

COMSC230 - FINAL PRESENTATION

Made By Brian Brimner, Cam
Sjostedt, and Alexander Oliveira



DATASET

- Sales Transactions Weekly



UC Irvine
Machine Learning
Repository

Datasets Contrib



Sales Transactions Weekly

Donated on 7/15/2017

Contains weekly purchased quantities of 800 over products over 52 weeks.

GitHub Account Details:

<https://github.com/bbrims/COMSC230-Presentation>

Contains:

- Jupyter Notebook file
- Final Report Document
- Powerpoint Presentation
- README

Version Control:

- Continuous updates and collaboration
- Frequent commits
- Mergin for collaborative changes


This Repo is for our final project for COMSC230 - Principles of programming languages




☆ 0 stars 🍴 0 forks 👁 1 watching 🌿 1 Branch 🏷 0 Tags ↻ Activity

🌐 Public repository

🔗 main 🔗 📁

Go to file + <> Code ▾

 **bbrims** Added Graphs ... 2e261c5 · 1 hour ago ⌚

 COMSC230-FinalProject.docx	Added writeup document	3 days ago
 COMSC230-Presentation.ipynb	Added Graphs	1 hour ago
 README.md	Initial commit	3 days ago

Dataset Overview & Structure:

Dataset Overview:

- The dataset contains **819 rows** of weekly sales data across multiple products for **52 weeks**.
- The original dataset used generic placeholders such as P1, P2, and so on, to represent products. These labels were replaced with custom product names, enhancing the dataset's realism and providing a more accurate representation of the data.

Data Structure:

- **Product Codes:** Unique identifiers for each product (e.g., EcoTherm Bottle, SmartHome Speaker).
- **Sales:** Number of items sold per product each week. The key variable used to identify patterns and trends in product performance.
- **Week:** Week number (W0-W51) tracking sales trends across time.

Dataset features:

- Product names (Fictitious)
- Weeks of the year
- Number of sales transactions per week

	Product_Code	W0	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12
0	UltraSoft Towels	11	12	10	8	13	12	14	21	6	14	11	14	16
1	SmartHome Speaker	7	6	3	2	7	1	6	3	3	3	2	2	6
2	EcoTherm Bottle	7	11	8	9	10	8	7	13	12	6	14	9	4
3	PowerMax Charger	12	8	13	5	9	6	9	13	13	11	8	4	5
4	ComfySole Sneakers	8	5	13	11	6	7	9	14	9	9	11	18	8
5	FlexiGrip Hammer	3	3	2	7	6	3	8	6	6	3	1	1	5

Hypotheses

- **Research Hypothesis:** Products show seasonality, with higher / lower sales during specific weeks or periods.
- **Statistical Hypothesis:**
 1. **Null Hypothesis (H_0):**

There is no significant difference in weekly sales across different weeks or periods (no seasonality).
 2. **Alternative Hypothesis (H_1):**

There is a significant difference in weekly sales across different weeks or periods, indicating seasonality.

Potential Sales Trends & Seasonality

Sales Trends Exploration: Weekly sales data reveals product variability:

- Some products consistently perform well, while others have limited sales throughout the year.

Potential Seasonal Patterns: At first glance, the sales data appears stable, but deeper analysis shows potential seasonal fluctuations.

These patterns may be driven by factors like holidays, back-to-school shopping, or other events that influence consumer behavior.

Initial Observations: Sales may experience surges or declines during certain periods, reflecting possible seasonal shifts or external influences. This could result in notable peaks or dips in sales across the year.

Importing CSV file and creating a DataFrame using pandas

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import random
```

Read data from the CSV file into a DataFrame and display

```
# Load the data
data = '/Users/brianbrimner/Downloads/Sales_Transactions_Dataset_Weekly.csv'
product_data = pd.read_csv(data)

# Select all rows and the first 53 columns (Product_Code + 52 weeks of sales data)
product_data_52_weeks = product_data.iloc[:, :53]
```



