# Chapter 05. 회귀

## 01. 회귀 소개

회귀 분석: 데이터 값이 평균과 같은 일정한 값으로 돌아가려는 경향을 이용한 통계학 기법

- 여러 개의 독립변수와 한 개의 종속변수 간의 상관관계를 모델링
- Y = W1X1 + W2X2 + ... + WnXn
    - Y: 종속변수(결정 값)
    - Xn: 독립변수(피처)
    - Wn: 독립변수의 값에 영향을 미치는 회귀 계수
- 머신러닝 회귀 예측의 핵심은 **최적의 회귀계수**를 찾아내는 것
- 회귀의 유형
    - 독립변수 개수에 따라
        - 1개: 단일 회귀
        - 여러 개: 다중 회귀
    - 회귀 계수의 결합
        - 선형: 선형 회귀
        - 비선형: 비선형 회귀
- 분류 vs 회귀: 분류는 예측값이 이산형 클래스 값, 회귀는 연속형 숫자 값

선형 회귀: 실제 값과 예측값의 차이를 최소화하는 직선형 회귀선을 최적화

- 규제: 선형 회귀의 과적합 문제를 해결하기 위해 회귀 계수에 페널티 값을 적용
- 규제 방법에 따른 유형 구분
    - 일반 선형 회귀: 예측값과 실제 값의 RSS를 최소화할 수 있도록 회귀 계수를 최적화, 규제 적용 X
    - 릿지: 선형 회귀에 L2 규제를 추가
        - L2 규제: 상대적으로 큰 회귀 계수 값의 예측 영향도를 감소시키기 위해 회귀 계수값을 더 작게 만듦
    - 라쏘: 선형 회귀에 L1 규제를 적용
        - L1 규제: 예측 영향력이 작은 피처의 회귀 계수를 0으로 만들어 회귀 예측 시 피처가 선택되지 않게 함
    - 엘라스틱넷: L2, L1 규제를 함께 결합
    - 로지스틱 회귀: 매우 강력한 분류 알고리즘

## 02. 단순 선형 회귀를 통한 회귀 이해

단순 선형 회귀: 독립변수와 종속변수가 모두 하나

- 예측값 Y = w0 + w1X
  - w0, w1: 회귀 계수
  - 잔차: 실제 값과 회귀 모델의 차이에 따른 오류 값
- 최적의 회귀 모델: 잔차 합이 최소가 되는 모델
  - 오류 합 구하는 방식
    - Mean Absolute Error: 절댓값을 취해서 더함
    - Residual Sum of Square(RSS): 오류 값의 제곱을 더함
  - RSS를 최소로 하는 회귀 계수를 찾는 것이 머신러닝 기반 회귀의 핵심!
    - 비용함수: w 변수로 구성되는 RSS

$$RSS(w_0, w_1) = \frac{1}{N}\sum_{i=1}^{N}(y_i - (w_0 + w_1 * x_i))^2$$

(i는 1부터 학습 데이터의 총 건수 N까지)

## 03. 비용 최소화하기 - 경사 하강법 소개

경사 하강법: '점진적으로' 반복적인 계산을 통해 W 파라미터 값을 업데이트하면서 오류 값이 최소가 되는 W 파라미터를 구하는 방식

- 고차원 방정식을 푸는 것보다 훨씬 직관적이고 빠르게 비용 함수가 최소가 되는 W 파라미터 값 구하기 가능
- 반복적으로 비용 함수의 반환 값이 작아지는 방향성을 가지고 W 파라미터를 지속해서 보정
  - 오류 값이 더 이상 작아지지 않으면 그 오류 값을 최소 비용으로 판단, 그때의 W값을 최적 파라미터로 반환
- R(w)를 미분해서 미분 함수의 최솟값을 구해야 함 (편의상 RSS(w0,w1)를 R(w)로 지칭)
  - w0, w1 각 변수에 편미분 적용

$$\frac{\partial R(w)}{\partial w_1} = \frac{2}{N}\sum_{i=1}^{N} -x_i * (y_i - (w_0 + w_1 x_i) = -\frac{2}{N}\sum_{i=1}^{N} x_i * (\text{실제값}_i - \text{예측값}_i)$$

$$\frac{\partial R(w)}{\partial w_0} = \frac{2}{N}\sum_{i=1}^{N} -(y_i - (w_0 + w_1 x_i) = -\frac{2}{N}\sum_{i=1}^{N} (\text{실제값}_i - \text{예측값}_i)$$

