

Pandas Syllabus

Introduction to Pandas

1. What is Pandas?
 - 1.1 Overview of the Pandas library
 - 1.2 Importance of Pandas in data analysis and machine learning
 - 1.3 Key components of Pandas: Series and DataFrame
2. Setting Up the Environment
 - 2.1 Installing Pandas using pip or conda
 - 2.2 Installing Anaconda and Jupyter Notebook
 - 2.3 Importing Pandas and understanding the pd alias
3. First Steps with Pandas
 - 3.1 Creating Series and DataFrames manually
 - 3.2 Loading data from various formats: CSV, Excel, JSON, HTML, SQL, Pickle
 - 3.3 Exploring datasets with head(), tail(), info(), and describe()

Pandas Basics

4. Data Structures
 - 4.1 Series: One-dimensional labeled arrays
 - 4.2 DataFrame: Two-dimensional labeled data structures
 - 4.3 Index objects: Labels for rows and columns
5. Basic DataFrame Operations
 - 5.1 Accessing rows and columns using loc and iloc
 - 5.2 Modifying data: Adding and removing rows or columns
 - 5.3 Transposing data using .T
6. Data Selection and Slicing
 - 6.1 Selecting rows and columns by labels or positions
 - 6.2 Conditional selection using Boolean indexing
7. Data Types in Pandas
 - 7.1 Identifying data types with dtypes
 - 7.2 Converting data types using astype()
 - 7.3 Handling categorical data

Data Cleaning and Preprocessing

8. Handling Missing Values
 - 8.1 Identifying missing data with isnull() and notnull()
 - 8.2 Dropping missing values using dropna()
 - 8.3 Filling missing values with fillna() (mean, median, or specific values)
9. Renaming Columns and Rows
 - 9.1 Using rename() for labels
 - 9.2 Renaming directly
10. Removing Duplicates
 - 10.1 Identifying duplicates with duplicated()
 - 10.2 Dropping duplicates using drop_duplicates()
11. Replacing Data
 - 11.1 Using replace() to substitute values

Data Transformation

12. Sorting Data
 - 12.1 Sorting by index using sort_index()
 - 12.2 Sorting by values using sort_values()

- 13. Merging, Joining, and Concatenation
 - 13.1 Combining datasets using merge() for SQL-like joins
 - 13.2 Using concat() for stacking datasets vertically or horizontally
 - 13.3 Using join() for index-based joining
- 14. GroupBy Operations
 - 14.1 Aggregation functions (sum, mean, count, etc.)
 - 14.2 Grouping data based on multiple columns
 - 14.3 Custom aggregation functions
- 15. Pivot Tables
 - 15.1 Creating pivot tables for multi-dimensional aggregation
 - 15.2 Difference between pivot and groupby
- 16. Melt and Stack
 - 16.1 Using melt() to convert wide to long format
 - 16.2 Transforming hierarchical indices using stack() and unstack()

Advanced Techniques

- 17. Apply and Map Functions
 - 17.1 Element-wise operations with apply(), map(), and applymap()
- 18. Crosstab
 - 18.1 Creating summary tables of categorical data using pd.crosstab()
- 19. Cut and Binning
 - 19.1 Creating intervals for numerical data using cut()
 - 19.2 Equal-width vs. custom-width bins
- 20. Datetime Operations
 - 20.1 Converting columns to datetime using to_datetime()
 - 20.2 Extracting date components (year, month, day, hour, etc.)
 - 20.3 Performing date arithmetic and filtering
- 21. Handling Large Datasets
 - 21.1 Optimizing memory usage
 - 21.2 Processing large datasets in chunks using iterators

Data Analysis and Visualization

- 22. Exploratory Data Analysis (EDA)
 - 22.1 Performing summary statistics
 - 22.2 Identifying trends and patterns in data
- 23. Data Visualization
 - 23.1 Plotting using Pandas: Line, bar, histogram, scatter, and box plots
 - 23.2 Integrating Pandas with Matplotlib and Seaborn
- 24. Boolean Indexing
 - 24.1 Advanced conditional filtering
 - 24.2 Combining multiple conditions using logical operators

