Corrected Project 2: Regression

Shariq Mallick

05/10/2021

The Setup

This section is taken from when this data was used in Project 1. Using the website Kaggle, multiple datasets pertaining to content found on different streaming services were collected. Two datasets that had the most similar data to one another were chosen in order to have the cleanest joining and most reasonable conclusions to the analysis. One dataset contains details regarding 50 Netflix TV shows, while the other contains details regarding Amazon Prime TV shows. This data exploration will focus on rating, year, and audience age. This data is interesting to me as an exploration of the range of quality on different services could inform one's decision on choosing one service over another.

A limitation of this data is that the Netflix data is specifically the top 50 most popular shows on Netflix while the prime shows are a wider range. The prime TV shows dataset contain many of the most popular shows on Amazon Prime, but it also contains many of the less acclaimed shows as well. It is expected that shows aimed at younger audiences will have lower IMDB scores than those aimed at older audiences, and that shows provided in the Netflix data set will see higher IMDB scores.

```
prime<-read.csv("D:/Downloads/Rstuff/Prime.csv")
netflix<-read.csv("D:/Downloads/Rstuff/Netflix.csv")

# Due to a lack of numeric variables, another dataset which includes an extra numeric variable f
or many of the shows being tested is being used.
tv_shows<-read.csv("D:/Downloads/Rstuff/tv_shows.csv")

glimpse(netflix)
```

```
## Rows: 50
## Columns: 5
## $ Titles      <chr> "Breaking Bad", "Game of Thrones", "Rick and Morty", "Dark~
## $ Year        <int> 2008, 2011, 2013, 2017, 2016, 2005, 2010, 2019, 1994, 2005~
## $ Rating      <chr> "18+", "18+", "18+", "16+", "16+", "7+", "16+", "18+", "16~
## $ IMDB_Rating <dbl> 9.5, 9.3, 9.2, 8.8, 8.8, 9.2, 9.1, 9.4, 8.9, 8.9, 8.7, 9.3~
## $ Netflix     <int> 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1~
```

```
glimpse(prime)
```

```
## Rows: 404
## Columns: 8
## $ S.no.                   <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14,~
## $ Name.of.the.show        <chr> "Pataal Lok", "Upload", "The Marvelous Mrs. Ma~
## $ Year.of.release         <int> 2020, 2020, 2017, 2019, 2016, 2019, 2018, 2018~
## $ No.of.seasons.available <int> 1, 1, 3, 2, 2, 1, 2, 1, 1, 1, 2, 2, 2, 2, 1, 1~
## $ Language                <chr> "Hindi", "English", "English", "Hindi", "Engli~
## $ Genre                   <chr> "Drama", "Sci-fi comedy", "Drama, Comedy", "Dr~
## $ IMDb.rating             <dbl> 7.5, 8.1, 8.7, 5.3, 8.7, 8.3, 7.5, 8.5, 8.6, 8~
## $ Age.of.viewers          <chr> "18+", "16+", "16+", "18+", "18+", "18+", "16+~
```

These datasets contain some unneeded variables for this exploration. Therefore, the data must be tidied.

Citations for data: Yadav, Ritesh. “Trending TV Shows on Netflix” Kaggle, 18 Jan. 2021, https://www.kaggle.com/ritesh2000/trending-tv-shows-on-netflix (https://www.kaggle.com/ritesh2000/trending-tv-shows-on-netflix). Jauhari, Neelima. “Amazon Prime TV Shows.” Kaggle, 13 Oct. 2020, www.kaggle.com/nilimajauhari/amazon-prime-tv-shows. Bhatia, Ruchi. “TV Shows on Netflix, Prime Video, Hulu and Disney+.” Kaggle, 25 May 2020, www.kaggle.com/ruchi798/tv-shows-on-netflix-prime-video-hulu-and-disney.

Tidying Up and Joining

This section is taken from when this data was used in Project 1. The data must be tidied in order to be ready for use.

```
# The tidyverse package contains the needed functions to tidy and visualize the data.
library(tidyverse)

# The prime data contains a redundant variable in s.no, and there are several variables that are
not contained in the netflix dataset. These must be removed
tdprime<-prime%>%
  select(-S.no.,-Language, -Genre,-No.of.seasons.available)%>%
# The remaining variables need to be renamed to be the same as the netflix data
  rename(Titles="Name.of.the.show", Year="Year.of.release", IMDB_Rating="IMDb.rating", Rating="A
ge.of.viewers")%>%
# Missing values need to be removed as well
  drop_na()

# Creating a new variable that labels these shows as being from Prime video will help with categ
orization when joining
tdprime$Service<-"Prime"
glimpse(tdprime)
```

```
## Rows: 182
## Columns: 5
## $ Titles      <chr> "Pataal Lok", "Upload", "The Marvelous Mrs. Maisel", "Four~
## $ Year        <int> 2020, 2020, 2017, 2019, 2016, 2019, 2018, 2018, 2019, 2019~
## $ IMDB_Rating <dbl> 7.5, 8.1, 8.7, 5.3, 8.7, 8.3, 7.5, 8.5, 8.6, 8.0, 8.0, 8.7~
## $ Rating      <chr> "18+", "16+", "16+", "18+", "18+", "18+", "16+", "18+", "1~
## $ Service     <chr> "Prime", "Prime", "Prime", "Prime", "Prime", "Prime", "Pri~
```

The netflix dataset is already tidy except for the netflix variable which denotes whether a show is a netflix original. This data is unnecessary, but using pivot_longer this variable can be used to denote netflix as the streaming service

```
tdnetflix<-netflix%>%
  pivot_longer(cols = Netflix, names_to = "Service", values_to = "Netflix")%>%
# Once the new variable was created, the redundant one could be removed
  select(-Netflix)
glimpse(tdnetflix)
```

```
## Rows: 50
## Columns: 5
## $ Titles      <chr> "Breaking Bad", "Game of Thrones", "Rick and Morty", "Dark~
## $ Year        <int> 2008, 2011, 2013, 2017, 2016, 2005, 2010, 2019, 1994, 2005~
## $ Rating      <chr> "18+", "18+", "18+", "16+", "16+", "7+", "16+", "18+", "16~
## $ IMDB_Rating <dbl> 9.5, 9.3, 9.2, 8.8, 8.8, 9.2, 9.1, 9.4, 8.9, 8.9, 8.7, 9.3~
## $ Service     <chr> "Netflix", "Netflix", "Netflix", "Netflix", "Netflix", "Ne~
```

```
# Now that the datasets are more similar, a full join function can be used to merge them togethe
r fully
fulldata<-tdnetflix%>%
  full_join(tdprime)
```

```
## Joining, by = c("Titles", "Year", "Rating", "IMDB_Rating", "Service")
```

In order to have another numeric variable to test, another dataset containing information on Rotten Tomatoes scores will be joined as well. Rotten Tomato scores differ from IMDB ratings as RT scores are a measure of how many positive vs negative critical review there are, while IMDB ratings are an average of the scores critics gave out of 10.

```
RT<-tv_shows%>%
# First the variables were renamed to be easier to type and match the previous data
  rename(Titles="Title",RT="Rotten.Tomatoes")%>%
# Only the RT variable is necessary, while the Titles serve as the key for the joining
  select(Titles,RT)%>%
# Missing values can be removed since they add no useful information to the data
  drop_na()
# Using a leftjoin ensuress only data that is already relevant is added.
fulldata<- fulldata%>%
  left_join(RT,by="Titles")

glimpse(fulldata)
```

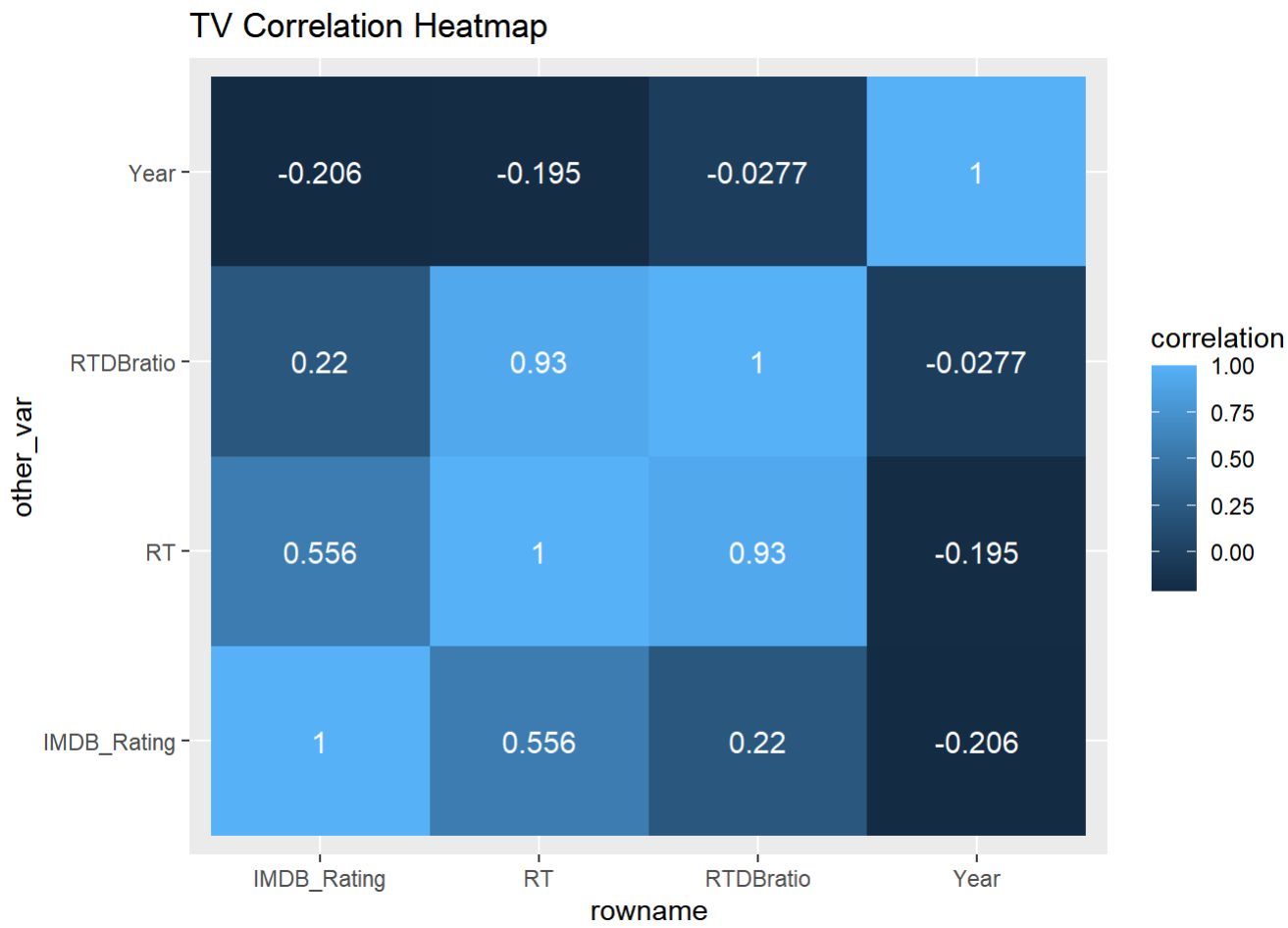
```
## Rows: 233
## Columns: 6
## $ Titles      <chr> "Breaking Bad", "Game of Thrones", "Rick and Morty", "Dark~
## $ Year        <int> 2008, 2011, 2013, 2017, 2016, 2005, 2010, 2019, 1994, 2005~
## $ Rating      <chr> "18+", "18+", "18+", "16+", "16+", "7+", "16+", "18+", "16~
## $ IMDB_Rating <dbl> 9.5, 9.3, 9.2, 8.8, 8.8, 9.2, 9.1, 9.4, 8.9, 8.9, 8.9, 8.7~
## $ Service     <chr> "Netflix", "Netflix", "Netflix", "Netflix", "Netflix", "Ne~
## $ RT          <dbl> 0.96, NA, 0.94, 0.94, 0.93, 1.00, 0.78, NA, NA, 0.81, 0.96~
```

