지도학습 -> 분류(예측 값이 카테고리와 같은 이산형 클래스 값)와 회귀(연속형 숫자 값)

**회귀**

회귀는여러 개의 독립변수와 한 개의 종속변수 간의 상관관계를 모델링 하는 기법

**선형 회귀식** Y = W1\*X1 + W2\*X2 + W3\*X3 + … 에서 Y는 **종속변수**, X1, X2, …는 **독립변수**, W1, W2, W3, … 는 독립변수 값에 영향을 미치는 **회귀 계수**

|  |  |
| --- | --- |
| **독립변수 개수** | **회귀 계수의 결합** |
| 1개: 단일 회귀 | 선형: 선형 회귀 |
| 여러 개: 다중 회귀 | 비선형: 비선형 회귀 |

머신러닝 관점에서 독립변수는 피처, 종속변수는 결정 값 🡪 회귀 예측의 핵심은 주어진 피처와 결정 값 데이터 기반에서 학습을 통해 최적의 회귀 계수를 찾아내는 것

선형회귀가 가장 많이 사용됨 🡪 선형회귀는 실제 값과 예측 값의 차이(오류의 제곱 값)를 최소화하는 직선형 회귀선을 최적화하는 방식, 규제(과적합 문제 해결하기 위해 회귀계수에 패널티 적용) 방법에 따라 다음과 같이 나뉨

1. 일반 선형 회귀: 예측 값과 실제 값의 RSS(Residual Sum of Squares)를 최소화할 수 있도록 회귀 계수 최적화, 규제 X
2. 릿지(Ridge): 선형 회귀에 L2 규제 추가한 회귀 모델, 상대적으로 큰 회귀 계수 값의 예측 영향도를 감소시키기 위해 회귀 계수 값을 더 작게 만드는 규제 모델
3. 라쏘(Lasso): 선형 회귀에 L1 규제 적용한 모델, 예측 영향력이 작은 피처의 회귀 계수를 0으로 만들어 회귀 예측 시 피처가 선택되지 않도록 함
4. 엘라스틱넷(ElasticNet): L2, L1 규제를 함께 결합한 모델, 피처가 많은 데이터 세트에 적용, L1 규제로 피처의 개수 줄이면서 L2 규제로 계수 값 조정
5. 로지스틱 회귀(Logistic Regression): 분류에 사용되는 선형 모델, 이진 분류/텍스트 분석

**단순 선형 회귀**: 독립 변수도 하나, 종속 변수도 하나인 선형 회귀

예측 값: Y = w0 + w1 \* X 🡪 실제 값: w0 + w1 \* X + 오류 값

잔차: 실제 값과 회귀 모델의 차이에 따른 오류 값, 남은 오류

최적의 회귀 모델 만드는 것은 잔차 합이 최소가 되는 모델 만드는 것

오류 합을 계산할 때는 절댓값을 취해서 더하거나(Mena Absolute Error), 오류 값의 제곱을 구해서 더하는 방식(Residual Sum of Square) 🡪 미분 등의 계산 편리하게 하기 위해서 RSS

**RSS를 최소로 하는 w0, w1 즉 회귀 계수를 찾는 것이 핵심**

RSS(, ) = 🡪 비용 함수, 손실 함수

**이 비용 함수가 반환하는 값(즉, 오류 값)의 최소를 구하는 것**

**경사 하강법(Gradient Descent)**

점진적으로 반복 계산 통해 w 파라미터 값을 업데이트 하면서 오류 값이 최소가 되는 w 파라미터 구하는 방식

RSS(, ) =

편미분 결과

w1 값의 업데이트는 새로운 w1을 이전 w1에서 편미분 결과값을 빼면서 적용 🡪 편미분 결과 값이 너무 클 수 있기 때문에 보정 계수를 곱하는데 이를 ‘학습률’이라고 함.

비용 함수의 값이 감소했으면 업데이트 반복하고 더 이상 비용 함수의 값이 감소하지 않으면 그때의 w1, w0 구하고 반복 중지

경사 하강법은 수행 시간이 오래 걸려서 실전에서는 확률적(Stochastic) 경사 하강법 이용

일부 데이터만 이용해 w가 업데이트되는 값을 계산하므로 경사 하강법에 비해서 빠른 속도 보장

🡪 확률적 경사 하강법이나 미니 배치 확률적 경사 하강법 이용해 최적 비용함수 도출

**독립변수, 즉 피처가 여러 개인 경우**

피처가 M개 있다면 회귀 계수는 M+1개 도출

Y = w0 + w1 \* X1 + w2 \* X2 + … + wM \* XM

입력 행렬을 Xmat, 회귀 계수들로 이루어진 배열을 W라 하면 예측 행렬 Y = np.dot(Xmat, WT) + w0

* w0을 W안에 포함시키기 위해 Xmat 맨 처음 열에 모든 데이터 값이 1인 피처를 추가하면 회귀 예측 값 Y = Xmat \* WT

**LinearRegression**

선형 모델 중 규제가 적용되지 않은 선형 회귀를 사이킷런에서 구현한 클래스

입력 파라미터 LinearRegression (fit\_intercept=True, normalize=False, copy\_X=True, n\_jobs=1)

* fit\_intercept: 절편 값을 계산할 것인지 말지를 지정, False면 절편 값 0으로 지정
* normalize: fit\_intercept가 False이면 무시됨, fit\_intercept가 True이면 회귀 수행하기 전에 입력 데이터 세트를 정규화함.

