Lecture 4
Split-Apply-Combine Paradigm

Dennis Sun
Stanford University
DATASCI / STATS 112

January 18, 2023

```
[2] # Read in the Titanic data set using the Pandas `read_csv` function.
    df_titanic = pd.read_csv("https://dlsun.github.io/stats112/data/titanic.csv")

    # To look at the data, we make `df_titanic` the last line of the cell so that
    # the output is printed.
    df_titanic
```

	name	pclass	survived	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	home.dest
0	Allen, Miss. Elisabeth Walton	1	1	female	29.0000	0	0	24160	211.3375	B5	S	2	NaN	St Louis, MO
1	Allison, Master. Hudson Trevor	1	1	male	0.9167	1	2	113781	151.5500	C22 C26	S	11	NaN	Montreal, PQ / Chesterville, ON
2	Allison, Miss. Helen Loraine	1	0	female	2.0000	1	2	113781	151.5500	C22 C26	S	NaN	NaN	Montreal, PQ / Chesterville, ON
3	Allison, Mr. Hudson Joshua Creighton	1	0	male	30.0000	1	2	113781	151.5500	C22 C26	S	NaN	135.0	Montreal, PQ / Chesterville, ON
4	Allison, Mrs. Hudson J C (Bessie Waldo Daniels)	1	0	female	25.0000	1	2	113781	151.5500	C22 C26	S	NaN	NaN	Montreal, PQ / Chesterville, ON
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1304	Zabour, Miss. Hileni	3	0	female	14.5000	1	0	2665	14.4542	NaN	C	NaN	328.0	NaN
1305	Zabour, Miss. Thamine	3	0	female	NaN	1	0	2665	14.4542	NaN	C	NaN	NaN	NaN
1306	Zakarian, Mr. Mapriededer	3	0	male	26.5000	0	0	2656	7.2250	NaN	C	NaN	304.0	NaN
1307	Zakarian, Mr. Ortin	3	0	male	27.0000	0	0	2670	7.2250	NaN	C	NaN	NaN	NaN
1308	Zimmerman, Mr. Leo	3	0	male	29.0000	0	0	315082	7.8750	NaN	S	NaN	NaN	NaN

1309 rows × 14 columns

1. Booleans in Pandas
2. The Split-Apply-Combine Paradigm
3. In-Class Exercise
4. Reminders

What do you think the following code will produce?

```
df_titanic["pclass"] == 3
```

```
0       False
1       False
2       False
3       False
4       False
        ...
1304     True
1305     True
1306     True
1307     True
1308     True
Name: pclass, Length: 1309, dtype: bool
```

a Series of Booleans

indicates whether each passenger was in 3rd class or not

another example of vectorization!

What about the following?

```
(df_titanic["pclass"] == 3).sum()
```

```
709
```

the number of passengers in 3rd class

Boolean Series

How would you interpret the following?

```
(df_titanic["pclass"] == 3).mean()
```

```
0.5416348357524828
```

the proportion of passengers in 3rd class

What You Need to Know about Booleans

- Applying a relational operator like `==`, `<`, `>`, and `!=` on a **`Series`** produces a **`Series`** of booleans, by vectorization.
- Arithmetic operations can be performed on booleans in **`Series`**, treating `True` as 1 and `False` as 0.

Boolean Masks

We can pass a boolean `Series` as a mask to a `DataFrame` to filter the data.

```
df_titanic[df_titanic["pclass"] == 3]
```

Note the index!

	name	pclass	survived	sex	age	sibsp	parch	ticket	fare	cabin	embarked	boat	body	home.dest
600	Abbing, Mr. Anthony	3	0	male	42.0	0	0	C.A. 5547	7.5500	NaN	S	NaN	NaN	NaN
601	Abbott, Master. Eugene Joseph	3	0	male	13.0	0	2	C.A. 2673	20.2500	NaN	S	NaN	NaN	East Providence, RI
602	Abbott, Mr. Rossmore Edward	3	0	male	16.0	1	1	C.A. 2673	20.2500	NaN	S	NaN	190.0	East Providence, RI
603	Abbott, Mrs. Stanton (Rosa Hunt)	3	1	female	35.0	1	1	C.A. 2673	20.2500	NaN	S	A	NaN	East Providence, RI
604	Abelseth, Miss. Karen Marie	3	1	female	16.0	0	0	348125	7.6500	NaN	S	16	NaN	Norway Los Angeles, CA
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1304	Zabour, Miss. Hileni	3	0	female	14.5	1	0	2665	14.4542	NaN	C	NaN	328.0	NaN
1305	Zabour, Miss. Thamine	3	0	female	NaN	1	0	2665	14.4542	NaN	C	NaN	NaN	NaN
1306	Zakarian, Mr. Mapriededer	3	0	male	26.5	0	0	2656	7.2250	NaN	C	NaN	304.0	NaN
1307	Zakarian, Mr. Ortin	3	0	male	27.0	0	0	2670	7.2250	NaN	C	NaN	NaN	NaN
1308	Zimmerman, Mr. Leo	3	0	male	29.0	0	0	315082	7.8750	NaN	S	NaN	NaN	NaN

Exercise

How would we calculate the average fare paid by a passenger in 3rd class?

```
df_titanic[df_titanic["pclass"] == 3]["fare"].mean()
```

```
13.302888700564973
```

1 Booleans in Pandas

2 The Split-Apply-Combine Paradigm

3 In-Class Exercise

4 Reminders

Another Exercise

How would we calculate the average fare paid by a passenger in *each* class?

```
for i in range(1, 4):
  print(df_titanic[df_titanic["pclass"] == i]["fare"].mean())
```

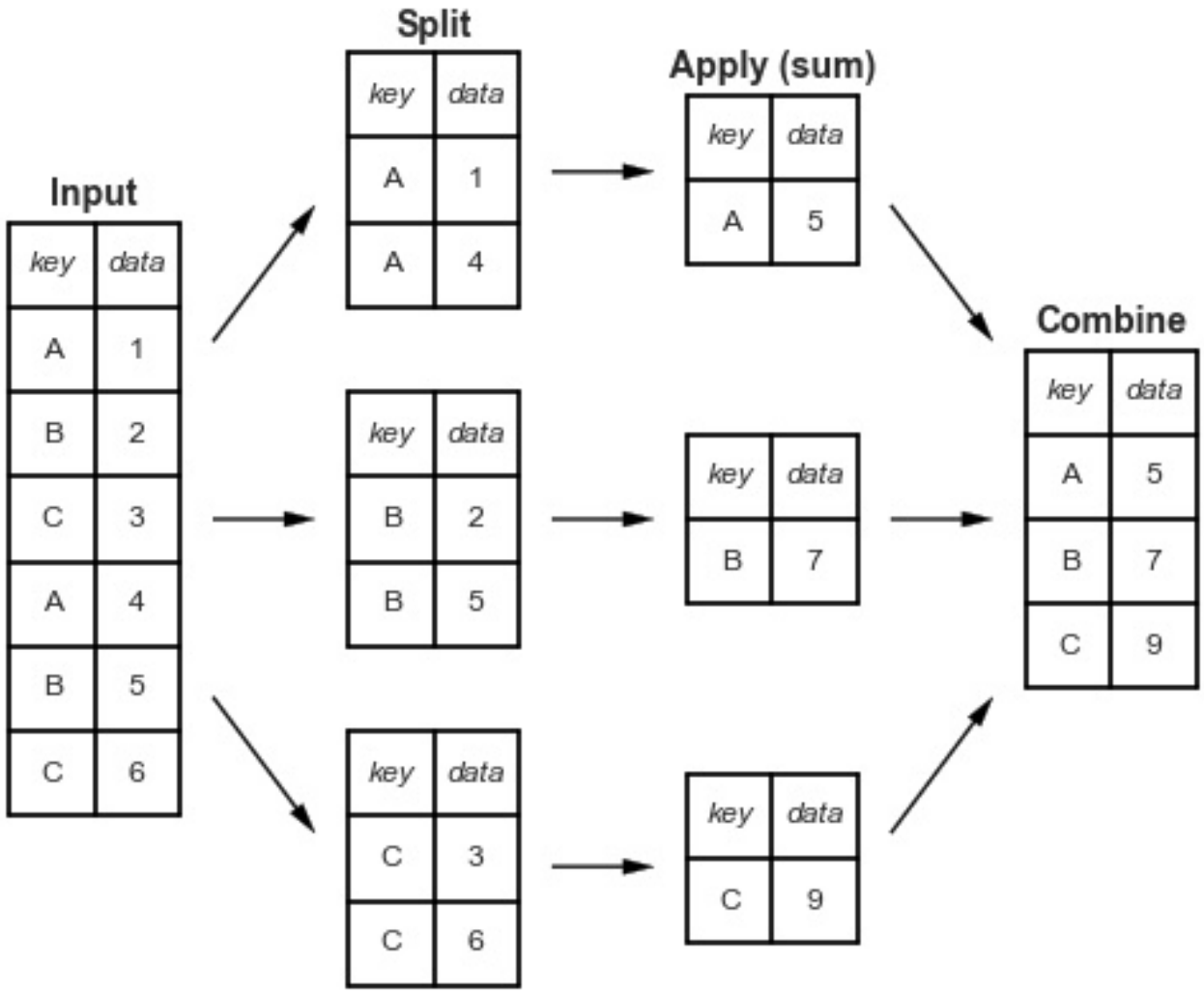
```
87.50899164086688
21.179196389891697
13.302888700564973
```

