Assignment 2.3: Coffee Shop Sales Time Series Analysis

Edgar Rosales

2024-11-18

```
#library(<PACKAGE_DEPENDENCIES>)
# Load libraries
library(readxl)
library(dplyr)
library(ggplot2)
library(lubridate)
library(forecast)
library(zoo)
```

Data Source

https://www.kaggle.com/datasets/f02d450f34d1dda2c29da2c31e4650dd98562f4887f4dbb1b7b3cd9ec3348191?select=Coffee+Shop+Sales.xlsx

Importing the Data

```
coffee_sales <- read.csv("/Users/edgarrosales/Desktop/UniversitySandiego/MastersProgram/programA/ADS506/

str(coffee_sales)
```

```
## 'data.frame':    149116 obs. of  11 variables:
##  $ transaction_id  : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ transaction_date: chr  "1/1/23" "1/1/23" "1/1/23" "1/1/23" ...
##  $ transaction_time: chr  "7:06:11" "7:08:56" "7:14:04" "7:20:24" ...
##  $ transaction_qty : int  2 2 2 1 2 1 1 2 1 2 ...
##  $ store_id        : int  5 5 5 5 5 5 5 5 5 5 ...
##  $ store_location  : chr  "Lower Manhattan" "Lower Manhattan" "Lower Manhattan" "Lower Manhattan" ..
##  $ product_id      : int  32 57 59 22 57 77 22 28 39 58 ...
##  $ unit_price      : num  3 3.1 4.5 2 3.1 3 2 2 4.25 3.5 ...
##  $ product_category: chr  "Coffee" "Tea" "Drinking Chocolate" "Coffee" ...
##  $ product_type    : chr  "Gourmet brewed coffee" "Brewed Chai tea" "Hot chocolate" "Drip coffee" ..
##  $ product_detail  : chr  "Ethiopia Rg" "Spicy Eye Opener Chai Lg" "Dark chocolate Lg" "Our Old Time
```

```
head(coffee_sales)
```

```
##   transaction_id transaction_date transaction_time transaction_qty store_id
## 1              1           1/1/23          7:06:11               2        5
```

```
## 2              2          1/1/23          7:08:56                2         5
## 3              3          1/1/23          7:14:04                2         5
## 4              4          1/1/23          7:20:24                1         5
## 5              5          1/1/23          7:22:41                2         5
## 6              6          1/1/23          7:22:41                1         5
##   store_location product_id unit_price   product_category
## 1 Lower Manhattan         32        3.0             Coffee
## 2 Lower Manhattan         57        3.1                Tea
## 3 Lower Manhattan         59        4.5 Drinking Chocolate
## 4 Lower Manhattan         22        2.0             Coffee
## 5 Lower Manhattan         57        3.1                Tea
## 6 Lower Manhattan         77        3.0             Bakery
##           product_type              product_detail
## 1 Gourmet brewed coffee                 Ethiopia Rg
## 2       Brewed Chai tea    Spicy Eye Opener Chai Lg
## 3         Hot chocolate           Dark chocolate Lg
## 4           Drip coffee Our Old Time Diner Blend Sm
## 5       Brewed Chai tea    Spicy Eye Opener Chai Lg
## 6                 Scone               Oatmeal Scone
```

```r
# Convert `transaction_date` to Date format
coffee_sales <- coffee_sales %>%
  mutate(transaction_date = mdy(transaction_date))  # Using mdy() for "MM/DD/YY" format

# Check for missing or invalid dates
sum(is.na(coffee_sales$transaction_date))  # Should return 0 if all dates are valid
```

```
## [1] 0
```

```r
# Aggregate data by week to calculate total weekly sales
weekly_sales <- coffee_sales %>%
  mutate(week = floor_date(transaction_date, "week")) %>%  # Extract week
  group_by(week) %>%
  summarise(weekly_sales = sum(transaction_qty * unit_price, na.rm = TRUE)) %>%
  ungroup()

# Aggregate data by day to calculate total daily sales
daily_sales <- coffee_sales %>%
  group_by(transaction_date) %>%  # Group by exact transaction date
  summarise(daily_sales = sum(transaction_qty * unit_price, na.rm = TRUE)) %>%
  ungroup()

# Check the first few rows to confirm the aggregation worked
head(weekly_sales)
```

```
## # A tibble: 6 x 2
##   week       weekly_sales
##   <date>            <dbl>
## 1 2023-01-01       17009.
## 2 2023-01-08       18600.
## 3 2023-01-15       20619.
## 4 2023-01-22       18578.
## 5 2023-01-29       16988.
## 6 2023-02-05       17744.
```

```r
head(daily_sales)
```

```
## # A tibble: 6 x 2
##   transaction_date daily_sales
##   <date>                 <dbl>
## 1 2023-01-01             2508.
## 2 2023-01-02             2403.
## 3 2023-01-03             2565
## 4 2023-01-04             2220.
## 5 2023-01-05             2419.
## 6 2023-01-06             2274.
```

```r
# Convert the weekly sales data to a time series object
weekly_sales_ts <- ts(weekly_sales$weekly_sales, start = c(2023, 1), frequency = 52)

# Convert the daily sales data to a time series object
daily_sales_ts <- ts(daily_sales$daily_sales, start = c(2023, 1, 1), frequency = 365)

# Check the time series structure
print(weekly_sales_ts)
```

```
## Time Series:
## Start = c(2023, 1)
## End = c(2023, 26)
## Frequency = 52
##  [1] 17009.00 18600.35 20618.89 18578.21 16987.64 17744.05 19531.02 20325.92
##  [9] 20401.20 22003.31 23342.89 23424.50 21790.53 25566.86 28569.79 29396.61
## [17] 28155.59 31119.73 35546.93 38158.39 37637.28 33267.91 37023.80 40708.46
## [25] 41036.10 32267.37
```

