TS Project

Halen B., Ganady A., Santana M., Shepherd W.

December 5, 2016

# 1 Introduction

Since 2014, Yelp has been hosting a competition called the Yelp Dataset Challenge. Thus far, Yelp has offered 8 rounds of this competition to the general public. Since its inception, the Yelp Dataset Challenge has given the machine learning and data science communities a useful and thorough dataset in which to apply state of the art ML algorithms and advanced data analysis to. We chose this dataset because it offers highly relevant data that is granular enough to be useful in our analysis, but not too granular as to require computationally complex routines to pre-process the data.

# 2 Initial Discussion

The dataset consists of 2.7M reviews and 649K tips by 687K users for 86K businesses. Furthermore, there are a total of 566K attributes that can be applied to a business. In addition to data that is core to Yelp's business, the dataset also includes associations between the users creating a graph network of approximately 4.2M edges. All of the data is represented in json form stored in text files and takes up about 2.5GB on disk.

The portion of the data we have chosen to focus on are the review data. We are using reviews as a proxy measure of Yelp's popularity overtime. We are able to do this because each review contains a date in 'yyyy-mm-dd' format which enables us to measure popularity down to the granularity of a day. Because yelp has been in business since 2014, grouping reviews by day gives us TODO: Figure out how many samples we would have samples to build our predictive model with; however, to improve the accuracy of our predictions, we have decided to limit the granularity of time to a month.

library(ggplot2)  
library(xts)

## Loading required package: zoo

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

library(reshape2)  
  
  
library(zoo)  
  
# Downloads the 'bc2.R' file.  
download.file("https://raw.githubusercontent.com/brent-halen/TSProject2016/master/bc2.R", destfile = "bc2.R", method = "libcurl")  
source("bc2.R")  
# Downloads 'decom1.R'  
download.file("https://raw.githubusercontent.com/brent-halen/TSProject2016/master/decom1.R",destfile="decom1.R", method="libcurl")  
source("decom1.R")  
# Downloads the 'bulkfit.R' file.  
download.file("https://raw.githubusercontent.com/brent-halen/TSProject2016/master/bulkfit.R",destfile="bulkfit.R",method="libcurl")  
source("bulkfit.R")  
  
# The following is an upgraded version of 'Bulkfit' designed to test seasonal ARIMA models as well as stationary.   
# ###WARNING###   
# This modification will cause the function to test 729 models instead of just 27. It may take a while to complete.  
bulkfit2 <- function(x,y) {  
 w <- matrix(0,nrow=729,ncol=7)  
 ii <- 0  
   
 for(i in 0:2) {  
 for(k in 0:2) {  
 for(j in 0:2) {  
 for(I in 0:2){  
 for(K in 0:2){  
 for(J in 0:2){  
 ii <- ii + 1  
 fit <- try(arima(x,order=c(i,k,j),seasonal= list(order=c(I,K,J),period=y)))  
   
 if(inherits(fit,"try-error")) {  
 w[ii,7] <- 99999   
 }  
 else {  
 w[ii,7] <- fit$aic  
 w[ii,1] <- i  
 w[ii,2] <- k   
 w[ii,3] <- j  
 w[ii,4] <- I  
 w[ii,5] <- K  
 w[ii,6] <- J  
 }  
 }  
 }  
   
 }  
 }  
 }  
 }  
   
 dimnames(w) <- list(NULL,c("ar","d","ma","seasar","seasd","seasma","AIC"))  
 xxx <- which(w[,7]==min(w[,7],na.rm=TRUE))[1]  
 return(list(res=w,min=w[xxx,]))  
   
}  
  
  
  
library(forecast)

## Loading required package: timeDate

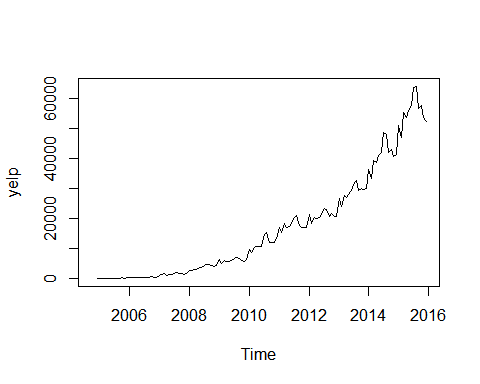
## This is forecast 7.3

library(MASS)

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:plotly':  
##   
## select

library(quadprog)  
  
# Downloads the final file from the web site.  
link <- "https://raw.githubusercontent.com/brent-halen/TSProject2016/master/reviews.csv"  
reviews <- "reviews.csv"  
download.file(link, destfile = reviews, method = "libcurl")  
  
#yelp = read.zoo(file = reviews, sep = "," , FUN = as.yearmon)  
  
yelp <- read.csv(file = reviews, header = TRUE)  
yelp <- as.data.frame(yelp)  
yelp\_forzoo <- yelp  
yelp\_forzoo$reviewDate <- as.yearmon(yelp\_forzoo$reviewDate)  
yelp <- zoo(yelp\_forzoo$reviews,yelp\_forzoo$reviewDate)  
plot.ts(yelp)