```
# Define lists of adjectives and product types
adjectives = [
    "Ultra", "Smart", "Eco", "Power", "Comfy", "Flexi", "Mega", "Bright", "Quick", "Max",
    "Silent", "Fresh", "Speed", "Cloud", "Pure", "Safe", "Glow", "Hyper", "Active", "Clean",
    "Pro", "Fit", "Charge", "Aqua", "Super", "Cool", "Warm", "Grip", "Auto", "Travel", "Secure"
]

product_types = [
    "Towels", "Speaker", "Bottle", "Charger", "Sneakers", "Hammer", "Air Purifier", "Desk Lamp",
    "Oven", "Umbrella", "Keyboard", "Monitor", "Vacuum", "Blender", "Kettle", "Power Bank",
    "Headphones", "Yoga Mat", "Fan", "Lockbox", "Water Filter", "Coffee Maker", "Camera", "Cooler",
    "Bike", "Hair Dryer", "Backpack", "Knife Set", "Gloves", "Dishwasher", "Blanket", "Flashlight",
    "Travel Cup", "Washing Machine", "Helmet", "Skateboard", "Mixer", "Pillow", "Dumbbells",
    "Alarm", "Mouse", "Router", "Clock", "Jacket", "Socks", "Phone Case", "Sunglasses", "Fridge",
    "Generator", "Speakers", "Car Charger", "Resistance Bands", "Bulbs", "Food Processor", "Lock",
    "Wireless Charger", "Fitness Tracker", "Lamp", "Shaver", "Torch", "Humidifier", "Water Flask",
    "Cake Pan", "Cooler Box", "Diffuser", "Ladder", "Surfboard", "Paddleboard", "Dish Soap",
    "Iron", "Bag", "Dish Soap", "Thermostat", "Air Conditioner", "Garage Opener"
]
```

```
# Generate unique product names by combining adjectives and product types
unique_product_names = set()
while len(unique_product_names) < len(product_data_52_weeks):
    adjective = random.choice(adjectives)
    product_type = random.choice(product_types)
    product_name = f"{adjective} {product_type}"
    unique_product_names.add(product_name)

# Convert the set to a list
unique_product_names_list = list(unique_product_names)

# Assign the generated names to the Product_Code column
product_data_52_weeks['Product_Code'] = unique_product_names_list[:len(product_data_52_weeks)]
```

- Providing fictitious product names to replace "P1, P2" etc..
- Using random to create unique product names with adjectives and product types.
- More personal feel to the dataset

```
# Display all rows and columns
pd.set_option('display.max_rows', 819) # Adjust to show all rows
pd.set_option('display.max_columns', 53) # Show all columns