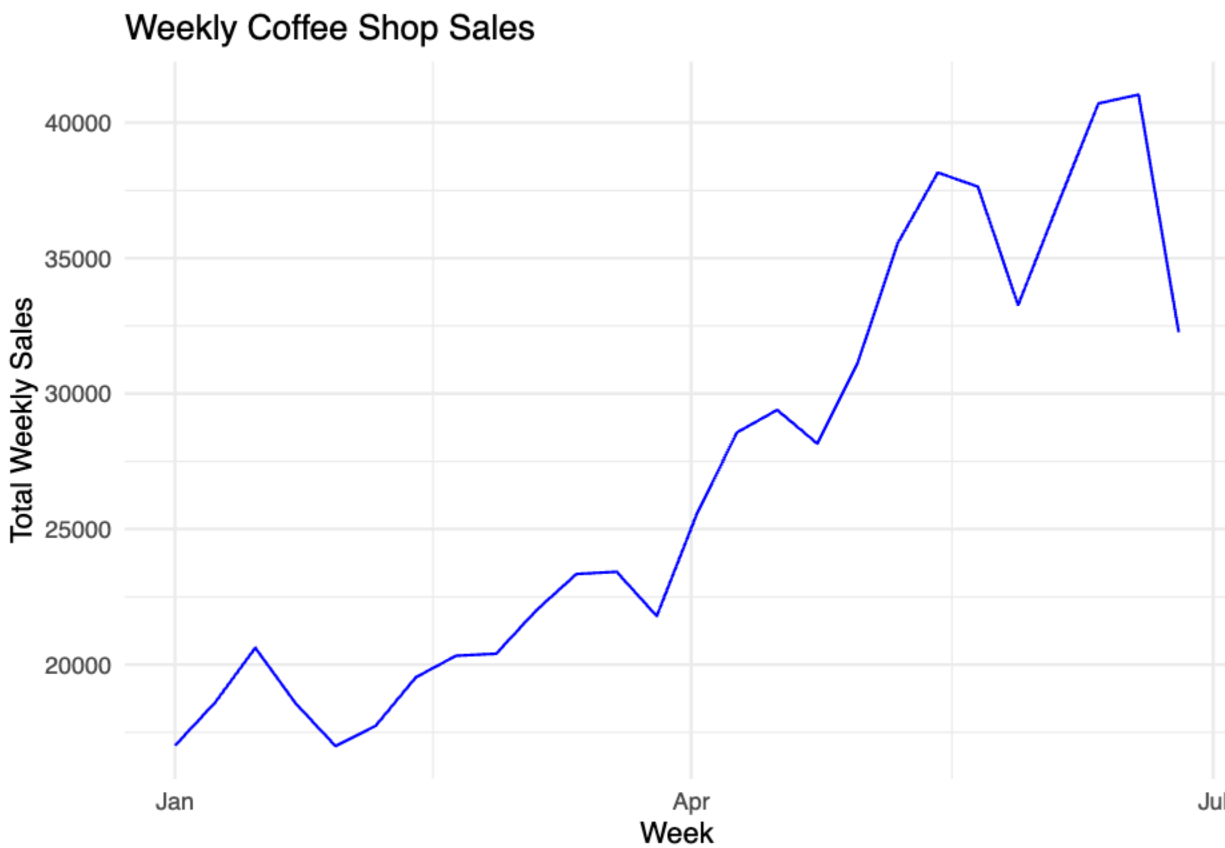
```r
print(daily_sales_ts)
```

```
## Time Series:
## Start = c(2023, 1)
## End = c(2023, 181)
## Frequency = 365
##   [1] 2508.20 2403.35 2565.00 2220.10 2418.85 2273.85 2619.65 2638.53 2676.61
##  [10] 2685.65 2555.75 2327.70 3033.60 2682.51 3167.71 2829.16 3285.80 2735.96
##  [19] 2913.68 2603.73 3082.85 2367.33 2853.15 2868.95 2846.55 2863.03 2742.10
##  [28] 2037.10 2060.75 2476.41 2334.13 2466.30 2506.90 2591.45 2551.70 2304.70
##  [37] 2203.40 2434.55 2762.43 2610.63 2901.60 2526.74 2894.00 2845.48 2673.93
##  [46] 2928.05 3023.33 2300.75 2865.48 3219.60 2883.63 2783.53 2928.70 2746.21
##  [55] 2940.70 2823.55 2956.75 3160.00 2311.10 3040.25 2996.05 3155.15 2781.90
##  [64] 2945.30 2618.05 2803.50 3523.26 3459.97 3441.58 3211.65 3088.33 3627.65
##  [73] 3312.66 3338.03 3386.11 3181.75 3408.36 3340.03 3262.28 3209.80 3284.11
##  [82] 3361.13 3586.20 3380.95 3310.83 3674.35 2792.55 2492.00 2932.82 2888.08
##  [91] 3699.90 3575.85 3604.95 3327.30 3552.70 3250.20 3682.80 4573.06 4088.88
## [100] 4220.30 3852.86 4040.18 4114.53 4131.25 4121.79 4500.39 4332.10 4354.07
## [109] 4318.31 3924.78 4005.38 3961.58 4321.29 4265.45 4255.00 4559.45 4427.10
## [118] 3373.80 2953.50 3552.33 4731.45 4625.50 4714.60 4589.70 4701.00 4205.15
## [127] 4542.70 5604.21 5100.97 5256.33 4850.06 4681.13 5511.53 5052.65 5384.98
## [136] 5542.13 5418.00 5583.47 5657.88 5519.28 5370.81 5541.16 5242.91 5391.45
```

