Python-Pandas

Dr. Sarwan Singh

pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$

Introduction to Pandas

What is Pandas?

- *A Python library for data analysis and manipulation.*
- *Built on top of NumPy.*

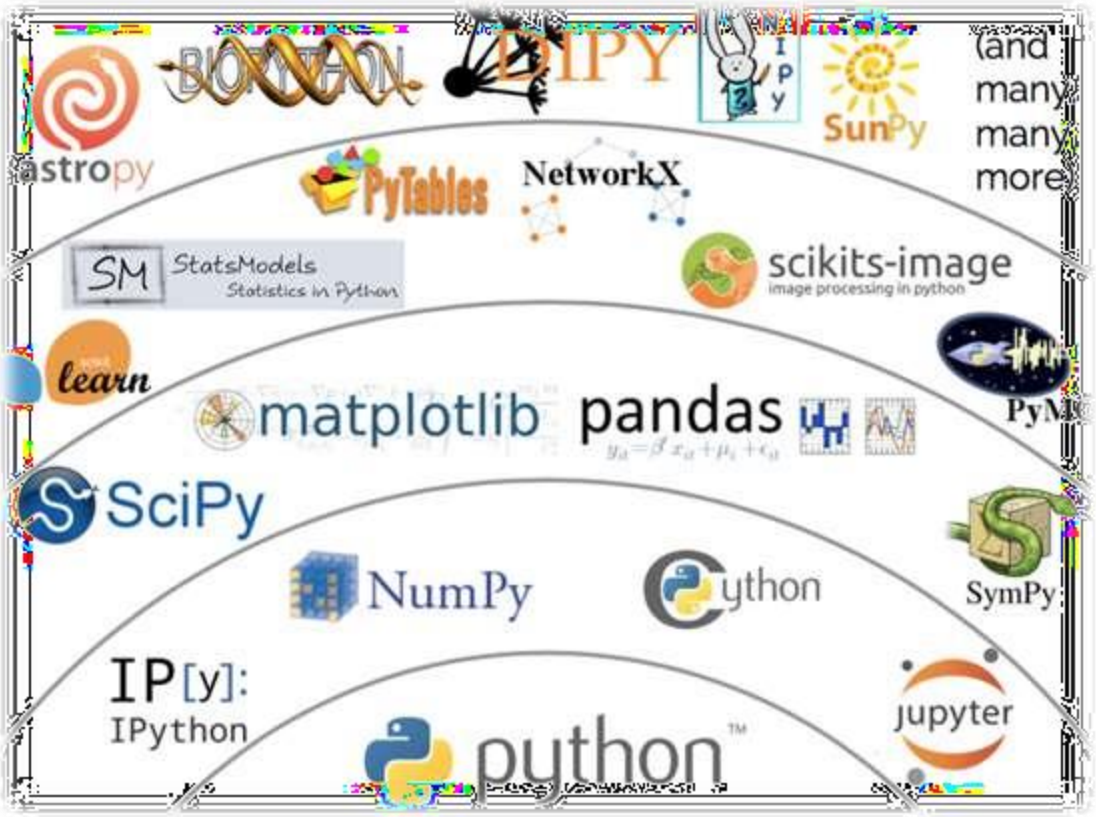
Why Use Pandas?

- Easy to handle missing data.
- Efficient for data wrangling.
- Supports operations for structured data.

Agenda

- Introduction – Panel Data System
- History and usage
- Series, DataFrame, Panel
- Basic Operations

One guiding principle of Python code is that
"explicit is better than implicit"



Introduction

- Pandas is Python package for data analysis.
- Pandas is an open-source Python Library built on top of Numpy
- Provides high-performance data manipulation and analysis tool using its powerful data structures.
- The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data.
- In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data. {at AQR capital Management LLC}
- 30,000 lines of tested Python/Cython code

Introduction

```
import pandas
pandas.__version__
```

```
'0.20.3'
```

- Pandas can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data —

  load → prepare → manipulate → model → analyze.

- Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.
- Pandas is easy to use and powerful, but

  "*with great power comes great responsibility*"

eases data munging

Numpy vs Pandas

- Pandas is designed for working with tabular or heterogeneous data.
- NumPy, is best suited for working with homogeneous numerical array data
- **Numpy** is required by **pandas** (and by virtually all numerical tools for Python)
- numpy consumes (roughtly 1/3) less memory compared to pandas
- numpy generally performs better than pandas for 50K rows or less
- pandas generally performs better than numpy for 500K rows or more
- for 50K to 500K rows, it is a toss up between pandas and numpy depending on the kind of operation

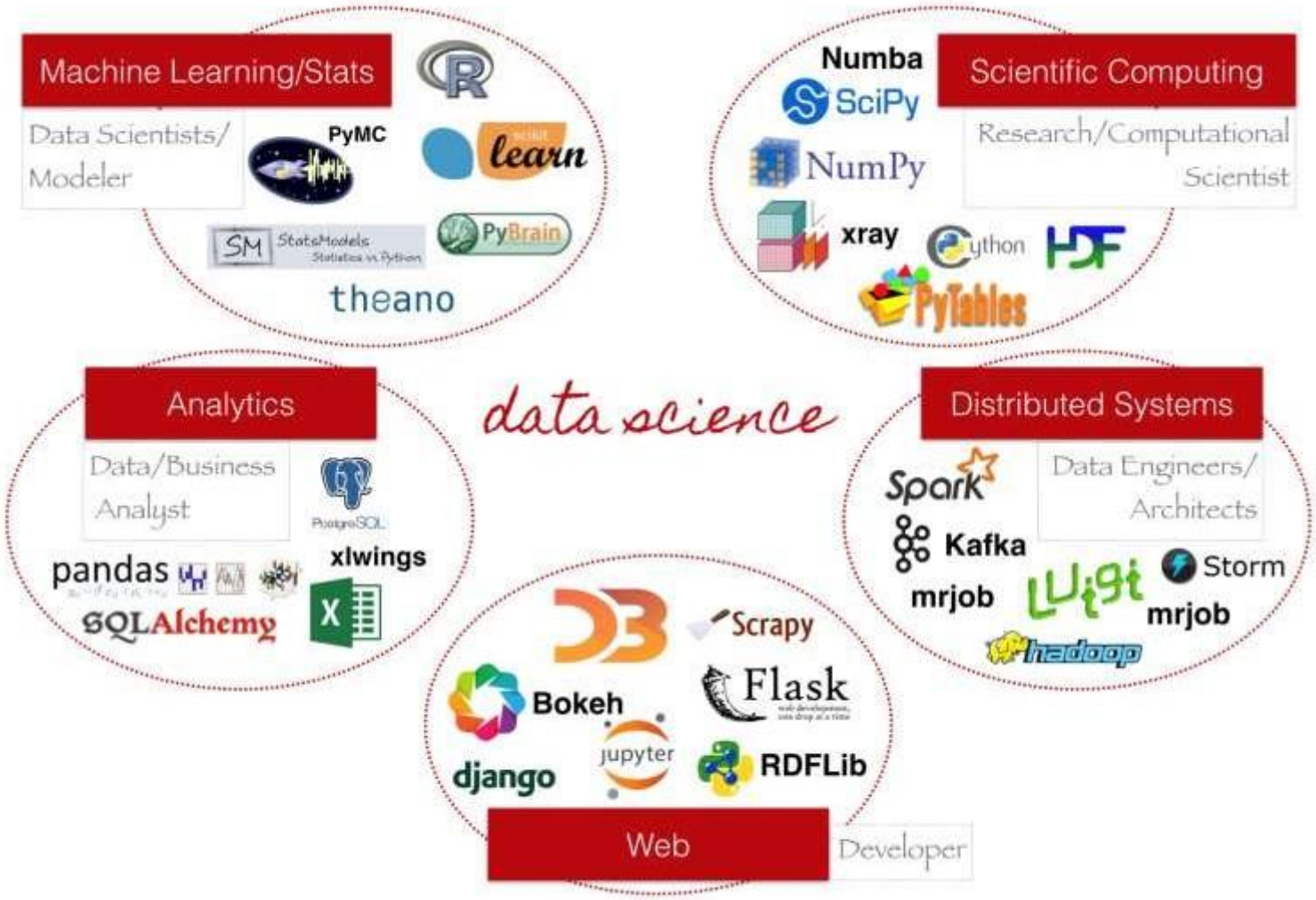
✓Pandas became an open source project in 2010

✓Now has 800 distinct contributors in developer community

Source:gouthamanbalaraman.com

Why pandas

- Big part of Data Science is Data Cleaning.
- Pandas is a power tool for data cleaning



Key Features of Pandas

- Fast and efficient DataFrame object with default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of date sets.
- Label-based slicing, indexing and subsetting of large data sets.
- Columns from a data structure can be deleted or inserted.
- Group by data for aggregation and transformations.
- High performance merging and joining of data.
- Time Series functionality.

