Predicting the Best Network TV Episode/Script

The Office

Business Analytics 780
4/28/2020

Gary Capriotti, Malvika Ravindra,

Table of Contents

Problem Description

Scope

In the midst of the streaming wars, network television is finding the lifespan of it's shows extended indefinitely. The best example of this is *The Office*, which has led all other content in streams on Netflix over the past few years. With this in mind, the team will be conducting an analysis on which components of an episode have the largest effect on its IMDb rating. At the conclusion of the analysis, the team hopes to be able to pinpoint what aspects of a television episode led to a higher rating.

Background

- 9 Seasons
- 201 episodes
- 42 Emmy Nominations
- Most watched show on Netflix in 2018 and 2019
- 45.8 billions minutes watched in 2019
- 3% of total user minutes in 2019
- NBCUniversal paid $500 million to bring the sitcom back to its own streaming platform

https://www.washingtonpost.com/arts-entertainment/2019/06/26/office-was-always-popular-netflix-made-it-phenomenon/

Approach

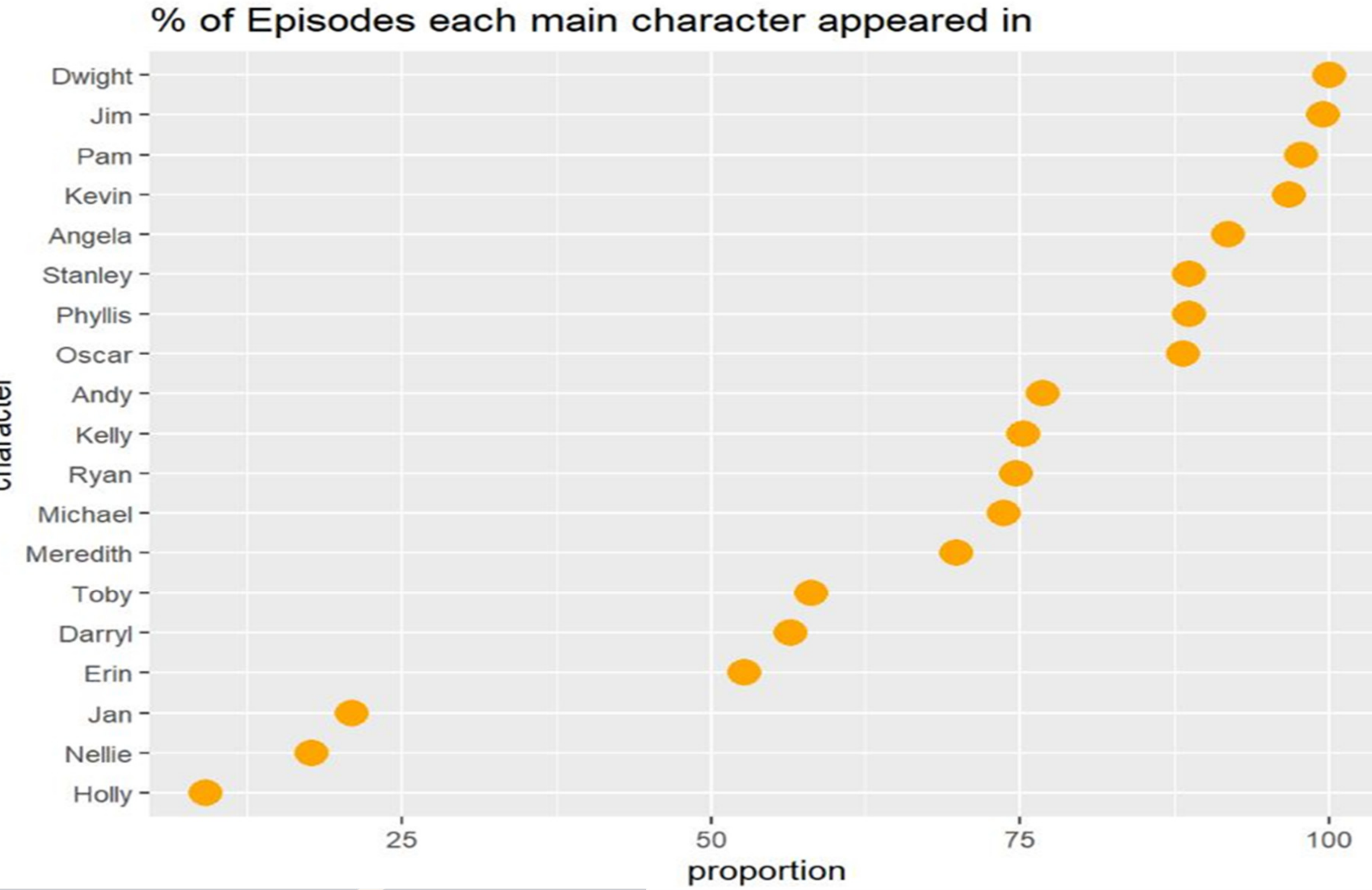
Exploratory Data Analysis

We obtained our data from the ‘schrute’ package in R and began exploring the variables the package contained. Each row in the data contains one single line spoken through seasons 1 to 9, along with different variables such as the episode number, speaker, writers for the episode, etc. Below is a glimpse of the data we obtained.

	index	season	episode	episode_name	director	writer	character	text	text_
1	1	1	1	Pilot	Ken Kwapis	Ricky Gervais;Stephen Merchant;Greg Daniels	Michael	All right Jim. Your quarterlies look very good. How are thing...	All ri
2	2	1	1	Pilot	Ken Kwapis	Ricky Gervais;Stephen Merchant;Greg Daniels	Jim	Oh, I told you. I couldn't close it. So...	Oh, I
3	3	1	1	Pilot	Ken Kwapis	Ricky Gervais;Stephen Merchant;Greg Daniels	Michael	So you've come to the master for guidance? Is this what yo...	So yo
4	4	1	1	Pilot	Ken Kwapis	Ricky Gervais;Stephen Merchant;Greg Daniels	Jim	Actually, you called me in here, but yeah.	Actu

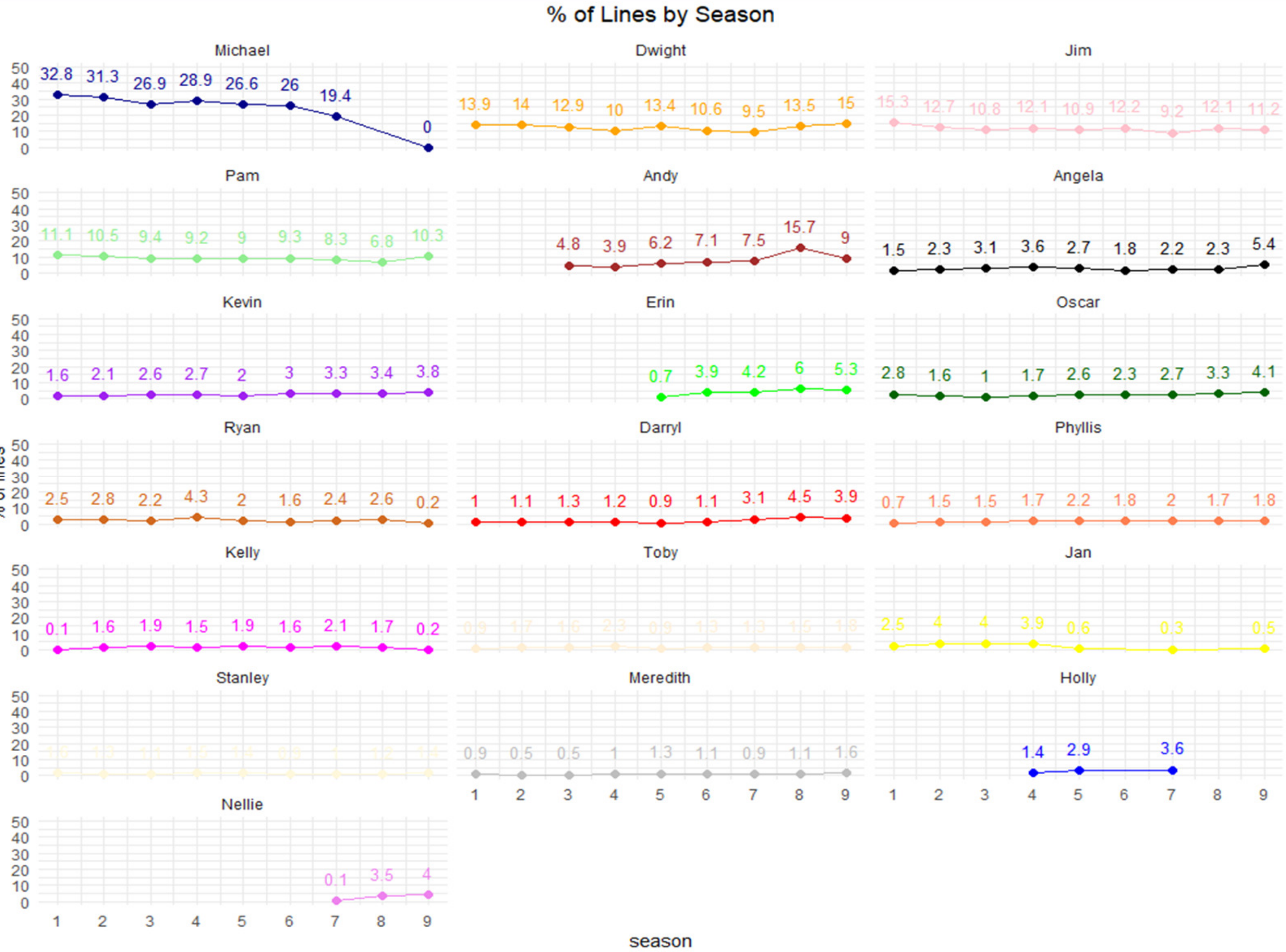
	character	text	text_w_direction	imdb_rating	total_votes	air_date
hant;Greg Daniels	Michael	All right Jim. Your quarterlies look very good. How are thing...	All right Jim. Your quarterlies look very good. How are thing...	7.6	3706	2005-03-24
hant;Greg Daniels	Jim	Oh, I told you. I couldn't close it. So...	Oh, I told you. I couldn't close it. So...	7.6	3706	2005-03-24
hant;Greg Daniels	Michael	So you've come to the master for guidance? Is this what yo...	So you've come to the master for guidance? Is this what yo...	7.6	3706	2005-03-24
hant;Greg Daniels	Jim	Actually, you called me in here, but yeah.	Actually, you called me in here, but yeah.	7.6	3706	2005-03-24

