05 회귀

01 회귀 소개

회귀(Regression)은 **데이터값이 평균과 같은 일정한 값으로 돌아가려는 경향을 이용한 통계학 기법**이다.

$Y = W_1 * X_1 + W_2 * X_2 + W_3 * X_3 + ... + W_n * X_n$ 이라는 **선형회귀식**

Y는 **종속변수**, X는 **독립변수**, W는 **회귀 계수**이다.

이 때, **회귀 계수**를 구하는 것이 **회귀 예측의 핵심**이다.

여러가지 회귀 중에 **선형 회귀**가 가장 많이 사용된다.

대표적인 선형 회귀 모델

- 일반 선형 회귀 : 예측값과 실제값의 RSS(Residual sum of Squares)를 최소화 할 수 있도록 회귀계수를 최적화하며, 규제를 적용하지 않은 모델
- 릿지 : L2 규제 적용, 큰 회귀 계수 값의 예측 영향도 감소시키기 위해 회귀 계수값을 더 작게 만듦
- 라쏘 : L1 규제 적용, 예측 영향력이 작은 피처의 회귀 계수를 0으로 만들어 피처가 선택되지 않게 하는 것
- 엘라스틱넷 : L1, L2 결합, 피처가 많은 데이터 세트에 적용
- 로지스틱 회귀 : 분류에 사용되는 선형 모델, 뛰어난 예측 성능을 보임

02 단순 선형 회귀를 통한 회귀 이해

선형 회귀는 직선 형태로 표현이 가능하며, 예측값을 1차 함수식으로 표현이 가능하다.

실제 값과 회귀 모델의 차이에 따른 오류 값을 남은 오류, **잔차**라고 부른다.

최적의 회귀 모델 : 잔차 합이 최소가 되는 모델

RSS : 오류 값의 제곱의 합으로 구하는 방식

$$RSS = \sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

$$= \sum_{i=1}^{n}(Y_i - \hat{b}_0 - \hat{b}_1 X_i)^2$$

03 비용 최소화 하기 - 경사하강법

경사하강법

1. w_1, w_0 을 임의의 값으로 설정하고 첫 비용 함수(RSS)의 값을 계산한다.
2. w_1 을 기존 w_1 + 편미분 결과값, w_0 을 기존 w_0 + 편미분 결과값으로 업데이트한 후 다시 비용 함수의 값을 계산한다.
3. 비용 함수의 값이 감소했으면 다시 2번을 반복한다. 더 이상 감소하지않으면 그 때의 w_1, w_0 를 구하고 반복을 중지한다.

04 사이킷언 LinearRegression을 이용한 보스턴 주택 가격 예측

`LinearRegression()` 클래스

`fit()` 메서드로 X, y 배열을 입력 받으면 회귀 계수인 **W**를 `coef_` 속성에 저장합니다.

회귀 평가 지표

MAE : Mean Absolute Error이며 실제 값과 예측값의 차이를 절댓값으로 변환해 평균한 것

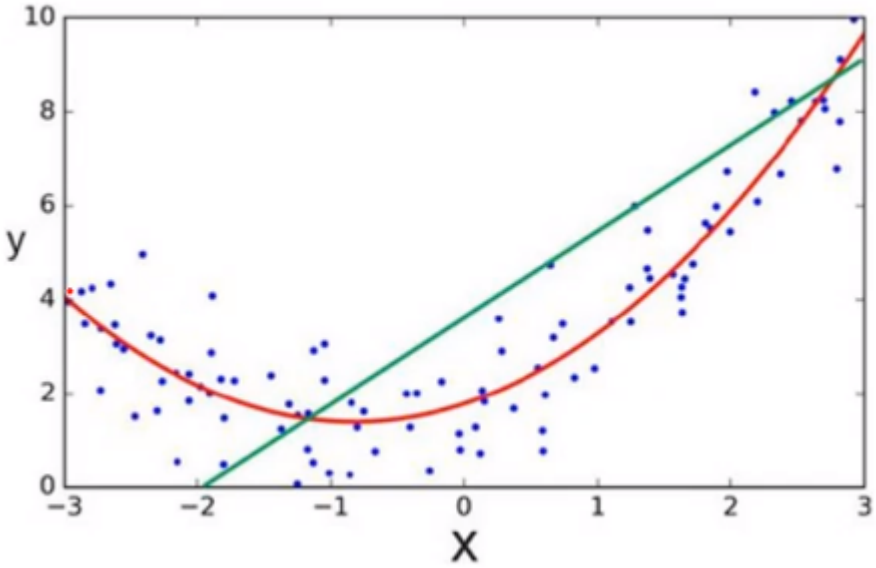
MSE : Mean Squared Error이며 실제 값과 예측값의 차이를 제곱해 평균한 것

