Agricultural Goods Price Forecast

Introduction

Agriculture is referred to as the production, processing and distribution of agricultural goods. It can be also the main source income for most developing countries. In addition to providing food and raw material, agriculture also provides employment opportunities to a very large percentage of the population.

Objetive

We are going to make use of our analytical knowledge to analyze and forecast agricultural goods prices of Rwanda, a developing country.

Data

For this project we'll use the United Nations Humanitarian Data Exchange Global Food Price Database that can be found here.

The database covers agricultural goods like beans, rice, maize, fish, and sugar for 76 countries and some 1,500 markets. The data is updated weekly and goes back as far as 1992 for a few countries, although many countries started reporting from 2003 or thereafter.

```r
library(dplyr)
library(readr)
library(lubridate)
library(knitr)
library(ggplot2)
library(magrittr)
library(forecast)

potato <- read_csv("Potato.csv", col_types = cols_only(adm1_name = col_character(),
                        mkt_name = col_character(), cm_name = col_character(),
                        mp_month = col_integer(), mp_year = col_integer(),
                        mp_price = col_number()))
potato_rename <- rename(potato, "region" = "adm1_name", "market" = "mkt_name",
                "commodity_kg" = "cm_name", "month" = "mp_month", "year" = "mp_year",
                "price_rwf" = "mp_price")
glimpse(potato_rename)
```

```
Rows: 4,320
Columns: 6
$ region       <chr> "$West/Iburengerazuba", "$West/Iburengerazuba", "$West...
$ market       <chr> "Birambo", "Birambo", "Birambo", "Birambo", "Birambo",...
$ commodity_kg <chr> "Potatoes (Irish)", "Potatoes (Irish)", "Potatoes (Iri...
$ month        <int> 11, 12, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 1, 2, 3...
$ year         <int> 2010, 2010, 2011, 2011, 2011, 2011, 2011, 2011, 2011, ...
$ price_rwf    <dbl> 157.0000, 133.3333, 96.5000, 97.0000, 107.8000, 125.50...
```

Data cleaning

As usually raw data doesn't arrives as we would want, so we have to make some arrengments and manipulate the data so we can work with it.

```
potato_clean <- potato_rename %>%
    mutate(date = ymd(paste(year, month, '01'))) %>%
    select(-year, -month)
head(potato_clean)
```

```
# A tibble: 6 x 5
  region               market  commodity_kg     price_rwf date
  <chr>                <chr>   <chr>                <dbl> <date>
1 $West/Iburengerazuba Birambo Potatoes (Irish)     157   2010-11-01
2 $West/Iburengerazuba Birambo Potatoes (Irish)     133.  2010-12-01
3 $West/Iburengerazuba Birambo Potatoes (Irish)      96.5 2011-01-01
4 $West/Iburengerazuba Birambo Potatoes (Irish)      97   2011-02-01
5 $West/Iburengerazuba Birambo Potatoes (Irish)     108.  2011-03-01
6 $West/Iburengerazuba Birambo Potatoes (Irish)     126.  2011-04-01
```

Now we have our cleaned data, an can follow the same process for other food types, we have many options:

- Beans
- Chili
- Cassava
- Oranges
- Maize
- Peas
- Sorghum
- Tomatoes

So, let's repeat the same process over and over! Just kidding, even though you can do that it is very unconvenient so we are going to build some functions to make our lives easier. However, we are going to continue with our potato as example of how you could do this analysis if it was just about one topic (potatoes), but you will find it is very much convenient to write functions in the end.

First of all, let's get a function that can read our data and define each column type.

```
read_data <- function(commodity){
    data_file <- paste0(commodity, '.csv')
    prices <- read_csv(
     data_file,
     col_types = cols_only(
      adm1_name = col_character(),
      mkt_name = col_character(),
      cm_name = col_character(),
      mp_month = col_integer(),
      mp_year = col_integer(),
      mp_price = col_double()))

prices_renamed <- prices %>%
  rename(
    region = adm1_name,
    market = mkt_name,
    commodity_kg = cm_name,
    month = mp_month,
    year = mp_year,
    price_rwf = mp_price
```

