Regression on US YouTube Trending Data

Andrew Gerungan

Linear regression attempts to draw a relationship between a target and its predictor(s); the distinguishing trait of linear regression is that it is heavily biased in that it always assumes that the relationship between the target and predictor(s) is linear. It's strength lie in its simplicity and low variance. It works the best when the data presented follows a linear pattern (linear regression for linear pattern). The major weakness of linear regression is that it has a high bias since it assumes that the data has a linear shape to it, meaning that it does not fit any patterns that don't have a natural linearity to it.

Setting everything up

The code below loads in the dataset "US YouTube Trending Data" and then removes any nulls in the dataset before checking the structure

```
df <- read.csv("US_youtube_trending_data.csv")
```

```
 警告:  embedded nul(s) found in input
```

```
df <- df[complete.cases(df),]
str(df)
```

```
'data.frame':	155790 obs. of  16 variables:
 $ video_id         : chr  "3C66w5Z0ixs" "M9Pmf9AB4Mo" "J78aPJ3VyNs" "kXLn3HkpjaA" ...
 $ title            : chr  "I ASKED HER TO BE MY GIRLFRIEND..." "Apex Legends | Stories from the Outlands – “Th
e Endorsement”" "I left youtube for a month and THIS is what happened." "XXL 2020 Freshman Class Revealed - Off
icial Announcement" ...
 $ publishedAt      : chr  "2020-08-11T19:20:14Z" "2020-08-11T17:00:10Z" "2020-08-11T16:34:06Z" "2020-08-11T16:
38:55Z" ...
 $ channelId        : chr  "UCvtRTOMP2TqYqu51xNrqAzg" "UC0ZV6M2THA81QT9hrVWJG3A" "UCYzPXprvl5Y-Sf0g4vX-m6g" "UC
bg_UMjlHJg_19SZckaKajg" ...
 $ channelTitle     : chr  "Brawadis" "Apex Legends" "jacksepticeye" "XXL" ...
 $ categoryId       : int  22 20 24 10 26 24 26 27 24 10 ...
 $ trending_date    : chr  "2020-08-12T00:00:00Z" "2020-08-12T00:00:00Z" "2020-08-12T00:00:00Z" "2020-08-12T00:
00:00Z" ...
 $ tags             : chr  "brawadis|prank|basketball|skits|ghost|funny videos|vlog|vlogging|NBA|browadis|chall
enges|bmw i8|faze rug|faze r"| __truncated__ "Apex Legends|Apex Legends characters|new Apex Legend|Apex Legends
Rampart|Apex Legends Season 6|Apex Legends Bo"| __truncated__ "jacksepticeye|funny|funny meme|memes|jackseptice
ye memes|reddit|subreddit|community|community memes|community s"| __truncated__ "xxl freshman|xxl freshmen|2020
xxl freshman|2020 freshman|2020 freshmen|xxl freshman class|2020 xxl freshman cl"| __truncated__ ...
 $ view_count       : int  1514614 2381688 2038853 496771 1123889 949491 470446 1050143 1402687 741028 ...
 $ likes            : int  156908 146739 353787 23251 45802 77487 47990 89190 95694 113983 ...
 $ dislikes         : int  5855 2794 2628 1856 964 746 440 854 2158 4373 ...
 $ comment_count    : int  35313 16549 40221 7647 2196 7506 4558 6455 6613 5618 ...
 $ thumbnail_link   : chr  "https://i.ytimg.com/vi/3C66w5Z0ixs/default.jpg" "https://i.ytimg.com/vi/M9Pmf9AB4M
o/default.jpg" "https://i.ytimg.com/vi/J78aPJ3VyNs/default.jpg" "https://i.ytimg.com/vi/kXLn3HkpjaA/default.jp
g" ...
 $ comments_disabled: chr  "False" "False" "False" "False" ...
 $ ratings_disabled : chr  "False" "False" "False" "False" ...
 $ description      : chr  "SUBSCRIBE to BRAWADIS ▶ http://bit.ly/SubscribeToBrawadis\n\nFOLLOW ME ON SOCIAL\n
▶ Twitter: https://twitter.co"| __truncated__ "While running her own modding shop, Ramya Parekh (aka Rampart)
built her rep in underground gauntlet circuits. "| __truncated__ "I left youtube for a month and this is what h
appenedMY COFFEE COMPANY: https://twitter.com/TOTMCoffeeJoin the s"| __truncated__ "Subscribe to XXL → http://b
it.ly/subscribe-xxl \nXXL presents the 2020 XXL Freshman Class featuring NLE Choppa,"| __truncated__ ...
```