Source : tutorialpoint.com

Pandas and NumPy

- Pandas and NumPy both hold data
- Pandas has column names as well
- Makes it easier to manipulate data

Pandas NumPy Scikit-Learn workflow

- Start with CSV
- Convert to Pandas
- Slice and dice in Pandas
- Convert to NumPy array to feed to scikit-learn
- NumPy is faster than Pandas
- Both are faster than normal Python arrays

Data Structures

- Pandas deals with the following three data structures –

  Series , DataFrame , Panel

- These data structures are built on top of NumPy array, which means they are fast.
- DataFrame is a container of Series, Panel is a container of DataFrame.

Data Structure	Dimensions	Description
Series	1	1D labeled homogeneous array, size immutable.
Data Frames	2	General 2D labeled, size-mutable tabular structure with potentially heterogeneously typed columns.
Panel	3	General 3D labeled, size-mutable array.

Installing and Importing Pandas

- **Installation**

```
pip install pandas
```

- **Importing Pandas**

```
import pandas as pd
```

Pandas Data Structures

Series (1D data)

- A one-dimensional labeled array.

DataFrame (2D data)

- A two-dimensional, size-mutable, tabular data structure.

Creating a Series

- **From a List**

```
import pandas as pd

data = [10, 20, 30, 40]
s = pd.Series(data)
print(s)
```

- **Custom Index**

```
s = pd.Series(data, index=['a', 'b', 'c', 'd'])
print(s)
```

Creating a DataFrame

- **From a Dictionary**

```
import pandas as pd
data = {
'Name': ['Alice', 'Bob', 'Charlie'],
'Age': [25, 30, 35],
'City': ['New York', 'Los Angeles', 'Chicago']
}
df = pd.DataFrame(data)
print(df)
```

Reading and Writing Data

- **Reading a CSV File**

```
df = pd.read_csv('data.csv')
print(df)
```

- **Writing to a CSV File**

```
df.to_csv('output.csv', index=False)
```

Viewing Data

- **Viewing Rows and Columns**

```
print(df.head()) # First 5 rows
print(df.tail(3)) # Last 3 rows
```

- **DataFrame Info**

```
print(df.describe())
print(df.info())
```

Selecting Data

- **Selecting Columns**

```
print(df['Name'])
print(df[['Name', 'Age']])
```

- **Selecting Rows**

```
print(df.iloc[0]) # By index
print(df.loc[0]) # By label
print(df.iloc[1:3]) # Slice rows
```

Data Manipulation

- **Adding a Column**

```
df['Salary'] = [50000, 60000, 70000]
```

- **Filtering Rows**

```
filtered_df = df[df['Age'] > 30]
```

- **Dropping Columns or Rows**

```
df = df.drop('Salary', axis=1) # Drop column
df = df.drop(0, axis=0) # Drop row
```

Aggregation and Grouping

- **Aggregations**

```
print(df['Age'].mean())
print(df['Age'].sum())
```

- **Grouping**

```
grouped = df.groupby('City')
print(grouped['Age'].mean())
```

Handling Missing Data

- **Filling Missing Values**

```
df['Age'] = df['Age'].fillna(0)
```

- **Dropping Missing Values**

```
df = df.dropna()
```

Advanced Features

- **Merging DataFrames**

```
df1 = pd.DataFrame({'ID': [1, 2], 'Name': ['Alice', 'Bob']})
df2 = pd.DataFrame({'ID': [1, 2], 'Age': [25, 30]})
merged = pd.merge(df1, df2, on='ID’)
```

- **Pivot Tables**

```
pivot = df.pivot_table(values='Age', index='City', aggfunc='mean')
```

Practice Exercises on Pandas

Demo on Collab

Additional Resources

- Pandas Documentation: https://pandas.pydata.org/docs/
- Practice Datasets: Kaggle Datasets
- Tutorials: Pandas Tutorials on w3schools, RealPython

https://www.w3schools.com/python/pandas/pandas_getting_started.asp