Question

~~1. 잔차가 왜 오차를 대표~~

-> beta를 잘 추정하면 잔차와 오차가 매우 비슷해지기 때문에

~~2. 예측된 값의 가중평균으로 최종 프레딕트 가능~~

ex) Lasso에서 예측된 값, Ridge에서 예측된 값의 가중평균으로 final predicted\_value 구함

3. 로지스틱에서 회귀계수의 해석

~~4. lasso 회귀에 대한 목적함수~~



(회귀계수에 절댓값 씌운 것의 합)

5. 행렬미분 복습

~~6. 잔차로 오차항의 모분산 추정~~

-> 잔차제곱합/SSE의 자유도(n-p-1)

~~7. 더빈왓슨~~

-> 잔차의 독립성 가정을 충족하는 지 확인할 수 있는 통계량



(2에 가까울수록 무상관, 0에 가까울수록 양의 상관, 4에 가까울수록 음의 상관)

~~8. 오즈 오즈비 정리~~

<https://nittaku.tistory.com/478>

<http://statistics4everyone.blogspot.com/2016/02/blog-post_7.html>

odds = p/1-p

예를 들어 odds=0.25 -> 5번 중에서 1번 이기고, 4번 진다.

logit(p) = log(odds)=log(p/1-p) : -무한대 ~ +무한대 범위

odds ratio = odds\_1 / odds\_2 (오즈의 비)

ex) odds(A) = 0.61 (p=0.38), odds(B) = 1.7 (p=0.63)

-> B에 대한 A의 odds ratio = 0.61/1.7 = 0.36 (B에 비해 A일 때 성공확률 0.36배)

(B보다 A일 때 성공확률이 더 낮아지는 거지)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| A | p=0.2 | p/1-p = 1/4 | 1/4 = 7/28 |  | B에 비해 A일 때 성공할 확률 7/12배 |
| 1-p=0.8 |
| B | p=0.3 | p/1-p = 3/7 | 3/7 = 12/28 |
| 1-p=0.7 |

~~9. design matrix에서 column의 수가 늘어나면 분산이 커지는 문제가 있던가?~~

그건 아니고, correlated 되어있는 설명변수의 수가 많아지면

X에서의 min(eigenvalue)가 0에 가까워짐

-> np.dot(X.t, X)의 행렬식이 0에 가까워 짐(determinant = eigenvalue의 곱)

-> inv(np.dot(X.t, X))의 값이 매우 커짐

-> 회귀계수의 분산이 커짐

회귀분석

회귀분석의 목적?

반응변수 평균에 설명변수가 얼마나 영향을 끼치는 가 확인

여기서 오차항은 분산일정 + 정규분포 + 평균이 0 + iid 가정을 지닌다.

회귀분석에서 도출된 회귀계수가 0이라고 해서 설명변수와 반응변수 사이 아무런 관계가 없는 것은 아니다. (단순 직선적인 관계가 없다는 의미일 뿐, 2차 혹은 다른 관계를 지닐 수 있다.)

회귀계수를 추정하는 방식은 OLS방식과 projection 방식이 있다.

OLS 방식은 (실제-기댓값)의 제곱합을 minimize하는 회귀계수를 찾는 것

(각 회귀계수에 대해서 편미분하여 0이 되는 지점을 찾는다)

OLS방식으로 구해진 회귀계수에서 나온 잔차는 각 설명변수와 직교하게 된다.

정사영 방식은 y 벡터를 col(design matrix) 위로 projection한 벡터가 y\_hat이 되는 것

**잔차가 왜 오차를 대표**

r\_i = y\_i – yhat\_i

= (beta\_0 + beta\_1\*x\_i + eps\_i) – (betahat\_0 + betahat\_1\*x\_i)

= (beta\_0 – betahat\_0) + (beta\_1-betahat\_1)\*x + eps\_i

**∴ beta\_0-betahat\_0 ≒ 0, beta\_1-betahat\_1≒ 0 -> r\_i ≒ eps\_i**

따라서 beta를 잘 추정하면 잔차와 오차가 매우 유사할 것이다.

