Pandas

Muhammad Samir Assawalhy

2023-04-15

1 Task 11 - Pandas

Pandas is a Python library used for data manipulation and analysis. It provides data structures and functions necessary for working with structured data seamlessly.

```
import pandas as pd # common way to import pandas
```

2 pd.Series

```
s = pd.Series(data=(...), index=(...))
# you can access multiple items using there indices at once
print(s[[index_1, index_2]])
```

Methods and properties

- `shape`, `size`, `ndim`
- `index`: a list which represent the indexes of the series
- `loc`: labeled index
- `iloc`: numerical index
- `drop`: remove element from the series, but return the modified series, modification doesn't happen in-place unless use set the kwarg `inpalce=True`.
- `dropna`, `drop_duplicates`, `unique`
- `apply`: takes a lambda function to manipulate the values of the series, it is similar to `map` function in python
- `idxmax`, `idxmin`: index of the max or min value

3 pd.DataFrame

```
df = pd.DataFrame(items, index=[...], columns=[...])
df = pd.read_csv("./data.csv"), pd.read_table(".tsv")
df[column], df.loc[row] # instance of pd.Series
(df.loc[:, 'id'] == df['id']).all() # True
```

```
df.loc[row1:row2, col1:col2] # slice your data frame
df[column][row] # column first to access a value in the data frame
df[new_column] = ... # list of values or another column
df.isnull().sum().sum() # get the number of nan values
# imagine a data in which we want to get the sum of all salary spent in every year
df.groupby(["Year"])["Salary"].sum()
```

- `shape`, `size`, `ndim`
- `values`: return a n-dimensional list with the same shape as the data frame
- `loc`: to access rows in data frames
- `append(DataFrame)`: to add new row
- `insert` to insert a new column in a specific position
- `pop`: to remove columns
- `drop`: to remove both columns and rows depending on `axis` argument
- `rename`: to rename columns or rows (index), takes dictionary of old name as key and new name as value
- `isnull`: returns a data frame of boolean type, which indicate if some value are `NaN` for example
- `count`: return the count of non-NaN values
- `dropna`, `drop_duplicates`
- `sort_values`, `query`, `groupby`
- `fillna`: to replace any NaN with a specific value
- `sum`, `mean`, `min`, `max`, `std`, `corr`, `describe`
- `head`, `tail`
- `all`, `any`: if all are true or any is true