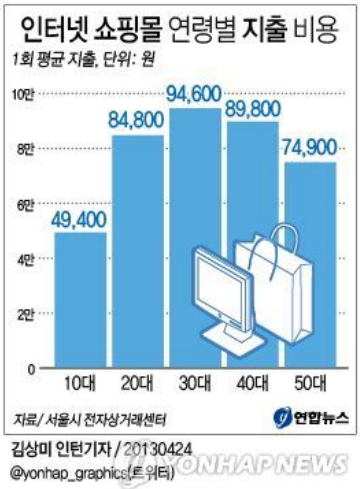
Chapter 3 Descriptive Statistics

인터넷쇼핑몰 큰손은 '30대'…1회평균 지출 9만5천원



<그래픽> 인터넷 쇼핑몰 연령별 지출 비용

(서울=연합뉴스) 박영석 기자 = 서울시 전자상거래센터는 최근 1년간 인터넷 쇼핑몰을 이용한 경험이 있는 4천명을 대상으로 이용실태를 설문한 결과 30대 1회 평균 지출액이 가장 많은 것으로 파악됐다고 24일 밝혔다. zeroground@yna.co.kr @yonhap_graphics(트위터)

1회 지출 男 10만5천원 〉 女 8만원…서울시 이용실태 조사결과

(서울=연합뉴스) 국기헌 기자 = 서울에 사는 30대가 인터넷 쇼핑몰의 '큰 손'인 것으로 나타났다. 또 남성이 여성보다 인터넷 쇼핑몰에서 돈을 더 많이 쓰는 것으로 파악됐다 서울시 전자상거래센터는 최근 1년간 인터넷 쇼핑몰을 이용한 경험이 있는 4천명을 대상으로 이용실태를 설문한 결과 이같이 파악됐다고 24일 밝혔다.

인터넷쇼핑몰에서의 1회 평균 지출액은 약 9만원으로 2011년의 약 8만원에서 1만원가량 늘었다. 11만원 이상 지출하는 비율도 11%에서 13.3%로 높아졌다.

연령별로는 30대의 1회 평균 지출이 9만4천600원으로 가장 높았다. 이어 40대 8만9천800원, 20대 8만4천800원, 50대 7만4천900원, 10대 4만9천400원 등 순이었다.

성별로는 남성이 여성보다 인터넷 쇼핑몰에서 돈을 더 많이 쓰는 것으로 나타났다. 남성의 1회 평균 지출비용은 10만4천600원으로 여성의 8만400원을 웃돌았다.

그러나 여성이 남성에 견줘 더 자주 인터넷 쇼핑몰을 이용하는 것으로 조사됐다. 주 2회 이상 이용한

비율은 여성이 27.8%로 남성의 21.2%보다 높았다.

조사대상의 92.9%는 인터넷 쇼핑몰을 월 1회 이상 이용하며, 48.9%는 주 1회 이상 물건을 사는 것으로 나타났다.

인터넷 쇼핑을 하는 소비자들이 가장 많이 사는 품목은 의류·패션 관련 상품이었다. 이어 화장품, 서적, 생활용품 순이었으며, 과거보다 가전제품이나 컴퓨터 등의 구매는 줄어드는 추세인 것으로 나타났다.

인터넷 쇼핑몰 이용 중 피해를 경험했다고 응답한 비율은 28.2%로 전년에 비교해 소폭 감소했다.

피해 내용은 제품 불량·하자에 따른 청약철회 관련이 37.8%로 가장 많았고 배송지연(18.6%), 허위·과장광고(13.6%), 상품정보 오기(7.4%)가 뒤를 이었다.

소비자 불만을 줄이기 위한 개선점으로는 상품정보를 정확하고 상세하게 표기해야 한다는 응답이 18.7%로 가장 많았다.

피해 대응방법에 대해서는 70.1%가 '사업자에게 직접 연락해 해결하고 있다'고 응답했으며 '소비자 보호기관이나 단체에 신고한다'는 소비자는 6.8%에 그쳤다.

(c)연합뉴스 2013/04/24

길거리 매장도 이용을 하지만 집에서 같은 물건의 가격 비교도 쉽고 집까지 배송도 해주는 온라인 쇼핑몰의 이용이 많아졌습니다. 이러한 현상에 대한 이해를 돕기 위해서 신문 기사는 인터넷 쇼핑몰 이용자들이 한달 지출 비용, 이용률등을 조사하여 작성되었습니다. 그런데 기사를 보면 조사된 모든 사람의 한달 지출 비용과 이용한 날짜를 등을 기록하지 않고 이 데이타으로 부터 내용을 정리한 숫자값들을 알려 주고 있습니다. 이러한 방식의 기사들은 신문, 뉴스, 잡지 등과 같은 글에서 많이 보실 것입니다. 그런데 기사를 보시면 지출 비용을 설명하면서 평균값만을 알려주지 지출의 다양한 정도를 알려 주지는 않습니다. 이와 같이 수집한 데이타를 설명하기 위해서 모든 데이타를 하나하나 말하는 것 보다 데이타를 정리하여 표현할 수 있도록 특정 값들을 계산하여 의미 전달과 이해를 빠르게 할 수 있습니다. 이렇게 데이타를 요약 정리할 수 있도록 표현하는 것이 descriptive statistics 입니다.