The code below separates the data into train and test (80/20)

```
set.seed(1234)
i <- sample(1:nrow(df), .8*nrow(df), replace=FALSE)
train <- df[i,]
test <- df[-i,]
```

Data Exploration

The next block of code uses the training data and checks the summary of the training data, then checks the view_count column's summary, head & tail, mean, and median

```
summary(train)
```

```
   video_id            title            publishedAt       
 Length:124632      Length:124632      Length:124632     
 Class :character   Class :character   Class :character  
 Mode  :character   Mode  :character   Mode  :character  



  channelId         channelTitle         categoryId   
 Length:124632      Length:124632      Min.   : 1.00  
 Class :character   Class :character   1st Qu.:17.00  
 Mode  :character   Mode  :character   Median :20.00  
                                       Mean   :18.83  
                                       3rd Qu.:24.00  
                                       Max.   :29.00  
 trending_date          tags             view_count     
 Length:124632      Length:124632      Min.   :        0  
 Class :character   Class :character   1st Qu.:   496437  
 Mode  :character   Mode  :character   Median :  1013472  
                                       Mean   :  2602443  
                                       3rd Qu.:  2283361  
                                       Max.   :277791741  
     likes             dislikes      comment_count    
 Min.   :       0   Min.   :     0   Min.   :      0  
 1st Qu.:   19881   1st Qu.:     0   1st Qu.:   1390  
 Median :   46371   Median :   316   Median :   3155  
 Mean   :  137567   Mean   :  1950   Mean   :  11680  
 3rd Qu.:  116683   3rd Qu.:  1213   3rd Qu.:   7590  
 Max.   :16021534   Max.   :848974   Max.   :6738537  
 thumbnail_link     comments_disabled  ratings_disabled  
 Length:124632      Length:124632      Length:124632     
 Class :character   Class :character   Class :character  
 Mode  :character   Mode  :character   Mode  :character  



 description       
 Length:124632     
 Class :character  
 Mode  :character  
```

```
col <- train$view_count

summary(col)
```

```
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
        0    496437   1013472   2602443   2283361 277791741 
```

```
head(col)
```

```
[1]  914669 1951490 1571071  544598 1748851  428460
```

```
tail(col)
```

```
[1]  246432 4167712 1427297  970732 1428757  192474
```

```
mean(col)
```