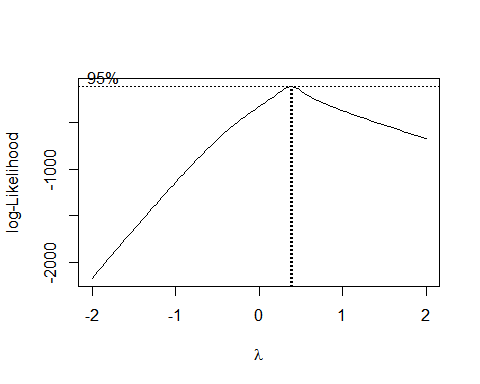


yelp.ts <- as.ts(yelp)

# 3 Model Building

First, we ran the 'bc2' function on the original data.

bc2(yelp.ts,ploty=TRUE)



## [1] 0.3838384

#bc2(yelp.zoo,ploty=TRUE)  
#0.3838384

#yelp.tr1 <- yelp  
#yelp.tr1$reviews = yelp.tr1$reviews^(383/1000)  
  
yelp.tr = yelp.ts^(383/1000)  
  
bulkfit(yelp.tr)

## $res  
## ar d ma AIC  
## [1,] 0 0 0 1180.7103  
## [2,] 0 0 1 1027.3342  
## [3,] 0 0 2 897.8681  
## [4,] 0 1 0 539.0409  
## [5,] 0 1 1 535.4975  
## [6,] 0 1 2 535.9028  
## [7,] 0 2 0 652.6826  
## [8,] 0 2 1 532.7091  
## [9,] 0 2 2 520.9066  
## [10,] 1 0 0 552.5392  
## [11,] 1 0 1 549.0796  
## [12,] 1 0 2 549.4349  
## [13,] 1 1 0 534.5251  
## [14,] 1 1 1 536.1724  
## [15,] 1 1 2 537.6569  
## [16,] 1 2 0 586.4044  
## [17,] 1 2 1 522.3971  
## [18,] 1 2 2 516.3467  
## [19,] 2 0 0 548.1023  
## [20,] 2 0 1 549.8817  
## [21,] 2 0 2 548.5923  
## [22,] 2 1 0 535.9705  
## [23,] 2 1 1 537.9192  
## [24,] 2 1 2 534.7806  
## [25,] 2 2 0 568.8199  
## [26,] 2 2 1 524.0569  
## [27,] 2 2 2 525.7721  
##   
## $min  
## ar d ma AIC   
## 1.0000 2.0000 2.0000 516.3467

# 1.0000 2.0000 2.0000 516.3467   
yelp.bulkfit2 <- bulkfit2(yelp.tr,12)

## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1  
  
## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced  
  
## Warning in log(s2): NaNs produced

## Warning in arima(x, order = c(i, k, j), seasonal = list(order = c(I, K, :  
## possible convergence problem: optim gave code = 1

yelp.tr.fit = arima(yelp.tr,order =c(1,2,2))  
yelp.tr.fit2 <- arima(yelp.tr, order = c(0,1,2),seasonal = list(order=c(0,1,2),period=12))  
  
yelp.tr.fit

##   
## Call:  
## arima(x = yelp.tr, order = c(1, 2, 2))  
##   
## Coefficients:  
## ar1 ma1 ma2  
## 0.5384 -1.9144 0.9144  
## s.e. 0.1318 0.1004 0.0993  
##   
## sigma^2 estimated as 2.596: log likelihood = -254.17, aic = 516.35

#Coefficients:  
# ar1 ma1 ma2  
# 0.5384 -1.9144 0.9144  
#s.e. 0.1318 0.1004 0.0993  
  
#sigma^2 estimated as 2.596: log likelihood = -254.17, aic = 516.35  
  
yelp.tr.fit2

##   
## Call:  
## arima(x = yelp.tr, order = c(0, 1, 2), seasonal = list(order = c(0, 1, 2), period = 12))  
##   
## Coefficients:  
## ma1 ma2 sma1 sma2  
## -0.1155 -0.2259 -0.6475 0.1870  
## s.e. 0.0907 0.0884 0.1093 0.1181  
##   
## sigma^2 estimated as 1.021: log likelihood = -175.85, aic = 361.69

# Call:  
# arima(x = yelp.tr, order = c(0, 1, 2), seasonal = list(order = c(0, 1, 2), period = 12))  
#   
# Coefficients:  
# ma1 ma2 sma1 sma2  
# -0.1155 -0.2259 -0.6475 0.1870  
# s.e. 0.0907 0.0884 0.1093 0.1181  
#   
# sigma^2 estimated as 1.021: log likelihood = -175.85, aic = 361.69

The more accurate ARIMA model, the (0,1,2)(0,1,2) one, has an aic of 361.69, so we're going to model that one. The equation yields:

yelp.tr.pred = predict(yelp.tr.fit, n.ahead=12)  
yelp.tr.pred2 = predict(yelp.tr.fit2, n.ahead=12)  
  
  
length(yelp)

