

Statement of Problem

For this exercise, we were asked to participate in the Recruit Restaurant Forecasting Kaggle competition. The goal of the competition is to forecast the number of visitors that each of 829 restaurants in Japan will have over 39 days following the time period covered by the training data. The data set contained historical data of per date, per restaurant number of guests, data on reservations made through different systems, restaurant cuisine, location, and holiday information. Given the short time available to work on this, and the nature of this course, I confined my efforts to working only with the data set of per day per restaurant visitors.

Significance

I would describe this problem as a form of demand forecasting. Specifically, we are trying to anticipate and plan for labor, materials and capacity needed to meet a demand for the service (dinner at the restaurant) over the next 39 days. From the perspective of time series analysis, this problem is interesting in that we were given less than 2 years of data. So, many time series analysis tools for seasonality would not work on a series created using an annual period as the frequency. Also, we are told that the data span the period called “Golden Week” which is the longest vacation stretch of the year for many Japanese. (Various, 2017) Vacation time-off is likely to have an impact on the number of people eating out, so the combination of factors made for an intriguing, real-world problem.

Data – Exploration and processing

The competition sponsors provided 7 data files, all in CSV format. There were data files from two different reservation systems. Each system provided two files, one on reservation data and one on store data (cuisine, location) for a total of 4 “reservation system” files. There was a file on date information covering day of the week and whether it was a holiday, a mapping file to permit aggregation of the data, and finally, the main file I worked with, a per-date break down of the number of visitors to each of the restaurants. In the main file, we are given 252,108 observations of 3 variables; store id and visit date are in strings, the number of visitors was an integer. All of my data exploration was done using this main file.

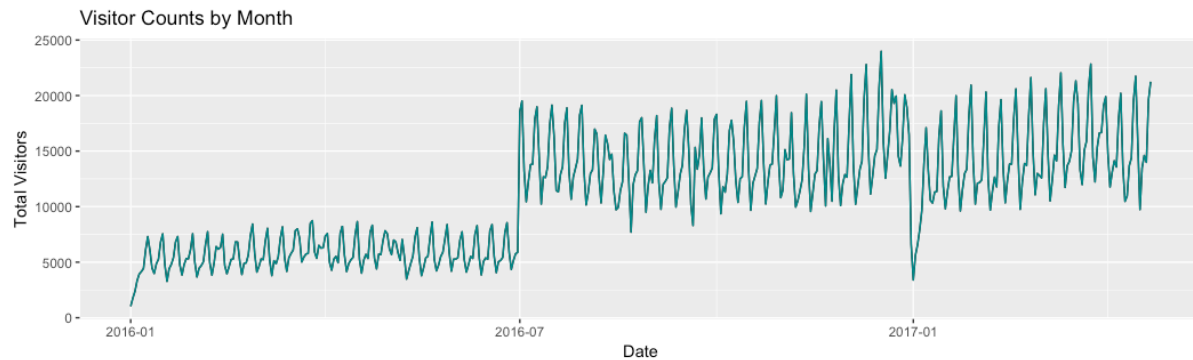
The data we are trying to predict are the number of visitors. The aggregate historical data look like:

Table 1

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.00	9.00	17.00	20.97	29.00	877.00

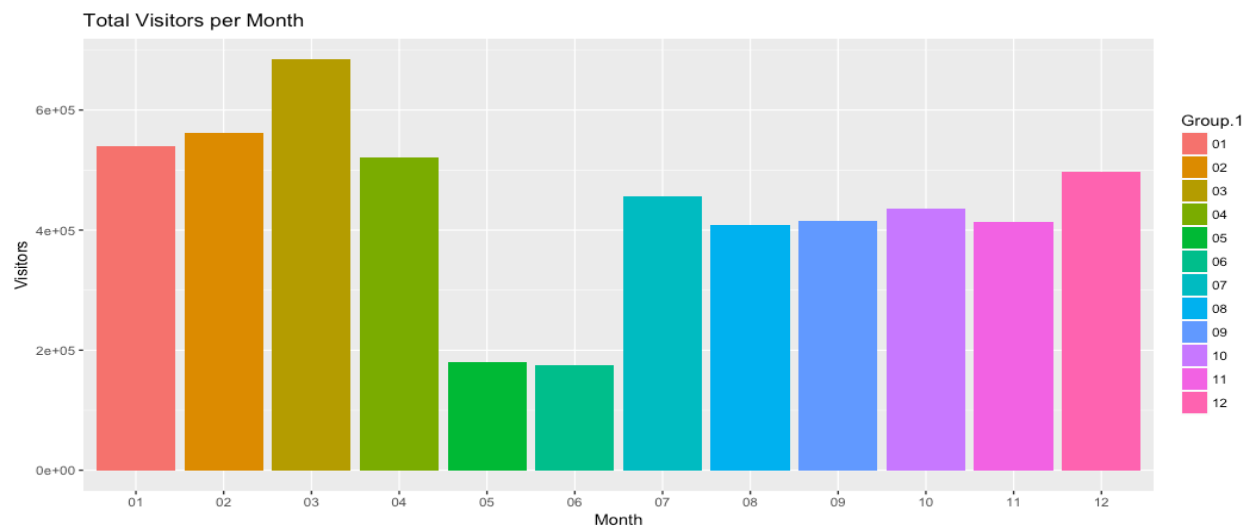
As we can see, there is a huge separation between the 3rd quartile and the maximum value. First, I looked at the number of visitors by month to look for any sort of obvious trend (Figure 1).

Figure 1



Then I looked at the aggregate count of visitors by month. May and June have a depressed number of visitors relative to other months.

