

# Demand Forecasting: Predicting the Demand for Avocados

Demand forecasting is a crucial area for retail businesses. Inventory management, assortment selection, and pricing are all deeply affected by the quality of demand forecasts.

In this project, I have attempted to forecast the demand for avocados over the next quarter. In this case, avocados are not a new product and thus, we have access to historical data. I have used this dataset from Kaggle to perform my analyses:

<https://www.kaggle.com/neuromusic/avocado-prices>

After preliminary data exploration to explore the relationship between the demand and other crucial factors like pricing strategy, product type, and geographic region of sale, I attempted the following models to predict the demand:

1. **Simple linear regression**
2. **Time series regression** (which allows for fourier terms to capture seasonality)
3. **ARIMA forecasting without regressors** along with the **naive forecast**, which will serve as *benchmark metrics*
4. **ARIMA forecasting with regressors**
5. **Seemingly Unrelated Regression (SUR)** to capture the correlation of errors across region-specific linear models
6. An **ensemble** of SUR and ARIMA models, which is a simple average over the two

As we will learn later, the ensemble model performs the best on an aggregate level.

A few points to consider before moving forward with the analysis:

1. We don't observe the true demand for avocados, but only the sales that were made
2. Because we have access to weekly data, we can make meaningful predictions at the weekly level. But forecasts at the daily level would be even better (we would model in the day-by-day seasonality in this case).

## Loading packages and data

```
#loading packages
library(readxl)
library(plyr)
library(dplyr)
library(ggplot2)
library(lubridate)
library(forecast)
library(tidyr)
library(excelR)
library(readxl)
library(writexl)
```

```

library(caret)
library(systemfit)
library(stringr)
library(viridis)
library(MASS)
library(rcompanion)

#loading data
avocado = read_excel('avocado.xlsx', sheet='Sheet1')

#adding the trend and month seasonality variables
avocado$trend = as.numeric((avocado$date) - 1420329600)/604800
avocado$month = format(as.Date(avocado$date), "%m")
avocado$month = as.factor(avocado$month)

#filtering out aggregate-region rows
avo_clean = avocado%>%filter(!region %in% c('Southeast','Midsouth',
                                             'West', 'SouthCentral','NorthEast','WestTexNewMexico',
                                             "TotalUS"))

```

## 1. Exploratory Data Analysis

A quick snapshot of the data:

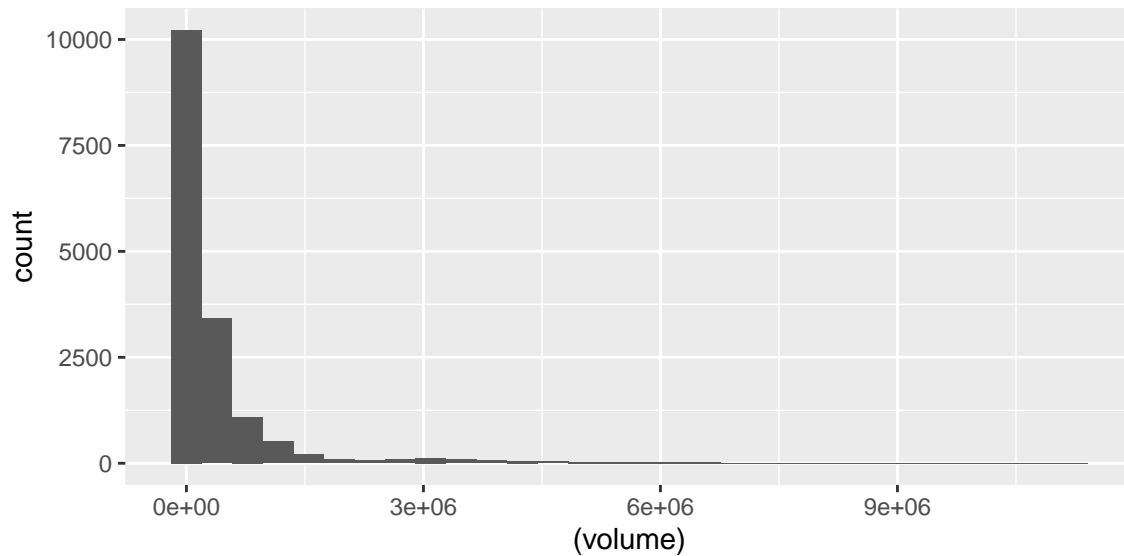
```
head(avocado)
```

```
## # A tibble: 6 x 8
```

	date	price	volume	type	year	region	trend	month
	<dtm>	<dbl>	<dbl>	<chr>	<dbl>	<chr>	<dbl>	<fct>
## 1	2015-12-27 00:00:00	1.33	64237.	conventional	2015	Albany	51	12
## 2	2015-12-20 00:00:00	1.35	54877.	conventional	2015	Albany	50	12
## 3	2015-12-13 00:00:00	0.93	118220.	conventional	2015	Albany	49	12
## 4	2015-12-06 00:00:00	1.08	78992.	conventional	2015	Albany	48	12
## 5	2015-11-29 00:00:00	1.28	51040.	conventional	2015	Albany	47	11
## 6	2015-11-22 00:00:00	1.26	55980.	conventional	2015	Albany	46	11

Let's plot the demand histogram to explore what transformations can be applied.