RMSE : MSE 값은 오류의 제곱을 구하므로 실제 오류 평균보다 더 커지는 특성이 있으므로 MSE에 루트를 씌운 것이다.

R^2 : 예측값$Variance$/실제값$Variance$

05 다항 회귀와 과적합/과소적합 이해

다항(Polynomial) 회귀 : 회귀가 독립변수의 단항식이 아닌 2차, 3차 방정식과 같은 다항식으로 표현 되는 것.



주의 : 다항회귀는 선형 회귀라는 점

→ 회귀에서 선형 회귀/비선형 회귀를 나누는 기준은 **회귀 계수가 선형/비선형인지**에 따른 것

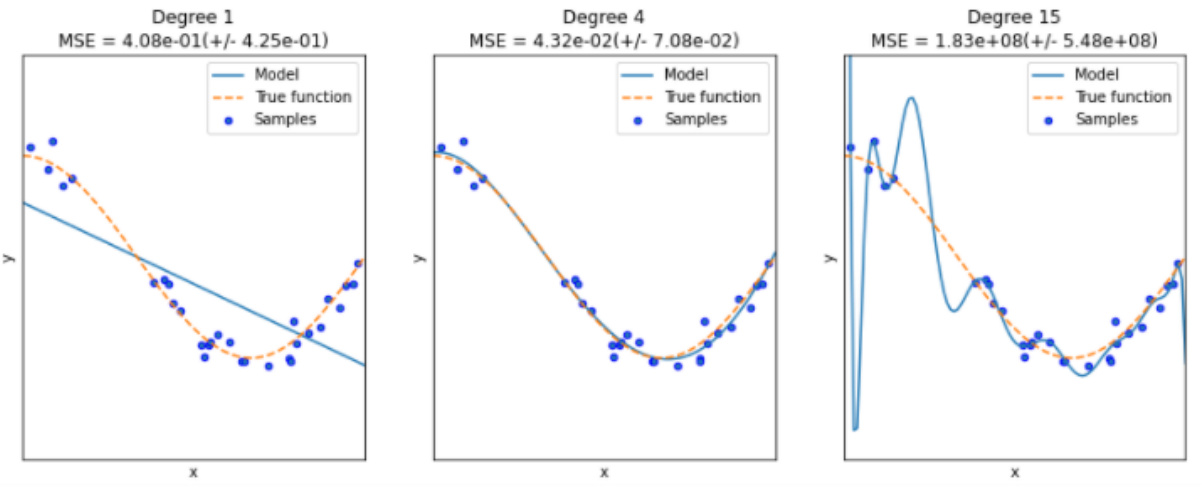
`PolynomialFeatures(degree = n)` **클래스**

