Forecast of Restaurant Visitors using AIR Regi and Hot Pepper Gourmet Data

ADEC 7460 Predictive Analytics

Nick Howard

Contents

1 Introduction ... 3
1 Data ... 4
1.1 Target Variable: Visits ... 5
1.2 Variable: Reservations ... 7
2 Type of Models and Formulation ... 10
2.1 ARIMA ... 11
2.3 ARIMA for 2017 ... 14
2.4 ARIMA for individual stores ... 15
2.5 Wrap Up ... 20
Appendix: Works Cited ... 21
Appendix: Dataset Descriptions ... 22
Appendix: R Code ... 24

1 Introduction

Problem Statement : Manager's in the Food and Beverage Industry are tasked with making management decisions in the face of uncertainty. How many waiters and waitresses to schedule each shift and the quantity of ingredients to have in stock are decisions that are influenced by the amount of customers that visit a restaurant on a daily basis. Customers tastes change and they will not always consistently visit the same restaurant but by using forecast methods to get a rough estimate of the amount of customers that they can expect each day a manager will be able to make much better decisions when faced with the two questions mentioned above.

Significance: The Food Waste Reduction Alliance found that 84% of unused food ends up being desposed of by America's restaurants while only 1.4% is donated. This impacts a restaurants bottom line and the restaurant will have to raise their prices to stay profitable. It is difficult to make an accurate forecast of the visitors to a restaurant on any given day and on some days the forecast may be incorrect but this information may be accurate enough to reduce the staggering amount of food waste in this industry.

Literature: ETS and ARIMA models are widely used in time series forecasting. They were used successfully in 2008 to model domestic tourism in Australia[2]. Panigrahi et al.[3] tells us that ETS has linear and non-linear modeling capability. This is especially useful in time series forecasting where our data can have both characteristics. In retail sales, ARIMA was shown to be similar to state-spaced models (on-step and multi-step), on the metrics of RMSE, MAE and MAPE[4]. Finally, in the electric power markets, ARIMA models were used to analyze time series data from mainland Spain and Californian markets[5]. These models are used very often in academia and industry with very positive results.

1 Data

There are four data sets that we can use for out analysis.

Air Reserve (1.1) and HPG Reserve (1.2) contain observations related to reservations that were made in the Air REGI and Hot Pepper Gourmet Reservation systems. These data sets will allow us to view summary statistics and data visualizations that will give us information on the amount of visitors that restaurants have had in the past.

Air Store Info (1.3) and HPG Store Info (1.4) contain data on the type of restaurants that we are looking into. When we join this table with the reservation data we will be able to determine if the type of food served impacts the amount of visitors a restaurant can expect.

1.1 Target Variable: Visits

We are attempting to predict daily visits to the restaurants in question. We will explore the descriptive statistics and the trend and seasonality of this data. There are 252,108 observations in this data set. Visits that occurred in 2016 were more frequently observed than visits in 2017. The mean amount of visitors on any given day for a single observation is 21. The most visitors on any day was 877. This analysis was performed on the Air Visit data.

air_store_id	visit_date	visitors
Length:252108	Min. :2016-01-01	Min. : 1.00
Class :character	1st Qu.:2016-07-23	1st Qu.: 9.00
Mode :character	Median :2016-10-23	Median : 17.00
	Mean :2016-10-12	Mean : 20.97
	3rd Qu.:2017-01-24	3rd Qu.: 29.00
	Max. :2017-04-22	Max. :877.00

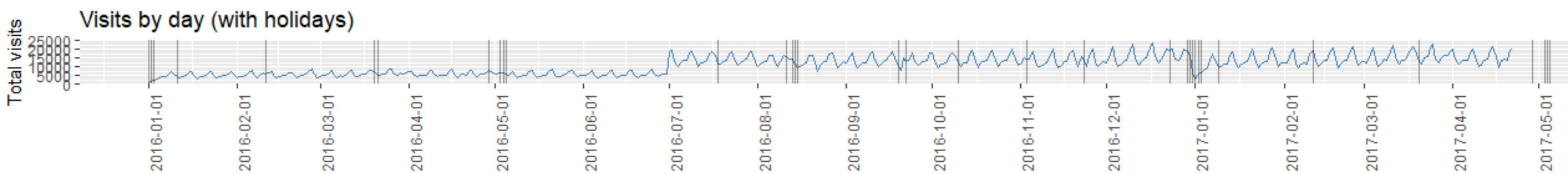


Figure 1: Total Visits by day with holidays

The plot above shows the total visits by day. We see that in July 2016 the number of visitors that booked a reservation using this service increased. Holidays are marked with a black line on the plot. We can see that there is usually a drop in visitors on holidays.

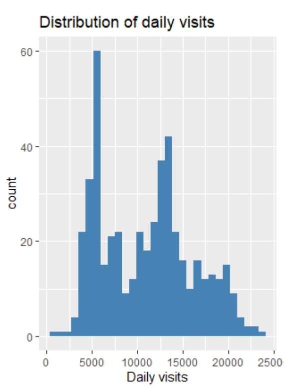


Figure 2: After grouping the data to show the total visitors by day we find that the maximum number of visitors is 60.

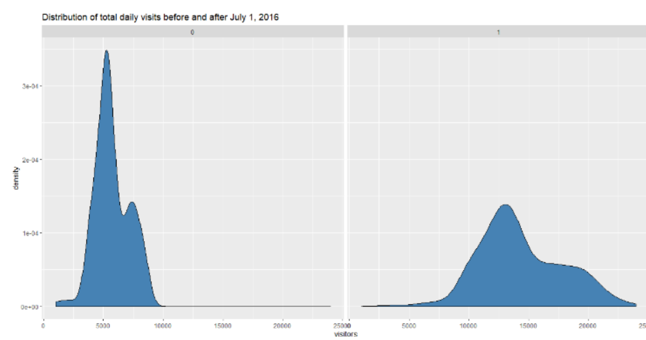


Figure 3: The figure above illustrates the difference in mean visitors before and after July of 2016. We can see that the mean visitors is much higher after July 2016.

