数据科学 3 统计学

描述单个数据集

描述你部门成员有多少朋友?

描述方式1：直接显示所有的数据：

```
num_friends =
[100,49,41,40,25,21,21,19,19,18,18,16,15,15,15,15,14,14,13,13,13,13,12,12,11,10
,
10,10,10,10,10,10,10,10,10,10,10,10,10,10,9,9,9,9,9,9,9,9,9,9,9,9,9,9,9,8
,
8,8,8,8,8,8,8,8,8,8,8,8,7,7,7,7,7,7,7,7,7,7,7,7,7,7,7,6,6,6,6,6,6,6,6,6,6,6,6,6
,
6,6,6,6,6,6,6,6,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,5,4,4,4,4,4,4,4,4,4,4,4,4,4
,
4,4,4,4,4,4,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,2,2,2,2,2,2,2,2,2,2,2,2,2
,2,2,2,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1]
```

描述方式2: 直接绘图

```
def make_friend_counts_histogram():
    friend_counts = Counter(num_friends)
    xs = range(101)
    ys = [friend_counts[x] for x in xs]
    plt.bar(xs, ys)
    plt.axis([0,101,0,25])
    plt.title("Histogram of Friend Counts")
    plt.xlabel("# of friends")
    plt.ylabel("# of people")
    plt.show()
```

描述方式3: 统计信息
统计数据点个数

```
num_points = len(num_friends)
```

数据集最大与最小值

```
largest_value = max(num_friends)
smallest_value = min(num_friends)
```

特定位置的值

```
sorted_values = sorted(num_friends)
smallest_value = sorted_values[0]
second_smallest_value = sorted_values[1]
second_largest_value = sorted_values[-2]
```

中心倾向

有关数据中心位置的一些概念

均值 (mean 或 average)

数据和除以数据个数

```
def mean(x):
    return sum(x) / len(x)

mean(num_friends)
```

中位数 （median)

数据中间点的值（如果数据点的个数是奇数）， 或者中间两个点的平均值（如果数据点的个数是偶数）

例如：如果在排序向量x上有5个数据点，那么中位数就是x[5 2] 或 x[2]。如果有6个数据点，中位数=(x[2] + x[3]) / 2

均值与中位数的差别：

均值会随着数据变化而变化，中位数不依赖于每一个数据的值。计算中位数需要对数据先排序。均值受异常值影响很大。

取中位数

```
def median(v):
    """finds the 'middle-most' value of v"""
    n = len(v)
    sorted_v = sorted(v)
    midpoint = n // 2

    if n % 2 == 1:
        # if odd, return the middle value
        return sorted_v[midpoint]
    else:
        # if even, return the average of the middle values
        lo = midpoint - 1
        hi = midpoint
        return (sorted_v[lo] + sorted_v[hi]) / 2


median([1, 3, 8, 15, 20])
median([1, 3, 15, 20])
median(num_friends)
```

分位数 (quantity)

少于数据中特别百分比的一个值

```
def quantile(x, p):
    p_index = int(p * len(x))
    return sorted(x)[p_index]

quantile(num_friends, 0.10)
quantile(num_friends, 0.25)
quantile(num_friends, 0.75)
quantile(num_friends, 0.90)
```

众数 (mode)

出现次数最多的一个或多个数

```
def mode(x):
    """returns a list, might be more than one mode"""
    counts = Counter(x)
    max_count = max(counts.values())
    return [x_i for x_i, count in counts.items()
            if count == max_count]

mode(num_friends)
```

离散度

数据离散程度的一种度量，如果统计的值接近零，表示数据聚集在一起，离散程度很小，如果值很大，表示数据的离散度很大。

方法1: 极差 (range) 最大元素与最小元素的差

```
def data_range(x):
    return max(x) - min(x)
```

方法2: 方差 (variance)

```
def de_mean(x):
    x_bar = mean(x)
    return [x_i - x_bar for x_i in x]

def variance(x):
    n = len(x)
    deviations = de_mean(x)
    return sum_of_squares(deviations) / (n - 1)
```

方差的单位是“平方”， 我们更常使用标准差

方法3: 标准差 (standard deviation)

```
def standard_deviation(x):
    return math.sqrt(variance(x))
```

极差标准差都有可能受异常值影响

方法4: 计算75%的分位数和25%的分位数之差

```
def interquartile_range(x):
    return quantile(x, 0.75) - quantile(x, 0.25)
```

