Friends Crowd Favorite Character Analysis

Olivia Rui, Elle Su

ESE326 Professor Wormleighton

1 Introduction

Within the Friends universe, there are some characters that are more well-liked than others. We would like to explore how the screen-time of each of the main characters, Rachel, Ross, Monica, Chandler, Joey, and Phoebe, and the screen time of guest characters affects the episode's IMDB rating.

2 Data

We obtained our dataset from Kaggle. The original dataset is a table that contains every line spoken in each episode, as well as the character who spoke the line. The characters we considered were Monica Geller, Ross Geller, Rachel Green, Joey Tribbiani, Phoebe Buffay, Chandler Bing, and Guest characters. The "Guest" category is defined by all lines not spoken by one of the main characters, which includes lines spoken together and does not include scene directions.

To clean this data, we counted the total number of lines spoken by each character for each episode and created one array per character to store the results. To normalize values across different episodes, we found the proportion of the number of lines spoken by each character to the total number of lines in that episode. The proportion of lines spoken is the final value that gets stored into each character's array. Finally, we created a dataframe to combine all of the characters' arrays. *(See Appendix for code)*

The IMDB ratings of each episode were part of the same dataset from Kaggle, but they were stored in a column of a different table. To clean this data, we simply selected the ratings column from that table and stored it into a one-dimensional array.

Equation (1) is our theoretical model, where

$\hat{y}$ = the predicted rating of an episode
X_1 = number of times Ross speaks
X_2 = number of times Rachel speaks
X_3 = number of times Joey speaks
X_4 = number of times Phoebe speaks

X_5 = number of times Monica speaks
X_6 = number of times Chandler speaks
X_7 = number of times a guest star speaks

$$\hat{y} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \beta_4 X_4 + \beta_5 X_5 + \beta_6 X_6 + \beta_7 X_7 + \epsilon \quad (1)$$

Figure 1 shows how we used R to estimate the equation, and equation (2) is our updated regression model.

```
Call:
lm(formula = Rating ~ (Ross + Rachel + Joey + Phoebe + Monica +
    Chandler + Guest), data = character_data)

Residuals:
     Min       1Q   Median       3Q      Max
-1.36373 -0.26946 -0.03574 0.24094 1.27238

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 7.3439690 0.3352567 21.906 < 2e-16 ***
Ross        1.7501916 0.5703706  3.069  0.00241 **
Rachel      0.3162024 0.5826563  0.543  0.58787
Joey        0.3000993 0.5651909  0.531  0.59596
Phoebe      0.7174477 0.7120022  1.008  0.31469
Monica      1.1691950 0.6084657  1.922  0.05591 .
Chandler    0.1211053 0.5848159  0.207  0.83613
Guest       0.0019334 0.0007218  2.679  0.00793 **
---
Signif. codes: 0   ***   0.001   **   0.01   *   0.05   .   0.1
    1

Residual standard error: 0.3882 on 228 degrees of freedom
Multiple R-squared: 0.07959, Adjusted R-squared: 0.05133
F-statistic: 2.817 on 7 and 228 DF, p-value: 0.007834
```

$$\hat{y} = 7.344 + 1.749X_1 + 0.316X_2 + 0.299X_3 + 0.717X_4 + 1.168X_5 + 0.121X_6 + 0.002X_7 \quad (2)$$

3 Analysis

Our analysis of this data mainly focused on performing a multi-variable linear regression on each of the characters' proportional lines spoken per episode. We

also included all guest appearances as a variable to see if guest appearances would boost the episode's rating.

Surprisingly, our regression indicates that Ross, commonly known as the most hated character in Friends, gave the biggest boost to an episode's rating. On average and holding all else equal, for every one percent increase in the proportion of lines spoken by Ross in an episode, the episode's IMDB rating increases by 1.7. The coefficient estimate for β_1 is significant at the 5% level and the 1% level, which suggests that there is a strong correlation between the proportion of lines spoken by Ross and the episode rating.

Monica was a close second in terms of boosting an episode's rating, with a coefficient estimate of 1.16, meaning that each additional percent of lines spoken by her increased the episode's ratings by 1.16. It is also important to note that the coefficient estimate for Monica was only significant at the 11% level.

3.1 Chi-Squared Goodness of Fit Test

We wanted to do a chi-squared goodness of fit test at the 5% level of significance to see if the linear model is a good fit for using lines spoken to predict the rating of the episode *(See Appendix for Code.)*
H_0: The linear model is a good fit for
H_A: H_0 is false

```
Chi-squared test for given probabilities \\

data: observed \\
X-squared = Inf, df = 5, p-value < 2.2e-16
```

Based on the chi-squared test, we can reject the null hypothesis and conclude that the linear model is not a good fit for the data.

4 Conclusion

One of the main limitations of our model is that number of lines spoken is not equivalent to the amount of screen time. The amount of screen time the character gets or even the length of each line spoken can make a difference, since an "okay" would count the same as a monologue in our model. In the future, it would be interesting to regress the amount of screen time, both speaking and non-speaking, on the IMDB ratings.