1.2 Variable: Reservations

There are four variables included in the HPG data set. One will give us the data and time of the visit and one will give us information on the date and time that the reservation was made. One variable will give us information on the amount of visitors that the reservation was made for. There are over 2 million observations in this data set and the mean visitors for any reservation was 5.

hpg_store_id	visit_datetime
Length: 2000320	Length: 2000320
Class :character	Class :character
Mode :character	Mode :character

reserve_datetime	reserve_visitors
Length:2000320	Min. : 1.000
Class :character	1st Qu.: 2.000
Mode :character	Median : 3.000
	Mean : 5.074
	3rd Qu.: 6.000
	Max. :100.000

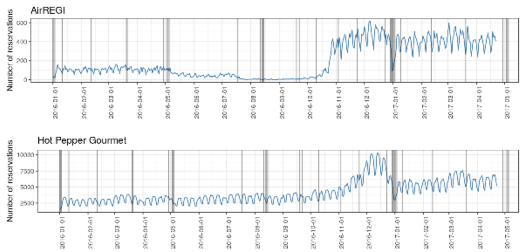


Figure 4: Hot Pepper is a much more popular reservation service. They book reservations for thousands of people while Air REGI is still a young company and only books reservations for hundreds of customers.

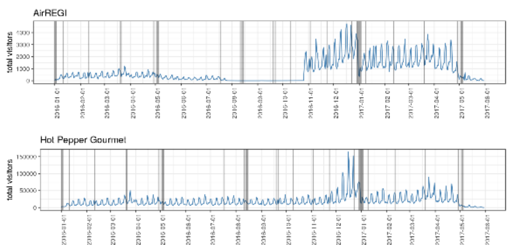


Figure 5: Hot Pepper is a much more popular reservation service. Since HPG books more reservations they also book for more people.

1.3 Stores

Choice is an important factor to today's consumer. When using an online reservation service a customer will most likely use a service that provides them with the most choices. Consumers book reservations through HPG much more often then they do with Air REGI and this may be due to the fact that the consumer has more choices on HPG.

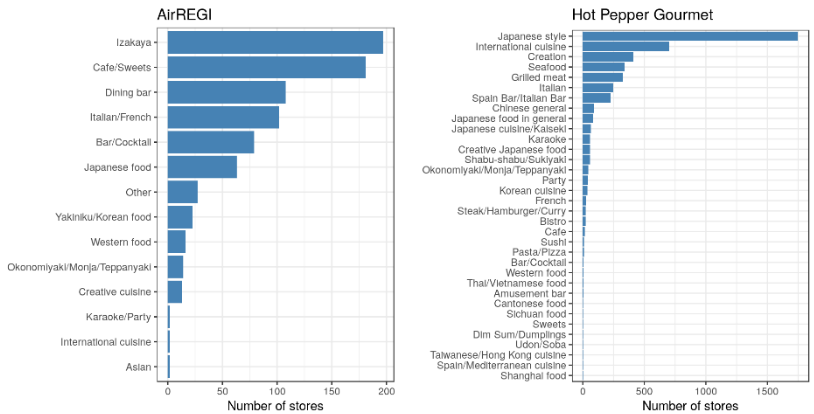


Figure 6: Hot Pepper has more categories than Air REGI. This could be the reason that HPG is a more popular service than Air REGI.

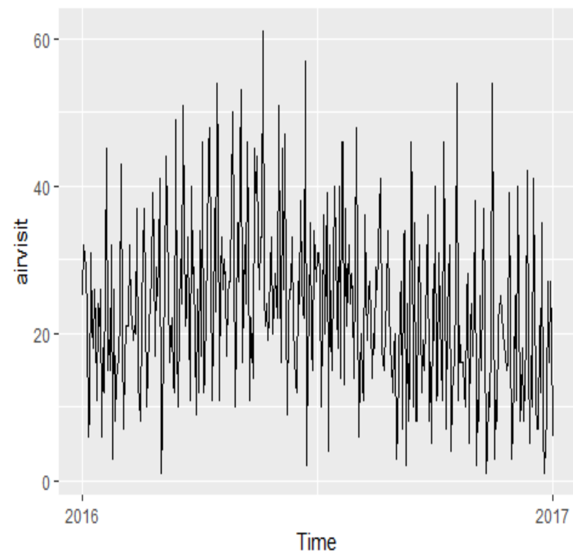
2 Type of Models and Formulation

- What model classes did you build? Why?
- How did you formulate / implement you model in Python / R?

I used the Forecast and GGplot libraries to complete my first forecasts.

I started by creating a time series object of the air visit data using the ts() function and the parameters listed below.

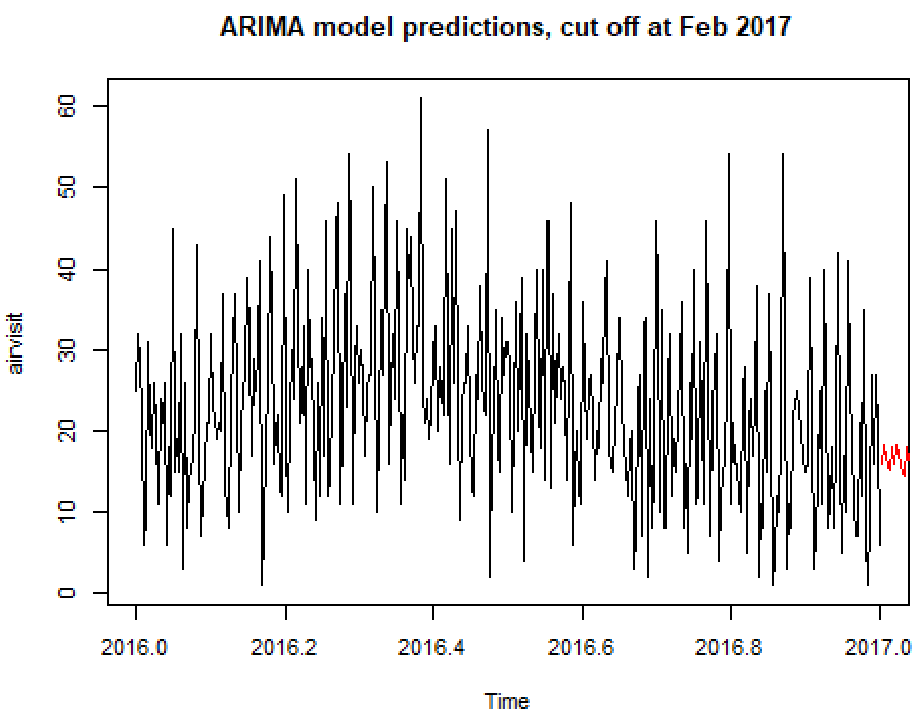
ts(air_visit_data$visitors, frequency = 365, start=2016, end=2017, deltat=1/365)



autoplot(airvisit)

