4.3 描述性统计方法

2023 年 9 月 18 日

描述性统计方法

1. 描述性统计方法

常用的描述性统计方法

统计学里的描述性统计方法	中文
Count	个数
Minimum	最小值
Maximum	最大值
Mean	均值
Median	中位数
Variance	方差
Standard deviation	标准差
Quintile	分位数

1.1 平均值

算术平均值用来描述一组数据，即“平均值”。它被定义为

$$\mu = \frac{\sum_{i=1}^{n} x_i}{n}$$

这里 x_1,$\cdots x_n$ 是我们的观测值

练习

找到以下序列的平均值 [2, 2, 4, 5, 5, 5, 8, 9, 9, 9, 12]，可以使用 numpy 的 `mean()` 方法。

[ ]:

1.2 标准差

数据离散程度的度量最常用的指标就是方差和标准差。它的计算公式如下：

$$\sigma^2 = \frac{\sum_{i=1}(x_i - \mu)^2}{n}$$

这里 x_1,$\cdots x_n$ 是我们的观测值、μ 为均值

找到以下序列的标准差 [2, 2, 4, 5, 5, 5, 8, 9, 9, 9, 12]，可以使用 numpy 的 `std()` 方法。

[ ]:

1.3 中位数

顾名思义，一组数据的中位数是当以递减或递增顺序排列时出现在数据中间位置的数字。

数据中位数不容易受极端值的影响。

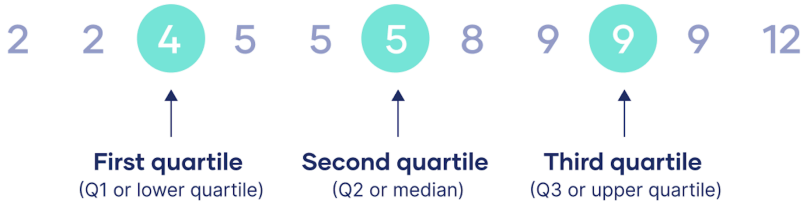
练习

找到以下序列的中位数 [2, 2, 4, 5, 5, 5, 8, 9, 9, 9, 12]，可以使用 numpy 的 `median()` 方法。

[ ]:

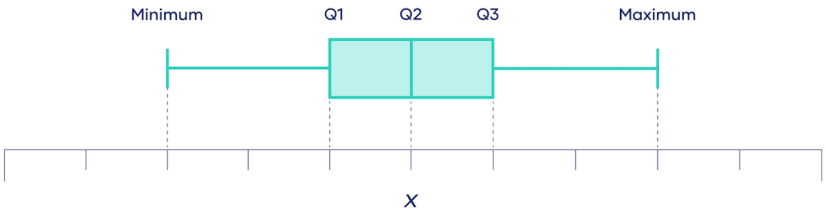
1.4 分位数

四分位数是将分类的数据分成四个部分的三个数值，每个部分的观察值的个数都是相等的。四分位数是分位数的一种类型。



第一个四分位数（Q1，或最低四分位数）是第 25 个百分位数，意味着 25% 的数据落在第一个四分位数以下。
第二个四分位数（Q2，或中位数）是第 50 个百分位数，意味着 50% 的数据低于第二个四分位数。
第三个四分位数（Q3，或上四分位数）是第 75 个百分位数，意味着 75% 的数据落在第三个四分位数以下。

箱型图可以在视觉上很好得描述数据集的分位数，它们由显示四分位数的方框和显示最低和最高观测值的“须”组成：



练习

找到以下序列的 75% 分位数 [2, 2, 4, 5, 5, 5, 8, 9, 9, 9, 12]，可以使用 numpy 的 `np.quantile(a=` 数组、`q=` 分位数）方法。

[ ]:

2.Pandas 下的描述性统计方法

首先，加载一个数据集文件到内存里

[7]:
```
import pandas
df=pandas.read_csv("数据/world-happiness-report-china.csv", index_col=0)
df
```

[7]:

国家名称	年份	幸福指数	经济生产	社会支持	预期寿命	自由	慷慨
China	2006	4.560	8.696	0.747	66.88	NaN	NaN
China	2007	4.863	8.824	0.811	67.06	NaN	-0.176
China	2008	4.846	8.911	0.748	67.24	0.853	-0.092
China	2009	4.454	8.996	0.798	67.42	0.771	-0.160
China	2010	4.653	9.092	0.768	67.60	0.805	-0.133
China	2011	5.037	9.179	0.787	67.76	0.824	-0.186
China	2012	5.095	9.249	0.788	67.92	0.808	-0.185
China	2013	5.241	9.319	0.778	68.08	0.805	-0.158
China	2014	5.196	9.386	0.820	68.24	NaN	-0.217
China	2015	5.304	9.449	0.794	68.40	NaN	-0.244
China	2016	5.325	9.510	0.742	68.70	NaN	-0.228
China	2017	5.099	9.571	0.772	69.00	0.878	-0.175

```
China  2018  5.131  9.632  0.788  69.30  0.895 -0.159
China  2019  5.144  9.688  0.822  69.60  0.927 -0.173
China  2020  5.771  9.702  0.808  69.90  0.891 -0.103
```

它返回的是一个 pandas.DataFrame 的对象，我们称之为“数据框”类型。

我们把它赋给变量名为 df 的变量。

如果只是想查看头和尾巴，可以使用 `dataframe.head()` 或者 `dataframe.tail()` 这两个方法。

[8]:
```
df.head()
```

[8]:
```
          年份    幸福指数    经济生产    社会支持    预期寿命      自由      慷慨
国家名称
China   2006  4.560  8.696  0.747  66.88     NaN     NaN
China   2007  4.863  8.824  0.811  67.06     NaN -0.176
China   2008  4.846  8.911  0.748  67.24   0.853 -0.092
China   2009  4.454  8.996  0.798  67.42   0.771 -0.160
China   2010  4.653  9.092  0.768  67.60   0.805 -0.133
```

2.1 describe 方法

它会返回我们的数据集的一些**基础描述性统计信息**。

[9]:
```
df.describe()
```

[9]:
```
                年份       幸福指数       经济生产       社会支持       预期寿命 ␣
↪       自由  \
count    15.000000  15.000000  15.000000  15.000000  15.000000  10.000000
mean   2013.000000   5.047933   9.280267   0.784733  68.206667   0.845700
std       4.472136   0.334580   0.322896   0.025689   0.933860   0.050524
min    2006.000000   4.454000   8.696000   0.742000  66.880000   0.771000
25%    2009.500000   4.854500   9.044000   0.770000  67.510000   0.805750
50%    2013.000000   5.099000   9.319000   0.788000  68.080000   0.838500
75%    2016.500000   5.218500   9.540500   0.803000  68.850000   0.887750
max    2020.000000   5.771000   9.702000   0.822000  69.900000   0.927000

             慷慨
count  14.000000
```

```
mean   -0.170643
std     0.042878
min    -0.244000
25%    -0.185750
50%    -0.174000
75%    -0.158250
max    -0.092000
```

这些会按照每一列的数据进行统计，结果包括：- count：个数 - mean：均值 - std：标准差 - min：最小值 - 25%：分位数为 25% 的数值 - 50%：分位数为 50% 的数值 - 75%：分位数为 75% 的数值 - max：最大值

Pandas 常用统计方法

函数名称	作用
.count()	非 NA 值的数量
.min()	最小值
.max()	最大值
.mean()	均值
.median()	中位数
.var()	方差
.std()	标准差
.skew()	偏度
.kurt()	峰度

如果我们想要计算某一列的最大值，可以这样：

[22]:
```
df["幸福指数"].max()
```

[22]:
```
5.771
```

同样，计算标准差，可以使用命令：

[21]:
```
df["幸福指数"].std()
```

[21]:
```
0.33458043377739044
```

2.2 包含缺失值的情况

当数据框包含缺失值时，比如

[24]:
```
df["自由"]
```

[24]:
```
国家名称
China      NaN
China      NaN
China    0.853
China    0.771
China    0.805
China    0.824
China    0.808
China    0.805
China      NaN
China      NaN
China      NaN
China    0.878
China    0.895
China    0.927
China    0.891
Name: 自由, dtype: float64
```

统计这一列的个数，是不考虑缺失值的：

[28]:
```
df["自由"].count()
```

[28]:
```
10
```

计算平均值，也是同样自动忽略缺失值，然后计算的。

[30]:
```
df["自由"].mean()
```

[30]:
```
0.8456999999999999
```

参考

- 分位数计算：https://www.scribbr.com/category/statistics/

[ ]: