

Cumulative Statistics (Solution)

This document includes the question, the solution, and a breakdown of the code provided in the screenshot.

Uploaded Screenshot

Below is the screenshot of the task:

The screenshot shows a web browser window with the URL `campus.datacamp.com/courses/data-manipulation-with-pandas/aggregating-dataframes?ex=5`. The page is titled "Learn / Courses / Data Manipulation with pandas". The exercise is titled "Cumulative statistics" and is worth 100 XP. The instructions are as follows:

- Sort the rows of `sales_1_1` by the `date` column in ascending order.
- Get the cumulative sum of `weekly_sales` and add it as a new column of `sales_1_1` called `cum_weekly_sales`.
- Get the cumulative maximum of `weekly_sales`, and add it as a column called `cum_max_sales`.
- Print the `date`, `weekly_sales`, `cum_weekly_sales`, and `cum_max_sales` columns.

There is a "Take Hint (-30 XP)" button. The code editor on the right shows the following code:

```
7 # Get the cumulative max of weekly_sales, add as cum_max_sales col
8 ----
9
10 # See the columns you calculated
11 print(sales_1_1[["date", "weekly_sales", "cum_weekly_sales", "cum_max_sales"]])
```

Below the code editor is an "iPython Shell" with "In [1]:" and an empty output area. The bottom of the screenshot shows a Windows taskbar with various icons and the system clock showing 22:26 on 24/11/2024.

Question

1. Sort the rows of `sales_1_1` by the `date` column in ascending order.
2. Get the cumulative sum of `weekly_sales` and add it as a new column of `sales_1_1` called `cum_weekly_sales`.
3. Get the cumulative maximum of `weekly_sales` and add it as a column called `cum_max_sales`.
4. Print the `date`, `weekly_sales`, `cum_weekly_sales`, and `cum_max_sales` columns.

Answer

```
# Sort rows by date
sales_1_1 = sales_1_1.sort_values('date')
```

```
# Get the cumulative sum of weekly_sales
sales_1_1['cum_weekly_sales'] = sales_1_1['weekly_sales'].cumsum()
```

```
# Get the cumulative maximum of weekly_sales
sales_1_1['cum_max_sales'] = sales_1_1['weekly_sales'].cummax()

# Print the specified columns
print(sales_1_1[['date', 'weekly_sales', 'cum_weekly_sales',
'cum_max_sales']])
```

Code Explanation

Explanation of the code:

1. `sales_1_1.sort_values('date')`: Sorts the `sales_1_1` DataFrame by the `date` column in ascending order.
2. `sales_1_1['cum_weekly_sales'] = sales_1_1['weekly_sales'].cumsum()`: Calculates the cumulative sum of the `weekly_sales` column and adds it as a new column called `cum_weekly_sales`.
3. `sales_1_1['cum_max_sales'] = sales_1_1['weekly_sales'].cummax()`: Calculates the cumulative maximum of the `weekly_sales` column and adds it as a new column called `cum_max_sales`.
4. `print(sales_1_1[['date', 'weekly_sales', 'cum_weekly_sales', 'cum_max_sales']])`: Prints the `date`, `weekly_sales`, `cum_weekly_sales`, and `cum_max_sales` columns from the `sales_1_1` DataFrame.