2.1 ARIMA

The first model that I used was an ARIMA Model. The ARIMA model did model the seasonality well. The problem is not that the ARIMA model did not model the variance well the problem is that it did not accurately model the magnitude of the variance well. As you can see below the ARIMA model predictions ranged from around 18-22 visitors while the actual data ranged from 0 to 60 visitors.

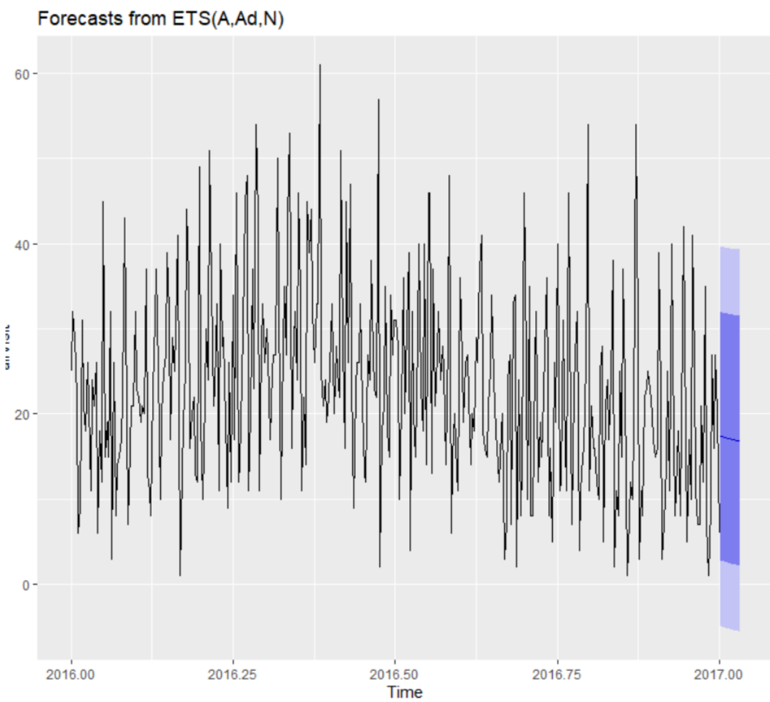



```
arimaAV<-arima(airvisit, order=c(2,1,2), seaso
nal= list(order=c(1,1,1), period=7))

ypred2<-forecast::forecast(arimaAV,h=80)

par(mfrow=c(1,1), cex=0.7)
plot(airvisit, main="ARIMA model predictions,
cut off at Feb 2017")
lines(ypred2$mean, col='red')
```

```
arimaAV

Call:
arima(x = airvisit, order = c(2, 1, 2), season
al = list(order = c(1, 1, 1),
period = 7))

Coefficients
ar1       ar2        ma1      ma2     sar1      sma1

0.6196   -0.1831   -1.6357   0.6610   0.0602   0.99
99

s.e.   0.1356   0.0670   0.1350   0.1283   0.0612
0.0547

sigma^2 estimated as 122:
log likelihood = -1383.62,   aic = 2781.24

arimaAV$aic
[1] 2781.244
```

2.2 ETS

The ETS model provided us with confidence interval predictions. The confidence interval predictions are provided below. The forecast was 17 visitors. The confidence intervals ranged from 3 to 32 visitors (80%) and -5 to 50 visitors (95%). The AIC of this model was much higher than the AIC of the ARIMA model.




```
ets1<-ets(airvisit)
```

```
## Warning in ets(airvisit): I can't handle data with fre
quency greater than
## 24. Seasonality will be ignored. Try stlf() if you nee
d seasonal forecasts.
```

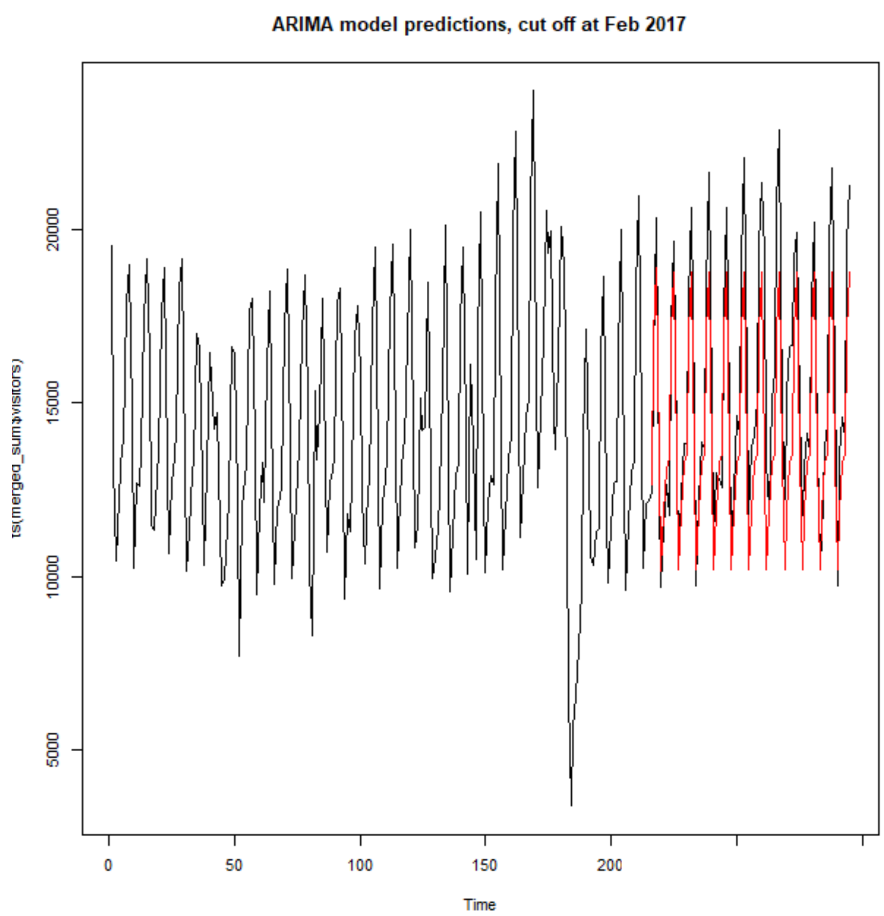
```
fc1<-forecast(ets1, h=ifelse(ets1$m>1, 2*ets1$m, 10),
level=c(80,95), fan=FALSE, simulate=FALSE, bootstrap=FALS
E,
npaths=5000, PI=TRUE, lambda=ets1$lambda, biasadj=NULL)
```

```
ets1
ETS(A,Ad,N)

Call:
 ets(y = airvisit)

  Smoothing parameters:
    alpha = 0.007
    beta  = 0.0056
    phi   = 0.9094

  Initial states:
    l = 24.8388
    b = -0.5938

  sigma:  11.3853

     AIC     AICc      BIC
3947.789 3948.023 3971.205
```