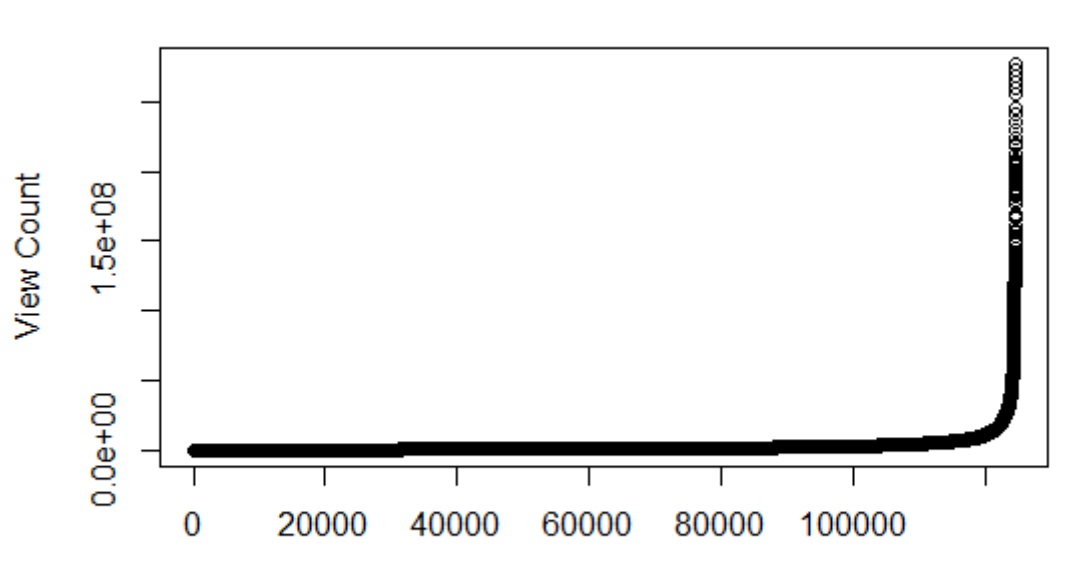
```
[1] 2602443
```

```
median(col)
```

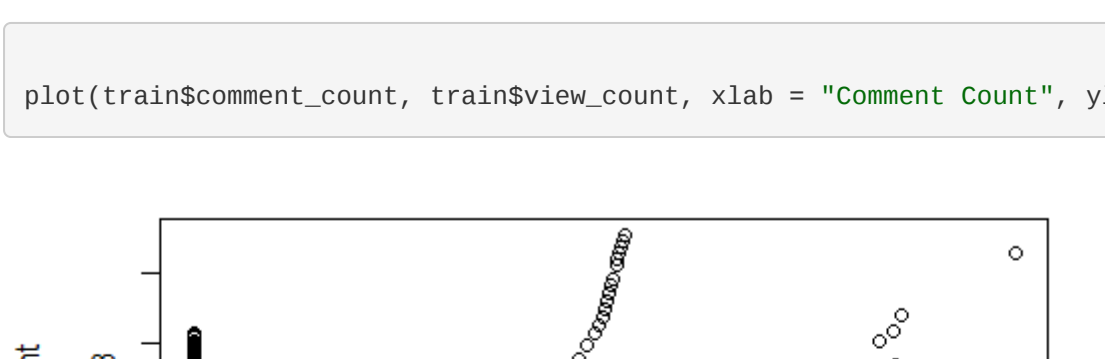
```
[1] 1013472
```

The median and mean are not close, so I want to create some graphs to take a closer look at which value I should consider more. I also want to see the relationship between view_count and like_count and comment_count so I both plot them and check the cor and cov values