相关

验证用户在某网站上花费的时间与他的朋友数相关

研究流量日志，得出daily_minutes列表，每个用户每天花费多长时间，元素顺序与num_friends对应

```
daily_minutes =
[1,68.77,51.25,52.08,38.36,44.54,57.13,51.4,41.42,31.22,34.76,54.01,38.79,47.59
,
49.1,27.66,41.03,36.73,48.65,28.12,46.62,35.57,32.98,35,26.07,23.77,39.73,40.57
,
31.65,31.21,36.32,20.45,21.93,26.02,27.34,23.49,46.94,30.5,33.8,24.23,21.4,27.9
4,32.24,40.57,25.07,19.42,22.39,18.42,46.96,23.72,26.41,26.97,36.76,40.32,35.02
,
29.47,30.2,31,38.11,38.18,36.31,21.03,30.86,36.07,28.66,29.08,37.28,15.28,24.17
,
22.31,30.17,25.53,19.85,35.37,44.6,17.23,13.47,26.33,35.02,32.09,24.81,19.33,28
.
77,24.26,31.98,25.73,24.86,16.28,34.51,15.23,39.72,40.8,26.06,35.76,34.76,16.13
,
44.04,18.03,19.65,32.62,35.59,39.43,14.18,35.24,40.13,41.82,35.45,36.07,43.67,2
4.61,20.9,21.9,18.79,27.61,27.21,26.61,29.77,20.59,27.53,13.82,33.2,25,33.1,36.
65,18.63,14.87,22.2,36.81,25.53,24.62,26.25,18.21,28.08,19.42,29.79,32.8,35.99,
28.32,27.79,35.88,29.06,36.28,14.1,36.63,37.49,26.9,18.58,38.48,24.48,18.95,33.
55,14.24,29.04,32.51,25.63,22.22,19,32.73,15.16,13.9,27.2,32.01,29.27,33,13.74,
20.42,27.32,18.23,35.35,28.48,9.08,24.62,20.12,35.26,19.92,31.02,16.49,12.16,30
```

```
.
7,31.22,34.65,13.13,27.51,33.2,31.57,14.1,33.42,17.44,10.12,24.42,9.82,23.39,30
.93,15.03,21.67,31.09,33.29,22.61,26.89,23.48,8.38,27.81,32.35,23.84]
```

协方差 (convariance)

方差衡量了单个变量对均值的偏离程度，协方差衡量两个变量对均值的串联偏离程度。如果协方差是个大的正数，表示y很大，x也很大，或者y很小，x也很小。协方差为负，而且绝对值很大，表示x和y一个很大，一个很小。协方差接近0表示关系不存在。

```
def covariance(x, y):
    n = len(x)
    return dot(de_mean(x), de_mean(y)) / (n - 1)
```

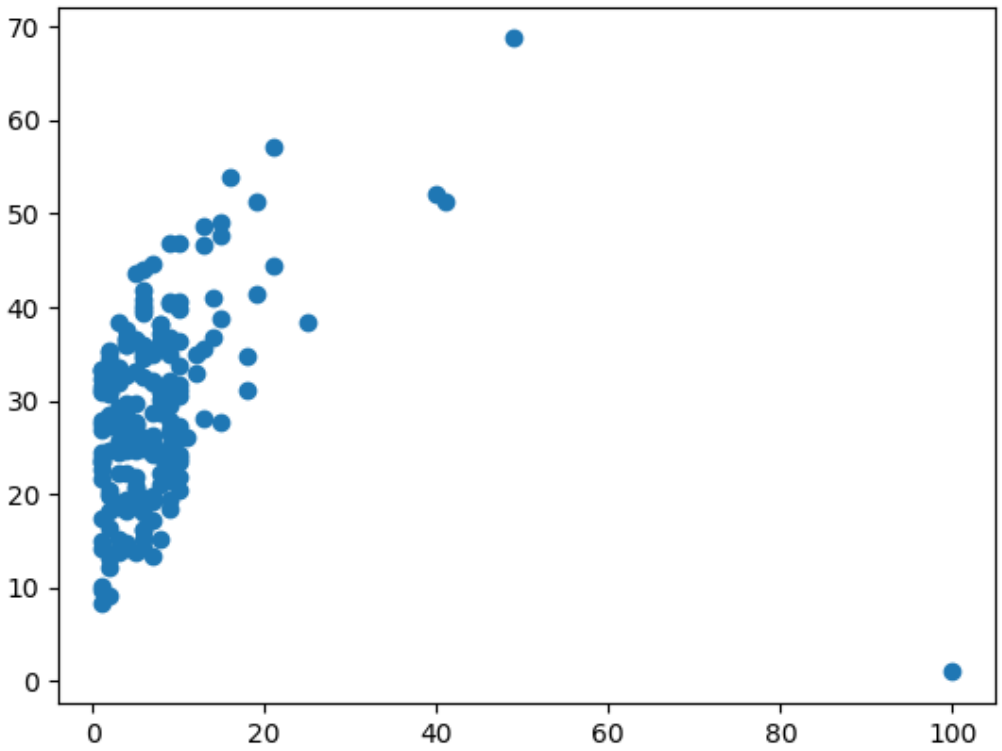
相关 (correlation)

协方差除以两个变量的标准差

```
def correlation(x, y):
    stdev_x = standard_deviation(x)
    stdev_y = standard_deviation(y)
    if stdev_x > 0 and stdev_y > 0:
        return covariance(x, y) / stdev_x / stdev_y
    else:
        return 0
```

