3주차 - 회귀분석의 변형

목차

Table of Contents

작성자 : 권남택 Phone : 010-7518-8810 github.com/Namtaek

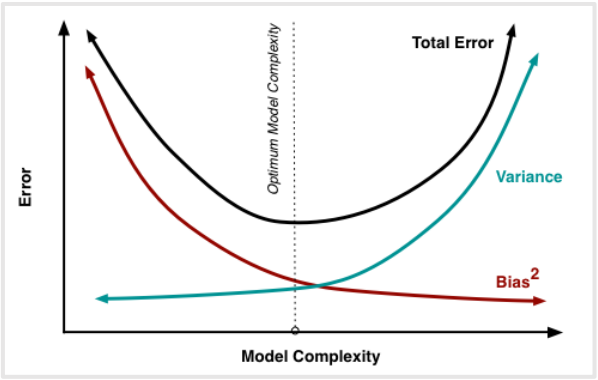
0. 복습

- 회귀분석의 가정은 모델의 선형성, 오차의 등분산성, 정규성, 독립성 네 가지이다.
  - 가정의 확인은 잔차 플랏으로 확인할 수 있지만, Test를 통해 더 객관적인 기준을 더할 수 있다.
  - 모델의 선형성이 깨지면 모델을 과소추정 (underestimate) 했다는 뜻이고, 고차항을 추가할 수 있다.
  - 오차의 등분산성, 독립성이 깨지면 LSE가 최소분산을 갖지 않게되고, 검정결과를 신뢰할 수 없게 된다. 이를 해결하기 위해서는 등분산은 WLS, 독립성은 시계열 모델링을 고려하게 된다.
  - 오차의 정규성이 깨질 경우 예측 성능에 큰 문제가 생긴다. 이를 해결하기 위해 Y변수에 대한 변환을 시행한다.
  - 회귀분석의 가정은 꼭 지켜져야 하지만, 데이터가 충분히 많을 경우, 가정 위배에 따른 모델의 분산 증가를 완화할 수 있다.
- 다중공선성을 회귀분석의 결과를 위태롭게 한다.
  - 다중공선성이 있을 경우 회귀계수의 분산이 매우 커져서, 전체 회귀식이 유의함에도 개별 회귀계수들이 유의하지 않게 된다. 그에 따라 모델의 분산이 너무 커져 Prediction Accuracy에도 문제가 생긴다.
  - 다중공선성은 상식과 벗어나는 회귀 결과값을 통해 짐작할 수 있다.
  - 변수간의 상관계수가 0.7이 넘어갈 경우와 VIF가 10을 넘을 경우 다중공선성이 크게 의심된다.
  - 다중공선성을 해결하는 세 가지 방법으로는 변수선택법, 차원축소, 축소추정량이 있다.
  - 회귀 3주차의 핵심은 Convex Optimization & Shrinkage Method이다. 변수선택법의 중요도를 매우 낮출 것이고, 그 이유를 명확하게 설명하려 한다. 컨벡스 최적화의 관점을 통해 Best Subset Selection을 Lasso로 대체할 수 있다.

1. 변수선택법 (Variable Selection)

(1) 변수선택의 개념

- 우리에게 주어져 있는 가능한 후보 변수(Candidate Regressor)들은 많고, 그에 따라 후보 변수들의 조합도 매우 많다. 하지만 이 중에 일부분만 중요하거나 예측에 유의미할 수 있다. 따라서 우리는 후보 변수들의 적절한 부분집합을 찾으려고 한다.
- 변수선택법은 다중공선성이 존재할 때 많이 사용된다. 물론 변수선택법이 다중공선성의 완벽한 제거를 보장하는 것은 아니지만, 1)높은 상관관계를 가지는 변수들 중 일부만을 선택하도록 해주고, 2)높은 상관관계를 가지는 변수들의 존재를 정당화해줄 수 있다.
  - 더불어 변수선택법은 다중공선성 만의 이유로 사용하지 않는다. 다중공선성이 발견되지 않더라도, 우리 최종 모델에 대한 확신을 얻기 위해 시행할 수 있다.



- 우리가 모델을 만들 때, 두 목적은 충돌한다. 먼저 우리는 최대한 많은 변수들을 사용해서 y를 예측하기 위한 많은 정보를 포함하고 싶기도 하지만, 최대한 적은 변수들을 사용해서 모형의 분산을 줄이고 싶어한다. 이 두 trade-off를 잘 고려해서 '최적의 회귀식 (Best Regression Equation)'을 찾는 방법이 변수선택법이고, 이렇게 구해진 회귀식이 언제나 최적의 회귀식이라는 보장은 할 수 없지만, 우리의 결론을 뒷받침하는 좋은 근거가 된다.

(2) 변수선택의 기준

- Partial F-test를 통한 변수 선택
  - Partial F-test를 통해 변수를 선택할 수 있다. Full Model에서 Null Model로 가거나, Null Model에서 Full Model로 가는 두 방향의 방법을 통해 어떤 변수들이 살아남는지를 확인할 수 있다. 이 방법을 전진선택법과 후진제거법이라고 하는데 뒤에서 다뤄보자. 하지만 이는 오직 비교하려는 두 모델이 Nested되어 있을 때 가능하다.
  - 예를 들어 이런 상황이 있다고 하자.

    $$1: y = \beta_0 + \beta_1 x_1 + \beta_2 x_2, \qquad 2: y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

    2번 모델에서 x_3만 제외하면 1번 모델이 된다. 이렇게 변수들 집합의 포함관계가 성립하는 경우 nested되었다고 표현한다. 이런 경우에는 Partial F-test를 통한 변수선택이 가능하다.
  - 반면에 이런 상황을 가정하자.

    $$1: y = \beta_0 + \beta_1 x_1 + \beta_4 x_4, \qquad 2: y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

    두 모델 사이에는 어떠한 포함관계가 성립하지 않기 때문에, FM과 RM을 비교하는 Partial F-test로는 비교가 불가능하다. 따라서 **포함관계와 무관하게 Global한 모델 간의 비교를 가능하게 해주는 기준이 필요하다.**
- 어떤 방식으로 기준을 선정해야 할까?
  - R^2은 변수개수가 증가함에 따라 자동적으로 증가하는 성질이 있기 때문에 좋은 평가기준이 될 수 없다. ${R_{adj}}^2$도 변수선택의 기준이 될 수 있지만, 이것보다는 다른 방법을 많이 쓴다.
  - 왜냐하면 다른 평가지표가 갖는 성질들이 유용하기 때문이라고 여겨진다. -2log(likelihood)는 GLM에서 적합도(goodness of fit)을 보는 이탈도(Deviance)의 지표이며, -2log(likelihood)의 경우 점근적으로(Asymptotic) 카이제곱분포를 따른다고 알려져 있다. 또한 뒤의 후술할 AIC와 BIC는 어떤 값의 추정치로 이해될 수 있기 때문이다.
  - 변수선택법의 핵심은 적은 변수안에서 데이터를 제일 잘 설명해야 한다는 것이다. 변수가 너무 적으면 간결하고 해석이 쉬워지더라도 예측력은 떨어질 것이고, 변수가 너무 많으면 과적합(Overfitting)될 가능성이 높다. 이 trade-off를 고려한 기준을 만들어야 한다.
  - 당장의 SSE/RSS (Residual Sum of Square)를 충분히 작게 함과 동시에, 변수의 개수에 대한 penalty를 통해 줄이는 방식이 제일 직관적으로 간단해보인다! 여러 기준이 있지만 AIC와 BIC만 보겠다.
- AIC (Akaike Information Criterion)

  $$AIC = -2\log(likelihood) + 2p, \qquad p: \textit{변수갯수}$$

  $$AIC_p = n \ln\left(\frac{SSE_p}{n}\right) + 2p$$

  - 앞부분은 Goodness of fit이고, 뒷부분은 Penalty이다. 이 둘의 합을 동시에 작게 하기 때문에 적절한 복잡도를 가진 변수 조합이 만들어지겠죠?
  - 낮을수록 더 좋은 모형이라는 뜻
  - AIC는 KL-Divergence (Kullback-Leibler 쿨백-라이블러 발산)의 추정치로 알려져 있다. KL-Divergence는 1) 실제 데이터의 분포와, 2)통계모형이 예측하는 분포 사이의 차이이다. 즉 KL-Divergence가 작을수록 통계모형이 데이터의 참된 분포를 잘 묘사한다는 뜻이지만, 우리는 데이터의 참 분포(실제 생성되는 형태)를 모르기 때

문에 AIC를 이에 대한 추정치로 간주한다. 따라서 AIC가 작다는 것은 우리가 가진 모형이 자료의 true distribution을 잘 묘사한다는 뜻이다.

- 다만 AIC은 우리의 모형이 진짜 'true model'인지는 관심이 없다. 참 모형이 아니더라도 참 모형을 잘 근사하기만 하면 된다. 표현이 어렵지만, 실제 데이터가 인공신경망 모델처럼 생성되지 않더라도, 우리는 신경망 모델로 그 구조를 잘 근사할 수 있는 것과 맥락이 같다. 즉 Prediction 문제에 사용되는 지표이지, True model을 찾기 위한 맥락에서 쓰는 기준은 아니다.
- AIC는 Cross-Validation에 대한 추정치로 이해되기도 한다.

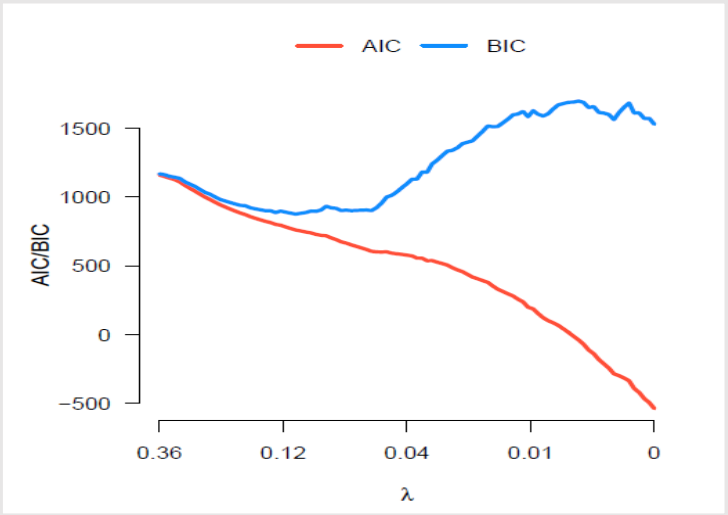
- BIC (Bayesian Information Criterion)

$$BIC = -2\log(likelihood) + p\ln n, \qquad p : \text{변수 개수}$$

$$BIC_p = n\ln\left(\frac{SSE_p}{n}\right) + p\ln n$$

- AIC보다 변수개수에 더 많은 페널티를 부여하는 방식. N > 8이면 BIC는 AIC보다 더 많은 페널티를 부여하게 되고, 따라서 변수 개수가 더 적은 모델을 선호하게 된다.
- AIC와 같이 낮을수록 더 좋은 모형이다.
- BIC는 Bayes Factor (BF)의 로그 값에 대한 추정치로 이해할 수 있다. BF는 Consistency라는 성질이 있는데, 비교대상이 되는 모형들 중에 'True Model'이 있다는 가정하에, 관측치가 많아짐에 따라 참된 모형을 선택할 확률이 1에 가까워진다. AIC는 이런 성질이 없기 때문에, True model을 고르려는 목적일 경우 BIC를 사용하는 게 목적에 맞다.