Some cases had to be dropped, specifically cases that did not include IMDB ratings in the prime video dataset as missing values in crucial numeric variables would make it difficult to explore the data. Later on, values that are missing in the RT variable may need to be dropped as well; however, for now it is possible to work around the missing data. There are cases of multiple RT scores per title, and this is due to the presence of multiple seasons.

EDA

This section is partially taken from when this data was used in Project 1. A correlation heatmap that shows the similarity between the RT scores and the IMDB ratings would be a suitable way of visualizing the ratio between the two that is displayed by the ratio variables above.

```
fulldata_num<-fulldata%>%
# Mutate to include the ratio from above
  mutate(RTDBratio=RT*10/IMDB_Rating)%>%
# Select for numeric variables
  select_if(is.numeric)
# Create heatmap
cor(fulldata_num,use = "pairwise.complete.obs")%>%
# Save as data frame
  as.data.frame()%>%
# Convert rownames to an explicit variable
  rownames_to_column()%>%
# Pivot so that all the correlations appear in the same column
  pivot_longer(-1, names_to = "other_var",values_to = "correlation")%>%
# Use geom_tile, add a title, and label the values based on the correlation
  ggplot(aes(rowname, other_var, fill=correlation)) + geom_tile() + ggtitle("TV Correlation Heat
map") +   geom_text(aes(label = signif(correlation,digits=3)), color = "white", size = 4)
```

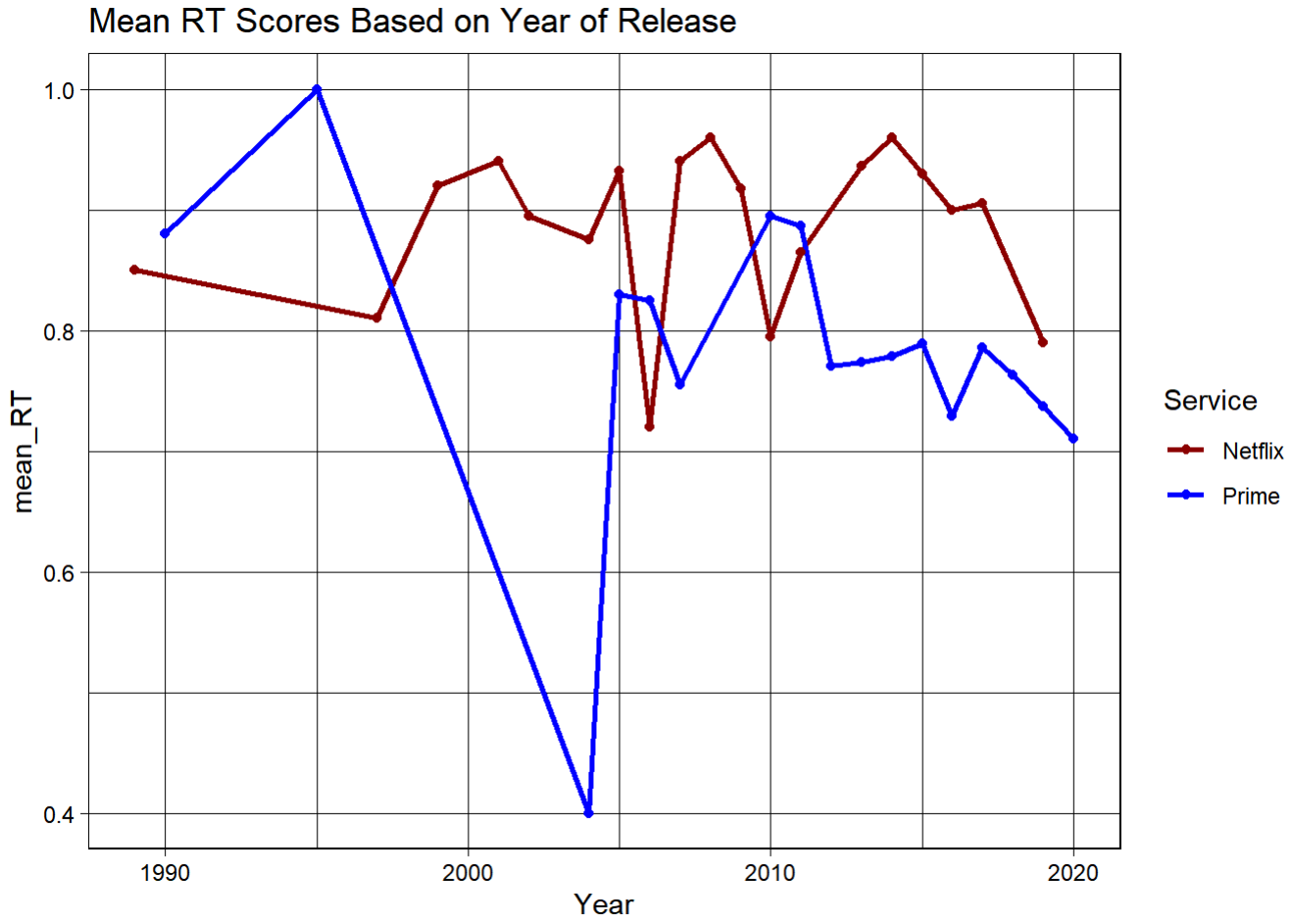


This heatmap is slightly uninformative due to the low number of numeric variables in this dataset. Interestingly the ratio correlates more strongly with the RT scores than the IMDB ratings. This is likely due to the larger degree of variation in the RT scores more heavily affecting the ratio, or it could simply be due to the fact that the RT scores are the numerator of the ratio.

```
fulldata%>%
# Group by the categorical variables to be depicted/explored
  group_by(Year,Service)%>%
# Use the summarize function to output means for each chosen categorical variable
  summarize(mean_RT=mean(RT, na.rm=T))%>%
# Drop Missing values to avoid gaps or errors in the data
  drop_na()%>%
# Use ggplot to plot the desired variables
  ggplot(aes(x=Year,y=mean_RT,color=Service))+
# Adjust the size to increase visibility
  geom_line(size=1,stat = "summary")+
# Set a theme to customize the background
  theme_linedraw() +
# Choose a title
  ggtitle("Mean RT Scores Based on Year of Release") +
# Add points to make individual observations more visible
  geom_point() +
  scale_color_manual(values=c("dark red","blue"))
```

```
## `summarise()` has grouped output by 'Year'. You can override using the `.groups` argument.
```

```
## No summary function supplied, defaulting to `mean_se()`
```

