

Pivoting on One Variable - Corrected Instruction 1

The screenshot shows a web browser window with multiple tabs. The active tab is 'campus.datacamp.com/courses/data-manipulation-with-pandas/aggregating-dataframes?ex=14'. The page content is from a DataCamp course titled 'Data Manipulation with pandas'. It features an 'Exercise' section for 'Pivoting on one variable'. The instructions explain that pivot tables are a standard way of aggregating data in spreadsheets and that in pandas, they are essentially another way of performing grouped calculations. The exercise involves using `.pivot_table()` to replicate calculations from a previous lesson using `.groupby()`. The instructions are numbered 1 through 3, with a 'Take Hint (-10 XP)' button. To the right of the instructions is a code editor with a Python script. The script defines `mean_sales_by_type` using `sales.pivot_table()`, prints it, then defines `mean_med_sales_by_type` using `np.median()` and `sales.pivot_table()`, prints it, and finally defines `mean_sales_by_type_holiday` using `sales.pivot_table()` with `is_holiday` as a column, prints it, and submits the answer. The bottom of the browser shows a Windows taskbar with the date 25/11/2024 and time 15:25.

Exercise

Pivoting on one variable

Pivot tables are the standard way of aggregating data in spreadsheets.

In pandas, pivot tables are essentially another way of performing grouped calculations. That is, the `.pivot_table()` method is an alternative to `.groupby()`.

In this exercise, you'll perform calculations using `.pivot_table()` to replicate the calculations you performed in the last lesson using `.groupby()`.

`sales` is available and `pandas` is imported as `pd`.

Instructions 1/3 35 XP

- Get the mean `weekly_sales` by type using `.pivot_table()` and store as `mean_sales_by_type`.
[Take Hint \(-10 XP\)](#)
- Get the mean and median (using NumPy functions) of `weekly_sales` by type using `.pivot_table()` and store as `mean_med_sales_by_type`.
- Get the mean of `weekly_sales` by type and `is_holiday` using `.pivot_table()` and store as `mean_sales_by_type_holiday`.

```
1 # Pivot for mean weekly_sales for each store type
2 mean_sales_by_type = sales.pivot_table(values="weekly_sales", index="type", aggfunc="mean")
3
4 # Print mean_sales_by_type
5 print(mean_sales_by_type)
6
7 # Pivot for mean and median weekly_sales by type
8 mean_med_sales_by_type = sales.pivot_table(values="weekly_sales", index="type", aggfunc=
9 ["mean", "median"])
10
11 # Print mean_med_sales_by_type
12 print(mean_med_sales_by_type)
13
14 # Pivot for mean weekly_sales by type and is_holiday
15 mean_sales_by_type_holiday = sales.pivot_table(values="weekly_sales", index="type",
16 columns="is_holiday", aggfunc="mean")
17
18 # Print mean_sales_by_type_holiday
19 print(mean_sales_by_type_holiday)
```

[Run Code](#) [Submit Answer](#)

Pivot tables are the standard way of aggregating data in spreadsheets.

In pandas, pivot tables are essentially another way of performing grouped calculations. That is, the `pivot_table()` method is an alternative to `.groupby()`.

In this exercise, you'll perform calculations using `.pivot_table()` to replicate the calculations you performed in the last lesson using `.groupby()`.

`sales` is available and `pandas` is imported as `pd`.

Corrected Final Answer - Instruction 1

```
# Pivot for mean weekly_sales for each store type
mean_sales_by_type = sales.pivot_table(values="weekly_sales",
index="type")
```

```
# Print mean_sales_by_type
print(mean_sales_by_type)
```