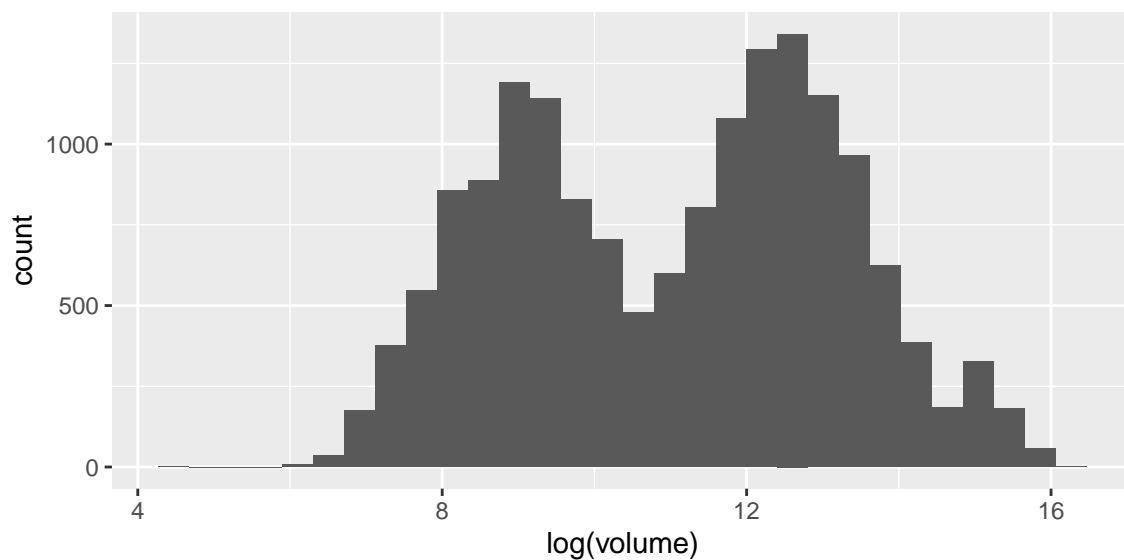
```
ggplot(avo_clean, aes((volume))) + geom_histogram()
```



Considering the extreme right-skew, we need to consider applying transformations. Through trial-and-error, log transformations turn out to be the best.

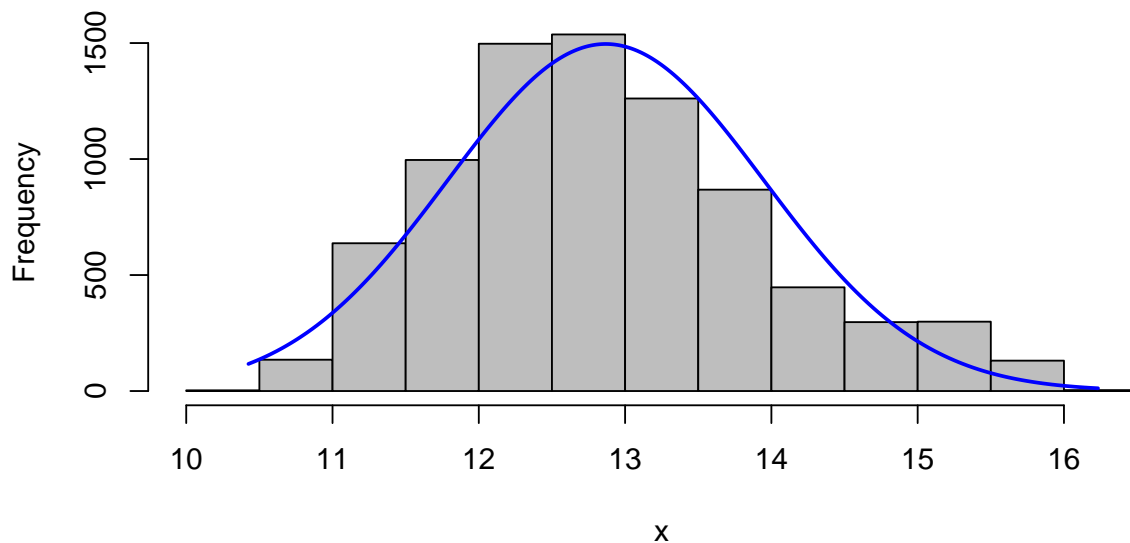
The two peaks are the two demand curves for **organic** and **conventional** avocado products.

```
ggplot(avo_clean, aes(log(volume))) + geom_histogram()
```



Looking at the individual demand histogram for 'conventional', we can see that it fits the normal distribution curve pretty well.

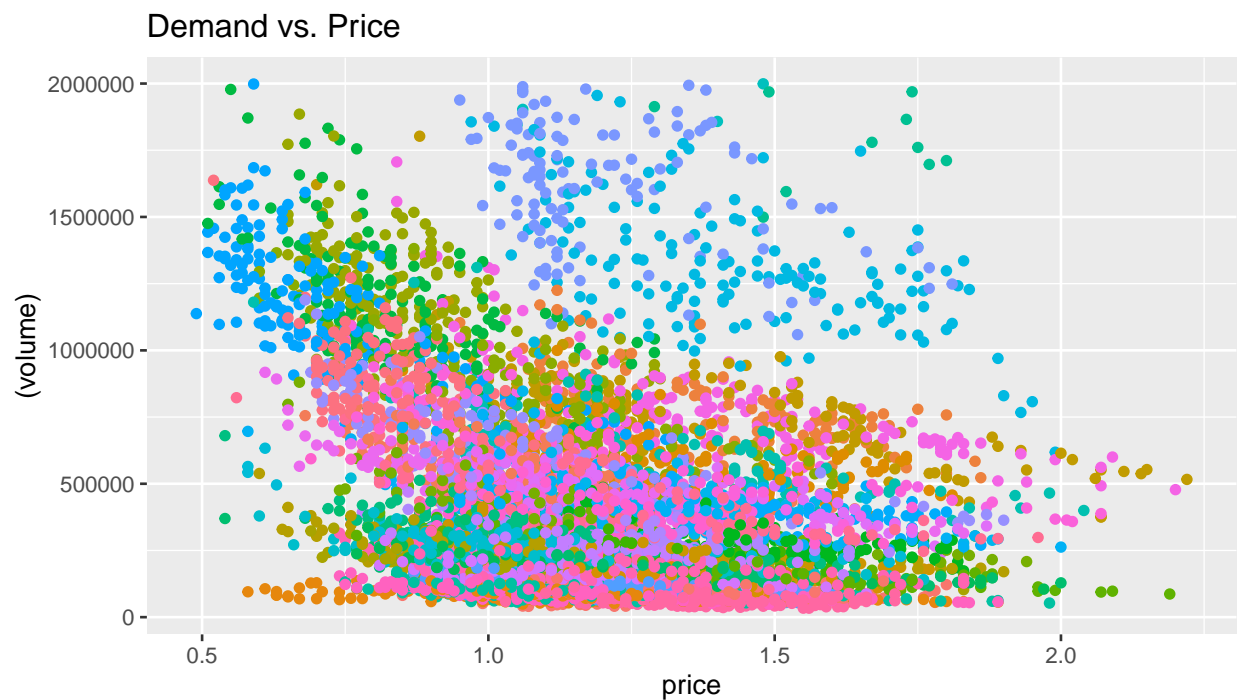
```
conventional <- avo_clean %>% filter(type == 'conventional')
organic <- avo_clean %>% filter(type == 'organic')
plotNormalHistogram(log(conventional$volume))
```



