Optimal Inventory Redistribution

Calvin Corey, Ian Neville, Kaisa Crawford-Taylor, Wanqian Huang

May 8th, 2017

Abstract

Fastenal spans North America with over 2,600 locations, making them the largest fastener distributor in the country. However, such expansiveness caused inventory distribution imbalances within the company. This project works towards optimizing Fastenal's existing inventory using R, Excel, and mySQL. Through the combined use of linear regression, time series, and our own advanced time series model, we are able to forecast future sales using past product sale history. After forecasting future sales we will evaluate at which retail locations products will sell best and relocate the products to those stores. This projects focuses on redistribution in Fastenal's smallest warehouse which has 9 retail locations.

Introduction

Fastenal is a wholesale distributor, retailer, and manufacturer of industrial tools and components. Their business model is structured around regional distribution centers called "hubs." These hubs distribute products to either wholesale customers or to their retail locations called "branches."

Currently, some branches have a surplus of items while others do not have enough. Our goal in this project is to redistribute Fastenal's existing inventory to solve this problem. We will use given data to forecast future sales for different branches to determine the best location for products.

Our Approach

We approached this problem by first exploring our data. Due to computational limitations, we were only able to analyze a subset of the data, the data set we received contained 14 hubs and over 2,600 branches, so we narrowed our scope to Fastenal's smallest hub: A hub of 9 branches.

After subsetting the data, we cleaned and reorganized the sales history table. This prepared the data to be ran through forecasting models.

We used a combination of Linear Regression, Time Series, and our own

Advanced Time Series Model. We ran the data through each of these models experimentally, whilst filtering out good and bad patterned predictions along the way.

After gathering predictions, we began to tackle the redistribution question. We use the forecasted sale intervals to determine which branches will sell a product well and which branches will not. Our goal is to transfer products to the branches where they will do best.

Given Data [1]

We were given the following tables by Fastenal:

- Products - Product information for Fastenal products
- Stocking and Inventory - Snapshot of stocked products for all branches
- Hub Hierarchy Table - data on past hub->branch stocking decisions
- 4 Years Consumption Table - 4 years of sales data for Fastenal products

These tables were given as '.csv' files containing several million lines of data each. (e.g. the size of the Consumption Table was over 146 million lines). Due to amateur processing power, we needed to reduce the data.

Data Reduction

We had to reduce the data for two main reasons. The first being hardware restrictions, the second being data encryption. We do not have access to the location data for any of the branches or hubs. Hubs could be within a district of each other or across the country. When redistributing items, shipping costs must be taken into consideration, so we cannot suggest shipping items between hubs without knowing their locations. However, because hubs stock their branches, we know that branches within a single hub are near each other. This is why we subset the data to a single hub.

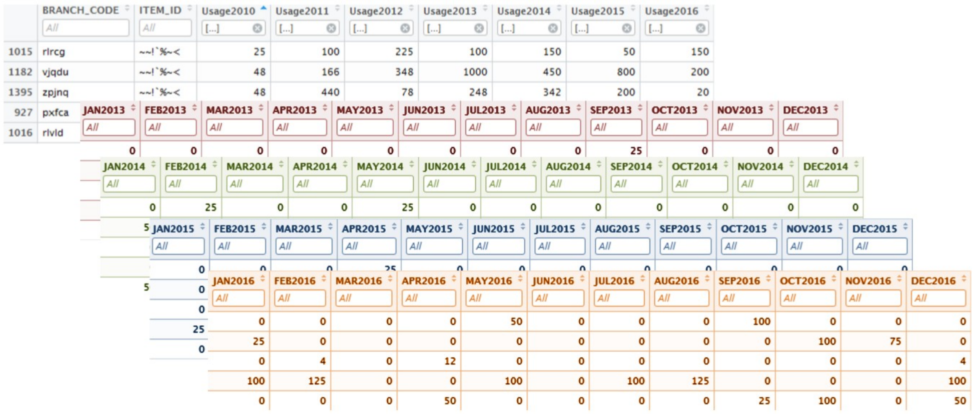
Deciding which hub to look at was done using the Hub Hierarchy Table. We found a hub containing only 9 branches. This hub will be referred to as the Smallest Hub, because Fastenal's hubs average at 171 branches per hub. We used a single hub as model, and provided the right amount of hardware, we can expand our findings and models to encompass any of Fastenal's hubs.

By focusing on only the Smallest Hub, we reduced our data set from millions of lines to a little over 100,000. We also further reduce the data utilizing linear regression, this will be explained in more depth when we talk about Linear Regression.

Reorganization

After reducing the data, we were able to reorganize the Consumption Table. We had to reformat the data because the Consumption Table was structured to have each year on a separate line. While 12 month lines are usable for Linear Regression analysis, we could not use this format for Time Series or Advanced Time Series. This is because those models require the data to be in a consecutive line.

The first thing we did to reorganize the data is fill the missing usage month. Every year we were given only had 11 Usage Months. However, we were able to use the sales amount given to us in the Usage Year field to calculate the 12th month. We simply summed the 11 given months and found the difference between that sum and the Usage Year value to be how much Fastenal sold in the 12th month. After we had all 12 months, we converted the table from 12 month lines to 48 month lines by reorganizing the rows. Finally, we converted this data frame into a time series. Below is an image of a snippet of the restructured data table before time series conversion.



	BRANCH_CODE	ITEM_ID	Usage2010	Usage2011	Usage2012	Usage2013	Usage2014	Usage2015	Usage2016
	All	All	[...]	[...]	[...]	[...]	[...]	[...]	[...]
1015	rlrcg	~~!`%~<	25	100	225	100	150	50	150
1182	vjqdu	~~!`%~<	48	166	348	1000	450	800	200
1395	zpjnq	~~!`%~<	48	440	78	248	342	200	20
927	pxfca								
1016	rlvld								

JAN2013	FEB2013	MAR2013	APR2013	MAY2013	JUN2013	JUL2013	AUG2013	SEP2013	OCT2013	NOV2013	DEC2013
All	All	All	All	All	All	All	All	All	All	All	All
0	0	0	0	0	0	0	0	25	0	0	0

JAN2014	FEB2014	MAR2014	APR2014	MAY2014	JUN2014	JUL2014	AUG2014	SEP2014	OCT2014	NOV2014	DEC2014
All	All	All	All	All	All	All	All	All	All	All	All
0	25	0	0	25	0	0	0	0	0	0	0