Exploratory Data Analysis



Dwight and Jim were the only characters who appeared in every single episode throughout the series. Jan was in the series for 6 seasons but did not appear in many episodes.

Exploratory Data Analysis



The proportion of lines for all the main characters throughout the seasons remained approximately consistent. Dwight and Jim had almost the same amount of lines and comparatively higher than other characters. Michael had the most number of lines in every season until he left the show after season 7.

Data Cleaning and feature engineering

- Data Cleaning
- Search and Elimination of Outliers in IMDb ratings
- Scaling Number of Voters
- One-Hot-Encoding for Writers and Directors

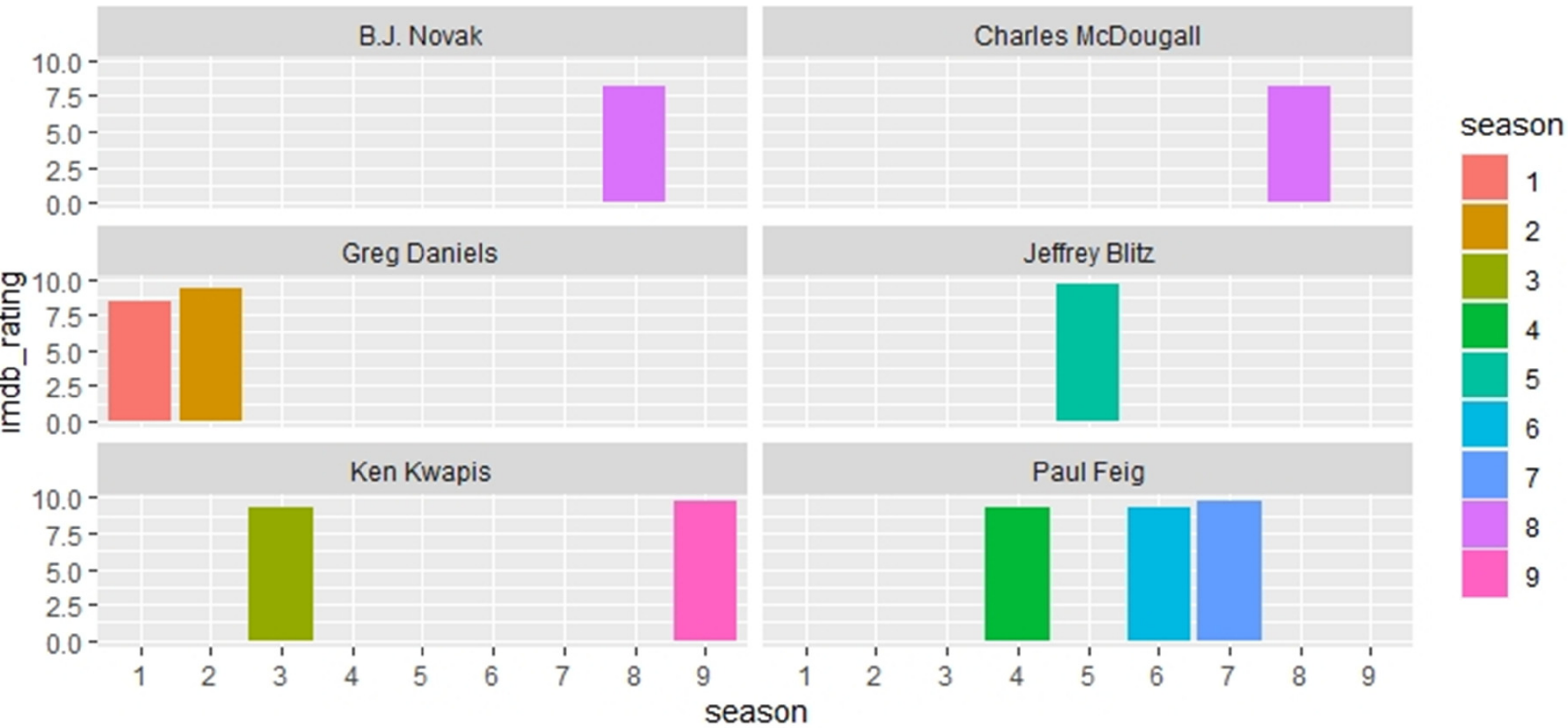
Model Creation

- Identified 132 indicator variables across directors, writers, characters
- Identified 4 continuous variables with sentiment analysis and imdb participation
- Ran through 72 different training iterations of a neural net
- Reviewed residuals and training results to select best possible parameters
- Used MSE, RMSE, and MAPE scores for statistical comparison

