Lesson 08: Basic Operations

Data Science with Python – BSc Course

45 Minutes

Learning Objectives

After this lesson, you will be able to:

- Creating new columns
- apply() for transformations
- Arithmetic operations
- Sorting with sort$_v alues$()

- Calculating returns and moving averages

Finance application: Stock data processing and analysis

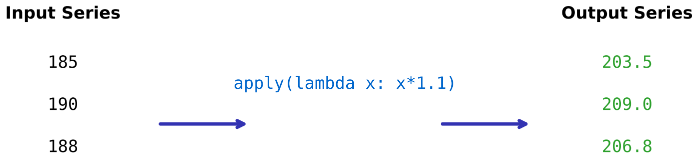
Column Creation

Creating New Columns

`df["Return"] = df["Close"].pct_change()`	Calculate returns
`df["MA20"] = df["Close"].rolling(20).mean()`	Moving average
`df["High_Low"] = df["High"] - df["Low"]`	Price range
`df["Signal"] = np.where(df["Return"]>0, 1, -1)`	Conditional

Apply Function

apply() Function



DataFrame Arithmetic

`df["A"] + df["B"]`	Element-wise addition
`df["A"] * 100`	Scalar multiplication
`df["A"] / df["B"]`	Division
`df.sum()`	Column sums
`df.mean(axis=1)`	Row means

Sorting DataFrames

```
df.sort_values("Price")
```

Sort by single column (ascending)

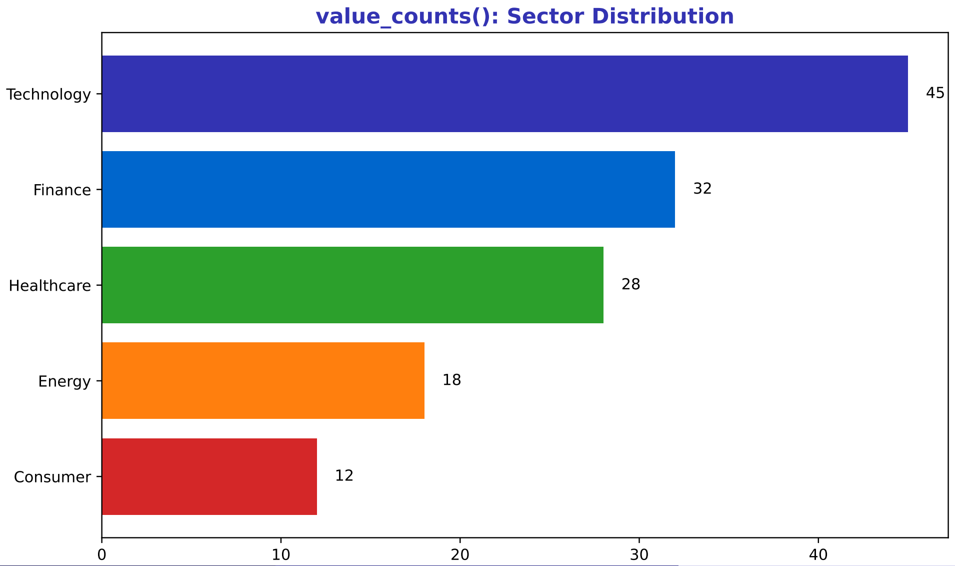
```
df.sort_values("Price", ascending=False)
```

Sort descending

```
df.sort_values(["Sector","Price"])
```

Sort by multiple columns

05 Value Counts



Calculating Returns

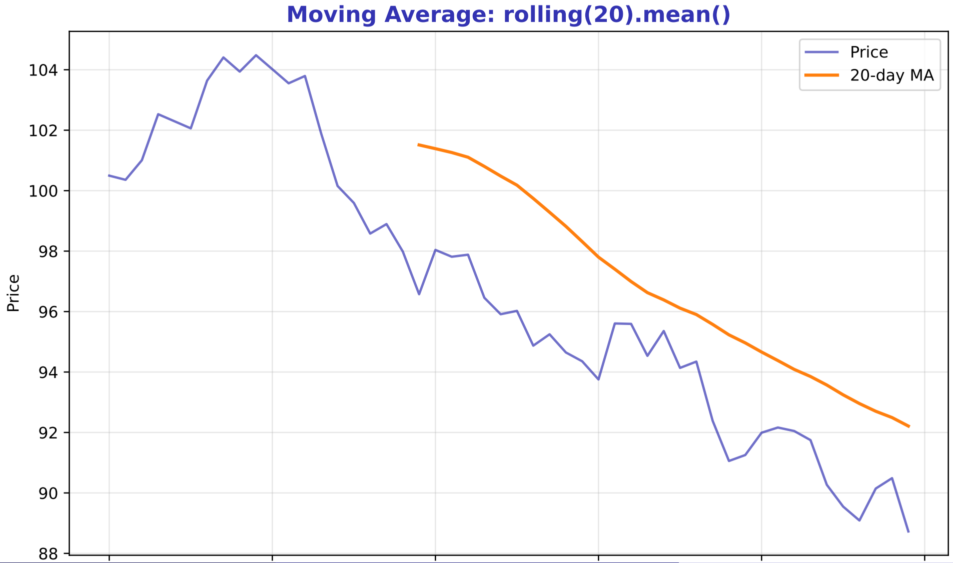
Simple Return: $r_t = \frac{P_t - P_{t-1}}{P_{t-1}}$

```
df["Return"] = df["Price"].pct_change()
```

Log Return: $r_t = \ln(P_t) - \ln(P_{t-1})$

```
df["LogRet"] = np.log(df["Price"]).diff()
```

07 Moving Average



08 Cheat Sheet

Operations Cheat Sheet

`pct_change()` - Returns	`diff()` - Differences
`cumsum()` - Cumulative sum	`cumprod()` - Cumulative product
`rolling(n)` - Rolling window	`shift(n)` - Lag values
`rank()` - Rankings	`clip(lower, upper)` - Bound values

Lesson Summary

Key Takeaways:

- Creating new columns
- apply() for transformations
- Arithmetic operations
- Sorting with sort$_v$*alues*()

- Calculating returns and moving averages

Practice: Apply these concepts to the stock price dataset.