- 그렇지만…
  - 둘 다 보고 종합적으로 판단하는 것이 더 좋다. 고차원 데이터에서는 정확성이 떨어질 수 있고, AIC와 BIC 모두 각각 문제가 발생한다. 따라서 둘을 모두 고려해서 모형을 선택해야 한다.
    by High-Dimensional Data Analysis (Patrick Breheny)

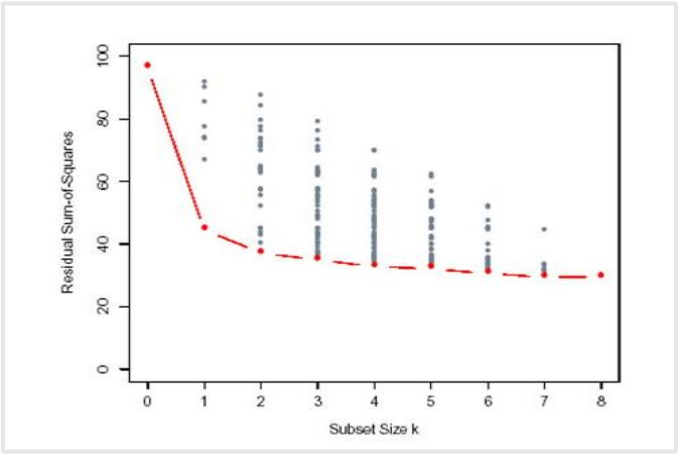


(3) 변수선택의 방법

변수선택법은 모두 경험적(Heuristic)인 방법이다. 딱 Closed form이 존재하는 것이 아니라, 직접 알고리즘에 따라 해당하는 모든 경우를 계산해서 제일 좋은 회귀식을 찾는 방법이다. 따라서 계산량이 꽤나 많다.

- Best Subset Selection (All Possible Regression)
  - 가능한 모든 변수들의 조합을 다 고려한다. 변수의 개수가 p개라면, 2^p 개의 모형을 모두 적합하고 비교해야 한다. 모든 조합을 다 고려해서 결과를 내기 때문에 Best Model에 대한 더 신뢰할 수 있는 결과를 산출한다.
  - 다만 p > 40인 경우 계산이 불가능하며, 적당한 p에서도 많은 관측치를 지니고 있다면 모든 모델을 고려한 계

산 비용이 많이 걸린다. 자동적으로 모델을 찾아주는 것이 아니라, 모든 가능성을 직접 찾기 때문! 2주차 패키지에서 직접 combination마다 비교했었죠…?? 직접 하다보니 10 combination 3도 진짜 많이 걸리는데, 모든 경우를 다 해봤다면 진짜 엄청날 것이다.



- Best Subset Selection의 알고리즘은 다음과 같다. M_1부터 M_p까지를 구하는데, 여기서 M_k $(k = 1, \dots, p)$란 변수의 개수를 k개로 적합했을 때 적합한 회귀식 중 RSS(MSE)가 제일 작은 식이다. 변수의 개수가 같은 선에서 먼저 모델을 고르고, 변수의 개수가 다른 경우에는 AIC 혹은 BIC로 골라서 최적의 회귀식을 찾는 것이다. 칠판으로 설명할게요!

- 전진선택법 (Forward Selection)
  - Null model : $y = \beta_0, \quad \widehat{\beta_0} = \bar{y}$ 에서 시작해서 변수를 하나씩 추가해가는 과정.
  - Null model에서 $x_1 \sim x_p$중 어떤 것을 추가했을 때 AIC와 BIC가 가장 낮은지를 판단한다.
  - 이후 만약 x_2가 추가되었다면, x_2를 제외한 나머지를 추가해보면서 AIC와 BIC가 가장 낮은 변수를 추가하는 과정을 반복한다. 모든 과정을 마쳤을 때, AIC와 BIC가 가장 낮은 모델을 선정한다.
  - 다만 변수를 추가하는 과정에서 모든 조합을 고려하지 않기 때문에 Best Model이라고 말할 수는 없다. 계산량이 비교적 적게 들기 때문에 선호되는 방법이다.

- 후진제거법 (Backward Elimination)
  - Full model : $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$ 에서 시작해서 변수를 하나씩 제거해가는 과정.
  - Full model에서 어느 변수를 뺄 때 AIC와 BIC가 제일 낮은지를 판단하고, 이 과정을 이어나간다. 모든 과정을 마쳤을 때, AIC와 BIC가 가장 낮은 모델을 선정한다.
  - Forward Selection보다 더 좋은 결과를 도출한다고 알려져 있지만, Best subset selection 방법과 마찬가지로 p > 40인 경우에는 적용할 수 없다. 또한 모든 조합을 고려하지 않기 때문에 Best Model이라고 말할 수 없다.

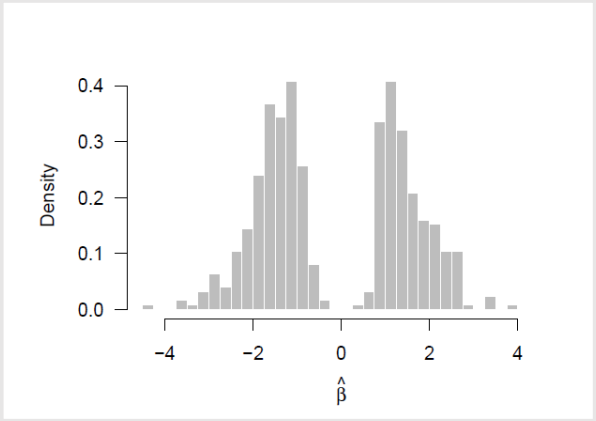
- 단계적 선택법 (Stepwise Selection)
  - Forward Selection과 Backward Elimination 과정을 섞었다. Null model에서 시작할 수도 있고, Full model에서 시작할 수도 있다. 변수를 선택하거나 제거하는 모든 경우를 포함했을 때, AIC와 BIC가 감소하는 방향으로 움직인다.
  - 변수를 선택할 수도, 제거할 수도 있기 때문에 더 유연하게 움직이지만, 당연히 모든 조합을 고려하지 않기 때문에 Best model이라고 말할 수 없다.

(4) 변수선택의 문제점

- 경험적인 방법이라 계산량이 적지 않다. Best Subset Selection은 진짜 많다….

- Stepwise selection의 문제점

  1) R 제곱 값이 실제보다 크게 편향되어 추정된다.
  2) F 통계치의 실제 분포가 가정된 F 분포로부터 벗어난다 .
  3) 모수 추정치들의 표준오차가 실제보다 작게 추정된다.
  4) 3 의 결과로, 모수들의 신뢰구간이 실제보다 좁게 보고된다.
  5) P 값들이 올바르지 않으며 교정하기 어렵다.
  6) 모수 추정치들이 0 이 아닌 것으로 편향되기 쉽다.



  7) 다중공선성 문제가 더 심각해진다.

  Stopping Stepwise : Why stepwise selection is bad and what you should use instead
  https://towardsdatascience.com/stopping-stepwise-why-stepwise-selection-is-bad-and-what-you-should-use-instead-90818b3f52df

  - 이러한 이유들 때문에 단계적 회귀분석은 잘 쓰이지 않고, 뒤에 후술할 LASSO가 많이 쓰인다.
  - BDA (Bayesian Data Analysis)의 저자인 Andrew Gelman도 관련해서 간단한 글을 쓴 적이 있다.

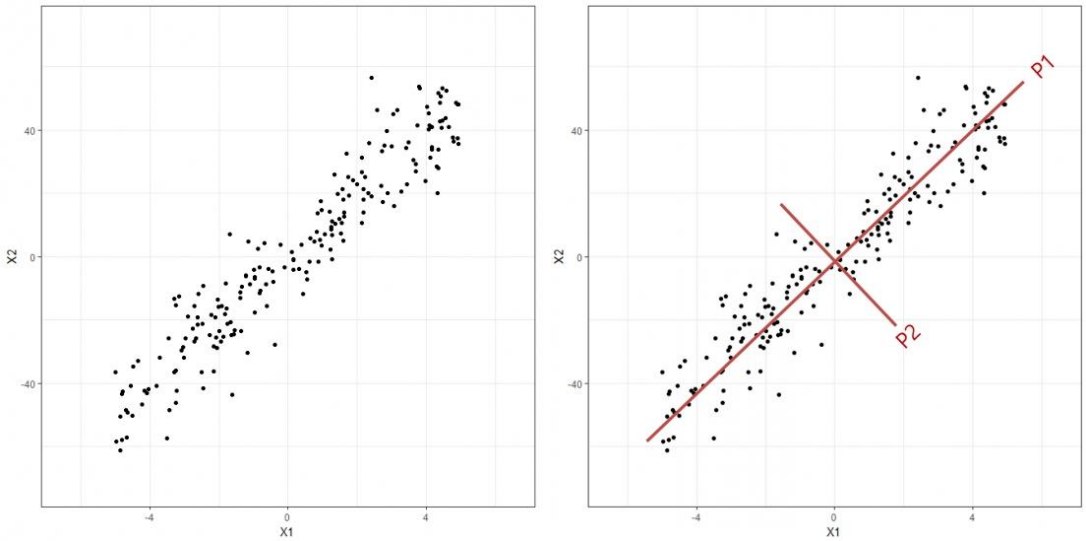
    Why We hate stepwise regression
    https://statmodeling.stat.columbia.edu/2014/06/02/hate-stepwise-regression/?fbclid=IwAR0QRNvNGR-rx9yquuJJh2a9ooU_hwPpnlY_28I-csI0_wi36fCWbnRoEUI

- 저도 LASSO를 더 추천합니다. Best Subset Selection을 합리적으로 대체할 수 있는 방법이라고 생각하는데, 어렵지만 이를 뒤에서 설명하겠습니다. PPT에서는 Forward부터 생략해버립시다.