이 장에서는 데이타를 요약 정리하기 위한 수단으로

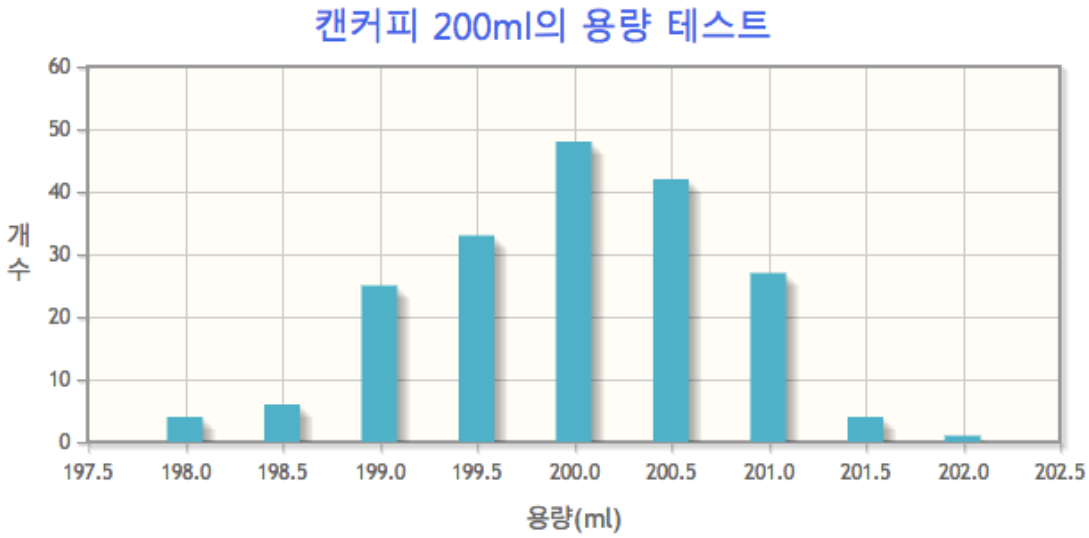
- 중앙값 표현 방법

- 데이타 분산 정도를 계산하는 방법
- 어느 값이 data 속에서 어느 위치에 있는지 혹은 어떤 위치에 어떤 값이 존재하지 알아내는 방법

들을 학습하겠습니다.

1. 중간값 표현하기

기사에와 같이 인터넷 쇼핑몰을 이용한 사람들의 1 회 지출액을 나타낼 때 중간값으로 표현을 하는 경우와 같이 중간값으로 정보를 제공하는 경우가 많습니다. 이렇게 중간값으로 표현을 하는 이유는 자료로 부터 얻은 값들은 어느 특정 값 주변에 모여 있다고 생각을 하기 때문입니다. 예를 들어 캔커피에 제조된 커피를 담는데 200ml 를 정확하게 담지 못하고 198ml 에서 202ml 사이로 담는다고 할 때 다음과 같은 histogram 이 생성이 된다 했을 경우 중간값으로 표현하는 것이 한 캔당 용량을 말하는데 편리합니다.



도표에서 보듯이 조사 결과 200ml 로 캔커피 용량이 치중되는데 이 값을 central tendency 라고 합니다. 그럼 central tendency 를 표현하는 mean(평균), median(중앙값), mode(최빈값), 기하 평균에 대해서 알아 보겠습니다.

1.1. mean(평균)

데이타를 표현할 때 가장 많이 사용되는 것이 평균입니다. 이 값을 계산하는 방법은 Population 일 경우나 Sample 일 경우 동일하지만 표기 방법이 다릅니다.

Population mean	Sample mean
$\mu = \frac{\sum_{i=1}^{N} x_i}{N}$	$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}$

(3.1)

여기서 N 은 population 의 data 값의 개수이고 n 은 sample data 값의 개수입니다.

예를 들어 6 개월간 휴대폰 요금으로 52,900 원, 51,200 원, 55,030 원, 53,100 원, 51,900 원, 52,400 원 이라면 6 개월간 평균 휴대폰 요금은 52,755 원이 됩니다.

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} = \frac{52900 + 51200 + 55030 + 53100 + 51900 + 52400}{6} = 52755$$

jMath 에서는 mean()함수를 사용하면 됩니다.

```
> jMath([52900,51200,55030,53100,51900,52400]).mean(3)
52755
```

이 계산을 sum()함수를 이용한다면 다음과 같습니다.

```
> a = jMath([52900,51200,55030,53100,51900,52400]);
> a.sum(3)/a.cols
52755
```

평균을 구하는 방식은 앞의 예제와 같이 단순히 값을 나열한 값을 더해서 총 개수로 나누는 방식과 반복되는 값들을 나열하고 반복되는 비율(weight)을 나타내는 frequency table 로부터 평균을 구하는 방식이 있습니다. 예를 들의 반도체 공장에서 하루 평균 불량 제품의 개수를 알기 위해서 불량품 개수가 나타난 날을 횟수를 조사를 할 경우로 예를 들어 1,2,0,3,1,2, ...와 같이 매일 나오는 불량제품의 수를 다음과 같은 표로 정리가 될 수 있습니다.

불량수(x)	횟수(weight)
0	27 일

1	31 일
2	28 일
3	5 일

이러한 자료를 통해서 하루 평균 불량 제품의 개수를 알기 위한 수식은 다음과 같습니다.

$$\overline{x} = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i}^{n} w_i} \tag{3.2}$$

이 수식을 적용하면 원하는 값을 얻을 수 있습니다.

$$\overline{x} = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i}^{n} w_i} = \frac{27 \times 0 + 31 \times 1 + 28 \times 2 + 5 \times 3}{27 + 31 + 28 + 5} = \frac{102}{91} = 1.12$$

이 결과를 통해서 하루 평균 1.12 개의 불량 제품이 나타나는 것으로 데이타를 요약할 수 있게 되었습니다.

jMath 의 wmean()함수를 통해서 계산이 가능합니다.

```
> jMath([[0, 27],[1,31],[2,28],[3,5]]).wmean();
[ Array[2]
0: 1.120879120879121
1: 91
length: 2
__proto__: Array[0]]
```

jMath 객체를 만들기 위해서 첫번째 column 은 평균을 구하려는 값이고 두번째 column 값이 발생되는 횟수로 weight 에 해당합니다. 결과 값은 JavaScript 의 배열값으로 첫번째 요소값이 평균이고 두번째 요소값이 weight 의 합입니다.

이 계산을 풀어서 계산하면 다음과 같습니다.

```
> a = jMath([[0, 27],[1,31],[2,28],[3,5]]);
> a.slice(':', 0)['.*']( a.slice(':', 1) ).sum(3) / ( a.slice(':',1).sum(3))
1.120879120879121
```

여기서 jMath.prototype.slice(rows, cols)함수는 Matrix 를 잘라서 필요한 부분만 추출하는 함수 입니다. Rows 와 cols 값은 양의 정수, 문자열, 배열의 입력값을 받습니다.

$$a = \begin{bmatrix} 0 & 27 \\ 1 & 31 \\ 2 & 28 \\ 3 & 5 \end{bmatrix}$$

두번째 column 만 빼기	세번째 row 만 빼기	(세번재, 네번째 row 첫번째, 두번째 column)
> a.slice(':',1).toString() "27 31 28 5"	> a.slice(2,':').toString() "2 28"	> a.slice('2:3',[0,1]).toString() "2 28 3 5" > a.slice('2:end',[0,1])

문자열의 경우는 ':'는 전체를 의미하고 시작과 끝은 's1:s2'로 s1 에서 부터 s2 까지 입니다. 만일 s2 가 'end'일 경우 끝까지 입니다. 배열의 경우는 해당 위치들 값을 나열한 값이고, 양의 정수는 한개의 위치값입니다.