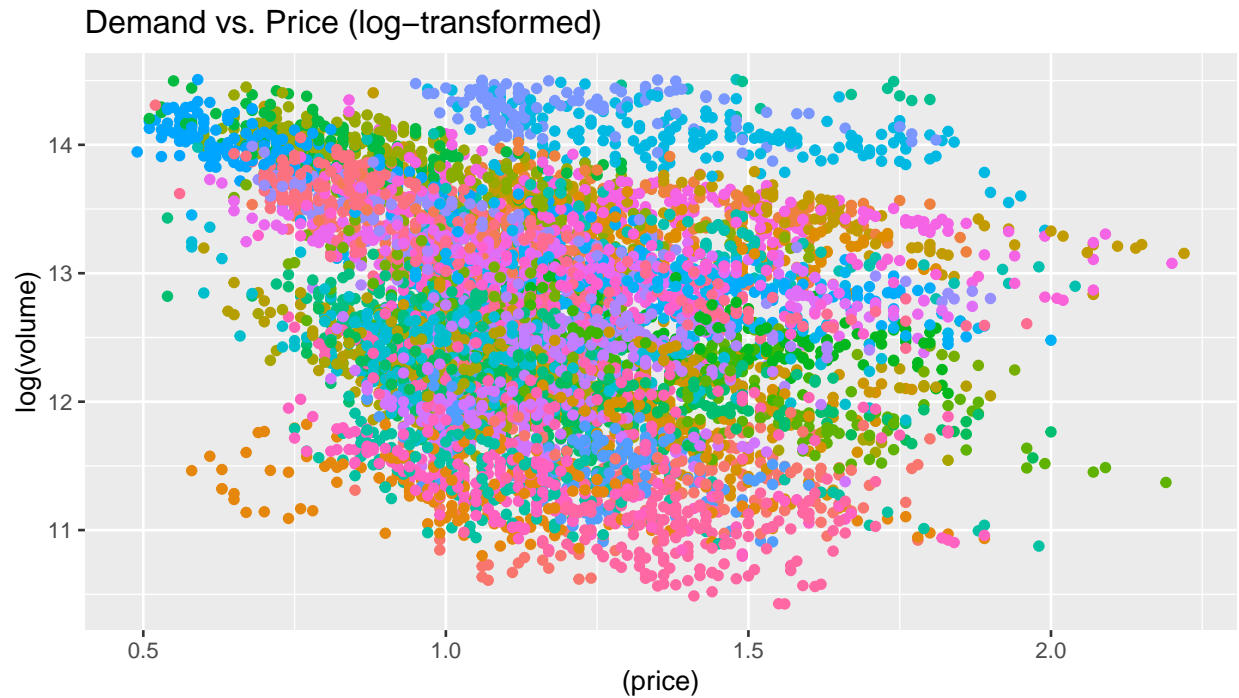
To further improve our transformations, we could explore box-cox transformations, but for this project, I stuck to using the log-transform.

**The demand-price relationship often follows a log-curve.**

Let's check for that. In the following graph, color indicates region.



As we can see, applying a log transformation to the Demand variable improves the linearity between demand and price.