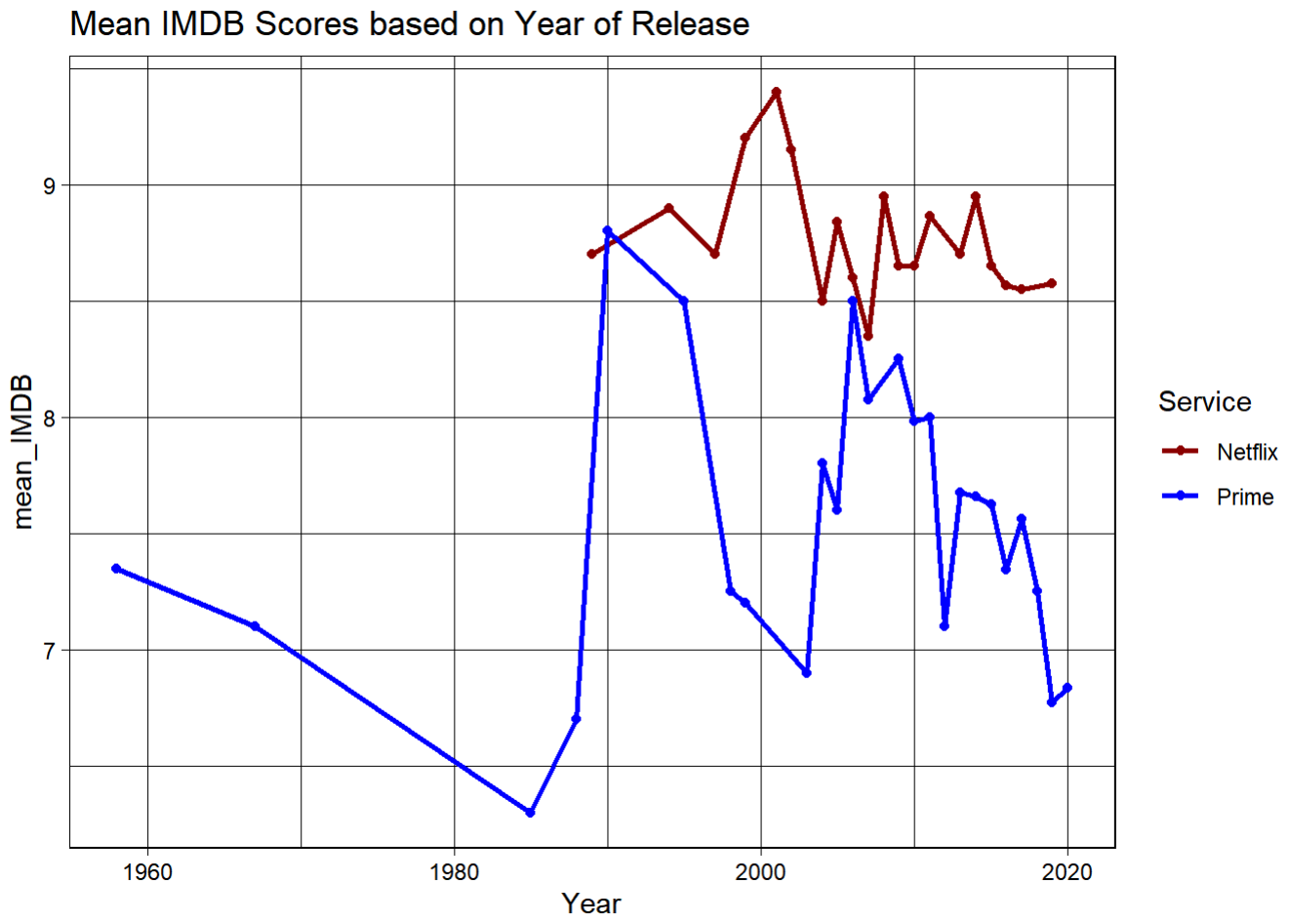


This line graph depicts the average Rotten Tomatoes score for each platform based on the year of the shows original release. Netflix has a more consistent output over time, which suggests that they search for content that has high audience appeal. Earlier analysis showed that there was a greater variation in the Prime Video dataset, so it is likely that this visualization is being skewed negatively for RT scores for the Prime platform due to imperfect sampling.

```
fulldata%>%
# Group by the categorical variables to be depicted/explored
  group_by(Year,Service)%>%
# Use the summarize function to output means for each chosen categorical variable
  summarize(mean_IMDB=mean(IMDB_Rating, na.rm=T))%>%
# Drop Missing values to avoid gaps or errors in the data
  drop_na()%>%
# Use ggplot to plot the desired variables
  ggplot(aes(x=Year,y=mean_IMDB,color=Service))+
# Adjust the size to increase visibility
  geom_line(size=1,stat = "summary")+
# Set a theme to customize the background
  theme_linedraw() +
# Choose a title
  ggtitle("Mean IMDB Scores based on Year of Release") +
# Add points to make individual observations more visible
  geom_point() +
  scale_color_manual(values=c("dark red","blue"))
```

```
## `summarise()` has grouped output by 'Year'. You can override using the `.groups` argument.
```

```
## No summary function supplied, defaulting to `mean_se()`
```



By creating a similar graph for the IMDB variable, a direct comparison between the two numeric variables can be made. It is clear that IMDB scores trend significantly lower in the Prime data as compared to the Netflix data. Prime seems to hold a much higher quantity of content from a larger range of time, but the quality of this content is less consistent. Rotten Tomatoes scores seem to have a greater degree of variation in general as the lowest scores are much lower, while the highest are much greater.

MANOVA

Performing a MANOVA tests whether any numeric variables are significantly affected by the different categorical variables.

```
# Perform MANOVA with 2 response variables listed in cbind()
mandata <- manova(cbind(IMDB_Rating,RT) ~ Rating, data = fulldata)
summary(mandata)
```

```
##            Df  Pillai approx F num Df den Df   Pr(>F)
## Rating      4 0.17474   2.6327      8    220 0.008974 **
## Residuals 110
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# MANOVA is significant, so we now perform one-way ANOVA for each variable
summary.aov(mandata)
```

```
##  Response IMDB_Rating :
##              Df Sum Sq Mean Sq F value   Pr(>F)
## Rating        4  7.051 1.76281  4.2603 0.003039 **
## Residuals   110 45.515 0.41377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Response RT :
##              Df Sum Sq  Mean Sq F value Pr(>F)
## Rating        4 0.1186 0.029656  0.9361 0.4459
## Residuals   110 3.4847 0.031679
##
## 118 observations deleted due to missingness
```

```
# ANOVA is significant for IMDB Rating(F = 4.2603, p<.01), but not for RT(F = 0.9361, p > .05).
  # We can perform post-hoc analysis for IMDB_Rating
  pairwise.t.test(fulldata$IMDB_Rating,fulldata$Rating, p.adj="none")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  fulldata$IMDB_Rating and fulldata$Rating
##
##     13+    16+    18+     7+
## 16+ 0.0679 -      -       -
## 18+ 0.0024 0.0100 -       -
## 7+  0.0769 0.6297 0.3849  -
## All 0.9750 0.0109 2.7e-05 0.0366
##
## P value adjustment method: none
```

```
  # To interpret the p-values, what is the value for the Bonferonni alpha?
    # Original p-value/Number of tests performed. Two tests were performed, a t-test and manova.
    0.05/12
```

```
## [1] 0.004166667
```

```
    # Given this alpha, there is one result, All ages vs 7+, that is no longer significant.
  # 13+ vs 18+, 18+ vs 16+, All  vs 16+, and All vs 18+ are shown to be significantly different
 in terms of RT and IMBD_Rating after adjusting for multiple comparisons (Bonferroni alpha = .02
5)
  # The chance of a type 1 error can be calculated using the formula 1 - 0.95^(#oftests)
    1-0.95^2 # The chance is equal to 0.0975.
```

```
## [1] 0.0975
```

Assumptions

There are many assumptions in MANOVA. This data does not pass all assumptions. While the sample has independent observations, the sample is not necessarily random as both datasets trended towards the more popular shows available on each platform. There is a lack of data in some of the categorical variables that leads to a lack of homogeneity in each group, but in many they are relatively homogeneous. There are no extreme outliers and no multicollinearity, but not all response variables have linear relationships. There is multivariate normality of the numeric response variables.

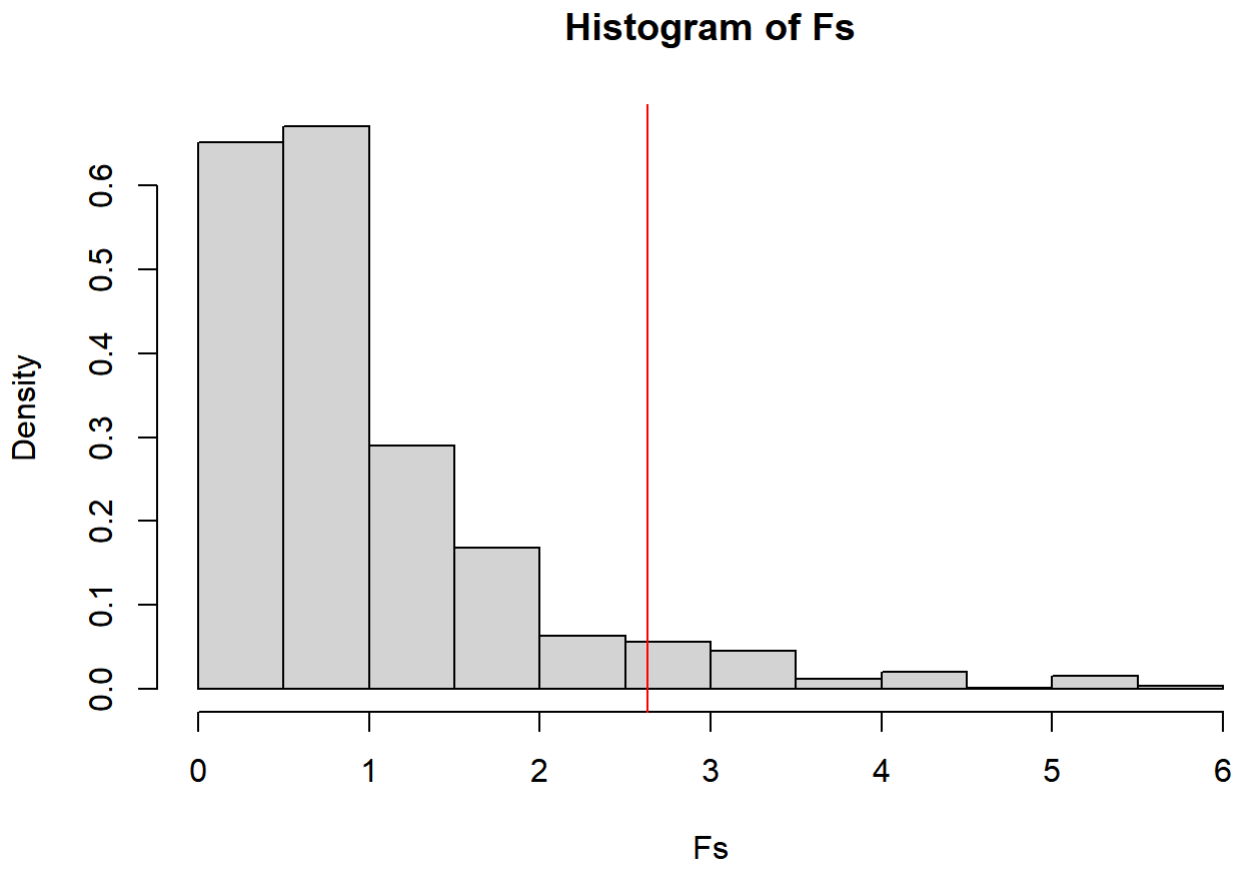
Interpretation

A one-way MANOVA was conducted to determine the effect of the Age Rating (All ages, 7+, 13+, 16+, 18+) on two dependent variables (RT Scores and IMDB Ratings). Significant differences were found among the three iris species for at least one of the dependent variables (Pillai's trace = 0.93, pseudo F(4,294) = 63.85, p< .0001).Univariate ANOVAs for each dependent variable were conducted as follow-up tests to the MANOVA, were also significant for Sepal Length (F(2,147) = 119.26, p< .0001) and Petal Width (F(2,147) = 960, p< .0001).Post hoc analysis was performed conducting pairwise comparisons to determine which species differed in sepal length and petal width. All three Species were found to differ significantly from each other in terms of sepal length and petal width after adjusting for multiple comparisons (Bonferroni α=.0056).