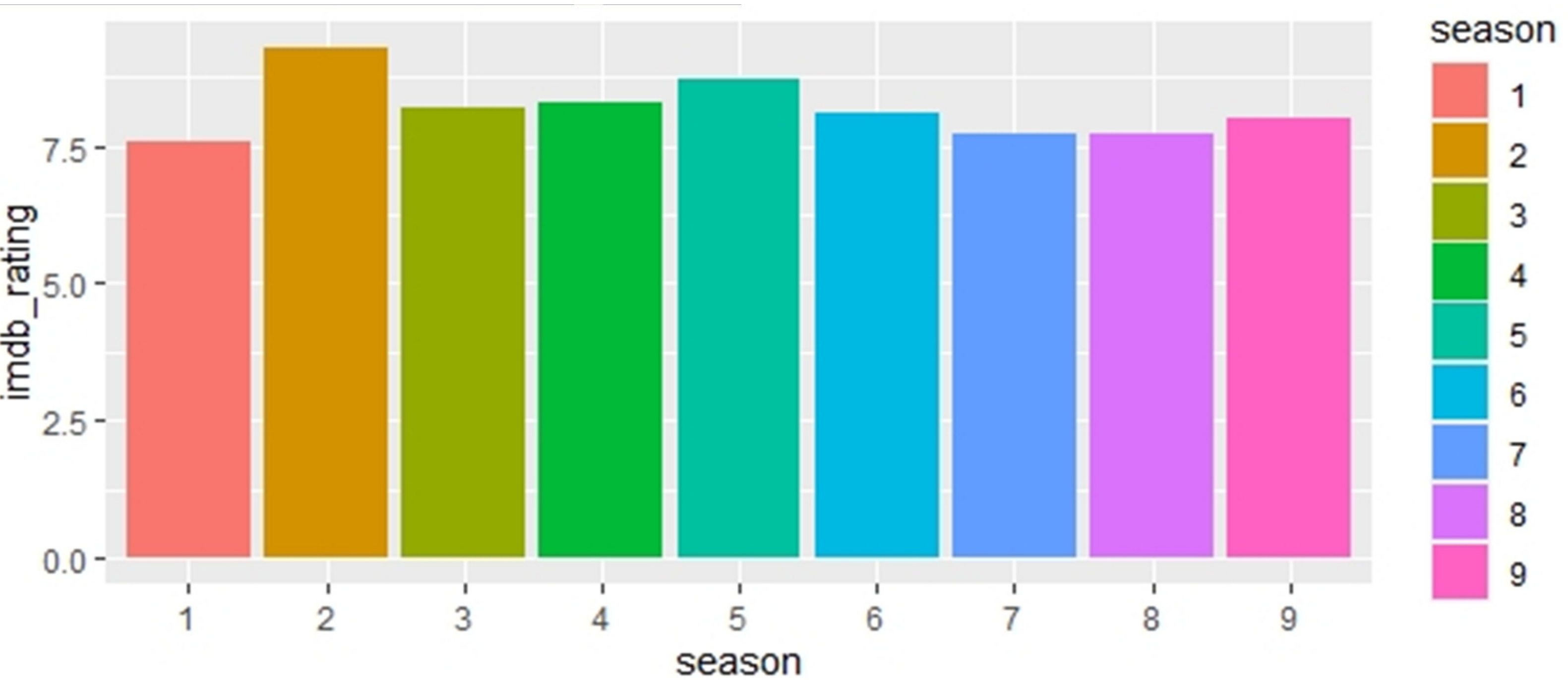
Results

Visualizations

Director with highest IMDb rating for each season

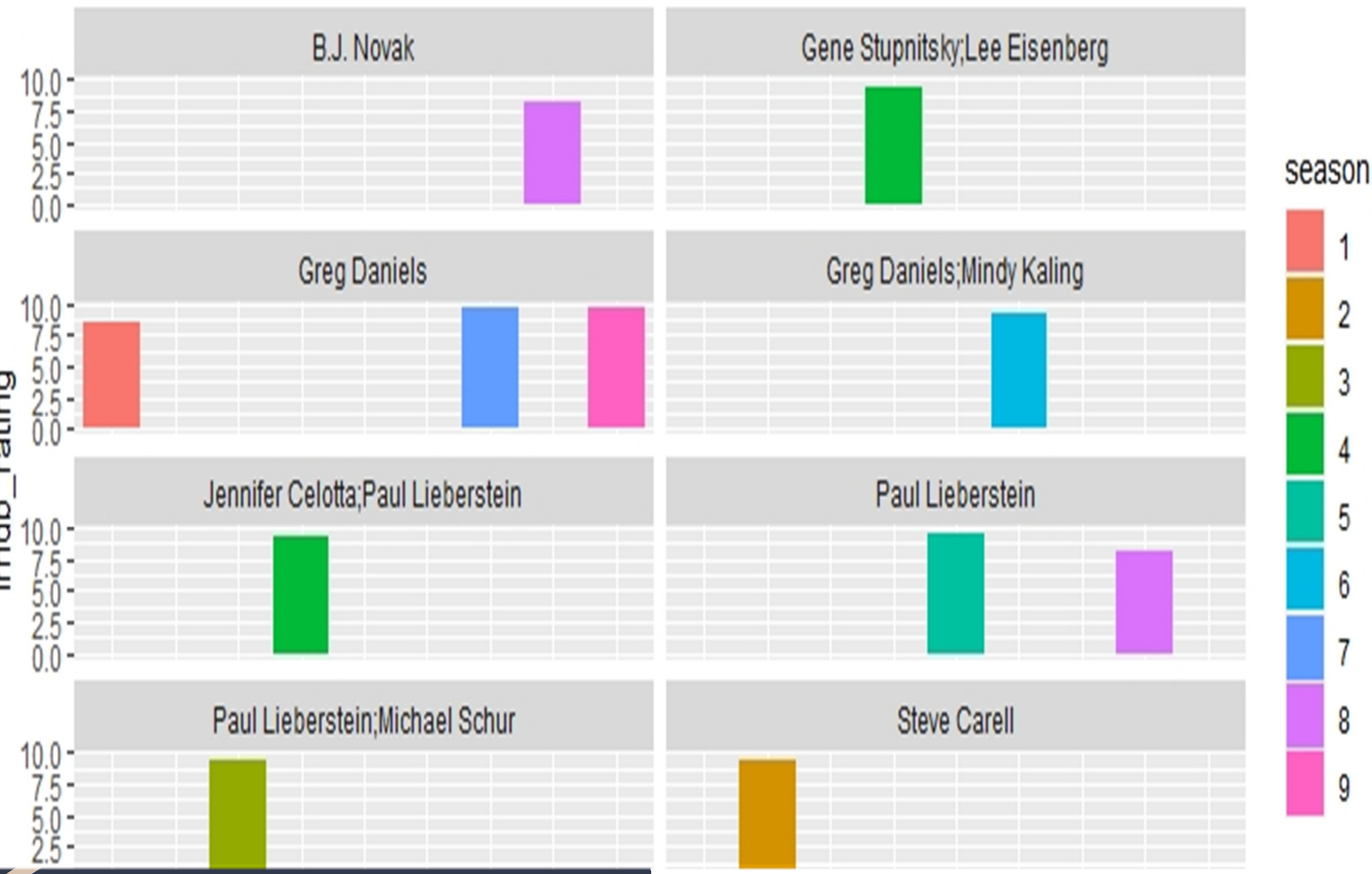


Average IMDb rating for each season



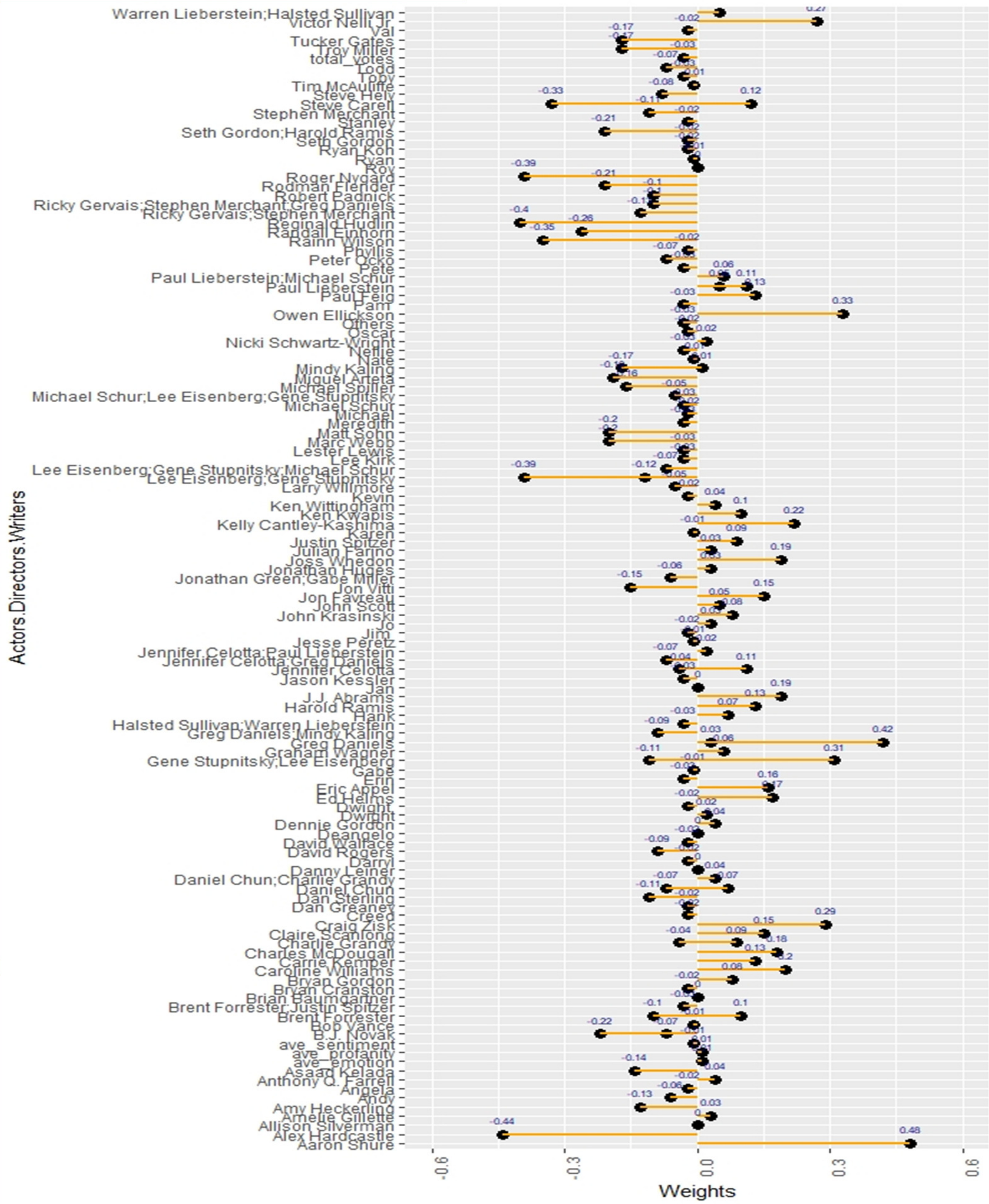
Visualizations

Writers with highest imdb rating for each season



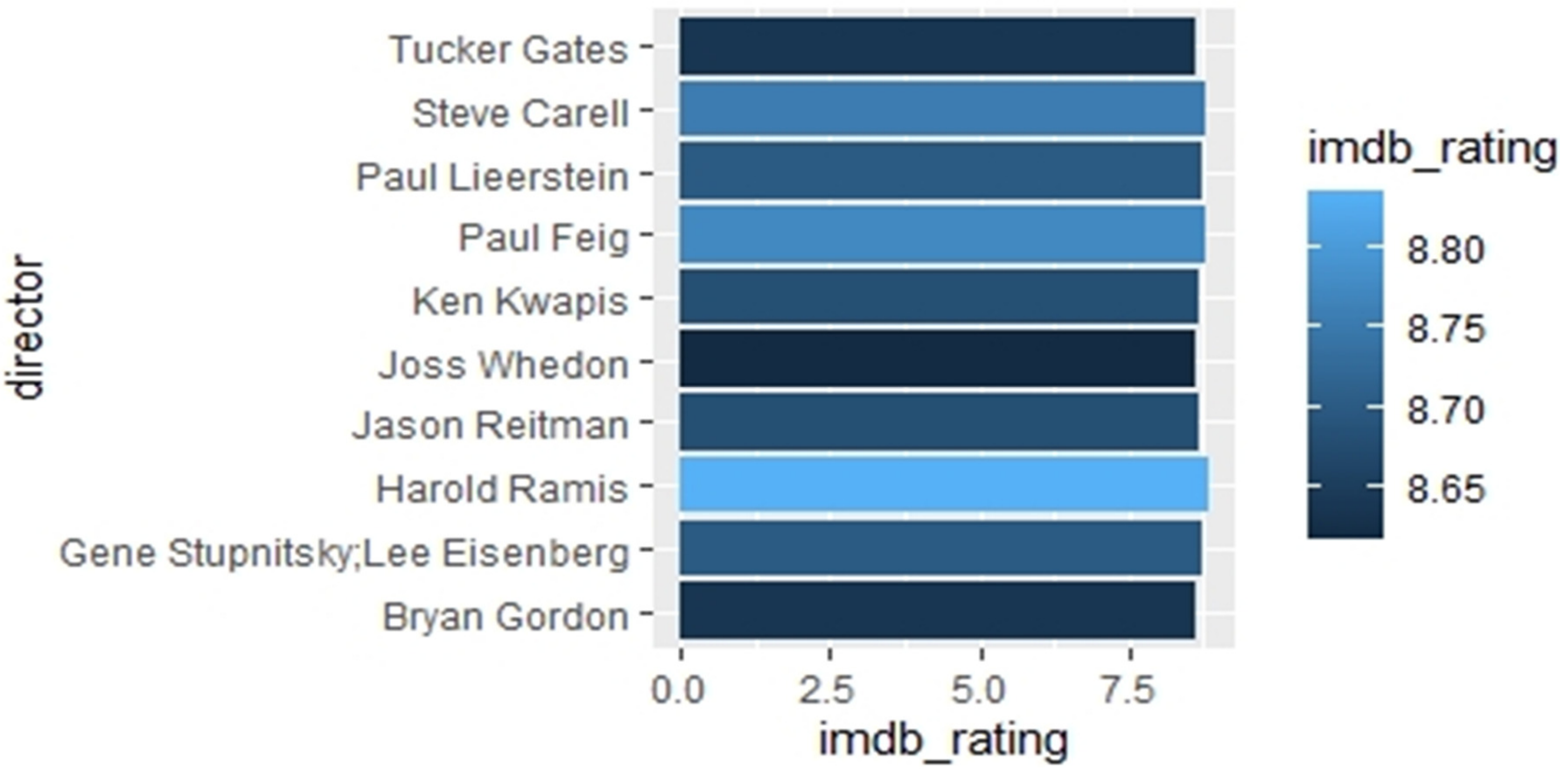
Visualizations

Plot of the weights of all the individual factors

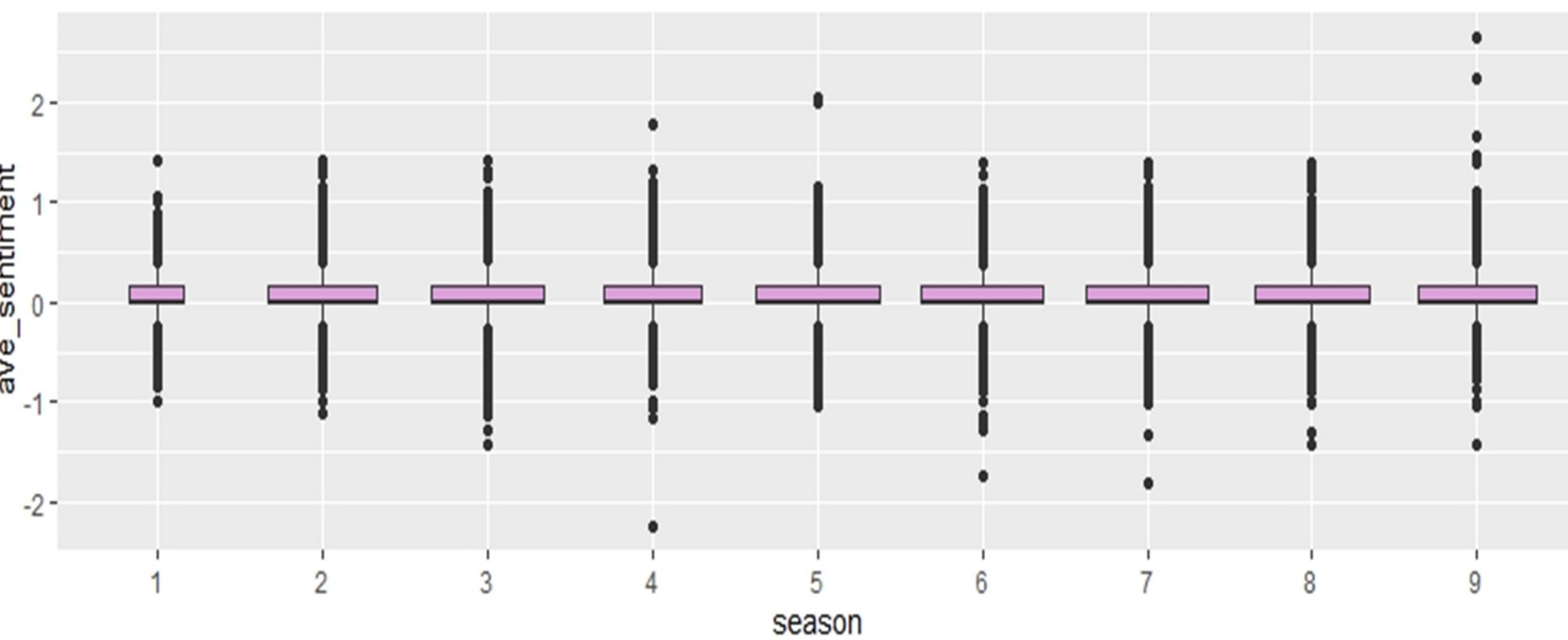


Visualizations

Top 10 directors by imdb ratings



Average sentiment score of the episode content for each season



Predictions

Hidden Nodes = 4

Decay Rate = .01

Max Iterations = 500

MAPE = 1.578

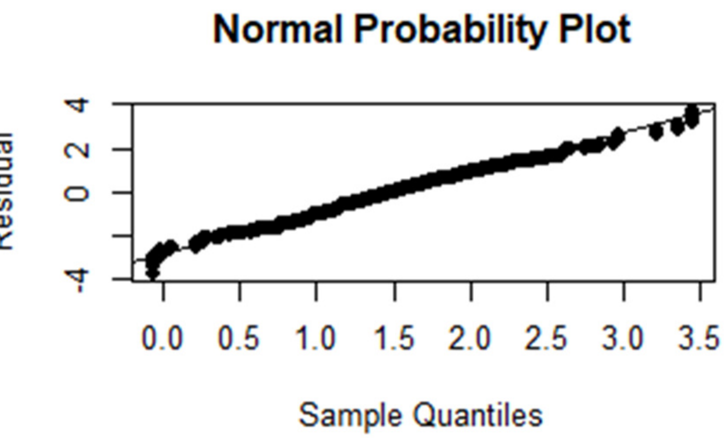
Roughly 98.5% prediction accuracy

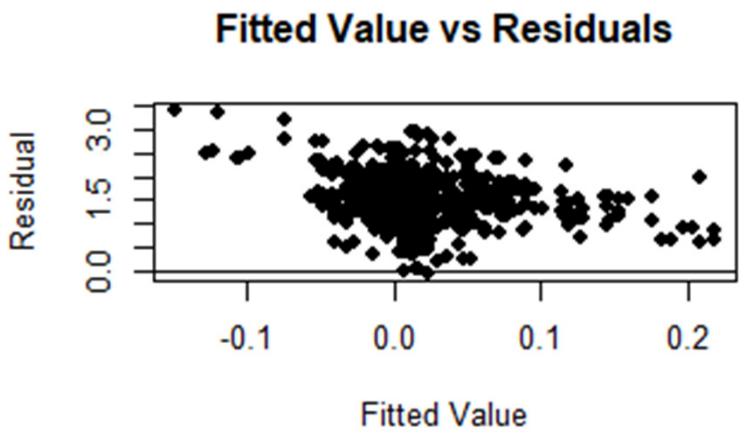
Predictions

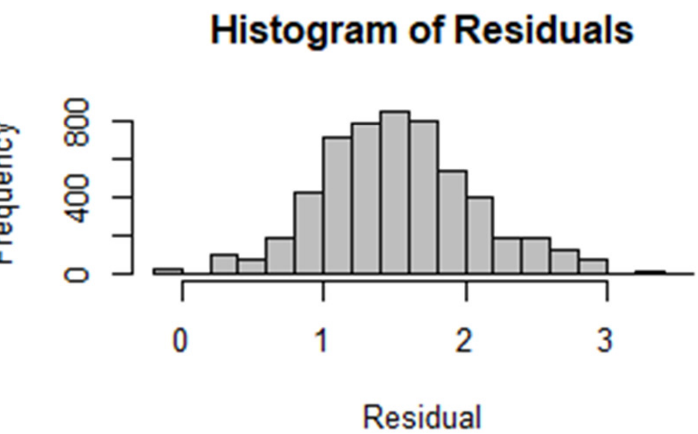
NPP plot and Histogram shows normality

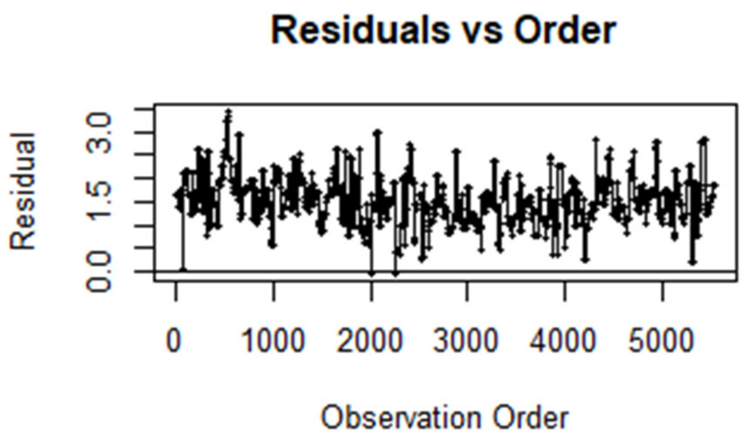
Fitted Values plot indicates non constant variance

Residuals plot indicates independence





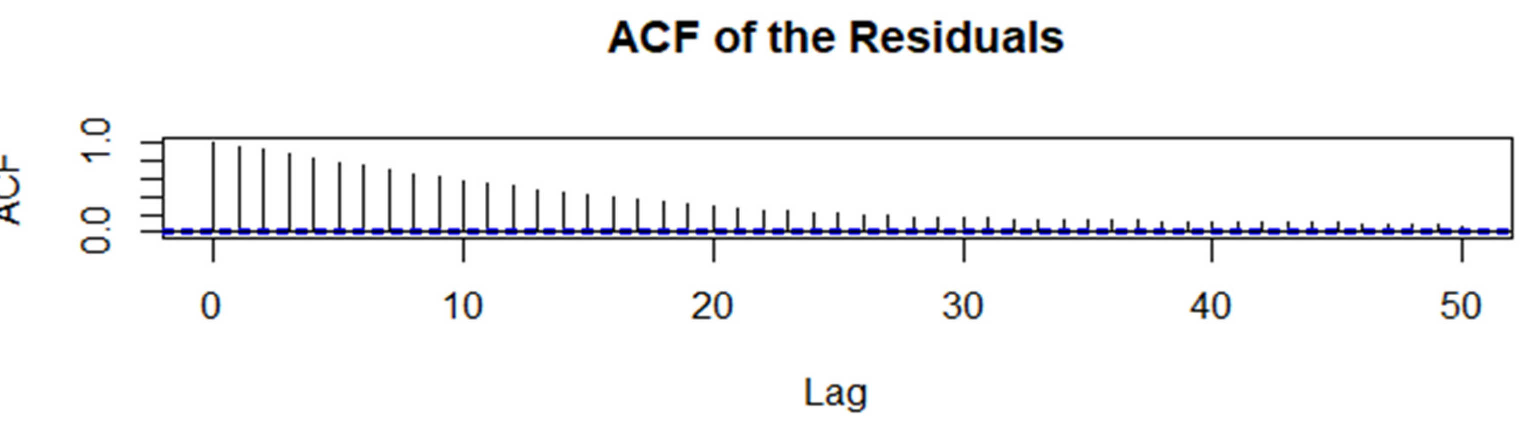


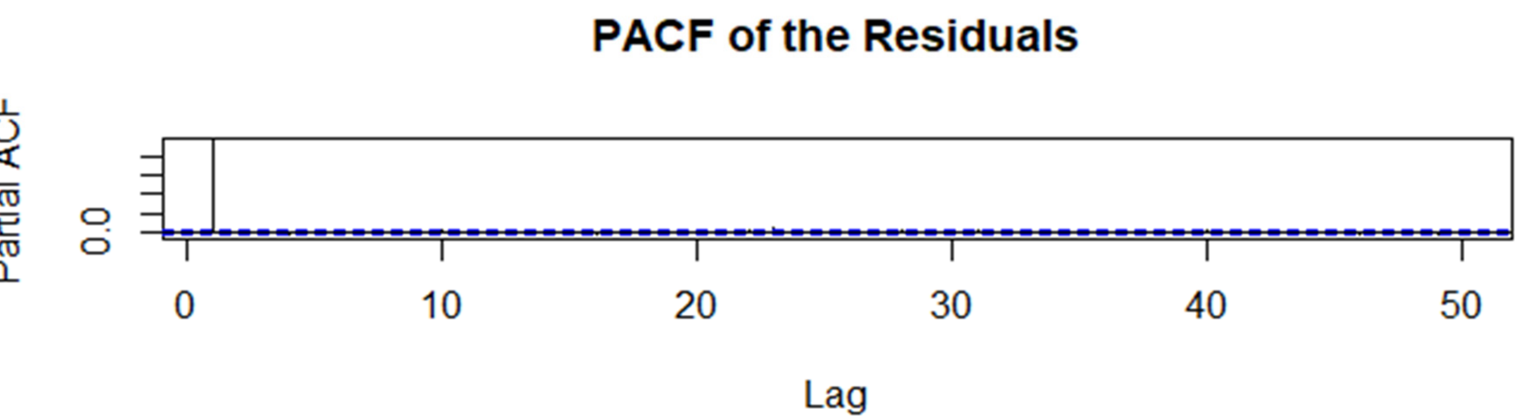


Predictions

ACF plot shows autocorrelation in the 20-30 observation range, in line with the number of episodes per season