2. 차원 축소 (Dimension Reduction)

1) PCR (Principle Component Regression, 주성분회귀)

- PCA (Principle Component Analysis, 주성분분석)



- 데이터들의 정보량(분산)을 최대한으로 보존하면서 직교하는 새로운 축을 찾는 방법. X1와 X2가 원래는 상관관계가 높았지만, 저렇게 새로운 직교 축을 찾아서 변수를 PC1, PC2로 바꿀 경우, 두 변수의 상관계수는 0이 된다. 더불어 PC1 하나의 변수만 사용해도 성능을 보장해준다.
  칠판상에서 추가적으로 설명할게요!
- 즉 고차원의 데이터를 상관관계가 없는 저차원의 공간으로 축소시켜주는 방법
- 선대팀 3주차에서 다룰 내용이니, 더 자세한 내용은 선대팀 참조

- PCR
  - PCA를 통해 X변수를 변환해주고, 이 변환된 Z와 y에 대해 그냥 회귀분석을 적용해준다.
  - 다중공선성을 완벽히 해결해주고, 적은 변수를 사용하기 때문에 과적합(Overfitting)을 방지한다.
  - 다만 다중공선성이 명확하지 않은 경우에는 성능이 떨어지고,
    성능이 좋더라도 해석이 어려워지는 단점이 있다.

2) PLS (Partial Least Square, 부분최소제곱법)

- 아이디어
  - PCR는 X들의 정보량을 최대화하는 축을 찾아서 y와 회귀분석을 했다. 저 선형변환에는 y의 정보가 전혀 사용되지 않았다.
  - PLS는 y와 X들의 선형결합의 공분산을 최대로하는 선형결합을 찾는다. 즉 y와 선형관계가 높은 x들의 선형결합을 찾아서 변수를 생성해내는 것! Y의 정보를 사용하기 때문에 예측성능이 PCR보다 조금 더 좋다고 알려져 있다.
  - 다만 해석이 PCR보다 어렵고, 사용하는 것을 잘 보지 못했다. 이런 아이디어의 방법이 있다는 정도로 알아두자.

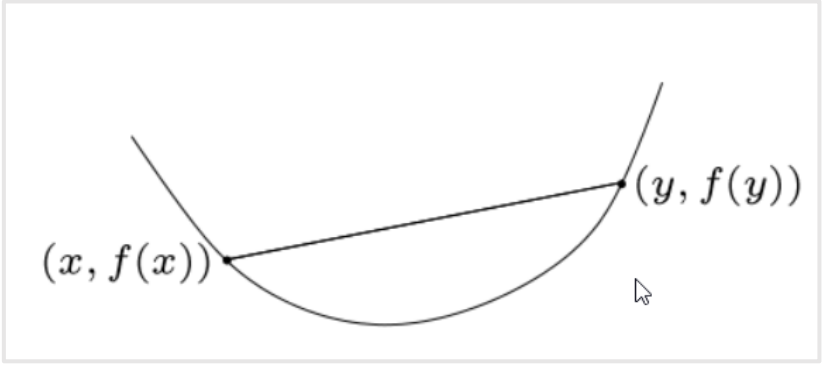
3. Convex Optimization (컨벡스 최적화)

1) Convex

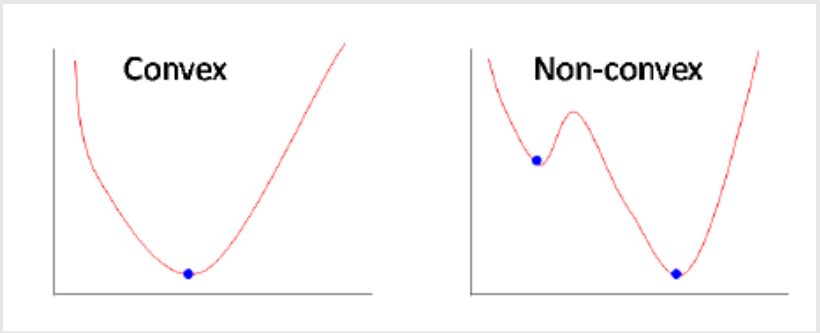
- Convex Function (볼록 함수)
  - 컨벡스 Convex 함수란 무엇인가? 우리가 아는 $y = x^2$ 형태의 함수를 컨벡스 함수라고 한다.
    컨벡스 함수의 조건을 수식으로 쓰게 되면,

$$f(\theta x + (1-\theta)y) \le \theta f(x) + (1-\theta)f(y), \qquad with\ \ 0 \le \theta \le 1, \qquad for\ all\ x, y \in dom\ f$$

  - 컨벡스 함수를 좀더 직관적으로 시각자료를 통해 이해해보자.



  - 이를 좀더 간단히 예시를 들어보자면, 이계도함수 $f''(x) \ge 0$ 이면 컨벡스 함수이다. 예를 들어 $f(x) = x^2$ 는 그림으로 이해했을 때 컨벡스 함수인데, 두 번 미분하면 $f''(x) = 2$ 이므로 역시 컨벡스 함수가 맞다.



- 위의 그림처럼 물결치게 되면 convex함수가 아니다.
- 그렇다면 f에 -1을 곱하면 어떻게 될까? $f(x) = -x^2$ 은 위로 봉우리가 있는 형태이고, 두 번 미분하면 음수가 나온다. 따라서 이는 컨벡스 함수가 아니라, Concave 함수라고 한다. Convex = - Concave

- Example – 어떤 함수가 컨벡스 함수일까?
  - 이차함수 형태는 컨벡스 함수다. 다변수여도 동일하다. 회귀분석에서 제곱합을 최소화했는데, 이 또한 제곱항이 있기 때문에 컨벡스 함수를 최소화한 형태다. 이를 행렬 상에서 다루더라도 컨벡스 함수로 정의할 수 있는데, 이는 생략하겠지만 아무튼 이차(Quadratic)의 형식은 컨벡스 함수다.
  - Affine Function은 컨벡스 함수다. Affine 함수란 선대팀 1주차 때 살짝 다뤘죠? ax+by+c 와 같은 일차 결합에 상수항까지 고려한 함수를 아핀 함수라고 한다. 그냥 일차함수라고 생각해도 크게 문제없다.
    이 아핀함수는 convex이면서 concave이다. 두 번 미분하면 0이니까!
  - 모든 R^n상의 norm은 컨벡스이다. norm이란 벡터의 길이를 나타내는 값이었는데, 이를 어떻게 계량화 하는지에 따라 Lp-Norm이라고 한다. 아무튼 Norm도 convex이다. L0-norm은 0이 아닌 성분의 개수로 이해하면 된다.

$$||x||_p = \sqrt[p]{\sum_i {x_i}^p} = \sqrt[p]{{x_1}^p + {x_2}^p + \cdots + {x_n}^p}$$

$$||x||_0 = \#\{\, i \mid x_i \neq 0\}, \qquad \left|\left|x\right|\right|_1 = \sum_i |x_i|, \qquad \left|\left|x\right|\right|_2^2 = \sum_i {x_i}^2$$

  - 조금 더 정확히 말하면, L1-norm과 L2-norm은 컨벡스 함수이지만, L0-norm은 엄밀한 의미의 Norm도 아니고 컨벡스 함수가 아니다. 이 점을 기억해두자. 더불어 L1-norm은 Convex이면서 Concave이다.

2) Optimization Problem

$$\begin{aligned} \min_{x \in D} \quad & f(x) \\ \text{subject to} \quad & g_i(x) \leq 0,\ i = 1, \ldots m \\ & h_j(x) = 0,\ j = 1, \ldots r \end{aligned}$$

- 최적화 문제
  - 최적화 문제란 제약조건 하에서 최적의 해(Optimal Solution) 또는 최적의 해에 근접한 값을 찾는 문제를 말한다. 일반적으로 기계학습 분야에서는 비용함수(Cost function)를 최소화하는 모델의 파라미터(parameter)를 구하게 되는데, 이것은 최적화 문제로 정의될 수 있다.
    하지만 보통 우리는 최소화 (Minimization) 문제를 한정해 다루는데, 왜냐하면 최대화 (Maximization) 문제는 목적함수에 -1을 곱하면 최소화 문제로 바꿀 수 있기 때문이다.
  - 이런 최적화 문제에는 다양한 것들이 있다. 예시들과 목적함수들을 나열해보면,

    Least Square : $min_\beta \sum_i (y_i - {x_i}^t \beta)^2$

Least Absolute Deviations : $min_\beta \sum_i |y_i - x_i{}^t\beta|$

Regularized Regression : $min_\beta \sum_i (y_i - x_i{}^t\beta)^2, \quad s.t. \quad ||\beta||_1 \le t$

Logistic Regression, SVM, Maximum Likelihood Estimation, Traveling-Salesman Problem (TSP)

- 반대로 최적화 문제가 아닌 것들의 예시로는 다음과 같다.

  Hypothesis Testing, Boosting, Random Forest, Cross-Validation, Bootstrap, etc…

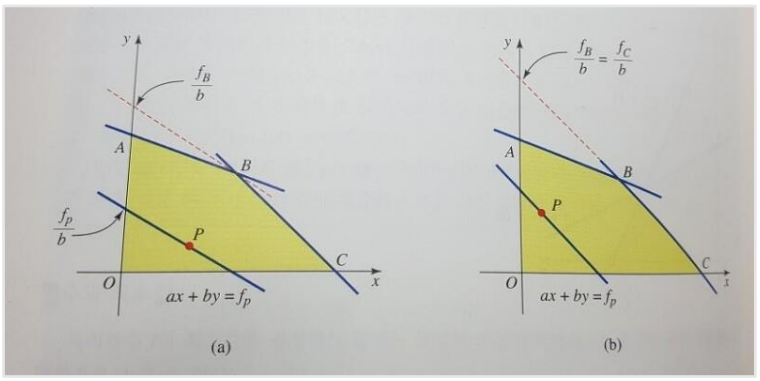
- 근데 왜 갑자기 최적화?
  - 우리가 지금까지 다룬 회귀분석은 전부 최적화 문제였다. 다만 이게 최적화 문제인지 모르고 있었을 뿐…. 최소제곱법, 최대가능도법, Median regression, Huber's M, Best Subset Selection, PCR 모두 최적화의 관점에서 목적함수와 제약식을 정의할 수 있다.
  - 특히 앞으로 우리가 다룰 Ridge와 LASSO의 경우, 사실 최적화와 떼어놀 수 없다. **결론적으로 왜 LASSO가 Best Subset Selection보다 좋은 지를 이론적으로 다루기 위해서 이 부분을 넣었다**. 다른 회귀분석 교안 3주차에는 전혀 다뤄오지 않았기 때문에 낯설 수 있지만, 새로운 관점을 공부할 수 있으니 알아 두면 매우 좋다고 생각한다!

- 최적화의 종류 – Linear Programming (LP)
  - 선형계획법(LP)는 고등학교 1학년 때 다 배웠다.