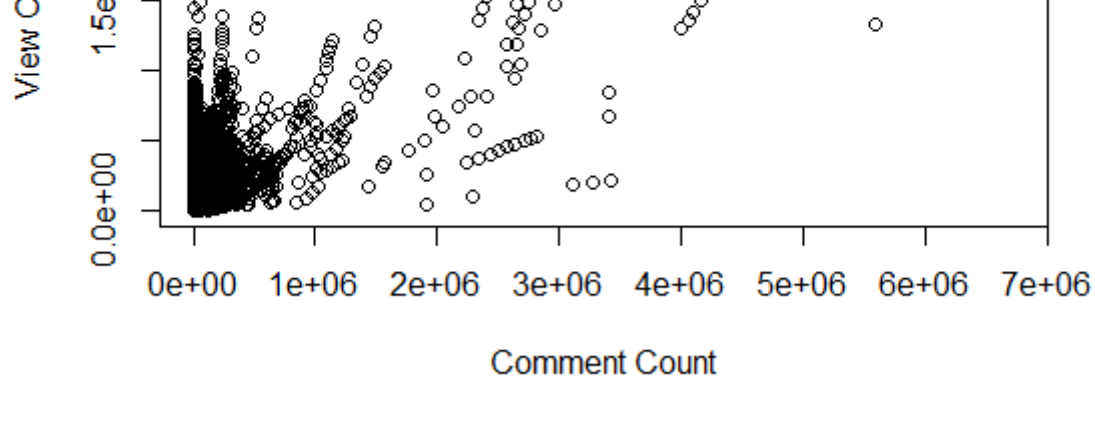
```
plot(sort(col, decreasing=FALSE), ylab = "View Count")
```




```
plot(train$comment_count, train$view_count, xlab = "Comment Count", ylab = "View Count")
```




```
plot(train$likes, train$view_count, xlab = "Likes", ylab = "View Count")
```




```
cor(train$view_count, train$comment_count)
```

```
[1] 0.5966027
```

```
cor(train$view_count, train$likes)
```

```
[1] 0.8558898
```

```
cov(train$view_count, train$comment_count)
```

```
[1] 390635527533
```

```
cov(train$view_count, train$likes)
```

```
[1] 2.624208e+12
```

The plot of increasing values of the view_count shows that median is probably not all that useful since the vast majority of videos have a low view_count. I can also deduce that view_count is more related to likes than comment_count, as evident in both the plot and the correlation value being closer to 1 than comment_count.

Linear regression

Linear regression time - I'll go ahead and use view_count against likes, since that's what I determined would be the closest relationship based on the information presented prior

```
lm1 <- lm(view_count~likes, data=train)
lm1
```

```

Call:
lm(formula = view_count ~ likes, data = train)

Coefficients:
(Intercept)        likes  
  561319.54        14.84  
```

```
summary(lm1)
```

```

Call:
lm(formula = view_count ~ likes, data = train)

Residuals:
      Min        1Q    Median        3Q       Max 
-65321998   -626124   -377413    115475 104147093 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 5.613e+05  1.124e+04   49.95   <2e-16 ***
likes       1.484e+01  2.540e-02  584.26   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3770000 on 124630 degrees of freedom
Multiple R-squared:  0.7325,	Adjusted R-squared:  0.7325 
F-statistic: 3.414e+05 on 1 and 124630 DF,  p-value: < 2.2e-16
```

```
pred <- predict(lm1, newdata=test)
cor1 <- cor(pred, test$view_count)
mse1 <- mean((pred-test$view_count)^2)
print(paste("cor1=",cor1))
```

```
[1] "cor1= 0.84934091058312"
```

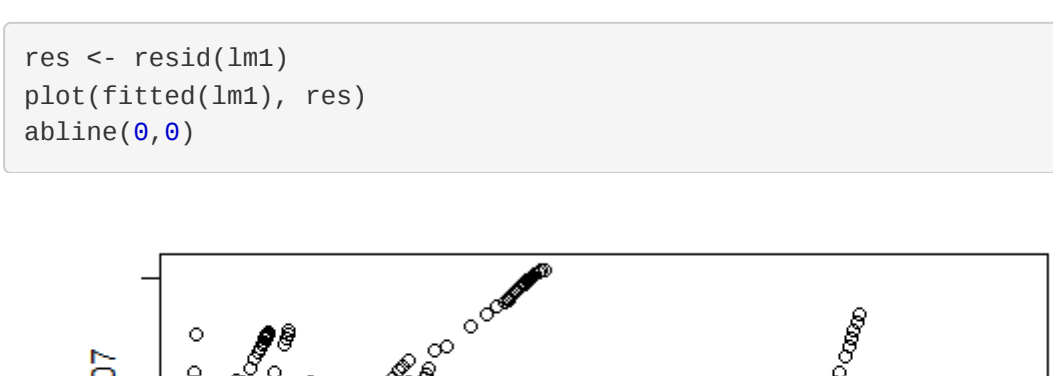
```
print(paste("mse1=",mse1))
```

```
[1] "mse1= 13096261347363.4"
```