PACF plots indicates no autocorrelation among the residuals





Predictions

Directors (Top/Bottom 5)

lex Hardcastle	-0.441322469
eginald Hudlin	-0.398002112
oger Nygard	-0.387600024
ainn Wilson	-0.348154877
teve Carell	-0.325539262
elly Cantley-Kashima	0.216033348
ictor Nelli Jr.	0.266022689
raig Zisk	0.285320521
ene Stupnitsky; ee Eisenberg	0.310961825
reg Daniels	0.417658113

Characters directed numerous episodes, and Rainn Wilson/Steve Carell were ranked very negatively

Greg Daniels as show runner and executive producer outperformed all directors, to no surprise

Writers (Top/Bottom 5)

ee Eisenberg; ene Stupnitsky	-0.391930976
on Vitti	-0.147962361
icky Gervais;Stephen erchant	-0.132360308
an Sterling	-0.110672266
icky Gervais; tephen Merchant; reg Daniels	-0.101474094
teve Carell	0.124592588
arrie Kemper	0.132313109
aroline Williams	0.196083788
wen Ellickson	0.326420723
aron Shure	0.480221537

Gene Stupnitsky and Lee Eisenberg were great directors but terrible writers

Original creators Ricky Gervais and Stephen Merchant struggled for success as Season 1 underperformed

Predictions

Characters

Todd	-0.067135767
Andy	-0.062385538