Randomization Test

Assumptions for the MANOVA test were violated, so a comparison will be made between the categorical variable “Rating” and the “IMDB_Rating” numeric variable. This means that a F statistic test will be performed. The null hypothesis is that the difference in the observed F statistic is not significantly different from the ones found in the randomization test. The alternative hypothesis is that the difference in the F-statistic is significant.

```
# Observed F-statistic, running anova
obs_F <-  2.6327

# Randomization test (using replicate)
Fs <- replicate(1000,{
  # Randomly permute the response variable across doses
  new <- fulldata %>%
    drop_na()%>%
    mutate(RT = sample(RT))
  # Compute variation within groups
  SSW <- new %>%
    group_by(Rating) %>%
    summarize(SSW = sum((RT - mean(RT))^2)) %>%
    summarize(sum(SSW)) %>%
    pull
  # Compute variation between groups
  SSB <- new %>%
    mutate(mean = mean(RT)) %>%
    group_by(Rating) %>%
    mutate(groupmean = mean(RT)) %>%
    summarize(SSB = sum((mean - groupmean)^2)) %>%
    summarize(sum(SSB)) %>%
    pull
  # Compute the F-statistic (ratio of MSB and MSW)
  # df for SSB is 5 groups - 1 = 4
  # df for SSW is 115 observations - 5 groups = 110
  (SSB/4)/(SSW/110)
})

# Represent the distribution of the F-statistics for each randomized sample
hist(Fs, prob=T); abline(v = obs_F, col="red",add=T)
```




```
# Calculate the proportion of F statistic that are greater than the observed F-statistic
mean(Fs > obs_F)
```

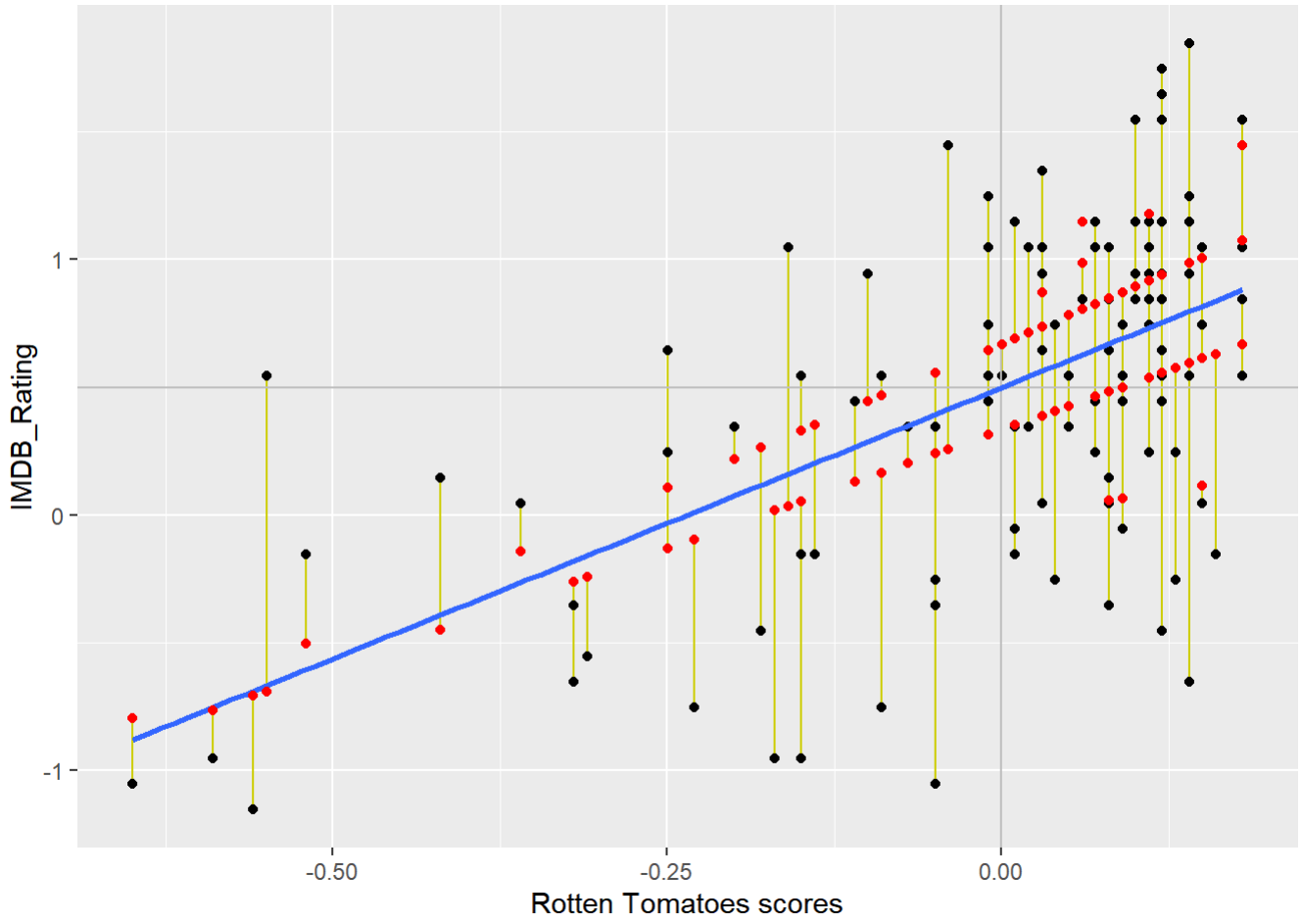
```
## [1] 0.066
```

Only 8.4% of the F statistics were greater than the observed F stat. The histogram shows that there is a large left skew to the F stat data, which suggests that variation within between groups is greater than variation within groups.

Linear Regression Model

```
## Numeric variables must be mean-centered
# Create a function to center scale the numeric variable
center_scale <- function(x) {
    scale(x, scale = FALSE)
}
# Apply it and add it to a dataset
cendata = fulldata
cendata$IMDB_Rating<-center_scale(fulldata$IMDB_Rating)
cendata$RT<-center_scale(fulldata$RT)
# Drop NA values
cendata<-cendata%>%
  drop_na()
# Create a fit of the IMDB_Rating and RT
fit<-lm(data = cendata,IMDB_Rating~RT*Rating)
# Save the fitted values in a vector
fits <- fit$fitted.values
## Create a graph of the data and include the fit
# Start by using ggplot
ggplot(cendata, aes(RT,IMDB_Rating)) +
  # Segments between points and fitted line
  geom_segment(aes(xend = RT, yend = fits), color = "yellow3") +
  # Points (on top of line so added as a second layer)
  geom_point()  +
  # Horizontal line representing the mean distance
  geom_hline(yintercept = mean(cendata$IMDB_Rating), col = 'gray') +
  # Vertical line representing the mean speed
  geom_vline(xintercept = mean(cendata$RT), col='gray') +
  # Regression line (without confidence interval)
  geom_smooth(method = "lm", se = FALSE) +
  # Fitted values
  geom_point(aes(x = RT, y = fits), color = "red") +
  # Label the axes
  xlab("Rotten Tomatoes scores") + ylab("IMDB_Rating")
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
summary(fit)
```

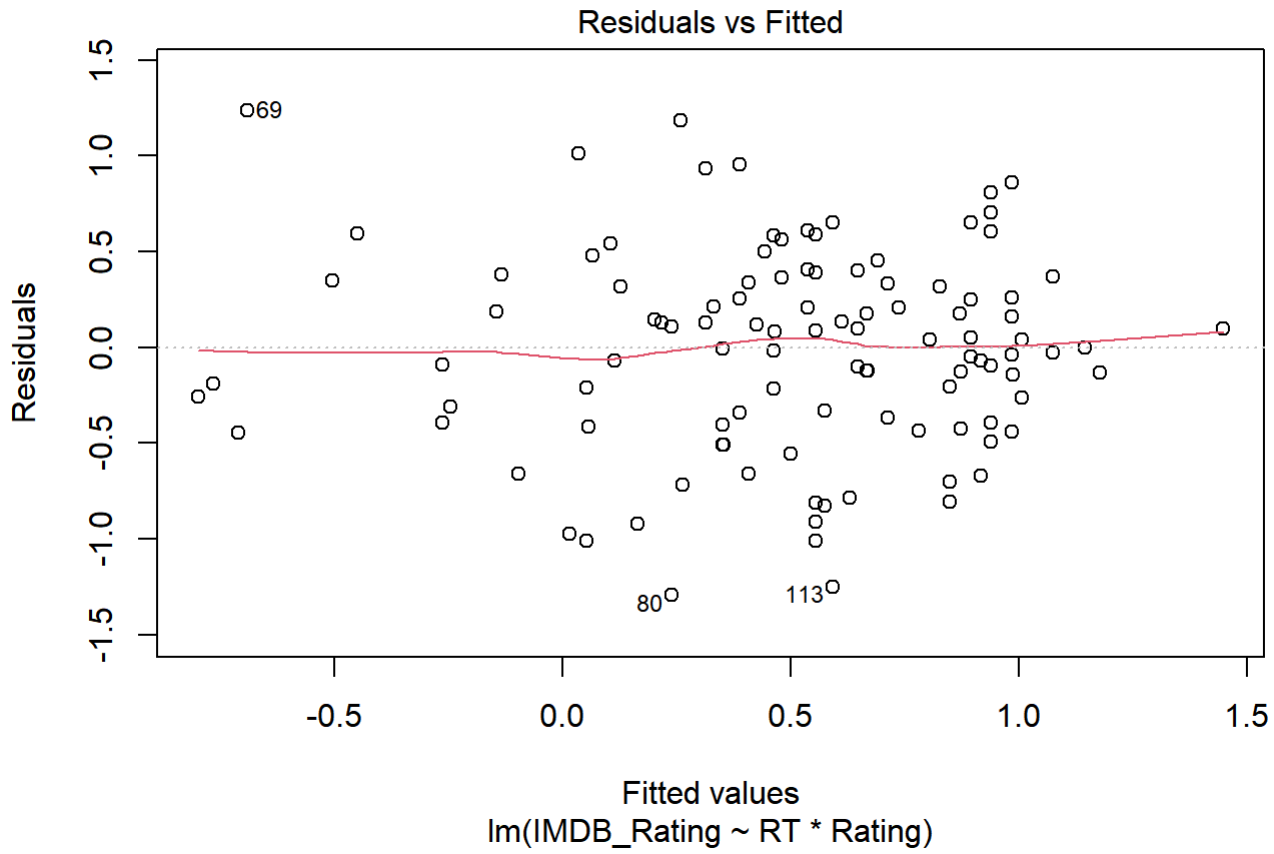
```
##
## Call:
## lm(formula = IMDB_Rating ~ RT * Rating, data = cendata)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.29410 -0.37998  0.04155  0.36402  1.23620
##
## Coefficients: (1 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.0963     0.8150   1.345    0.181
## RT             0.8140    10.0944   0.081    0.936
## Rating16+     -0.7645     0.8183  -0.934    0.352
## Rating18+     -0.4287     0.8186  -0.524    0.602
## Rating7+      -0.3420     0.9714  -0.352    0.725
## RatingAll     -1.1047     0.7819  -1.413    0.161
## RT:Rating16+   1.0466    10.1016   0.104    0.918
## RT:Rating18+   1.4417    10.1044   0.143    0.887
## RT:Rating7+    3.0233    11.1598   0.271    0.787
## RT:RatingAll       NA         NA      NA       NA
##
## Residual standard error: 0.5405 on 106 degrees of freedom
## Multiple R-squared:  0.411,  Adjusted R-squared:  0.3665
## F-statistic: 9.245 on 8 and 106 DF,  p-value: 1.369e-09
```