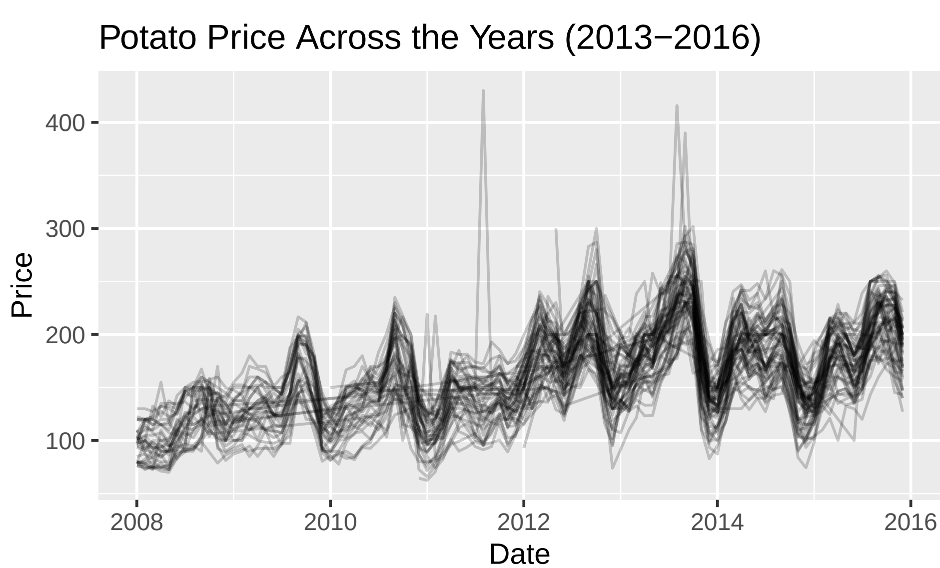
```
  )

prices_renamed  %>%
  mutate(
    date = ymd(paste(year, month, "01"))
  ) %>%
  select(-month, -year)
}

pea <- read_data("Pea")
glimpse(pea)
```

```
Rows: 1,893
Columns: 5
$ region       <chr> "$West/Iburengerazuba", "$West/Iburengerazuba", "$West...
$ market       <chr> "Birambo", "Birambo", "Birambo", "Birambo", "Birambo",...
$ commodity_kg <chr> "Peas (fresh)", "Peas (fresh)", "Peas (fresh)", "Peas ...
$ price_rwf    <dbl> 403.5000, 380.0000, 277.5000, 450.0000, 450.0000, 375....
$ date         <date> 2011-01-01, 2011-02-01, 2011-04-01, 2011-05-01, 2011-...
```

Exploratory Data Analysis

One first good step when you have historical data and want to start the analysis, its to plot the data you have gather. So, let's see how the prices of potatoes has change over time.

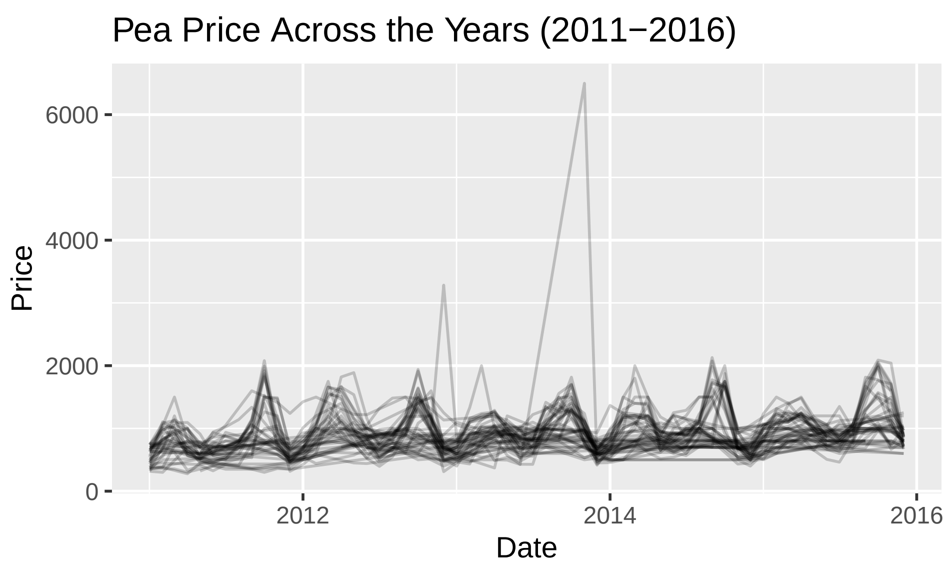
```
ggplot(data = potato_clean, aes(x = date, y = price_rwf, group = market)) +
    geom_line(alpha = 0.2) +
    ggtitle("Potato Price Across the Years (2013-2016)") +
    labs(x='Date', y='Price')
```



As you can see, there is a increasing trend on the prices of potatoes. More particularly, it seems that the prices relays on the season: for each year the price seems to drop at December-January and rise through the year until a peak around August.

