DS 220

Prof. Dongwon Lee

April 23, 2019

Group 9

Isha Baxi

Zachary Vrabel, Jaehoon Ha, Zhejing Shi, Sophie Massolas, Wilfred Fontanez

## Main idea

For our application we have made a program that will give users an estimate as to whether his/her video will trend on Youtube. According to fortune.com Youtube will soon become the main format used for entertainment in the US replacing standard cable televisions. Keeping this in mind we felt that this type of application would be very useful for new and upcoming content creators since it will allow them to receive an estimate of there potential success based on trending videos from previous years. Youtube is not a new platform and has been around for many years now, allowing people to upload and view videos. There are a variety of reasons as to why one would want to create and upload these videos be it for fame, revenue, or simply for fun. Thus an application that helps these users estimate there chance at success would be in great demand because for the most part people are going to want as many views as they can and it is no surprise that some videos are more successful than others. Which is why our application will allow users to obtain an idea on how bad/good there video will do and allow them to make changes that will give them a higher chance of success.

## Motivation/Justification

Youtube has become a widely used video streaming website. One can find any kind of video from a diy tutorial to cat videos on this site. Youtube has gained so much popularity that uploaders are even able to make money from simply uploading videos. However, the video must get a certain amount of views before it can become profitable. Understanding what viewers want to see now a days can be difficult. The trends in such a connected world are changing everyday. Our Viral Video Predictor app helps to alleviate this stress. The app uses a dataset that has information on viral videos in every country to predict how likely it is that the user;s video will go viral.

The app will ask the user a series of questions such as 'how many likes has the video gotten?'. After answering these questions, the app will compare the user's inputs to the information in the dataset. The code behind the app will run statistical tests to output a graph and percentage. The graph will show where your video stands amongst the videos from the dataset. The percentage will represent how likely it is that the user's will go viral based on the information they provided.

This app can be very helpful for both aspiring and established youtubers. Established youtubers are always struggling to keep up with the interests of their viewers. Many of these youtubers have even publicly complained that they never know how their video will do until after they have already spent hours creating the content and have uploaded it. By giving them a tool where they can get a prediction on the success of the video , they can save their hard work and put it towards a video that has a higher chance of going viral. For aspiring youtubers this app can help them break into the industry. They can learn what makes a popular video and gain fame faster. Rather than going through trial and error of uploading videos and trying to figure out their target audience, this app can help them to pick an audience and upload videos of their viewers interests.

## Design of the App

The design of the app is based on the shiny extension for R Studio. Shiny allows users to build a web app directly in the R Studio interface. It allows the user to have stand alone webpages. The app itself relies on the user interface and the server modules built into Shiny. The user interface in this particular app displays a side panel as well as a main panel. The side panel has multiple selection options including, number of likes and dislikes, number of comments. The app also takes into account the fact that category id is a categorical variable and assigns each categorical variable a numeric value. These variables are all contained in the user interface of the app. The server function is the function that does all of the output. The output of the app includes a regression graph as well as a textual display of the expected view count.

## Dataset Used and Why

Our group mainly focused on building an app that estimates the trends of Youtube videos. In order to estimate the video trends, we needed a dataset that contains Youtube information such as video title, number of likes, number of comments, etc. We looked over Data.gov, Kaggle, Datausa, and Amazon datasets to search for the Youtube datasets. Finally, our group found the CSV datasets files on Kaggle that contains information about Youtube videos.

Kaggle is one of the most popular dataset websites where we can easily access and download millions of data onto our computer. Therefore, we could easily find and download the datasets about Youtube videos in CSV files. The Kaggle dataset of Youtube videos contained information about not only the US Youtube videos, but also many other countries' videos such as

France, Canada, and Germany. Using this data our app can estimate not only the US Youtube trends but also the trends of each countries. With Kaggle's CSV files, we used Python programming to insert the datasets into Redis, which is what we used for NoSQL server. Our dataset is comprised of thousands of videos with multiple attributes such as video ID, trending date, title, views, likes, dislikes, comments, etc. In this case, Redis, a Key-value database, Redis is the best database we can think of.

## NoSQL DB Used and Why

Our data is stored in a Redis database. Redis allows us to store hundreds of thousands of tuples in a relatively short amount of time. Furthermore, the key and value pair structure is perfect for our dataset, which is comprised of hundreds of thousands of unique videos each with their own attributes such as trending date, title, channel category, publish date, Tags, Views, Likes, Dislikes, number of comments, etc. The hash data type supported by redis is perfect for this type of data. A hash is comprised of a primary key which references the entire hash, and subkeys and values within the hash. Video ID is the hash key, and the video's attributes are subkeys. For example, "likes" would be a subkey and "13750" would be its corresponding value. This allows us to easily query attributes from specific videos for our regression model. The data was stored into the redis database using python. The program reads .csv files from the dataset, converts each row in the .csv file into a dictionary, and stores that dictionary as a hash with video id as the primary key.