따라서 잔차를 이용해 오차의 분산을 추정한다.

2개의 feature를 뽑아서 scatter plot을 보고, 관계를 나타내기 위해 regression line를 그려보는데,

어떻게 해야 데이터에 가장 적합한 regression line을 그릴 수 있을까?

1. x가 변함에 따라 y가 어떻게 변하는 지 알 수 있어야 함

2. 수직거리의 제곱의 합을 minimize해야 한다.

3. b\_0은 intercept를 b\_1은 기울기(slope)를 의미한다.

6p

x가 변함에 따라 y가 어떻게 변하는 지 보고자 한다.

x=50를 넣었을 때 회귀선에 대입하여 predicted\_y를 얻는다.

7p

s\_i : 오차를 의미한다.

모든 데이터셋이 직선위에 있을 수 없다. 단지 오차를 줄이는 line을 찾는 것

9p

b\_0 : x=0일 때 y의 평균적인 값

x의 범위가 0와는 멀리 떨어져 있다면 해석적인 의미보다는 회귀선에서 intercept로서의 의미만 갖음(전체적인 그래프의 위치를 조절한다) (외삽을 주의하자)

10p

b\_1 : 기울기

그렇다면 beta를 어떻게 추정할까?

실제 y와 예측된 y 와의 거리제곱합을 minimize하는 beta를 찾자.

12p

미분을 하기 위해 절댓값이 아닌 square를 취한 것

단순선형회귀에서 우리가 minimize 해야하는 것은 2차식 (잔차제곱합 <- beta의 관점에서 바라봄)

잔차제곱합을 각 beta에 대해서 미분한 것이 0이 되는 지점을 찾자

14p

잔차 = 실제값(Y) – 예측값(Y\_hat)

잔차가 큰 것 -> unusual observation (이상치?)

잔차가 큰 지점에 의해 추정된 회귀계수의 변동이 심해짐 (high leverage)

15p

loss function = 잔차제곱합

16p



설명변수 k (intercept를 제외한 설명변수의 수)

SSE = 잔차제곱합 = (실제값-예측값)의 제곱합, df = n-k-1

SSR = (예측값-평균값)의 제곱 합, df = k

SST = (실제값-평균값)의 제곱 합, df = n-1

SSR/SST = R\_square (회귀선에 의해 설명되는 변동의 비율)

p19.

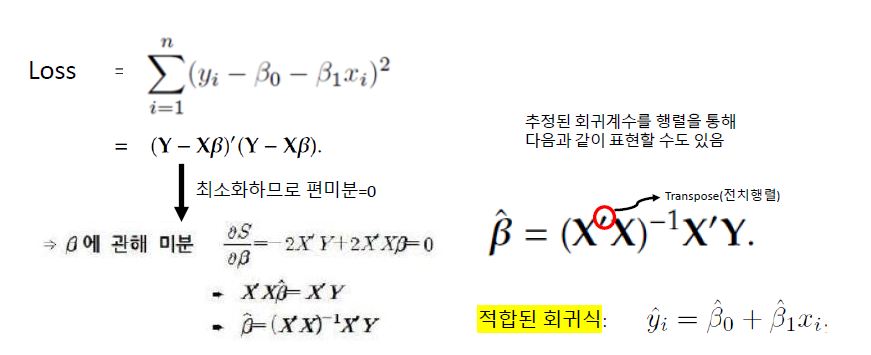
다중선형회귀에서 하나하나 미분하기는 힘들다 ☹

-> 행렬을 이용하자

y = np.matmul(X, b\_hat)

SSE = np.matmul((y-np.dot(X, b\_hat)).T, (y-np.dot(X, b\_hat)))

SSE를 b에 대해서 미분한 것이 영벡터가 되어야 한다.



X matrix만 달라질 뿐 계산 과정은 동일하다

data의수 : n

Y : n\*1

X : n\*(p+1)

b : (p+1)\*1

b\_추정 = inverse(X’X)X’y



p26.

