Python数据分析与可视化

——Pandas与matplotlib入门

1. 数据分析基础

1.1 数据分析的基本流程

数据分析通常包括以下几个步骤：

- 数据收集：从各种渠道获取原始数据
- 数据清洗：对收集到的数据进行整理、去重、缺失值处理等，使其适合分析
- 数据探索：通过统计分析、可视化等手段，对数据进行初步的分析和理解
- 数据建模：根据业务需求，使用合适的数据模型进行建模和预测
- 结果评估：评估模型的性能，调整模型参数，优化模型效果
- 结果呈现：将数据分析和建模的结果以图表、报告等形式呈现给相关人员

1.2 数据分析中常用的Python库

在Python的数据分析领域，有许多成熟的库可供选择，以下是其中最常用的一些：

- NumPy：提供多维数组对象及其相关操作，是许多数据分析库的基础
- Pandas：强大的数据处理和分析库，提供了方便的数据结构和数据操作方法
- matplotlib：数据可视化库，支持多种图形绘制和自定义
- seaborn：基于matplotlib的高级数据可视化库，提供更美观的图形和更简洁的接口
- scikit-learn：机器学习库，提供了大量的算法和模型，便于进行数据建模和预测
- statsmodels：统计建模库，提供了广泛的统计模型、测试和数据探索功能
- scipy：科学计算库，提供了一系列数值计算方法，如积分、优化等

2. Pandas 安装与使用

Python的设计者从来都没有想过能够大包大揽干完所有活。

Pandas是一个开源的Python库，专为数据处理和分析而设计。Pandas提供了强大的数据结构和函数，让数据分析变得更加简单高效。以下是使用Pandas的一些优点：

- 灵活的数据结构：提供了Series和DataFrame两种数据结构，能轻松处理不同类型的数据
- 数据处理：支持数据筛选、排序、缺失值处理、数据合并等多种操作，方便对数据进行处理和清洗
- 数据分析：内置许多统计分析功能，如数据分组、聚合、描述性统计等
- 高性能：基于NumPy构建，充分利用了NumPy的高性能计算能力，同时进行了一些优化，使得数据操作更高效

- 易于集成：与其他数据分析和可视化库（如matplotlib、seaborn、scikit-learn等）具有良好的兼容性，可

2.1安装Pandas

在命令行窗口输入：pip install pandas

安装完成后，可以通过以下代码检查Pandas是否安装成功：

In [2]:

```
import pandas as pd
print(pd.__version__)
```

1.4.4

2.2 读取和存储数据

读取excel文件

In [14]:

```
import pandas as pd

data = pd.read_excel('data.xlsx', sheet_name='Sheet1')

data
```

Out[14]:

	姓名	成绩
0	Alice	485
1	Bob	56
2	Cindy	56
3	David	39
4	Ele	569
5	Frank	526
6	Gurobi	89
7	Haha	789
8	Iraon	416

保存到excel文件

In [13]:

```
data.to_excel('output.xlsx', sheet_name='Sheet1', index=False)
```

2.3数据选择与筛选

Pandas提供了多种数据选择和筛选方法：

通过列名选择数据：

In [16]:

```
data['姓名']
```

Out[16]:

```
0     Alice
1       Bob
2     Cindy
3     David
4       Ele
5     Frank
6    Gurobi
7      Haha
8     Iraon
Name: 姓名, dtype: object
```

通过行号选择数据

In [19]:

```
data.loc[2]
```

Out[19]:

```
姓名     Cindy
成绩        56
Name: 2, dtype: object
```

使用条件筛选数据：

In [20]:

```
data[data['成绩'] > 100]
```

Out[20]:

	姓名	成绩
0	Alice	485
4	Ele	569
5	Frank	526
7	Haha	789
8	Iraon	416

使用多个条件筛选数据

In [23]:

```
data[(data['成绩'] > 100) & (data['成绩'] < 500)]
```

Out[23]:

	姓名	成绩
0	Alice	485
8	Iraon	416

2.4数据排序

按某一列单列升序排序

In [28]:

```
data.sort_values(by='成绩')
```

Out[28]:

	姓名	成绩
3	David	39
1	Bob	56
2	Cindy	56
6	Gurobi	89
8	Iraon	416
0	Alice	485
5	Frank	526
4	Ele	569
7	Haha	789

降序排序

In [30]:

```
data.sort_values(by='成绩', ascending=False)
```

Out[30]:

	姓名	成绩
7	Haha	789
4	Ele	569
5	Frank	526
0	Alice	485
8	Iraon	416
6	Gurobi	89
1	Bob	56
2	Cindy	56
3	David	39

2.5 数据缺失值处理

判断数据数据是否存在缺失值

In [36]:

```
data.isnull()
```

Out[36]:

	姓名	成绩
7	False	False
4	False	False
5	False	False
0	False	False
8	False	False
6	False	False
1	False	False
2	False	False
3	False	False

使用指定值填充缺失值:

In [38]:

```
data.fillna(123)
```

Out[38]:

	姓名	成绩
7	Haha	789
4	Ele	569
5	Frank	526
0	Alice	485
8	Iraon	416
6	Gurobi	89
1	Bob	56
2	Cindy	56
3	David	39

2.6 更通用和灵活的操作：数据索引

获取数据（表格）的元素

In [45]:

```
data.values
```

Out[45]:

789

索引某行某列的元素

In [51]:

```
data.values[0][1]
```

Out[51]:

789

2.7 判断某元素是否在数据中

In [56]:

```
a=789
if a in data.values:
    print("a在data中")
```

a在data中

2.7 实例演示

用Pandas进行数据处理

题目背景：你收到了一份来自某高校的学生信息Excel表格（文件名：students_info.xlsx），其中包含了学生的姓名、学号、年龄、性别、成绩等信息。由于表格存在一些数据问题，你需要使用Pandas进行数据预处理。

具体要求：

- 读取Excel表格数据。
- 检查数据中是否存在缺失值，对缺失值进行合适的处理。
- 对数据进行筛选，只保留成绩大于等于60分的学生信息。
- 对筛选后的数据按成绩降序排序。
- 将处理后的数据导出为新的Excel表格（文件名：students_info_cleaned.xlsx）。

In [ ]:

```

```

Have a try

敏感词文本文件 filtered_words.xlsx，里面的内容为以下内容，当用户输入敏感词语，则用 星号 * 替换，例如当用户输入「北京是个好城市」，则变成「**是个好城市」

> 北京
> 程序员
> 公务员
> 领导
> 牛比
> 牛逼
> 你娘
> 你妈
> love
> sex
> jiangge

(题目来自git@github.com (mailto:git@github.com):BigdogManLuo/show-me-the-code.git)

In [ ]:

```

```

In [ ]:

```
```