문제 1 네 부등식

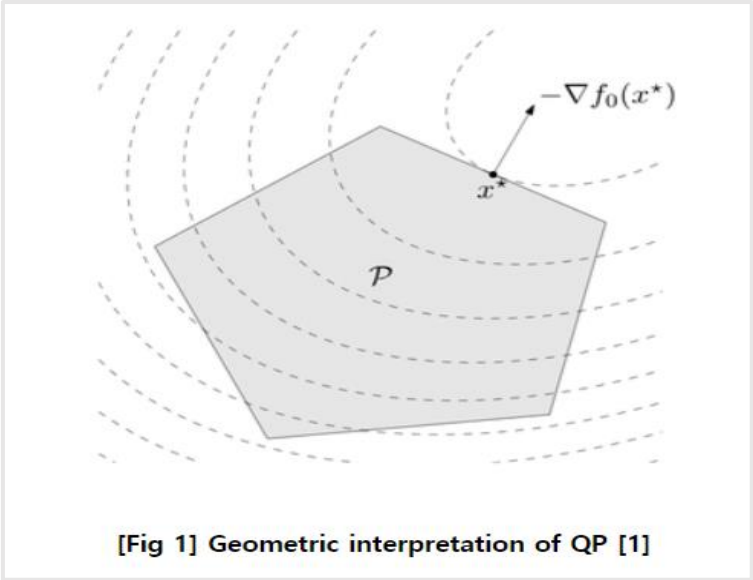
$$x \ge 0,\ y \ge 0,\ 2x+y-5 \le 0,\ x+2y-4 \le 0$$

을 동시에 만족하는 x, y에 대하여 $3x+2y$의 최댓값과 최솟값을 구하여라.



  - 목적함수가 3x+2y로, 두 변수의 선형결합 형태이기 때문에 선형계획법이라고 한다. 주어진 제약조건 하에서, 목적함수를 최대화하는 문제를 다루고 있다.

- 최적화의 종류 – Quadratic Programming (QP)
  - 저 제약조건 하에서 목적함수가 이차식인 함수를 최적화하는 것이 QP이다. 우리가 지금까지 해왔던 회귀분석이 다 QP였고, 최적화는 미분을 통해 진행했다. LP와 QP이외에도 Semi-definite Programming (SDP), Coinc Programming (CP)라는 방법이 있지만 짱 어렵습니다.
  - 이차계획법(QP)는 바로 회귀분석의 문제다. 일반적인 최소제곱회귀에서는 제약조건이 없었다. 제약조건이 없는 상황에서 최적화를 하는게 최소제곱법이고, 제약조건이 있을 때 최소제곱회귀를 하는 것이 뒤에 다룰 Regularized Regression (Shrinkage Method)이다.



[Fig 1] Geometric interpretation of QP [1]

3) Convex Optimization

- 모두를 위한 컨벡스 최적화

$$\begin{array}{ll} \min\limits_{x\in D} & f(x) \\ \text{subject to} & g_i(x)\le 0,\ i=1,\dots m \\ & h_j(x)=0,\ j=1,\dots r \end{array}$$

  - **컨벡스 최적화란 컨벡스 함수의 형태 안에서 최적화를 하는 것이다**. '컨벡스 형태 안'이라 함은, 목적함수도 컨벡스하고, 제약조건도 컨벡스한 상황을 말한다. f, g, h가 모두 컨벡스하다면, 우리는 매우 간단하게 미분을 통하거나, 미분이 불가능한 점이 있는 경우 비교적 간단한 수치적(Numerical) 방법들을 통해 구할 수 있다. 수치최적화 방법에는 경사하강법, 뉴튼-랩슨 등등 기본적인 방법 이외에도 매우 다양한 방법들이 있는데, 여기서는 기본적인 내용도 전혀 다루지 않겠다. 하지만 아무튼 상대적으로 매우 간단하다는 것이 핵심!
  - 미분이 가능한 컨벡스 최적화

    Least Square : $min_\beta \sum_i (y_i - {x_i}^t\beta)^2$

    Ridge Regression : $min_\beta \sum_i (y_i - {x_i}^t\beta)^2, \quad s.t. \quad ||x||_2^2 \le t \ \ \left(\sum_j^p {\beta_j}^2 \le t\right)$
  - 미분이 불가능한 컨벡스 최적화

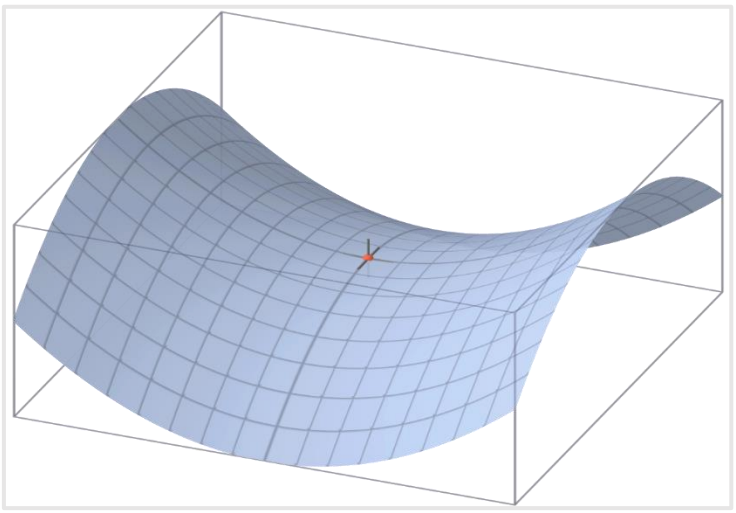
    Least Absolute Deviations : $min_\beta \sum_i |y_i - {x_i}^t\beta|$

    LASSO Regression : $min_\beta \sum_i (y_i - {x_i}^t\beta)^2, \quad s.t. \quad ||\beta||_1 \le t \ \ \left(\sum_j^p |\beta_j| \le t\right)$
  - Non-convex Optimization

    Best Subset Selection : $min_\beta \sum_i (y_i - {x_i}^t\beta)^2, \quad s.t. \quad ||\beta||_0 \le t \ \ (\#\{i \mid x_i \ne 0\} \le t)$

- 컨벡스 최적화를 다루는 이유
  - 일단 미분이 가능한 컨벡스 최적화의 경우 매우 간단하다. 우리는 고등학교때 최대, 최소값을 구할 때, 미분값 =0으로 놓고 풀었다. 최소제곱법의 경우에도 미분하면 간단하게 풀렸다. 이렇게 미분가능한 컨벡스 함수 형태일 경우 문제가 간단해진다. 미분이 불가능할 경우에는 수치적 방법으로 해결할 수 있다.
  - 물론 nonconvex optimization의 경우가 더 많다. 목적 함수가 고차항일 경우 nonconvex할 것이고, 혹은 best subset selection의 경우처럼 제약식이 L0-norm인 경우 nonconvex할 수 있다. 무엇보다 일반적으로 다변수 함수는 안장점 (Saddle Point) 때문에 컨벡스하지 않은 경우가 많다. 하지만 이런 Nonconvex

Optimization을 푸는 가장 일반적인 방법은 이를 컨벡스한 형태로 완화 (Convex Relaxation) 해주는 것이다.



4) 라그랑주 승수법 (Lagrange Multiplier)

- 어떻게 간단한 형태로 만들 수 있을까?

$$\begin{aligned} \min_{x \in D} \quad & f(x) \\ \text{subject to} \quad & g_i(x) \leq 0,\ i = 1, \dots m \\ & h_j(x) = 0,\ j = 1, \dots r \end{aligned}$$

  - 다음과 같이 제약식을 포함하는 형태에서 최적화하는 것은 쉽지 않다. 시각적으로는 이해하는데 도움되지만, 조금 관점을 바꾸면 두 개 이상의 함수를 각각 다뤄야 하는 형태이기 때문이다. f의 미분값이 0이라고 두었는데, 만약에 그 값이 g나 h에서 유효하지 않다면? 그러므로 제약식이 있는 상황에서 최적화는 매우 어렵다. 따라서 우리는 이를 다룰 수 있게 목적함수를 간단한 형태로 바꿔주는 방법이 필요하다.
  - 다행히도 라그랑주 승수법이라는 것을 사용하면 단 하나의 목적함수로 최적화가 가능해진다. 목적함수와 제약식이(f, g, h) 컨벡스하다면, 이를 하나의 식으로 바꿔도 동일한 결과를 산출한다. 그래서 지금까지 컨벡스가 뭔지 따져왔다! 더 엄밀하게 정의하려면 KKT condition을 정의해야 하지만 다루지 않는다.

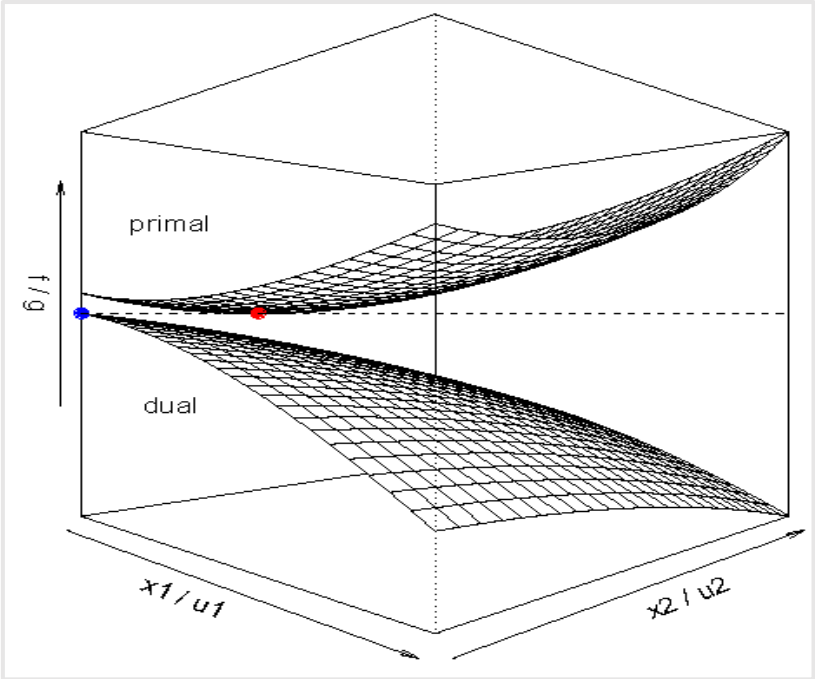
- Duality
  - 계속 정의했던 형태를 수식으로 써보자. 이를 원 문제 (Primal Problem)이라고 한다.