Problems with this Solution

- This is inconvenient (have to write a `for` loop over the possible values).
- The values are not stored in a Pandas object for further analysis.

The Split-Apply-Combine Paradigm

The problem fits into the **split-apply-combine paradigm** (Wickham, 2011).



Split-Apply-Combine in Pandas

The split-apply-combine paradigm is implemented in Pandas using the `.groupby()` method.

```
df_titanic.groupby("pclass")["fare"].mean()
```

```
pclass
1    87.508992
2    21.179196
3    13.302889
Name: fare, dtype: float64
```

The values are in a Series for easy analysis!

Splitting on Multiple Variables

You can call `.groupby()` on multiple variables.

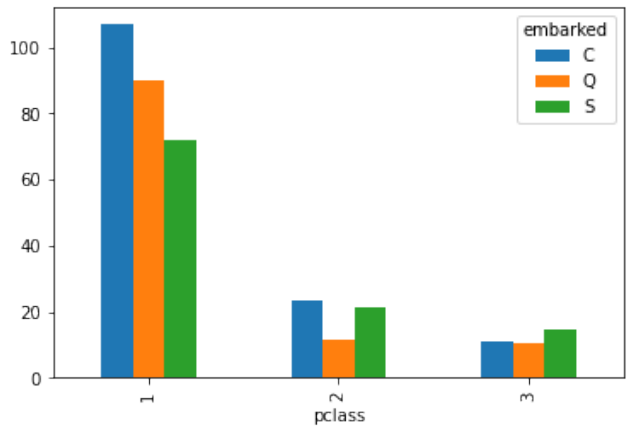
```
df_titanic.groupby(["pclass", "embarked"])["fare"].mean()
```

```
pclass  embarked
1       C           106.845330
        Q            90.000000
        S            72.148094
2       C            23.300593
        Q            11.735114
        S            21.206921
3       C            11.021624
        Q            10.390820
        S            14.435422
Name: fare, dtype: float64
```

```
.unstack("embarked")
```

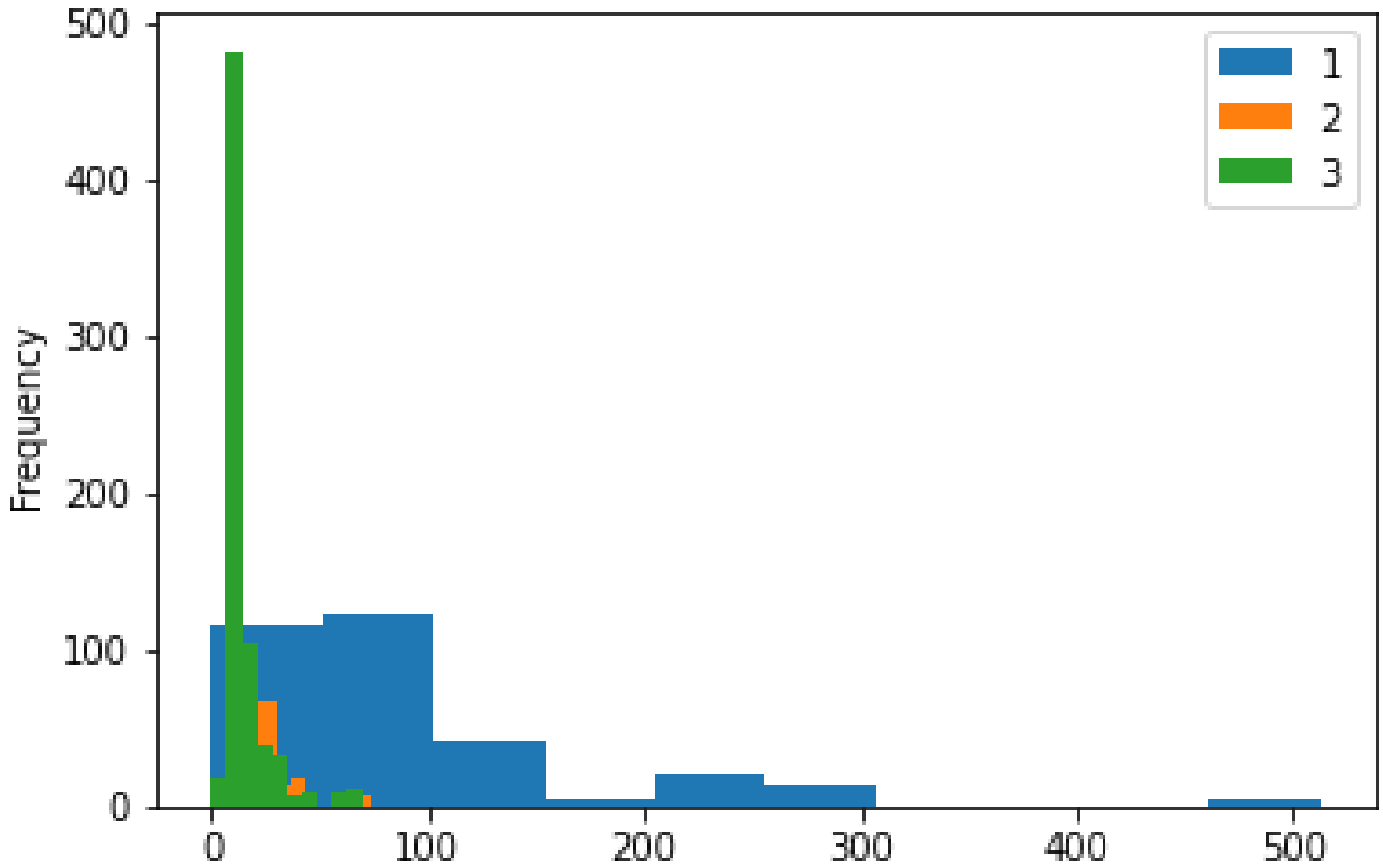
embarked	C	Q	S
pclass			
1	106.845330	90.000000	72.148094
2	23.300593	11.735114	21.206921
3	11.021624	10.390820	14.435422

```
.plot.bar()
```



This Trick Works on Lots of Methods!

```
df_titanic.groupby("pclass")["fare"].plot.hist(legend=True)
```

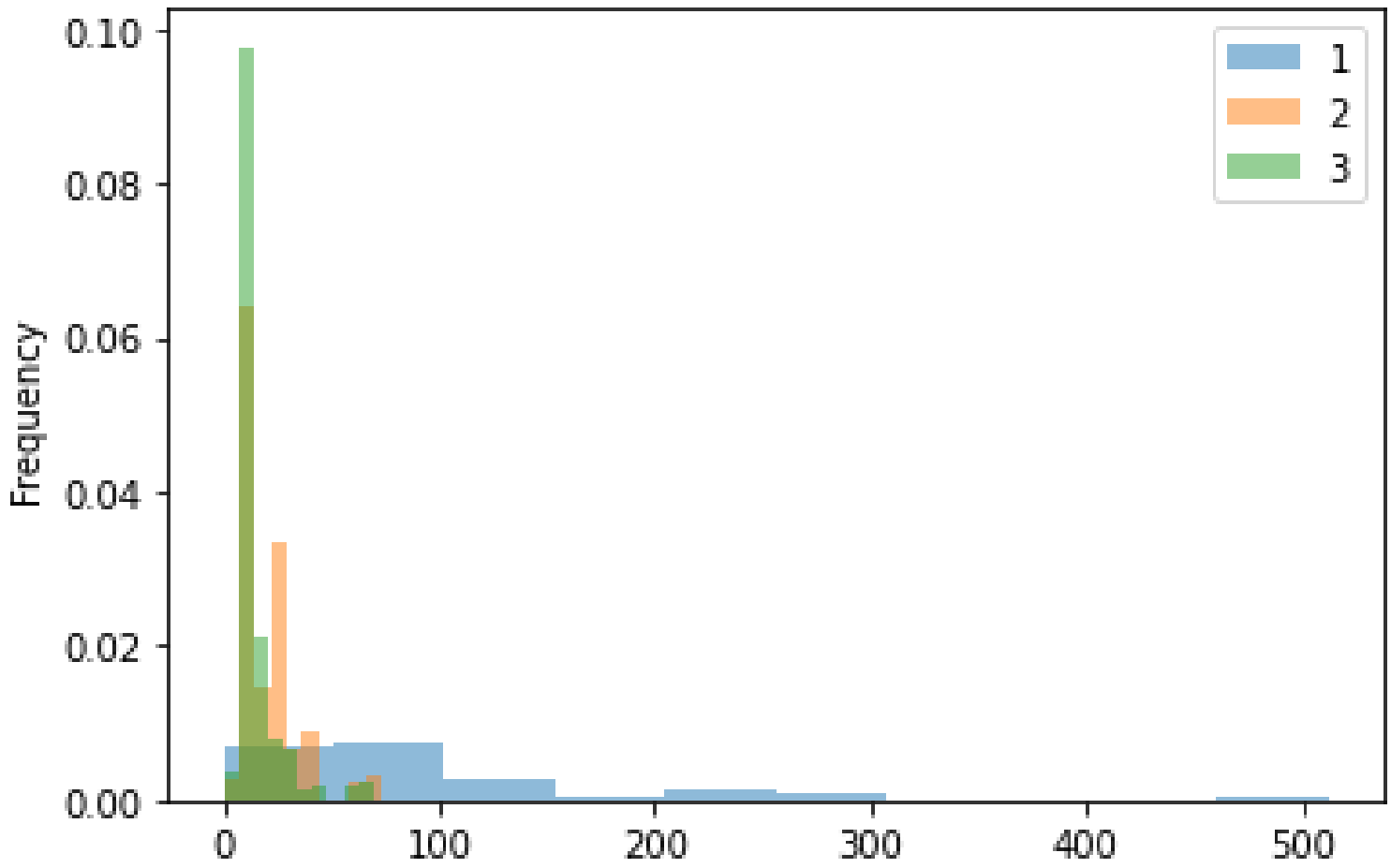


Why is this graph misleading?

Comparing Distributions

Density histograms are better for comparisons.

```
df_titanic.groupby("pclass")["fare"].plot.hist(legend=True,
                                               alpha=0.5,
                                               density=True)
```



1 Booleans in Pandas

2 The Split-Apply-Combine Paradigm

3 In-Class Exercise

4 Reminders

In-Class Exercise

Click on the logo below to be taken to the Colab.

1. Booleans in Pandas
2. The Split-Apply-Combine Paradigm
3. In-Class Exercise
4. Reminders

In-Class Exercise

- Work on the Colab for tomorrow's section!
- Assignment 1 due Friday. Uploaded to Gradescope by 9 AM.
- Exam 1 is next Friday. More details on Friday.