Nellie	-0.032361569
Toby	-0.032123781
Erin	-0.02916079
Others	-0.027900697
Pete	-0.0262674
Pam	-0.025496331
Meredith	-0.025032227
David Wallace	-0.024116274
Val	-0.02359773
Stanley	-0.023401795
Phyllis	-0.023256742
Darryl	-0.020730325
Michael	-0.020686071
Jim	-0.018599409
Kevin	-0.018157687
Angela	-0.017987574
Oscar	-0.016159334
Creed	-0.015725039
Bob Vance	-0.011286989
Ryan	-0.010593212
Karen	-0.009166396
Nate	-0.007099576
Gabe	-0.005150321
Roy	-0.002900876
Deangelo	-0.001614506
Jan	0.001842643
Dwight	0.016947901
Jo	0.030726349
Hank	0.065364894

Interesting to note that only Jan, Dwight, Jo, and Hank had a positive effect on IMDb rating

In general, lesser shown characters were positive while more common characters were slightly negative

Overall, characters had very little effect on IMDb rating compared to writers and directors

Conclusions

Episode ratings seem to be mostly story driven, as poor/strong writers and directors have heightened impacts on results

Sentiment, emotion, and profanity have little to no impact on IMDb ratings

Characters have -.0157 of an impact on ratings (essentially 0)

Of all 4 categories, writers had the most impact on the imdb rating.