Figure 2



I looked at the boxplot of visitors and $\log(\text{visitors})$ by month. I tried the $\log(\text{Visitors})$ to see if there were trends masked by the compressed scaled on the regular boxplot. March, April, July and August show outliers, which given the scale are likely influential observations. The “Golden Week” holiday period is in April to May which does not correspond to the most extreme outliers. The mean of the $\log(\text{visitor})$ counts doesn’t indicate any clear quarterly seasonality.

Figure 3

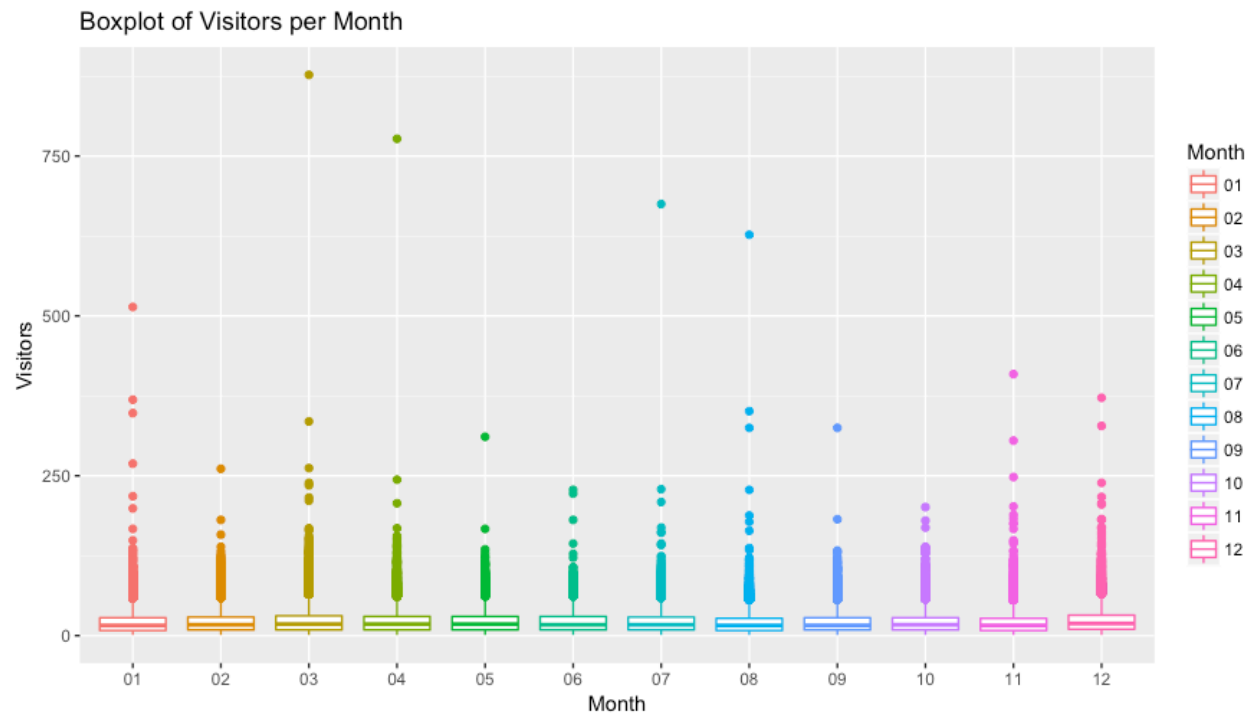
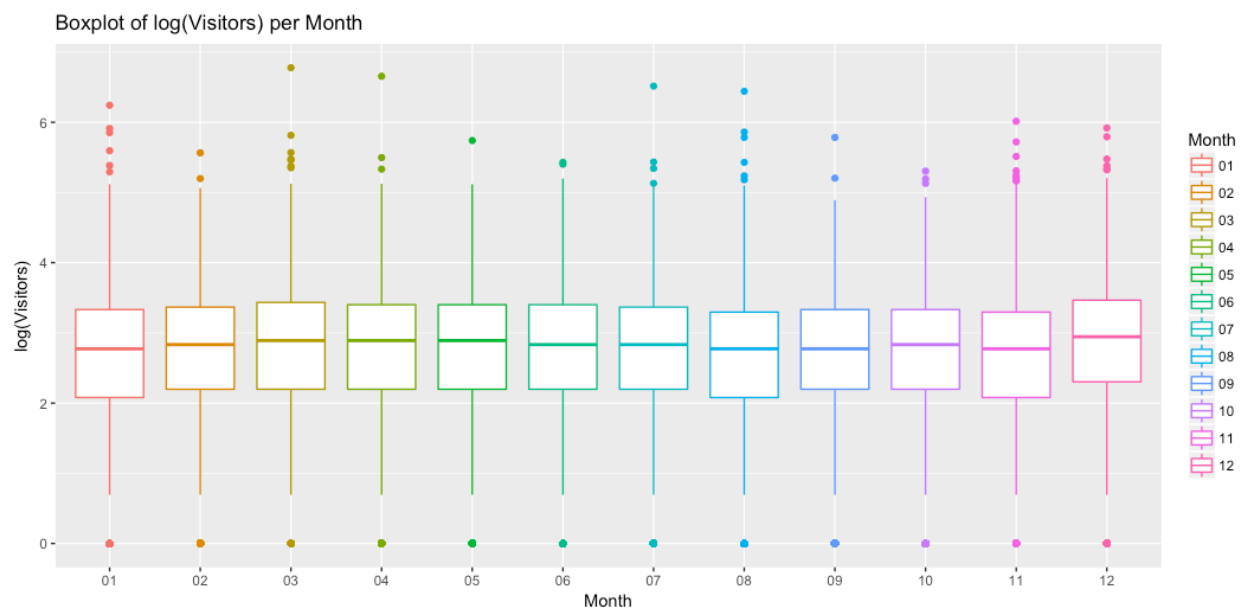
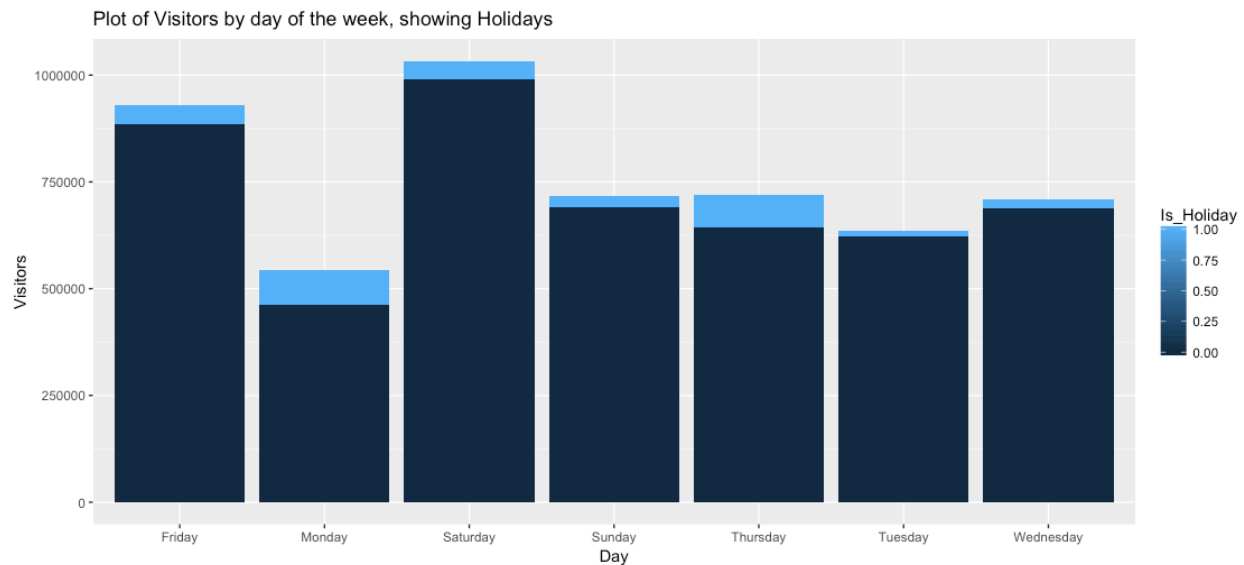