속성

* coef\_: fit()메서드 수행했을 때 회귀 계수가 배열 형태로 저장하는 속성

shape는 (target 값 개수, 피처 개수)

* intercept\_: intercept 값

Ordinary Least Squares 기반의 회귀 계수 계산은 입력 피처의 독립성에 많은 영향을 받음, 피처 간의 상관관계가 매우 높은 경우 분산이 커져서 오류에 민감해짐 🡪 다중 공선성(multi-collinearity) 문제

🡪 독립적인 중요한 피처만 남기고 제거하거나 규제 적용/PCA를 통해 차원 축소 수행

**회귀 평가 지표**

실제 값과 회귀 예측 값의 차이 값을 기반으로 한 지표

MAE (Mean Absolute Error): 차이를 절댓값으로 변환해 평균

MSE (Mean Squared Error): 차이를 제곱해 평균

RMSE (Root Mean Squared Error): MSE에 루트를 씌운 것

R2: 실제 값의 분산 대비 예측 값의 분산 비율, 1에 가까울수록 예측 정확도 높음

MSLE(Mean Squared Log Error), RMSLE(Root Mean Squared Log Error) 등

|  |  |  |
| --- | --- | --- |
| 평가 방법 | 사이킷런 평가 지표 API | Scoring 함수 적용 값 |
| MAE | metrics.mean\_absolute\_error | ‘neg\_mean\_absolute\_error’ |
| MSE | metrics.mean\_squared\_error | ‘neg\_mean\_squared\_error’ |
| R2 | metrics.r2\_score | ‘r2’ |

Crow\_val\_score, GridSearchCV 같은 Scoring 함수에 회귀 평가 지표 적용할 때 유의점

Scoring 함수에 ‘neg\_mean\_absolute\_error’ 적용해 음수 값을 반환하는 이유는 Scoring 함수가 score 값이 클수록 좋은 평가 결과로 자동 평가하기 때문이다. 회귀 평가 지표의 경우 값이 커지면 오히려 나쁜 모델이라는 의미이므로 Scoring 함수에 일반적으로 반영하려면 보정 필요

🡪 ‘neg\_mean\_absolute\_error’ = -1 \* metrics.mean\_absolute\_error

**다항 회귀**

다항식으로 표현되는 것

Y = w0 + w1\*x1 + w2\*x2 + w3\*x1\*x2 + w1\*x12 + …

**다항 회귀는 선형 회귀임**. 🡪 선형/비선형 회귀는 회귀 계수가 선현/비선형인지에 따른 것이지 독립 변수의 선형/비선형 여부와는 무관

사이킷런은 다항 회귀를 위한 클래스를 명시적으로 제공 X, but 비선형 함수를 선형 모델에 적용시키는 방법을 사용해 구현 🡪 PolynomialFeatures 클래스를 통해 피처를 다항식 피처로 변환

🡪 PolynomialFeatures 클래스는 degree 파라미터를 통해 입력 받은 단항식 피처를 degree에 해당하는 다항식 피처로 변환

다항 회귀로 복잡한 피처 간의 관계까지 모델링이 가능하지만 다항 회귀 차수를 높일수록 과적합의 문제가 발생함(학습 데이터에만 너무 맞춘 학습 이루어져서 테스트 데이터에는 예측 정확도 떨어짐)

**편향-분산 Trade off**

예제의 Degree 1같은 모델은 지나치게 한 방향성으로 치우친 경향 🡪 저분산, 고편향, High Bias

Degree 15는 학습데이터 모든 특성을 반영하여 매우 복잡한 모델 🡪 저편향, 고분산, High Variance

편향과 분산은 한 쪽이 높으면 한 쪽이 낮아지는 경향이 있음.

