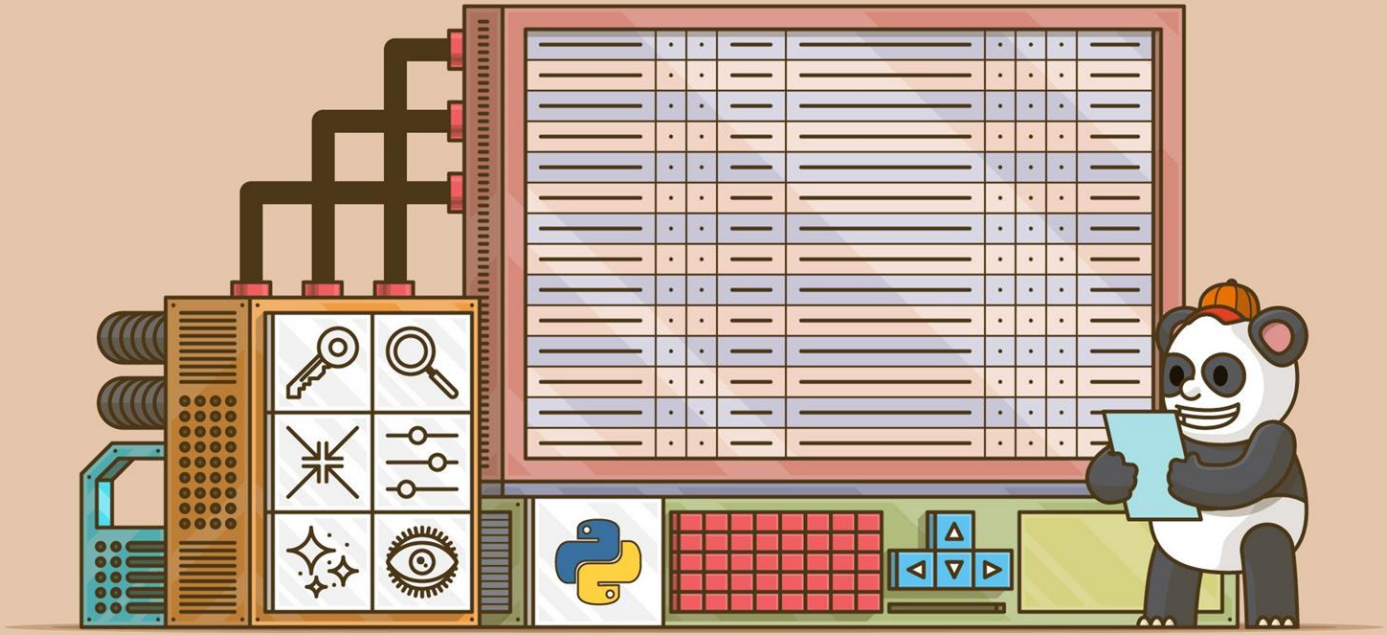
APSSDC

Andhra Pradesh State Skill Development Corporation



Data Analysis using Pandas

Lavanya P

Content

- Introduction to Pandas
- Advantages of Pandas
- Pandas series and Indexing
- DataFrames
- Merging
- File I/O
- Grouping
- Sorting
- Statistical
- Plotting

Skill AP
APSSDC

pandas

Pandas

- Pandas is an open-source python library providing efficient easy-to-use data structures and analysis tools
- Derived from "*PANel Data – an Econometrics from Multidimensional data*"
- It is an excellent tool for processing and analyzing real world data
- There are two main data structures in Pandas - Series and Dataframes
- **Series**:
  - One-dimensional ndarray with axis labels (including time series).
- **DataFrame:**
  - Two-dimensional, size-mutable, potentially heterogeneous tabular data

Advantages of Pandas

The following are some of the advantages of pandas:

- **Less intuition:** Many methods, such as joining, selecting, and loading files, are used without much intuition and without taking away much of the powerful nature of pandas.
- **High level of abstraction:** Pandas have a higher abstraction level than NumPy, which gives it a simpler interface for users to interact.
- **Faster processing:** The internal representation of DataFrames allows faster processing for some operations. of course, this always depends on the data and its structure.
- **Easy DataFrame design:** DataFrames are designed for operations with and on large datasets.