Figure 4



Finally, I looked at the total number of visitors, by day showing the difference for holidays.

Figure 5



Mondays and Thursdays seem to get more of a bump in visitors when they are holidays than do other days of the week.

Ultimately, I tried 2 different approaches to processing the dataset. First, I took the raw file of daily visitors per restaurant and used *dcast* to morph it from “long” to “wide” data. What I did not realize immediately, was that this also had the effect of removing NA values, since the aggregation used a sum of the visitors; basically, missing data became 0s. I built a number of models this way and burned thru a number of my daily Kaggle submissions before realizing this might not be optimal. The second method I used was to iterate through the restaurants, starting the time series at the first date with non-0 visitors, and experimenting with different imputation methods. Using the second method of processing the raw data, I saw there were missing values. VIM summarized them as:

Table 2

Variable	Count
Visitors	0.3635656
air_store_id	0.0000000
visit_date	0.0000000

Visit date and Store ID had no missing values, so only number of visitors needed imputation. I tried the “mice” package for imputation, but it didn’t seem to have much of an impact on the outcome and it is slow to run. For many of the models built using the second method, I simply

stuck with using the per-restaurant mean. Had I started earlier, I could have done more interesting feature engineering, making use of the day of the week and holiday data. As it was, I only made use of the visitors per day data.

Literature Review

I began my search by looking for articles dealing with demand forecasting. The first that I found interesting deals with forecasting tourism in Paris (Gunter & Önder, 2015). In the article, the authors compare the predictive power of 7 different univariate and multivariate models. Their data came from several different countries and was monthly in frequency. RMSE and MAE were used to evaluate model performance. Among the models they looked at were ARIMA(1,1,1), ETS(A,N,N) and naïve, which relate to the material we've covered. The ARIMA and ETS models came in at the bottom of the study rankings, naïve was used as the benchmark.

Continuing in the demand forecast vein, I looked at a study on forecasting bus demand in a suburban area (Cyprich, Konečný, & Kilianová, 2013). This study focused on use of the ARIMA model. The data were comprised of a monthly time series of the total (aggregate summed) passenger demand. The models were manually designed. The ARIMA(0,1,0)(0,1,1) where Q (the seasonal moving average order) was logarithmically transformed was selected by the authors as the model worth additional investigation.

Curious about the seasonal complexity created by "Golden week" I began to look for material that addressed daily data with more complex patterns. This led me to a piece on complex seasonal patterns (De Livera, Hyndman, & Snyder, 2011). In this article De Livera and Hyndman make the point that most time series models are designed to work with simple seasonal patterns with small integer frequencies (12 – annual, 4 – quarterly). These tools are not likely to work well with data having multiple seasonal patterns. The authors lay out a case for the use of TBATS modeling which is a trigonometric versions of a double seasonal Holts-Winter additive model (a.k.a. BATS) for this more complex data. The BATS acronym stems from the use of BoxCox, ARMA errors, Trends and Seasonal components.