The mse is …yikes… but the correlation is pretty solid, so it has that going for it

let's just plot the residuals then

```
res <- resid(lm1)
plot(fitted(lm1), res)
abline(0,0)
```



[stuff about residuals]

now let's try doing multiple linear regression using both likes AND comment_count, also plotting residuals

```
lm2 <- lm(view_count~likes+comment_count, data=train)
lm2
```

```

Call:
lm(formula = view_count ~ likes + comment_count, data = train)

Coefficients:
  (Intercept)          likes  comment_count  
   493850.957         15.896         -6.696  
```

```
summary(lm2)
```

```

Call:
lm(formula = view_count ~ likes + comment_count, data = train)

Residuals:
      Min        1Q    Median        3Q       Max 
-58148119   -592751   -327953    150528  96989580 

Coefficients:
                Estimate Std. Error t value Pr(>|t|)    
(Intercept)    4.939e+05  1.131e+04   43.66   <2e-16 ***
likes          1.590e+01  3.757e-02  423.06   <2e-16 ***
comment_count -6.696e+00  1.759e-01  -38.06   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3749000 on 124629 degrees of freedom
Multiple R-squared:  0.7356,	Adjusted R-squared:  0.7356 
F-statistic: 1.734e+05 on 2 and 124629 DF,  p-value: < 2.2e-16
```

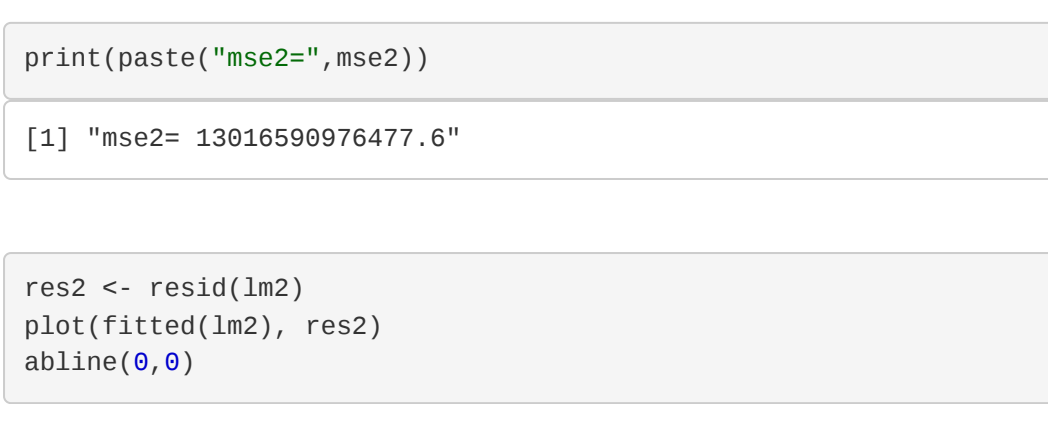
```
pred2 <- predict(lm2, newdata=test)
cor2 <- cor(pred2, test$view_count)
mse2 <- mean((pred2-test$view_count)^2)
print(paste("cor2=",cor2))
```

```
[1] "cor2= 0.850563132303151"
```

```
print(paste("mse2=",mse2))
```

```
[1] "mse2= 13016590976477.6"
```

```
res2 <- resid(lm2)
plot(fitted(lm2), res2)
abline(0,0)
```