## Implementation of App

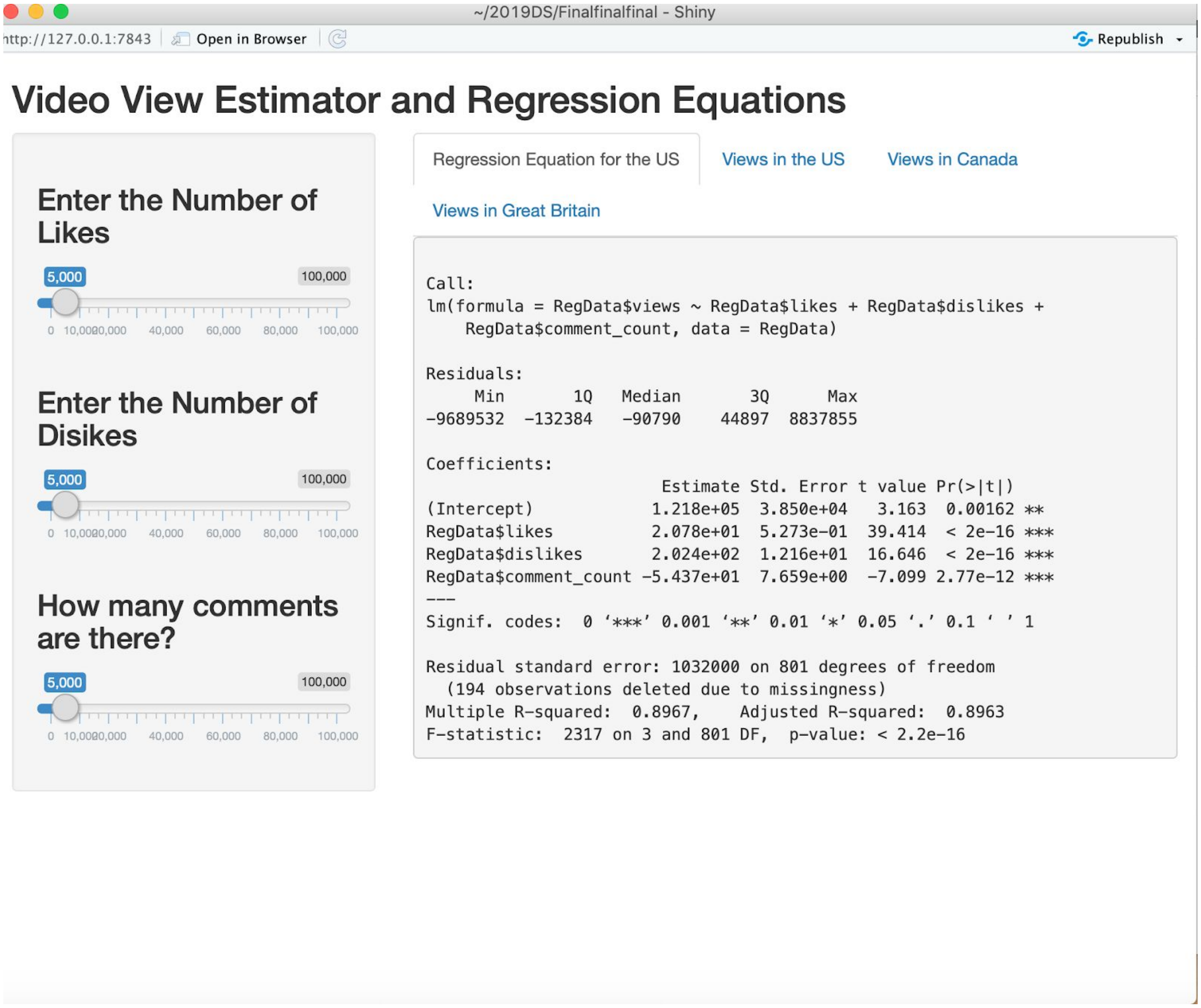
The app was implemented using the Shiny UI and Server. Shiny allows your computer to serve as a host for the app or to be hosted on the shiny server as long as there is less than 5 GB of data. The app uses reactive tabs based on the regression equations to display the expected view count. In the UI, there are subsections for the tabs as well as the slider inputs. In the server, values are stored and the outputs are displayed.

Github Link (includes the code to run app on everyone's computer) :