$$min_x f(x), \quad s.t. \quad g_i(x) \leq 0, \quad h_j(x) = 0$$

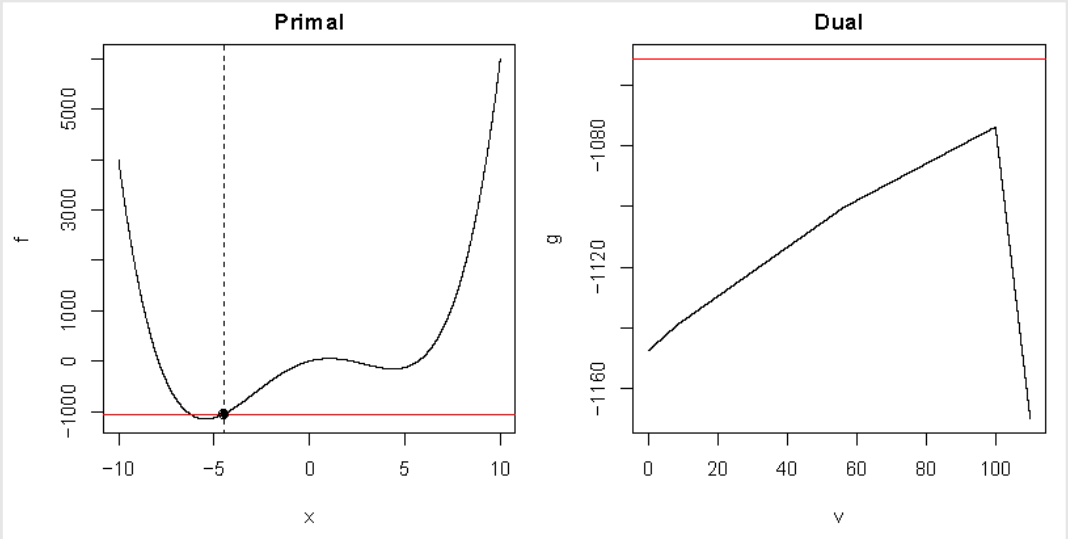
    이를 라그랑주 승수법을 통해 바꾸면 다음과 같고, 이를 쌍대 문제 (Dual Problem)이라고 한다.

$$max_{u,v}\ min_x \left\{ f(x) + \sum_{i=1}^{m} u_i g_i(x) + \sum_{j=1}^{r} v_j h_j(x) \right\}$$

  - Primal은 최소화의 문제라면, Dual은 최대화의 문제다. 만약 f, g, h가 모두 컨벡스하다면, 최소화와 최대화는 완전히 동일한 결과를 산출한다. 아래는 QP에서의 Primal과 dual solution이 동일함의 시각화다.



- 만약 nonconvex하다면 어떻게 될까?
  두 결과는 동일한 결과를 산출하지 못하고, Primal solution > Dual solution 이 된다.



- 하지만 이때 u와 v는 우리가 최대화하지 않는다. Ridge와 Lasso에서 튜닝 파라미터 lambda를 최적화하진 않죠? 현실에서는 u와 v는 고정시키고, 해당 목적함수를 최소화한다! 이를 u^*, v^*라 두면, Primal을 다음과 같이 dual로 풀 수 있다.

  Primal : $min_x f(x),\ s.t. \quad g_i(x) \le 0,\ h_j(x) = 0$

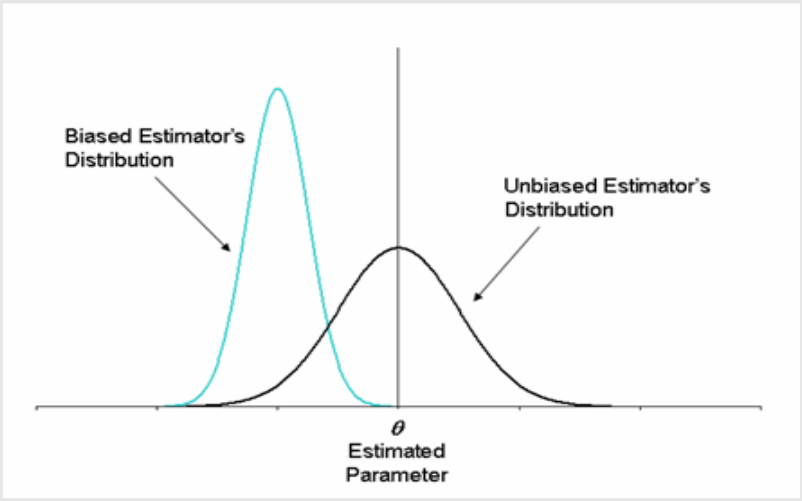
  Dual : $min_x \left\{ f(x) + \sum_{i=1}^{m} u_i{}^* g_i(x) + \sum_{j=1}^{r} v_j{}^* h_j(x) \right\}$

- 이 결과를 통해 우리는 제약조건이 있는 최적화를 할 수 있게 되었다! 이제 볼 Ridge와 Lasso를 컨벡스 최적화의 관점으로 바라보자!

4. Ridge Regression

- Shrinkage Method
  - 축소추정량이란 각각 개별 베타추정량을 0으로 수축시키는 방법이다. 다중공선성이 존재할 경우 각각 개별 베타계수의 분산이 매우 크게 상승하게 된다. 기존의 LSE 방법은 BLUE라는 점에서 매우 이론적으로 좋은 통계

량이지만, 다중공선성이 존재할 경우 분산이 너무 큰 추정량이다. 그렇다면 만약 우리 추정량의 일부 편향을 허용하되, 분산을 더 줄일 수 있다면 어떨까? 불편성을 포기하되, 전체 MSE (Bias^2 + Variance)를 더 작게 하는 추정량을 얻는 방법이다.



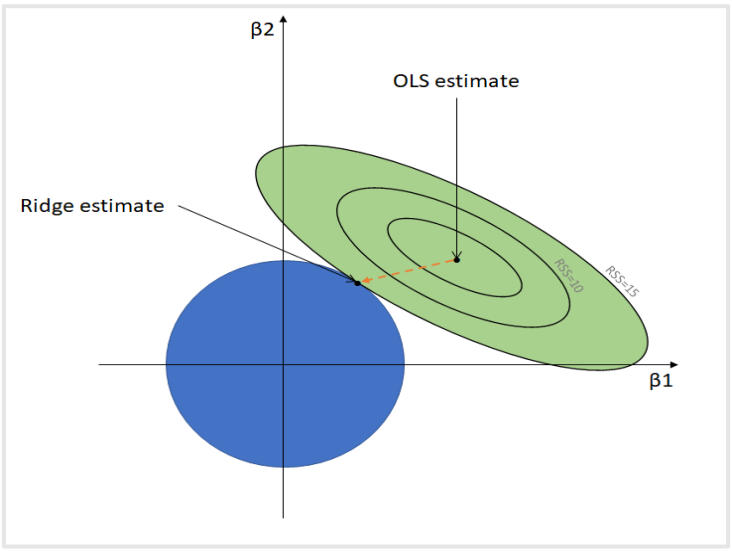
- 최소제곱 형태의 기존 목적함수에, 베타의 크기에 대한 제약을 두는 방식으로 진행한다. 원래 최소제곱 방법에서는 베타 크기에 제약이 없었고, 그에 따라 커지는 정도에 제한이 없었다. 그에 따라 베타는 BLUE가 될 수 있었는데, 베타 크기에 제약을 둠으로써 베타를 다소 과소추정하더라도 전체적인 MSE를 줄이는 방식이다.

Ridge Regression (능형 회귀)

- Primal Problem
  - 아까 배운 컨벡스 최적화의 관점에서 접근해서, Primal을 Dual로 풀어내는 방식으로 접근해보자. 먼저 Ridge의 Primal Problem의 수식은 다음과 같다.

$$\hat{\beta}^{ridge} = min_{\beta} \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2, \quad subject\ to \sum_{j=1}^{p}{\beta_j}^2 \le s$$

  - 이는 제약조건과 목적함수가 모두 convex한 QP (이차계획법) 형태이다.
    이에 대해 beta1과 beta2만 있는 상황을 고려해서 시각화를 해본다면,



  - 파란 범위가 릿지의 제약식 부분이다. 고등학교 수학 감성으로 이해해보면, $x^2 + y^2 \le r^2$ 인 원 내부에 해가 있어야 한다는 제약이다. 따라서 우리 목적함수의 해는 저 파란 부분 안에 존재해야 한다. 따라서 기존의 LSE 추정량이 타원의 중심이라면, 이를 원과 타원의 접점으로 이동시킨다. 그에 따라 최소제곱 추정량보다 목적함수의 결과값은 커질 수 있지만, 베타 값에 대한 제약 때문에 베타값이 커지는 것을 막는다. 릿지 추정량이 기존

LSE의 beta1, beta2보다 작음을 확인할 수 있다.

- 제약식 (원)의 범위를 정하는 s값은 tuning parameter로, 우리가 미리 설정하는 값이다. s값이 클수록 제약범위가 넓고, 제약범위가 충분히 넓다는 것은 제약이 없는 것과 마찬가지이다. 따라서 s가 무한히 클 경우, 릿지 추정량은 LSE와 동일하다. 반대로 s값이 작으면 작을수록 우리 제약범위가 0과 가까이 위치하게 된다. 그 결과 릿지 추정량은 0과 매우 가까운 값을 갖는다. 하지만 그림에서 보면 알 수 있듯이, 정확히 0을 찍지는 못한다!

- 이를 Bias-Variance Trade off 관점에서 이해해보자. s가 커지는 것은 LSE와 비슷한 추정을 하기 때문에 Low bias, High variance의 형태이고, s가 작아지는 것은 High bias, Low variance의 형태이다. 따라서 MSE가 최소가 되는 적절한 람다를 찾는 것이 중요하다. 그런데 Primal 형태에서는 적절한 s를 선택하는 것이 어렵다.