1.2. median(중앙값)

mean 의 단점으로 이 값 하나만을 갖고는 데이타를 표현하는데 잃어 버리는 정보가 너무 많습니다. 그래서, 만약 데이타에서 어느 한 값이 다른 값들에 비해서 너무 크거나 작으면 결과로 얻는 mean 값의 의미는 사라집니다. 예를 들어 일인당 평균 기부 금액을 알기 계산해보도록 하겠습니다.

20,000 30,000 50,000 30,000 40,000 30,000 40,000 50,000 30,000 10,000,0000

일반적으로 3 만원에서 5 만원을 기부 금액으로 내는데 한 사람이 천만원을 기부로 평균 일인당 기부 금액을 약 1 백만 3 만 2 천원입니다. 즉 천만원의 기부 금액은 평균을 왜곡 시키게 됩니다. 이러한 데이타에서 다른 값과 달리 너무 크거나 너무 작은 값을 outlier(이상점)들이라고 합니다.

이러한 단점을 보안하는 방법으로 median(중앙값)을 이용할 수 있습니다. 이 값은 데이타의 값을 순서대로 정렬을 했을 때 데이타 중앙에 위치한 값입니다. 즉 이 값을 중심으로 위, 아래의 데이타 값의 개수가 같아야 합니다. 예를 들어서 5 개의 값이 다음과 같이 정렬이 되어 있을 경우에

4, 5, 6, 7, 9

중앙에 있는 6 이 아래 2 개 위로 2 개의 데이타가 있습니다. 바로 6 이 이 데이타의 median 입니다.

그런데 만약에 다음과 같이 6 개의 값이 정렬되었을 때 median 을 구하는 방식은 조금 다릅니다.

4, 5, 6, 8, 8, 9

여기서 중앙은 6 과 8 사이에 있기 때문에 이 사이에 반값인 7 이 median 값이 됩니다. 다음은 이러한 계산 과정을 정리한 내용입니다.

데이타 개수가 홀수 일 때	데이타 개수가 짝수 일 때
$x_{\lfloor n/2 \rfloor}$	$\frac{x_{\lfloor n/2 \rfloor} + x_{\lfloor n/2 \rfloor - 1}}{2}$

예를 들어 5 개의 데이타가 있으면 ⌊5/2⌋인 index 가 2 인 위치값이 median 이 되고 6 개일 경우는 index 가 2 와 3 위치에 값에 평균값이 됩니다.

그럼 median 이 어떻게 outlier 의 영향을 없애는 가를 보기 위해서 기부 데이타를 다시 보시면 금액을 정렬하겠습니다.

20,000 30,000 30,000 30,000 30,000 40,000 40,000 50,000 50,000 10,000,0000

10 개값의 중앙은 30,000 과 40,000 사이이기 때문에 median 값은 3 만 5 천원으로 천만원의 기부금액을 제외했을 때의 평균과 거의 비슷합니다.

jMath 에서 median 값을 구하는 것은 median()함수를 사용합니다.

```
> jMath([20000,30000,50000,30000,40000,30000,40000,50000,30000,
10000000]).median(3)
35000
```

1.3. mode(최빈값)

데이타 값이 반복해서 나타날 때 가장 빈번하게 나타나는 값이 mode 입니다. 예를 들어 자동차 2014 년 1 월 자동차 판매 순위 집계 자료를 보면 다음과 같습니다.

모델	판매대수
현대 쏘나타	5,117
현대 아반떼	5,154
기아 모닝	6,235
현대 그랜저	8,134
기아 K5	4,000

출처: 메가오토(www.megaauto.com)

여기서 mode 는 8,134 대가 팔린 그랜저 입니다. 보시는 것과 같이 이 central tendency 는 categorical data 에 적합 합니다.

하지만 mode 는 항상 존재 하지 않습니다. 예를 들어서 하루 휴대폰 판매량이 다음과 같다고 할 때 모든 값은 한번만 나타납니다.

```
10, 9, 8, 5, 11, 13, 7, 12
```

Mode 의 다른 경우는 두 개의 값이 발생된 횟수가 같을 경우로 이러한 distribution 을 bimodal distribution 이라고 합니다.

jMath 에서 제공하는 mode 함수는 최대 반복되는 값들을 찾아 줍니다.

Mode 의 개수	예
1	`> jMath([10, 9, 8, 5, 10, 13, 7, 12]).mode()` `["10"]`
2	`> jMath([1,1,1,3,2,3,5,3,4,6,2]).mode()` `["1", "3"]`
0	`> jMath([1,2,3,4,5,6]).mode()` `null`

만일 mode 가 없으면 null 값을 돌려 줍니다.

1.4. Geometric mean(기하평균)

$$G = \left(\prod_{i=1}^{n} x_i\right)^{\frac{1}{n}} = \sqrt[n]{x_1 x_2 \dots x_n} \tag{3.3}$$

예를 들어 직사각형 가로 세로 길이가 2 와 8 일 때 넓이는 16 으로 이것은 변 길이가 4 인 정사각형과 같습니다. 이처럼 기하학적으로 각 길이의 평균을 구하는 것이 geometric mean 으로 단위는 같지만 다른 특성을 갖고 있는 값들의 평균을 얻기 위해서 사용됩니다.