SST = SSR + SSE



R\_square = SSR/SST = 1-SSE/SST



adjusted\_R^2 = 1-(SSE/n-p-1)/(SST/n-1)

R^2, adjusted\_R^2는 클수록 좋음



AIC, BIC는 설명변수의 개수에 대해 penalty 부여 -> 작을수록 좋음

(설명변수의 개수가 많을수록 AIC, BIC값이 커짐)

p30.



여기서 sigma -> 오차의 모분산

근데 우리 오차의 분산 모름 -> 잔차를 이용해 오차항의 분산 분산을 추정하기 때문에

t분포를 이용해 회귀계수의 신뢰구간 구하는 것 같음

오차의 모분산 추정 = 잔차제곱합 / (n-k-1) \* (n-k-1 : SSE의 자유도)

MVN(multi variable normal)

beta는 정규분포 따른다 (왜냐하면 오차항이 정규분포 따르기 때문에)

추정된 beta의 분포에 따라 p-value 구해진다.

std.err : 추정된 회귀계수의 표준편차

p-value에 따라 개별 회귀계수가 유의한 지 유의하지 않은 지 판단

**marginal, joint 이 부분 개인보충**

p-value는 개별 회귀계수가 유의한 지에 대해서 판단

하지만 2개 이상의 parameter의 joint confidence region은 elliptical(타원형)이지 rectangular(직사각형)이 아니다.

일부 회귀계수가 0인지 아닌지 테스트 -> 부분 F 검정

p31.

<가정> 잔차들의 독립성 -> DW 통계량을 통해 확인

error는 독립이라는 가정을 잔차를 통해 확인

(+ 잔차에 패턴 보이면 반응변수를 설명하는 feature가 누락된 것이라고 해석)

(+ 잔차의 정규성 판단하는 방법? -> QQplot)



앞의 잔차와 다음의 잔차가 얼마나 연관성이 있는 지 나타내주는 통계량

(혹시 다다음 잔차와 연관성이 있지 않을까? -> e\_i – e\_i-2 이런식으로 식 변경해서 사용)



잔차의 부호의 변화를 통해서도 독립성을 확인해볼 수 있다. (run test라고 함)

(1) +++++/-----/+++/--- -> run의 수 : 4 양의 상관

(2) ++/--/+/-/++/-/+/--/+/--/+ -> run의 수 : 11 음의 상관

(3) ++/-/+/--/++/--/+/--/++ -> run의 수 : 9

(run의 수가 너무 적으면 양의 상관, run의 수가 너무 많으면 양의 상관이 있다고 판단)

p34.

범주형 변수 -> 숫자로 (단, 특성의 수보다 1개 적게 설정하도록 한다)

**하나의 범주를 baseline으로 잡고, 범주의 수보다 1개 적은 feature를 만든다.**

**(-> 다중공선성 문제를 막고자 이렇게)**

(남, 여) -> (0, 1)

(초등, 중등, 고등) -> 초등 = (0, 0), 중등 = (1, 0), 고등 = (0, 1)

이렇게 dummy variable을 사용할 때는 회귀계수의 해석이 달라진다.

‘이전까지는 나머지 설명변수는 일정하고, 특정 설명변수가 1단위 증가할 때 y는 평균적으로 beta만큼 변화’로 해석했으나, 이제는 ‘baseline에 해당하는 집단 대비 이 집단의 경우 y는 평균적으로 beta만큼 변화’로 해석한다. (=즉 집단 간 평균차이로 해석)

설명변수의 곱 (교호작용)

범주형변수 \* 범주형변수 -> intercept의 차이

범주형변수 \* 연속형변수 -> slope의 차이

p37.

