关于本节中要掌握的内容

- pandas的基本数据类型
- DataFrame的合并

pandas的基本数据类型

在上节课中我们学习了np.array，它可以理解为一种功能丰富的list。今天我们将学到pandas，它可以理解为功能丰富的dict，在处理复杂数据/图表数据时，非常好用。
首先，pandas中我们需要学习的数据类型包含两种:

- Series

Series可以理解为是一维的dict，它包含index和values

- DataFrame
  DataFrame可以理解为是二维的dict，它包含index，column，values

```
import pandas as pd
import numpy as np
s = pd.Series([3, 5, 7, 9])
print(s)
arr = np.zeros((3, ), dtype=[('A', 'i8'), ('B', 'f8'), ('C', 'a8')])
arr[:] = [(1, 1.0, 'a'), (2, 2.0, 'b'), (3, 3.0, 'c')]
print(pd.DataFrame(arr))
```

- 创建Series对象

  pd.Series(data,index=index)

  我们可以使用list、array、dict来初始化Series对象，并且可以手动指定index，让我们来看看

```
s1 = pd.Series([3, 5, 7, 9],index=['a','b','c','d'])
s2 = pd.Series(np.random.randn(3),index=['x1,','x2','x3'])
s3 = pd.Series({'d1': 1, 'd2': 2, 'd3': 3})
s4 = pd.Series({'x1': 1, 'x2': 2, 'x3': 3},index=['x0', 'x2', 'x3', 'x4'])
print(s4.index,s4.values)
```

- 访问Series对象

对于Series对象，我们可以像使用np.array一样去使用它，而不同的地方仅在于索引，让我们来看一下

```
print(s4['x2'])
print(s4[2])
print(s4[1:3])
print(s4[s4<3])
print(s4[[1,2]])
```

可以看到，我们访问Series对象，既可以通过显示索引(x2)，也可以通过隐式索引(2)来访问，为了避免混淆，我们处理panda索引时，对于

- 显示索引，使用loc属性来访问
- 隐式索引，使用iloc属性来访问

```
print(s4.loc['x2'])
print(s4.loc[2])
```

- 创建DataFrame

  pd.DataFrame(data,index=index,columns=columns)

  我们可以使用list、array、dict来初始化DataFrame对象，并且可以手动指定index和columns，让我们来看看

```
arr = np.zeros((3, ), dtype=[('A', 'i8'), ('B', 'f8'), ('C', 'a8')])
arr[:] = [(1, 1.0, 'a'), (2, 2.0, 'b'), (3, 3.0, 'c')]
data = pd.DataFrame(arr,index=['row1', 'row2', 'row3'],columns=['C', 'A', 'B'])
print(data)
data2 = {'s1': pd.Series([1, 2, 3], index=['r1', 'r2', 'r3']),
's2': pd.Series([4, 5, 6, 7], index=['r1', 'r2', 'r3', 'r4'])}
print(s2)
data3 = pd.DataFrame({'A': [0, 1, 2, 3],'B': [9, 8, 7, 6]},
index=['r1', 'r2', 'r3', 'r4'], columns=['A', 'C'])
print(data3.index,data3.values,data3.columns)
```

- 访问DataFrame索引

```
goods_quantity = {'苹果': 10,'香蕉':20,'橘子':15,'西瓜':50}
goods_price = {'苹果': 3.75,'香蕉':4.5,'橘子':1.6,'西瓜':8.0}

df = pd.DataFrame({'quantity': goods_quantity,'price':goods_price})
print(df['quantity'])
df['sum_price'] = df['quantity'] * df['price']
print(df.T)
print(df.iloc[1:3,:])
print(df.loc[df.price>2,['quantity','price']])
df.loc['苹果','price'] = 4
print(df)
```

