Lesson 10 Summary: Merging and Joining

Data Science with Python – Key Concepts

Data Science Program

Merging Overview

Combining DataFrames

Join Types:

`inner` | `left` | `right` | `outer`

Always specify merge keys to avoid unexpected results

Choose the right method based on your data structure

pd.concat()

Stack DataFrames vertically or horizontally:

- **Vertical:** `pd.concat([df1, df2], axis=0)`
- **Horizontal:** `pd.concat([df1, df2], axis=1)`
- **Ignore index:** `pd.concat([df1, df2], ignore_index=True)`

Finance use case:

```
# Combine monthly data files
all_data = pd.concat([jan, feb, mar])
```

concat() is fastest for stacking similar DataFrames

pd.merge() Basics

SQL-style joins on columns:

- **Basic:** `pd.merge(df1, df2, on='Symbol')`
- **Different names:** `merge(left_on='A', right_on='B')`
- **Multiple keys:** `merge(on=['Date', 'Symbol'])`

Example:

```
# Merge prices with company info
merged = pd.merge(prices, info, on='Symbol')
```

merge() is the most flexible joining method

Join Types

Four types of joins:

- **inner:** Only matching keys (default)
- **left:** All left keys + matching right
- **right:** All right keys + matching left
- **outer:** All keys from both DataFrames

Syntax:

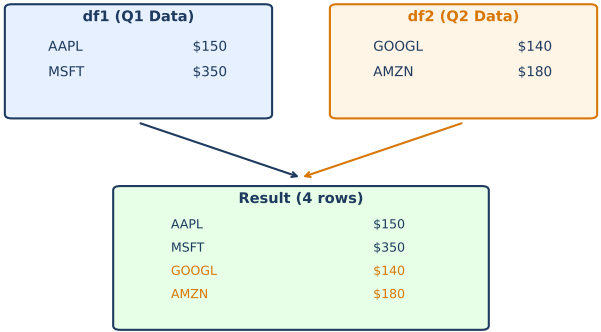
```
pd.merge(df1, df2, on='key', how='left')
```

Choose join type based on which rows you need to keep

Vertical Concatenation

pd.concat([df1, df2], axis=0)

Vertical Stacking (Row-wise)

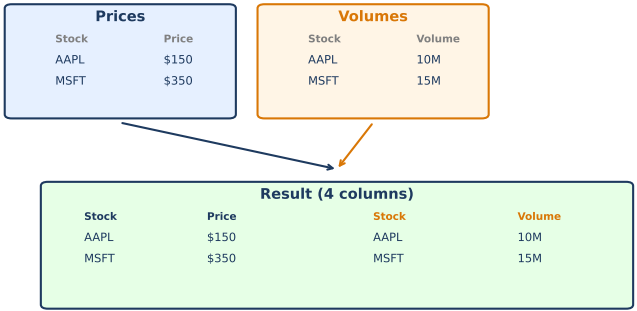


axis=0 stacks rows from multiple DataFrames

Horizontal Concatenation

`pd.concat([df1, df2], axis=1)`

Horizontal Stacking (Column-wise)



axis=1 adds columns side by side (aligns by index)

df.join() Method

Join on index (faster than merge):

- **Basic:** `df1.join(df2)`
- **With suffix:** `df1.join(df2, lsuffix='_left')`
- **Join type:** `df1.join(df2, how='outer')`

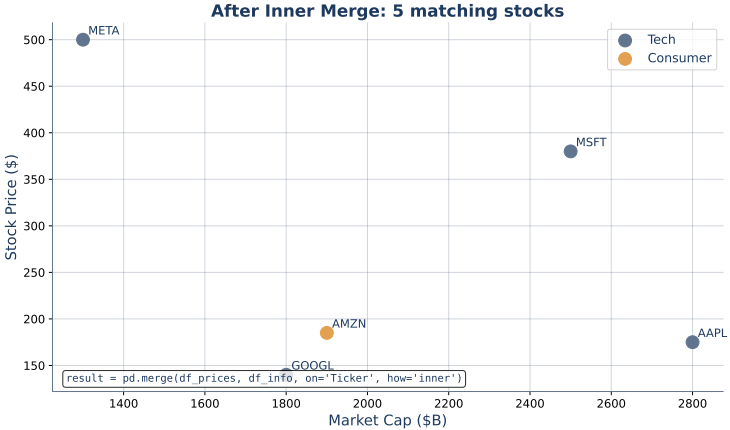
When to use:

- DataFrames share index (e.g., DatetimeIndex)
- Need speed over flexibility

---

join() uses index by default; merge() uses columns

Merge Example: Prices and Info



Merge enables analysis across multiple data sources

Quick Reference

Essential Merge Operations:

Operation	Syntax
Stack rows	`pd.concat([df1, df2], axis=0)`
Stack columns	`pd.concat([df1, df2], axis=1)`
Merge on column	`pd.merge(df1, df2, on='key')`
Left join	`pd.merge(..., how='left')`
Different keys	`merge(left_on='A', right_on='B')`
Index join	`df1.join(df2)`
Handle suffixes	`suffixes=('_x', '_y')`
Validate	`validate='one_to_one'`

Combining data is essential for real-world analysis