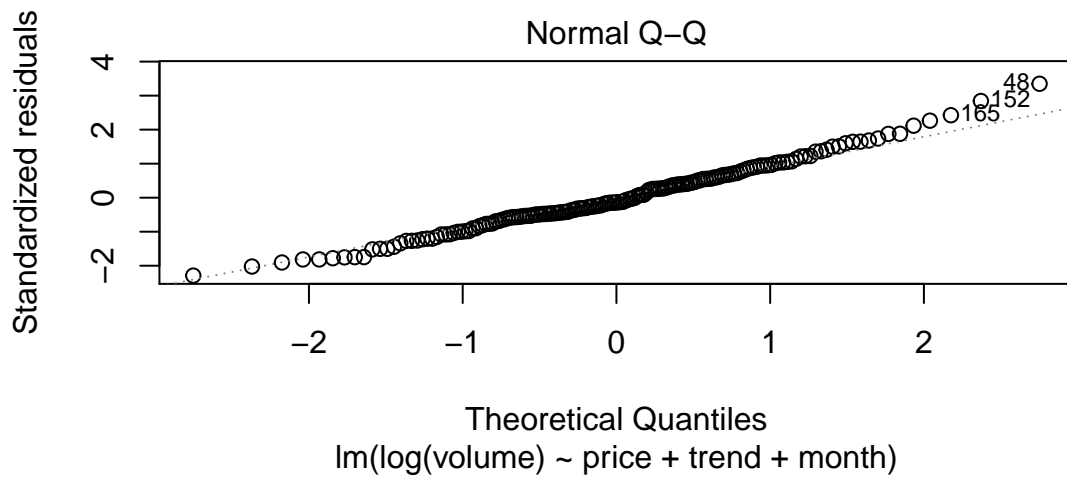
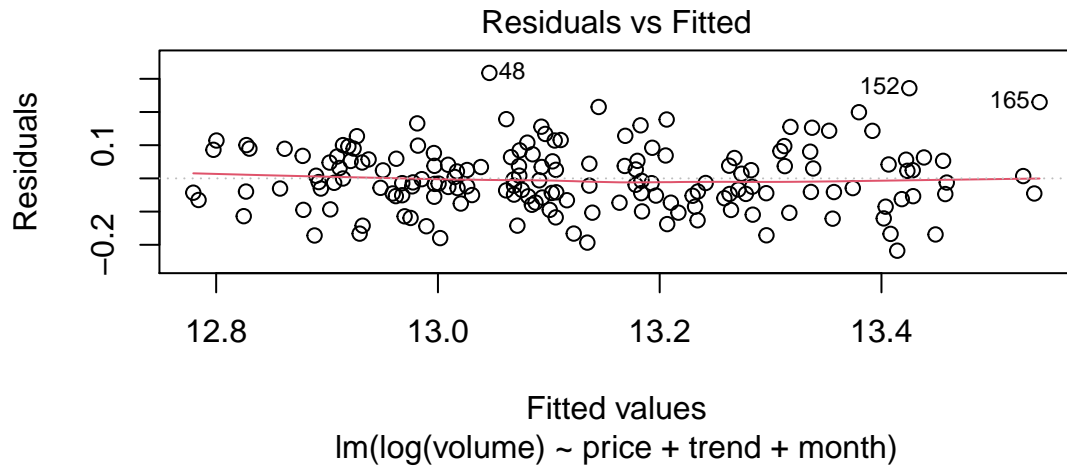


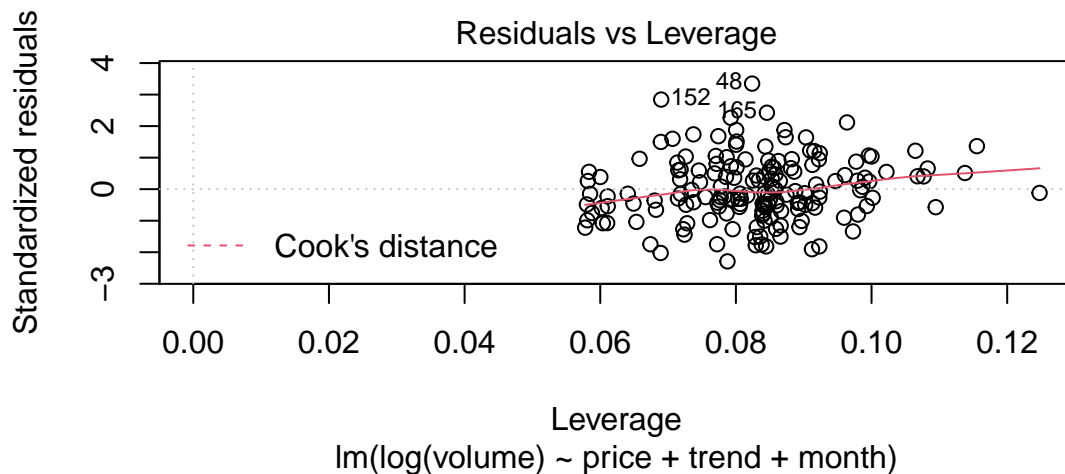
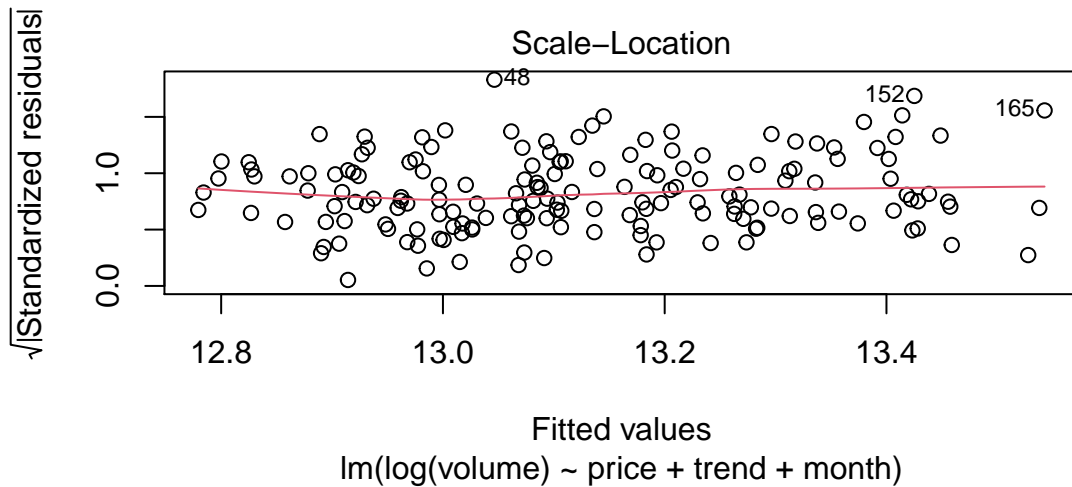
## Exploring linear regression fits

The main assumptions of linear regression are that: 1. residual plots show no pattern when plotted against fitted values, and that 2. error terms are homoscedastic (equal variance)

After applying log transformations, let us explore the error plots for **Atlanta, conventional type**.

```
example_1 <- avo_clean%>%filter(region=='Atlanta', type=='conventional')  
  
#the linear model  
m1 = lm(log(volume) ~ price + trend + month, example_1)  
  
#error plots  
plot(m1)
```





Our errors appear randomly distributed, so the model seems to work well in this case.

However, we will be building models for different regions, and not all will fit well. With box-cox transformations and fourier terms, we aim to capture more information over seasonality and produce better linear fits.