The ARIMA-12 model is also supposed to handle seasonal adjustments (Findley, Monsell, Bell, Otto, & Chen, 1998). Nearly two decades old, this article is not as relevant to what we are doing in the class. The article talked about implementations in FORTRAN. However, since it was mentioned in passing in our text, so I wanted to read a bit more about it. ARIMA-12 is supposed to handle moving holidays, like Easter; moving holiday forecasting techniques are highly relevant to our restaurant dataset.

The final article I found, which seemed especially relevant, studies forecasting time series with intraday frequency (Taylor & Snyder, 2012). While the data set I worked with did not include time of day data, other files within the data suite did. It could be theorized that time of day cycles are also involved in our data problem. In the article, the authors study the use of two different models in exponential smoothing intraday data and look at the effect of modifying the models to use an additional kind of smoothing: parsimonious exponential seasonal smoothing.

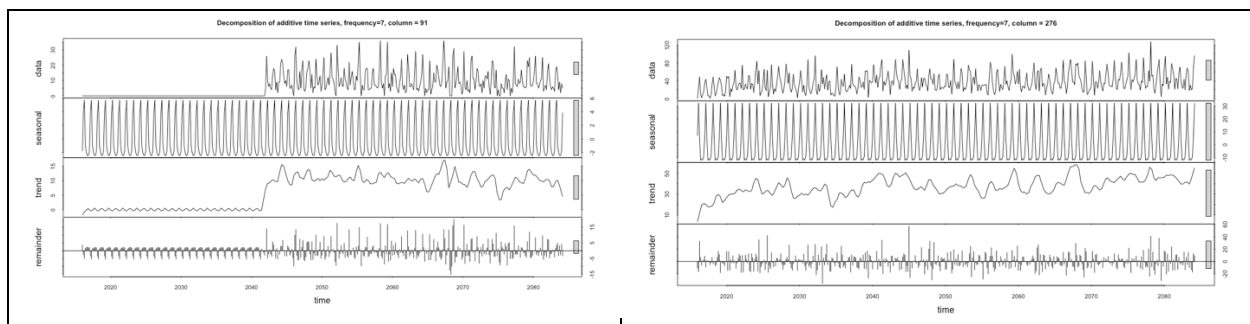
One model is a variation on the Holt-Winters method (HWT), and the second is focused on intraday cycles (IC). The HWT model is described as using the same cycle for all days of the week and requiring a lot of initialization; the IC model allows for different days of the week to have unique cycles. The authors then extend the models to use parsimonious exponential seasonal smoothing, which improved both the IC and HWT model performance. For our data, these techniques looked highly promising, but I don't have the time (or most likely, the math chops) to implement this from scratch.

The Models – Formulation, Performance/Accuracy, Limitations

Formulation

Since the text for the class uses R, I elected to use that as my development language. After my graphical exploration of the data, I looked at the STL data for random restaurants to get a feel for the data. An example of the STL graphs are given here:

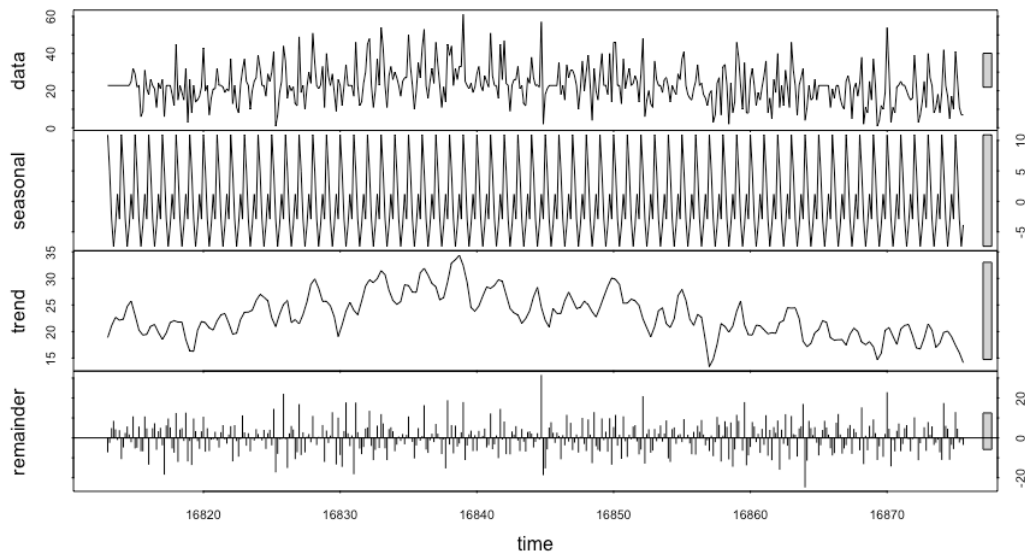
Figure 6



From the charts, it is clear that some stores have no data at the beginning of the raw data time period. I began to evaluate the performance of various models we covered in class. Evaluations were done against the data for the first restaurant in the dataset, which I refer to as Store 1. I used Store 1 as a fixed basis for all comparisons. As I found a promising looking model, I would try alternate data cleaning techniques, or alternate model parameters to see if I could make any improvements to my Kaggle score. Because of the opportunity for head-to-head model comparisons, I built far more models than are probably needed; I learn best through experimentation, so I felt it was a worthwhile effort.

Looking at Store 1's STL when the time series period is set to 7, I saw what looked like a clear seasonal trend; it makes intuitive sense that there could be a weekly pattern to eating in restaurants. It also looked as if there could be a downward trend.