Appendix

Prediction Model

```
#LIBRARIES______________________________________________________________________________
ibrary(fpp)
ibrary(scatterplot3d)
ibrary(nnet)
ibrary(lubridate)
ibrary(forecast)
ibrary(plyr)
ibrary(tidyr)
ibrary(corrplot)
ibrary(schrute)
ibrary(sentimentr)
ibrary(readr)
ibrary(stringr)

estdata <- (schrute::theoffice)
write.csv(testdata,"ogdata.csv")
#PULLING IN THE DATA ____________________________________________________________________
rawdata <- read.csv("OfficeDataConditioned.csv")
data<- data.frame(rawdata)
rawtestdata <- read.csv("OfficeTestData.csv")
estdata<- data.frame(rawtestdata)
#sentimenttext <- read_lines("Office.csv")
#Sentiment_Score <- data.frame(sentiment_by(sentimenttext, by=NULL))
#Sentiment_Score <- Sentiment_Score[-c(1:3)]
#Emotion_Score <- data.frame(emotion_by(sentimenttext, by=NULL))
#Emotion_Score <- Emotion_Score[-c(1,3,5)]
#Profanity_Score <- data.frame(profanity_by(sentimenttext, by=NULL))
#Profanity_Score <- Profanity_Score[-c(1,2,4)]
#data <- cbind(data, Sentiment_Score)
#data <- cbind(data, Emotion_Score)
#data <- cbind(data, Profanity_Score)
#write.csv(Emotion_Score, "Emotion_Score.csv")
#write.csv(Sentiment_Score, "Sentiment_Score.csv")
#write.csv(Profanity_Score, "Profanity_Score.csv")
#INDICATORS & VARIABLES__________________________________________________________________
CharacterIndicator<-class.ind(data$character_index)
DirectorIndicator<-class.ind(data$director_index)
WriterIndicator<-class.ind(data$writer_index)
allIndicators<-data.frame(DirectorIndicator[,1:53],WriterIndicator[,1:47],CharacterIndicator[,1:31])

#NORMALIZATION FUNCTION
normalizefunction = function(x) {
  num = x - min(x)
  denom = max(x) - min(x)
  return(num/denom)
}
#UNNORMALIZATION FUNCTION
unnormalizefunction = function(x,min,max) {
  return(x*(max-min)+min)
}
#CONTINUOUS DATA
minRating<- 6.7
maxRating<- 9.7
allContinuous = data.frame(data$total_votes, data$ave_sentiment, data$ave_emotion, data$ave_profanity,data$imdb_rating)
colnames(allContinuous) = c("total_votes","ave_sentiment","ave_emotion","ave_profanity","imdb_rating")
```

Prediction Model

```
normContinuous = data.frame(sapply(allContinuous, normalizefunction))
#SELECTING TRAINING DATA
allDataReady<-data.frame(data$season.episode,allIndicators,normContinuous)
training_sample_size <- floor(0.80 * nrow(allDataReady))
set.seed(1234567)
#TRAINING DATA LIST
train_ind <- sample(seq_len(nrow(allDataReady)), size = training_sample_size)
train <- allDataReady[train_ind, ]
validation <- allDataReady[-train_ind, ]
predictorsTrain <-train[,2:136]
targetTrain<-train[,137]
predictorsValidation<-validation[,2:136]
targetValidation<-validation[,137]
trainingResults<-data.frame(HiddenNodes=numeric(),Decay=numeric(),Iterations=numeric(),MSE_Fit=numeric(),
MSE_Validation=numeric(),RMSE_Validation_Unnorm=numeric(),MAPE=numeric())
names(trainingResults)=c("HiddenNodes","Decay","Iterations","MSE","MSE_Validation","RMSE_Validation_Unnorm","MAPE")


#TRAINING NEURAL NET_____________________________________________________________
for(h in c(1:6)){
  for(d in c(0.01,0.05,0.1)){
    for(maxIter in c(3:6)){
      maxiter=maxIter*100
      nnetFit<-nnet(predictorsTrain, # the regressor variables
                    targetTrain, #what you are trying to predict
                    size=h, #number of hidden nodes
                    decay = d, #gives a penalty for large weights
                    linout = TRUE, #says you want a linear output (as oppposed to a classification output)
                    trace=FALSE, #reduces amount of output printed to screen
                    maxit = maxiter, # increases max iterations to 500 from default of 100
                    MaxNWts = h*(ncol(predictorsTrain)+1)+h+1) #says you can have one weight for each input + an additional intercept term
      #CALCULATE ERROR AND RESULTS
      MSE_Fit<-mean((nnetFit$residuals)^2)
      #FORECASTING VALIDATION SET
      predictions<-predict(nnetFit,predictorsValidation)
      MSE_Validation <-(mean(predictions - targetValidation)^2)
      #RMSE ON ORIGINAL SCALE
      unnormalizedPredictions = unnormalizefunction(x=predictions,min=minRating,max=maxRating)
      unnormalizedTargetValidation = unnormalizefunction(x=targetValidation,min=minRating,max=maxRating)
      RMSE_Validation_Unnorm <-sqrt(mean((unnormalizedPredictions-unnormalizedTargetValidation)^2))
      MAPE <-mean(abs(unnormalizedTargetValidation - unnormalizedPredictions)/unnormalizedTargetValidation)*100
      results<-data.frame(h,d,maxIter,MSE_Fit,MSE_Validation,RMSE_Validation_Unnorm,MAPE)
      print(results)
      names(results)=c("HiddenNodes","Decay","Iterations","MSE_Fit","MSE_Validation","RMSE_Validation_Unnorm","MAPE")
      trainingResults<-rbind(trainingResults,results)
    }}}
write.csv(trainingResults,file="trainingResultsfinal.csv")
#CALCULATING RESIDUALS
Residuals<-data.frame(validation$data.index,unnormalizedTargetValidation-unnormalizedPredictions)
colnames(Residuals)=c("Time","Residuals")
Res<-as.vector(unnormalizedTargetValidation-unnormalizedPredictions)
#ACF/PACF OF RESIDUALS
par(mfrow=c(2,1))
acf(Res,lag.max=50,type="correlation",main="ACF of the Residuals",na.action = na.pass)
acf(Res,lag.max=50, type = "partial",main="PACF of the Residuals",na.action = na.pass)
```

Prediction Model

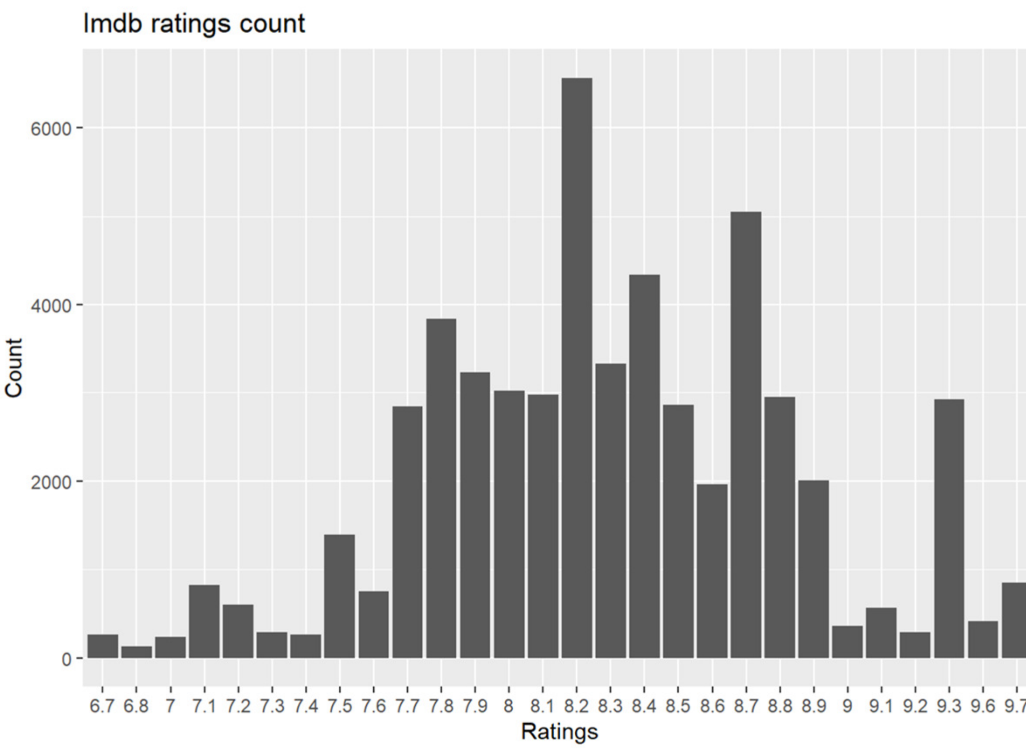
```
#CONTINUOUS VARIABLES
allContinuousForecast = data.frame(testdata$total_votes, testdata$ave_sentiment, testdata$ave_emotion, testdata$ave_profanity)
colnames(allContinuous) = c("total_votes","ave_sentiment","ave_emotion","ave_profanity")
sum(is.na(allContinuousForecast))
normContinuousForecast = data.frame(sapply(allContinuousForecast, normalizefunction))
allDataReadyForecast<-data.frame(testdata$season.episode,allIndicatorsForecast,normContinuousForecast)

#MAKING FORECASTS
testPredictions <-predict(nnetFinalFit,allDataReadyForecast[,2:33])
forecasts <-data.frame(season.episode=allDataReadyForecast$testdata.season.episode,testPredictions)
testPredictionsUnnorm<-unnormalizefunction(x=testPredictions,min=minRating,max=maxRating)
forecastOriginalScale<-data.frame(Date=allDataReadyForecast$testdata.season.episode,testPredictionsUnnorm)


#RESIDUALS____________________________________________________________________________________________________________________________________________________________
#CALCULATING RESIDUALS
Residuals<-data.frame(testdata$season.episode,testdata$imdb_rating-forecastOriginalScale$testPredictionsUnnorm)
colnames(Residuals)=c("Season.Episode","Residuals")
Res<-as.vector(testdata$imdb_rating-forecastOriginalScale$testPredictionsUnnorm)
#ACF/PACF OF RESIDUALS
par(mfrow=c(2,1))
acf(Res,lag.max=50,type="correlation",main="ACF of the Residuals",na.action = na.pass)
acf(Res,lag.max=50, type = "partial",main="PACF of the Residuals",na.action = na.pass)
#RESIDUAL PLOTS
par(mfrow=c(2,2), oma=c(0,0,0,0))
qqnorm(Res,datax=TRUE,pch=16,xlab="Residual",main="Normal Probability Plot")
qqline(Res,datax=TRUE)
plot(testPredictions,Res,pch=16,xlab="Fitted Value",ylab="Residual", main="Fitted Value vs Residuals")
abline(h=0)
hist(Res,col="gray",xlab="Residual", main="Histogram of Residuals")
plot(Res,type="l",xlab="Observation Order",ylab="Residual", main="Residuals vs Order")
points(Res,pch=16,cex=0.5)
abline(h=0)

#h    d maxIter    MSE_Fit MSE_Validation RMSE_Validation_Unnorm     MAPE
#1 4 0.01      500 0.01998326   1.705192e-06              0.4367599 1.578055

forecastOriginalScale = cbind(forecastOriginalScale)
write.csv(forecastOriginalScale,file="forecast.csv")
```