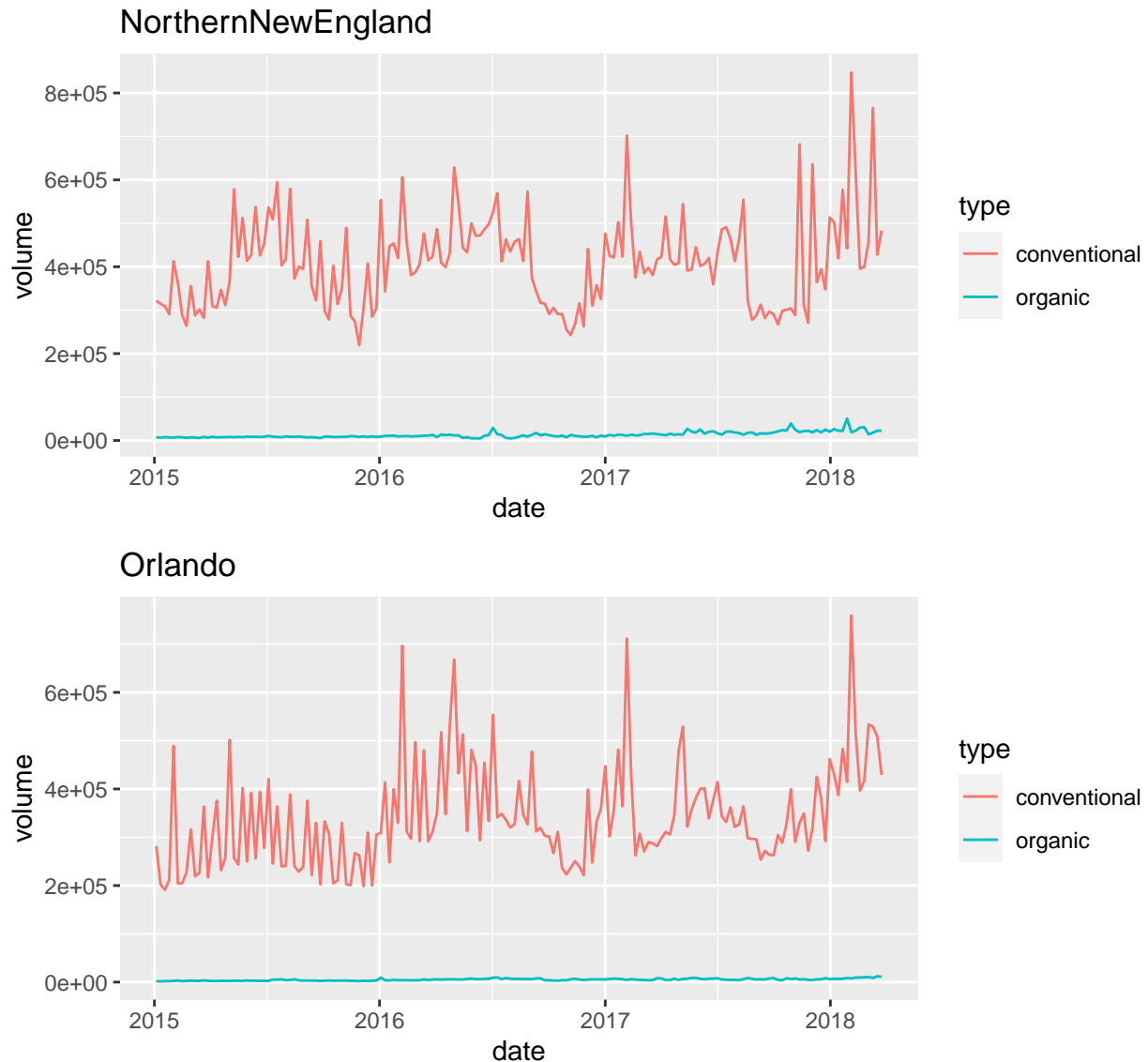
Let us explore how the demand varies over time for specific regions.

## Example of a time series: Northern New England, Orlando

```
#plotting each individual time series
for(reg in unique(avocado$region)[31:32]){

  ts = avocado %>% filter(region == reg)
  print(ggplot(ts, aes(date, volume, color = type)) + geom_line() + labs(title=eval(reg)))

}
```



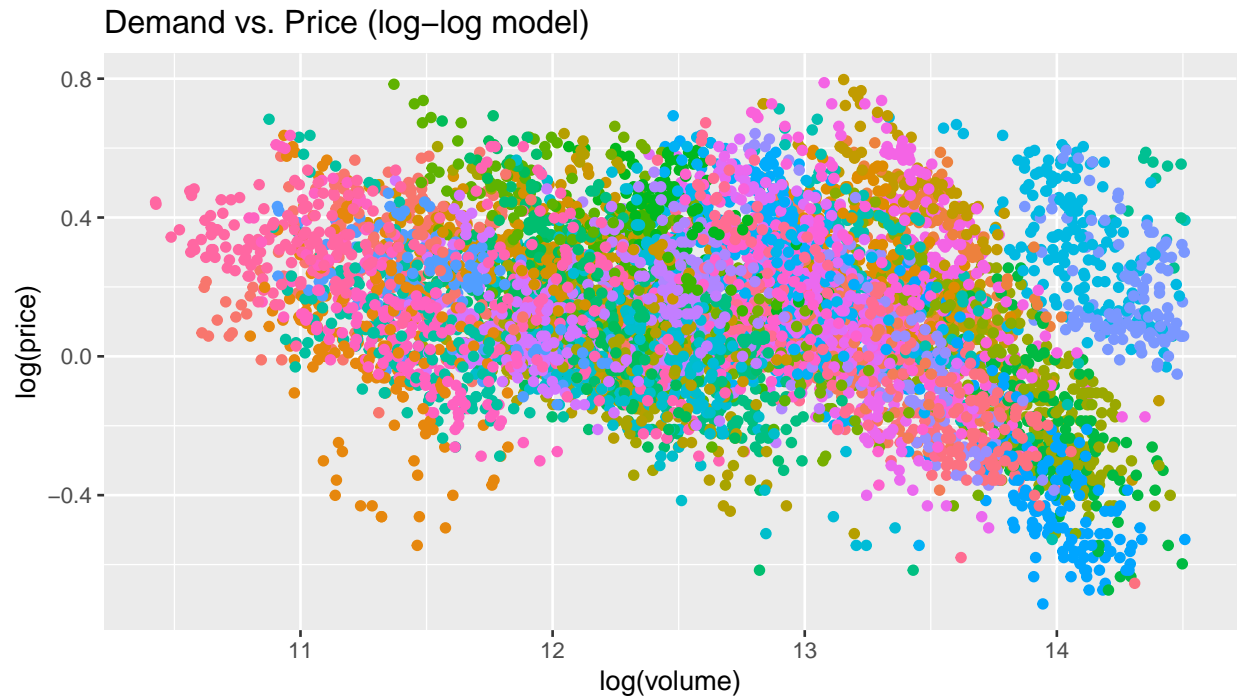
All regions are showing distinct seasonal patterns, indicating the need to include seasonal terms as regressors. Let's move into building models to predict demand.