Installation and Importing Pandas

- Working with conda?

  conda install pandas

- Prefer pip?

  pip install pandas

- Importing library

  Import pandas as pd

Pandas Series and Indexing

- **A series is similar to a 1-D numpy array, and contains scalar values of the same type (numeric, character, datetime etc.).**
- A dataframe is simply a table where each column is a “Pandas Series”
- Creating Series object by using List,tuple,dict and also numpy array.
- create pandas series from array-like objects using **pd.Series()**

Note : the Number of elements in the Index list is always equal to the number of elements in the specified series.

Pandas DataFrame

- Dataframe is the most widely used data-structure in data analysis
- DataFrame is a table with rows and columns, with rows having an index and columns having meaningful names.
- Usually, dataframes are imported as CSV files, but sometimes it is more convenient to convert dictionaries into dataframes

DataFrame Indexing

- An important concept in pandas dataframes is that of **row indices**. By default, each row is assigned indices starting from 0, and are represented at the left side of the dataframe.
  - By using set_index() Method
- Selecting rows
  - df[start_index:end_index]
- Selecting columns
  - df[‘column_name’] or df.column_name
- Selecting subset of dataframe
  - df[[‘column_name1’,’column_name2’..]]

- **iloc** -- for accessing rows using integer indicies
- **loc** -- for accessing rows other than integer indicies

Reading and Writing Data

- There are various ways of creating dataframes, such as creating them from dictionaries, JSON objects, reading from txt, CSV files, etc.
  - pd.DataFrame()
  - pd.read_csv(),pd.read_excel()
- We have methods in pandas for exporting cleaned data into files
  - pd.to_csv()
  - pd.to_excel()

Basic Functionalities of a Data Object

- **df.head()** – to get the first n rows from the dataframe.
- **df.tail()** – to get the last n rows from the dataframe.
- **df.columns** - to get columns names of the dataframe
- **df.index** – to get index values of the dataframe.
- **df.describe()** - returns the descriptive statistics summary
- **df.sum()** – return the sum of each columns
- **df.count()** – return the count the values in the columns
- **df.max()** – returns the maximum value in the dataframe
- **df.idxmax()** – returns the maximum value index from the dataframe

Merging of Data Objects

- Merging is one of the most common operations you will do, since data often comes in various files.
- To combine the information of two dataframes into a single DataFrame
  - Merge multiple dataframes using common columns/keys using pd.merge()
- Concatenation is used when you have dataframes having the same columns and want to append them, or having the same rows and want to append them side-by-side.
  - Concatenate dataframes using pd.concat()
- Append dataframes
  - df1.append(df2)

Grouping and Summarising Dataframes

- Grouping and aggregation are some of the most frequently used operations in data analysis
- **groupby()** function is used to split the data into groups based on some criteria.
- After grouped then getting first row in every group by using df.first(),df.last()
- Grouping analysis can be thought of as having three parts:

1. **Splitting** the data into groups
2. **Applying** a function to each group
3. **Combining** the results into a data structure showing the summary statistics

Sorting

- sort dataframes in two ways
  - by the indices
    - axis = 0 indicates that you want to sort rows (use axis=1 for columns)
    - sort_index(axis = value)
  - by the values.
    - df.sort_values(by= ‘column_name’)
    - Sorting by more than two columns
      - df.sort_values(by= [‘column_name1’,’column_name2’],inplace=True)

Statistical

- Use pandas to obtain statistical metrics for data.
- We have different methods for Calculating aggregations / mathematical operations, for numerical data
  - **df.mean()** - return the mean of the columns
  - **df['column_name'].median() -** return median of particular column
  - **df.sum() -** return sum of the column
  - **df.describe() -** Summary of statistics
  - df.std() - standard deviation

Plotting

- After all the processing and manipulation of the data, the most important step that comes is visualization.
- On a DataFrame, the **plot()** method is a convenience to plot all of the columns with labels
  - For entire dataframe - df.plot()
  - For a particular column df[‘column_name’].plot()

Skill AP
APSSDC

pandas