Performance Optimization

- 25. Vectorized Operations
 - 25.1 Using NumPy under the hood for faster computations
- 26. Working with Sparse Data
 - 26.1 Reducing memory usage with sparse matrices
- 27. Profiling and Debugging
 - 27.1 Profiling performance with %timeit
 - 27.2 Debugging slow code

Real-World Applications

28. Case Studies

28.1 Analysing real-world datasets: Titanic dataset, financial data, sales data

29. Integration with Other Libraries

29.1 Using Pandas with Scikit-learn for machine learning preprocessing

29.2 Exporting cleaned data to formats like Excel and SQL

MATPLOTLIB AND SEABORN

Module 1: Introduction to Data Visualization

1.1 Overview of Data Visualization

- Importance of visualization in data analysis
- Types of data visualization (static vs dynamic)
- Key principles of effective data visualization
- Tools for data visualization in Python

1.2 Introduction to Matplotlib

- Overview of Matplotlib and its history
- Installing and setting up Matplotlib
- Anatomy of a Matplotlib plot (Figure, Axes, Axis)
- Basic syntax and structure of a Matplotlib plot

Module 2: Basic Plotting with Matplotlib

2.1 Basic Plotting Concepts

- Line plots
- Scatter plots
- Bar plots
- Histogram plots
- Pie charts

2.2 Matplotlib Components

- Figure and Axes
- Plot, Subplot, and Grid
- Axis, ticks, and labels
- Titles, labels, and legends

2.3 Customizing Plots

- Customizing colors, markers, and line styles
- Adjusting axis limits and scales (linear, logarithmic)
- Adding gridlines
- Annotating plots (text and arrows)

2.4 Advanced Plotting

- Creating subplots and multi-plot grids
- 3D plotting (3D scatter, surface plots)
- Saving and exporting plots (PNG, PDF, SVG)
- Practical Exercise:
- Create multiple plots (line, scatter, bar, histogram) with customization and annotation.

Module 3: Introduction to Seaborn

3.1 Overview of Seaborn

- Difference between Matplotlib and Seaborn
- Installing Seaborn

- Introduction to Seaborn's high-level interface
- Seaborn's built-in themes and color palettes

3.2 Seaborn Basics

- Creating simple plots with Seaborn (line, scatter, bar, etc.)
- Understanding Seaborn's syntax and default styling
- Visualizing categorical vs continuous data with Seaborn

Module 4: Advanced Data Visualizations with Seaborn

4.1 Seaborn's Advanced Plot Types

- Pair plots (pairwise relationships)
- Heatmaps (correlation matrix, clustered heatmap)
- Box plots, violin plots, and distribution plots
- Swarm plots and strip plots
- Facet grids (subplots based on categories)

4.2 Customization in Seaborn

- Customizing color palettes
- Customizing plot styles and themes
- Adjusting axes and legends in Seaborn plots
- Working with multiple categorical variables

4.3 Integration of Seaborn with Pandas

- Visualizing data directly from Pandas DataFrames
- Handling missing data in visualizations
- Aggregating data for Seaborn plots

Module 5: Comparing Matplotlib and Seaborn

5.1 When to Use Matplotlib vs Seaborn

- Strengths of Matplotlib
- Strengths of Seaborn
- Performance and flexibility considerations
- Combining Matplotlib and Seaborn for customized visualizations

5.2 Case Study: Visualization with Real Data

- Using both Matplotlib and Seaborn to visualize a real-world dataset (e.g., sales data, weather data, or a dataset from Kaggle)
- Visualizing trends, distributions, and relationships in the data
- Customizing the visualization for publication-quality plots

Module 6: Interactive Visualization (Optional)

6.1 Introduction to Interactive Visualization

- Importance of interactive visualizations
- Using Plotly for interactive plotting
- Integrating Matplotlib/Seaborn with Plotly

6.2 Basic Interactive Plots

- Creating interactive line, scatter, and bar plots
- Adding interactivity (zooming, hovering, etc.)
- Saving interactive plots as HTML