## Price Elasticity of Demand

Price elasticity of demand can best be understood by the log-log model (price and demand are both transformed into the log variable). Let's take a look.

```
#visualising the log-log model
ggplot(x_conventional, aes(log(volume), log(price), color = region)) +
  geom_point() + theme(legend.position = 'none') +
  labs(title='Demand vs. Price (log-log model)')
```





Let's see which region is showing the least and most price sensitivity in terms of demand:

```
price_elasticity<-data.frame()

for(t in unique(avo_clean$type)){
  for(r in unique(avo_clean$region)){

    series = avo_clean %>% filter(region == r, type == t)

    lmm<-(lm(log(volume) ~ log(price) + trend + month, series))
    row<-data.frame(region= r, type = t,
                    elasticity = coef(summary(lmm))["log(price)", "Estimate"])
    price_elasticity<-rbind(price_elasticity, row)
    #print(coef(summary(lmm))["log(price)", "Estimate"])
  }
}

sorted = price_elasticity%>%group_by(type)%>%
  dplyr::arrange(elasticity, .by_group=TRUE)

sorted %>%group_by(type) %>%
  slice(c(1, n())) %>%
  ungroup()
```

```
## # A tibble: 4 x 3
##   region          type      elasticity
##   <chr>          <chr>      <dbl>
## 1 NorthernNewEngland conventional -1.52
## 2 Boise          conventional -0.471
## 3 GrandRapids    organic      -2.52
## 4 Pittsburgh     organic       1.67
```

Interestingly, the demand for organic avocados is actually increasing with an increase in price!

## Building Models

Based on the histogram plots and residual errors, we apply a log transformation to the 'volume' (representing demand).

We then build the following models to predict the demand:

- 1) naive forecast: Baseline metric to compare whether our models are performing better than no models
- 2) ARIMA without regressors: we use this model to check whether our predictors ie. price and trend are improving our forecasts.
- 3) ARIMA with price
- 4) ARIMA with fourier terms (to model weekly seasonality)
- 5) Time series regression (tslm) with fourier terms for seasonality, trend, and price
- 6) Seemingly Unrelated regression (SUR): simultaneously solving different equations for different region-and-type combinations. Assumption here is that while they will have separate parameters, they all suffer from the same noise/error elements which are modeled into the equation
- 7) Ensemble over best-performing models to check if it improves accuracy. Here we average our forecasts over ARIMA and SUR

The rmd file contains the whole code.

### SUR model

```
SUR_model <- systemfit(formulas, method = "SUR", data=train)

SUR_pre_transform = predict(SUR_model, test_predictors)
SUR_predicted = data.frame(exp(SUR_pre_transform)) #back to original scale

SUR_clean = data.frame(exp(SUR_pre_transform), week = seq.Date(from = as.Date("2017-08-06"),
                                                                by = 7,
                                                                length.out = 34))

SUR_long = gather(SUR_clean, key='region_type', value='SUR', -week)
SUR_long = separate(data = SUR_long, col = region_type, into = c("region", "type"))
compare_final = merge(SUR_long, compare_all,
                      by.x = c('region', 'type', 'week'), by.y = c('region', 'type', 'week'))
```