Point	Forecast	Lo 80	Hi 80	Lo 95	Hi 95
2017.0027	17.40313	2.81229629	31.99397	-4.911627	39.71789
2017.0055	17.32550	2.73359760	31.91740	-4.990888	39.64189
2017.0082	17.25490	2.66096661	31.84882	-5.064593	39.57438
2017.0110	17.19069	2.59358630	31.78779	-5.133652	39.51503
2017.0137	17.13229	2.53076039	31.73383	-5.198825	39.46341
2017.0164	17.07919	2.47189447	31.68649	-5.260741	39.41912

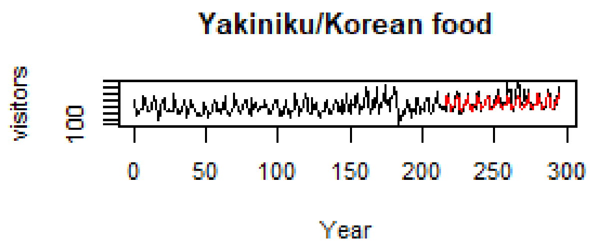
2.3 ARIMA for 2017

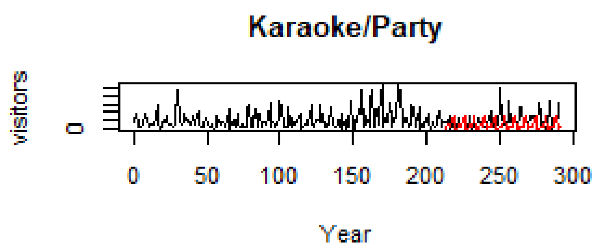
The ARIMA model for a subset of 2017 data was very good. It models the data accurately and provides useful predictions.

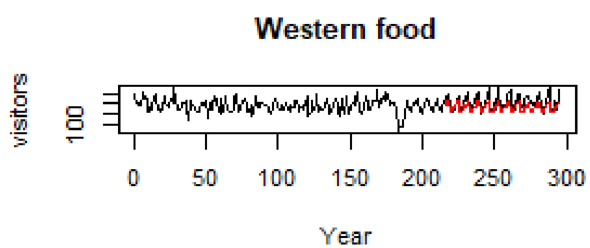


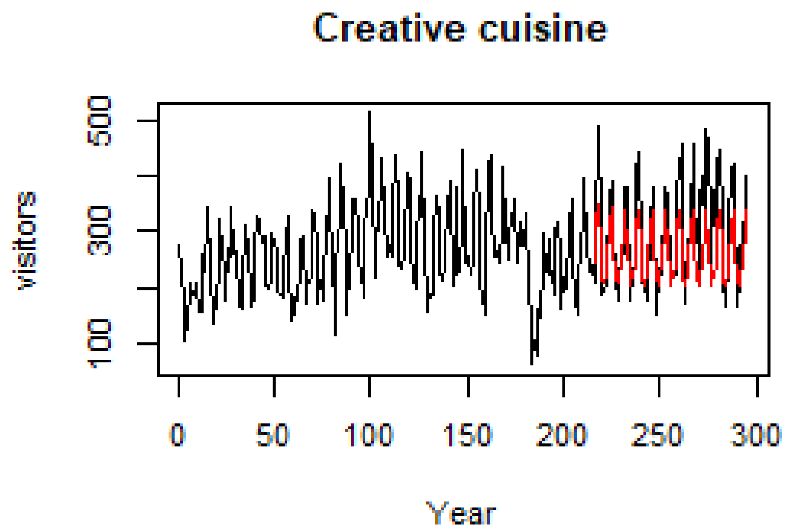
2.4 ARIMA for individual stores

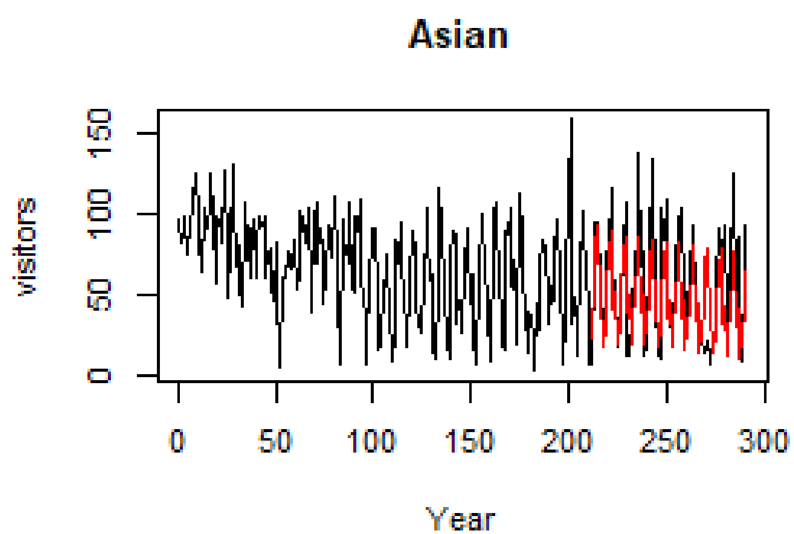
It was helpful to see predictions by individual store type.