EDA

```
mydata <- schrute::theoffice
dplyr::glimpse(mydata)
```

```
## Observations: 55,130
## Variables: 12
## $ index            <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,...
## $ season           <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ episode          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ episode_name     <chr> "Pilot", "Pilot", "Pilot", "Pilot", "Pilot", "Pilo...
## $ director         <chr> "Ken Kwapis", "Ken Kwapis", "Ken Kwapis", "Ken Kwa...
## $ writer           <chr> "Ricky Gervais;Stephen Merchant;Greg Daniels", "Ri...
## $ character        <chr> "Michael", "Jim", "Michael", "Jim", "Michael", "Mi...
## $ text             <chr> "All right Jim. Your quarterlies look very good. H...
## $ text_w_direction <chr> "All right Jim. Your quarterlies look very good. H...
## $ imdb_rating      <dbl> 7.6, 7.6, 7.6, 7.6, 7.6, 7.6, 7.6, 7.6, 7.6, 7.6, ...
## $ total_votes      <int> 3706, 3706, 3706, 3706, 3706, 3706, 3706, 3706, 37...
## $ air_date         <fct> 2005-03-24, 2005-03-24, 2005-03-24, 2005-03-24, 20...
```

```
mydata %>%
  group_by(imdb_rating)%>%
  count()%>%
  ggplot()+
  geom_bar(mapping =
aes(as.character(imdb_rating), n), stat =
'identity')+
  ggtitle("Imdb ratings count")+
  xlab("Ratings")+
  ylab("Count")
```



EDA

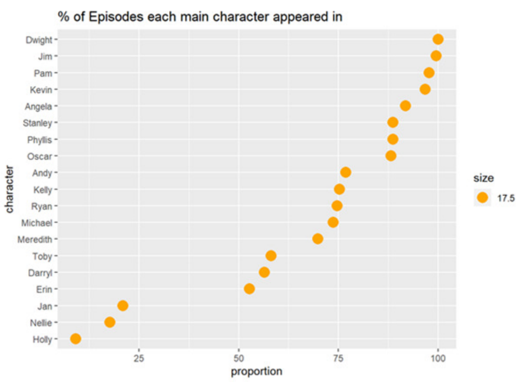
```
# proportion of episodes each character was in
episode_proportion <- mydata %>%
  unite(season_ep, season, episode, remove = FALSE) %>%
  group_by(character) %>%
  summarise(num_episodes = n_distinct(season_ep)) %>%
  mutate(proportion = round((num_episodes / total_episodes) * 100, 1)) %>%
  arrange(desc(num_episodes))
episode_proportion
```

```
## # A tibble: 773 x 3
##    character num_episodes proportion
##    <chr>            <int>      <dbl>
##  1 Dwight             186      100
##  2 Jim                185       99.5
##  3 Pam                182       97.8
##  4 Kevin              180       96.8
##  5 Angela             171       91.9
##  6 Phyllis            165       88.7
##  7 Stanley            165       88.7
##  8 Oscar              164       88.2
##  9 Andy               143       76.9
## 10 Kelly              140       75.3
## # ... with 763 more rows
```

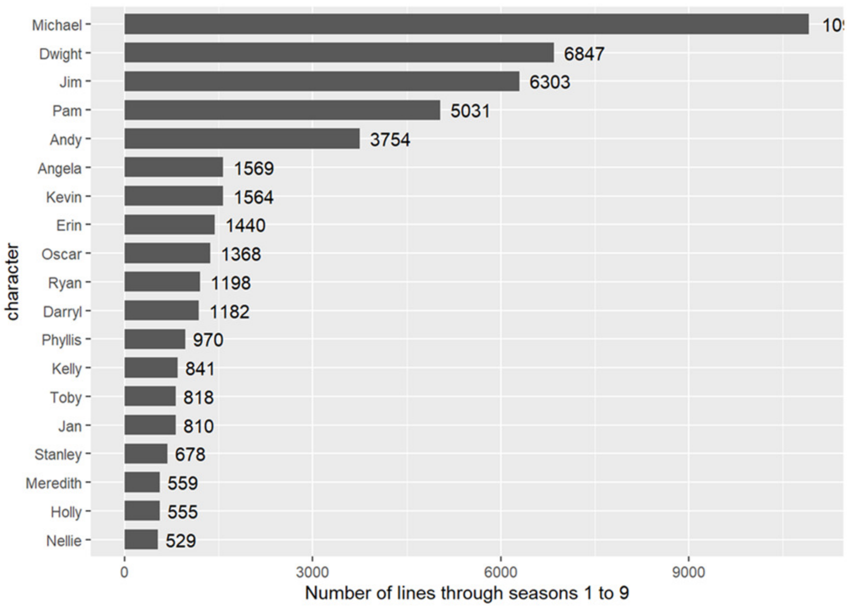
```
line_proportion <- mydata %>%
  count(character) %>%
  mutate(proportion = round((n / sum(n)) * 100, 1)) %>%
  arrange(desc(n))

# define main characters based on line proportion
main_characters <- factor(line_proportion %>%
                            filter(proportion >= 1) %>%
                            pull(character) %>%
                            fct_inorder()
                          )

main_characters_episodes <- episode_proportion[episode_proportion$character %in% main_characters, ]
```

EDA

```
plot1 <- ggplot(main_characters_episodes, aes(x=proportion,y=character, laber=proportion)) +
geom_point(aes(size=17.5),colour="orange") + ggtitle("% of Episodes each main character appeared in")
```

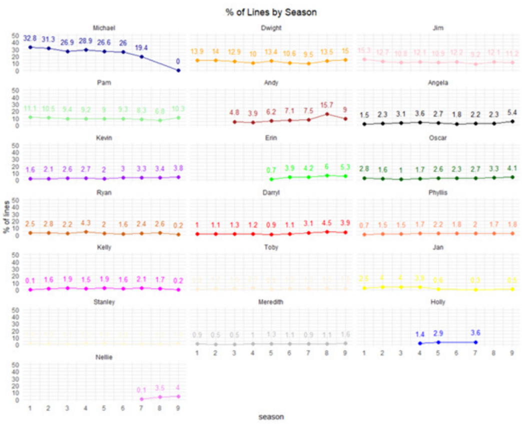



```
#who had the most lines
lines_characters <- mydata %>%
  count(character) %>%
  arrange(desc(n))
lines_characters <- lines_characters[lines_characters$character %in% main_characters,]
lines_characters$character <- factor(lines_characters$character, levels =
lines_characters$character[order(lines_characters$n)])
lines_characters %>% ggplot(aes(x = character, y = n, label =n))+
  geom_col( width = 0.7)  + coord_flip() +ylab("Number of lines through seasons 1 to 9") +
geom_text(aes(label=n),position=position_dodge(width=0.9), hjust=-0.25)
```



EDA

```
line_proportion_by_season <- mydata %>%
  group_by(season) %>%
  count(character) %>%
  mutate(proportion = round((n / sum(n)) * 100, 1)) %>%
  arrange(season, desc(proportion))

office_colors <-
c("brown","black","red","orange","green","blue","yellow","pink","magenta","purple","grey","darkblue","violet","darkgreen",
"lightgreen","coral", "chocolate", "cornsilk", "papayawhip", "blanchedalmond")

line_proportion_over_time <- line_proportion_by_season %>%
  filter(character %in% main_characters) %>%
  ggplot(aes(x = season, y = proportion, color = character, label = proportion)) +
  geom_point(size = 2) +
  geom_line() +
  scale_x_continuous(breaks = seq(1, 9, 1)) +
  theme_minimal() +
  theme(legend.position = "none") +
  labs(y = "% of lines",
       title = "% of Lines by Season") +
  theme(plot.title = element_text(hjust = 0.5)) +
  facet_wrap(~ factor(str_to_title(character), levels = str_to_title(main_characters)), ncol = 3) +
  geom_text(vjust = -1.2, size = 3.5) +
  ylim(0, 50) +
  scale_color_manual(values = office_colors)
Line_proportion_over_time
```