The coefficient estimate suggests that for every 1.1 point of increase in the IMDB_scores, there is a 0.8 increase in RT score. The R squared value shows the proportion of variance in Y explained by X, which is very low at 0.3665 This means that the fit is not strong. The interaction between RT score and any form of age rating is not significant, and the RT score itself does not vary significantly with IMDB scores.

Assumptions

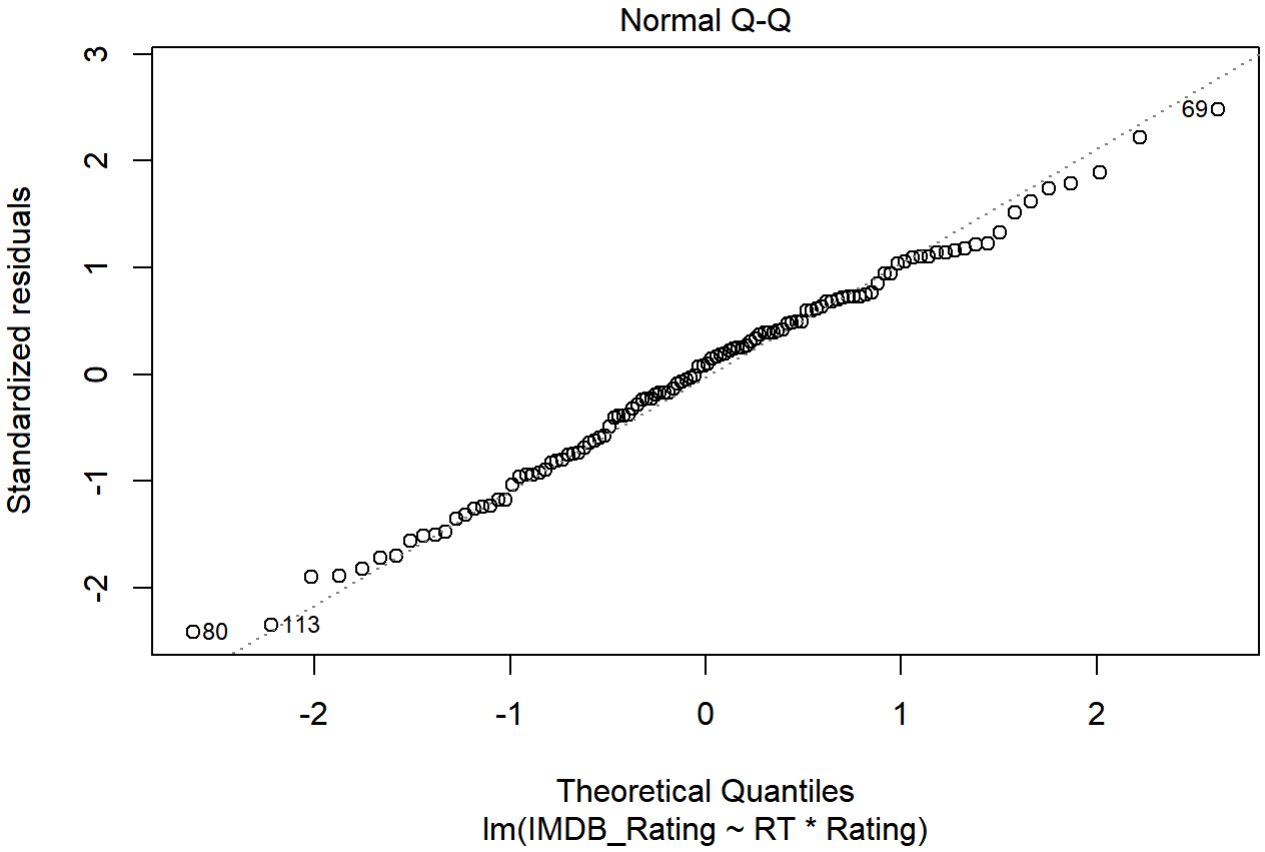
Residuals against fitted values plot to check for any problematic patterns (nonlinear, equal variance)

```
plot(fit, which = 1)
```




```
# Q-Q plot to check for normality of the residuals
plot(fit, which = 2)
```

```
## Warning: not plotting observations with leverage one:
##   107
```



The equal occurrence above and below the 0 suggests that the relationship is linear. There is a higher density at the right side, and a slight funnel shape to the plot. It is linear, and there is relatively equal variance across the range horizontally. This means that no data transformation is necessary. The residuals appear to be normal based on the linearity of the Q-Q plot.

```
## Robust SE allows for the use of corrected SE values, therefore they will be more accurate to
 the truth of the data.
# Uncorrected Standard Errors
summary(fit)$coef
```

```
##                Estimate Std. Error     t value  Pr(>|t|)
## (Intercept)   1.0963025  0.8150299  1.34510705 0.1814614
## RT            0.8139535 10.0944361  0.08063387 0.9358852
## Rating16+    -0.7644881  0.8182800 -0.93426221 0.3522920
## Rating18+    -0.4286727  0.8185563 -0.52369360 0.6015849
## Rating7+     -0.3420121  0.9713581 -0.35209685 0.7254647
## RatingAll    -1.1046512  0.7819117 -1.41275700 0.1606567
## RT:Rating16+  1.0466479 10.1015864  0.10361224 0.9176729
## RT:Rating18+  1.4416599 10.1044363  0.14267594 0.8868170
## RT:Rating7+   3.0232558 11.1598190  0.27090545 0.7869906
```

```
# Robust SE
coeftest(fit, vcov = vcovHC(fit))
```

```
##
## t test of coefficients:
##
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   1.09630         NA      NA       NA
## RT            0.81395         NA      NA       NA
## Rating16+    -0.76449         NA      NA       NA
## Rating18+    -0.42867         NA      NA       NA
## Rating7+     -0.34201         NA      NA       NA
## RatingAll    -1.10465         NA      NA       NA
## RT:Rating16+  1.04665         NA      NA       NA
## RT:Rating18+  1.44166         NA      NA       NA
## RT:Rating7+   3.02326         NA      NA       NA
```

Since assumptions were violated (homoscedasticity and small sample size) using bootstrap samples to estimate coefficients, SEs, fitted values, etc. will adjust for these broken assumptions.

```
# Use the function replicate to repeat the process (similar to a for loop)
samp_SEs <- replicate(5000, {
  # Bootstrap your data (resample observations)
  boot_data <- sample(cendata, replace = TRUE)
  # Fit regression model
  fitboot <- lm(IMDB_Rating ~ RT*Rating, data = cendata)
  # Save the coefficients
  coef(fitboot)
})

# Estimated SEs
samp_SEs %>%
  # Transpose the obtained matrices
  t %>%
  # Consider the matrix as a data frame
  as.data.frame %>%
  # Compute the standard error (standard deviation of the sampling distribution)
  summarize_all(sd)
```

```
##   (Intercept) RT Rating16+ Rating18+ Rating7+ RatingAll RT:Rating16+
## 1           0  0         0         0        0         0            0
##   RT:Rating18+ RT:Rating7+ RT:RatingAll
## 1            0           0           NA
```

```
# We can also consider a confidence interval for the estimates
samp_SEs %>%
  # Transpose the obtained matrices
  t %>%
  # Consider the matrix as a data frame
  as.data.frame %>%
  # Pivot longer to group by and summarize each coefficient
  pivot_longer(everything(), names_to = "estimates", values_to = "value") %>%
  group_by(estimates) %>%
  summarize(lower = quantile(value,.025,na.rm=T), upper = quantile(value,.975,na.rm=T))
```

```
## # A tibble: 10 x 3
##    estimates     lower  upper
##    <chr>         <dbl>  <dbl>
##  1 (Intercept)   1.10   1.10
##  2 Rating16+    -0.764 -0.764
##  3 Rating18+    -0.429 -0.429
##  4 Rating7+     -0.342 -0.342
##  5 RatingAll    -1.10  -1.10
##  6 RT            0.814  0.814
##  7 RT:Rating16+  1.05   1.05
##  8 RT:Rating18+  1.44   1.44
##  9 RT:Rating7+   3.02   3.02
## 10 RT:RatingAll NA     NA
```

```
# Compare to original fit
confint(fit, level = 0.95)
```

```
##                    2.5 %     97.5 %
## (Intercept)   -0.5195736  2.7121785
## RT           -19.1992478 20.8271548
## Rating16+     -2.3868078  0.8578316
## Rating18+     -2.0515401  1.1941947
## Rating7+      -2.2678241  1.5837998
## RatingAll     -2.6548671  0.4455647
## RT:Rating16+ -18.9807296 21.0740255
## RT:Rating18+ -18.5913678 21.4746877
## RT:Rating7+  -19.1021707 25.1486823
## RT:RatingAll          NA         NA
```

The results of the bootstrap SEs suggest that age ratings don't vary greatly on their own, but when looking at the interaction between Age rating and RT scores there is a much greater variation due to the large variation in RT.