`PolynomialFeatures(degree = n)` **: 피처를 n차 다항식으로 변환하는 클래스**

다항 회귀를 이용한 과소적합 및 과적합 이해

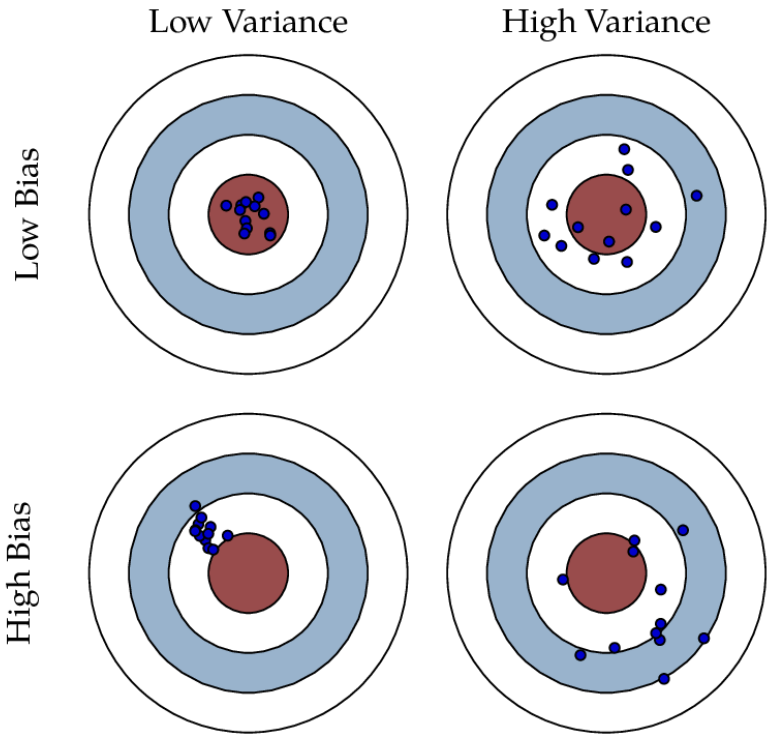
다항 회귀의 차수가 높아질수록 매우 복잡한 피처 간의 관계까지 모델링이 가능하다.

그러나, 다항 회귀의 **차수를 높일수록 학습 데이터에만 너무 맞춘 학습이 이뤄져서** 정작 테스트 데이터 환경에서는 오히려 **예측 정확도가 떨어진다**.

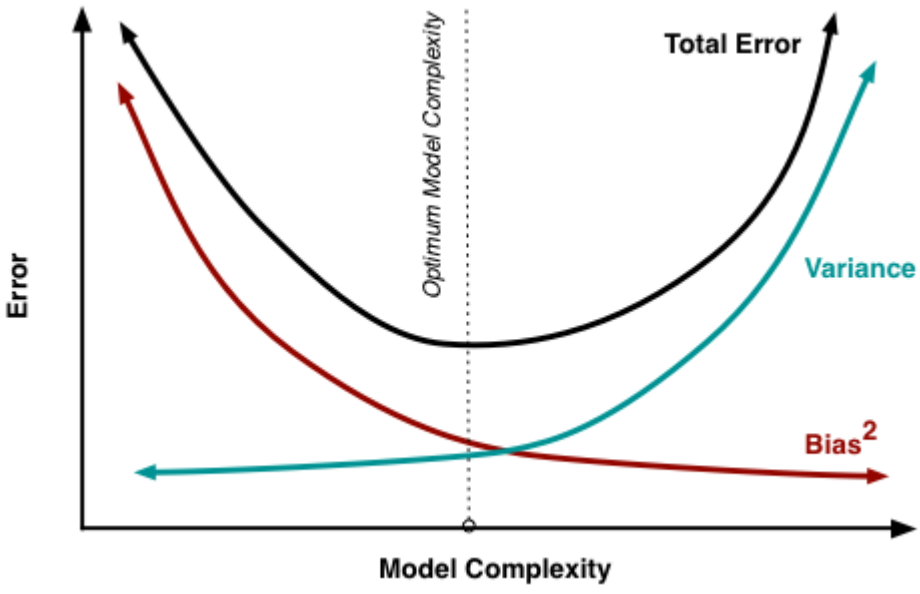


- Degree 1 예측 곡선은 단순한 직선으로서 단순 선형 회귀와 같다. 실제 데이터 세트인 코사인 데이터 세트를 직선으로 예측하기에는 너무 단순하다. **과소 적합 모델**이 되었다.
- Degree 4 예측 곡선은 실제 데이터 세트와 유사한 모습이다. 학습 데이터 세트를 비교적 잘 반영해 테스트 데이터를 잘 예측한 곡선을 가진 모델이 되었다.
- Degree 15 예측 곡선은 MSE 값이 1.83e가 될 정도로 어처구니 없는 값이 발생했다. 예측 곡선을 보면 데이터 세트의 변동잡음 값을 지나치게 반영한 결과, 예측곡선이 학습 데이터만 정확히 예측하고, 테스트 값의 실제 곡선과는 **완전히 다른 형태의 예측 곡선**이 만들어졌다.