* 편향이 너무 높으면 전체 오류가 높음(과소적합 되기 쉬움)
* 편향을 점점 낮추면 동시에 분산 높아지고 전체 오류가 낮아짐.
* 전체 오류가 가장 낮아지는 ‘골디락스’ 지점 통과하면서 분산 계속 높이면 전체 오류 값이 오히려 증가하면서 예측 성능 저하됨(과적합 되기 쉬움)
* **편향과 분산이 서로 트레이드오프 이루면서 오류 값이 최대로 낮아지는 모델 구축해야 함**

**규제 선형 모델 - Ridge, Lasso, Elastic Net**

이전 선형 모델의 비용함수는 RSS를 최소화하는 것만 고려해서 학습 데이터에 지나치게 맞추게 되고, 회귀 계수가 쉽게 커짐 🡪 변동성이 심해져 테스트 데이터에는 예측 성능 저하되기 쉬움

🡪 학습 데이터의 잔차 오류 값을 최소로 하는 RSS 최소화 방법 + 회귀 계수 값 커지지 않도록 제어

🡪 비용 함수 목표 = Min(RSS(W) + alpha \* )

🡪 alpha는 학습 데이터 적합 정도와 회귀 계수 값 크기 제어를 수행하는 튜닝 파라미터

alpha 0 또는 매우 작은 값: 비용 함수 = Min(RSS(W) + 0)

alpha 무한대 또는 매우 큰 값: RSS(W)에 비해 alpha \* 가 너무 커지게 되므로 W값을 0 또는 매우 작게 만들어야 cost 최소화

* alpha를 크게 하면 W의 값을 작게 해 과적합 개선하고 alpha를 작게 하면 W의 값이 커져도 어느정도 상쇄가 가능하므로 학습 데이터 적합 개선 가능

즉, alpha를 0에서부터 지속적으로 값을 증가시키면서 회귀 계수 값을 감소시킬 수 있음.

**비용함수에 alpha 값으로 페널티 부여해 회귀 계수 값의 크기를 감소시켜 과적합 개선하는 방식을 규제라고 함**

1. L2 규제: alpha \* 와 같이 W의 제곱에 대해 페널티 부여하는 방식 🡪 **릿지**
2. L1 규제: alpha \* 와 같이 W의 절댓값에 대해 페널티 부여 🡪 영향력이 크지 않은 회귀 계수 값을 0으로 변환 🡪 **라쏘**

**릿지**

사이킷런은 Ridge 클래스 통해 릿지 회귀 구현

🡪 주요 생성 파라미터 alpha(릿지 회귀의 alpha L2 규제 계수)

🡪 alpha 값이 증가하면서 회귀 계수가 지속적으로 작아지지만 회귀 계수를 0으로 만들지는 않음

**라쏘**

라쏘 회귀 비용함수의 목표는 RSS(W) + alpha \* 식을 최소화하는 W를 찾는 것

L1 규제는 불필요한 회귀 계수를 급격하게 감소시켜 0으로 만들고 제거 🡪 적절한 피처만 회귀에 포함시키는 피처 선택의 특성 가짐

사이킷런은 Lasso 클래스를 통해 라쏘 회귀 구현

🡪 주요 생성 파라미터 alpha(라쏘 회귀의 alpha L1 규제 계수)

**엘라스틱넷 회귀**

L2규제와 L1규제를 결합한 회귀

엘라스틱넷 회귀 비용함수의 목표는 RSS(W) + alpha2 \* + alpha1 \* 식을 최소화하는 W 찾는 것

라쏘 회귀가 서로 상관관계가 높은 피처들 중에서 중요 피처만 선택하고 나머지는 모두 회귀계수를 0으로 만드는 성향이 강한데 이 성향으로 alpha값에 따라 회귀 계수 값이 급격히 변동할 수 있음

🡪 이를 완화하기 위해 L2 규제를 라쏘 회귀에 추가한 것, 그래서 수행시간이 상대적으로 오래 걸림

사이킷런은 ElasticNet 클래스를 통해 엘라스틱넷 회귀 구현

🡪 주요 생성 파라미터 alpha와 l1\_ratio

🡪 엘라스틱넷의 규제는 a \* L1 + b \* L2로 정의됨, a는 L1 규제의 alpha 값, b는 L2규제의 alpha 값

alpha 파라미터 값은 a+b, l1\_ratio 파라미터 값은 a / (a + b)

l1\_ratio가 0이면 a도 0이므로 L2 규제와 동일, l1\_ratio가 1이면 b는 0이므로 L1 규제와 동일

**선형 회귀 모델을 위한 데이터 변환**

선형회귀는 데이터 분포도의 정규화와 인코딩 방법이 매우 중요

선형모델은 일반적으로 피처와 타깃 값 간에 선형의 관계가 있다고 가정하고 최적의 선형함수를 찾아내 결과값 예측, 또한 피처 값과 타깃 값의 분포가 정규분포 형태를 선호함