Logistic Regression

```
## First a binary categorical variable must be created
# Use dplyr to create a new variable
GRTdata<-fulldata%>%
  # Create a variable that equals 1 if the RT score is greater than the IMDB score
  mutate(GRT=ifelse(RT*10>IMDB_Rating,1,0))%>%
  # Remove missing values
  drop_na()
## Create a logistic regression model
# Define the logit link function (logarithm of odds)
logit <- function(p) log(odds(p))
# Fit a new regression model
GRTfit <- glm(GRT ~ Rating+Service, data = GRTdata, family = binomial(link="logit"))
summary(GRTfit)
```

```
##
## Call:
## glm(formula = GRT ~ Rating + Service, family = binomial(link = "logit"),
##     data = GRTdata)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6698  -1.2879   0.7585   0.9762   1.0708
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -15.9506  2399.5448  -0.007    0.995
## Rating16+      17.0596  2399.5447   0.007    0.994
## Rating18+      16.8222  2399.5447   0.007    0.994
## Rating7+       17.0492  2399.5450   0.007    0.994
## RatingAll      33.1321  2770.7556   0.012    0.990
## ServicePrime   -0.6155     0.4394  -1.401    0.161
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 148.60  on 114  degrees of freedom
## Residual deviance: 141.57  on 109  degrees of freedom
## AIC: 153.57
##
## Number of Fisher Scoring iterations: 15
```

```
# Interpret coefficients as odds ratios
table(GRTdata$GRT, GRTdata$Service)
```

```
##
##     Netflix Prime
##   0      12    28
##   1      32    43
```

```
# Odds of average watcher enjoying a show on Netflix more than a critic does
odds_N = (32/44) / (12/44)
# Odds of average watcher enjoying a show on Prime more than a critic does
odds_P = (43/71) / (28/71)
# Odds ratio of audience, Prime vs Netflix
 odds_P / odds_N
```

```
## [1] 0.5758929
```

```
# Do the same for Rating
table(GRTdata$GRT, GRTdata$Rating)
```

```
##
##     13+ 16+ 18+ 7+ All
##   0   1  19  19  1   0
##   1   0  37  32  3   3
```

```
# 13+,7+, and All don't have enough data to be informative, so I will only be looking at 16+ and
18+.
# Odds of an average watcher enjoying a show rated for 16+ more than a show rated 18+
odds_16 = (37/56)/(19/56)
# Odds of an average watcher enjoying a show rated for 18+ more than a show rated 18+
odds_18 = (32/51)/(19/51)
# Odds ratio of audience, 16+ vs 18+
odds_16/odds_18
```

```
## [1] 1.15625
```

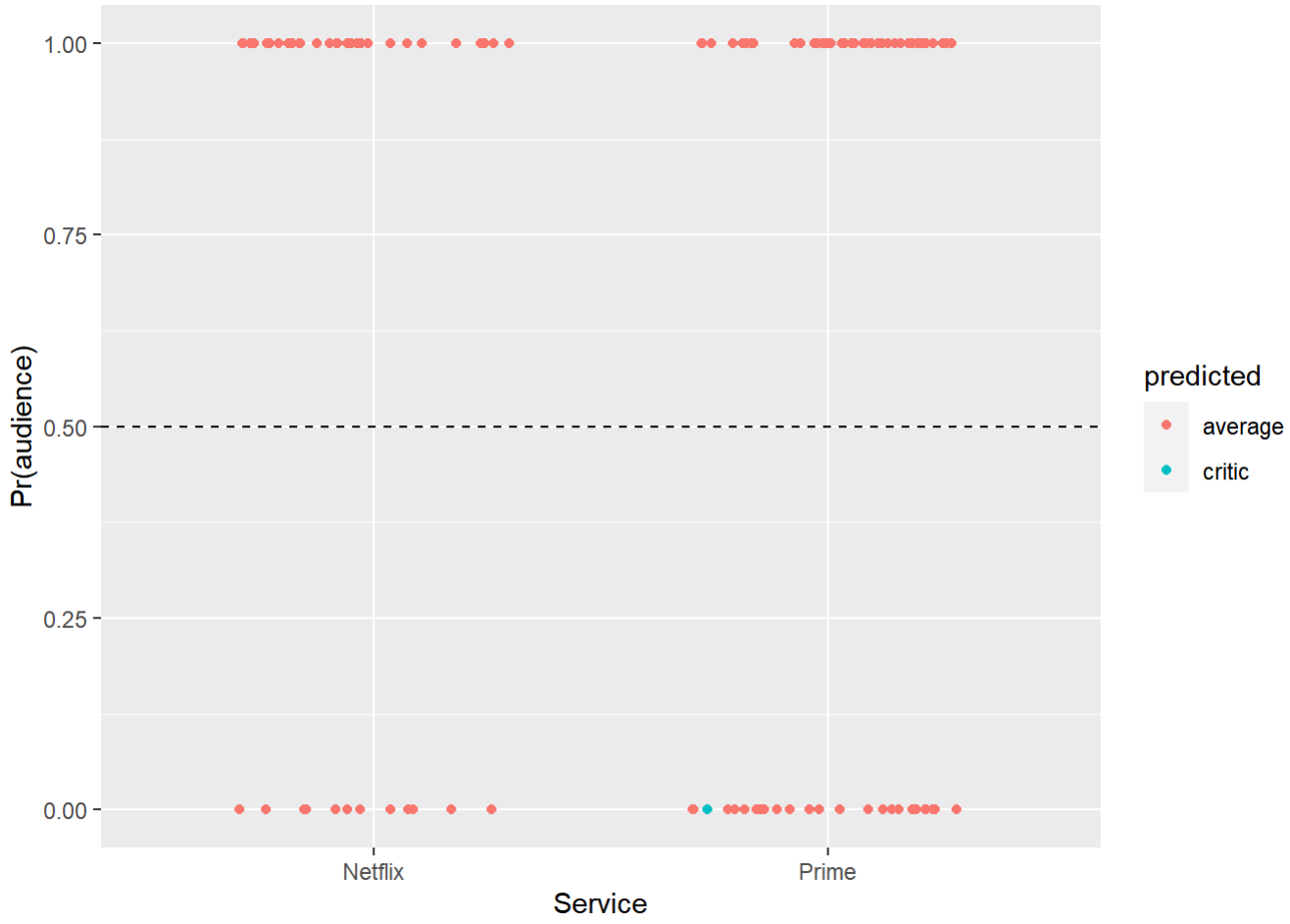
```
# Compare to the exponentiated coefficients of the model
exp(coef(GRTfit))
```

```
##  (Intercept)    Rating16+    Rating18+     Rating7+    RatingAll ServicePrime
## 1.182355e-07 2.563909e+07 2.022026e+07 2.537309e+07 2.449651e+14 5.403810e-01
```

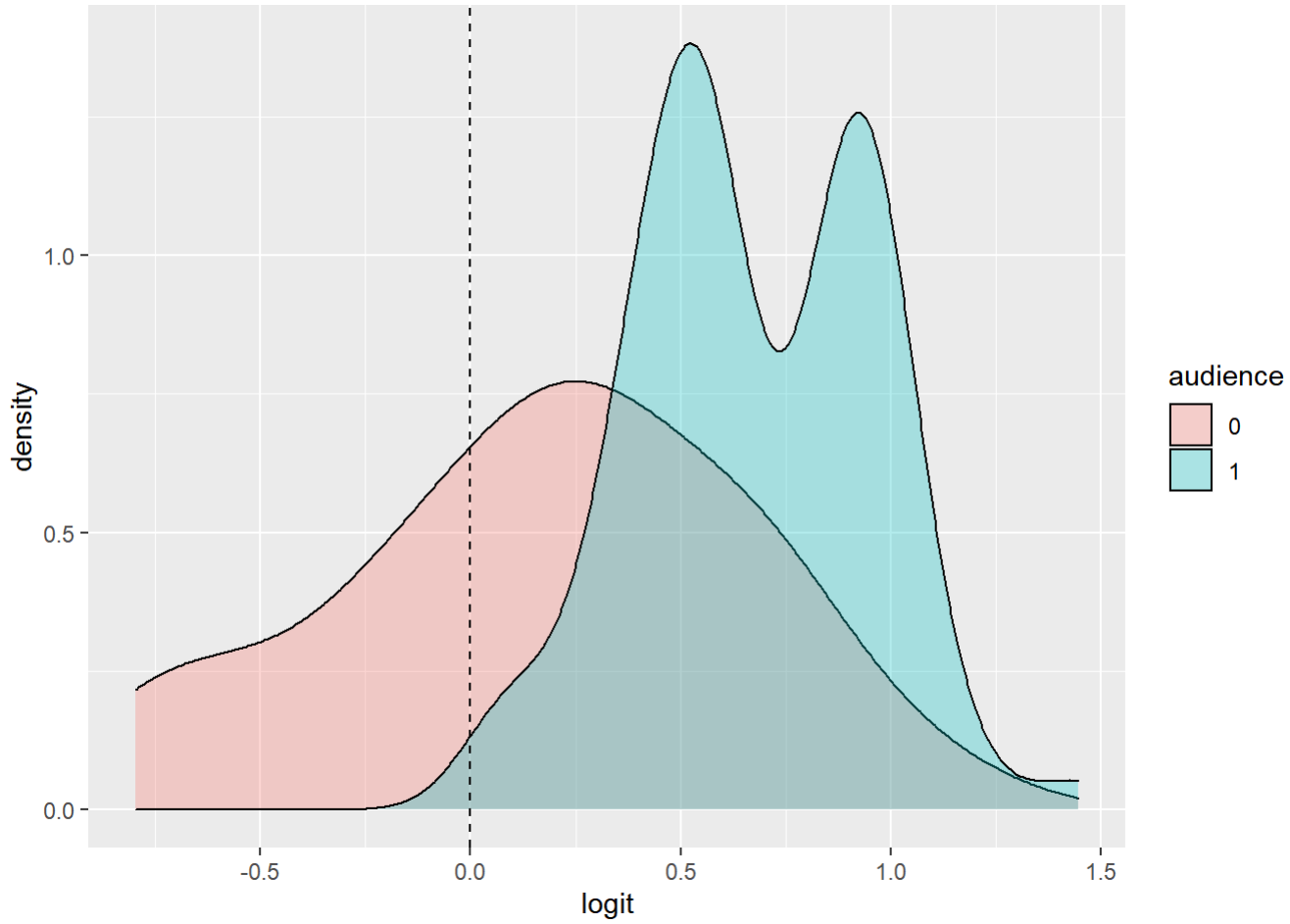
None of the coefficients had a significant p-value, with the Prime service being the one closest to having a significant effect on the likelihood of GRT. It seems like GRT is independent of the variables that I had used to predict it. The odds ratios for the service suggested that the Prime shows were just over half as likely to more enjoyed by the average watcher rather than critics. The odds ratios for the service was similar, but the Ratings all had incredibly high odds ratios that were very different from the ones found through the table.

```
# Add predicted probabilities to the dataset
GRTdata$prob <- predict(GRTfit, type = "response")

# Predicted outcome is based on the probability of GRT being positive
# If the probability is greater than 0.5, the RT value is found to be more popular among the ave
rage watcher than among critics
GRTdata$predicted <- ifelse(GRTdata$prob > .5, "average", "critic")

# Plot the model
ggplot(GRTdata, aes(Service,GRT)) +
  geom_jitter(aes(color = predicted), width = .3, height = 0) +
  stat_smooth(method="glm", method.args = list(family="binomial"), se = FALSE) +
  geom_hline(yintercept = 0.5, lty = 2) +
  ylab("Pr(audience)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
# Save the predicted log-odds in the dataset
GRTdata$logit <- predict(fit)

# Compare to the outcome in the dataset with a density plot
ggplot(GRTdata, aes(logit, fill = as.factor(GRT))) +
  geom_density(alpha = .3) +
  geom_vline(xintercept = 0, lty = 2) +
  labs(fill = "audience")
```




```
# Create a confusion matrix to test the accuracy of the prediction
table(true_condition = GRTdata$GRT, predicted_condition = GRTdata$predicted) %>%
  addmargins
```

```
##              predicted_condition
## true_condition average critic Sum
##            0        39      1  40
##            1        75      0  75
##            Sum     114      1 115
```

```
# Accuracy (correctly classified cases)
(39+1)/115
```

```
## [1] 0.3478261
```

```
# Sensitivity (true positive rate)
0 / 75
```

```
## [1] 0
```

```
# Specificity (true negative rate)
39 / 40
```