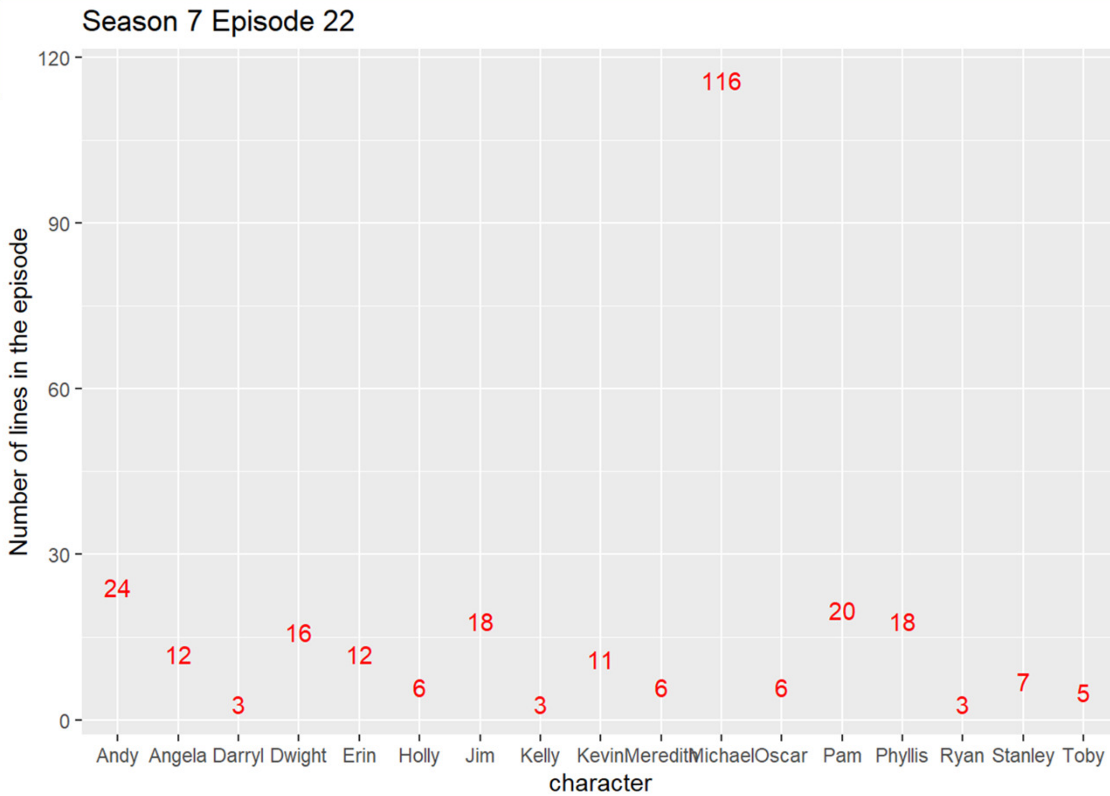


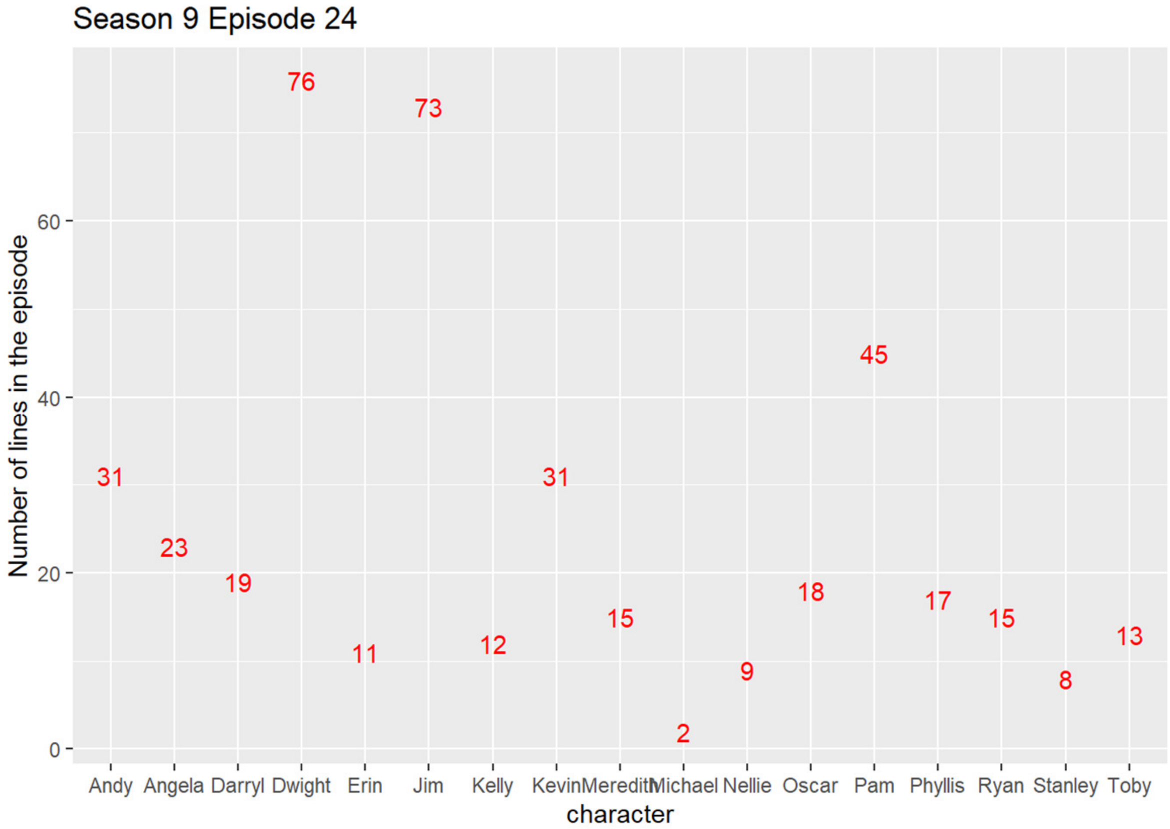
EDA

```
highest_rating <- mydata %>%
  filter(imdb_rating == max(imdb_rating)) %>%
  group_by(season)

cat("Highest imdb rating for the show:",unique(highest_rating$imdb_rating))

## Highest imdb rating for the show: 9.7

highest_rating_characters <- unique(highest_rating$character)

highest_rating_characters1 <- highest_rating_characters[highest_rating_characters%in% main_characters]
#highest_rating_characters1

highest_rating_episodes <- highest_rating %>%
  unite(season_ep, season, episode, remove = FALSE)

cat("Episodes with highest imdb rating of 9.7:",unique(highest_rating_episodes$season_ep))

## Episodes with highest imdb rating of 9.7: 7_22 9_24

highest_rating_lines <- highest_rating_episodes %>% group_by(season_ep) %>% count(character) %>%
arrange(season_ep,desc(n))

highest_rating_lines %>% filter(season_ep == "7_22") %>% filter(character %in% main_characters) %>%
  ggplot(aes(x=character,y=n,label=n)) + geom_text(aes(label=n),position=position_dodge(width=0.9),colour="red")
+ylab("Number of lines in the episode") +ggtitle("Season 7 Episode 22")
```

EDA

Number of lines by each character for the highest imdb rating(9.7) episodes:





Code for Plots

```
library(ggplot2)

library(dplyr)

library(schrute)

mydata <- schrute::theoffice

mydata$season <- as.character(mydata$season)

ggplot(conditionedData,aes(x = season,y = ave_sentiment)) + geom_boxplot(varwidth = T,fill
="plum")

bydirector <-  aggregate(imdb_rating ~ director,data = mydata,FUN = function(x) c(m =
mean(x)))%>% arrange(desc(imdb_rating)) %>% top_n(10)

bydirector %>% ggplot() + geom_bar(aes(x=director,y = imdb_rating,fill = imdb_rating),stat =
"identity") + coord_flip()

byimdb_rating <- mydata %>% select(season,director,writer,imdb_rating)  %>% group_by(season)
%>% arrange(desc(imdb_rating)) %>% top_n(1)

# imdb_rating for the season

mydata %>% ggplot() + geom_bar(mapping=aes(x=season,y=imdb_rating,fill=season),stat =
"summary")

#SEASON_DIRECTORAVERAGE_IMDB

byimdb_rating %>% ggplot() + geom_bar(mapping=aes(x=season,y=imdb_rating,fill=season),stat =
"summary")+ facet_wrap( ~ director, nrow = 4)

#SEASON_WRITERAVERAGE_IMDB

byimdb_rating %>% ggplot() + geom_bar(mapping=aes(x=season,y=imdb_rating,fill=season),stat =
"summary")+ facet_wrap( ~ writer, nrow = 4)
```