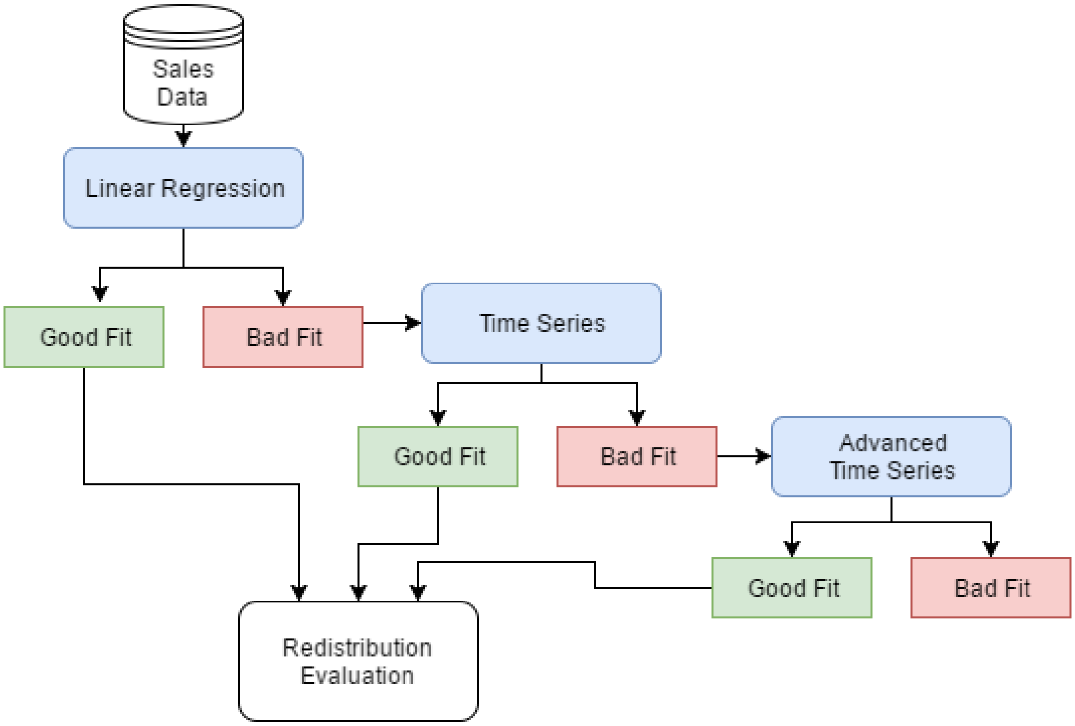
JAN2015	FEB2015	MAR2015	APR2015	MAY2015	JUN2015	JUL2015	AUG2015	SEP2015	OCT2015	NOV2015	DEC2015
All	All	All	All	All	All	All	All	All	All	All	All
0	0	0	25	0	0	0	0	0	0	0	0
0											
0											
25											
0											

JAN2016	FEB2016	MAR2016	APR2016	MAY2016	JUN2016	JUL2016	AUG2016	SEP2016	OCT2016	NOV2016	DEC2016
All	All	All	All	All	All	All	All	All	All	All	All
0	0	0	0	50	0	0	0	100	0	0	0
25	0	0	0	0	0	0	0	0	100	75	0
0	4	0	12	0	0	0	0	0	0	0	4
100	125	0	0	100	0	100	125	0	0	0	100
0	0	0	50	0	0	0	0	25	100	0	50

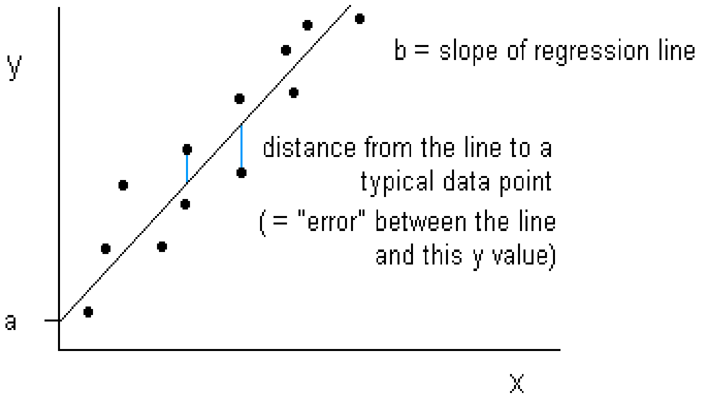
In addition to making the data usable, reorganizing the data helped us figure out which rows had a lot of zero entries. A year or more of zero entries indicate that an item was not being carried by Fastenal at that time. This could be because the item is a recently added item or because it was discontinued. It is important to know which items excess zeros because Linear regression and Time Series can be inaccurate if not given enough data.

Math Models

As mentioned in “Our Approach”, we used a combination of math models to predict future sales. Below is a flowchart of how we used each model complementary to each other.



Linear Regression

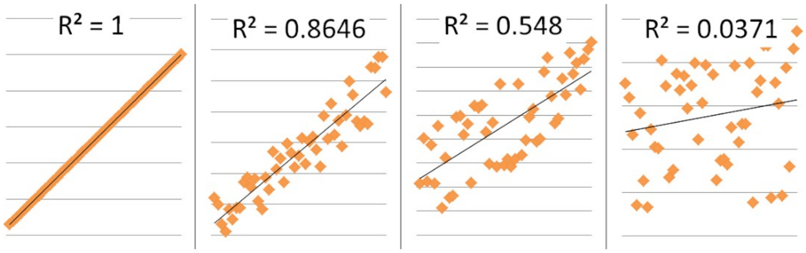


Linear Regression is a mathematical model that allows us to fit a straight line to a dataset, following this line allows us to make predictions about the future. This line is found using an algorithm that minimizes the error between the line and the various data points within the dataset. Once this line is found, we can look at its attributes, the slope and the intercept, to draw conclusions about the data. The intercept: point a in the figure is the starting point for the line, it allows us to shift the line up and down. The slope: *b* in the figure. is a measure of how much the model changes with a 1 unit increase in *x*. There are certain limitations to using this model, and theses limitations affect the model's accuracy. We will discuss these limitations and how we plan to correct for them.

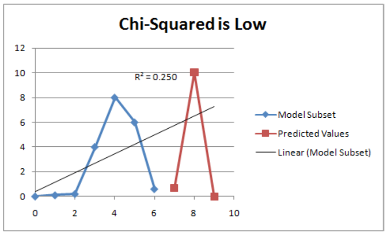
In the context of Fastenal's problem, we are looking at how sales change with respect to time. That is, how much of a specific product do they plan to sell at a given location (branch) at a specific time (month). As mentioned before, we were given 48 Months of Fastenal sales data, and asked to use this data to build a predictive model. So to specify, the slope (*b*) is a measure of how much more or less of a Fastenal product should be sold in comparison to the previous month. For example, if in the month of January, 50 items of a specific product are sold, and the slope/rate of the model for this dataset is 5 (items per month), we can assume that 55 items will be sold in February, 60 will be sold in March, etc.

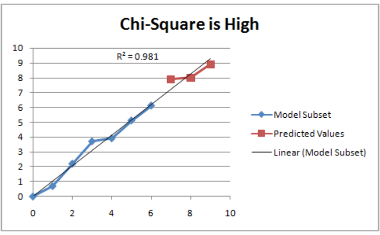
Here, linear regression is only capable of measuring change with respect to time. Therefore, linear regression cannot account for things like seasonal changes in product demand, along with many other things. So we will need methods of measuring model accuracy to see if it is useful.

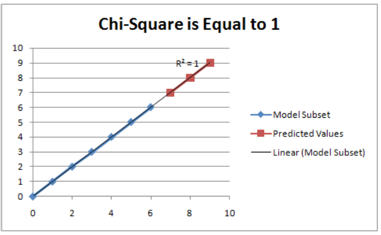
The first method we look at for determining accuracy is Coefficient of Determination, or R^2 for short. This is a measure of the residual error in the model. We can use this as a measure of how good the model fits the historic data set. R^2 takes on values from [0, 1], with 1 meaning that the model is a perfect fit to the dataset. Below is a graphic that illustrates different R^2 values.



