Lesson 09 Summary: GroupBy Operations

Data Science with Python – Key Concepts

Data Science Program

GroupBy Overview

GroupBy: Split-Apply-Combine

Key Functions:

`mean()` | `sum()` | `count()` | `std()` | `min()` | `max()` | `first()`

agg() returns one row per group; transform() keeps original shape

Split-Apply-Combine is the core groupby paradigm

The Split-Apply-Combine Pattern

Three-step process for grouped operations:

1. **Split:** Divide data into groups by key column
2. **Apply:** Perform operation on each group independently
3. **Combine:** Merge results into output structure

Basic syntax:

```
df.groupby('Sector')['Return'].mean()
```

GroupBy creates a GroupBy object for deferred computation

Aggregation Functions

Common aggregation methods:

- **Central tendency:** `mean()`, `median()`, `first()`, `last()`
- **Dispersion:** `std()`, `var()`, `min()`, `max()`
- **Counting:** `count()`, `nunique()`, `size()`
- **Sums:** `sum()`, `cumsum()`, `prod()`

Example:

```
df.groupby('Sector')['Volume'].sum()
```

All aggregations reduce each group to a single value

agg() Method

Apply multiple aggregations at once:

- **List:** `df.groupby('Sector')['Return'].agg(['mean', 'std'])`
- **Dict:** `.agg({'Return': 'mean', 'Volume': 'sum'})`
- **Named:** `.agg(avg_ret=('Return', 'mean'))`

Finance example:

```
df.groupby('Sector').agg(
    mean_return=('Return', 'mean'),
    volatility=('Return', 'std')
)
```

agg() provides flexible multi-column aggregation

transform() Method

Broadcast group results to original shape:

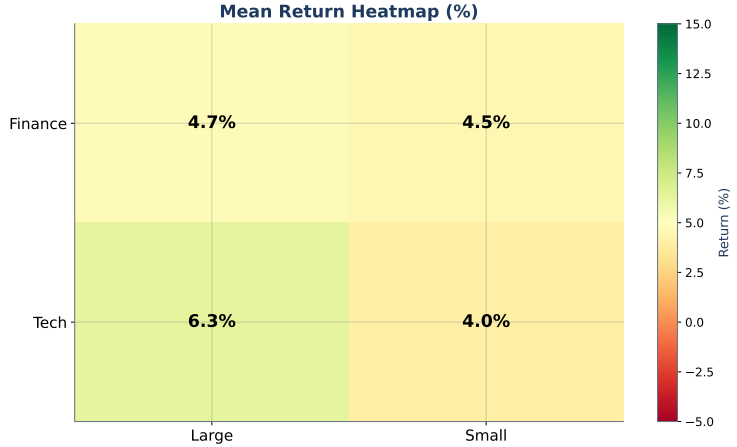
- **Returns same length as input** (unlike agg)
- **Use case:** Normalize within groups
- **Syntax:** `df.groupby('Sector')['Return'].transform('mean')`

Normalize returns by sector:

```
df['zscore'] = df.groupby('Sector')['Return'].transform(
    lambda x:  (x - x.mean()) / x.std()
)
```

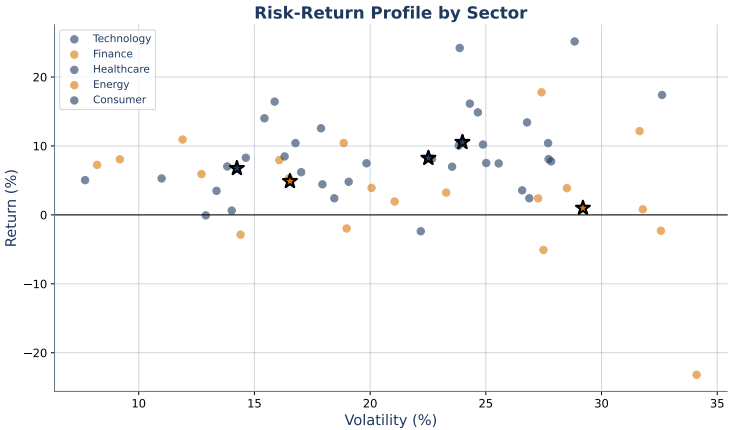
transform() keeps original index alignment

Multi-Column Grouping



groupby(['Sector', 'Year']) creates hierarchical groups

Sector Analysis Example



Calculate risk-return metrics by sector for comparison

Common Patterns

Useful GroupBy techniques:

- **Filter:** `.filter(lambda x: x['Return'].mean() > 0)`
- **Apply:** `.apply(custom_function)`
- **Reset index:** `.reset_index()` or `as_index=False`

Performance tips:

- Use `sort=False` if order doesn't matter
- Use `observed=True` for categorical columns

GroupBy is optimized for common operations

Quick Reference

Essential GroupBy Operations:

Operation	Syntax
Basic groupby	`df.groupby('col')`
Single agg	`.mean() / .sum() / .count()`
Multiple agg	`.agg(['mean', 'std'])`
Named agg	`.agg(name=('col', 'func'))`
Transform	`.transform('mean')`
Filter groups	`.filter(lambda x: ...)`
Multi-group	`df.groupby(['A', 'B'])`
Reset index	`.reset_index()`

Master groupby for powerful data aggregation