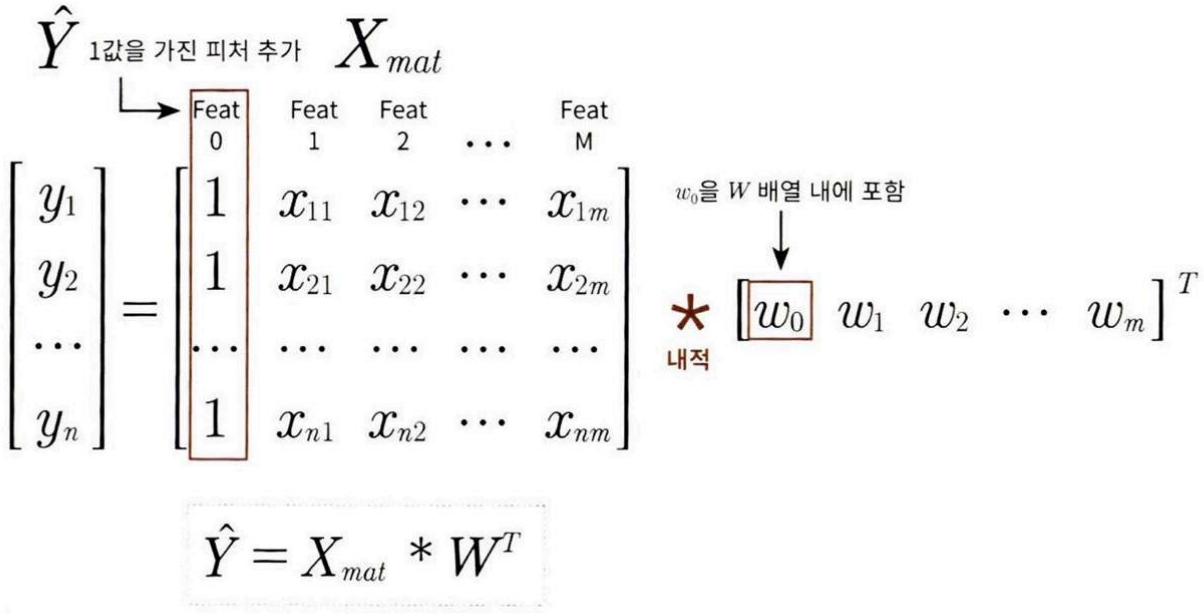
- 경사 하강법의 일반적인 프로세스
  - Step 1: $w_1, w_0$를 임의의 값으로 설정하고 첫 비용 함수의 값을 계산합니다.
  - Step 2: $w_1$을 $w_1 + \eta \frac{2}{N} \sum_{i=1}^{N} x_i * (\text{실제값}_i - \text{예측값}_i)$, $w_0$을 $w_0 + \eta \frac{2}{N} \sum_{i=1}^{N} (\text{실제값}_i - \text{예측값}_i)$으로 업데이트한 후 다시 비용 함수의 값을 계산합니다.
  - Step 3: 비용 함수의 값이 감소했으면 다시 Step 2를 반복합니다. 더 이상 비용 함수의 값이 감소하지 않으면 그때의 $w_1, w_0$를 구하고 반복을 중지합니다.

확률적 경사 하강법: 전체 입력 데이터로 w가 업데이트되는 값을 계산하는 것이 아니라 일부 데이터만 이용해 w가 업데이트되는 값을 계산

- 수행 시간이 오래 걸린다는 경사 하강법의 단점을 보완

예측값 Y = w0 + w1X1 + w2X2 + ... + wmXm

- 피처가 M개, 회귀 계수가 M+1개




## 04. 사이킷런 LinearRegression을 이용한 보스턴 주택 가격 예측

LinearRegression 클래스: 예측값과 실제 값의 RSS를 최소화해 OLS 추정 방식으로 구현

- fit( ) 메서드로 X,y 배열을 입력 받아 회귀 계수인 W를 coef_ 속성에 저장
- 입력 파라미터
  - fit_intercept: 절편 값을 계산할 것인지 말지를 지정
  - normalize: fit_intercept가 True일 때만, 회귀를 수행하기 전에 입력 데이터 세트를 정규화
- 속성
  - coef_: fit() 메서드를 수행했을 때 회귀 계수가 배열 형태로 저장

    - intercept_: intercept값
- Ordinary Least Squares 기반의 회귀 계수 계산은 입력 피처의 독립성에 많은 영향을 받음
    - 다중 공선성: 피처 간의 상관관계가 매우 높은 경우 분산이 매우 커져서 오류에 민감

회귀 평가 지표

- MAE: 실제 값과 예측값의 차이를 절댓값으로 변환해 평균
- MSE: 실제 값과 예측값의 차이를 제곱해 평균
- RMSE: MSE에 루트를 씌운 것
- R제곱: 실제 값의 분산 대비 예측값의 분산 비율을 지표로 함