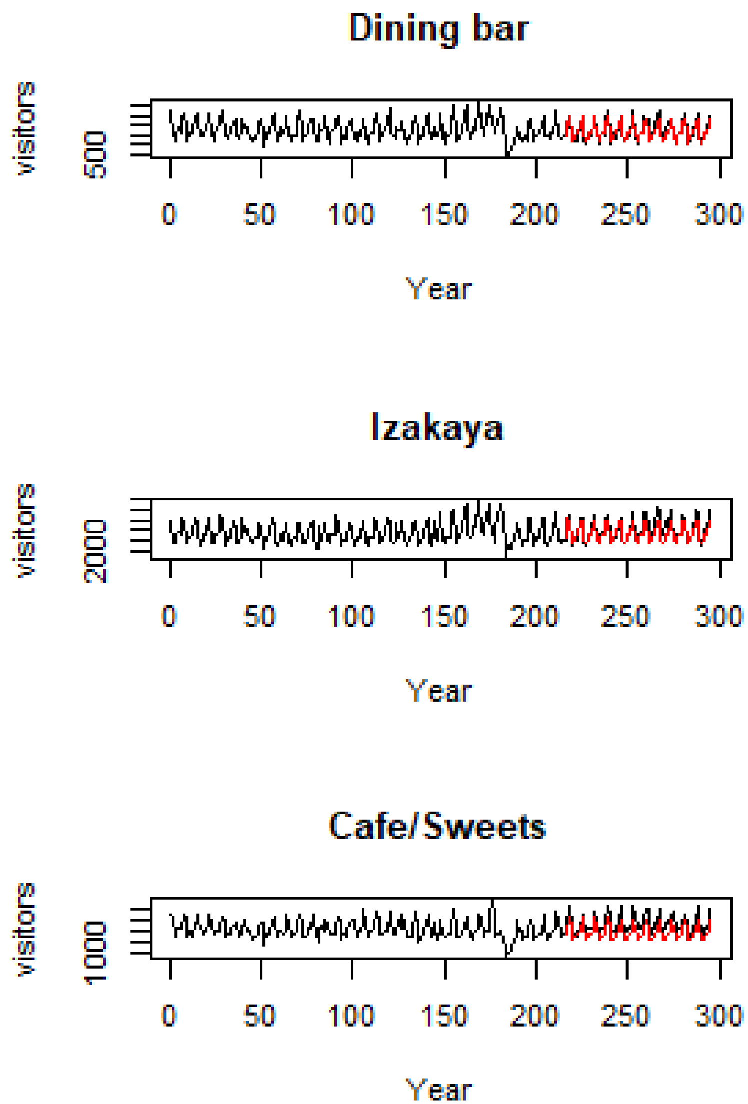




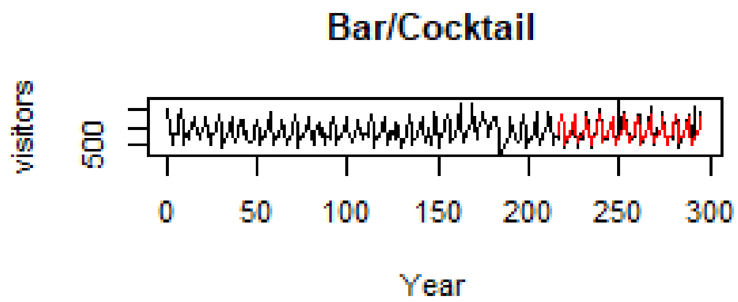


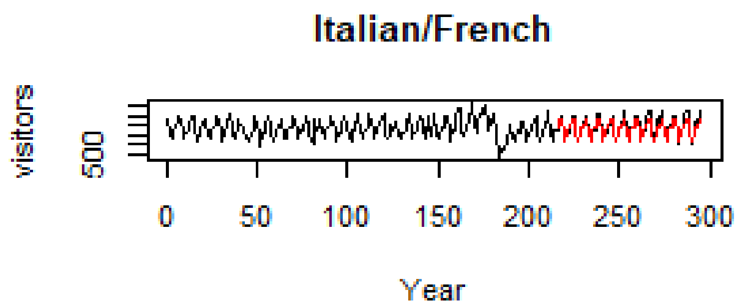

Creative cuisine
visitors
100
300
500
0
50
100
150
200
250
300
Year

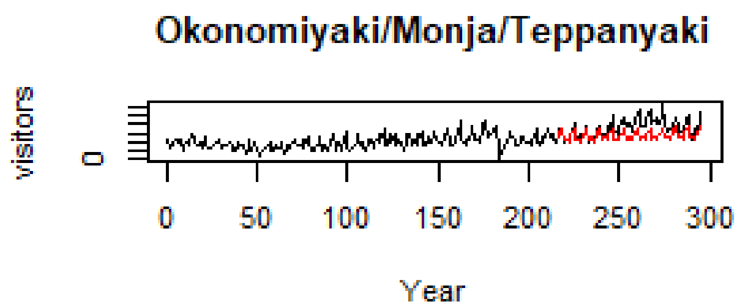

Asian
visitors
0
50
100
150
0
50
100
150
200
250
300
Year



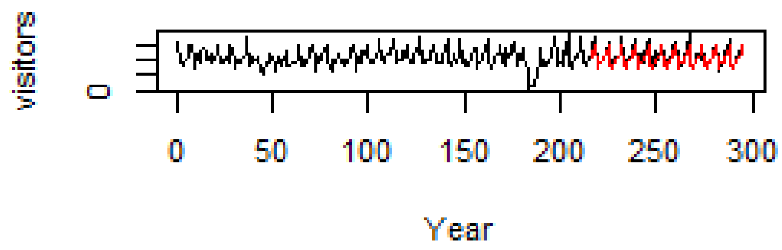
Dining bar
visitors
500
0
50
100
150
200
250
300
Year
Izakaya
visitors
2000
0
50
100
150
200
250
300
Year
Cafe/Sweets
visitors
1000
0
50
100
150
200
250
300
Year