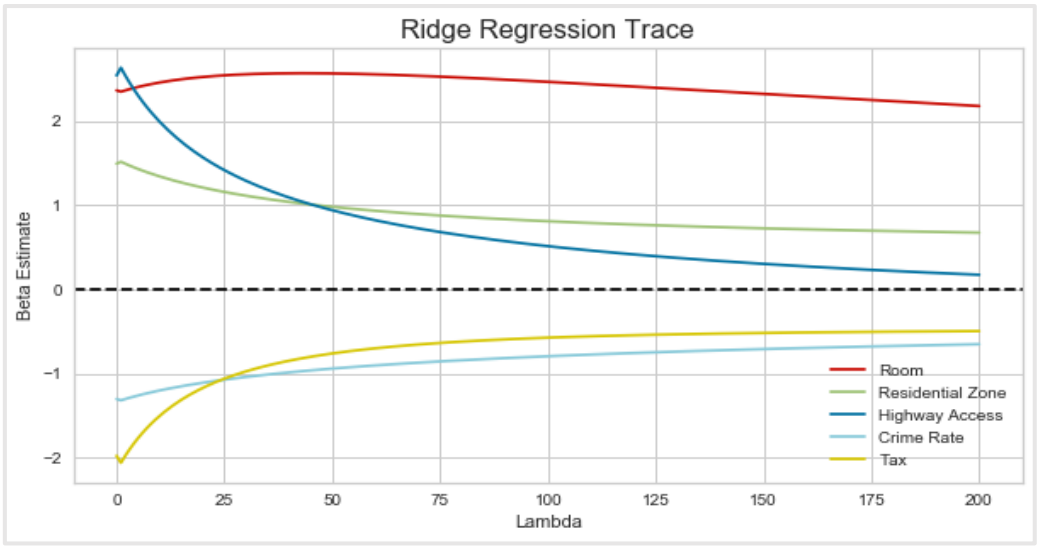
- Dual Problem

  - 주어진 원(Primal) 문제를 쌍대(Dual) 문제로 다루기 위해 식을 변형시켜보자. 현재 목적함수와 제약식이 모두 Convex하기 때문에 Primal과 Dual의 해는 완전히 같다.

$$\hat{\beta}^{ridge} = min_{\beta} \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}{\beta_j}^2, \qquad \lambda \geq 0$$

  - 이에 대해 이해해보자. 해당 식은 기존의 최소제곱 목적함수도 최소화하면서, 개별 베타 제곱합도 동시에 작게 만든다. 원래 베타가 마음껏 팽창할 수 있을 때, 최소제곱이 목적함수가 최소화되었다. 그런데 베타에 대한 페널티가 주어지면서, 베타가 마음껏 팽창하면 해당 식은 최소화될 수 없다. 따라서 베타가 기존 LSE보다 작아지게 된다.

  - 아까는 원의 제약범위 s가 tuning parameter였다면, 여기서는 λ(람다)가 tuning parameter가 된다. 만약 람다가 0이라고 하면, 이는 기존의 최소제곱 형태와 완전히 동일하다. 반면 람다가 매우 커진다고 생각하면, 전체 목적함수를 작게 만들기 위해서는 전체 식의 앞부분 (최소제곱)보다는 뒷부분 (페널티)에 영향을 많이 받을 것이다. 그 결과 람다가 커지면 beta값이 매우 작아져서 0과 가까운 값이 나올 것이다. 하지만 아까도 말했듯, 정확히 0을 찍지는 못하고, 0에 근접한 값이 나온다. 다음은 람다값을 변화시킴에 따른 베타값 변화의 시각화이다.



  - 이를 Bias-Variance Trade off 관점에서 이해해보자. 람다가 작아지면 LSE와 비슷하면서 Low bias, High Variance한 형태이다. 람다가 커지면 High bias, Low variance의 형태이다. 따라서 MSE가 최소가 되는 적절한 람다를 찾는 것이 중요하다.

- Properties of Ridge

- 미분이 가능하다. 전체 식을 보았을 때, 그 어디를 봐도 미분 불가능해 보이지 않는다. 이를 행렬로 이해하고, 행렬상에서 미분을 통해 해를 구하면 매우 간단한 결과를 도출한다. 해당 결과는 다음과 같고 간단한 증명 가자!

$$\hat{\beta}^{ridge} = (X^tX + \lambda I)^{-1}X^ty$$

- X^tX가 full rank가 아니어도 unique beta solution이 존재한다. X^tX의 역행렬은 없어도, $X^tX + \lambda I$의 역행렬은 존재할 수 있겠지? 그 결과, n <<< p 인 high-dimension의 경우에도 추정이 가능하다.

- 릿지 추정량은 개별 베타값을 0에 가깝게 만들지만, 정확히 0으로 만들지는 않는다. 다중공선성을 해결해 좋은 예측을 가능하게 하지만, 변수 선택을 통한 해석력을 증가시켜 주진 못한다. 진짜 안중요한 베타가 있을 수도 있는데, 이거를 0에 가깝게 수축시키더라도 결국에는 0이 아니다. 따라서 분석가는 해당 변수를 자의적으로 제외하는 것에 고민과 한계를 가질 수밖에 없다.

- 릿지 추정량은 biased estimator이다. 하지만 LSE보다 훨씬 더 많은 분산을 줄이기 때문에, 더 좋은 성능을 발휘하는 경우가 많다. 또한 Consistent estimator (일치추정량)이기 때문에, 관측치가 늘어남에 따라 실제 beta값에 다가가는 좋은 성질을 지니고 있다.

- 릿지의 베타 값은 1st Principle Component 축의 방향으로 축소한다고 알려져 있다. 이는 SVD를 통한 Ridge의 해를 구하고 trace를 구할 수 있는데 무시하자. 그냥 그쪽으로 수축함에 따라 다중공선성 문제를 해결한다고 어디서 주워들은 이해를 해보자. 관심 있으면 해당 링크와 Elements of Statistical Learning 참고 https://online.stat.psu.edu/stat508/lesson/5/5.1

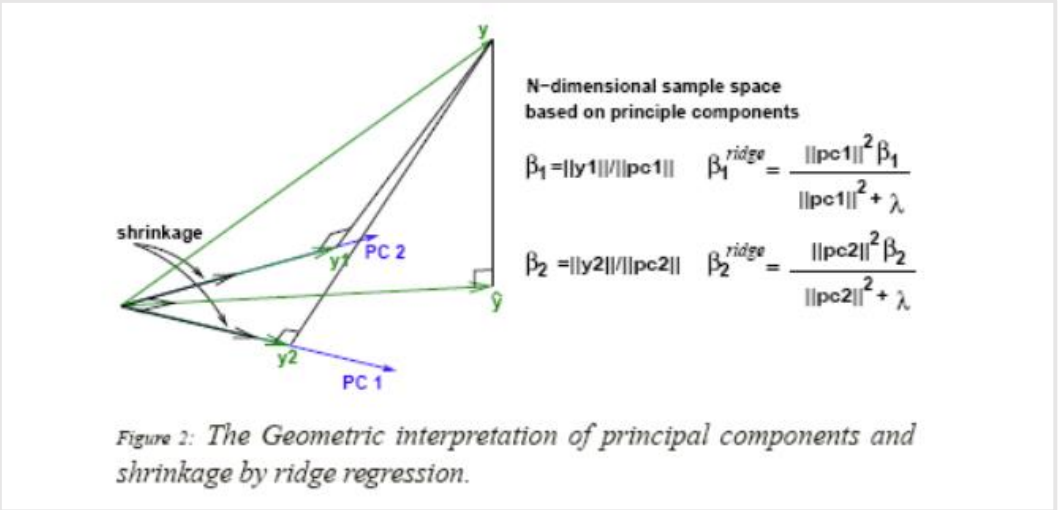


Figure 2: The Geometric interpretation of principal components and shrinkage by ridge regression.

- 람다 정하기

  - 저렇게 Dual로 풀면 편하고, 미분도 가능하다는 것을 알겠다. 그러면 이제 적절한 람다를 구해야겠지? 이 람다를 구하는 과정을 이해해보자.

  - 먼저 개별 변수들을 scaling해줘야 한다. Shrinkage method는 각각 베타계수를 수축시킨다. 근데 각각 변수의 단위가 다르면 페널티의 영향력도 달라질 수 있다. 따라서 x 변수들을 scaling해줘야 한다. 이런 통계적인 방법에서는 보통 min-max scaling보다는 standardize를 주로 한다고 한다.

  - 개별 변수들의 단위를 맞춰주는 준비를 마쳤다면 이제 람다를 구해야 한다. 보통 람다는 CV (Cross Validation)을 통해 선정한다. 후보가 될 람다 값들을 정하고, 개별 람다들에 대한 CV error를 구한다. 이때 최소의 CV error를 만드는 람다를 선택한다. 해당 람다를 가지고, 이번에는 cv를 하는게 아니라 전체 관측치에 대해 적합해서 우리의 최종 베타를 추정한다. 잠시 뒤에 R과 Python을 통해 알아보자!!

5. Lasso Regression

- Sparse Solution

- 릿지는 다중공선성을 해결해주면서 예측성능까지 매우 좋다. 하지만 베타 계수가 0에 가까워지더라도 정확히 0이 되지는 않기 때문에, 변수 선택의 기능이 없다. 그렇다면 변수선택을 하는 shrinkage method는 없을까? 제약식을 조금만 조정해보면 변수선택이 가능해질 수 있으니까!

- 회귀식의 변수 중에 몇 개만 중요하다고 생각해보자! 그렇다면 몇몇 베타는 0이 될 테니 다음과 같은 방식으로 제약식을 만들 수 있을 것이다.

$$\hat{\beta}^{L0-norm} = min_{\beta} \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2, \qquad subject\ to\ \ ||x||_0 \le s$$

- 즉, '0이 아닌 베타의 개수'를 페널티로 삽입하는 것이다. 근데 이 아이디어는 뭐와 비슷해 보이는데? 바로 Best Subset Selection의 아이디어를 목적식으로 만들면 다음과 같다.

- Convex Relaxation

  - 하지만 다음과 같은 식은 큰 문제가 있다. 제약식인 L0-norm이 nonconvex하다. 단순한 nonconvex도 아니고 개수를 세는 문제다. 아까 best subset selection의 경우에서 말했든, 조합을 고려하는 문제는 p의 개수가 늘어나면 그에 따른 계산비용이 지수적으로 증가해서 결국 계산이 불가능하다. 따라서 이를 위해서는 nonconvex한 제약식을 convex한 형태로 완화 (Relaxation) 해줘야 한다.

