Predicting the Growth Rate of YouTube Videos

Team: MLA - Ashlyn Jew, Haozhen Ni, Huimin Zhang

1. Introduction

YouTube is a large global video-sharing platform where users can interact with videos in various ways. Content creators can create and upload videos that users can watch, comment on, and give ratings to. Views are one of the most important metrics of a video, as they measure the amount of user engagement and help determine how much revenue the video will bring to the content creator. Knowing how fast a video's number of views grow in the first few hours after it has been uploaded is a useful indicator for content creators. In this project, we aim to predict the percentage change in views on a video between the second and sixth hour since its publishing. This will help the content creator determine the eventual performance of their video and the overall success of their channel.

2. Methodology

2.1 Data Preprocessing

There are 260 variables and 7242 observations in our data set, including 1 response variable "growth_2_6".

First of all, we believe that the date and time a video is published influences the number of early views. For example, if a video is published at midnight (in the timezone of the channel's largest audience), then there would not be many views between the second and sixth hour because most people are sleeping during that period. Therefore, we split the "PublishedDate" variable into separate month, day, hour, and minute variables and deleted the "PublishedDate" variable. We applied the same process on the test data.

Secondly, we deleted 12 columns that contained only 0 values, since they would have no effect on predicting the response variable.

Lastly, we excluded quantitative variables that have an absolute correlation of 0.9 or above. Multicollinearity can affect the variable importances, so we removed variables that were highly correlated.

2.2 Statistical Model

Our best-performing model on the public leaderboard was constructed using a bagged regression tree with the top 25 most important predictors.

We elected to use tree-based methods, such as bagging and random forests, because they can handle datasets with high dimensions and have excellent predictive power. Our dataset has 179 predictors after preprocessing, so using tree-based methods was a good choice for our high dimensional dataset. We first compared bagged models against random forest models and consistently found that bagged models performed better. For instance, using the same cleaned dataset and same predictors, our initial random forest model had a root mean squared error (RMSE) of 1.62805 on the public leaderboard while our initial bagged model had an RMSE of 1.49846. Thus, we chose to use bagging and improve our bagged model.

To improve its performance, we tried to see if using less predictors would improve our model. We used the rfcv() function in the randomForest library to perform 5-fold cross-validation for feature selection. As shown in Figure 1 below, the 5-fold cross-validation results indicate that a model between 22 and 45 predictors would have the lowest error rate. Thus, we chose to use the top 25 most important predictors.

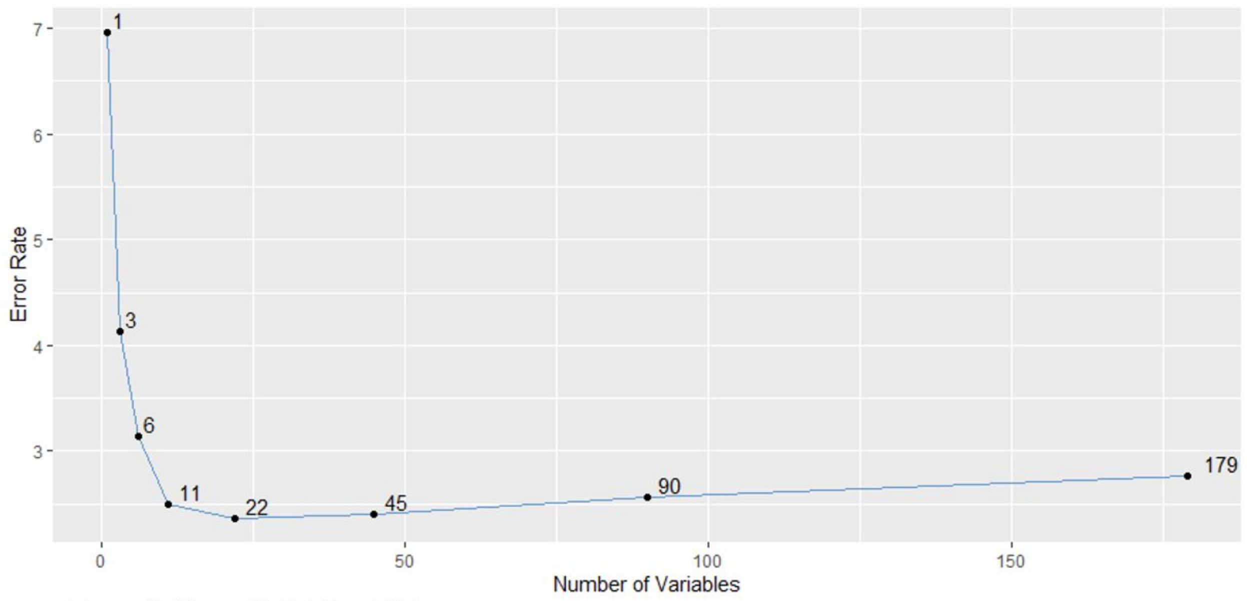

Figure 1: Cross-Validation Plot

To find the top 25 most important predictors, we built a bagged regression model using all the predictors in our cleaned dataset. Then, we chose the top 25 most important variables based upon the mean decrease of accuracy in predictions on the out of bag samples when a variable is excluded from the model. The measure of importance for the 25 variables are shown in Figure 2 below.