경제에서 사용되는 예는 인구 증가률, 연평균 물가 상승률, 경제 성장률, 투자에 평균값을 얻기 위해서 사용될 수 있습니다. 투자에 대한 예를 들면 상품 A,B,C 에 투자를 해서 난 수익률일 각각 110%, 120%, 105%일 때 평균 수익률은

$$G = \sqrt[3]{1.1 \times 1.2 \times 1.05} = 1.115$$

로 원금에 11.5% 평균 수익률을 얻은 것으로 나타납니다.

```
> jMath('1.1 1.2 1.05').geomean(3)
1.11494747954535
```

jMath 기본 함수와 방향

jMath 의 몇 기본 함수들은 계산을 할 방향에 따라서 계산 결과가 다르게 나타납니다. 예를 들어 입력된 matrix 에 합을 계산을 할 때 다음의 3 가지 방향으로 계산을 할 수 있습니다.

```
var a = jMath([ [1,2], [3,4] ])
```

$$a = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

Column 별	Row 별	전체
`a.sum(1)` $[4 \quad 6]$	`a.sum(2)` $\begin{bmatrix} 3 \\ 7 \end{bmatrix}$	`a.sum(3)` 10

이와 같이 방향에 따라 계산 방식이 다른 함수들은 다음에 함수들이 있습니다.

jMath.prototype.sum, jMath.prototype.prod, jMath.prototype.mean,
jMath.prototype.geomean, jMath.prototype.median
jMath.prototype.var, jMath.prototype.std, jMath.prototype.max, jMath.prototype.min,
jMath.prototype.range, jMath.prototype.percentile

여기서 jMath.prototype.prod 는 element 의 값들을 곱하는 함수이고, var, std, max, min, range, percentile 은 다음에 다룰 함수들 입니다.

Column 별	Row 별	전체
`a.prod(1)` $[3 \quad 8]$	`a.prod(2)` $\begin{bmatrix} 2 \\ 12 \end{bmatrix}$	`a.prod(3)` 24

jMath.prototype.wmean()함수에 평균값을 jMath.prototype.prod()와
jMath.prototype.sum()으로 표현하면 다음과 같습니다.

```
> a = jMath([[0, 27],[1,31],[2,28],[3,5]])
> a.prod(2).sum(3)/a.sum(1)[0][1]
1.120879120879121
```

계산 과정을 보면

$$a = \begin{bmatrix} 0 & 27 \\ 1 & 31 \\ 2 & 28 \\ 3 & 05 \end{bmatrix}, \quad a.\mathrm{prod}(2) = \begin{bmatrix} 0 \\ 31 \\ 56 \\ 15 \end{bmatrix}, \quad a.\mathrm{prod}(2) = 102, \quad a.\mathrm{sum}(1) = [6 \quad 91]$$

이 결과 값으로 부터 평균값을 얻을 수 있습니다.

방향이 column 과 row 만 있는 함수들

jMath.prototype.percentrank, jMath.prototype.sort, jMath.prototype.diff

2. 분산된 정도 표현하기

신문이나 뉴스를 통해서 우리가 자주 접하는 평균과 같은 Central tendency 는 우리에게 익숙하지만 단점있습니다. 이 값은 자주 나타나는 값이나 가운데 값이기 때문에 데이타 값들의 밀집 정도를 알지는 못합니다. 예들 들어 인터넷 쇼핑몰 이용에 대한 기사를 보면 평균만 나타나지 실제로 어떻게 데이타가 밀집되어 있는가는 기사를 통해서 알수가 없습니다.

예를 들어서 다음의 두가지 경우를 보시겠습니다.

1, 2, 3, 4, 5
2, 2, 3, 4, 4

둘다 평균값은 3 입니다. 하지만 두 번째 데이타가 첫 번째 데이타보다 밀집해 있습니다. 따라서 데이타를 표현하는 데이타의 밀집 정도를 알려주는 값이 필요합니다. 이를 위해 range(범위), variance(편차), standard deviation(표준편차)를 사용할 수 있습니다.

2.1. Range(범위)

데이타에서 가장 큰 값에서 가장 작은 값을 뺀 값이 range(범위)입니다.

$$\text{Range} = \text{Maximum value} - \text{minimum value} \tag{3.4}$$

예를 들어 10 일을 동안 꽃이 판매된 개수를 갖고 range 를 계산 하려고 합니다.

10, 8, 9, 5, 2, 4, 10, 5, 7, 7

이 데이타에서 range 는 최대값 10, 최소값 2 이기 때문에 8 이 됩니다.
그런데 range 문제는 단지 최대 최소값 만으로 값을 구하기 때문에 만일 outlier 들이 존재를 했을 경우 값의 다양성을 알기 어렵습니다. 예를 들어 앞의 기부 금액의 경우 한 사람이

천만원 기부로 인해서 range 는 9 백 9 십 8 만원이 됩니다. 천만원을 제외하면 range 는 3 만원으로 많은 차이가 나게 됩니다.

jMath 에서는 range() 함수로 부터 최대값, 최소값을 얻을 수 있습니다.

```
> jMath([10, 8, 9, 5, 2, 4, 10, 5, 7, 7]).range(3)
[2, 10]
```

2.2. Variance(편차)와 Standard deviation(표준편차)

Range 와 다르게 모든 데이타 값을 갖고 데이타 값의 퍼트려져 있는 정도를 측정하는 방식입니다. 계산 방식은 Population 과 sample 에 따라 다름니다.

- **Variance**

Population	Sample
$\sigma^2 = \frac{\sum_{i=1}^{N}(x_i - \mu)^2}{N}$	$s^2 = \frac{\sum_{i=1}^{n}(x_i - \overline{x})^2}{n-1}$

(3.5)

여기서 N 은 population 데이타 값 개수이고 n 은 sample 데이타 값 개수 입니다.

여기서 분자를 sum of square(SS)라고 합니다. 정확하게 표현을 하면 corrected sum of square 라고 하고 uncorrected sum of square 는

$$\sum_{i=1}^{n} x_i^2$$

입니다.

Sum of square 이 의미는 오류값의 제곱에 합으로

$$x_i = \mu + e_i$$

$$\sum_{i=1}^{N}(x_i - \mu)^2 = \sum_{i=1}^{N} e_i^2$$

이 값의 평균이 편차(variance)입니다. 그런데 population 과 sample 에 평균을 위한 분모값이 다른것에 주의 해야 합니다.