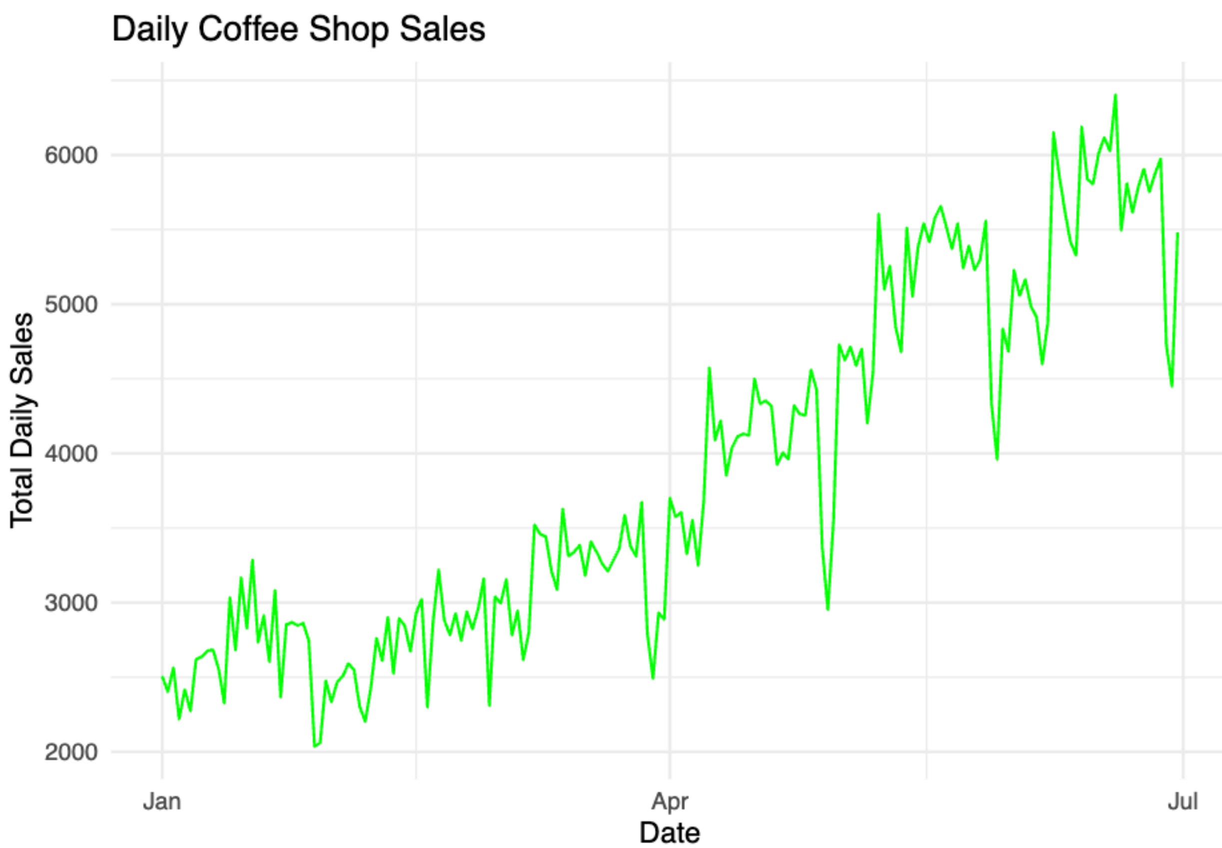
```
## [145] 5230.85 5300.95 5559.15 4338.65 3959.50 4835.48 4684.13 5227.00 5056.50
## [154] 5166.65 4985.15 4911.15 4598.90 4883.10 6151.59 5867.16 5626.75 5418.61
## [163] 5328.70 6189.36 5836.52 5806.24 6011.43 6117.60 6026.09 6403.91 5494.66
## [172] 5808.38 5615.10 5781.86 5906.10 5754.85 5875.90 5975.65 4728.90 4450.75
## [181] 5481.32
```