## 05. 다항 회귀와 과(대)적합/과소적합 이해

다항 회귀: 회귀가 독립변수의 다항식이 아닌 2차, 3차 방정식과 같은 다항식으로 표현

- y = w0 + w1x1 + w2x2 + w3x1x2 + w4x1^2 + w5x2^2
- 다항 회귀는 **선형 회귀**!!
- 사이킷런은 PolynomialFeatures로 피처를 변환한 후에 LinearRegression 클래스로 다항 회귀를 구현

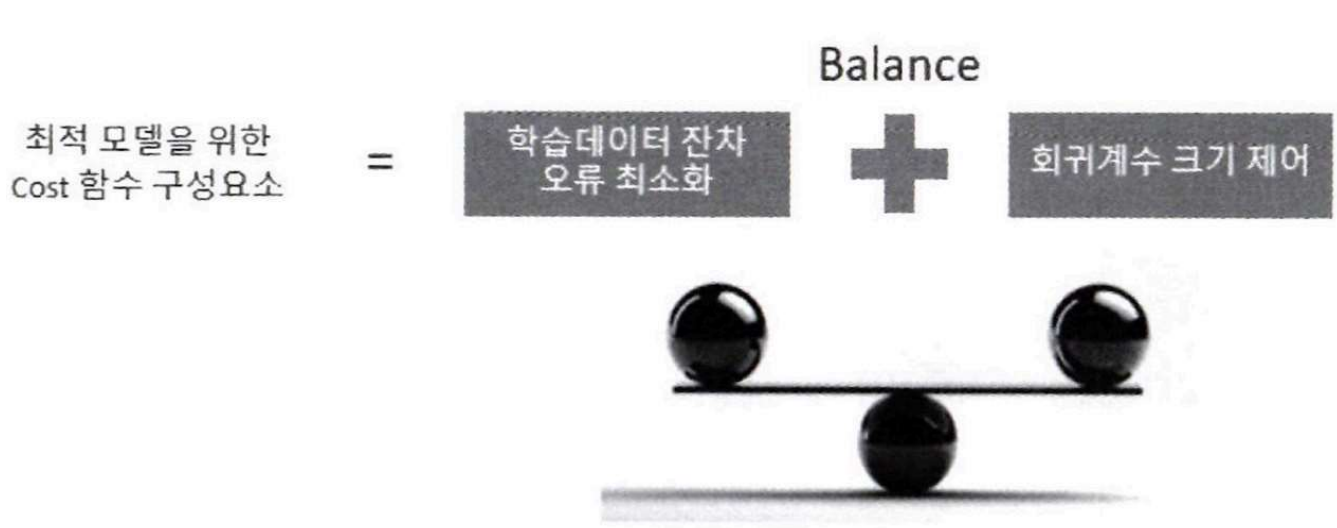
다항 회귀를 이용한 과소적합 및 과적합 이해

- 차수(degree)를 높일수록 학습 데이터에만 너무 맞춘 학습이 이루어져서 정작 테스트 데이터 환경에서는 오히려 예측 정확도가 떨어짐
- 좋은 예측 모델: Degree 1처럼 패턴을 지나치게 단순화한 과소적합 모델도, Degree 15처럼 패턴 하나하나 감안한 지나치게 복잡한 과적합 모델도 아닌 균형 잡힌 모델

편향-분산 트레이드오프

- '양궁 과녁' 그래프
    - 저편향/저분산: 예측 결과가 실제 결과에 매우 근접하면서도 예측 변동이 크지 않고 특정 부분에 집중돼 있는 아주 뛰어난 성능을 보여줌
    - 저편향/고분산: 예측 결과가 실제 결과에 비교적 근접, 예측 결과가 실제 결과를 중심으로 꽤 넓은 부분에 분포
    - 고편향/저분산: 정확한 결과에서 벗어나면서도 예측이 특정 부분에 집중
    - 고편향/고분산: 정확한 예측 결과를 벗어나면서도 넓은 부분에 분포
- 일반적으로 편향과 분산은 한 쪽이 높으면 한 쪽이 낮아지는 경향
- '골디락스' 지점: 편향을 낮추고 분산을 높이면서 전체 오류가 가장 낮아지는 지점
- 편향과 분산이 서로 트레이드오프를 이루면서 오류 값이 최대로 낮아지는 모델을 구축하는 것이 가장 효율적

## 06. 규제 선형 모델 - 릿지, 라쏘, 엘라스틱넷




$$\text{비용 함수 목표} = \text{Min}(\text{RSS(W)} + \text{alpha} * \| W \|_2^2)$$

- alpha: 학습 데이터 적합 정도와 회귀 계수 값의 크기 제어를 수행
  - alpha 값을 크게 하면 비용 함수는 회귀 계수 W의 값을 작게 해 과적합 개선
  - alpha 값을 작게 하면 회귀 계수 W의 값이 커져도 어느 정도 상쇄 가능, 적합 더 개선
- 규제: 비용 함수에 alpha 값으로 페널티를 부여해 회귀 계수 값의 크기를 감소시켜 과적합 개선
  - L2 규제: W의 제곱에 대해 페널티 부여 - 릿지 회귀
  - L1 규제: W의 절댓값에 대해 페널티 부여 - 라쏘 회귀
    - 적절한 피처만 회귀에 포함시킴