练习题

1.创建一个包含学生信息（学号、姓名、年龄）的 Pandas DataFrame，并通过学号进行索引。

2.创建一个包含销售数据的 Pandas Series，其中索引是产品名称，值是销售数量。

3.创建一个包含两个 DataFrame 的字典，其中一个 DataFrame 包含学生信息，另一个包含考试成绩。使用学号作为键合并这两个 DataFrame。

4.创建一个包含日期范围的 Pandas DataFrame，索引为日期，包含两列数据：温度和湿度。

5.创建一个包含多个 Pandas Series 的数据框，每个 Series 表示一个城市的温度。使用城市名称作为列的列标签。

可能的答案

```
import pandas as pd
import numpy as np

# 1. 创建学生信息的 DataFrame，并通过学号索引
data1 = {'学号': [1, 2, 3],
         '姓名': ['张三', '李四', '王五'],
         '年龄': [20, 21, 22]}
df_students = pd.DataFrame(data1).set_index('学号')
print(df_students)

# 2. 创建销售数据的 Series
data2 = {'产品A': 100, '产品B': 150, '产品C': 200}
sales_series = pd.Series(data2, name='销售数量')
print(sales_series)

# 3. 创建包含学生信息和考试成绩的 DataFrame 字典，通过学号键合并
data3 = {'学生信息': df_students, '考试成绩': pd.DataFrame({'学号': [1, 2, 3], '成绩': [90, 85, 88]}).set_index('学号')}
df_merged = pd.concat(data3.values(), axis=1)
print(df_merged)

# 4. 创建日期范围的 DataFrame，包含温度和湿度数据
date_rng = pd.date_range(start='2023-01-01', end='2023-01-05', freq='D')
data4 = {'温度': np.random.randint(20, 30, size=len(date_rng)),
         '湿度': np.random.randint(40, 60, size=len(date_rng))}
df_weather = pd.DataFrame(data4, index=date_rng)
print(df_weather)

# 5. 创建包含多个城市温度的 DataFrame
data5 = {'城市A': pd.Series([25, 28, 23], name='温度'),
         '城市B': pd.Series([30, 32, 29], name='温度')}
df_cities = pd.DataFrame(data5)
print(df_cities)
```

DataFrame的合并

上面我们学到了DataFrame的创建、索引、修改，现在我们来看一下如何对两个DataFrame合并

- 合并两个DataFrame

```
import pandas as pd
df1 = pd.DataFrame({'ID': [1, 2, 3],
                    'Name': ['Alice', 'Bob', 'Charlie']})

df2 = pd.DataFrame({'ID': [2, 3, 4],
                    'Age': [25, 30, 35]})
concatenated_df = pd.concat([df1, df2], ignore_index=True)
print("Concatenated DataFrame:")
print(concatenated_df)
```

- 这里的ignore_index意味着忽略索引值
- 还可以使用axis='columns'，来进行列合并
- 还使用join='inner'，来进行内联合并

- 合并两个DataFrame（内连接）

```
import pandas as pd

df1 = pd.DataFrame({'ID': [1, 2, 3],
                    'Name': ['Alice', 'Bob', 'Charlie']})
df2 = pd.DataFrame({'ID': [2, 3, 4],
                    'Age': [25, 30, 35]})
merged_df = pd.merge(df1, df2, on='ID', how='inner')
print("Merged DataFrame:")
print(merged_df)
```

- how指明了连接方式，包括：'inner','outer','left','right'
- on指明了外键

练习题

1.创建两个包含学生信息的 DataFrame（包括学号、姓名），然后按照学号合并它们。

2.创建两个包含订单信息的 DataFrame（订单号、产品、数量），然后按照订单号合并它们。

3.创建两个包含销售数据的 DataFrame（日期、销售额），然后按照日期合并它们。

4.创建两个包含公司员工信息的 DataFrame（员工号、姓名、部门），然后按照员工号合并它们。

5.创建两个包含城市天气信息的 DataFrame（城市、温度、湿度），然后按照城市合并它们。

可能的答案

```
import pandas as pd

# 1. 合并学生信息
df_students1 = pd.DataFrame({'学号': [1, 2, 3], '姓名': ['张三', '李四', '王五']})
df_students2 = pd.DataFrame({'学号': [2, 3, 4], '姓名': ['Tom', 'Jerry', 'Mickey']})
merged_students = pd.merge(df_students1, df_students2, on='学号', how='inner')
print("Merged Students:")
print(merged_students)

# 2. 合并订单信息
df_orders1 = pd.DataFrame({'订单号': [101, 102, 103], '产品': ['A', 'B', 'C'], '数量': [5, 10, 8]})
df_orders2 = pd.DataFrame({'订单号': [102, 103, 104], '产品': ['B', 'C', 'D'], '数量': [15, 7, 12]})
merged_orders = pd.merge(df_orders1, df_orders2, on='订单号', how='inner')
print("\nMerged Orders:")
print(merged_orders)

# 3. 合并销售数据
df_sales1 = pd.DataFrame({'日期': ['2023-01-01', '2023-01-02', '2023-01-03'], '销售额': [1000, 1500, 800]})
df_sales2 = pd.DataFrame({'日期': ['2023-01-02', '2023-01-03', '2023-01-04'], '销售额': [1200, 900, 1100]})
merged_sales = pd.merge(df_sales1, df_sales2, on='日期', how='inner')
print("\nMerged Sales:")
print(merged_sales)

# 4. 合并员工信息
df_employees1 = pd.DataFrame({'员工号': [101, 102, 103], '姓名': ['John', 'Alice', 'Bob'], '部门': ['HR', 'Finance', 'IT']})
df_employees2 = pd.DataFrame({'员工号': [102, 103, 104], '姓名': ['Eva', 'Charlie', 'David'], '部门': ['Finance', 'IT', 'Marketing']})
merged_employees = pd.merge(df_employees1, df_employees2, on='员工号', how='inner')
print("\nMerged Employees:")
print(merged_employees)

# 5. 合并城市天气信息
df_weather1 = pd.DataFrame({'城市': ['A', 'B', 'C'], '温度': [25, 30, 28], '湿度': [60, 50, 55]})
df_weather2 = pd.DataFrame({'城市': ['B', 'C', 'D'], '温度': [32, 29, 27], '湿度': [45, 58, 50]})
merged_weather = pd.merge(df_weather1, df_weather2, on='城市', how='inner')
print("\nMerged Weather:")
print(merged_weather)
```