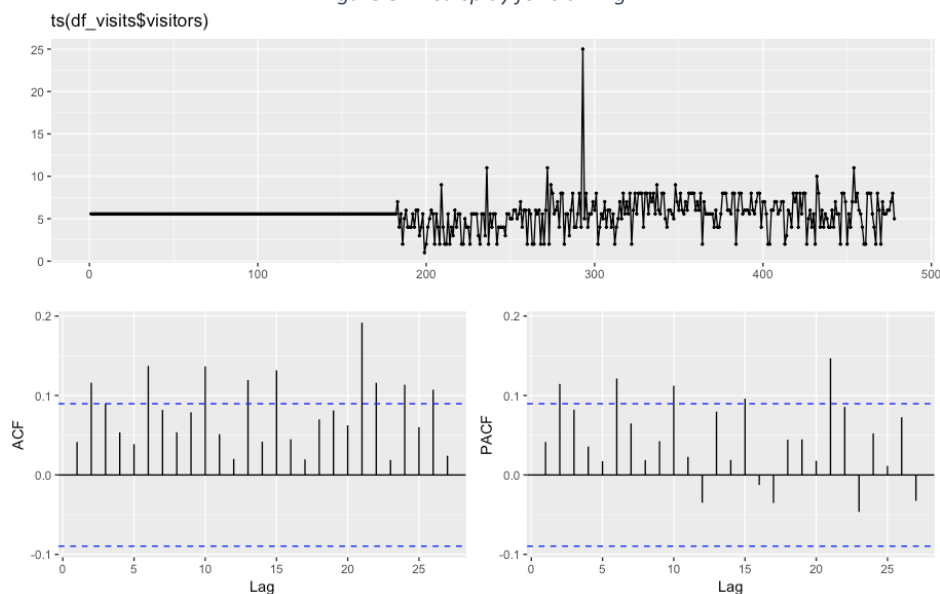
Figure 7 - STL Store 1



From this I concluded I needed to use a seasonal value of 7, and experiment with trend values.

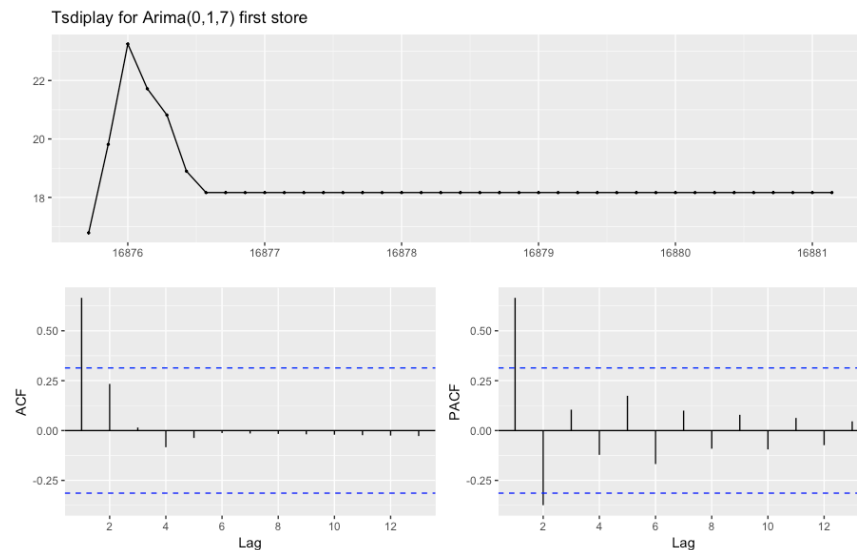
After my literature review, ARIMA and Holt-Winters seemed like good choice to experiment with. Our data may have intraday patterns; a variation on Holt-Winter (Taylor & Snyder, 2012) worked well in that study, so I tried some Holt-Winter models. ARIMA was mentioned in several of my sources, (Cyprich, Konečný, & Kilianová, 2013) (Gunter & Önder, 2015) as was BATS (De Livera, Hyndman, & Snyder, 2011) so I tried each of those as well. Initially, a Holt-Winters Multiplicative model gave the best results in this exploration; it was eventually surpassed by a Random walk with drift model. I was convinced from my reading that there should be and ARIMA model that outperformed both, so I tried ARIMA combinations.

Figure 8 – Tsdisplay for training



Using the `tsdisplay` command to view the full data set, I noticed spikes on the ACF starting at 2 and occurring on the later even numbers, I inferred from this that there are seasonal components. I methodically tried a number of the ARIMA settings; starting with ARIMA model (0,1,1). Because of the 7-day patterns in the data, I also tried ARIMA (0,1,7) which looked better than (0,1,1) and was an improvement in my Kaggle score.

Figure 9



The PACF for (0,1,7) still showed a spike at 2, so I experimented with adding a seasonal term. I iterated over a number of combinations and ultimately found ARIMA(1,1,7)(1,2,7), which was my best model.