## [1] 134

#134  
  
Lower = yelp.tr.pred$pred - 1.96\*yelp.tr.pred$se/sqrt(134)  
Lower2 = yelp.tr.pred2$pred - 1.96\*yelp.tr.pred2$se/sqrt(134)  
Upper = yelp.tr.pred$pred + 1.96\*yelp.tr.pred$se/sqrt(134)  
Upper2 = yelp.tr.pred2$pred + 1.96\*yelp.tr.pred2$se/sqrt(134)  
  
Predict = yelp.tr.pred$pred  
Predict2 = yelp.tr.pred2$pred  
  
pred.mat = cbind(Lower,Predict,Upper)  
pred2.mat = cbind(Lower2,Predict2,Upper2)  
  
  
  
  
final.mat = pred.mat^(1000/383)  
final2.mat = pred2.mat^(1000/383)  
  
  
final.mat

## Lower Predict Upper  
## Jan 2016 55748.98 56357.62 56970.34  
## Feb 2016 58410.38 59151.87 59899.13  
## Mar 2016 60420.78 61227.58 62040.99  
## Apr 2016 62061.33 62909.14 63764.06  
## May 2016 63501.07 64379.64 65265.68  
## Jun 2016 64836.08 65740.74 66653.16  
## Jul 2016 66119.83 67048.39 67984.96  
## Aug 2016 67381.79 68333.26 69292.99  
## Sep 2016 68638.12 69612.14 70594.63  
## Oct 2016 69897.70 70894.20 71899.40  
## Nov 2016 71165.39 72184.47 73212.51  
## Dec 2016 72443.85 73485.73 74536.81

final2.mat

## Lower2 Predict2 Upper2  
## Jan 2016 62598.15 63005.87 63415.23  
## Feb 2016 58578.62 59101.50 59627.25  
## Mar 2016 66681.60 67313.26 67948.60  
## Apr 2016 65336.05 66018.34 66705.00  
## May 2016 67881.30 68635.02 69393.88  
## Jun 2016 69576.52 70394.01 71217.41  
## Jul 2016 76453.31 77371.70 78296.86  
## Aug 2016 76371.03 77338.06 78312.62  
## Sep 2016 68857.40 69808.72 70768.10  
## Oct 2016 70288.88 71294.94 72309.83  
## Nov 2016 67077.36 68094.62 69121.34  
## Dec 2016 66971.65 68026.15 69090.85

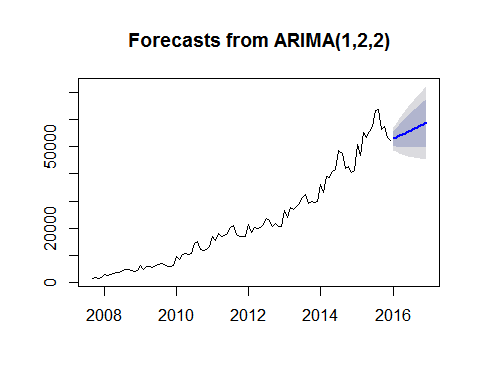
######decom   
yelp.dec = decom1(yelp,fore1 =12,se1 =12)

## (Intercept) t1   
## -11673.7925 421.1653   
## pred lower upper  
## 1 45183.52 31814.18 58552.87  
## 2 45604.69 32230.94 58978.43  
## 3 46025.85 32647.65 59404.06  
## 4 46447.02 33064.28 59829.76  
## 5 46868.19 33480.86 60255.51  
## 6 47289.35 33897.37 60681.33  
## 7 47710.52 34313.82 61107.21  
## 8 48131.68 34730.21 61533.16  
## 9 48552.85 35146.53 61959.16  
## 10 48974.01 35562.79 62385.23  
## 11 49395.18 35978.99 62811.37  
## 12 49816.34 36395.12 63237.56

yelp.dec.final = yelp.dec$pred.df  
  
yelp.dec.final

## pred lower upper  
## 1 45183.52 31814.18 58552.87  
## 2 45604.69 32230.94 58978.43  
## 3 46025.85 32647.65 59404.06  
## 4 46447.02 33064.28 59829.76  
## 5 46868.19 33480.86 60255.51  
## 6 47289.35 33897.37 60681.33  
## 7 47710.52 34313.82 61107.21  
## 8 48131.68 34730.21 61533.16  
## 9 48552.85 35146.53 61959.16  
## 10 48974.01 35562.79 62385.23  
## 11 49395.18 35978.99 62811.37  
## 12 49816.34 36395.12 63237.56

#######using forecast to make plot  
fit\_ar = arima(yelp, order= c(1,2,2))  
fit\_ar2 = arima(yelp, order = c(0,1,2),seasonal = list(order=c(0,1,2),period=12))  
  
fit\_ar\_f = forecast(fit\_ar, h =12)  
fit\_ar\_f2 = forecast(fit\_ar2,h=12)  
  
plot(fit\_ar\_f, include =100)



plot(fit\_ar\_f2,include=100)