# Display the DataFrame
product_data_52_weeks
```

	Product_Code	W0	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10
0	Super Helmet	11	12	10	8	13	12	14	21	6	14	11
1	Smart Hair Dryer	7	6	3	2	7	1	6	3	3	3	2
2	Mega Gloves	7	11	8	9	10	8	7	13	12	6	14
3	Active Travel Cup	12	8	13	5	9	6	9	13	13	11	8
4	Pro Bag	8	5	13	11	6	7	9	14	9	9	11
5	Fresh Fan	3	3	2	7	6	3	8	6	6	3	1
6	Max Car Charger	4	8	3	7	8	7	2	3	10	3	5

- Final DataFrame contains unique product names, along with the number of sales transactions recorded each week for each product throughout the year.

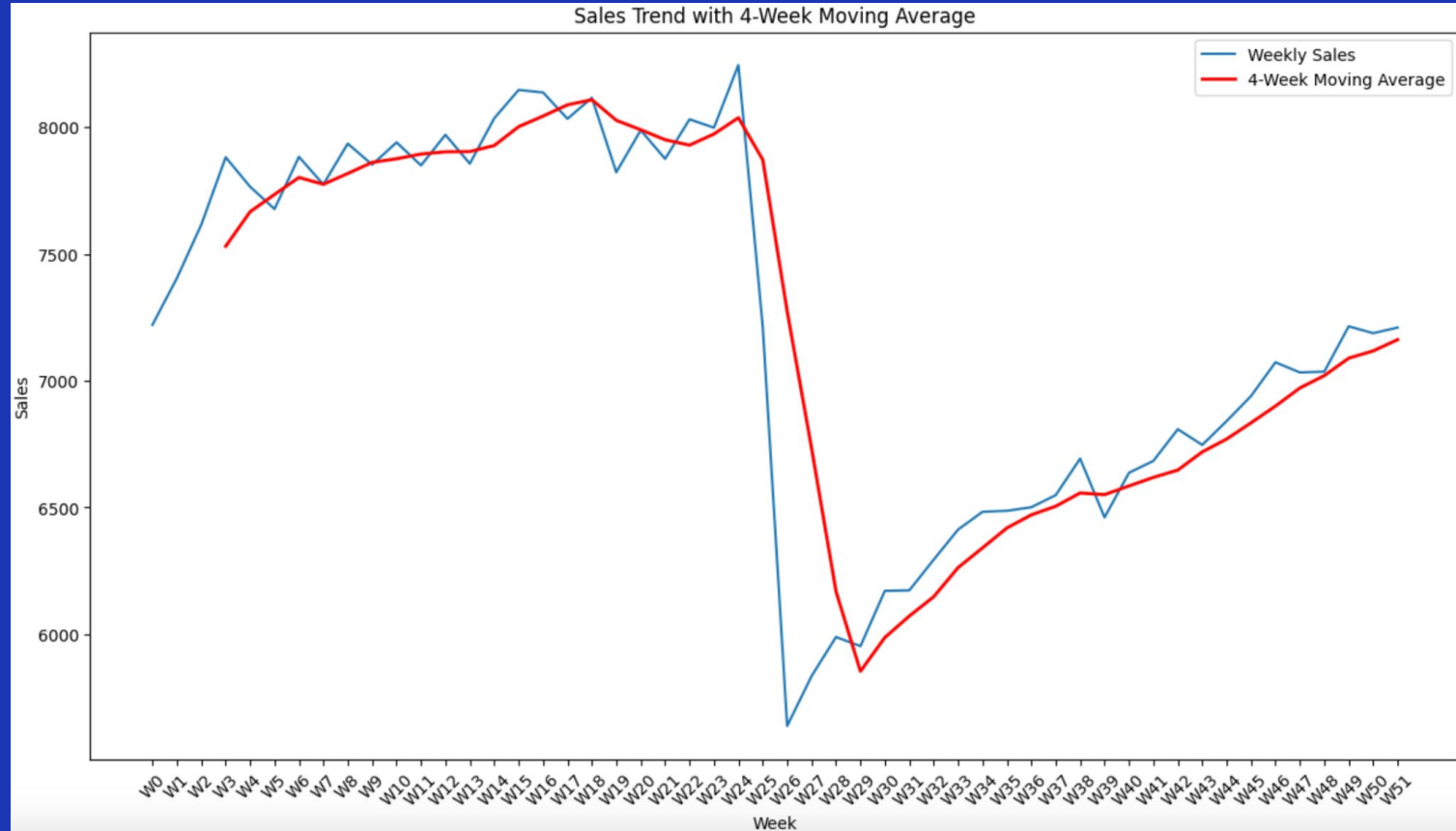
- Descriptive statistics for the average sales of each product
- Average of ~9 sales a week per product

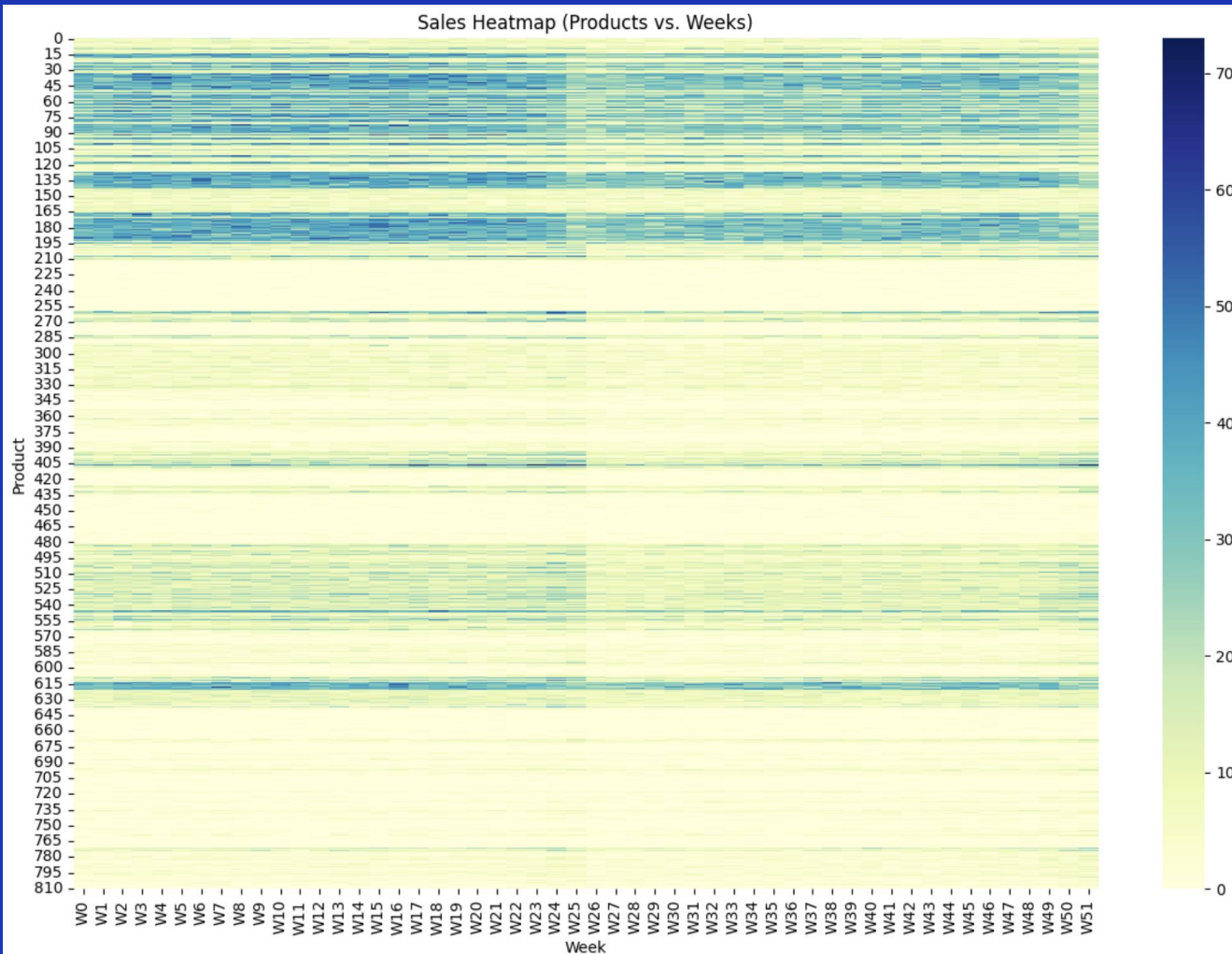
```
# Calculate the average sales for each product across all 52 weeks
average_sales_per_product = product_data_52_weeks.drop(columns='Product_Code').mean(axis=1)