Performance and Accuracy

For reference, a summary of most of the models I built is shown in Table 3 in the appendix. My best model had the following results:

Figure 10

```
ARIMA(1,1,7)
Coefficients:
    ar1      ma1      ma2      ma3      ma4      ma5      ma6      ma7      sar1      sma1      sma2      sma3
-0.9834 -0.2568 -0.5623 -0.3303 -0.3646  0.0404 -0.0133  0.4928 -0.9834 -0.7136 -1.6873  1.3099
s.e.      NaN      NaN      NaN      NaN      NaN      NaN      NaN      NaN      0.1315  0.1378  0.2686
    sma4      sma5      sma6      sma7
  0.8457 -0.7173 -0.1513  0.1138
s.e.  0.2265  0.2613  0.0997  0.1103

sigma^2 estimated as 95.02: log likelihood=-1620.2
AIC=3274.4  AICc=3275.87  BIC=3343.72

Training set error measures:
              ME      RMSE      MAE      MPE      MAPE      MASE      ACF1
Training set -0.3544203 9.534427 7.262434 -39.97978 59.82703 0.6968517 -0.001915235
```


Figure 11

	ME	RMSE	MAE	MPE	MAPE
Test set	1.038847	10.56571	8.617965	-110.5635	143.0594

Figure 12

```

Box-Pierce test

data: fcast117_127
X-squared = 8.2534, df = 1, p-value = 0.004068

> Box.test(fcast117_127, type = 'Lj')

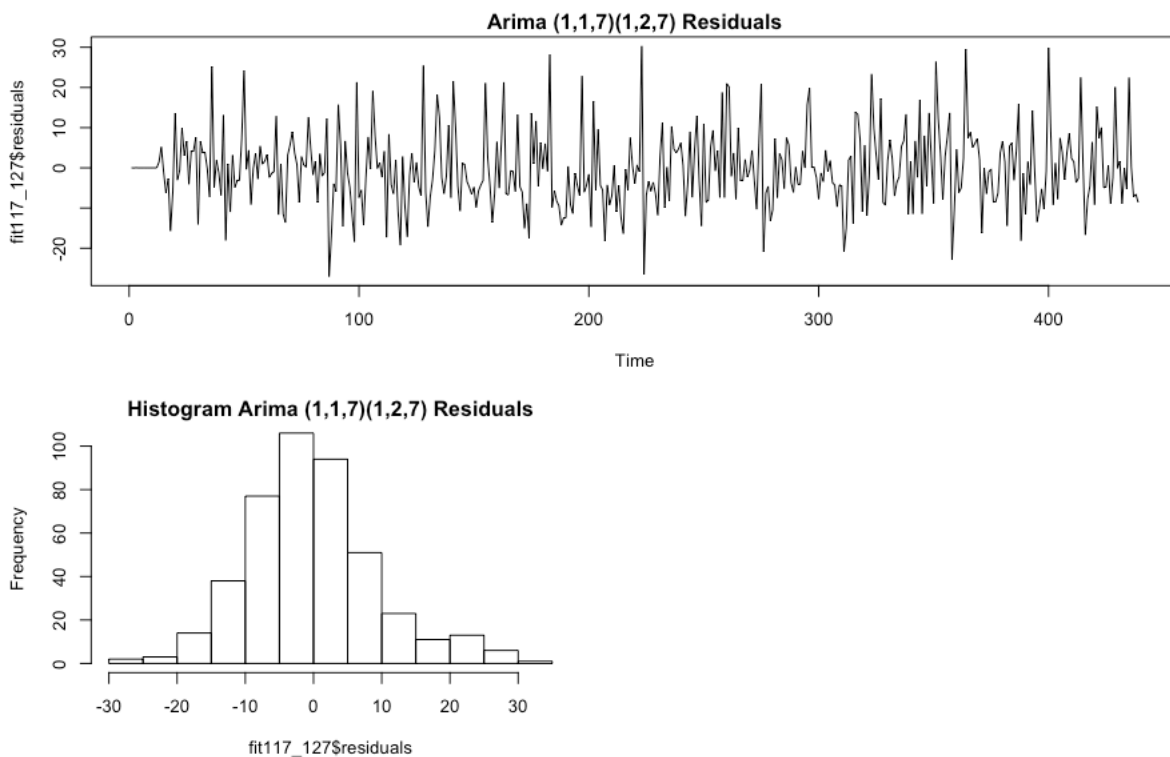
Box-Ljung test

data: fcast117_127
X-squared = 8.905, df = 1, p-value = 0.002844

```

The Kaggle ranking (a late submission) was .830 Private/.886 Public (see Appendix - Figure 14), the AICc was 3275.87, compared to 3285.41 for ARIMA(0,1,1). The residuals look good, they appear normally distributed. The portmanteau results give p-values less than .05 indicating the residuals are independent. A graphical view of the residuals show them to be normally distributed.

Figure 13



While this model had the best Kaggle rating, it did not have the best RMSE or MAE scores. An ETS model which trimmed datasets to start on the first non-0 day performed the best on those measures, at least for Store 1 (see Appendix – Table3). The graphical results for this model are shown in Figures 14 and 15. The ACF and PACF charts show that except for the first spike, all other bars are within the boundaries.

I want to note that due to a copy/paste error, this model was actual built using the training data set, which is comprised of the first 440 observations, the last 39 being held out for testing. I reran the model using the full data set and it did not perform as well. I infer from this that there are influential outliers in those last 39 observations and conclude more work could be done on data cleaning.