The second method we use is the Chi-Square test. This is a measure of how good the model makes predictions. In this test we subset the data into two parts, a training set and a test set. The training set is a section of the data that we build a model from, and the test set is what we use to test how well the model makes a predictions. This is different from R^2 in the sense that it only considers the last few months when determining model fit, which is arguably the most important time to make predictions. Below are graphics illustrating different Chi-Squared values.







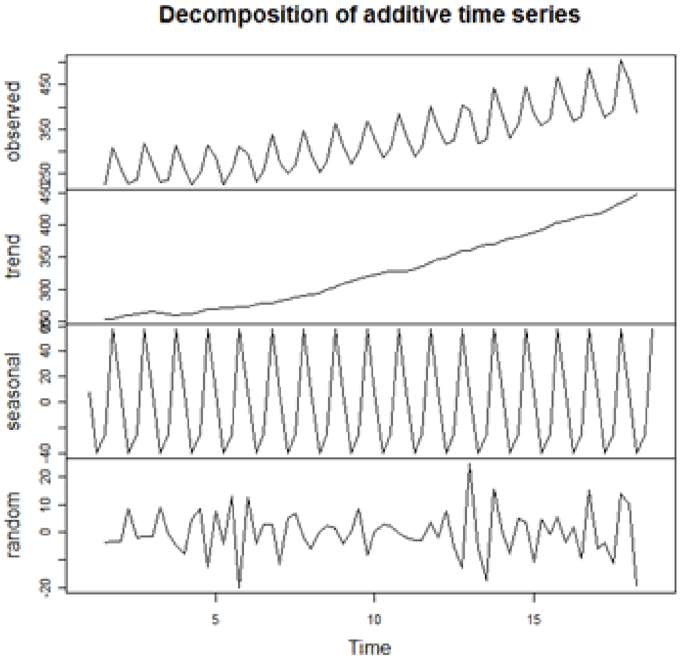
By using these two tests, we determine whether the model is accurate enough in predictions to be useful, if not, we pass along that section of data to a method called time series. This technique is much more accurate in almost all cases. So, why use linear regression? The main reason for using linear regression is that it is a very efficient technique. This algorithm can process large amounts of data in a very short period of time, and without stressing computational ability, we were able to build around 108,000 predictions in only around 15 minutes. Looking specifically at the top 4,000 items we determined that about 25% of these were able to be accurately modeled with linear regression.

It is also important to mention that there is large percentage of items that have 0 sales during many months. Using linear regression, we were able to filter out the items with slopes and intercepts of 0 to Fastenal for using other methods of distribution decisions.

Time Series

As mentioned, Linear Regression is accurate for *25%* of the top 4000 items. This leaves 75% of the data without accurate sales forecasts. Time series can forecast for some of the remaining data by incorporating more complexities. Below is a decomposition of a basic, additive Time Series.

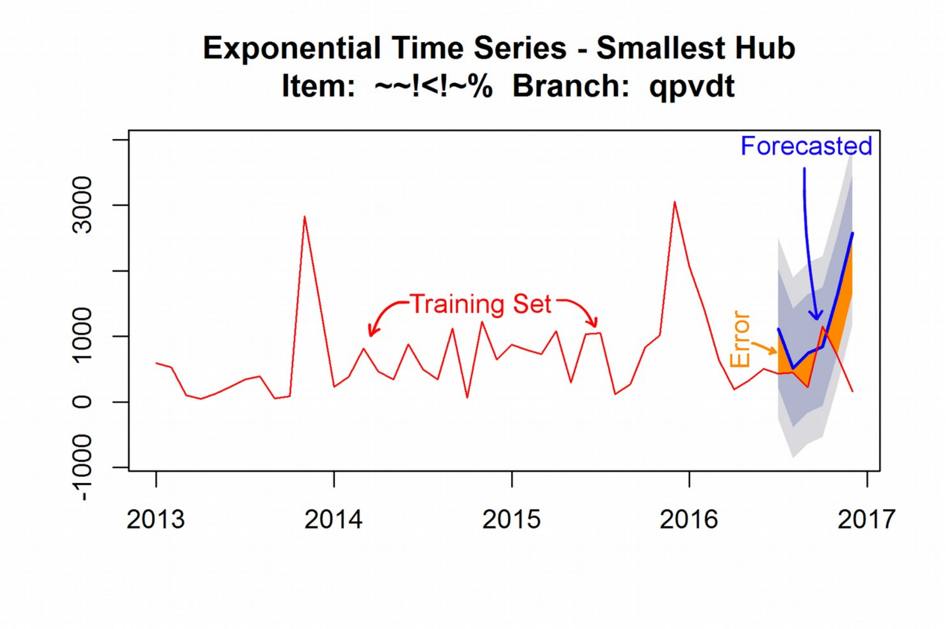
There are four parts to a Basic Time Series model, the observed, trend, seasonal, and random remainder.



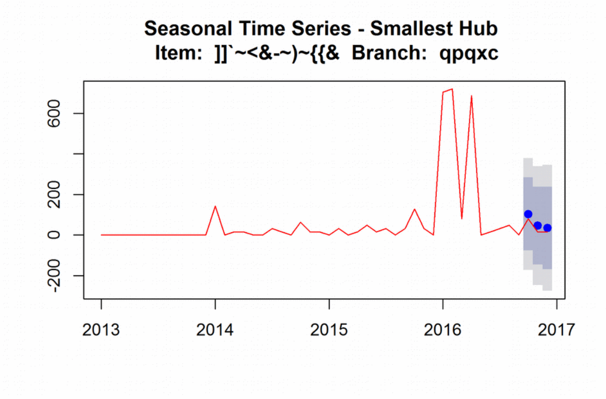
The observed component is simply the amount of sales on the *y-axis* and the month of the sale on the *x-axis*. The observed component can be separated out into the seasonal part and the general trend of the sales. The seasonal component indicates cyclical patterns found in the basic Time Series. The main reason Time Series is more powerful than Linear Regression is because it accounts for seasonality in the sales data.The trend is what is left after the seasonal component is accounted for. This indicates whether a product’s sales are generally increasing, decreasing, or staying the same. The Time Series’ trend is similar to a Linear Regression model. The last part is random noise. This section shows the amount of sales that the seasonal and trend component cannot account for. We want a low amount of noise so that the model and make accurate predictions.

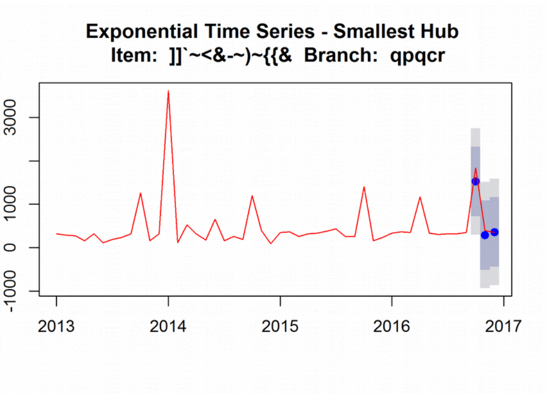
The accuracy of Time Series can be measured in many ways, but all methods involve using training sets to measure error. Training sets are sections of the known data. For example, we were given 48 months of sales data, but we only used 45 months for our training set. We run the basic Time

Series model on just 45 months of data and use the models to forecast the next 3 months based off of the previous 45 month portion. We measure model performance by comparing the predicted values to the actual data we were given, the remaining 3 months: months 46 - 48. The smaller the difference between the numbers means a smaller error. We summed these three months' errors to generate a summed error value. The most accurate Time Series have the smallest summed error values. This concept is illustrated in below (please note, the graph shows a prediction for 6 months while we used 3 month predictions).