Time Series Plot

```r
ggplot(weekly_sales, aes(x = week, y = weekly_sales)) +
  geom_line(color = "blue") +
  labs(title = "Weekly Coffee Shop Sales", x = "Week", y = "Total Weekly Sales") +
  theme_minimal()
```




```r
ggplot(daily_sales, aes(x = transaction_date, y = daily_sales)) +
  geom_line(color = "green") +
  labs(title = "Daily Coffee Shop Sales", x = "Date", y = "Total Daily Sales") +
  theme_minimal()
```



Check for Outliers

```
summary(weekly_sales$weekly_sales)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   16988   20345   24496   26877   33018   41036
```

```
summary(daily_sales$daily_sales)
```

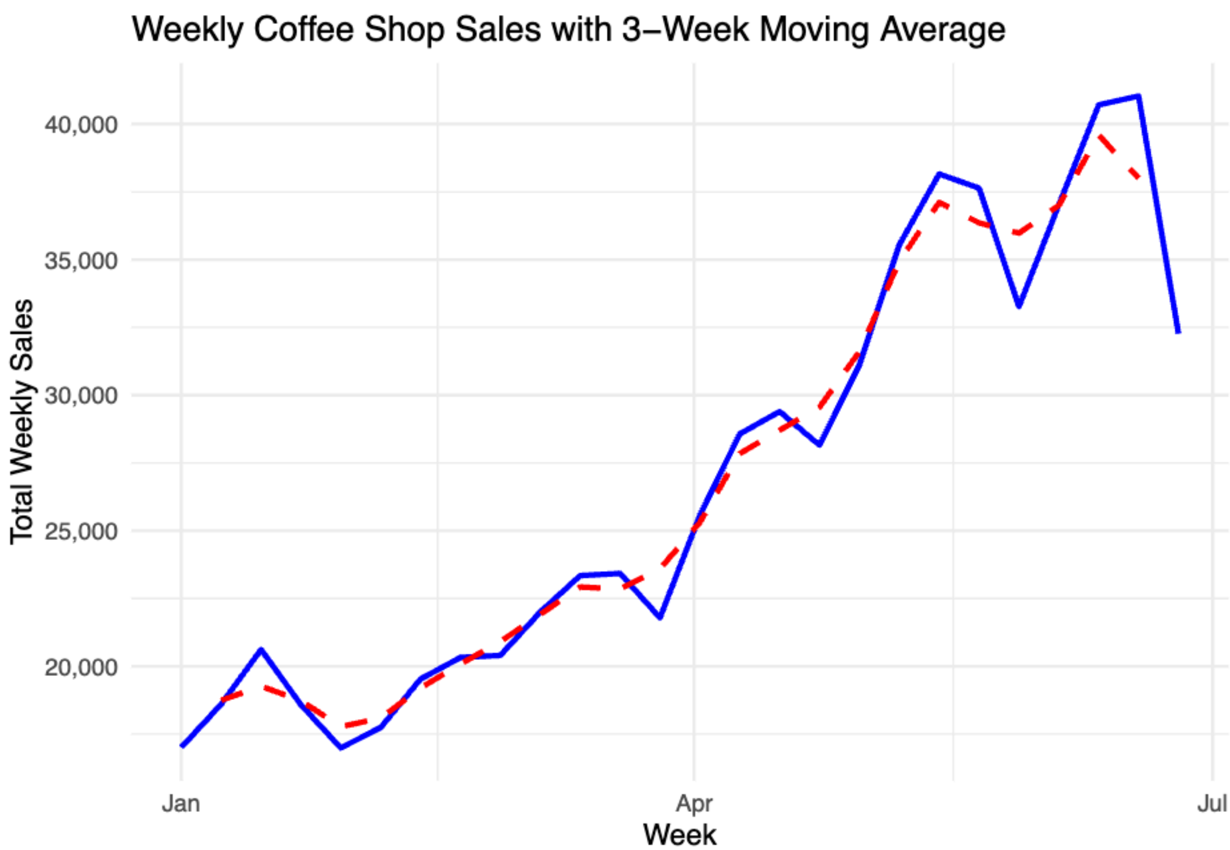
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    2037    2853    3523    3861    4850    6404
```

Weekly Sales with Movign Average

```
# 3-week moving average to smooth the weekly series
weekly_sales <- weekly_sales %>%
  mutate(smoothed_weekly_sales = rollmean(weekly_sales, k = 3, fill = NA))

# Plot original weekly series and smoothed trend line
ggplot(weekly_sales, aes(x = week)) +
```

```
  geom_line(aes(y = weekly_sales), color = "blue", size = 1) +
  geom_line(aes(y = smoothed_weekly_sales), color = "red", size = 1, linetype = "dashed") +
  labs(title = "Weekly Coffee Shop Sales with 3-Week Moving Average",
       x = "Week", y = "Total Weekly Sales") +
  theme_minimal() +
  scale_y_continuous(labels = scales::comma)
```



Daily Sales with Moving Average

```
# 7-day moving average to smooth the daily series (1 week)
daily_sales <- daily_sales %>%
  mutate(smoothed_daily_sales = rollmean(daily_sales, k = 7, fill = NA))

# Plot original daily series and smoothed trend line
ggplot(daily_sales, aes(x = transaction_date)) +
  geom_line(aes(y = daily_sales), color = "green", size = 1) +
  geom_line(aes(y = smoothed_daily_sales), color = "red", size = 1, linetype = "dashed") +
  labs(title = "Daily Coffee Shop Sales with 7-Day Moving Average",
       x = "Date", y = "Total Daily Sales") +
  theme_minimal() +
  scale_y_continuous(labels = scales::comma)
```

Discussion

Weekly

The time series plot for weekly coffee shop sales from January to June 2023 reveals a noticeable upward trend, with a significant increase in sales from early to mid-year. Starting with moderate weekly sales in January, there is a steady rise throughout the following months, particularly in April and May, reaching the highest weekly sales by June. The application of a 3-week moving average helps smooth out short-term fluctuations, emphasizing the ongoing growth in customer demand. This trend suggests that the coffee shop is experiencing increasing popularity and that strategies like marketing or product adjustments are likely contributing to the higher sales figures. Given the observed pattern, the business may need to prepare for continued growth by adjusting inventory, staffing, and operational capacity. However, with just six months of weekly data, it is still difficult to identify clear seasonal effects, which would become more evident with a full year's worth of data. Incorporating longer-term sales trends would allow the business to make more accurate predictions regarding demand cycles. Additionally, external factors such as weather, local events, and promotions could further explain spikes in sales and provide better forecasting models.

Daily

The daily sales data for the same period shows more volatility than the weekly data, with sharp fluctuations in daily sales. Despite this, an upward trend emerges as the months progress, particularly toward the end of the observed period. The application of a 7-day moving average smooths out the extreme fluctuations, highlighting a more consistent increase in demand over time. Similar to the weekly analysis, the daily data indicates a rise in sales as the months progress, with a noticeable jump around mid-year. These daily fluctuations may reflect factors such as varying customer traffic, time-of-day effects, or specific promotions. While the daily data provides a finer granularity, its variability makes it harder to detect longer-term trends without smoothing, which is why the weekly data may offer clearer insights into overall growth patterns. To strengthen the analysis and improve forecasting, collecting data across a full year would be ideal, as it would allow for the identification of seasonal variations that could be crucial for operational planning.

...