We then used these 25 variables to build another bagged regression model. We observed that the RMSE on the public leaderboard decreased from 1.46013 to 1.38825 after we reduced the number of variables and selected the most important predictors.

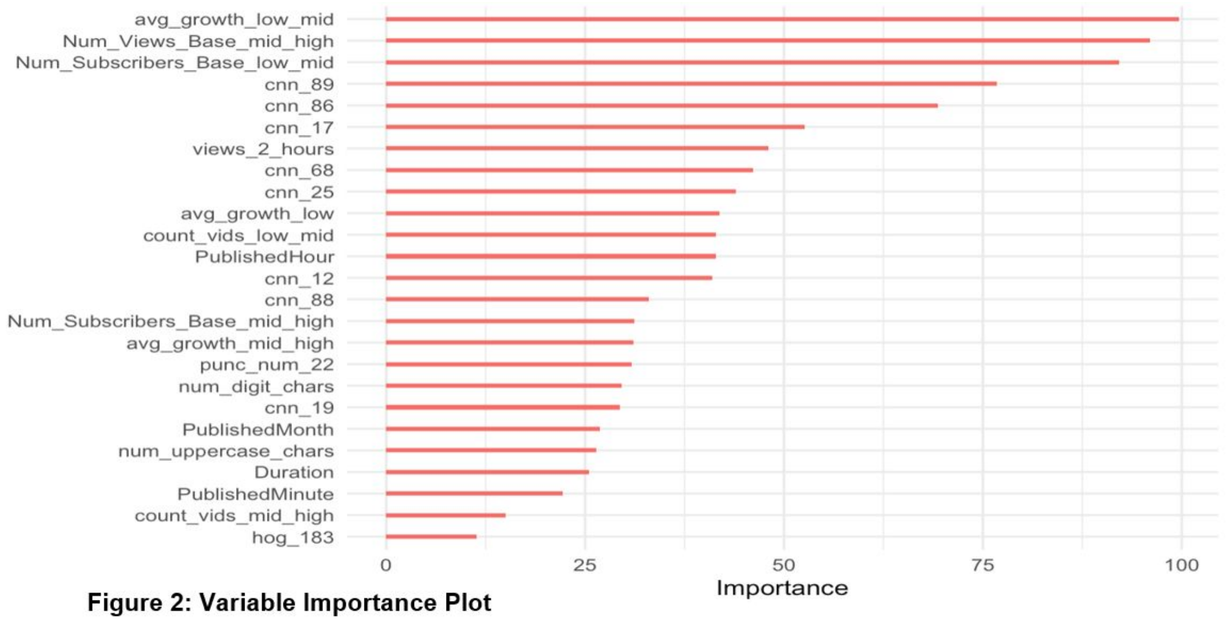

Figure 2: Variable Importance Plot

3. Results

The best evaluation metric value with this model in the Kaggle public leaderboard is a root mean squared error of 1.38825.

4. Conclusion

We believe that this model works well because we have cleaned and transformed the dataset well enough for the bagged regression model to output accurate predictions. Since bagging and random forests create powerful prediction models and are a good choice for high dimensional data, we believe that using a bagged model was the right choice. In addition, transforming the "PublishedDate" variable also helped improve the model significantly, as we observed that the published month, hour, and minute are all among the top 25 most important variables. Reducing the number of predictors in our model to the most important variables may have also helped reduce noise and increase our prediction accuracy. Some ways that we can improve our model is to further reduce multicollinearity so that we have a more accurate importance measure and to investigate the optimal number of predictors to include in our model.

The best model in private leaderboard has an RMSE of 1.38810 and was constructed using a bagged decision tree with the top 40 most important variables. The best model in the public leaderboard has an RMSE of 1.38825 and was constructed using a bagged decision tree with the top 25 most important variables. We believe that the model with 40 variables performed better than the model with 25 variables because including more variables captured more variability. In the 25-variable model, we may have excluded some variables that were important to the overall data. As we increase the number of predictors, the flexibility of the model increases, and thus the bias may initially decrease faster than the variance increases, resulting in more accurate predictions.