Statistic 을 계산을 할 때 degree of freedom 을 많이 이용합니다. Degree of freedom 이라는 말은 데이타 중 자유로운 값의 개수입니다. Sample variance 의 경우 모든 x 의 값을 평균값으로 빼서 합을 하게 되면 무조건 결과는 0 입니다. 예를 들어 1,2,3,4,5 라는 5 개의 값의 평균은 3 으로 각각의 값에 3 을 빼서 다시 합해 보면 0 이 됩니다.

$$(1-3)+(2-3)+(3-3)+(4-3)+(5-3)=0$$

이 수식에서 4 개의 값은 아무값이나 하고 한개의 값만 합이 0 이 되게 만들면 됩니다. 다시 말해, n-1 개는 자유롭고 한개만 자유롭지 못합니다. 그래서 sample variance 에 degree of freedom 은 n-1 이고 이 값을 분모로 사용한 것입니다.

예를 들어 10 일을 동안 꽃이 판매된 개수

10, 8, 9, 5, 2, 4, 10, 5, 7, 7

으로 population variance 와 sample variance 를 계산하면 다음과 같습니다.

Population mean 과 sample mean 계산 방식은 같기 때문에 동일한 값을 적용 할 수 있습니다.

$$\mu=\bar{x}=\frac{10+8+9+5+2+4+10+5+7+7}{10}=\frac{67}{10}=6.7$$

이 평균값을 Sum of Square(SS)에 적용을 하게 되면 다음과 같습니다.

$$(10-6.7)^2+(8-6.7)^2+(9-6.7)^2+(5-6.7)^2+(2-6.7)^2+(4-6.7)^2+(10-6.7)^2 \\ +(5-6.7)^2+(7-6.7)^2+(7-6.7)^2=64.1$$

$$\sigma^2=\frac{SS}{N}=\frac{64.1}{10}=6.41$$
$$s^2=\frac{SS}{n-1}=\frac{64.1}{9}=7.12$$

jMath 에서 var(isPopulation, dir)함수를 사용하여 편차를 계산할 수 있습니다.

```
> jMath([10, 8, 9, 5, 2, 4, 10, 5, 7, 7]).var(true,3)
6.410000000000001

> jMath([10, 8, 9, 5, 2, 4, 10, 5, 7, 7]).var(false,3)
7.122222222222224
```

계산 과정을 풀어서 계산을 하면 다음과 같습니다.

```
> a = jMath([10, 8, 9, 5, 2, 4, 10, 5, 7, 7]);
> ss = a['-'](a.mean(3))['.^'](2).sum(3);
> ss/a.cols
6.410000000000001
> ss/(a.cols-1)
7.122222222222224
```

- **Standard deviation**

Variance 의 제곱근입니다.

Population	Sample
$\sigma = \sqrt{\sigma^2} = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \mu)^2}{N}}$	$s = \sqrt{s^2} = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \overline{x})^2}{n-1}}$

(3.6)

Standard deviation 의 의미는 data 값과 평균(mean)의 평균 거리 즉 오류의 평균값 입니다. 또한 Variance 는 값과 평균의 차에 제곱을 했지만 standard deviation 은 제곱근 때문에 standard deviation 의 단위는 원래 데이타 단위와 같습니다. 따라서 data 를 기술 할 때 standard deviation 이 더 적합합니다.

예를 들어 철판을 판매하는 업체에서 철판을 200mm 단위로 자르려고 합니다. 제대로 잘려졌는가를 알기위해서 15 개를 sample 의 길이를 측정해 보았습니다.

192mm, 198mm, 199mm, 199mm, 197mm,
201mm, 202mm, 203mm, 197mm, 196mm,
204mm, 206mm, 207mm, 202mm, 199mm

이 값들로 variance 를 구하면 단위가 mm^2 가 됩니다. 하지만 standard deviation 은 mm 가 됩니다. Sample 의 평균이 200.133mm 일 때 편차는

$$\frac{(192 - 200.133)^2 + (198 - 200.133)^2 + \cdots + (199 - 200.133)^2}{15 - 1} = 15.981$$

jMath 의 var 와 stdev 로 편차와 표준편차를 알아 보겠습니다.

```
> jMath([192,198, 199,199, 197, 201, 202, 203, 197, 196, 204, 206, 207, 202,
199]).var(false,3)
15.980952380952386

> jMath([192,198, 199,199, 197, 201, 202, 203, 197, 196, 204, 206, 207, 202,
199]).std(false,3)
3.9976183385801582
```

이 예제는 sample 에 대한 편차와 표준 편차를 얻기 위해서 각 함수의 첫번째 함수에 false 값을 넣은 것인데 만일 population 에 대한 값을 얻고 싶다면 true 로 호출을 하면 됩니다.

Standard deviation 이 평균 거리라는 의미의 이해를 돕기 위해서 처음에 것을 Case1 이라고 하고 다른 2 개 장비로 자른 결과 값들에 sample 을 추가 했을 경우 standard deviation 을 비교하면 이해가 되실 겁니다.

Case 2: Standard deviation 이 커짐	Case 3: Standard deviation 이 작아짐
192, 193, 194, 193, 197, 201, 203, 207, 195, 196, 205, 206, 207, 204, 199	197, 198, 199, 199, 197, 201, 202, 203, 197, 197, 202, 201, 203, 202, 199

단위: mm

	평균(mean)	편차(variance)	표준편차(stddev)
Case 1	200.133mm	15.98mm^2	3.9976mm
Case 2	199.467mm	30.69mm^2	5.5403mm
Case 3	199.8mm	5.31mm^2	2.305mm

데이타를 자세히 보시면 Case 2 값들이 많이 떨어져 있고, Case 3 값들이 가장 밀집하다는 것을 알 수 있습니다. 이러한 정보는 Standard deviation 을 보면 쉽게 알 수 있습니다.

Population variance 수식을 변경하여 간략하게 만들어 보겠습니다.

$$\frac{1}{N}\sum_{i=1}^{N}(x_i-\mu)^2=\frac{1}{N}\left(\sum_{i=1}^{N}x_i^2-2\mu\sum_{i=1}^{N}x_i-N\mu^2\right)=E(\boldsymbol{X^2})-\mu^2$$

여기서 $E(X^2)$는 데이타 값들의 제곱에 대한 평균값을 의미합니다.

Sample variance 수식의 변경은 N 이 아니라 n-1 이기 때문에 위의 수식과 다르게 나타납니다.

$$\frac{1}{n-1}\sum_{i=1}^{N}(x_i-\overline{x})^2=\frac{n}{n-1}\frac{1}{n}\left(\sum_{i=1}^{n}x_i^2-2\bar{x}\sum_{i=1}^{n}x_i-n\overline{x}^2\right)=\frac{n}{n-1}\left(E(\boldsymbol{X^2})-\overline{x}^2\right)$$

이 수식에서 보듯이 만일 sample 의 크기 n 이 커질 수록 n-1 과 차이가 거의 없어 1 이 되어 population 의 값과 같게 됩니다.

2.3. Coefficient of Variance (CV)

Standard deviation 만 사용을 했을 경우 데이타가 얼마나 밀집하고 퍼트려져 있는가를 알 수 있습니다. 하지만 만일 데이타 값이 숫자 100 만, 200 만와 같이 큰 숫자로 구성된 것과 데이타 값이 10,20 인 것과 같이 작은 숫자로 구성된 것을 비교할 때 standard deviation 은 숫자값이 클 수록 크게 나타나기 때문에 밀집도 비교를 하기에 무리가 있습니다.

이러한 문제를 해결하기 위해서 standard deviation 을 mean 의 비율로 값을 구하는 Coefficient of Variance(CV)를 활용하면 해결이 됩니다.

Population	Sample
$CV=\frac{\sigma}{\mu}$	$CV=\frac{s}{\overline{x}}$

(3.7)

그럼 2014 년 3 월 무와 대파의 kg 당 가격을 CV 의 활용도를 알아 보도록 하겠습니다.

	03/17	03/18	03/19	03/20	03/21	03/22	03/23	03/24	03/25

무	6248	6631	6567	6624	6297	5686	5883	6488	6884
대파	1118	1039	907	756	736	668	813	1203	1051

KREI 농업통계(www.krei.re.kr)

	평균	표준편차	CV(100%표기)
무	6367	383	6%
대파	921	189	21%

무 가격이 대파 가격 보다 높기때문에 표준편차가 대파보다 크게 나타나는건 무 가격의 밀집도가 매우 높지 않고는 당연합니다. 하지만 CV 를 보시면 대파의 가격이 더 불안하다는 것을 알 수 있습니다.

jMath 에서 Coefficient of Variance 를 계산하는 별도의 함수를 제공하지 않기 때문에 직접 평균과 표준편차를 계산해야 합니다. 무와 대파 가격의 예를 보시면

```
var data = jMath([[6248,1118], [6631,1039], [6567,907], [6624,756],[6297,736],
                  [5686,668],[5883,813],[6488,1203],[6884,1051]]);
var mean = data.mean();
var stdev = data.std();
var cv = stdev['./'](mean);
```

jMath 객체를 생성을 하고 mean 과 std 함수에 계산 방향을 column 로 하면 되기 때문에 jMath.prototype.mean(1)과 같이 호출하면 되는데 입력값 1 이 없으면 자동으로 column 방향 계산으로 하기 때문에 위의 예제에서는 생략이 되었습니다.

표준 편차를 계산하는 jMath.prototype.std(isPopulation,dir)는 호출시 입력값이 없으면 isPopulation 은 false, dir 은 1 로 되어 sample 표준편차에 column 방향으로 계산이 됩니다.

함수	결과값(jMath object)
mean()	[6367.555555555556 921.2222222222222]
std()	[383.2020064897596 189.23383535838522]

이 결과 값으로 각각의 element 별로 계산을 하기 위해서 [‘./’]으로 계산을 하여 CV 값을 얻을 수 있습니다.

2.4. z-Score

개별 값들에 대한 평균과의 거리를 평균거리(표준편차)에 비율로 값을 측정하는 것입니다.

Population	Sample
$z = \frac{x - \mu}{\sigma}$	$z = \frac{x - \overline{x}}{s}$

(3.8)

이 z-Score 는 분자(numerator)의 단위와 분모(denominator)의 단위가 같기 때문에 단위가 없습니다.

이러한 z-Score 는 평균거리 대비 데이타 값의 평균으로 부터의 거리를 알려주기 때문에 거리 절대값이 클 수록 평균로부터 멀리 떨어져 있다라는 의미입니다. 일반적으로 z 값이 ±3 보다 큰 값이면 해당 데이타 값은 outlier 에 속합니다.
다른 중요한 특징은 이 값은 CV 처럼 평균과 표준편차가 다양해도 단위가 없기 때문에 서로 다른 분포의 데이타를 비교하는데 유용합니다.

철판을 200mm 단위로 자르는 Case 1 을 갖고 z-Score 를 계산하면 다음과 갔습니다.

```
> a = jMath([192,198, 199,199, 197, 201, 202, 203, 197, 196, 204, 206, 207, 202,
199]);
> a.zscore(3);
[-2.0345447325074204, -0.5336510773789941, -0.28350213485758974, -
0.28350213485758974, -0.7838000199003984, 0.21679575018521902,
0.46694469270662337, 0.7170936352280277, -0.7838000199003984, -
1.0339489624218028, 0.9672425777494321, 1.4675404627922408, 1.7176894053136451,
0.46694469270662337, -0.28350213485758974]
```

결과를 보시면 sample data 의 z-Score 가 음수이면 평균보다 작다는 뜻이고 양수이면 평균보다 크다는 뜻입니다. 평균으로 부터 가장 멀리 떨어진 192 가 z-Score 값이 -2.03 으로 outlier 에 속하지는 않습니다.

2.5. Empirical rule

경험상으로 만일 data 분포가 평균값을 중심으로 대칭으로 존재 한다면 몇 %의 데이타 값은 어떤 z-Score 범위 내에 존재할거라는 예측입니다.

z-Score 범위	%
-1 ~ 1	68%
-2 ~ 2	95%
-3 ~ 3	99.7%

z-Score 로 부터 실제 값을 얻는 방법은 다음과 같습니다.

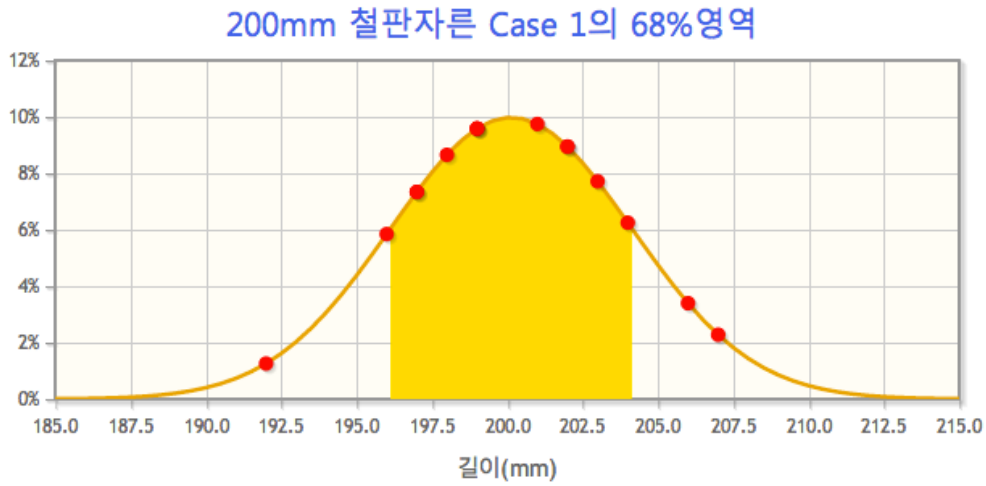
Population	Sample
$x = \mu + z\sigma$	$x = \overline{x} + zs$

(3.9)

(3.9)수식이 의미하는 바는 데이타가 평균을 중심으로 대칭으로 있다면 평균으로 부터 3 배의 표준편차내에 모든 데이타가 있다는 의미 입니다.

철판자른 길이에 대한 Case 1 의 z-Score 범위를 철판 길이로 환산을 하면 다음과 같습니다.

z-Score 범위	값의 범위(mm)	%
-1 ~ 1	196 ~ 204	68%
-2 ~ 2	192 ~ 208	95%
-3 ~ 3	188 ~ 212	99.7%



총 15 개 자료 값 중 11 개가 z-Score 범위 -1 에서 1 사이에 존재를 합니다. 즉 73%가 이 사이 있고 empirical rule 에 68%범위 이상이기 때문에 성립은 되지만 데이타 분포가 이 rule 에 적합하지 않은다면 유효하지 않을 수 있습니다.

2.6. Chebyshev's Theroem

Empirical rule 은 벨모양의 평균을 중심으로 대칭을 이루고 있을 경우에만 적용이 되는 반면 z-Score 로 어느 유형의 분포에도 최소 데이타 값의 퍼센트를 얻기 위한 수식입니다.

$$\left(1-\frac{1}{z^2}\right) \tag{3.10}$$

이 수식을 적용하기 위해서는 z 값은 1 보다 커야 합니다.
데이타 값의 퍼센트가 다음과 같습니다.

z-Score 범위	최소 %
-2 ~ 2	75%
-3 ~ 3	89%
-4 ~ 4	94%

2014 년 1 월 부터 3 월까지 US dollar 환률 기준값으로 이 내용을 확인해 보겠습니다.

2014 년 1 월

1050.3, 1055.2, 1065.4, 1068.3, 1064.9, 1062.9, 1061.4, 1056.7,

1059.1, 1062.7, 1063.4, 1059.7, 1063.7, 1065.3, 1067.4, 1073.9,
1080.4, 1083.6, 1081.2, 1070.4,

2014 년 2 월

1084.5, 1083.8, 1077.9, 1079, 1074.3, 1071.2, 1071.1, 1062.4,
1066.4, 1063.7, 1060.5, 1065.7, 1065.5, 1072.2, 1072, 1074.5,
1072.9, 1065.4, 1068.8, 1067.5

2014 년 3 월

1070.2, 1073.5, 1070.9, 1064.1, 1062, 1066.5, 1065.1, 1070.4, 1069,
1072.8, 1067.4, 1069.2, 1070.5, 1076.2, 1081, 1077.8, 1079.4, 1075,
1071.5, 1069.3

환률의 평균값은 1069.28 이고 표준편차는 7.2767 로 z-Score 가 ±2 인 범위에 값은 1054.73~1083.837 입니다. 이 값 범위 안에 있는 날짜의 수는 60 개중 58 개로 97%에 해당하는 값이 이 범위안에 포함됩니다. 따라서 수식(3.10)으로 얻은 최소 퍼센트가 75%이므로 이 수식이 성립을 한다는 것을 알 수 있습니다.

2.7. Group Data

데이타의 값이 범위로 나타날 경우 평균과 표준 편차를 구하기 위해서는 범위에 대한 조작이 요구 됩니다. 예를 들어 1595 명을 대상으로 1 년 동안 남여 연령별 영화 관람수를 조사 하였는데 조사된 값이 개개인의 나이로 하지 못하고 연령 범위로 나이를 조사를 하게 되었습니다.

	15-18 세	19-23 세	24-29 세	30-34 세	35-39 세	40-49 세	50-59 세
남자	133	147	169	119	90	75	72
여자	138	126	140	120	96	89	81

이 데이타에서 평균 연령을 얻기 위한 방법으로 연령의 범위의 중간 값을 이용하여 앞에서 배운 방식으로 적용하여 수식(3.2)에 weighted mean 계산 방식을 적용하면 됩니다.

Population	Sample	(3.11)

$\mu \approx \frac{\sum_{i=1}^{N} f_i m_i}{\sum_{i}^{N} f_i}$	$\bar{x} \approx \frac{\sum_{i=1}^{n} f_i m_i}{\sum_{i}^{n} f_i}$

여기서 m 은 범위의 중간 값이고 f 는 조사된 횟수 정보 입니다. 여기서 =를 사용하지 않고 ≈을 사용한 이유는 m 이 중간 값으로 결과가 정확한 값이 아닌 대략적인 값이기 때문입니다.