편향-분산 트레이드오프(Bias - Variance Trade Off)



1. **저편향/저분산 : 예측 결과가 실제 결과에 매우 근접하며 예측 변동이 크지 않고 특정 부분에 집중돼 있는 아주 뛰어난 성능을 보여준다.**
2. **저편향/고분산 :** 예측 결과가 실제 결과에 비교적 근접하지만, **예측 결과가 실제 결과를 중심으로 꽤 넓은 부분에 분포되어있음을 알 수 있다.**
3. **고편향/저분산 : 정확한 결과에서 벗어나면서도 예측이 특정 부분에 집중되어 있다.**
4. **고편향/고분산 : 정확한 예측 결과를 벗어나고, 넓은 부분에 분포되어 있다.**



높은 편향/낮은 분산에서 **과소적합**되기 쉬우며, **낮은 편향/높은 분산**에서 **과적합**되기 쉽다.

06 규제 선형 모델 - 릿지, 라쏘, 엘라스틱넷

규제 선형 모델의 개요

최적의 모델을 위한 Cost 함수 구성요소 = 학습데이터 잔차 오류 최소화 + 회귀계수 크기 제어

비용 함수 목표 = $Min(RSS(W) + alpha * W_2^2)$

$alpha$ 는 학습 데이터 적합 정도와 회귀 계수 값의 크기 제어를 수행하는 튜닝 파라미터이다.

- alpha = 0 인 경우 W가 커도 전체 값이 0이 되어 비용함수는 $Min(RSS(W))$
- alpha = 무한대인 경우 전체 값도 무한대가 되므로 W는 0에 가깝게 최소화해야한다.

규제(Regularization) : L1 방식, L2 방식으로 구분된다.

- **라소 회귀는 L1 규제를 적용한 회귀이다.**

  → L1 규제는 예측 영향력이 작은 피처의 회귀 계수를 0으로 만들어 회귀 예측 시 피처가 선책되지 않게 하는 것

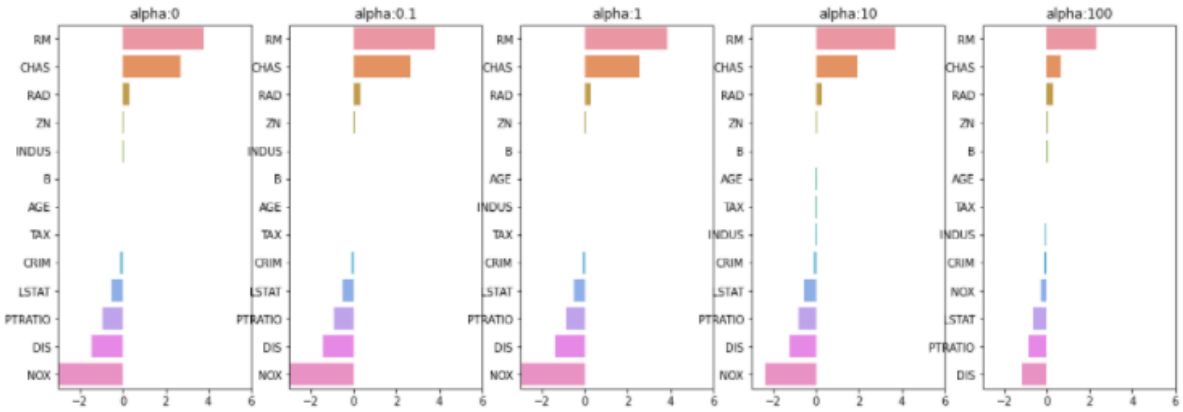
- **릿지 회귀는 L2 규제를 적용한 회귀이다.**

  → L2 규제는 상대적으로 큰 회귀 계수 값의 예측 영향도를 감소시키기 위해서 회귀 계수값을 더 작게 만드는 규제 모델.

