Pandas 进阶

点击一键运行代码→ 获取更多速查表→ 使用说明书 [视频 & 图文]→ bilibili ShowMeAI

1. 数据重塑

透视

```
> import pandas as pd
> df2 = pd.DataFrame({'Date':['2021-12-25', '2021-12-26', '2021-12-25', '2021-12-27', '2021-12-26', '2021-12-27'],
                      'Type':['a', 'b', 'c', 'a', 'a', 'c'],
                      'Value':[1.34, 10.2, 20.43, 50.31, 0.26, 20.64]
                     })
> df3= df2.pivot(index='Date', columns='Type', values='Value') # 将行变为列
```

透视表

```
> df4 = pd.pivot_table(df2, values='Value', index='Date', columns='Type'])    # 将行变为列
```

堆叠 (轴旋转)

```
> stacked = df2.stack() # 透视列标签
> stacked.unstack() # 透视索引标签
```

ShowMeAI

融合 / Melt

```
> pd.melt(df2, id_vars=["Date"], value_vars=["Type", "Value"], value_name="Observations") # 将列转为行
```

2. 迭代

迭代遍历数据帧

```
> df2.iteritems() #( 列索引 , 序列 ) 键值对
> df2.iterrows()  #( 行索引 , 序列 ) 键值对
```

4. 数据滤重

数据帧自带一系列函数对数据重复值进行处理

```
# 返回唯一值
> s3.unique()
# 查找重复值
> df2.duplicated('Type')
# 去除重复值
> df2.drop_duplicates('Type', keep='last')
# 查找重复索引
> df.index.duplicated()
```

5. 数据分组

分组聚合

```
> df2.groupby(by=['Date', 'Type']).mean() # 分组求均值
> df4.groupby(level=0).sum()
> df4.groupby(level=0).agg({'a':lambda x:sum(x)/len(x), 'b': np.sum})
```

转换

```
> customSum = lambda x: (x+x%2)
> df4.groupby(level=0).transform(customSum)
```

6. 缺失值

```
> df.dropna()  # 去除缺失值 NaN
> df3.fillna(df3.mean()) # 用预设值填充缺失值 NaN
> df2.replace("a", "f") # 用一个值替换另一个值
```

7. 合并数据

合并 -Merge

```
> data1 = pd.DataFrame({'key': ['K0', 'K1', 'K2', 'K3'],
                        'A': ['A0', 'A1', 'A2', 'A3'],
                        'B': ['B0', 'B1', 'B2', 'B3']})
> data2 = pd.DataFrame({'key': ['K0', 'K1', 'K3', 'K4'],
                        'C': ['C0', 'C1', 'C2', 'C3'],
                        'D': ['D0', 'D1', 'D2', 'D3']})
> pd.merge(data1, data2, how='left', on='key')
> pd.merge(data1, data2, how='right', on='key')
> pd.merge(data1, data2, how='inner', on='key')
> pd.merge(data1, data2, how='outer', on='key')
```

连接 -Join

```
> data1.join(data2, how='right', lsuffix='_1', rsuffix='_2')
```

拼接 -Concatenate

纵向

```
> s.append(s2)
```

横向 / 纵向

```
> pd.concat([s, s2], axis=1, keys=['One', 'Two'])
> pd.concat([data1, data2], axis=1, join='inner')
```

Pandas 是一个构建于 Numpy 之上的 Python 库，它提供了高性能的数据操作。

Pandas 提供了大量能使我们快速便捷地处理数据的函数和方法，经常在数据科学中用于数据处理、数据变换、特征工程、数据探索分析与呈现等过程中。

作者 | 韩信子 @ShowMeAI
设计 | 南 乔 @ShowMeAI
参考 | datacamp cheatsheet

3. 高级索引

基础选择

```
> df3.loc[:, (df3>1).any()] # 选择任一值大于 1 的列
> df3.loc[:, (df3>1).all()] # 选择所有值大于 1 的列
> df3.loc[:, df3.isnull().any()] # 选择含 NaN 值的列
> df3.loc[:, df3.notnull().all()] # 选择不含 NaN 值的列
```

通过 isin 选择

```
> df2[(df2.Type.isin(['b', 'c']))] # 选择指定列为某一类型的数值
> df3.filter(items=['a', 'b']) # 选择特定值
```

通过 where 选择

```
> s.where(s > 0) # 选择子集
```

通过 query 选择

```
> df2.query('Value > 10') # 查询 DataFrame
```

设置 / 取消索引

```
> df.set_index('Country') # 设置索引
> df4 = df.reset_index() # 重置索引 0-n
# 重命名 DataFrame 列名
> df = df.rename(index=str,
                 columns={"Country":"cntry", "Capital":"cptl", "Population":"ppltn"})
```

重设索引

```
> s2 = s.reindex(['a', 'c', 'd', 'e', 'b'])
```

前向填充

```
> df.reindex(range(4), method='ffill')
```

后向填充

```
> s3 = s.reindex(range(5), method='bfill')
```

多重索引

```
> arrays = [np.array([1, 2, 3]), np.array([5, 4, 3])]
> df5 = pd.DataFrame(np.random.rand(3, 2), index=arrays)
> tuples = list(zip(*arrays))
> index = pd.MultiIndex.from_tuples(tuples, names=['first', 'second'])
> df6 = pd.DataFrame(np.random.rand(3, 2), index=index)
> df2.set_index(["Date", "Type"])
```

8. 日期转换

pandas 包含对时间型数据变换与处理的函数

```
> df2['Date']= pd.to_datetime(df2['Date'])
> df2['Date']= pd.date_range('2021-12-25', periods=6, freq='M')

> import datetime
  dates = [datetime.date(2021,12,25), datetime.date(2021,12,26)]

  index = pd.DatetimeIndex(dates)
  index = pd.date_range(datetime.date(2021,12,25),
                        end=datetime.date(2022,12,26),
                        freq='BM')
```

9. 可视化

Series 和 Dataframe 都自带 plot 绘图功能

```
> import matplotlib.pyplot as plt

  s.plot()
  plt.show()

  df2.plot()
  plt.show()
```

	Country	Capital	Population
0	Belgium	Brussels	11190846
1	India	New Delhi	1303171035
2	Brazil	Brasília	207847528
3	Brazil	Brasília	207847528

0	3
1	3
2	3
3	3
4	3