### Ensemble model: mean over ARIMA and SUR forecasts

```
#ensemble
compare_final$ensemble <- rowMeans(subset(compare_final,
                                           select = c(SUR, arima)), na.rm = TRUE)

#rmse ensemble
ensemble_rmse = compare_final %>% group_by(type, region) %>%
  summarise(ensemble = mean((ensemble - actual)^2)^0.5) %>%
```

```

  unite(region_type, region, type, sep='_')
n<-ensemble_rmse$region_type
ensemble_rmse = data.frame(t(ensemble_rmse[,-1]))
colnames(ensemble_rmse)=n
ensemble_rmse = tibble::rownames_to_column(ensemble_rmse, 'errors')
rmse_errors = rbind(rmse_errors, ensemble_rmse)

#mape ensemble
ensemble_mape = compare_final %>% group_by(type,region)%>%
  summarise(ensemble = mean(abs(ensemble - actual)/actual)*100) %>%
  unite(region_type, region, type, sep='_')
n<-ensemble_mape$region_type
ensemble_mape = data.frame(t(ensemble_mape[,-1]))
colnames(ensemble_mape)=n
ensemble_mape = tibble::rownames_to_column(ensemble_mape, 'errors')
mape_errors = rbind(mape_errors, ensemble_mape)
#, mape = )

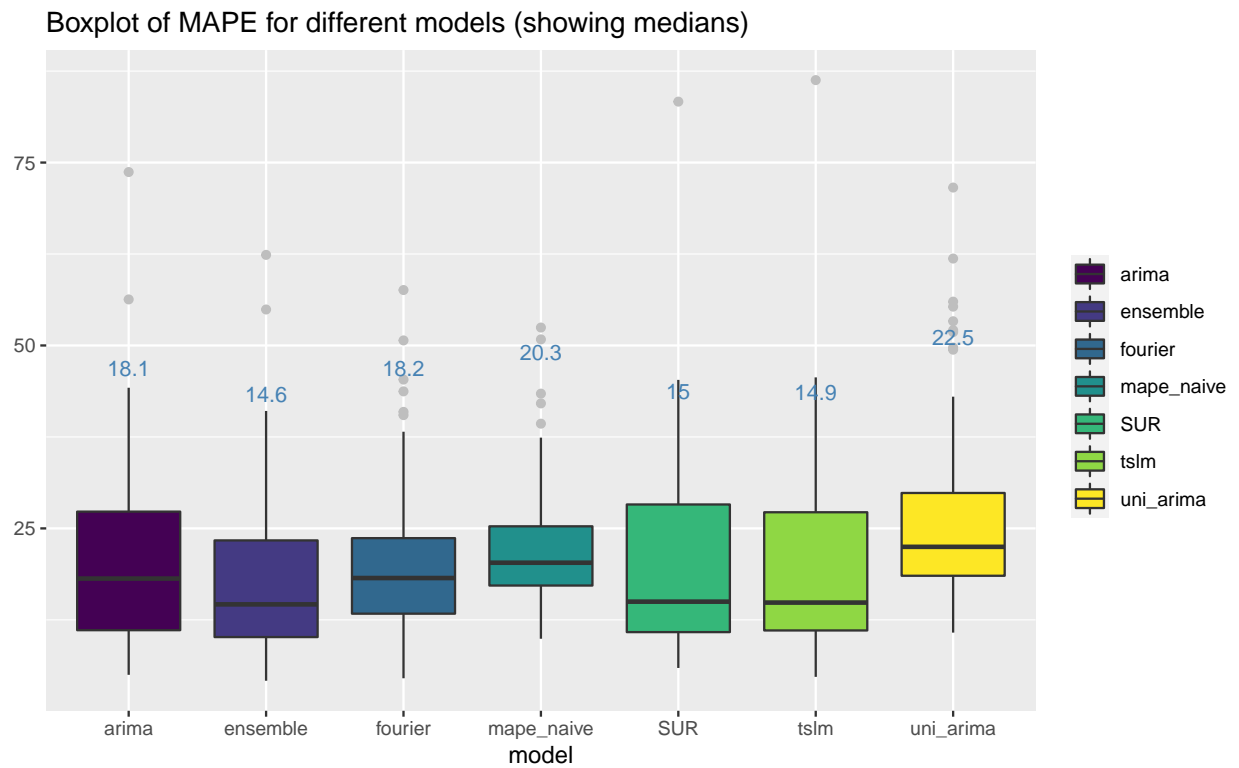
compare_long = gather(compare_final, key='model',
                      value = 'predicted', - c(region,type,week))

```

## Testing our models

Let us look at how our models are performing.

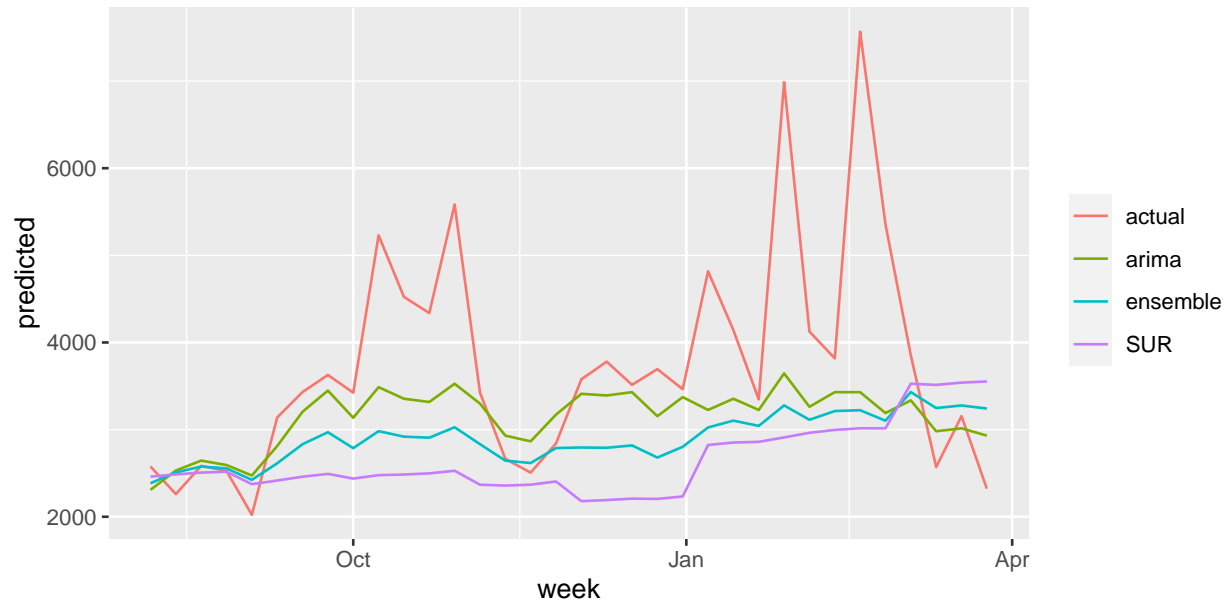
## Boxplot of MAPE over different regions and types