We want to make similiar plots to the one above, so let's wrap the plotting code into a function.

```
price_plot <- function(prices, commodity){
title <- paste(commodity, "Price Across the Years (2011-2016)")
prices %>%
  ggplot(aes(date, price_rwf, group = market)) +
  geom_line(alpha = 0.2) +
  ggtitle(title) +
  labs(x='Date', y='Price')
}
price_plot(pea, "Pea")
```



Forecasting

Yes, it can be really helpful to get insights about the present status of particular circumstances, but what can be way more exciting its to get insight of how the situation may change n the future, in other words, make predictions.

In this situation we are working with food prices, so, we will take the median price across markets and analyze the resulting time series.

The reason we are taking the madian and not the mean it is because, by looking at the plots and the big spike in the price, we can conclude That probably these spikes probably indicate a logistic problem. Whether the food wasn't easily available at the market, or the harvest season wasn't good, the consequence of these outliers is that it is a bad idea to use the mean price of each time point. Instead, the median makes more sense since it is robust against outliers.

```
potato_summary <- potato_clean %>%
    group_by(date) %>%
    summarize(median_price_rwf = median(price_rwf))
head(potato_summary)
```

```
# A tibble: 6 x 2
  date       median_price_rwf
```

```
  <date>                 <dbl>
1 2008-01-01              97.5
2 2008-02-01             100
3 2008-03-01              95
4 2008-04-01              96.2
5 2008-05-01              95
6 2008-06-01             110
```

```
potato_ts <- potato_summary %$%
    ts(median_price_rwf, start = c(year(min(date)), month(min(date))),
       end = c(year(max(date)), month(max(date))), frequency = 12)
potato_ts
```

```
          Jan      Feb      Mar      Apr      May      Jun      Jul      Aug
2008  97.5000 100.0000  95.0000  96.2500  95.0000 110.0000 116.6667 125.0000
2009 120.0000 122.5000 130.0000 131.2500 135.0000 124.3125 125.8333 144.2500
2010 109.6875 113.5000 131.2500 132.0833 140.4167 147.3750 142.5000 161.5000
2011 105.7000 108.1750 118.8750 145.0143 148.6667 148.0500 137.4048 137.2619
2012 150.7500 175.2500 186.0139 186.2500 182.5000 162.7500 179.1250 196.9643
2013 154.3333 157.0000 171.2500 187.5000 177.0000 202.2500 210.0000 233.1875
2014 138.3333 158.7500 186.2500 198.2500 191.0000 189.3333 182.5000 187.6191
2015 136.2500 157.6071 178.0000 190.2778 179.3750 168.3333 180.0000 202.1250
          Sep      Oct      Nov      Dec
2008 136.2500 130.0000 127.5000 114.3750
2009 181.2500 170.0000 150.2500 112.0000
2010 182.4000 162.5000 151.5000 122.5000
2011 141.6667 144.2000 133.1750 141.5000
2012 226.5000 203.5000 169.2500 144.0000
2013 241.3333 237.5000 176.7083 140.0000
2014 200.0000 183.1310 150.0000 133.9286
2015 223.5000 217.5000 216.1250 190.0000
```

```
time_series <- function(prices){
prices_summarized <- prices %>%
  group_by(date) %>%
  summarize(median_price_rwf = median(price_rwf))

time_series <- prices_summarized %$%
  ts(
    median_price_rwf,
    start = c(year(min(date)), month(min(date))),
    end   = c(year(max(date)), month(max(date))),
    frequency = 12
  )
}
pea_ts <- time_series(pea)
pea_ts
```