相关系数没有单位，取值在-1(完全反相关）和1（完全相关）之间。相关值0.25表示一个弱的正相关。

```
plt.scatter(num_friends, daily_minutes)
plt.show()
```



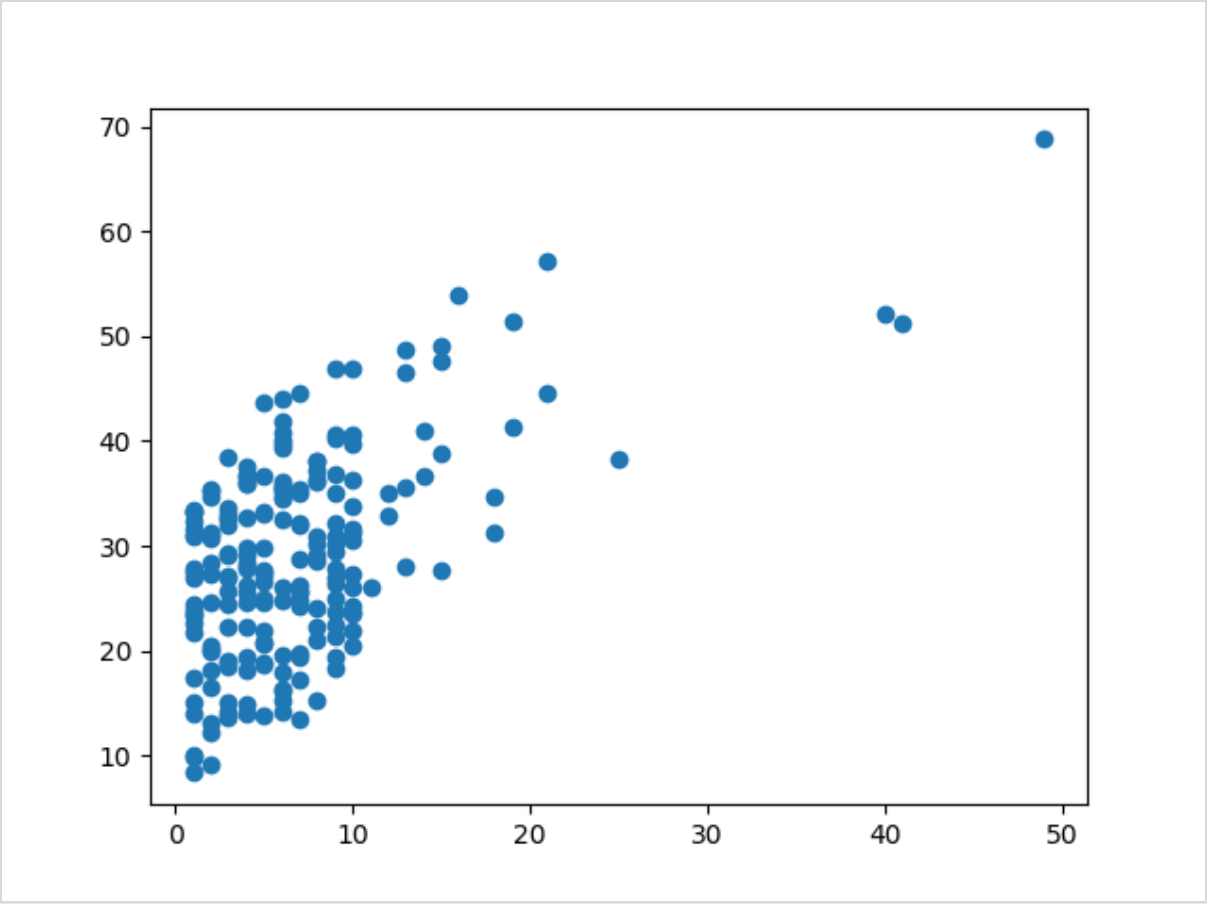
相关系数对异常值非常敏感，忽略100个朋友的那个用户

```
outlier = num_friends.index(100) # index of outlier

num_friends_good = [x
                    for i, x in enumerate(num_friends)
                    if i != outlier]

daily_minutes_good = [x
                      for i, x in enumerate(daily_minutes)
                      if i != outlier]

plt.scatter(num_friends_good, daily_minutes_good)
```



辛普森悖论

分析数据时可能发生的意外，如果忽略了混杂变量，相关系数会有误导性。

验证哪边好友关系多

海岸	成员数	平均朋友数
西海岸	101	8.2
东海岸	103	6.5

相关系数假设在条件都相同的前提下比较两个变量的关系，需要充分了解数据，核查所有可能的混杂因素。

海岸	学位	成员数	平均朋友数
西海岸	博士	35	3.1
东海岸	博士	70	3.2
西海岸	非博士	66	10.9
东海岸	非博士	33	13.4

相关和因果

如果x和y强相关，说明可能x引起了y,或者y引起了x, 或者互相引起，或者有第3三方因为同时引起x和y。可以将一组具有类似统计数据的用户随机分为两组，然后施加影响看变化。