Applied Data Analytics

Pandas basics

Grouping DataFrames by rows

Hans-Martin von Gaudecker and Aapo Stenhammar

Grouped data is required all the time

- size of groups
- calculate shares by some other variable
- calculate group differences in some variables

Example: PIAAC data

country	age_group	use_computer_at_work	programs_monthly
Germany	Aged 30-34	0.776	0.079
	Aged 55-59	0.679	0.035
Netherlands	Aged 30-34	0.872	0.096
	Aged 55-59	0.802	0.028

- Index: `country`, `age_group`
- Float columns: `use_computer_at_work`, `programs_monthly`
- *(Already grouped)* — Goal: Broaden groups, calculate statistics

Calculating grouped values: Two steps

1. Generate a grouped object
2. Perform an operation on that

Resulting object will be a DataFrame (*almost always of smaller size*).

Calculating means by country

```
[1] df.groupby("country")
[1] <pandas.core.groupby.generic.DataFrameGroupBy object at 0x7f6da889ba50>

[2] df.groupby("country").mean()
```

country	use_computer_at_work	programs_monthly
Germany	0.728	0.057
Netherlands	0.837	0.062

Calculating standard dev's by age group

```
[1] df.groupby("age_group")
[1] <pandas.core.groupby.generic.DataFrameGroupBy object at 0x7f6da88b5d90>

[2] df.groupby("age_group").std()
```

age_group	use_computer_at_work	programs_monthly
Aged 30-34	0.067	0.012
Aged 55-59	0.087	0.005

Important methods on *groupby*-objects

Method	Description	Applies to
mean	Averages	floats, (ints)
std	Standard deviation	floats, (ints)
median / quantile	Quantiles	floats, ints
min / max	Minimum / Maximum	anything that is ordered
count	Number of non-missing observations	any
value_counts	Number of observations per value	categorical, (ints)
apply	Pass your own function	depends

Semantics may change depending on whether you pass a single column or more!