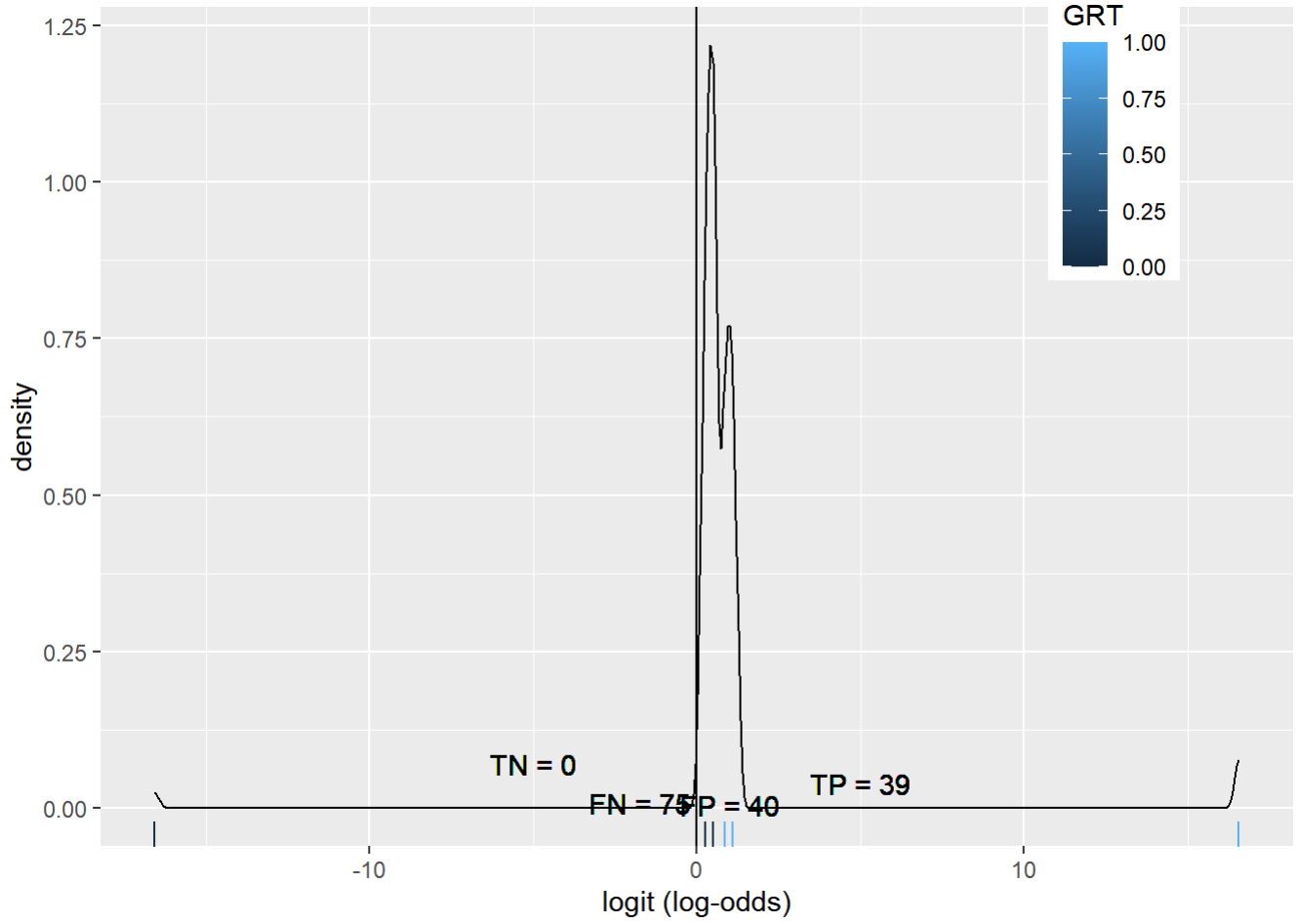
```
## [1] 0.975
```

```
# Recall (Positive Predictive Value)
0/1
```

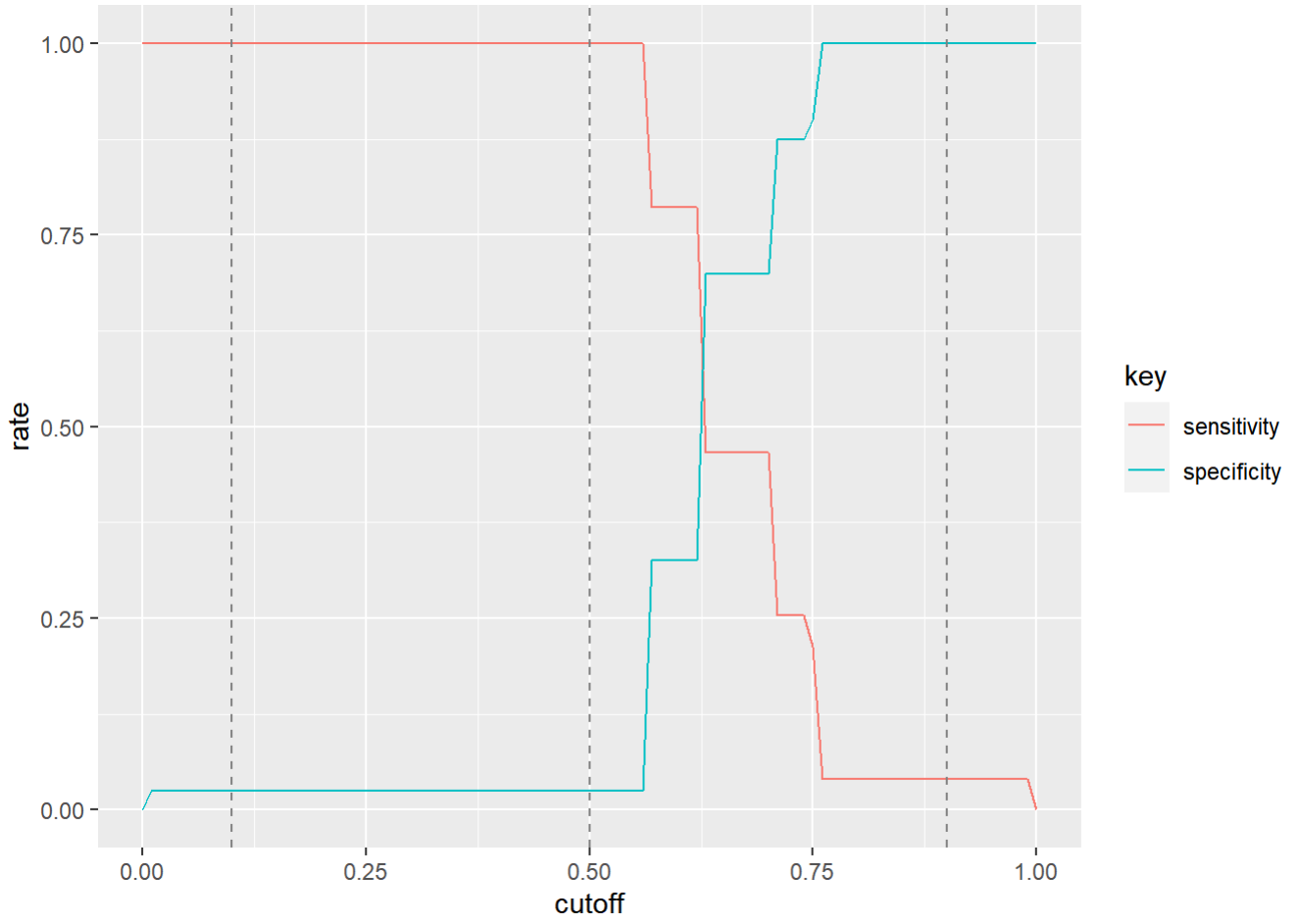
```
## [1] 0
```

The accuracy is about 35%, so approximately 35% of the time it was correct. The model seems to have had only one occurrence of critic, which would explain the low sensitivity and recall. The specificity was very high, but this is because nearly every prediction was a negative. In reality, it seems like the model skews heavily towards the average audience outcome, and is therefore very flawed.