  - 보통 이런 개수를 세는 문제의 경우, 제일 많이 쓰는 완화 방법은 절대값의 합으로 바꿔주는 방법이다. L0-norm을 L1-norm으로 바꿔주는 방법이다. 실제로 PCA, PCR, Sparse PCA와 같은 방법도 다 이런 방식의 완화된 형태를 통해 최적화한다. 단순히 라쏘가 변수선택이 되서 쓰는게 아니라, 이런 컨벡스 형태로의 완화라는 이론적 기반이 있기 때문에, LASSO가 Best Subset Selection을 합리적으로 대체할 수 있다.

$$||x||_0 = \#\{\, i \mid x_i \neq 0\}, \qquad ||x||_1 = \sum_i |x_i|$$

Lasso Regression

- Primal

  - 따라서 Convex Relaxation을 거친 결과, Primal Problem은 다음과 같이 정의된다.

$$\hat{\beta}^{Lasso} = min_{\beta} \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2, \qquad subject\ to\ \ \sum_j |x_j| \le s$$

  - 이는 Ridge와 마찬가지로 제약조건과 목적함수가 모두 convex한 QP (이차계획법) 형태이다.
  이에 대해 beta1과 beta2만 있는 상황을 고려해서 시각화를 해본다면,

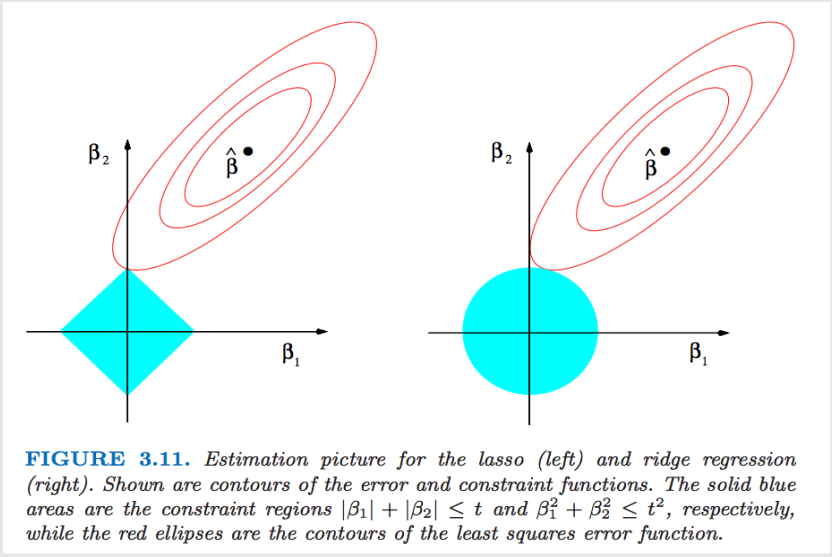


FIGURE 3.11. *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions* $|\beta_1| + |\beta_2| \le t$ *and* $\beta_1^2 + \beta_2^2 \le t^2$*, respectively, while the red ellipses are the contours of the least squares error function.*

- 아까 릿지와 거의 유사한 형태임을 이해할 수 있겠죠? 다만 제약식의 형태가 릿지에서는 원과 같았다면, 라쏘에서는 마름모와 같아진다. 아까 릿지처럼 하나하나 짚어보자. 고등학교 수학 감성으로 이해하면, $|x| + |y| \leq r$ 인 마름모 내부에 해가 있어야 한다는 제약이다. 따라서 우리 목적함수의 해는 색칠된 부분 내부에 존재해야 한다. 기존 LSE가 타원의 중심에 있었다면, 라쏘는 이를 마름모와 타원의 접점 위로 위치시킨다. 그에 따라 라쏘 추정량의 베타가 기존 LSE의 베타보다 작음을 확인할 수 있다.

- 제약식 (마름모)의 범위를 정하는 s값은 tuning parameter로, 우리가 미리 설정하는 값이다. s값이 클수록 제약범위가 넓고, 제약범위가 넓다는 뜻은 제약이 약하다는 것과 같다. 따라서 s가 무한히 클 경우, 릿지 추정량은 LSE와 동일하다. 반대로 s값이 작으면 작을수록 우리의 제약범위가 0과 가까이 위치하게 된다. 그 결과 라쏘 추정량 중 일부의 베타는 0을 해로 갖게 된다. 아까 릿지는 정확히 0을 찍지 못했지만, 라쏘는 정확히 0을 찍어버린다!!

- 이를 Bias-Variance Trade off 관점에서 이해해보자. s가 커지는 것은 LSE와 비슷한 추정을 하기 때문에 Low bias, High variance의 형태이고, s가 작아지는 것은 High bias, Low variance의 형태이다. 따라서 MSE가 최소가 되는 적절한 람다를 찾는 것이 중요하다. 그런데 Primal 형태에서는 적절한 s를 선택하는 것이 어렵다. 릿지와 동일한 특성이다.

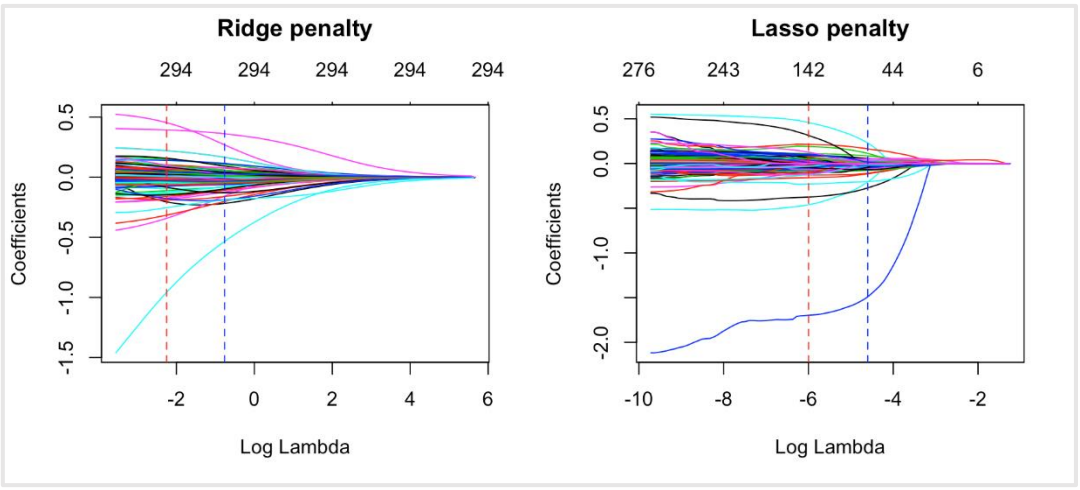
- Dual

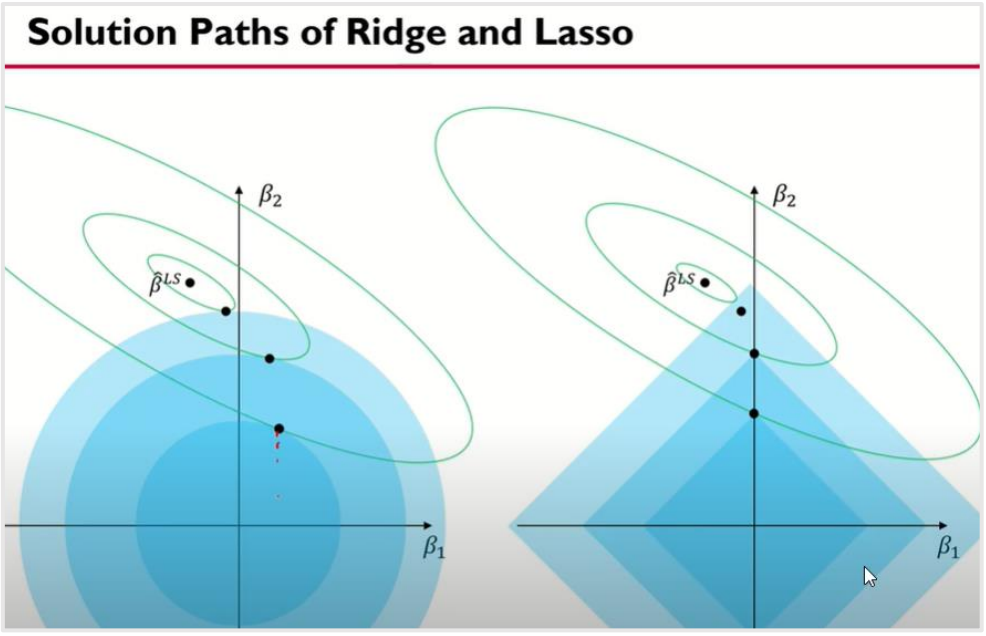
  - 주어진 원(Primal) 문제를 쌍대(Dual) 문제로 다루기 위해 식을 변형시켜보자. 현재 목적함수와 제약식이 모두 Convex하기 때문에 Primal과 Dual의 해는 완전히 같다.

$$\hat{\beta}^{lasso} = min_{\beta} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| , \qquad \lambda \geq 0$$

  - 이에 대해 이해해보자. 해당 식은 기존의 최소제곱 목적함수도 최소화하면서, 개별 베타 절대값의 합도 동시에 작게 만든다. 원래 베타가 마음껏 팽창할 수 있을 때, 최소제곱이 목적함수가 최소화되었다. 그런데 베타에 대한 페널티가 주어지면서, 베타가 마음껏 팽창하면 해당 식은 최소화될 수 없다. 따라서 베타가 기존 LSE보다 작아지게 된다.

  - 아까는 원의 제약범위 s가 tuning parameter였다면, 여기서는 λ(람다)가 tuning parameter가 된다. 만약 람다가 0이라고 하면, 이는 기존의 최소제곱 형태와 완전히 동일하다. 반면 람다가 매우 커진다고 생각하면, 전체 목적함수를 작게 만들기 위해서는 전체 식의 앞부분 (최소제곱)보다는 뒷부분 (페널티)에 영향을 많이 받을 것이다. 그 결과 람다가 커지면 beta값이 매우 작아져서 결국 0이 되는 베타들이 생길 것이다. 릿지 추정량은 정확히 0이 나오지 않았지만, 라쏘는 정확히 0의 베타값을 반환함으로써 변수를 자연스럽게 선택해준다! 다음은 라쏘에서 람다 값이 점점 커짐에 따라 개별 베타계수가 0으로 수렴하는 것을 릿지와 비교한 시각화이다. 람다 값이 증가함에 따라 0에 수렴하는 베타가 늘어난다.





- 이를 Bias-Variance Trade off 관점에서 이해해보자. 람다가 작아지면 LSE와 비슷하면서 Low bias, High Variance한 형태이다. 람다가 커지면 High bias, Low variance의 형태이다. 따라서 MSE가 최소가 되는 적절한 람다를 찾는 것이 중요하다.

- Properties of Lasso

  릿지와 라쏘의 차이를 비교하면서 이해해볼까?