Figure 14

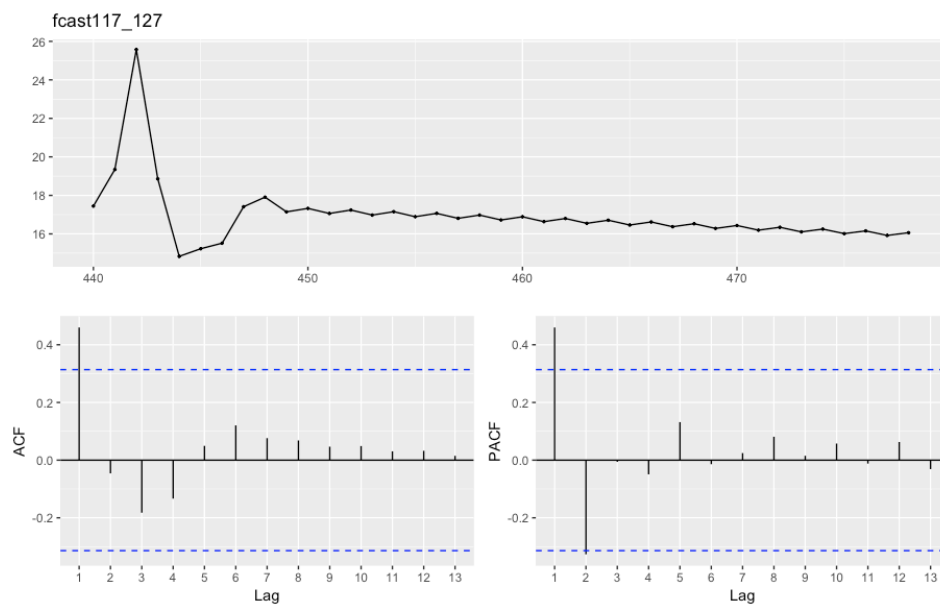
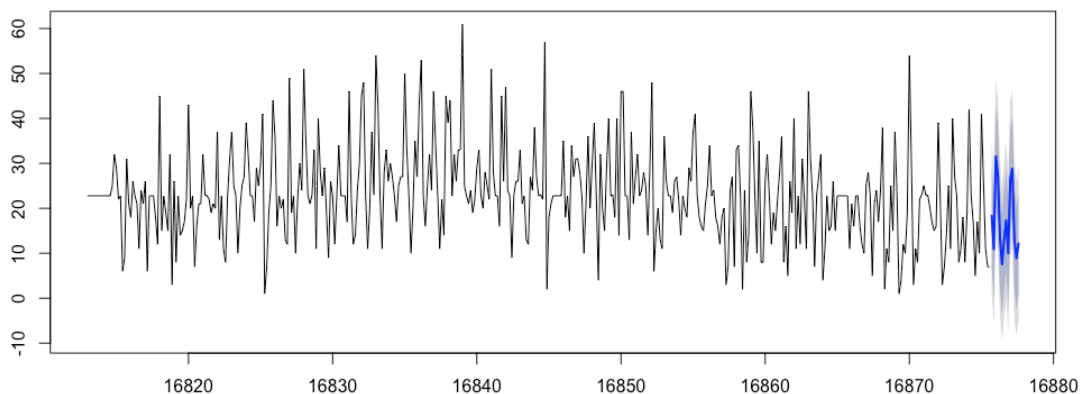


Figure 15

Forecasts from ARIMA(1,1,7)(1,2,7)[7]



Limitations

Because my model is using a single model for all data, there is a good possibility that it works very well for some restaurants, and poorly for others. If there are unique patterns to a given store, or type of cuisine, that approach is not going to capture that variation appropriately. Also, because I am using Store 1 as a benchmark for looking at the test/training data evaluation I may not be effective as possible since there is no guarantee that Store 1 is a good proxy for the entire set. Also, I believe there are multi-seasonality aspects to this data, such as time of day and day of week and possible annual cycles; I am not convinced that my model fully captures that. Finally, because this data come from one country, Japan, I would not try to leverage this model to other countries without testing its validity in those markets.

Future Work

To improve these models, first I'd like to do a programmatic evaluation of models per store and produce an ensemble of the best per-store results, rather than the aggregate of all stores using a single model. I'd like to spend time incorporating the additional Kaggle data. For example, make use of the type of cuisine, day of the week, and whether a date is a holiday to enrich the model. For example, if it is Monday, impute missing Monday values with Monday-specific means; if Monday is a holiday, add a holiday "bump" in visitors to the Monday mean. Are there patterns in cuisine consumption? Perhaps in Japan, Italian food is primarily eaten on weekends, there might be additional patterns to be found there. I also would like to spend some time working thru the underlying math of the ARIMA model to see if I could find better non-seasonal and seasonal values to input. Finally, after reading about TBATS, I believe a much better model than the one I came up with is possible, I just need to spend more time understanding the parameter selection criteria better. Finally, the data looked to me like there are multiple "seasons" or periods as well as the weekly rhythm. There is a clear weekly rhythm, but I suspect a broader annual/quarterly cycle may exist as well. Handling these sets with multiple possible views of the season would be interesting to try.

Learnings

I class my learnings from this exercise into three categories: what I learned about time series, what I learned about R, and what I class as personal/professional learnings.

Time Series Learnings

From the literature search I gained a better appreciation for how powerful these techniques are. The techniques we are learning in class are the same as many of the ones in the papers I reviewed, which investigated some interesting and complex issues. My experimentation with so many models allowed me to see how small changes in parameters to a model could have a large impact. I also gained an appreciation for the difference in results one can obtain based on the handling of missing data and outliers.

By the end of the exercise I felt I gained a better understanding of how to evaluate the model residuals and measures to aid in selection. Auto.arima did not perform as well as I expected which was a surprise; the simple Random walk with drift outperformed many ETS and ARIMA models, which was also a surprise.

R learnings

I learned a lot about how different packages handle NA values. For example, I experienced a short surprise when I realized that using `dplyr::dcast` to group the data cast my missing values to 0s. I was also surprised when I figured out that the `dplyr::filter` function I used dropped NA values by default. I was disappointed in the results I got using 'mice' to impute missing data had no significant improvement over using the mean. Too late to use it for this paper, I found a faster way to impute the mean using 'Hmisc'. There is a wide range of variety in what R packages do; I learned I need to read the documentation more closely for things like dropping (or auto-imputing) NA values. More packages automatically did things to NAs than I realized.

