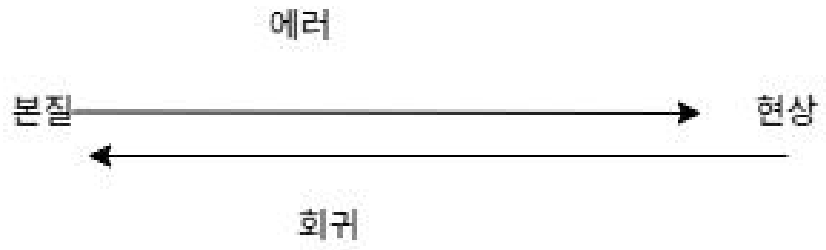
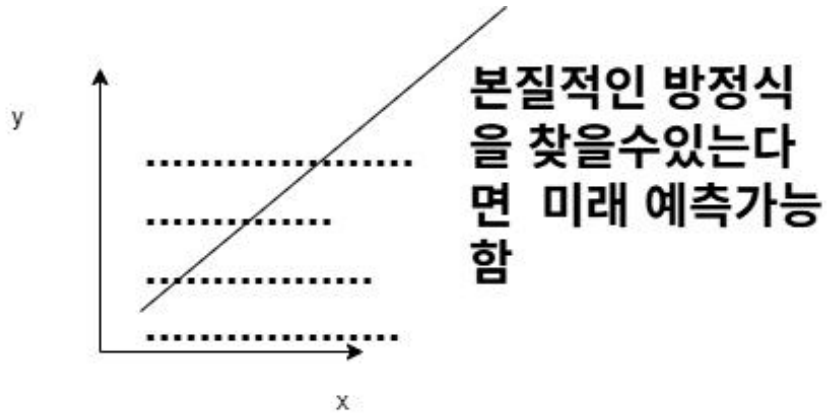
단순회귀

우리세상에서는 본질적인게있다 그 본질적인게 현상을 밝히는데 에러가 생기면 현상데대해서는 불확실한것이된다 그래서 어떤방법을 통해서 본질을 알아가는게 그게 회귀이다.

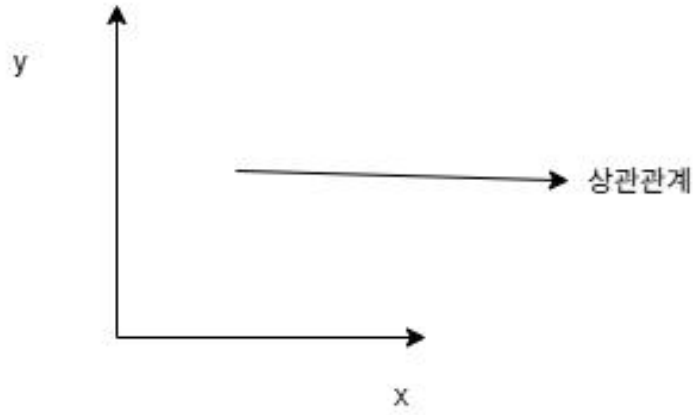


본질에서 y=ax+b는 공학적으로 가져오면 아름다운 방정식이된다.



일차로 되어있는 거는 단순회귀 2차 3차 지수로 되어있는거는 컴퓨터가한다.
저번에서는 x,y데이터를 동등하게 봤지만 실제로 많은경우에는 x가원인이고 y가 결과이다.x가어떻게 되면 y는 어떻게된다 이런식의현상 요사이의 관계가 있다.
y=ax+b 서비스친절도가 올라가면 매출은 얼마오를 것이다.그리고 여기에 지저분한 에러가 들어가있다. y=ax+b+e a는 기울기 b는 y절편 b는 bias라고 하지만 사실은 중요하지않다. 데이터의성질이나 크기로인해서 나타남, 중요한 것이 아니다. 핵심은 a하고 b를 알아야한다.애러는 알필요가없다.

우리가 이거를 구하는 연습을 해야하는데 a하면 코베리언스



cov(x,y)=cov=(x,ax+b+e)=cov(x,ax)+cov(x,b)+cov(x,e)
하나하나씩 가져감 바이어스하고 x랑은 아무상관이없다.
크기차이일뿐이다. 하나가지나가는데 하나가또지나가는데에는 bias가아니다
하나의 변수를 더많이 넣어야한다 제트를 넣거나 x가증가한다고 에러가
증가하는것도아니다 그거는에러가아니다.
=acov(x,x)
=avar(x)
a=cov(x,y)/var(x)
비는 0이기만해도 바이어스
에러의 평균은 0이되어야한다 그러를 화이트 노이즈라고한다.
에러가 0이아니면 작위적으로 문제가있다.
양쪽평균을 구해야한다. E(y)=E(ax)+E(b)+E(e) 일정해서 b가 0이된다.
b=E(y)-aE(x)
=E(y)-cov(x,y)/var(x)E(x) E(x)는 평균을 의미함
그래서 단순회귀를 식으로 바꾸면
y=cov(x,y)/var(x)*x+E(y)-cov(x,y)/var(x)*E(x)

```
[7]: import numpy as np # 넘파이
     import pandas as pd # 판다스
     import matplotlib.pyplot as plt #매트립
```

```
[9]: data=pd.read_csv("test2.csv") # csv불러오가
     data
```

[9]:

	height	weight
0	170	65
1	160	55
2	180	70
3	170	60
4	175	62
5	180	72
6	160	60
7	165	58
8	186	100
9	172	67

```
[11]: height=data['height'] # height 에있는 데이터만
      weight=data['weight']  # weight 에 있는 데이터만
```

```
[13]: height
```

```
[13]: 0    170
      1    160
      2    180
      3    170
      4    175
      5    180
      6    160
      7    165
      8    186
      9    172
      Name: height, dtype: int64
```

[15]:
```
weight
```

[15]:
```
0     65
1     55
2     70
3     60
4     62
5     72
6     60
7     58
8    100
9     67
Name: weight, dtype: int64
```

[17]:
```
height=np.array(height) # 배열로 변경
weight=np.array(weight) # 배열로 변경
```

[19]:
```
height
```

[19]:
```
array([170, 160, 180, 170, 175, 180, 160, 165, 186, 172], dtype=int64)
```

[21]:
```
weight
```

[21]:
```
array([ 65,  55,  70,  60,  62,  72,  60,  58, 100,  67], dtype=int64)
```

[23]:
```
cov_matrix=np.cov(height,weight) # 공분산행렬로
cov_matrix
```

[23]:
```
array([[ 75.28888889,  91.08888889],
       [ 91.08888889, 163.87777778]])
```

[25]:
```
np.corrcoef(height,weight) # 상관계수
```

[25]:
```
array([[1.        , 0.82004923],
       [0.82004923, 1.        ]])
```

[29]:
```
cov=np.cov(height,weight)
height_mean=np.mean(height)
weight_mean=np.mean(weight)
height_var=np.var(height)
cov=cov[0,1]
cov
```

[29]:
```
91.08888888888889
```

•[35]:
```
plt.scatter(height,weight) # 점들
plt.xlabel("height")
plt.ylabel("weight")
x2=np.linspace(160,185)
y2=cov/height_var*x2+weight_mean-cov/height_var*height_mean # y=ax+b
plt.plot(x2,y2,'r') # 단일차트
plt.show() # 보여주기
```