  - 라쏘는 릿지와 동일하게 편향을 일부분 허용하면서 추정량의 분산을 줄임으로써 전체 MSE를 작게하는 방법이다.
  - 라쏘는 일단 미분이 불가능하다. 그에 따라 LSE나 Ridge처럼 딱 정해진 형태가 없다. 이를 closed form이 없다고 한다. 그렇기 때문에 라쏘의 경우는 수치적으로 해를 찾게 되고, 가장 기본적인 수치적 방법으로는 경사하강법이 있다.
  - 라쏘 또한 릿지와 마찬가지로 n <<< p인 high-dimension의 경우에도 유일한 해를 갖는다.
  - 라쏘는 람다값이 증가함에 따라 정확히 0이 되는 베타가 존재하는 반면, 릿지는 정확히 0이 되는 베타가 없다. 따라서 라쏘는 변수선택의 효과가 있다.
  - 라쏘는 Best Subset selection의 convex approximation이다. 그에 따라 변수선택법 대신에 많이 쓰이며, 변수선택법의 경우에는 각각을 다 적합해보고 결정하지만, 라쏘는 최적화 방법에 의해 유일해에 접근하기 때문에 훨씬 빠르다.
  - 라쏘 추정량은 biased estimator이다. 하지만 릿지와 다르게 consistent estimator가 아니다. 관측치가 많다고 해도 참값에 다가서지 못한다. 이를 위해 adaptive lasso라는 방법이 존재한다.
  - 라쏘는 릿지에 비해 변수들 간의 상관관계가 클 경우 변수선택의 성능과 예측 성능 모두 다소 감소한다. 이를 위해 elastic net를 고려하게 되는 맥락도 존재한다.
    다음은 고려대 김성범 산공과 교수 youtube 내용인데, 관련 내용 관심있으면 참고하세요.
    https://www.youtube.com/watch?v=sGTWFCq5OKM

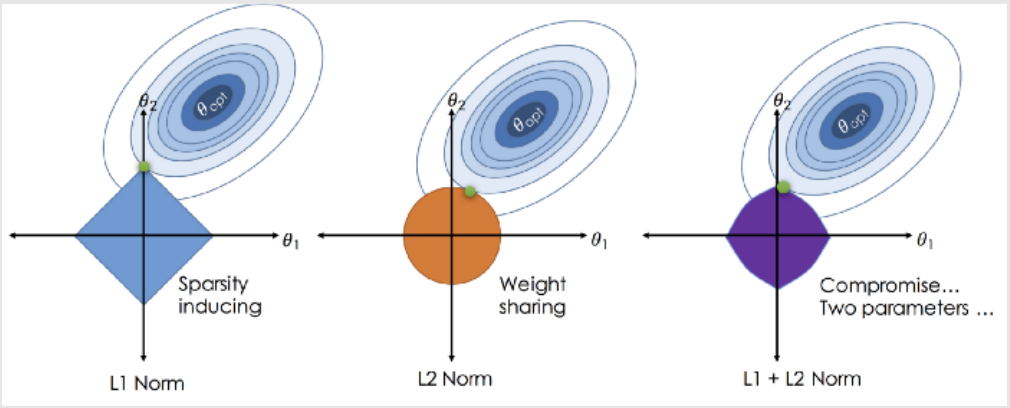
- 람다 정하기
  - 전체적인 과정은 릿지와 동일하다. 개별 변수들을 scaling해준다. 그런 다음 후보가 될 람다 값들에 대해 CV error를 구하고, 이때 최소의 CV error를 만다는 람다를 선택한다. 해당 람다를 가지고 전체 train set에 대해 적합해서 우리의 최종 베타(모델)을 만든다. R과 Python을 통해 확인하자.

- Elastic net

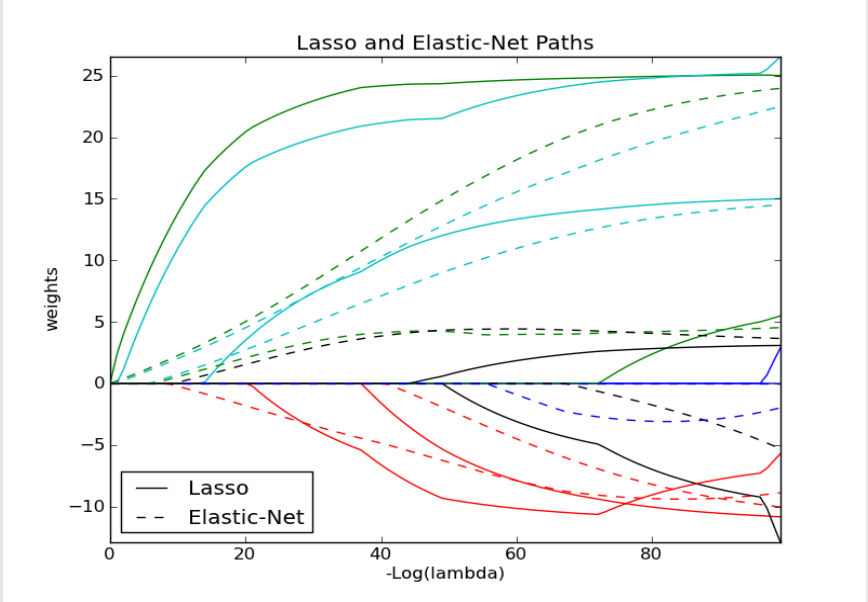
- Elastic net은 Ridge와 Lasso를 동시에 고려하는 모델이다. 릿지 페널티와 라쏘 페널티에 가중치를 줌으로써, 더 나은 모델을 고민하는 방법이다. 크게 다른 내용이 없다! 다만 라쏘와 릿지를 섞으면 변수선택하는 능력이 조금 떨어지겠지? 전체적으로 베타를 수축시키면서, 0도 만드는 방법이다. 그냥 Dual 형태만 보자!

$$\hat{\beta}^{elastic} = min_{\beta} \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\left(\alpha\sum_{j=1}^{p}|\beta_j| + (1-\alpha)\sum_{j=1}^{p}\beta_j{}^2\right), \qquad \lambda \geq 0, \qquad 0 \leq \alpha \leq 1$$

- alpha라는 새로운 tuning parameter가 생겼다. alpha=0이면 릿지고, alpha=1이면 라쏘겠지? 알파의 기준은 패키지마다 기준이 다를 수 있으니 확인해야 한다! 알파 값을 조정해줌으로써 적절한 가중치를 찾는다.



- 베타가 작아지는 형태를 보면 라쏘와 같이 정확히 0인 베타값이 존재하지만, 가중치 alpha에 따라 0으로 떨어지는 속도가 느릴 것이다. 다음과 같은 시각화를 보면 이해할 수 있다.



- 상관관계가 큰 변수들 간에는 유사한 베타값이 나오도록 추정하는 성질이 있다. 이를 Grouping effect라고 하고, 관련 수식이 있지만 생략한다.

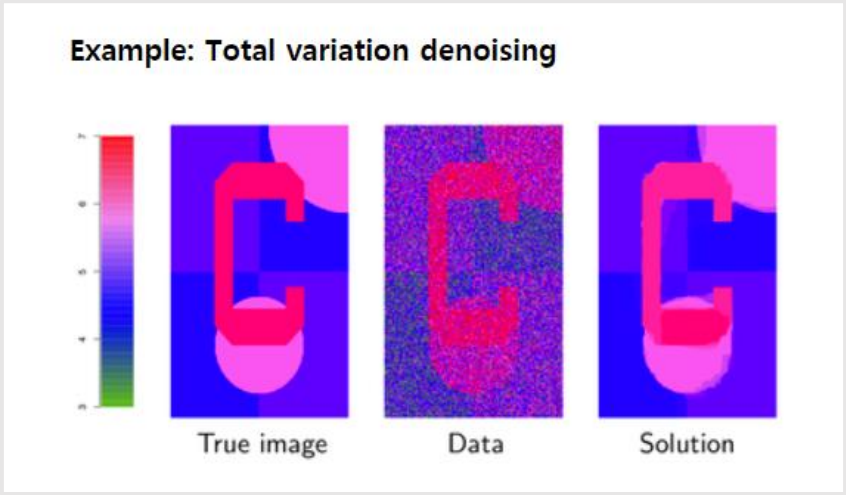
6. Another Regularization

다음의 내용은 피피티에 담기지 않을 그냥 제약에 대한 이해를 돕기 위한 자료입니다. 또다른 제약의 형태에는 무엇이 있고, 이런 제약들이 왜 중요한지 이야기하려 합니다.

- Fused Lasso
  - Fused Lasso는 순서, 형태를 고려하는 라쏘 방법이다. 물리적인 형태를 고려하는 것이다.

$$\hat{\beta}^{FL} = min_{\beta} \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda_1 \sum_{j=1}^{p}|\beta_j| + \lambda_2 \sum_{j=2}^{p}|\beta_j - \beta_{j-1}|$$

- 뒤의 텀이 추가된 형태이다. 상관관계와 관련없이 물리적으로 인접한 변수들을 가깝게 묶어주는 것이다. 물리적으로 인접한 변수들은 비슷한 효과를 고려하고 싶은 것!
- 이러한 형태는 신호처리, 아니면 이미지 처리에서 종종 쓰이는 페널티라고 알고 있다.



- Group Lasso
  - Group Lasso는 분석가가 정의한 그룹 단위로 변수를 선택/배제하기 위한 방법이다. 우리 보통 변수선택법 했을 때, 범주를 원핫 인코딩해서 넣은 것들 중에서 몇 개만 변수 선택되면 되게 난감했던 경험들이 있을 것이다. 이런 상황에 고려할 수 있는 방법이다. 같은 범주의 그룹들을 묶음으로써, 뽑으면 한 번에 뽑힐 수 있도록 해주는 방법이다. 꼭 범주가 아니어도, 같은 그룹의 변수들을 묶어주는 형태라고 이해하자!
  - 수식은 생략한다.
- Regularization
  - 이렇듯 Ridge, Lasso, Elastic net 이외에도 이렇게 다양한 페널티들이 있고, 소개하지 않은 더 다양한 페널티들이 존재한다. 우리가 정의한 상황에 맞게 페널티를 선택해서 사용할 수 있어야 한다. 물론 우리 수준에서는 함수가 구현되어 있는 선에서 사용해야겠죠? ㅠ 우리는 직접 이 문제를 최적화할 능력이 없습니다! 그냥 경사하강법으로는 수렴하지 않기 때문이죠….
  - 3주차에 이전에 아무도 다루지 않았던 Convex Optimization의 관점에서 Shrinkage Method들을 다룬 것은 의미있는 접근 방법이라고 생각합니다. 사실 대부분의 사람들은 Dual에서 원과 마름모 시각화를 통해 변수 선택하고 못하고를 접하다 보니, 두 형태를 하나로 이해하기 어려웠을 것이라고 생각합니다. 하지만 이런 Primal을 Dual로 더 간단하게 바꿔서 우리 문제를 바라보고 해결할 수 있다는 점을 통해 좀 더 Shrinkage Method를 더 잘 이해할 수 있을 것입니다. 이런 Dual로 푸는 문제는 SVM의 방법이기도 합니다만, SVM은 더 어렵습니다.
  - 사실 부스팅이나 딥러닝 모델에서도 이런 정규화 방법은 많이 쓰입니다. 우리가 회귀분석에서 배운 정규화 부분에 잘 이해하고 있다면, 부스팅이나 딥러닝 모델 튜닝할 때 이런 정규화도 바로 고려할 수 있습니다. 딥러닝의 경우에는 오버피팅을 막기 위해 매우 다양한 것을 고려할 수 있고, 관련한 방법들 중에 뭐가 좋은지에 대해서는 되게 많은 연구가 이루어지고 있다고 알고 있고, 특히 드랍아웃이 좋다고 알고 있습니다. 부스팅에서는 이런 과적합이 물론 천천히 일어나지만, 우리가 튜닝하는 이유는 결국에는 과적합 막으려는 이유 아닐까요? 그러니까 다른 기본적인 튜닝과 함께 먼저 고려해야 하는 튜닝이 이 L1, L2 penalty가 아닌가 생각합니다. 제일 직관적으로 간단하게 과적합을 방지할 수 있는 방법이 아닐까요?