# Descriptive statistics for the average sales of each product
average_sales_description = average_sales_per_product.describe()
print(average_sales_description)
```

```
count      811.000000
mean         8.898961
std         11.420443
min          0.019231
25%          0.403846
50%          3.826923
75%         11.057692
max         42.692308
dtype: float64
```

- Average sales amount per week
- Shows drastic drop in sales around week 25-26. (Mid-Summer)
- Sales begin to rise again throughout the winter





- Heatmap shows average transaction volume per week.
- Line down the middle, left side darker than right
- Shows a major drop off in middle of the year, (Weeks 25-26).
- Consistent with line plot


```
# Calculate a 4-week moving average for better trend visualization
moving_average = total_sales_by_week.rolling(window=4).mean()

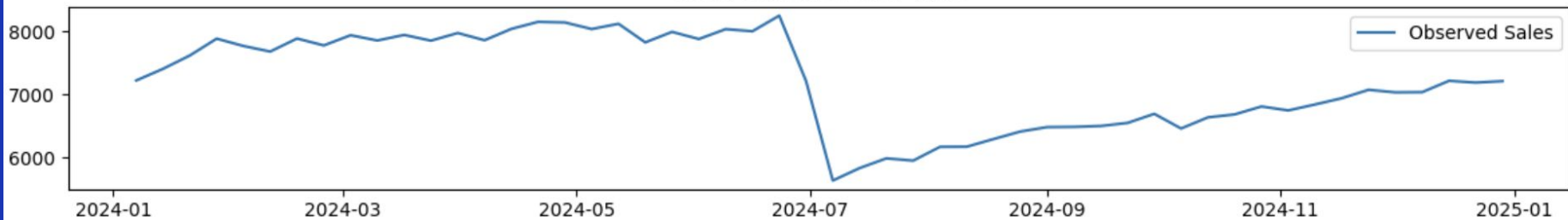
# Plot the moving average along with raw sales
plt.figure(figsize=(15, 8))
sns.lineplot(x=total_sales_by_week.index, y=total_sales_by_week.values, label='Weekly Sales')
sns.lineplot(x=moving_average.index, y=moving_average.values, label='4-Week Moving Average', color='red', linewidth=2)
plt.title('Sales Trend with 4-Week Moving Average')
plt.xlabel('Week')
plt.ylabel('Sales')
plt.xticks(rotation=45)
plt.legend()
plt.show()
```

- Sales Average line plot code^^

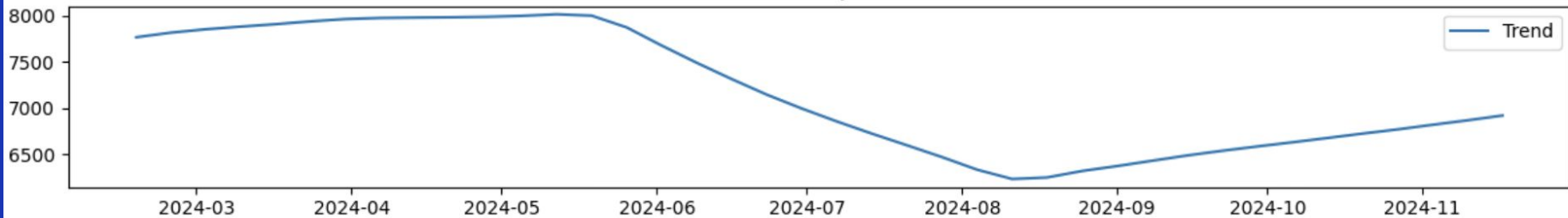
```
plt.figure(figsize=(15, 10))
sns.heatmap(sales_data, cmap='YlGnBu', annot=False)
plt.title('Sales Heatmap (Products vs. Weeks)')
plt.xlabel('Week')
plt.ylabel('Product')
plt.show()
```

← Heatmap Code

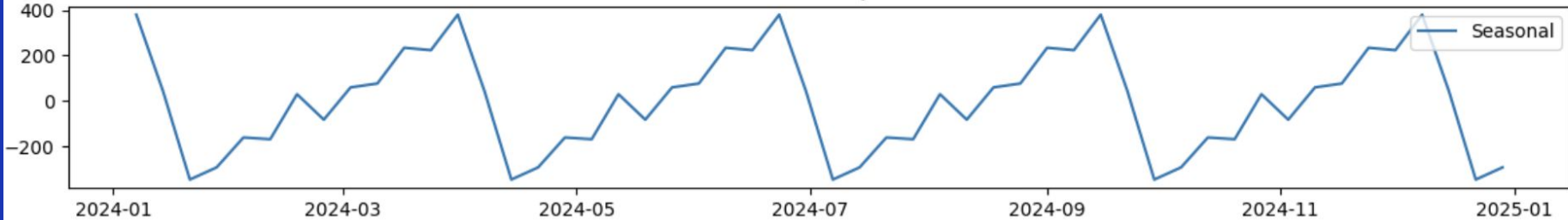
Observed Sales Over Time



Trend Component



Seasonal Component



```

# Step 1: Aggregate the sales data for all products across each week
sales_data = product_data_52_weeks.drop(columns='Product_Code')
weekly_sales = sales_data.sum(axis=0)

# Step 2: Create a time series with weeks as index
weeks = pd.date_range(start="2024-01-01", periods=52, freq="W")
weekly_sales.index = weeks

# Step 3: Decompose the time series into trend, seasonal, and residual components
# Set period=12 for monthly seasonality or period=4 for quarterly
decomposition = seasonal_decompose(weekly_sales, model='additive', period=12)

# Step 4: Plot the decomposition results
plt.figure(figsize=(12, 8))

# Observed sales (raw data)
plt.subplot(411)
plt.plot(decomposition.observed, label='Observed Sales')
plt.title('Observed Sales Over Time')
plt.legend()

# Trend component
plt.subplot(412)
plt.plot(decomposition.trend, label='Trend')
plt.title('Trend Component')
plt.legend()

# Seasonal component
plt.subplot(413)
plt.plot(decomposition.seasonal, label='Seasonal')
plt.title('Seasonal Component')
plt.legend()

# Residuals (Noise)
plt.subplot(414)
plt.plot(decomposition.resid, label='Residuals')
plt.title('Residuals (Noise)')
plt.legend()

plt.tight_layout()
plt.show()

```

- Time Decomposition code

CONCLUSION

Based on our analysis, we believe our alternative hypothesis (H_1) to be true: There is a significant difference in weekly sales across different weeks or periods, indicating seasonality.

- **Summer Dip:** We observed a noticeable dip in sales during the middle of the summer. This decline could be attributed to reduced consumerism, particularly as the school year ends and the summer season begins, which tends to lower demand for many products.
- **Fall Rebound:** As we moved into the fall months, sales saw a significant increase, likely driven by the lead-up to major shopping events like Black Friday and the Christmas season. These periods are known for high consumer spending, and our data supports the trend of rising sales as these events approach.

Thus, we can conclude that our data suggests clear seasonality in sales patterns for these products.