분포도가 심하게 왜곡되었을 경우 변환 작업 수행

피처 데이터 세트에 적용하는 변환 작업

1. StandardScaler 클래스를 이용해 표준 정규분포 가진 데이터 세트로 변환하거나 MinMaxScaler 클래스를 이용해 최솟값이 0이고 최댓값이 1인 값으로 정규화 수행
2. 스케일링/정규화한 데이터 세트에 다시 다항 특성을 적용하여 변환
3. 로그변환 – 1번의 경우 예측 성능이 크게 향상되지 않으며 2번은 과적합의 이슈 발생 가능

타깃 값의 경우는 일반적으로 로그 변환 수행

* 일반적으로 선형 회귀 적용하려는 데이터 세트의 값의 분포가 왜곡되어 있을 경우에 로그 변환을 적용하는 것이 좋은 성능 가질 수 있음

**로지스틱 회귀**

**선형 회귀** 방식을 **분류**에 적용한 알고리즘

로지스틱 회귀가 선형 회귀와 다른 점은 학습을 통해 선형 함수의 회귀 최적선을 찾는 것이 아니라 시그모이드(Sigmoid) 함수 최적선을 찾고 이 시그모이드 함수의 반환 값을 확률로 간주해 확률에 따라 분류 결정

시그모이드 함수 🡪 x 값이 +, -로 아무리 커지거나 작아져도 y 값은 항상 0과 1사이 값을 반환

LogisticRegression의 주요 하이퍼 파라미터로 penalty, C가 있음

1. Penalty: 규제의 유형 설정 🡪 기본은 ‘l2’
2. C: 규제 강도를 조절하는 alpha 값의 역수, C 값이 작을수록 규제 강도 큼

로지스틱 회귀는 가볍고 빠르지만 이진 분류 예측 성능도 뛰어남

희소한 데이터 세트 분류에도 뛰어난 성능을 보여서 텍스트 분류에서도 자주 사용됨

**회귀 트리**

지금까지는 선형 회귀에 대해 배움, 선형회귀는 회귀 계수를 선형으로 결합하는 회귀 함수를 구해 결과 값 예측

비선형 회귀는 회귀 계수의 결합이 비선형인 비선형 회귀 함수를 통해 결과 값 예측함

트기 기반의 회귀는 회귀 트리를 이용하는 것, 회귀 위한 트리 생성하고 이를 기반으로 회귀 예측

🡪 분류 트리가 특정 클래스 레이블을 결정하는 것과는 달리 회귀 트리는 리프 노드에 속한 데이터 값의 평균값을 구해 회귀 예측 값을 계산

결정 트리 기반으로 분할하면 피처의 균일도를 반영한 지니 계수에 따라 분할 🡪 리프 노드 생성 기준에 부합하는 트리 분할이 완료되었다면 리프 노드에 속한 데이터 값이 평균 값을 구해서 최종적으로 리프 노드에 결정 값으로 할당

트리 생성이 CART(Classification And Regression Tree) 알고리즘에 기반하고 있기 때문에 모든 트리 기반 알고리즘은 분류뿐만 아니라 회귀도 가능

|  |  |  |
| --- | --- | --- |
| 알고리즘 | 회귀 Estimator 클래스 | 분류 Estimator 클래스 |
| Decision Tree | DecisionTreeRegressor | DecisionTreeClassifier |
| Gradient Boosting | GradientBoostingRegressor | GradientBoostingClassifier |
| XGBoost | XGBRegressor | XGBClassifier |
| LightGBM | LGBMRegressor | LGBMClassifier |

**정리**

선형회귀는 실제 값과 예측 값의 차이인 오류를 최소로 줄일 수 있는 선형 함수를 찾아서 이 선형 함수에 독립변수(피처) 입력해 종속변수(타깃 값, 예측 값)를 예측하는 것

비용 함수를 최소화하는 방법 중 하나: 경사 하강법

단순 선형회귀는 실제 값과 예측 값의 차이를 최소화하는 것에만 초점을 맞춰서 학습 데이터에 과적합 되는 문제를 수반할 가능성이 높음 🡪 규제 선형 회귀

선형 회귀를 분류에 적용한 대표적 모델은 로지스틱 회귀 🡪 선형 함수 대신 최적의 시그모이드 함수를 도출하고 독립변수를 이 함수에 입력해 반환 결과를 확률 값으로 변환해 예측 레이블 결정

회귀 트리: 분류 트리와 크게 다르지 않으나 리프 노드에 속한 데이터 값의 평균 값을 구해 회귀 예측 값을 계산하는 차이가 있음. 🡪 결정 트리, 랜덤 포레스트, GBM, XGBoost, LightGBM 모두 회귀 제공

선형 모델 기반으로 하는 선형 회귀는 데이터 값의 분포도와 인코딩 방법이 많은 영향을 줌

🡪 데이터 값의 분포도가 정규 분포 형태 선호, 타깃값은 왜곡 X & 정규분포

🡪 왜곡 되어있으면 보통 로그 변환 적용

🡪 카테고리형 데이터는 원-핫 인코딩