Statement of Contributions

Ashlyn Jew worked on building the model and writing the report:

I transformed the data by splitting the "PublishedDate" variable into month, day, hour, and minute variables. Then I ran 5-fold cross-validation on the cleaned dataset. I performed bagging on the cleaned dataset and found the most important variables. I then performed bagging again using the most important variables. I performed bagging on a subset of the top 50 and 30 most important variables. I also wrote and edited most of the report.

Haozhen Ni worked on coding and data cleaning:

I first found all the columns whose observations were zero and excluded them, and then found the categorical columns to record their index of columns. Then I calculated the correlation between all the variables except the response variable "growth_2_6", and removed the correlation greater than 0.5, 0.7, and 0.9 for the other 80% of variables. Furthermore, I also converted the "PublishedDate" variable to the UTC date and time built into the R system for calculation. In addition, I drew the boxplot for all the categorical variables that I recorded one by one with the "growth_2_6".

Huimin Zhang worked on data cleaning and building models:

I tried to find high correlation variables inside each group feature, such as Histogram of Oriented Gradients (HOG), Convolutional Neural Network (CNN), Doc2Vec word and punctuation. I also tried to find high correlation variables between all the numeric variables in the dataset. I found out that finding high correlation variables between all the numeric variables in the dataset is more precise than in each group. I tried to shrink the regression coefficients by using ridge regression, lasso, and elastic net regression. I also tried bagging, random forests, and boosting to built models. It turned out that tree-based methods worked better than shrinkage methods in this project.

Appendix

```r
## Load data

youtube <- read.csv("training.csv")

dim(youtube)

test <- read.csv("test.csv")


## Preprocessing

library(tidyverse)

library(lubridate)

library(caret)

# Split PublishedDate into Month/Day/Hour/Minute variables

youtube$PublishedDate <- mdy_hm(youtube$PublishedDate)

PublishedMonth <- month(youtube$PublishedDate)

PublishedDay <- day(youtube$PublishedDate)

PublishedHour <- hour(youtube$PublishedDate)

PublishedMinute <- minute(youtube$PublishedDate)

youtube <- youtube[, -2]

youtube <- cbind("id" = youtube[, 1], PublishedMonth, PublishedDay,
PublishedHour, PublishedMinute, youtube[, 2:ncol(youtube)])

# Split PublishedDate into Month/Day/Hour/Minute variables for test data

test$PublishedDate <- mdy_hm(test$PublishedDate)

PublishedMonth <- month(test$PublishedDate)

PublishedDay <- day(test$PublishedDate)

PublishedHour <- hour(test$PublishedDate)

PublishedMinute <- minute(test$PublishedDate)

test <- test[, -2]

test <- cbind("id" = test[, 1], PublishedMonth, PublishedDay, PublishedHour,
PublishedMinute, test[, 2:ncol(test)])
```

```r
# Remove columns with only 0's

zero_cols <- which(colSums(youtube) == 0)

youtube_no0 <- youtube[ , -zero_cols]

# Exclude indicator columns, id and growth_2_6 columns

numeric_youtube <- youtube_no0[, -c(1, 239:251)]

# Remove variables that have an absolute correlation of 0.9 or above

high_cor <- findCorrelation(cor(numeric_youtube), cutoff = 0.9)

youtube_no0_cor <- youtube_no0[, -high_cor]


## Random forest 5-fold cross-validation

# rfcv(youtube_no0_cor[, 2:180], youtube_no0_cor[, 181])


## Bagging on whole cleaned training dataset to get important variables

library(randomForest)

# Bagging on cleaned dataset to get the top 25 most important variables

bag_model <- randomForest(growth_2_6 ~ ., data = youtube_no0_cor[, -1], mtry =
ncol(youtube_no0_cor[, -1]) - 1, importance = TRUE)

bag_importance <- importance(bag_model)

bag_importance[order(bag_importance[,1], decreasing = TRUE), ][1:25,]


## Bagging on top 25 most important variables

youtube_new <- cbind(youtube_no0_cor[,
row.names(bag_importance[order(bag_importance[,1], decreasing = TRUE),
][1:25,])], "growth_2_6" = youtube$growth_2_6)

bag_model_new <- randomForest(growth_2_6 ~ ., data = youtube_new, mtry =
ncol(youtube_new) - 1, importance = TRUE)

bag_pred <- predict(bag_model_new, test)

bag_output <- data.frame("id" = test[,1], "growth_2_6" = bag_pred)
```