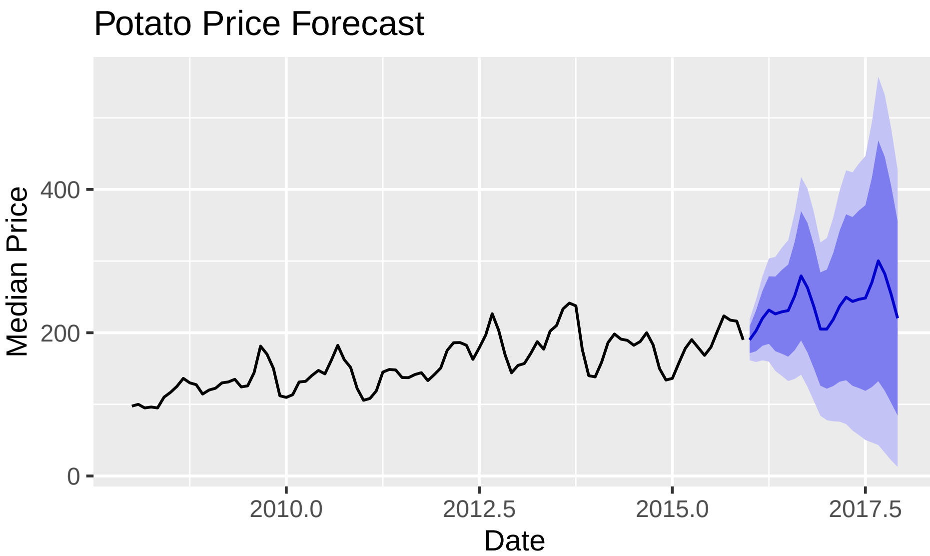
```
           Jan       Feb       Mar       Apr       May       Jun       Jul
2011  561.6667  700.0000  958.0000  710.0000  591.5000  597.8572  666.3572
2012  655.0000  950.0000 1272.1667 1166.0000  945.8750  822.3333  714.2857
2013  668.7500  781.6334  829.9875  975.0000  908.2500  789.9444  806.8000
2014  695.5000 1025.0000 1166.6250 1083.2500  825.0000  816.6667  809.5714
```

```
2015  800.0000 1066.6667 1100.0000 1051.8889  950.0000  873.6667  804.1250
           Aug       Sep       Oct       Nov       Dec
2011  758.5000  938.8333 1506.2500  787.5000  548.9375
2012  788.1250  990.7222 1413.7500  964.2619  661.8571
2013 1000.0000 1162.4583 1316.7500  916.6667  623.8571
2014 1000.0000 1000.0000 1666.6667  700.0000  633.3333
2015  900.0000 1166.6667 1550.0000 1066.6667  802.1250
```

```r
potato_forecast <- forecast(potato_ts)
potato_forecast
```

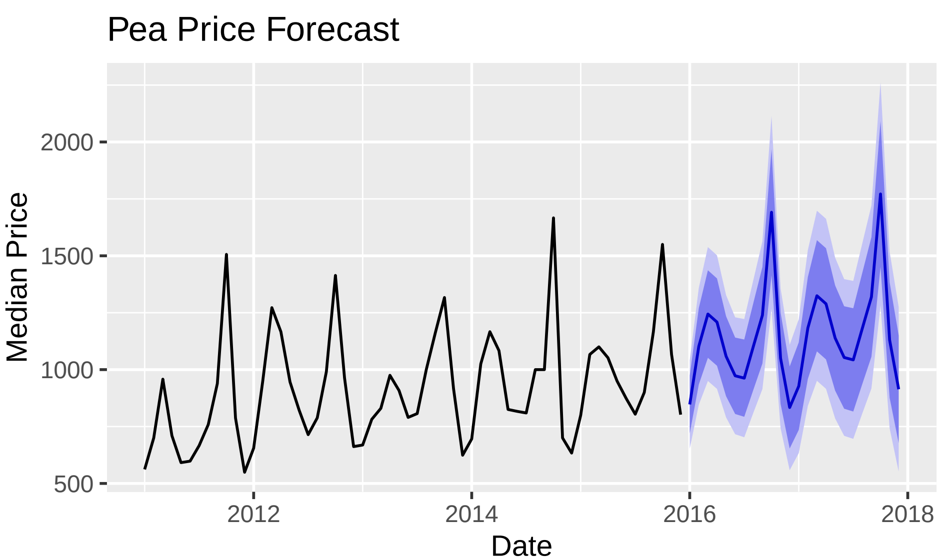
```
##          Point Forecast     Lo 80    Hi 80     Lo 95    Hi 95
## Jan 2016       190.0093 171.35706 208.6615 161.48317 218.5354
## Feb 2016       202.6099 174.14582 231.0740 159.07783 246.1420
## Mar 2016       220.0317 181.72222 258.3413 161.44238 278.6211
## Apr 2016       231.5932 184.48380 278.7026 159.54559 303.6408
## May 2016       226.2626 174.20438 278.3209 146.64641 305.8789
## Jun 2016       229.1587 170.73454 287.5829 139.80665 318.5108
## Jul 2016       230.8787 166.57270 295.1848 132.53113 329.2263
## Aug 2016       251.1739 175.53815 326.8096 135.49902 366.8487
## Sep 2016       279.3573 189.13187 369.5827 141.36943 417.3451
## Oct 2016       262.7887 172.33073 353.2467 124.44516 401.1323
## Nov 2016       236.0485 149.89274 322.2042 104.28465 367.8123
## Dec 2016       205.0924 126.05584 284.1290  84.21640 325.9684
## Jan 2017       205.0036 121.88813 288.1190  77.88948 332.1177
## Feb 2017       218.4941 125.58323 311.4050  76.39917 360.5891
## Mar 2017       237.1698 131.67270 342.6669  75.82591 398.5137
## Apr 2017       249.5154 133.68437 365.3465  72.36711 426.6638
## May 2017       243.6602 125.85363 361.4667  63.49061 423.8297
## Jun 2017       246.6667 122.68387 370.6496  57.05130 436.2822
## Jul 2017       248.4066 118.81644 377.9967  50.21556 446.5976
## Aug 2017       270.1226 124.07681 416.1684  46.76484 493.4804
## Sep 2017       300.3005 132.25584 468.3452  43.29837 557.3027
## Oct 2017       282.3675 119.02591 445.7092  32.55807 532.1770
## Nov 2017       253.5265 102.08787 404.9651  21.92111 485.1319
## Dec 2017       220.1852  84.51341 355.8570  12.69310 427.6773
```

```r
autoplot(potato_forecast, main = 'Potato Price Forecast', xlab='Date', ylab='Median Price')
```



