

Pandas Drop Duplicates

Removing duplicates is an essential skill to get accurate counts because you often don't want to count the same thing multiple times. In Python, this could be accomplished by using the Pandas module, which has a method known as drop_duplicates.

Let's understand how to use it with the help of a few examples.

Dropping Duplicate Names

Let's say you have a dataframe that contains vet visits, and the vet's office wants to know how many dogs of each breed have visited their office. However, there are dogs like Max and Stella, who have visited the vet more than once in your dataset. Hence, you cannot just count the number of each breed in the breed column.

```
print(vet_visits)
0 2018-09-02 Bella Labrador
                              24.87
1 2019-06-07
                              28.35
              Lucy Chow Chow
71 2018-01-20 Stella Chihuahua
                                2.83
               Max Chow Chow
72 2019-06-07
               Lucy Chow Chow
73 2018-08-20
74 2019-04-22
                               28.54
```





You would do this using the drop_duplicates method. It takes an argument subset, which is the column we want to find or duplicates based on - in this case, we want all the unique names.

```
vet_visits.drop_duplicates(subset="name")
0 2018-09-02 Bella Labrador 24.87
1 2019-06-07 Max Chow Chow 24.01
3 2018-01-17 Stella Chihuahua 1.51
               Max Labrador
```

But, what if we have dogs with the same name?

Dropping Duplicate Pairs

In that case, we need to consider more than just name when dropping duplicates. Since Max and Max are different breeds, we can drop the rows with pairs of names and breeds listed earlier in the dataset.

```
unique_dogs = vet_visits.drop_duplicates(subset=["name", "breed"])
print(unique_dogs)
```

```
0 2018-09-02 Bella Labrador 24.87
1 2019-03-13 Max Chow Chow 24.13
3 2018-01-17 Stella Chihuahua 1.51
                              24.07
```





To base our duplicate dropping on multiple columns, we can pass a list of column names to the subset argument, in this case, name and breed.

Now both Max's have been included.

Interactive Example

In this exercise, you'll create some new DataFrames using unique values from sales. sales is available, and pandas is imported as pd.

You will perform the following steps:

- First, you will remove rows of sales with duplicate pairs of store and type and save as store_types and print the head.
- Then, you will remove rows of sales with duplicate pairs of store and department and save as store_depts and print the head.
- Subset the rows that are holiday weeks, and drop the duplicate dates, saving as holiday_dates.
- Finally, select the date column of holiday_dates, and print the holiday_dates dataframe.

```
store_types = sales.drop_duplicates(subset=["store", "type"])
print(store_types.head())
store_depts = sales.drop_duplicates(subset=["store", "department"])
print(store_depts.head())
holiday_dates = sales[sales["is_holiday"]].drop_duplicates(subset="date")
```



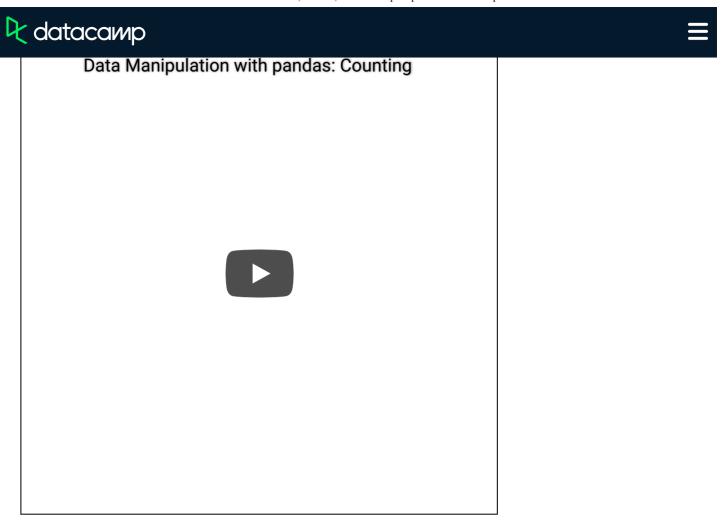


When we run the above code, it produces the following result:

store type of	lepartment c	late weekly_sale	s is_holida	y temperatur	e_c fuel_price	e_usd_per_l	une
0 1 A	1 2010-02-0	5 24924.50	False	5.728	0.679	8.106	
901 2 A	1 2010-02-	05 35034.06	False	4.550	0.679	8.324	
1798 4 A	1 2010-02	-05 38724.42	2 False	6.533	0.686	8.623	
2699 6 A	1 2010-02	-05 25619.00) False	4.683	0.679	7.259	
3593 10 B	1 2010-02	2-05 40212.8	4 False	12.411	0.782	9.765	
store type de	partment da	ite weekly_sales	is_holiday	temperature	_c fuel_price_	_usd_per_l u	unen
0 1 A	1 2010-02-05	24924.50	False	5.728	0.679	8.106	
12 1 A	2 2010-02-05	5 50605.27	False	5.728	0.679	8.106	
24 1 A	3 2010-02-05	13740.12	False	5.728	0.679	8.106	
36 1 A	4 2010-02-05	39954.04	False	5.728	0.679	8.106	
48 1 A	5 2010-02-05	32229.38	False	5.728	0.679	8.106	
498 2010-09-	-10						
691 2011-11-25							
2315 2010-02	2-12						
6735 2012-09	9-07						
6810 2010-12	2-31						
6815 2012-02	2-10						
6820 2011-09)-09						
Name: date, dty	pe: datetime64	[ns]					

Try it for yourself.

To learn more about counting and aggregating data, please see this video from our course Data Manipulation with pandas.



This content is taken from DataCamp's Data Manipulation with pandas course by Maggie Matsui and Richie Cotton.