Bar/Cocktail
visitors
500
0
50
100
150
200
250
300
Year

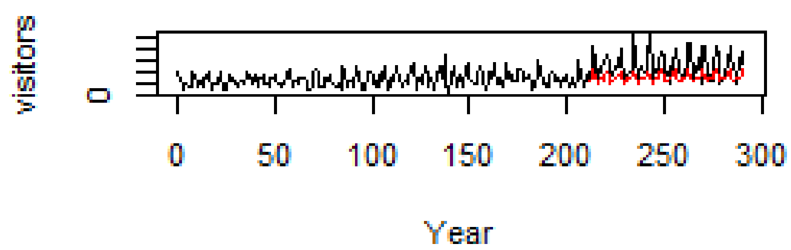

Italian/French
visitors
500
0
50
100
150
200
250
300
Year


Okonomiyaki/Monja/Teppanyaki
visitors
0
0
50
100
150
200
250
300
Year

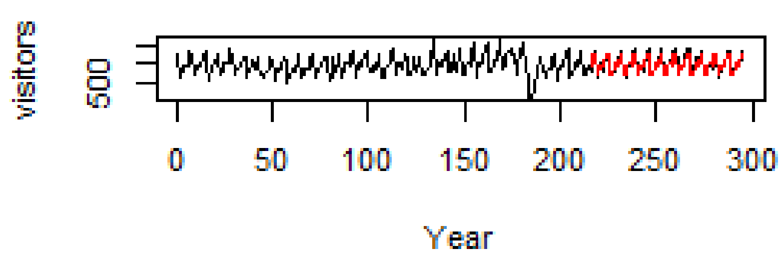
Other



International cuisine



Japanese food



2.5 Wrap Up

Performance / Accuracy: The models performed very well. The AIC on the ARIMA model was slightly less than the AIC on the ETS model. The ARIMA family of models also provided much more "legible" models. Instead of providing the confidence intervals, like the ETS, the ARIMA models provided actual data values that could be layered on a full dataset to see visualize the accuracy of the forecast.

Limitations: The ARIMA family of models were less accurate when making predictions about future results. They provided a very conservative result that did not look very accurate when viewed along with the observed historical data.

Future Work: In the future I would like to review different model types such as neural networks to see if these models provide a more accurate prediction.

Learning: I learned a lot about how to subset data and create ARIMA and ETS models. I also learned that there is a major difference between ARIMA and ETS models and that difference is that ARIMA can be used to provide realistic predictions while ETS can only be used to produce confidence intervals. Both have their place in an analysis and they can be useful under different circumstances. ARIMA would be more valuable when accuracy is important and ETS might be better when a big picture of the predicted values is needed.

Appendix: Works Cited

1) http://www.foodwastealliance.org/wp-content/uploads/2014/11/FWRA_BSR_Tier3_FINAL.pdf
2) George Athanasopoulos and Rob J.Hyndman. (2007). Modelling and forecasting Australian domestic tourism *Tourism Management: Volume 29, Issue 1, February 2008, Pages 19-31* Available at https://www.sciencedirect.com/science/article/pii/S0261517707001057
3) Sibarama Panigrahi and H.S. Behera (2017). A hybrid ETS–ANN model for time series forecasting *Engineering Applications of Artificial Intelligence Volume 66, November 2017, Pages 49-59* Available at https://www.sciencedirect.com/science/article/abs/pii/S0952197617301550
4) Patrícia Ramosa et al. (2015). Performance of state space and ARIMA models for consumer retail sales forecasting *Robotics and Computer-Integrated Manufacturing Volume 34, August 2015, Pages 151-163* Available at https://www.sciencedirect.com/science/article/abs/pii/S0736584515000137
5) J. Contreras et al. (2003) ARIMA models to predict next-day electricity prices IEEE Transactions on Power Systems Volume 18 Issue 3 . Available at https://ieeexplore.ieee.org/abstract/document/1216141

Appendix: Dataset Descriptions

1.1 air_reserve.csv

This file contains reservations made in the air system. Note that the reserve_datetime indicates the time when the reservation was created, whereas the visit_datetime is the time in the future where the visit will occur.

- air_store_id - the restaurant's id in the air system
- visit_datetime - the time of the reservation
- reserve_datetime - the time the reservation was made
- reserve_visitors - the number of visitors for that reservation

1,2 hpg_reserve.csv

This file contains reservations made in the hpg system.

- hpg_store_id - the restaurant's id in the hpg system
- visit_datetime - the time of the reservation
- reserve_datetime - the time the reservation was made
- reserve_visitors - the number of visitors for that reservation

1.3 air_store_info.csv

This file contains information about select air restaurants. Column names and contents are self-explanatory.

- air_store_id
- air_genre_name
- air_area_name
- latitude
- longitude
- Note: latitude and longitude are the latitude and longitude of the area to which the store belongs

1.4 hpg_store_info.csv

This file contains information about select hpg restaurants. Column names and contents are self-explanatory.

- hpg_store_id
- hpg_genre_name
- hpg_area_name
- latitude
- longitude

Note: latitude and longitude are the latitude and longitude of the area to which the store belongs

1.5 store_id_relation.csv

This file allows you to join select restaurants that have both the air and hpg system.

- hpg_store_id
- air_store_id
- air_visit_data.csv
- This file contains historical visit data for the air restaurants.

1.6 air_store_id

- visit_date - the date
- visitors - the number of visitors to the restaurant on the date

1.7 sample_submission.csv

This file shows a submission in the correct format, including the days for which you must forecast.

- id - the id is formed by concatenating the air_store_id and visit_date with an underscore
- visitors- the number of visitors forecasted for the store and date combination

date_info.csv

This file gives basic information about the calendar dates in the dataset.

- calendar_date
- day_of_week
- holiday_flg - is the day a holiday in Japan

Appendix: R Code

#Library:

```
library(fpp2)
library(forecast)
library(ggplot2)
library(readxl)
library(tidyverse)
library(data.table)
library(magrittr)
library(ggplot2)
library(lubridate)
library(xts)
library(forecast)
library(DT)
library(gridExtra)
library(leaflet)
library(htmltools)
library(mapdata)
library(maptools)
library(sp)
```

#Load Data:

```
air.visit <- read.csv("C:/Users/Nicholas Howard/Desktop/Applied Economics/Forecasting/recruit-restaurant-visitor-forecasting/air_visit_data.csv")

air.reserve <- read.csv("C:/Users/Nicholas Howard/Desktop/Applied Economics/Forecasting/recruit-restaurant-visitor-forecasting/air_reserve.csv", stringsAsFactors = FALSE)

air.store <- read.csv("C:/Users/Nicholas Howard/Desktop/Applied Economics/Forecasting/recruit-restaurant-visitor-forecasting/air_store_info.csv", stringsAsFactors = FALSE)

hpg.reserve <- read.csv("C:/Users/Nicholas Howard/Desktop/Applied Economics/Forecasting/recruit-restaurant-visitor-forecasting/hpg_reserve.csv", stringsAsFactors = FALSE)

hpg.store <- read.csv("C:/Users/Nicholas Howard/Desktop/Applied Economics/Forecasting/recruit-restaurant-visitor-forecasting/hpg_store_info.csv", stringsAsFactors = FALSE)

date <- read.csv("C:/Users/Nicholas Howard/Desktop/Applied Economics/Forecasting/recruit-restaurant-visitor-forecasting/date_info.csv", stringsAsFactors = FALSE)

store.id <- read.csv("C:/Users/Nicholas Howard/Desktop/Applied Economics/Forecasting/recruit-restaurant-visitor-forecasting/store_id_relation.csv", stringsAsFactors = FALSE)
```

#Visits

#Figure 1, 2, 3:

```
visits[, visit_date := as.Date(visit_date)]

holidays <- dates[holiday_flg == 1, ]

p1 <- visits[, .(total_visit = sum(visitors)), by = visit_date] %>%
   ggplot(aes(x = visit_date, y = total_visit)) +
   geom_line(color = 'steelblue') +
   geom_vline(data = holidays, aes(xintercept = as.Date(calendar_date)), alpha = 0.4) +
   scale_x_date(date_breaks = "1 month") +
   theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
   labs(x = '', y = 'Total visits', title = 'Visits by day (with holidays)')

p2 <- visits[order(wday(visit_date)), .(mean_visits = mean(visitors)), by = weekdays(visit_date)] %>%
   ggplot(aes(x = reorder(weekdays, seq(1,7)), y = mean_visits)) +
   geom_bar(stat = 'identity', fill = 'steelblue') +
   labs(x = '', y = "Mean visits", title = "Average visits by day of week")

p3 <- visits[, .(total_visitors = sum(visitors)), by = visit_date] %>%
   as.xts() %>%
   ggAcf() +
   labs(title = 'Autocorrelation plot of total visitors')

p4 <- visits[, .(total_visitors = sum(visitors)), by = visit_date] %>%
   as.xts() %>%
   ggPacf() +
   labs(title = 'Partial Autocorrelation plot of total visitors')



grid.arrange(p1, p2, p3, p4, nrow = 4)


visits[, .(visitors = sum(visitors), s = ifelse(visit_date < as.Date('2016-07-01'), 0, 1)), by = visit_date] %>%
+     ggplot(aes(x = visitors)) +
+     geom_density(fill = 'steelblue') +
+     facet_grid(~s) +
+     labs(title = 'Distribution of total daily visits before and after July 1, 2016')
```

#Reservations:

```
summary(hpg.reserve)

dt_cols <- c('visit_datetime', 'reserve_datetime')

air_res[, (dt_cols) := lapply(.SD, as_datetime), .SDcols = dt_cols]
hpg_res[, (dt_cols) := lapply(.SD, as_datetime), .SDcols = dt_cols]
```

#Figure 4:

```
p1 <- air_res[, .(number_reservations = .N), by = .(date = as.Date(reserve_datetime))] %>%
  ggplot(aes(x = date, y = number_reservations)) +
  geom_line(color = 'steelblue') +
  geom_vline(data = holidays, aes(xintercept = as.Date(calendar_date)), alpha = 0.4) +
  scale_x_date(date_breaks = "1 month") +
  labs(x = '', y = 'Number of reservations', title = 'AirREGI') +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

p2 <- hpg_res[, .(number_reservations = .N), by = .(date = as.Date(reserve_datetime))] %>%
  ggplot(aes(x = date, y = number_reservations)) +
  geom_line(color = 'steelblue') +
  geom_vline(data = holidays, aes(xintercept = as.Date(calendar_date)), alpha = 0.4) +
  scale_x_date(date_breaks = "1 month") +
  labs(x = '', y = 'Number of reservations', title = 'Hot Pepper Gourmet') +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

grid.arrange(p1, p2)
```

#Figure 5:

```
p1 <- air_res[, .(total_visitors = sum(reserve_visitors)), by = .(date = as.Date(visit_datetime))] %>%
  ggplot(aes(x = date, y = total_visitors)) +
  geom_line(color = 'steelblue') +
  geom_vline(data = holidays, aes(xintercept = as.Date(calendar_date)), alpha = 0.4) +
  scale_x_date(date_breaks = "1 month") +
  labs(x = '', y = 'total visitors', title = 'AirREGI') +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

p2 <- hpg_res[, .(total_visitors = sum(reserve_visitors)), by = .(date = as.Date(visit_datetime))] %>%
  ggplot(aes(x = date, y = total_visitors)) +
  geom_line(color = 'steelblue') +
  geom_vline(data = holidays, aes(xintercept = as.Date(calendar_date)), alpha = 0.4) +
  scale_x_date(date_breaks = "1 month") +
  labs(x = '', y = 'total visitors', title = 'Hot Pepper Gourmet') +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))

grid.arrange(p1, p2)
```