5 Appendix

R Code:

```
# DATA SOURCE: Friends Sitcom Dataset by Sujay Kapadnis
# LINK: https://www.kaggle.com/datasets/sujaykapadnis/friends

# open files
friends <- read.csv("~/Downloads/friends/friends.csv")
ratings <- read.csv("~/Downloads/friends/friends_info.csv")

# declare new arrays for each character
ross <- array(data = NA, dim = c(236))
rachel <- array(data = NA, dim = c(236))
joey <- array(data = NA, dim = c(236))
phoebe <- array(data = NA, dim = c(236))
monica <- array(data = NA, dim = c(236))
chandler <- array(data = NA, dim = c(236))
guest <- array(data = NA, dim = c(236))

# record total utterances to calculate percentage later
episode_total <- array(data = NA, dim = c(236))

# count number of utterances without scene directions
episode_count <- 0

# current index in all arrays
index <- 1

# current episode (starting on S01E01)
current_episode <- friends$episode[1]

# count the number of utterances per character
mon_count <- 0
ross_count <- 0
rach_count <- 0
joey_count <- 0
phoeb_count <- 0
chan_count <- 0
guest_count <- 0

for (i in 1:nrow(friends)) {
  if (!is.na(friends$speaker[i])) {
    if (friends$speaker[i] != "Scene Directions") {
      episode_count <- episode_count + 1
    }

    # check to see which character talked
    if (friends$speaker[i] == "Monica Geller") {
      mon_count <- mon_count + 1
```

```
    } else if (friends$speaker[i] == "Ross Geller") {
      ross_count <- ross_count + 1
    } else if (friends$speaker[i] == "Rachel Green") {
      rach_count <- rach_count + 1
    } else if (friends$speaker[i] == "Joey Tribbiani") {
      joey_count <- joey_count + 1
    } else if (friends$speaker[i] == "Phoebe Buffay") {
      phoeb_count <- phoeb_count + 1
    } else if (friends$speaker[i] == "Chandler Bing") {
      chan_count <- chan_count + 1
    } else if (friends$speaker[i] != "Scene Directions") {
      guest_count <- guest_count + 1
    }

    # run if the episode changes
    if (friends$episode[i] != current_episode) {

      # record counts
      monica[index] <- mon_count
      ross[index] <- ross_count
      rachel[index] <- rach_count
      joey[index] <- joey_count
      phoebe[index] <- phoeb_count
      chandler[index] <- chan_count
      guest[index] <- guest_count
      episode_total[index] <- episode_count

      # reset variables
      mon_count <- 0
      ross_count <- 0
      rach_count <- 0
      joey_count <- 0
      phoeb_count <- 0
      chan_count <- 0
      guest_count <- 0
      episode_count <- 0

      # update episode
      index <- index + 1
      current_episode <- friends$episode[i]

    }
  }
}

# Record counts for the last episode
monica[index] <- mon_count
ross[index] <- ross_count
rachel[index] <- rach_count
joey[index] <- joey_count
```

```r
phoebe[index] <- phoeb_count
chandler[index] <- chan_count
guest[index] <- guest_count
episode_total[index] <- episode_count

# copy over ratings into an array
rating <- array(data = NA, dim = c(236))
for(i in 1:nrow(ratings)){
  rating[i] <- ratings$imdb_rating[i]
}

# change values to proportion of lines spoken by each character
for(i in 1:nrow(ratings)){
  monica[i] <- monica[i]/episode_total[i]
  ross[i] <- ross[i]/episode_total[i]
  rachel[i] <- rachel[i]/episode_total[i]
  joey[i] <- joey[i]/episode_total[i]
  phoebe[i] <- phoebe[i]/episode_total[i]
  chandler[i] <- chandler[i]/episode_total[i]
  guest[i] <- episode_total[i]
}

# combine arrays into one dataframe
character_data <- data.frame(
  Episode = seq_along(ross),
  Ross = ross,
  Rachel = rachel,
  Joey = joey,
  Phoebe = phoebe,
  Monica = monica,
  Chandler = chandler,
  Guest = guest,
  Rating = ratings$imdb_rating
)

# regress each characters' proportions on the episode rating in a
# multi-variable linear regression
summary(lm(Rating ~ (Ross + Rachel + Joey + Phoebe + Monica + Chandler +
    Guest), data = character_data))

# CHI SQUARED

# bin declarations
bin8 <- 0
bin825 <-0
bin85 <- 0
bin875 <- 0
bin9 <- 0
binMax <- 0
```

```
for(i in 1:nrow(rating)){
  if(rating[i] < 8){
    bin8 <- bin8 + 1
  }
  else if(rating[i] < 8.25){
    bin825 <- bin825 + 1
  }
  else if(rating[i] < 8.5){
    bin85 <- bin85 + 1
  }
  else if(rating[i] < 8.75){
    bin875 <- bin875 + 1
  }
  else if(rating[i] < 9){
    bin9 <- bin9 + 1
  }
  else{
    binMax <- binMax + 1
  }
}

observed <- c(bin8, bin825, bin85, bin875, bin9, binMax)

# reset bins
bin8 <- 0
bin825 <-0
bin85 <- 0
bin875 <- 0
bin9 <- 0
binMax <- 0


for(i in 1:nrow(ross)){
  expected_rating <- 7.344 + (1.75 * ross[i])
      +0.316*rachel[i]+0.300*joey[i]+0.717*phoebe[i]+1.169*monica[i]+0.121*chandler[i]+0.002*guest[i]
  if(expected_rating < 8){
    bin8 <- bin8 + 1
  }
  else if(expected_rating < 8.25){
    bin825 <- bin825 + 1
  }
  else if(expected_rating < 8.5){
    bin85 <- bin85 + 1
  }
  else if(expected_rating < 8.75){
    bin875 <- bin875 + 1
  }
  else if(expected_rating < 9){
    bin9 <- bin9 + 1
  }
```

```
  else{
    binMax <- binMax + 1
  }
}

expected <- c(bin8, bin825, bin85, bin875, bin9, binMax)

chisq.test(x=observed, p=expected, rescale.p = TRUE)
```