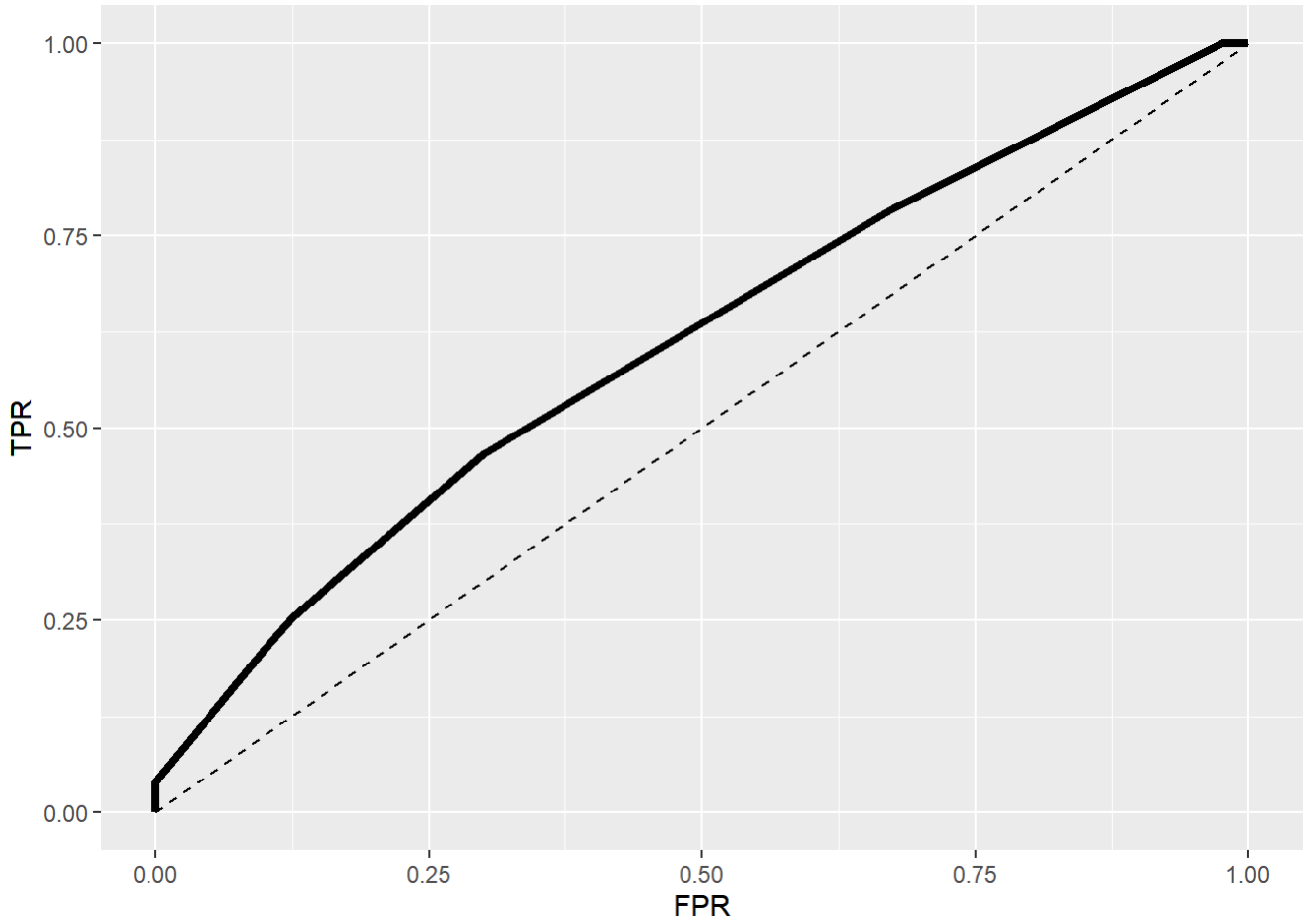
```
# Predicted log odds
GRTdata$logit <- predict(GRTfit, type = "link")

# Density plot of log-odds for each outcome
GRTdata %>%
  ggplot() +
  geom_density(aes(logit, color = GRT, fill = GRT), alpha = .4) +
    geom_rug(aes(logit, color = GRT)) +
  geom_text(x = -5, y = .07, label = "TN = 0") +
  geom_text(x = -1.75, y = .008, label = "FN = 75") +
  geom_text(x = 1, y = .006, label = "FP = 40") +
  geom_text(x = 5, y = .04, label = "TP = 39") +
  theme(legend.position = c(.85,.85)) +
  geom_vline(xintercept = 0) +
  xlab("logit (log-odds)")
```



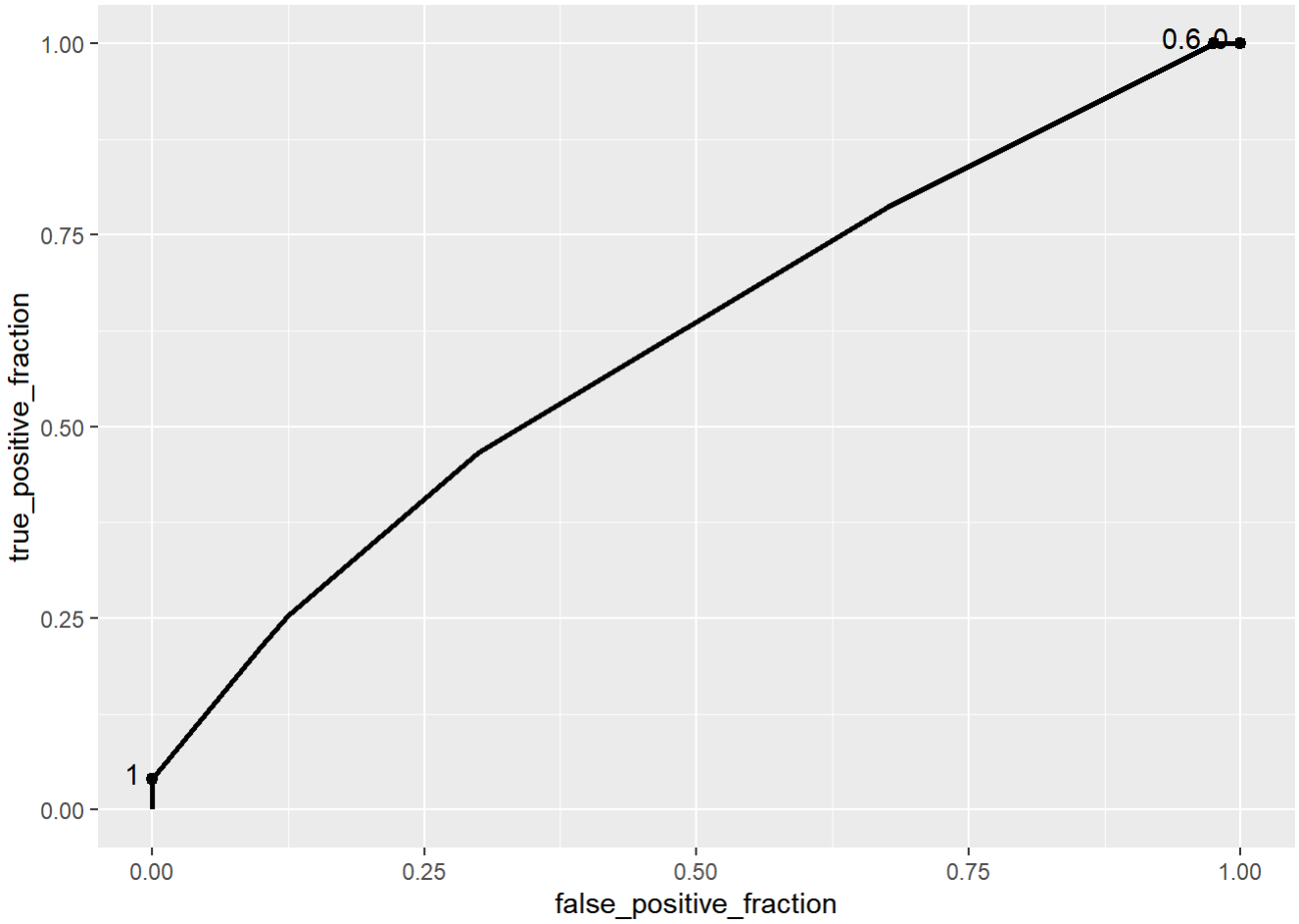

```
## ROC and AUC
# Define functions to calculate sensitivity and specificity for different cutoffs
sens <- function(p, data = data, y = y) mean(data[data$GRT == 1, ]$prob > p)
spec <- function(p, data = data, y = y) mean(data[data$GRT == 0, ]$prob <= p)
# Apply the functions to our data
sensitivity <- sapply(seq(0,1,.01),sens,GRTdata)
specificity<-sapply(seq(0,1,.01),spec,GRTdata)

# Store values of sensitivity and specificity in a dataframe with cutoff values
ROC <- data.frame(sensitivity, specificity, cutoff = seq(0,1,.01))

# Represent the relationship between sensitivity and specificity for different cutoffs
ROC %>%
  pivot_longer(-cutoff, names_to = "key", values_to = "rate") %>%
  ggplot(aes(cutoff, rate, color = key)) +
  geom_path() +
  geom_vline(xintercept = c(.1,.5,.9), lty = 2, color = "gray50")
```




```
# Instead plot Sensitivity (TPR) against 1 - Specificity (FPR): this is called a ROC curve!
ROC$TPR <- sensitivity
ROC$FPR <- 1-specificity

ROC %>%
  ggplot(aes(FPR, TPR)) +
  geom_path(size = 1.5) +
  geom_segment(aes(x = 0, y = 0, xend = 1, yend = 1), lty = 2) +
  scale_x_continuous(limits = c(0,1))
```




```
# Plot ROC depending on values of GRT and its probabilities displaying some cutoff values
ROCplot1 <- ggplot(GRTdata) +
  geom_roc(aes(d = GRT, m = prob), cutoffs.at = list(0.1, 0.5, 0.9))
ROCplot1
```




```
# Order dataset from least to greatest FPR
ROC <- ROC %>%
  arrange(FPR)

# Calculate horizontal distances
widths <- diff(ROC$FPR)

# Calculate vertical distances
heights <- vector()
for(i in 1:100) heights[i] <- ROC$TPR[i] + ROC$TPR[i+1]

# Add the area of each trapezoid
AUC <- sum(heights * widths / 2)
AUC
```

```
## [1] 0.6075
```

The area under the curve is equal to 0.607, which is poor according to the rules of thumb. This is in line with the conclusions drawn earlier about the odds ratios. The model has a poor prediction power.