Personal/Professional Learnings

First off, I have not done much with Kaggle since joining the MSPA program other than a few targeted experimental submissions to try out a new learning from a class. I was not prepared for how caught up I got in the game aspect of the competition. For the final I need to reflect more and submit less.

I need to find a better workflow. I tend to get very focused on producing a volume of code, and my classic tester habits of "what happens if I try *this*?" aren't necessarily the best for this sort of problem. I need to stop thinking like a tester and more like a data scientist. Finally, I should remember what I told every tester who ever worked for me: the second time you do something in code, make a function. I did a lot of copy/pasting in this; I am aware it makes the R code hard to read. For ease of reviewing the code, I split the R into the relevant models I discuss in this paper, and everything else.

Appendix

Works Cited

- Cyprich, O., Konečný, V., & Kilianová, K. (2013). Short-Term Passenger Demand Forecasting Using Univariate Time Series Theory. *Promet – Traffic & Transportation*, 533-541.
- De Livera, A. M., Hyndman, R. J., & Snyder, R. D. (2011). Forecasting Time Series With Complex Seasonal Patterns Using Exponential Smoothing. *Journal of the American Statistical Association*, 1513-1527.
- Findley, D. F., Monsell, B. C., Bell, W. R., Otto, M. C., & Chen, B.-C. (1998). New Capabilities and Methods of the X-1 2-ARIMA Seasonal-Adjustment Program. *Journal of Business & Economic Statistics*, 127-152.

Gunter, U., & Önder, I. (2015). Forecasting international city tourism demand for Paris: Accuracy of uni- and multivariate models employing monthly data. *Tourism Management*, 123-135.

Hyndman, R. J., & Athanasopoloulos, G. (2014). Forecasting Principles and Practice. OTexts.com.

Taylor, J. W., & Snyder, R. D. (2012). Forecasting intraday time series with multiple seasonal cycles using parsimonious seasonal exponential smoothing. *Omega*, 748–757.

Various. (2017, Dec 5). *Golden Week (Japan)*. Retrieved from Wikipedia:

[https://en.wikipedia.org/wiki/Golden_Week_\(Japan\)](https://en.wikipedia.org/wiki/Golden_Week_(Japan))

Table 3 – results of all models

Model (in the order they appear in the code)	Kaggle Score (private)	RMSE (test)	MAE (test)	MPE (test)	MAPE (test)	MASE(test) (if given)	ACF1 (test) (if given)
ARIMA(1,1,7)(1,2,7)	0.830	10.568	8.618	-110.564	143.0594	-	-
Basic ETS, frequency=365	1.065	10.013	14.003	-Inf	Inf	-	-
Random Walk with drift	0.846	16.922	13.857	-Inf	Inf	1.056	0.254
Naïve	2.942	17.721	17.544	-Inf	Inf	1.338	0.437
ETS frequency=7 (weeks) A,N,N was auto-selected	1.226	10.167	7.714	-Inf	Inf	1.218	0.018
auto.arima	1.064	14.175	12.02	-Inf	Inf	1.423	N/A
auto.arima, seasonal=FALSE	1.063	19.024	15.901	-Inf	Inf	1.423	N/A
Holt-Winters multiplicative on dcast data	1.103	10.363	7.969	-Inf	Inf	1.258	0.003
TBATS	2.28	10.855	8.972			0.414	0.086
TBATS with altered data handling	2.187	10.01	7.862	-86.995	124.9211	-	-
auto.arima, mice for imputation	1.032	10.805	8.703	-135.846	163.3643	-	-
ETS with mice imputation	1.022	8.07	6.414	-112.969	129.6997	-	-

ETS, imputation=mean, starting at 1st non-0 (Best RMSE and MAE)	1.024	8.059	6.383	-112.312	130.6682	-	-
Holt-Winters additive	0.913	8.059	6.383	-112.312	130.6682	-	-
ETS trying transformation of visitor numbers	1.026	15.013	14.45	-604.854	604.854		
Holt-Winters multiplicative (Train data)	0.881	8.886	7.062	-80.0784	113.24628	0.659	N/A
Holt-Winters multiplicative - damping	0.88	8.505	6.802	-91.989	119.09536	0.635	N/A
ARIMA(0,1,1)	0.842	10.203	7.611	-45.695	65.520	0.9179	0.07171
ARIMA(1,1,1)	0.903	}					
ARIMA(0,1,0)	1.113		Failed to	grab	these	results	
ARIMA(1,0,1)	0.903						
ARIMA(0,1,1)(0,1,1)	0.923	8.062419	6.395487	-35.628	54.755	0.7713	-0.0035
ARIMA(0,1,2)(0,1,1)	0.923	8.062	6.316	-88.0569	112.9416	-	-
ARIMA with winsorized data	0.831	8.7307	7.5661	-30.555	58.6658	-	-

Figure 16 – Kaggle submission view, best during competition


2026	▼ 605	Smuch		0.879	9	17d
2027	▲ 25	Twill		0.879	20	3d
2028	▲ 27	phani		0.883	4	2mo
2029	▲ 13	yangzhou123		0.884	2	3d

Figure 17 - best submission overall

61 submissions for Twill		Sort by Most recent	
All Successful Selected			
Submission and Description		Private Score	Public Score Use for Final Score
TW_Arima_117_127.csv a few seconds ago by Twill add submission details		0.830	0.886 <input type="checkbox"/>
TW_Arima_111.csv 2 hours ago by Twill		0.903	0.918 <input type="checkbox"/>