선형회귀분석의 가정 (+보충)

1. 선형성 (X, Y가 선형관계)

- 설명변수 1, 반응변수 1 -> scatter plot (선형형태인 지 확인)

non-linear 형태라면 다양한 변환(log, exp, sqrt, n제곱 등)을 수행하여 선형으로 변환

- 설명변수 여러 개, 반응변수 1개 -> 각각에 대해서 선형 아니라고 전체적으로 선형 아니라고 판단하기는 어렵다. (다차원에서 graphic 해석은 어렵다)

2. 독립성 (설명변수끼리 독립) -> X 가 full rank matrix

- DW 테스트

0<dw<4 : 0에 가까울수록 양의 상관, 4에 가까울수록 음의 상관, 2에 가까울수록 독립

(한 시점 전과 얼마나 종속되어 있는지 확인해줌)

eps\_t = p\*eps\_(t-1)+w\_t, w\_t ~ iid (0, sigma^2)

p\_hat =



d ≒ 2(1-p\_hat)

3. 등분산성 (오차항이 동일한 분산)

- 잔차 plot을 그려보았을 때 pattern이 발생한지는 않는지 혹은 퍼져나가는 형태로 그려지지는 않는지

- 분산이 다른 것을 추정에 반영 -> WLS(weighted least square)

(분산이 클수록 weight를 작게, 분산이 작을수록 weight를 크게)

(여러 개의 설명변수를 사용할 경우 -> X축은 y\_hat으로 설정하도록 한다.)

4. 정규성 (오차항이 정규분포 따른다)

- QQ plot 그려서 직선위에 위치하는 지 확인

+ 선형회귀는 outlier에 매우 민감하기 때문에 이 outlier를 detect하는 것도 중요하다.

2이상의 표준화된 잔차가 나타나는 관측치는 outlier라고 판단하는 것이 일반적

(leverage, cook’s distance 사용)

\* rank? 일차독립인 열벡터 혹은 행벡터의 max 수)

다중공선성? 어떤 설명변수를 반응변수, 나머지 설명변수를 설명변수로 두고 회귀분석 했을 때의 R\_2의 값이 큰 경우 다중공선성 문제가 있다고 판단

VIF = 1/1-R^2

다중 공선성 문제 -> np.matmul(X.T, X)에서 min(eigenvalue)가 0에 가까움

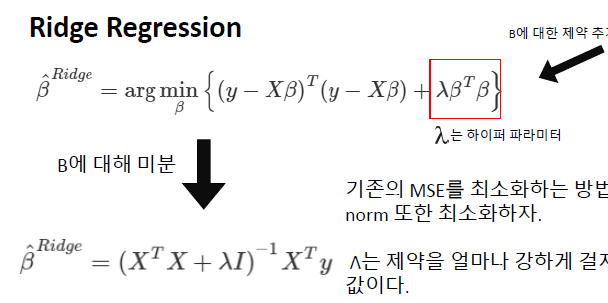
-> determinet = eigenvalue의 곱

-> inv(np.dot(X.t, X))가 커짐

-> OLS 추정량의 변동 커짐

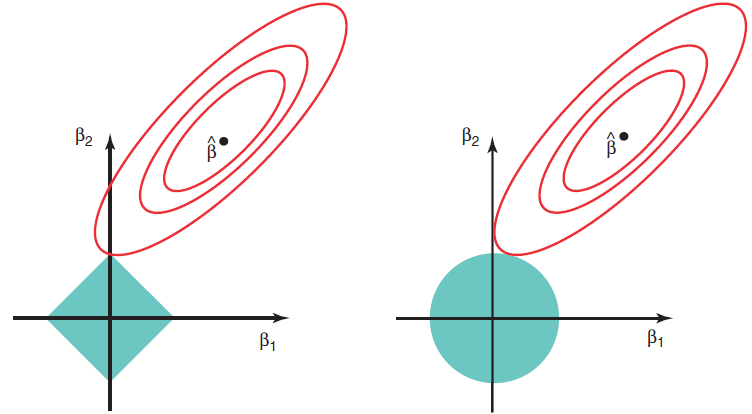
+ 다중공선성 문제는 회귀계수의 부호를 역전시킬 수 있다. (회귀계수의 왜곡)