https://github.com/wil-fontanez/Proj2-DS220.git

https://sophiemassolas.shinyapps.io/Finalfinalfinal/

**Screenshot of app**

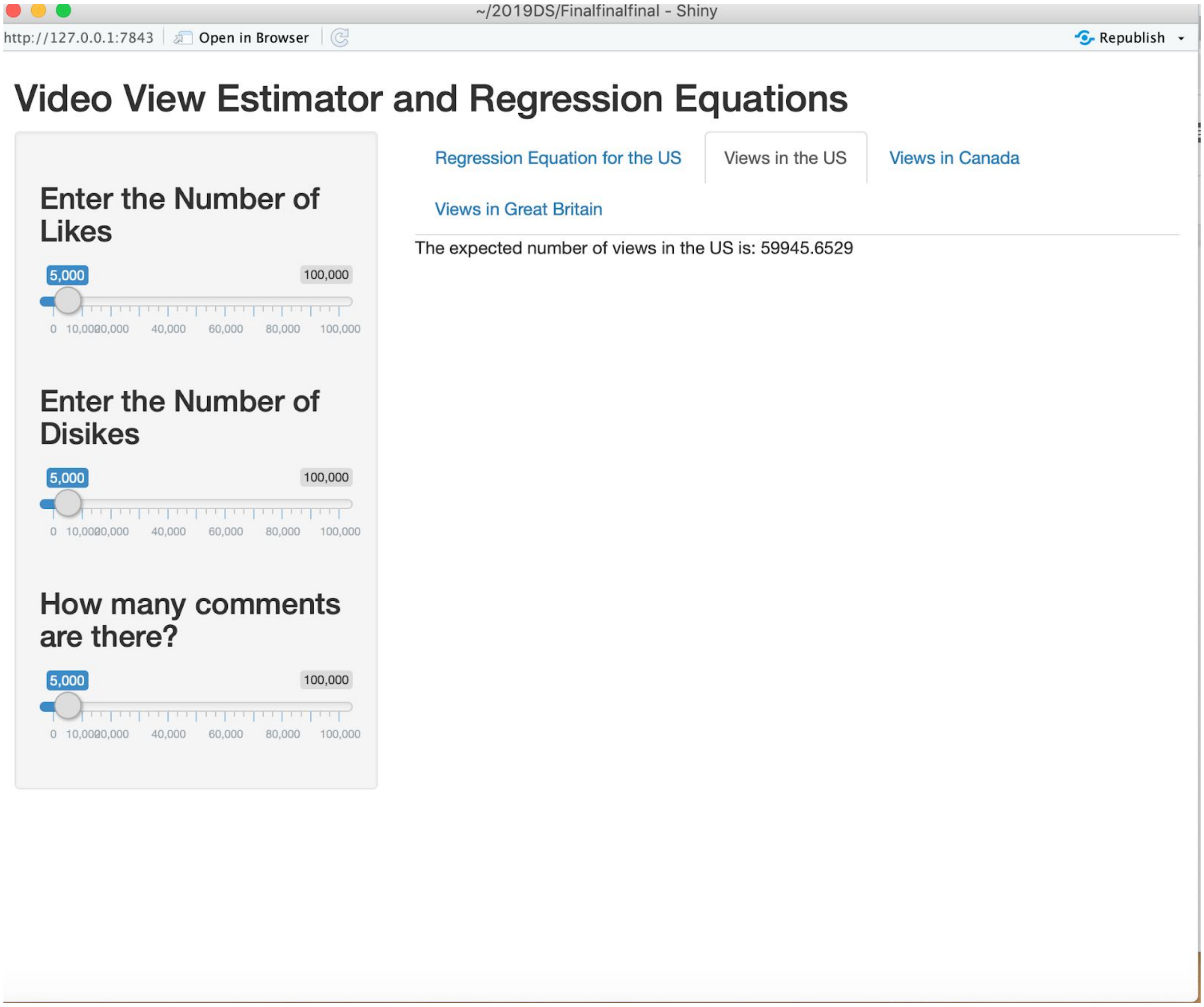


```
Call:
lm(formula = RegData$views ~ RegData$likes + RegData$dislikes +
    RegData$comment_count, data = RegData)

Residuals:
     Min       1Q   Median       3Q      Max
-9689532  -132384   -90790    44897  8837855

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)            1.218e+05  3.850e+04   3.163  0.00162 **
RegData$likes          2.078e+01  5.273e-01  39.414  < 2e-16 ***
RegData$dislikes       2.024e+02  1.216e+01  16.646  < 2e-16 ***
RegData$comment_count -5.437e+01  7.659e+00  -7.099 2.77e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1032000 on 801 degrees of freedom
  (194 observations deleted due to missingness)
Multiple R-squared:  0.8967,    Adjusted R-squared:  0.8963
F-statistic:  2317 on 3 and 801 DF,  p-value: < 2.2e-16
```



Video View Estimator and Regression Equations

Enter the Number of Likes: 5,000 (max 100,000)

Enter the Number of Disikes: 5,000 (max 100,000)

How many comments are there?: 5,000 (max 100,000)

The expected number of views in the US is: 59945.6529

### Who did what

As a group, in order to build this app and make it qualified for the presentation, we shared the work and cooperated with each other. Each one of us did a particular job.

- Sophie Massolas : Sophie plays one of the most important role in our team. She is responsible for making the application available on a website. She connected our app to the database and used the regression to predict likelihood of a trending video.
- Isha Baxi: Wrote the motivation/justification part which explains why our group felt this app should be made. She also helped to make sure the team was staying on task and on

track to finishing the project by checking up on each individual’s part frequently. She also assembled everyone’s respective parts into a final paper.

- Zachary Vrabel: He inserted the data from Kaggle into a redis database. Then he wrote a python code so that we could easily implement this database into the main application.
- Wilfred Fontanez: He wrote the Main Idea section of the paper. Wilfred also created the github for the group.
- Jaehoon Ha: He did research on which database would be the best to use and wrote the ‘Database Used and Why’ section.
- Zhejing Shi: Worked with Jaehoon figure out the best dataset we should work on, and justify why we are using it. And he is also responsible for summarizing everyone’s work.