#Figure 6:

```
n_air <- length(unique(air_store$air_store_id))
n_hpg <- length(unique(hpg_store$hpg_store_id))

print(paste('The AirREGI data has', n_air, 'stores', 'and the Hot Pepper Gourmet data has', n_hpg, 'stores.'))
## [1] "The AirREGI data has 829 stores and the Hot Pepper Gourmet data has 4690 stores."
```

```
p1 <- air_store[, .N, by = air_genre_name] %>%
  ggplot(aes(x = reorder(air_genre_name, N), y = N)) +
  geom_bar(stat = 'identity', fill = 'steelblue') +
  labs(x = '', y = 'Number of stores', title = 'AirREGI') +
  coord_flip()

p2 <- hpg_store[, .N, by = hpg_genre_name] %>%
  ggplot(aes(x = reorder(hpg_genre_name, N), y = N)) +
  geom_bar(stat = 'identity', fill = 'steelblue') +
  labs(x = '', y = 'Number of stores', title = 'Hot Pepper Gourmet') +
  coord_flip()

grid.arrange(p1, p2, ncol = 2)


library(forecast)
library(ggplot2)

air_visit_data <- read.csv("C:/Users/Nicholas Howard/Desktop/Applied Economics/Forecasting/recruit-restaurant-visitor-forecasting/air_visit_data.csv")
 View(air_visit_data)

airvisit <- ts(air_visit_data$visitors, frequency = 365, start=2016, end=2017, deltat=1/365)

autoplot(airvisit)

arimaAV<-arima(airvisit, order=c(2,1,2), seasonal= list(order=c(1,1,1), period=7))

ypred2<-forecast::forecast(arimaAV,h=80)

par(mfrow=c(1,1), cex=0.7)
plot(airvisit, main="ARIMA model predictions, cut off at Feb 2017")
lines(ypred2$mean, col='red')

ets1<-ets(airvisit)
## Warning in ets(airvisit): I can't handle data with frequency greater than
## 24. Seasonality will be ignored. Try stlf() if you need seasonal forecasts.
fc1<-forecast(ets1, h=ifelse(ets1$m>1, 2*ets1$m, 10),
level=c(80,95), fan=FALSE, simulate=FALSE, bootstrap=FALSE,
npaths=5000, PI=TRUE, lambda=ets1$lambda, biasadj=NULL)

library(ggplot2) # Data visualization
library(readr) # CSV file I/O, e.g. the read_csv function
library(knitr)
library(tidyverse)
# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Enter) will list the files in the input directory

system("ls ../input")

# Any results you write to the current directory are saved as output.
```

```r
df_air_store <- read.csv("C:/Users/Nicholas Howard/Desktop/Applied Economics/Forecasting/recruit-restaurant-visitor-forecasting/air_store_info.csv", stringsAsFactors = FALSE)
df_air <- read.csv("C:/Users/Nicholas Howard/Desktop/Applied Economics/Forecasting/recruit-restaurant-visitor-forecasting/air_visit_data.csv", stringsAsFactors = FALSE)


par(mfrow=c(2,1), cex=0.7)
df_air %>%
  group_by(visit_date) %>%
  summarize(visitors = sum(df_air$visitors)) %>%
  plot(type='l', main='Overall Visitors')

merged <- df_air %>%
  filter(visit_date > '2016-07-01') %>%
  dplyr::left_join(df_air_store, by='air_store_id', how='left')

merged_sum <- merged %>%
  group_by(visit_date) %>%
  summarize(visitors = sum(visitors))

merged_sum %>%
  plot(type='l', xlab='Year', main='Cut-off at July 2016')

merged_train <- merged_sum %>% filter(visit_date <='2017-02-01')
merged_test <- merged_sum %>% filter(visit_date >'2017-02-01')

#print(paste(nrow(merged_sum),nrow(merged_train),nrow(merged_test)))

m <- arima(merged_train$visitors, order=c(2,1,2), seasonal= list(order=c(1,1,1), period=7))
y_pred <- forecast::forecast(m, h=80)

par(mfrow=c(1,1), cex=0.7)
plot(ts(merged_sum$visitors), main="ARIMA model predictions, cut off at Feb 2017")
lines(y_pred$mean, col='red')

genre_sum <- merged %>%
  group_by(visit_date, air_genre_name) %>%
  summarize(visitors=sum(visitors))

genre_unique <- merged %>% select(air_genre_name) %>% unique %>% unlist
genre_unique

graph_list <- list()

plot_genre <- function(i){
  genre_specific_sum <- genre_sum %>% filter(air_genre_name==i)
  genre_train <- genre_specific_sum %>% filter(visit_date <='2017-02-01')
  genre_test <- genre_specific_sum %>% filter(visit_date >'2017-02-01')

  m <- arima(genre_train$visitors, order=c(2,1,2), seasonal= list(order=c(1,1,1), period=7))
  y_pred <- forecast::forecast(m, h=80)
```

```
  plot(ts(genre_specific_sum$visitors), main=i, xlab='Year', ylab='visitors')
  lines(y_pred$mean, col='red')
}

par(mfrow=c(3,1), cex=0.7)
plot_genre(genre_unique[1])
plot_genre(genre_unique[2])
plot_genre(genre_unique[3])


par(mfrow=c(3,1), cex=0.7)
plot_genre(genre_unique[4])
plot_genre(genre_unique[5])
plot_genre(genre_unique[6])

par(mfrow=c(3,1), cex=0.7)
plot_genre(genre_unique[7])
plot_genre(genre_unique[8])
plot_genre(genre_unique[9])

par(mfrow=c(3,1), cex=0.7)
plot_genre(genre_unique[10])
plot_genre(genre_unique[11])
plot_genre(genre_unique[12])

par(mfrow=c(2,1), cex=0.7)
plot_genre(genre_unique[13])
plot_genre(genre_unique[14])
```