다중 공선성 해결? -> feature selection, ridge regression, PCA



rambda\*I를 더해주어 eigenvalue를 강제로 키워줌 -> determinat 덜 작아짐 -> 덜 분산 커짐

rambda\*beta.T\*beta -> 원형의 모양



ridge



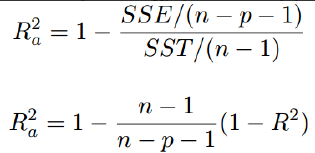
lasso



(회귀계수에 절댓값 씌운 것의 합)

**model의 선택기준**

1) adjusted R\_2 – 설명변수의 개수에 대해 페널티 (p크면 Adjusted\_R2 작아지도록)



2) forward selection, backward elimination, stepwise

이 과정에서 AIC, BIC를 지표로 사용한다.



모델에서의 적합도가 높아지면 SSE 낮아짐 -> AIC, BIC 낮아짐

모델에서 설명변수의 개수 늘어남 -> AIC, BIC 커짐 (설명변수의 개수에 대해서 penalty 부여)

(AIC가 작아지는 방향으로 feature를 넣었다가, 제거했다가 함)

전진선택법, 후진제거법의 단점

한 번 제거되거나 추가된 변수는 모델에 계속 남아있게 된다는 것

-> 이것의 단점을 보완하고자 단계적 선택법

기억해야 할 것은 forward selection, backward elimination, stepwise selection 방식으로 구한 model이 최적의 model은 아니다.

가장 best는 모든 model을 고려해보는 것이나, 계산량이 많기 때문에 이 방법을 사용하는 것

로지스틱 회귀

로지스틱 회귀는 target이 범주형일 때 사용

Y의 값에 대해서 모델링을 하는 게 아니라 P(Y=c)를 모델링한다.

1. 선형

P(Y=c) = ax + b

odds는 확률이 아니라, ‘성공확률/실패확률’으로 정의한다.

log(odds) : 로짓변환

양변의 range 동일하게 : 로짓변환 수행하면 -무한대 ~ +무한대의 범위를 갖게 된다.

# 2

**likelihood가 확률과 동일하나 시점을 다르게 한 것**

likelihood는 데이터가 주어졌을 때 어떤 결과가 일어날 가능성을 나타냄

(데이터를 알 때 모수 theta가 나올 정도라고 이해하면 된다)

joint distribution은 iid 가정 하에서 확률의 곱으로 나타냄

\* iid(identically independent) : 독립적으로 동일한 확률분포를 따른다.

likelihood는 확률과는 다른 것 (확률분포 그 자체에서의 값)

MLE를 이용해 회귀계수 추정

**MLE는 likelihood를 최대화하는 parameter를 찾는 것**

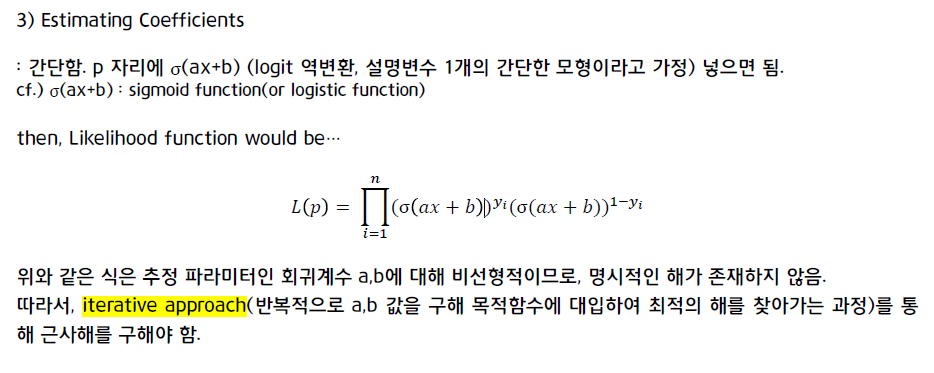
MLE를 미분해서 0이 되는 지점을 찾는 것 (여기서 계산의 용이성을 위해 log-likelihood를 사용하기도 함)

14p

logit( P(Y=c) ) = ax+b를 p에 대해서 전개하면 아래와 같은 식이 도출된다.

p 자리에 sigmoid(ax+b)를 넣으면 된다.