After some data manipulation we have the forecast at last. But do we know if we can trust in this forecast? Well, recall that we inquire that the data depends of the season (low prices at December-January, and a high peak at August). Consequently, a good forecast should show a similar shape throughout the seasons.

```
plot_forecast <- function(time_series, commodity){
price_forecast <- forecast(time_series)
autoplot(price_forecast, main = paste(commodity, 'Price Forecast'), xlab='Date', ylab='Median Price')
}
plot_forecast(pea_ts, "Pea")
```



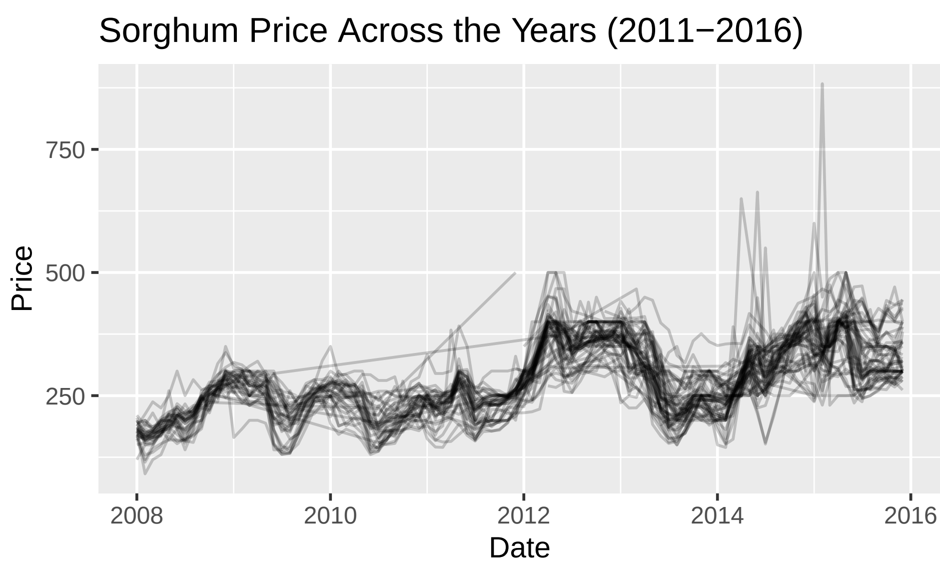
Conclusion and Recommendations

To run a forecasting model we need to convert data into a time series. This can be an usual tool to gather insights of not just the current status of a situation, but to get a view of the possible future it may have.

As shown, there was a lot of effort writing all that code to just analyze the potato data. Fortunately and by good practices, wrapping all the analysis into functions, we could easily reproduce the job with another type of food. Remember there is still more that that can be analyzed, just take the functions we already wrote and do a report.

Example: Sorghum Forecasting

```
commodity <- "Sorghum"
sorghum <- read_data(commodity)
price_plot(sorghum, commodity)
```




```
sorghum_ts <- time_series(sorghum)
plot_forecast(sorghum_ts, commodity)
```