…still pretty bad… but the correlation has gone up a little bit

Overall, the mse and residuals are about the same across, so utilizing both likes and comment_count doesn't really change much in the way of predicting the view_count. Let's try polynomial regression then

```
lm3 <- lm(view_count~poly(likes+comment_count, 5, raw = TRUE), data=train)
lm3
```

```

Call:
lm(formula = view_count ~ poly(likes + comment_count, 5, raw = TRUE), 
    data = train)

Coefficients:
                               (Intercept)  
                                 4.847e+05  
poly(likes + comment_count, 5, raw = TRUE)1  
                                 1.458e+01  
poly(likes + comment_count, 5, raw = TRUE)2  
                                -1.218e-08  
poly(likes + comment_count, 5, raw = TRUE)3  
                                -1.134e-13  
poly(likes + comment_count, 5, raw = TRUE)4  
                                 1.120e-20  
poly(likes + comment_count, 5, raw = TRUE)5  
                                -2.953e-28  
```

```
summary(lm3)
```

```

Call:
lm(formula = view_count ~ poly(likes + comment_count, 5, raw = TRUE), 
    data = train)

Residuals:
      Min        1Q    Median        3Q       Max 
-90264217   -595978   -331947    136383 122717596 

Coefficients:
                                              Estimate Std. Error
(Intercept)                                  4.847e+05  1.382e+04
poly(likes + comment_count, 5, raw = TRUE)1  1.458e+01  9.249e-02
poly(likes + comment_count, 5, raw = TRUE)2 -1.218e-08  6.271e-08
poly(likes + comment_count, 5, raw = TRUE)3 -1.134e-13  1.219e-14
poly(likes + comment_count, 5, raw = TRUE)4  1.120e-20  8.528e-22
poly(likes + comment_count, 5, raw = TRUE)5 -2.953e-28  1.926e-29
                                            t value Pr(>|t|)    
(Intercept)                                  35.082   <2e-16 ***
poly(likes + comment_count, 5, raw = TRUE)1 157.653   <2e-16 ***
poly(likes + comment_count, 5, raw = TRUE)2  -0.194    0.846    
poly(likes + comment_count, 5, raw = TRUE)3  -9.303   <2e-16 ***
poly(likes + comment_count, 5, raw = TRUE)4  13.137   <2e-16 ***
poly(likes + comment_count, 5, raw = TRUE)5 -15.329   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3871000 on 124626 degrees of freedom
Multiple R-squared:  0.718,	Adjusted R-squared:  0.718 
F-statistic: 6.347e+04 on 5 and 124626 DF,  p-value: < 2.2e-16
```

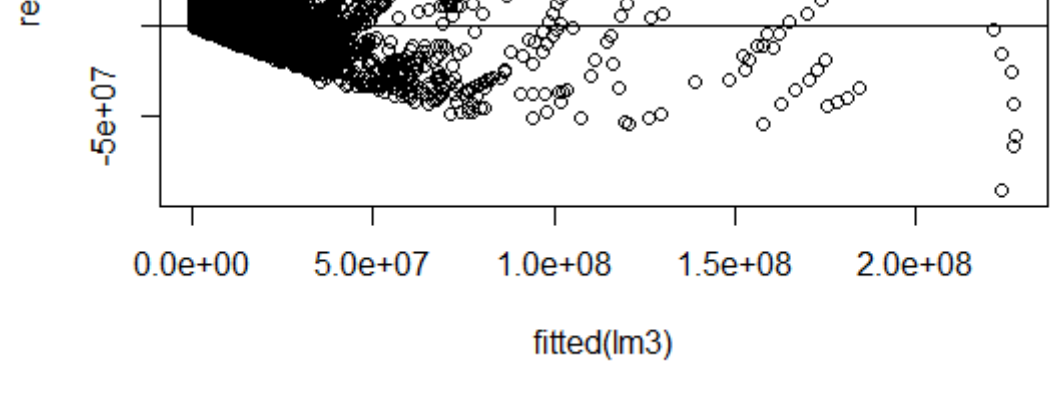
```
pred3 <- predict(lm3, newdata=test)
cor3 <- cor(pred3, test$view_count)
mse3 <- mean((pred3-test$view_count)^2)
print(paste("cor3=",cor3))
```

```
[1] "cor3= 0.844530835280631"
```

```
print(paste("mse3=",mse3))
```

```
[1] "mse3= 13443162717057.3"
```

```
res3 <- resid(lm3)
plot(fitted(lm3), res3)
abline(0,0)
```