**(sigmoid = 1/(1+exp(-(ax+b))**



명시적인 해가 존재하지 X -> 어떻게 해야할까? 반복적으로 수행

목적함수에 대입 -> 목적함수를 미분 (수정)

최적화? 목적함수를 최대화 혹은 최소화하는 파라미터를 찾는 것

즉 여기서는 우도를 최대화하는 혹은 negative log likelihood를 최소화하는 것!

convex한 형태면 최소화, concave형태는 최대화의 문제라고 보면 되겠다.

목적함수가 항상 미분 가능한 건 아니며, saddle point의 경우 미분해서 0이 되지만 최적화 점은 아니다.

gradient descent는 1차미분을 활용한 최적화 방법

(+ 2차미분을 활용한 최적화 방법 이름이 뭐였지?) -> newton Raphson algorithm

gradient 이용해 minimum point를 찾아서 이동

param\_new = param\_old – learning\_rate \* 목적함수를 param에 대해서 일차 편미분한 값

(여기서 learning\_rate는 hyper-parameter)

\* gradient = multi scalar function의 편미분



gradient descent algorithm의 문제점

1) learning\_rate의 조절의 어려움

overshooting : learning\_rate가 너무 클 때 (어디인지 모르는 곳으로 이동)

learn too slow : learning\_rate가 너무 작을 때

2) 초기 시작점을 어디서 잡느냐에 따라 수렴지점이 달라질 수 있다.

gradient descent algorithm은 모든 데이터에 대해서 계산

stochastic gradient descent algorithm은 하나의 데이터를 잡아서 계산

mini-batch gradient descent algorithm은 mini-batch

연산 시간 단축되며 shooting이 일어나 local minimum에 빠질 가능성이 더 낮다

단점은 global optimum을 찾지 못할 가능성이 있다는 것

model 평가 지표

accuracy, precision, recall, f1

**accuracy는 label의 분포가 imbalance할 때는 좋은 평가지표는 아니다. (성능 평가 왜곡)**

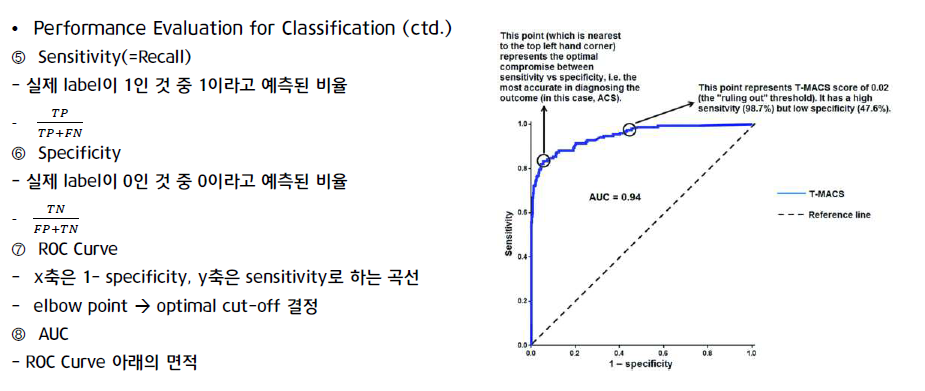
99명의 사람은 암X, 1명만 암 O 일 때, 모두 암 X라고 예측하면 accuracy 0.99 나옴 ☹

roc curve -> auc (roc curve 아래의 면적 : 0.5~1의 범위)

sensitivity(=recall=실제 1인 것 중에서 1이라고 예측한 것의 비율), specificity(실제 0인 것 중에 0이라고 예측한 것의 비율) 둘 다 높을수록 좋음

y축에 붙을수록 모델의 성능이 좋은 것

(x축 = 1-specificity, y축 = sensitivity)



elbow point – threshold를 변경해가면서 specificity, sensitivity의 값을 찍은 것인데,

꺾이는 지점이 optimal cut-off라고 할 수 있다.

stochastic -> random으로 1개의 데이터를 뽑음