all predictors that are significant in predicting the watch time for streamers. However, our computed values R^2, residual plots, MSE, AIC, and BIC do not indicate that this is an appropriate model to infer that our predictors are significant in predicting the response variable. We should take into other considerations, such as a polynomial regression.

Random Forest Model (Chichen Chao)

We are trying to answer the question: How do various channel metrics such as watch time, stream time, peak viewers, average viewers, total followers, and views gained contribute to the number of new followers gained on a Twitch channel?

For our preliminary model with all predictors, we use the standard approach for mtry when doing regression tasks: square root of the number of columns - 1.

```
        mtry_value <- round(sqrt(ncol(train_data)))
```

Then we exclude the NaN entries in our dataset by doing:

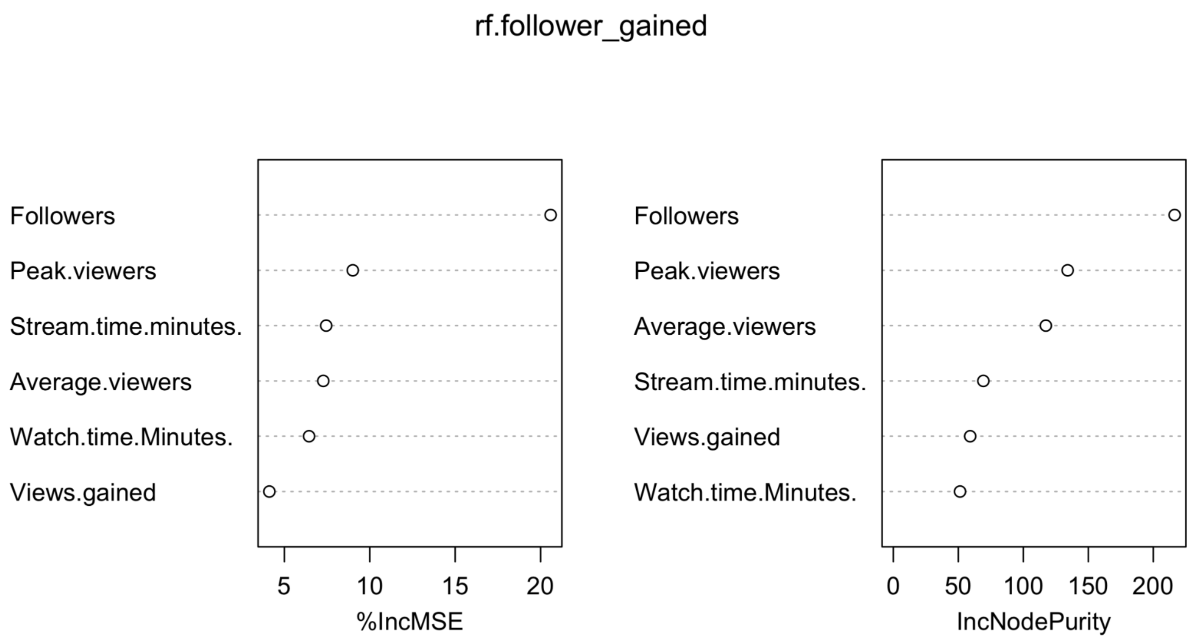
```
        train_data = na.omit(train_data)
```

For data preprocessing, we log and normalize the response variable:

```
twitch_data$Followers.gained <- log(twitch_data$Followers.gained + 1)
```

To find the most significant predictors and subsequently refine our model, we use varImpPlot. VarImpPlot() allows for visualization of the decision-making process of a random forest. From the %IncMSE plot, the three variables that stand out are Followers, Peak.viewers, Average.viewers.

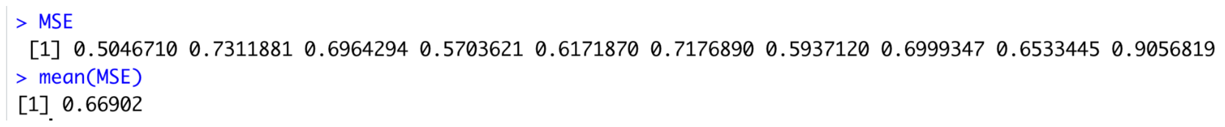
Thus, followers, Peak.viewers, Average.viewers appear to have statistically significant relationships with the new_followers_gained response variable. While Views.gained and Watch.time.Minutes appear to have little impact on MSE if removed, so we suspect they are unimportant enough to be excluded.



We can proceed to randomly divide our data into 80:20 (training : testing) datasets. Afterward, we will compute a cross-validate 10-fold to calculate our mean squared error (MSE).

```
MSE=rep(0,10)
for (i in 1:10) {
 # Set seed for reproducibility
 set.seed(i)

 # Sample 80% of the data as training set
 train_indices <- sample(1:nrow(train_data), 0.8 * nrow(train_data))
 test_data <- train_data[-train_indices, ]

 # Train the Random Forest model
 followers.rf <- randomForest(Followers.gained ~.,
                       data = train_data,
                       subset=train_indices,

                       mtry = (ncol(twitch_data)-1)/3,
                       importance=TRUE)

 # Make predictions on the test set
 yhat.rf <- predict(followers.rf, newdata = test_data)
 # Extract actual scores of the test set
 test.score <- test_data$Followers.gained

 # Compute and store MSE for this iteration
 MSE[i] <- mean((yhat.rf - test.score)^2)
}
```

Results:



```
> MSE
 [1] 0.5046710 0.7311881 0.6964294 0.5703621 0.6171870 0.7176890 0.5937120 0.6999347 0.6533445 0.9056819
> mean(MSE)
[1] 0.66902
```

Our average test MSE is 0.66902 throughout 10 sets of training and testing the data. This means that our predictions are, on average, 0.669 log(new_follower_gained) different from the actual observation. However, we want this MSE to be closer to 0, and our model is more accurate in predicting the new_viewers_gained than our linear regression model.

In summary, according to the importance plot and mse results, peak viewers, followers, average_viewers and stream_time in minutes are all predictors that are significant in predicting the 'new followers gained' for streamers.

Conclusion (Chichen Chao & Jason Suerte)

For multiple linear regression, we used the total watch time to predict the popularity of a streamer. This is important because we can use this to infer the ranking of a streamer. The model indicates that our model decently predicts the watch time of a streamer. We concluded with a high test error, unfit data, and inaccuracy. In general, linear regression did not show to be the appropriate model for predicting the popularity (watch time) of a streamer. To improve the model, we can try a polynomial regression in the future.

For random forest, we were trying to predict the total number of new followers gained, which is a key metric of a steamer's popularity. The importance plot of the random forest model identified followers, peak viewers, and average viewers as significant predictors.

The random forest model consistently yields lower test error rates than the multiple linear regression, indicating better accuracy and stability in predicting a streamer's popularity (new followers gained). We conclude that the random forest model is the better model to discover the factors that affect streamer popularity. However, the MSE is still relatively high and can be improved. We should hyper-tune the parameters in the future to improve the MSE and find the better 'mtry' and 'ntree' to use.

To improve our models, we can consider other factors that affect a streamer's popularity, such as a streamer's gender, background, time zone, stream genre, etc. We can also, expand our dataset outside of the top 1000 streamers.

Bibliography

Mishra, A. (2020, August 24). *Top streamers on twitch*. Kaggle. https://www.kaggle.com/datasets/aayushmishra1512/twitchdata/data

Suerte, J., Chao, C., & Khan, A. (2023, December 1). *Jpsuertee/twitch-streamer-popularity*. GitHub. https://github.com/jpsuertee/Twitch-Streamer-Popularity