Polynomial still isn't quite the best fit - the mse is still pretty bad and the correlation isn't as good as normal linear for just likes

One more thing to try, maybe the category of video matters

```
lm4 <- lm(view_count~categoryId+likes, data=train)
lm4
```

```

Call:
lm(formula = view_count ~ categoryId + likes, data = train)

Coefficients:
(Intercept)   categoryId        likes  
  203402.33     18839.33        14.86  
```

```
summary(lm4)
```

```

Call:
lm(formula = view_count ~ categoryId + likes, data = train)

Residuals:
      Min        1Q    Median        3Q       Max 
-65459170   -653038   -368371    125321 103895849 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 2.034e+05  3.205e+04   6.347  2.2e-10 ***
categoryId  1.884e+04  1.580e+03  11.925  < 2e-16 ***
likes       1.486e+01  2.545e-02 583.818  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3768000 on 124629 degrees of freedom
Multiple R-squared:  0.7329,	Adjusted R-squared:  0.7328 
F-statistic: 1.709e+05 on 2 and 124629 DF,  p-value: < 2.2e-16
```

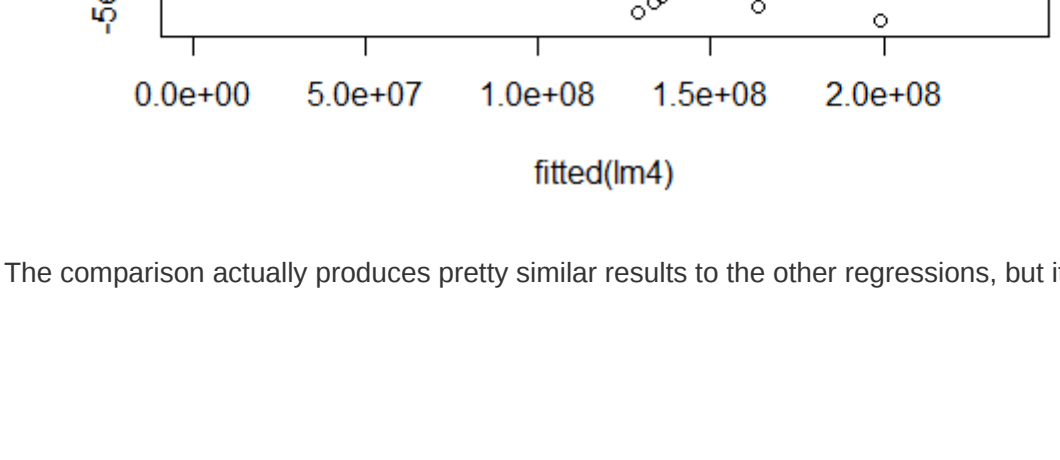
```
pred4 <- predict(lm4, newdata=test)
cor4 <- cor(pred4, test$view_count)
mse4 <- mean((pred4-test$view_count)^2)
print(paste("cor4=",cor4))
```

```
[1] "cor4= 0.8494804557286"
```

```
print(paste("mse4=",mse4))
```

```
[1] "mse4= 13084842416094.7"
```

```
res4 <- resid(lm4)
plot(fitted(lm4), res4)
abline(0,0)
```



The comparison actually produces pretty similar results to the other regressions, but it still doesn't produce a very good mse

Summary

Overall, the results of all 4 models were fairly similar, with some change in mse, cor, and R-squared. Seemingly, the most accurate model was the 2nd model, with a simple linear regression of view_count against both likes and comment_count. This is most likely due to it including the most components that might be able to predict view_count. The polynomial regression didn't impact the results that much since there weren't that many outliers seemingly.