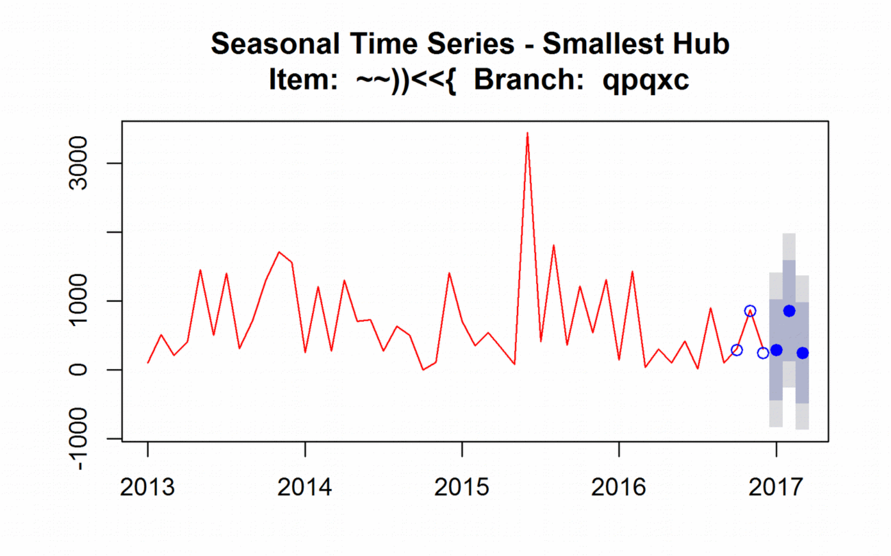


The next figures are examples of a good fitting Time Series from our data for both seasonal and exponential models. It is the same product but for different branches. The red line is our training set and the blue dots are our predictions. More graphs can be found in Appendix[2].

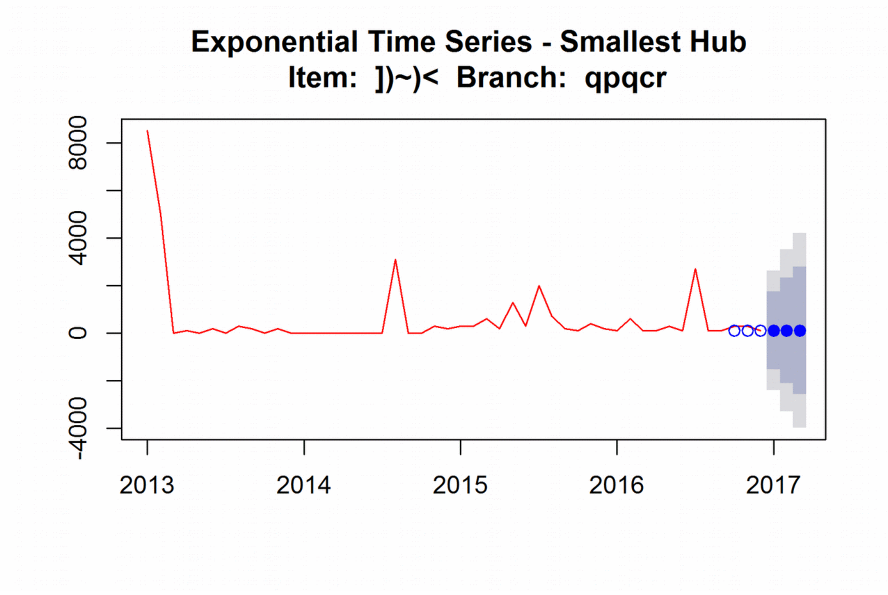




For this project, we used Time Series to predict what Linear Regression was unable to. We ran seasonal and exponential Time Series on the top 4000 items by volume. The Basic Time Series model indeed performed better than Linear Regression. When running the seasonal Time Series model, 1354 items out of the 4000 (*33.85%*) had error under 100 items. For the exponential Time Series model, 1300 out of the 4000 (*32.5%*) had error under 100 items. This means for 3 months, the model only miscalculated up to 100 sales. This could be from over or underestimation. On average, Basic Time Series was accurate for *33%* of the top 4000 items. Below is a seasonal Time Series that fit the training set well. It has future sales predicted for 3 months, denoted by the closed blue dots. The open blue dots are the training predictions.



Below is an exponential Time Series that fit the training set well. It also has future sales predicted for the next three months, denoted by the closed blue dots.



These graphs show that basic Time Series can predict future sales as well as future periods where items do not sell well. Predictions of stagnant sales are as important as dynamic sales because the product can be evaluated to be moved. We will discuss this further in the section about Redistribution.

For the items which could not be accurately modeled by either Linear Regression or Basic Time Series moved on to a model we created called Advanced Time Series. Additionally, Basic Time Series' results have very large confidence intervals. When investing, businesses want a balance between confidence levels and narrow confidence intervals. Advanced Time Series can accomplish this. This model is introduced in Advanced Time Series.

Advanced Time Series

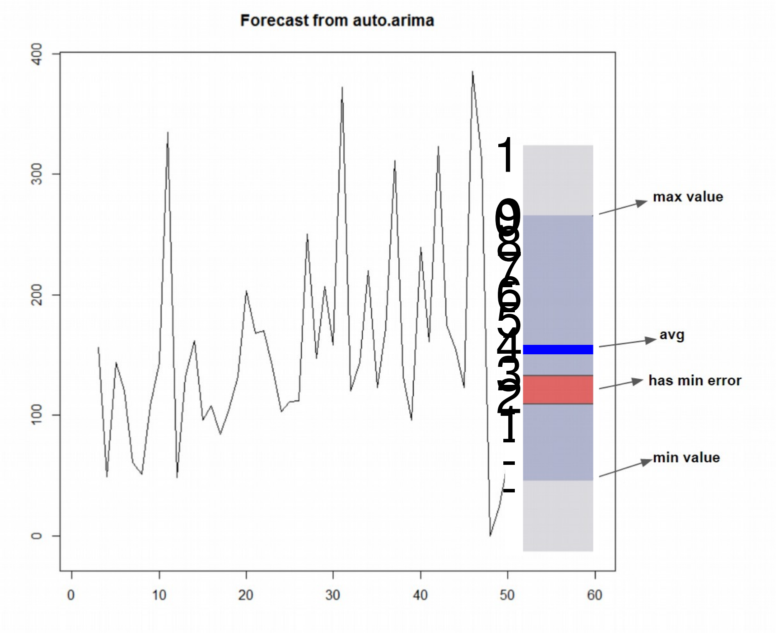
Advanced Time Series is a model we created ourselves for this project. It is able forecast future sales more accurately than Linear Regression and

Basic Time Series. It is more accurate because the confidence intervals are smaller than in Basic Time Series. Overall, the model performs *2.4%* better than Basic Time Series. Most companies use only Basic Time Series, so this percentage gives Fastenal a competitive edge.

We utilize Auto Arima in our Advanced Time Series model. What is an ARIMA model?

In statistics and econometrics, and other analysis that uses time series, an Autoregressive Integrated Moving Average (ARIMA) model is a generalization of the Autoregressive Moving Average (ARMA) model. Both of these models are fitted to time series data either to better understand the data or to predict future points in the series (forecasting). ARIMA models are applied in some cases where data show evidence of non-stationarity, where an initial differencing step (corresponding to the "integrated" part of the model) can be applied one or more times to eliminate the non-stationarity (Wikipedia). The most important component of Auto Arima is that it automatically decides which months to use as a training set. For example, if the data has many zeros in the beginning but a lot of sales data at the end of the row, Auto Arima chooses to only use the ending months for analysis.

What the Advanced Time Series model adds to Auto Arima is it collects the error of the Time Series' forecasts then predicts on the errors. Instead of just using `auto.arima()` to get prediction value. Below is a graph of the error collection. We found that interval from 3 to 4 gets minimum error. 4 can get second minimum error. Therefore, we guess there might be a minimum point between 3 and 4.



The Advanced Model[3] does the following:

1. Collects errors for each month (30 - 48): `P(mean)-Usages`
2. Do prediction on errors: Predict error for 49 month -> `E(max)`, `E(min)`.
3. Get new confidence interval: `P(mean) + E(max)`; `P(mean) + P(min)`
4. Normal Model: `P(mean) = arima()`

Through this process, the Advanced Model is better than Normal Model.

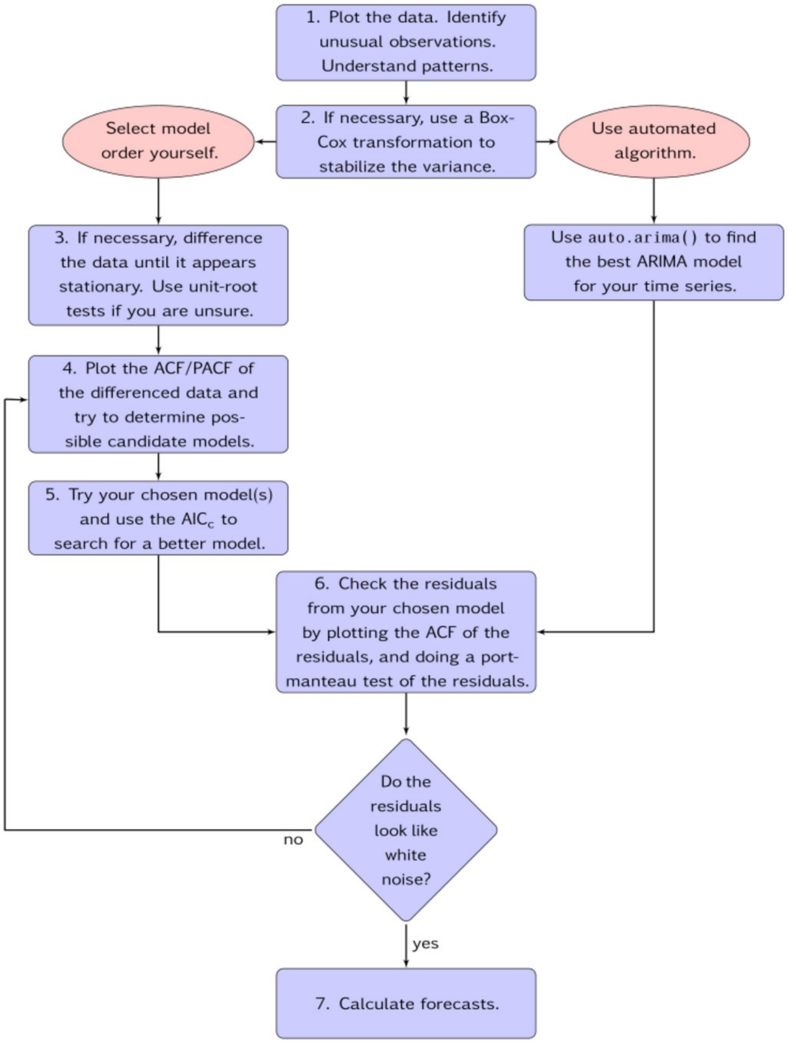
Accuracy: Predict months 30-48 (with same 62 items as normal model)

- Real value is in prediction interval 948 times (*80.5%*)
- Real value is in not prediction interval 230 times
- *80.5%* (Advanced Model) > *78.1%* (Normal model)

Most importantly, the prediction interval is smaller most of the time.

- Prediction Interval Size:
  - Advanced Model < Normal Model (711 times)
  - Normal Model < Advanced Model (529 times)

Below is a flowchart which illustrates the process of Advanced Time Series.

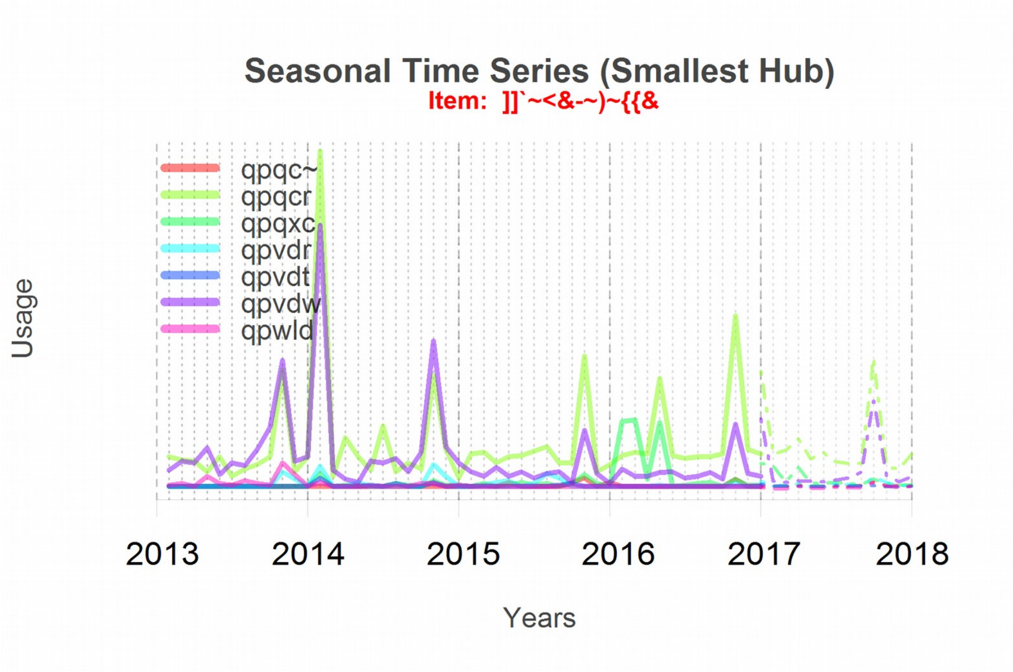

1. Plot the data. Identify unusual observations. Understand patterns.
2. If necessary, use a Box-Cox transformation to stabilize the variance.
Select model order yourself.
Use automated algorithm.
3. If necessary, difference the data until it appears stationary. Use unit-root tests if you are unsure.
Use auto.arima() to find the best ARIMA model for your time series.
4. Plot the ACF/PACF of the differenced data and try to determine possible candidate models.
5. Try your chosen model(s) and use the AIC$_c$ to search for a better model.
6. Check the residuals from your chosen model by plotting the ACF of the residuals, and doing a portmanteau test of the residuals.
Do the residuals look like white noise?
no
yes
7. Calculate forecasts.

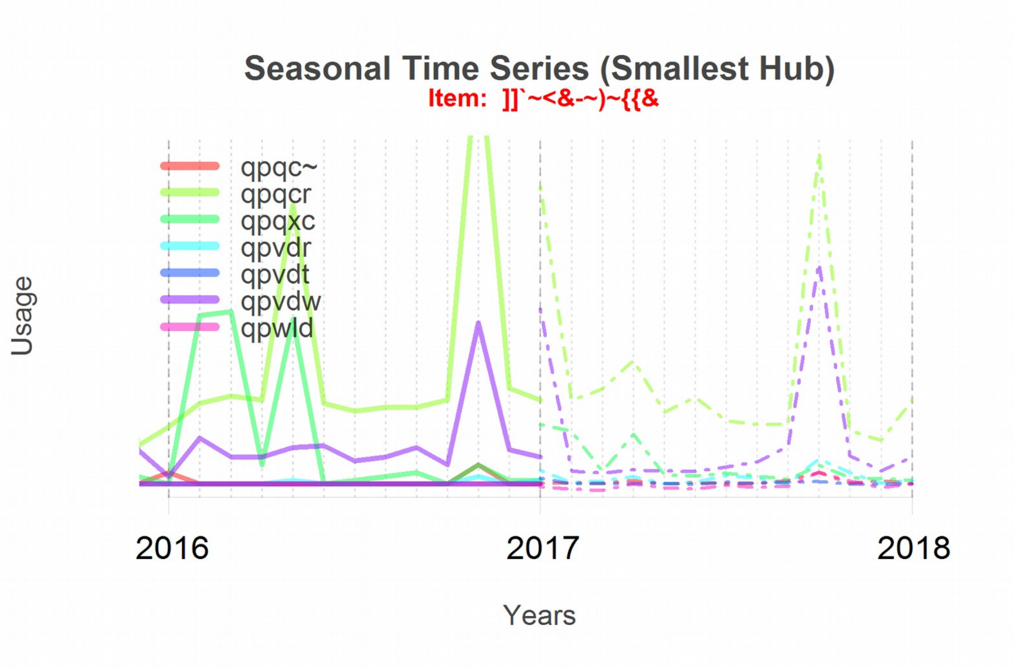
Redistribution Concept

We will use the previously mentioned 3 models, Linear Regression, Basic Time Series, and Advanced Time Series to determine if an item should be relocated. Once we know the future sales for a single item across different branches, we can see where the item sells best and worst.

There are two main situations where redistribution of products is necessary. First, we predict the product will be sitting on the shelf without selling. Second, we predict that a branch will run out of stock for a product. We want to transfer products from branches predicted not to sell said product, to branches where the product has high predicted sales.

Below is a graph of the time series for all of the Smallest Hub branches that carry a certain item overlaid onto one image. The prediction size is a year using the seasonal Basic Time Series model. The top figure is the entire time series while the bottom figure zooms in on the prediction. It is obvious that some branches perform better than others when selling products. We can transfer the products from the lower selling branches to the higher selling branche





As mentioned, we will be transferring products between branches of the same hub. This is because those branches will be stocked by the same distribution center so the redistribution of products can fall into order with the natural pattern of product shipments. Also, we do not know the distance between branches of different hubs. Transferring internally within a hub guarantees that the redistribution cost of the items will be minimal.

This will drastically improve Fastenal’s inventory turnover ratio. Products will not sit on the shelf and be a sunk cost. Additionally, Fastenal’s overhead will decrease as less products will have to be purchased or manufactured because existing inventory can be transferred to areas of demand. Most importantly, Fastenal’s customers will be served quicker as the products they want are already in the branches closest to them.

Conclusions

Based on the techniques we have discussed, we can use these models to sort Fastenal’s data and predict sales. First, we run our data through Linear Regression, to check to see if the model is a good fit. From there, we will either accept it as a reasonable fit or pass it along to Time Series. Once the Time Series has been ran, we again will check fit and either accept the fit or pass it to Advanced Time Series. With this information, we can begin designing an optimum inventory plan for Fastenal. Once we have this optimized inventory plan, we can begin to look at how Fastenal’s current inventory is being distributed.

This is the second half to the problem, that is, looking at how we can redistribute Fastenal’s inventory so that products will be more likely to sell. However, certain things need to be considered to see if it is economically feasible. For example, branches that are very far away from one another will mean the cost to ship that particular product will be substantially higher. The next step in this project is to build a model that analyzes the costs and benefits.

Future Work

While the project is concluded for now, there is the potential to improve our results and go further. Our data ends December 2016. We can use the gathered data from the past 5 months to test our predictions and further refine our forecasting models. Also, further investigation into redistribution is needed. We would focus on at least 5 products to create a comprehensive redistribution plan. This will involve looking at both future forecasted sales and past stocking history for every branch that carries the product. For this we are limited by the amount of data our computers can handle. Unless the data is pre-subsetted by Fastenal, the stocking tables for 4 years will be massive and unreasonable to analyze with our technology.

Appendix

[1]

Given Data Tables:

General Column Definitions

blank	A column of unique integers for every row
INV_DATE	Date inventory was taken on a given item. All dates are 12/1/2016
BRANCH_CODE	Unique identifier for a single Branch. Multiple branches in this file
INV_ITEM_ID	Unique identifier for every inventory item sold by Fastenal
QTY_BASE	Current quantity on hand of an item
STD_COST_USD	Adjusted cost of an item for the company to buy in USD
STD_VALUE_USD	What Fasten sells an item for at retail in USD
CATEGORY_ID	Unique identifier for a group of categories assigned to products
WHSL_PRICE	Extended sale price of an item
COGS_COST_USD	“Cost of Goods” cost in USD for Fastenal
COGS_VALUE_USD	“Cost of Goods” cost in USD at retail
COST_FLAG	Unknown

4 Years Consumption Table Structure

* *Every month is recorded under* USAGE_MONTH *except for October, in which* USAGE *for the entire year is calculated*

blank	A column of unique integers for every row

INVOICE_DT	Date inventory was taken on a given item. Dates recorded on are 10/1/2013, 10/1/2014, 10/1/2015, 10/1/2016
BRANCH_CODE	Unique identifier for a single Branch. Multiple branches in this file
INV_ITEM_ID	Unique identifier for every inventory item sold by Fastenal
USAGE1	Quantity sold of the item in 2013
USAGE2	Quantity sold of the item in 2014
USAGE3	Quantity sold of the item in 2015
USAGE4	Quantity sold of the item in 2016
USAGE_MONTH_1	Quantity sold of the item for the month of January
USAGE_MONTH_2	Quantity sold of the item for the month of February
USAGE_MONTH_3	Quantity sold of the item for the month of March
USAGE_MONTH_4	Quantity sold of the item for the month of April
USAGE_MONTH_5	Quantity sold of the item for the month of May
USAGE_MONTH_6	Quantity sold of the item for the month of June
USAGE_MONTH_7	Quantity sold of the item for the month of July
USAGE_MONTH_8	Quantity sold of the item for the month of August
USAGE_MONTH_9	Quantity sold of the item for the month of September
USAGE_MONTH_11	Quantity sold of the item for the month of November
USAGE_MONTH_12	Quantity sold of the item for the month of December

Smallest Hub Data:

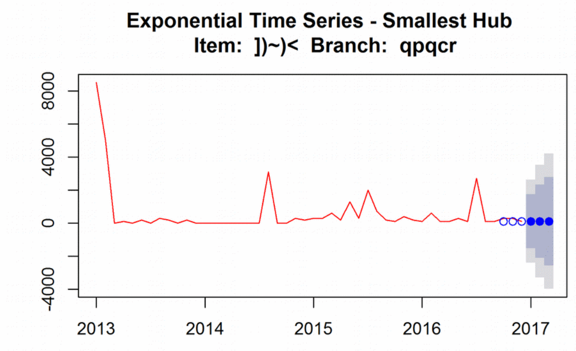
Branch Code	4 Year Total Consumption

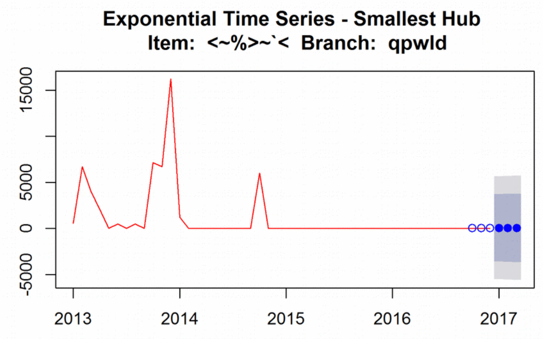
qp<<~	247,684
qpqc~	18,283,604
qpqcr	21,601,625
qpqxc	22,891,776
qpvdr	12,361,896
qpvdt	25,223,266
qpvdw	19,943,934
qpwld	12,878,353
qpzdt	20,780,009
Total:	154,212,147
Avg. Consumption Per Branch:	17,134,683

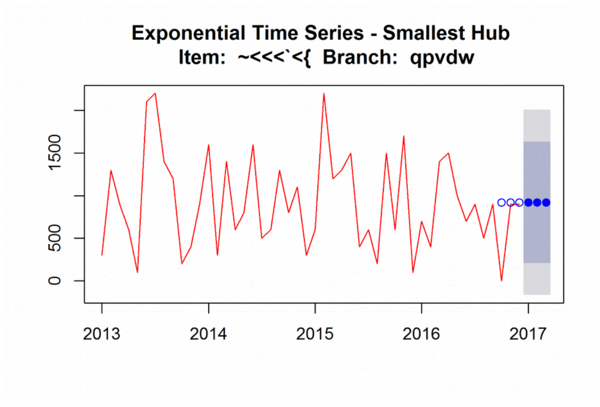
[2]

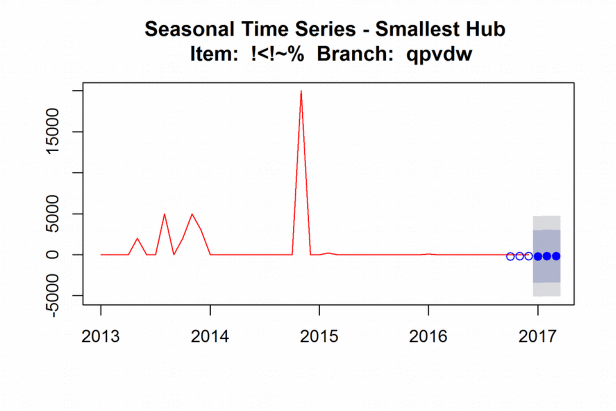
Time Series Graphs:

Stagnant Sale Forecasts:

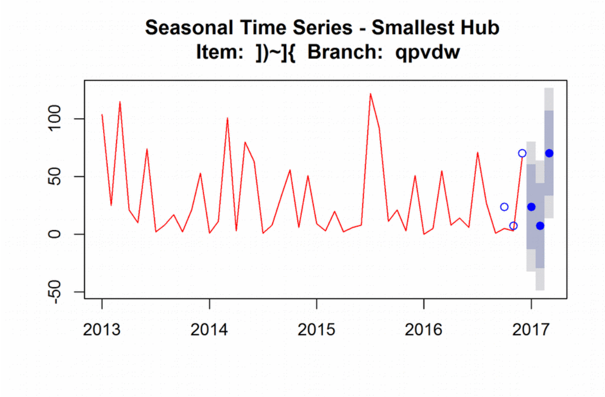


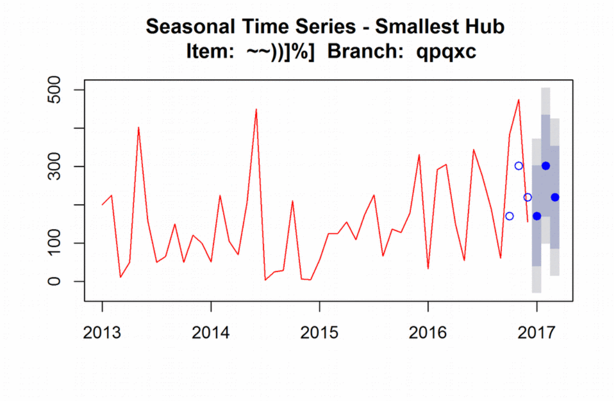




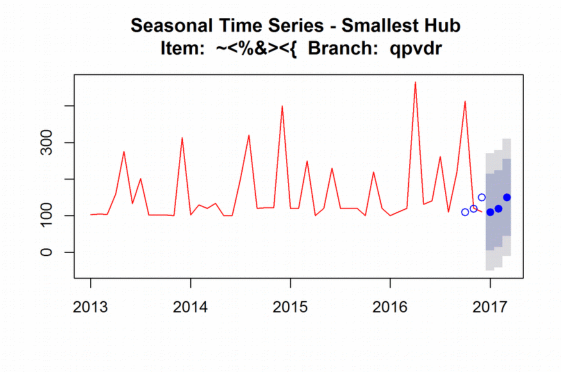


Seasonal Sales Forecast:

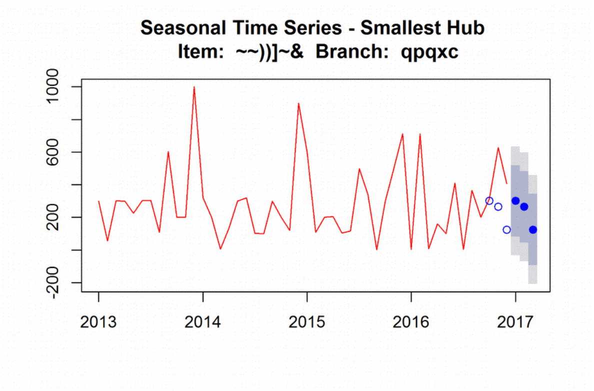




Increased Sales Forecast:

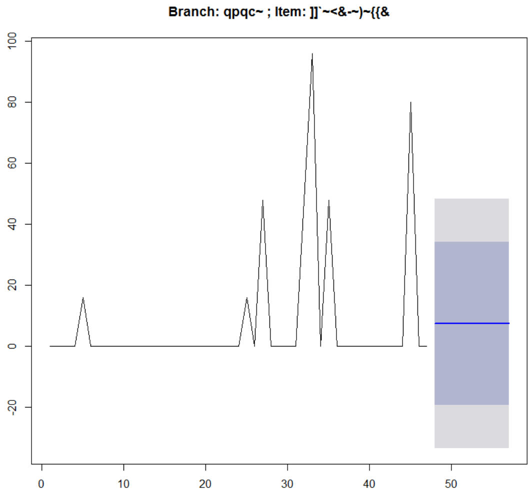


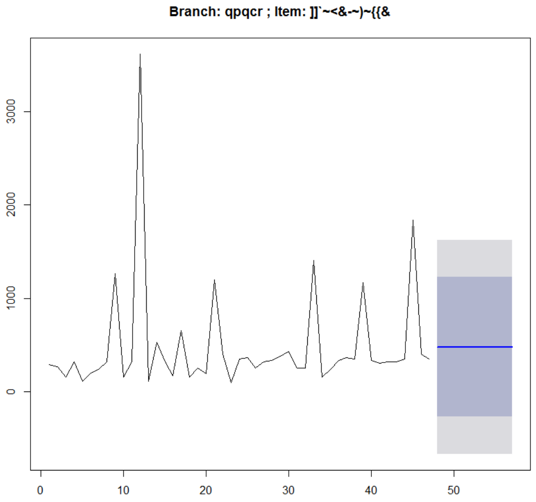
Decrease Sales Forecast:

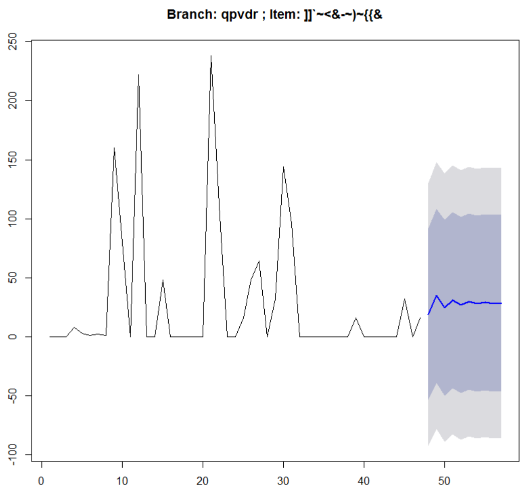


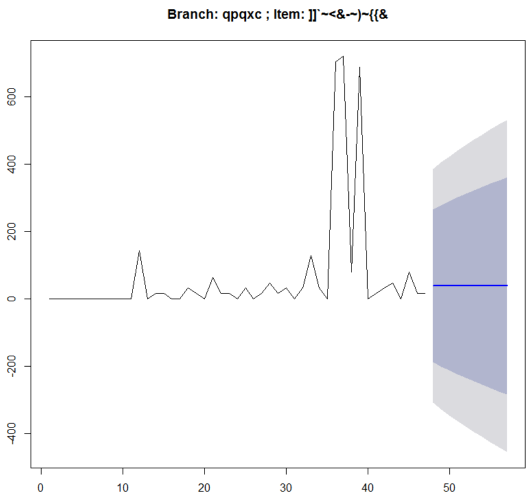
Advanced Time Series Graphs:

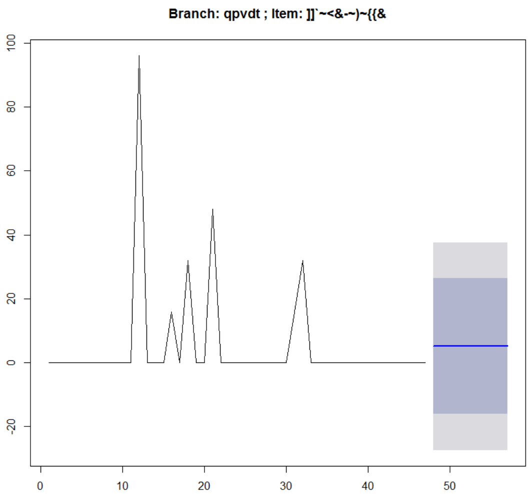
These graphs are for the product discussed in the “Redistribution” time series graphic.

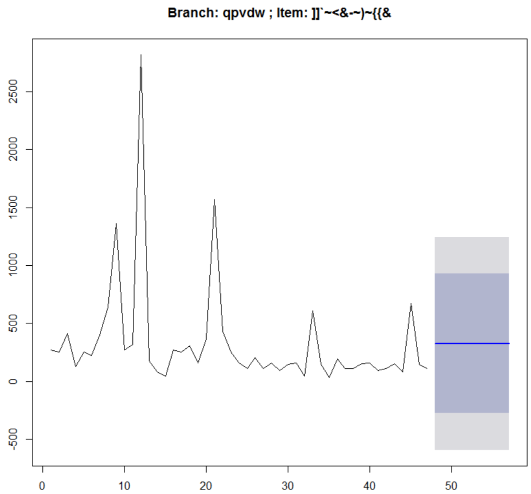


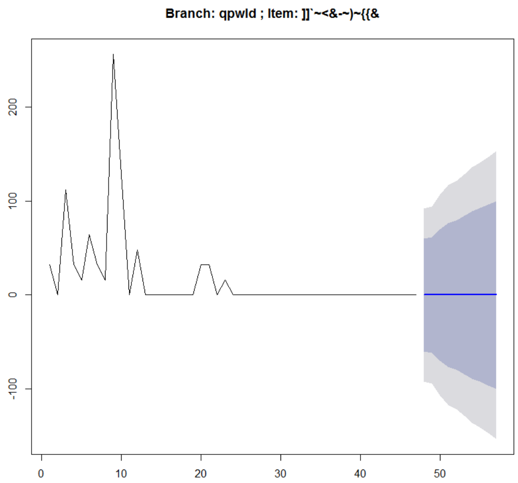



Branch: qpvdr ; Item:]]`~<&-~)~{{&


Branch: qpqxc ; Item:]]`~<&-~)~{{&


Branch: qpvdt ; Item:]]`~<&-~)~{{&


Branch: qpvdw ; Item:]]`~<&-~)~{{&


Branch: qpwld ; Item:]]`~<&-~)~{{&

[3]

Source code for Advanced Time Series

```
      library(fpp)
      x <- c(49)
      tep <- 0
      pridictRange <- 0
      errorRange <- 0
      pCount <- 0
      eCount <- 0
      total <- 0
      errout <- 48
      for(div in 1:9){
        div <- div * 0.1
      for (num in 1:69){
        e <- c()
        for(err in 25:48){
          low <- c(num)
          upp <- c(num)
          fit <- auto.arima(small[20:err, num])
          my.array <- forecast(fit)
          e <- append(e, as.numeric(my.array$mean[1]) -
                         as.numeric(small[err + 1, num]))
        }
        fiterr  <- auto.arima(e)
        err.array <- forecast(fiterr)
        min <- as.numeric(my.array$mean[1]) - as.numeric(err.array$upper[1])
        if(pridictRange > errorRange){
          max <- as.numeric(my.array$mean[1]) -
as.numeric(err.array$lower[1])
          point <- min + (max - min) * div
          e <-c()

            total<- total + as.numeric(small[48, num])
          }
          print("total")
          print(total)
          print(div)
      }
```

References

2.5 Evaluating forecast accuracy. (n.d.). Retrieved May 11, 2017, from https://www.otexts.org/fpp/2/5

8.1 Stationarity and differencing. (n.d.). Retrieved May 11, 2017, from https://www.otexts.org/fpp/8/1

Autoregressive integrated moving average. (2017, April 29). Retrieved May 11, 2017, from https://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average