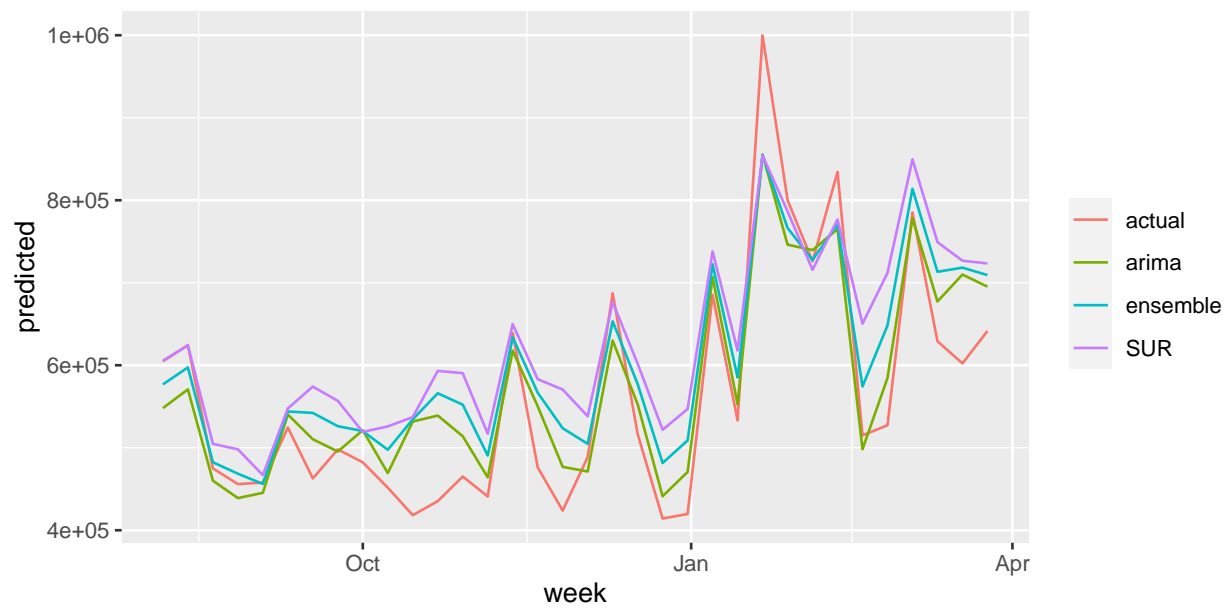
Through our boxplots, we can see that an ensemble model is performing the best overall. Thus, we stick with the *ensemble*.

## Example Forecasts

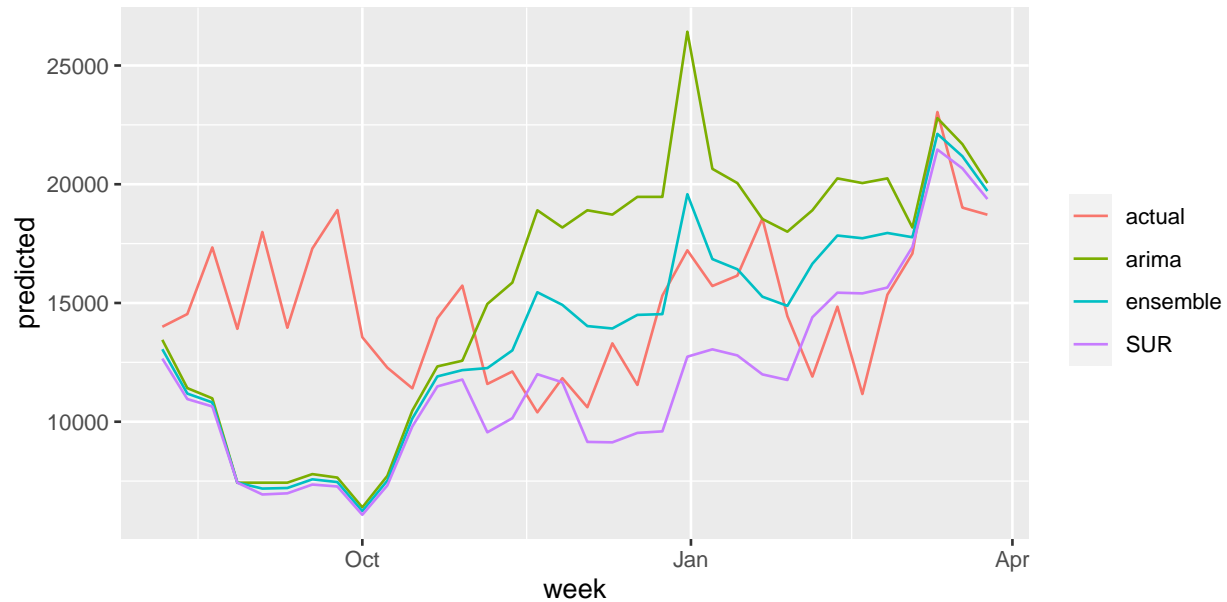
1. Albany, organic



2. Boston, conventional



3. Atlanta, organic (doing not so well)



## Next steps

To improve our demand forecasts, we could try the following steps:

- 1) Facebook's Prophet package, which is based on Bayesian interpretations.
- 2) Customised transformations: We assumed that a log-linear relationship for all areas and all types, but we need to account for the fact that different models will share different relationships.

Box-cox transformation might be worth exploring.

- 3) Time-series cross-validation:

Currently we are picking the ensemble model for all combinations of regions and types, but it is very much possible that different models are appropriate for different time series.

If we choose different models based on lowest RMSE/MAPE on test data, then we are overfitting on the test data. However, minimising cross-validated RMSE/MAPE will solve this problem.

The `tsCV()` function in R will allow us to calculate errors for different time windows, thus making our estimate of error more robust.

- 4) Data on anomalous events in the avocado industry would help in explaining sudden spikes/drops.
- 5) Because this data is in the realm of 'panel' data (as they call it in the field of econometrics), exploring fixed & random effects model would be beneficial. We already explored SUR and found favourable results compared to simpler models.