릿지 회귀

- 주요 생성 파라미터는 alpha, 이는 릿지 회귀의 alpha L2 규제 계수

라쏘 회귀

- 주요 생성 파라미터는 alpha, 이는 라쏘 회귀의 alpha L1 규제 계수

엘라스틱넷 회귀: L2 규제와 L1 규제를 결합한 회귀

- 주요 생성 파라미터는 alpha와 l1_ratio
- 규제는 a*L1 + b*L2로 정의
  - a는 L1 규제의 alpha 값, b는 L2 규제의 alpha 값
- alpha의 파라미터 값은 a+b, l1_ratio의 파라미터 값은 a/(a+b)

선형 회귀 모델을 위한 데이터 변환

- 선형 모델은 일반적으로 피처와 타깃값 간에 선형의 관계가 있다고 가정하고, 최적의 선형함수를 찾아내 결괏값을 예측
- 선형 회귀 모델은 피처값과 타깃값의 정규 분포 형태를 매우 선호

    - 타깃값의 경우 왜곡된 형태의 분포도일 경우 예측 성능에 부정적인 영향을 미칠 수 있음
- 선형 회귀 모델을 적용하기 전에 데이터에 대한 스케일링/정규화 작업을 수행하는 것이 일반적
    - 사이킷런을 이용해 피처 데이터 세트에 적용하는 변환 작업

1. StandardScaler 클래스를 이용해 평균이 0, 분산이 1인 표준 정규 분포를 가진 데이터 세트로 변환하거나 MinMaxScaler 클래스를 이용해 최솟값이 0이고 최댓값이 1인 값으로 정규화를 수행합니다.
2. 스케일링/정규화를 수행한 데이터 세트에 다시 다항 특성을 적용하여 변환하는 방법입니다. 보통 1번 방법을 통해 예측 성능에 향상이 없을 경우 이와 같은 방법을 적용합니다.
3. 원래 값에 log 함수를 적용하면 보다 정규 분포에 가까운 형태로 값이 분포됩니다. 이러한 변환을 로그 변환(Log Transformation)이라고 부릅니다. 로그 변환은 매우 유용한 변환이며, 실제로 선형 회귀에서는 앞에서 소개한 1, 2번 방법보다 로그 변환이 훨씬 많이 사용되는 변환 방법입니다. 왜냐하면 1번 방법의 경우 예측 성능 향상을 크게 기대하기 어려운 경우가 많으며 2번 방법의 경우 피처의 개수가 매우 많을 경우에는 다항 변환으로 생성되는 피처의 개수가 기하급수로 늘어나서 과적합의 이슈가 발생할 수 있기 때문입니다.

## 07. 로지스틱 회귀

로지스틱 회귀: 선형 회귀 방식을 분류에 적용한 알고리즘

- 시그모이드 함수 최적선을 찾고 이 시그모이드 함수의 반환 값을 확률로 간주해 확률에 따라 분류를 결정
    - 시그모이드 함수는 x값이 +,-로 아무리 커지거나 작아져도 y값은 항상 0과 1사이 값을 반환, x값이 커지면 1에 근사하며 x값이 작아지면 0에 근사
- 사이킷런 LogisticRegression 클래스의 주요 하이퍼 파라미터는 penalty와 C
    - penalty: 규제의 유형을 설정
    - C: 규제 강도를 조절하는 alpha값의 역수, 작을수록 규제 강도가 큼
- 로지스틱 회귀는 가볍고 빠르지만, 이진 분류 예측 성능도 뛰어남

## 08. 회귀 트리

회귀 트리: 회귀를 위한 트리를 생성하고 이를 기반으로 회귀 예측을 하는 것

- 리프 노드에 속한 데이터 값의 평균값을 구해 회귀 예측값을 계산
- 결정 트리, 랜덤 포레스트, GBM, XGBoost 등의 트리 기반 알고리즘은 분류 뿐만 아니라 회귀도 가능
    - 트리 생성이 CART 알고리즘에 기반하고 있기 때문
    - Estimator 클래스: 결정 트리, 랜덤 포레스트, GBM
    - 래퍼 클래스: XGBoost, LightGBM
- 회귀 트리 Regressor 클래스는 회귀 계수를 제공하는 coef_ 속성이 없음
    - feature_importances_를 이용해 피처별 중요도를 알 수 있음

- 회귀 트리는 분할되는 데이터 지점에 따라 브랜치를 만들면서 계단 형태로 회귀선을 만듦