	16.5	21	26.5	32	37	44.5	55.5
남자	133	147	169	119	90	75	72
여자	138	126	140	120	96	89	81

이 값을 통해서 남여 영화 관람 평균 연령에 대한 mean 을 구하면 다음과 같습니다.

```
> male=jMath('16.5 13.3; 21 14.7; 26.5 16.9; 32 11.9; 37 9; 44.5 7.5; 55.5 7.2');
> female=jMath('16.5 13.8; 21 12.6; 26.5 14.0; 32 12; 37 9.6; 44.5 8.9; 55.5
8.1');
> male.wmean()
[30.101242236024845, 805]
>female.wmean()
[30.98860759493671, 790]
```

결과에서 보면 여성의 영화 관람 평균 나이가 남성에 비해 대략 1 살 정도 많은 것으로 나타납니다. 이 결과는 평균값이기 때문에 값들의 밀집도를 알 수가 없는 상태입니다. 이를 위해서 standard deviation(표준편차)를 알아 보겠습니다. 표준 편차를 계산하는 방식은 횟수를 이용하기 때문에 수식에 변화가 있습니다.

Population	Sample
$\sigma^2 \approx \frac{\sum_{i=1}^{k} f_i (m_i - \mu)^2}{\sum_{i}^{k} f_i} = \frac{\sum_{i=1}^{k} f_i (m_i - \mu)^2}{N}$	$s^2 \approx \frac{\sum_{i=1}^{k} f_i (m_i - \bar{x})^2}{\left(\sum_{i}^{k} f_i\right) - 1} = \frac{\sum_{i=1}^{k} f_i (m_i - \bar{x})^2}{n - 1}$

(3.12)

여기서 k 는 Group 의 개수 입니다.

앞의 영화 관람의 연령에 대한 편차를 구하면 다음과 같습니다.

```
> male=jMath('16.5 13.3; 21 14.7; 26.5 16.9; 32 11.9; 37 9; 44.5 7.5; 55.5 7.2');
> female=jMath('16.5 13.8; 21 12.6; 26.5 14.0; 32 12; 37 9.6; 44.5 8.9; 55.5
8.1');
> male.wstd()
[11.464904199585645, 805]
```

```
>female.wstd()
[11.960321889748876, 790]
```

.wstd()함수는 입력값이 true 이면 population 표준편차를 계산하고 입력 값이 없거나 false 이면 sample 표준편차를 계산을 합니다.

3. 데이타 위치 검색

Median 은 데이타의 중간 위치에 있는 값으로 이것을 %로 나타내면 50%입니다. 이렇게 데이타를 정렬하여 값의 위치를 %로 나타내어 해당 위치에 있는 값을 percentile 이라고 하고, 역으로 어떤 값이 몇 번째 %에 있는지를 찾는 것을 percentile rank 라고 합니다. 이 두개의 값을 구하는 방법은 간단한 방법이 있지만 Microsoft Excel 과 같은 결과를 만들기 위해서 Excel 에서 채택된 방식으로 소개를 하겠습니다.

3.1. Percentile

1. 데이타를 오름 차순으로 정렬을 합니다.
2. $i = p \times (N - 1)$
3. i 값에서 정수 부분(idx)과 소수부분(decimal)을 분리를 해서 값을 구합니다.

$$v = list[idx] + decimal \times (list[idx + 1] - list[idx])$$

여기서 p 는 위치 %이고, N 은 데이타 개수입니다. 2 에서 i 값의 정수 부분의 값은 data 가 나열된 값에서 idx+1 번째 말합니다.

예를 들어 데스크탑 컴퓨터에 사용할 메모리의 가격을 검색을 했을 때 10 개의 다른 가격은 다음과 같이 정렬이 되어 있습니다.

78230 78660 79210 80140 80340 80940 82000 83000 85000 88700

이 값에서 40%에 위치한 값을 찾는다고 한다면 i 값은 0.4x(10-1)로 3.6 됩니다. 여기서 정수는 3 이고 0.6 가 소수부분입니다. 그럼 이 데이타의 40%에 percentile 값은 80260 원이 됩니다.

list[3] + 0.6 x (list[4]-list[3]) = 80140+ 0.6 * (80340-80140) = 80260

3.2.Percentile Rank

역으로 값을 넣어 값이 어느 정도 순위(rank)에 있는가를 판단하는 percentile rank 를 구하는 방식은 다음과 같습니다.

1. 데이타를 오름 차순으로 정렬을 합니다.
2. 만일 입력값이 최소값보다 작거나 최대값 보다크면 계산 불능으로 처리를 하지 않습니다.
3. 입력 값보다 처음으로 큰 값이 나타난 zero-based index 값에 -1 을 한 값을 idx 로 얻습니다.
4. $i = idx + \frac{v - list[idx]}{(list[idx+1] - list[idx])}$
5. 순위 % = $\frac{i}{N-1}$

앞의 예제에서 값 80000 원의 percentile rank 를 보시면 idx 는 79210 위치인 index 2 입니다. 그럼 i 값은 2 + (80000-79210)/(80140-79210)로 i 는 2.849 가 됩니다. 따라서 percentile rank 는 2.849/9 인 0.3166 이 됩니다. 즉 31.66%순위에 있는 값이 됩니다.

3.3.Quartiles

percentile 이 0(최소값), 25%(Q1), 50%(Q2), 75%(Q3), 100%(최대값)을 구하는 것입니다. 메모리 가격으로 quartiles 을 구하면 다음과 같습니다.

Min	78230
25%	79442.5
50%	80640
75%	82750
Max	88700

Interquartile range(IQR)이라고 하여 25%(Q1)와 75%(Q3)사이에 값의 범위를 말합니다.

$$IQR = Q3 - Q1 \tag{3.13}$$

메모리 가격으로 부터 IQR 은 82750 – 79442.5 로 3307.5 가 됩니다. 이 IQR 을 이용하여 Outlier 를 구하는 방법은 다음과 같습니다.

$$\begin{aligned} \text{Upper Limit} &= \text{Q3} + 1.5(\text{IQR}) \\ \text{Lower Limit} &= \text{Q1} - 1.5(\text{IRQ}) \end{aligned} \tag{3.14}$$

jMath 사용법

```
> obj = jMath('78230 78660 79210 80140 80340 80940 82000 83000 85000 88700');
> obj.percentile(0.4)
80260
> obj.percentrank(80000)
0.31660692951015534
> obj.quartiles()
[ 78230 79442.5 80640 82750 88700 ]
```