- **엘라스틱넷 : L1, L2 규제를 함께 결합한 모델이다. 주로 피처가 많은 데이터 세트에서 적용되며, L1 규제로 피처의 개수를 줄임과 동시에 L2 규제로 계수 값의 크기를 조절한다.**

릿지 회귀

```
alpha 0일 때 5 folds의 평균 RMSE : 5.829
alpha 0.1일 때 5 folds의 평균 RMSE : 5.788
alpha 1일 때 5 folds의 평균 RMSE : 5.653
alpha 10일 때 5 folds의 평균 RMSE : 5.518
alpha 100일 때 5 folds의 평균 RMSE : 5.330
```



alpha 값을 계속 증가시킬수록 **회귀 계수 값은 지속적으로 작아짐**을 **알 수 있다.**

	alpha:0	alpha:0.1	alpha:1	alpha:10	alpha:100
RM	3.809865	3.818233	3.854000	3.702272	2.334536
CHAS	2.686734	2.670019	2.552393	1.952021	0.638335
RAD	0.306049	0.303515	0.290142	0.279596	0.315358
ZN	0.046420	0.046572	0.047443	0.049579	0.054496
INDUS	0.020559	0.015999	-0.008805	-0.042962	-0.052826
B	0.009312	0.009368	0.009673	0.010037	0.009393
AGE	0.000692	-0.000269	-0.005415	-0.010707	0.001212
TAX	-0.012335	-0.012421	-0.012912	-0.013993	-0.015856
CRIM	-0.108011	-0.107474	-0.104595	-0.101435	-0.102202
LSTAT	-0.524758	-0.525966	-0.533343	-0.559366	-0.660764
PTRATIO	-0.952747	-0.940759	-0.876074	-0.797945	-0.829218

그러나, 회귀 계수를 0으로 만들지는 않는다.

라쏘 회귀

```
###### Lasso ######
alpha 0.07 일 때 5 폴드 세트의 평균 RMSE: 5.612
alpha 0.1 일 때 5 폴드 세트의 평균 RMSE: 5.615
alpha 0.5 일 때 5 폴드 세트의 평균 RMSE: 5.669
alpha 1 일 때 5 폴드 세트의 평균 RMSE: 5.776
alpha 3 일 때 5 폴드 세트의 평균 RMSE: 6.189
```

alpha가 0.07일 때 가장 좋은 평균 RMSE를 보여준다.

	alpha:0.07	alpha:0.1	alpha:0.5	alpha:1	alpha:3
RM	3.789725	3.703202	2.498212	0.949811	0.000000
CHAS	1.434343	0.955190	0.000000	0.000000	0.000000
RAD	0.270936	0.274707	0.277451	0.264206	0.061864
ZN	0.049059	0.049211	0.049544	0.049165	0.037231
B	0.010248	0.010249	0.009469	0.008247	0.006510
NOX	-0.000000	-0.000000	-0.000000	-0.000000	0.000000
AGE	-0.011706	-0.010037	0.003604	0.020910	0.042495
TAX	-0.014290	-0.014570	-0.015442	-0.015212	-0.008602
INDUS	-0.042120	-0.036619	-0.005253	-0.000000	-0.000000
CRIM	-0.098193	-0.097894	-0.083289	-0.063437	-0.000000
LSTAT	-0.560431	-0.568769	-0.656290	-0.761115	-0.807679
PTRATIO	-0.765107	-0.770654	-0.758752	-0.722966	-0.265072
DIS	-1.176583	-1.160538	-0.936605	-0.668790	-0.000000

alpha의 크기가 증가함에 따라 **일부 피처의 회귀 계수는 아예 0으로 바뀌고 있다.**

엘라스틱넷 회귀

```
###### ElasticNet ######
alpha 0.07 일 때 5 폴드 세트의 평균 RMSE: 5.542
alpha 0.1 일 때 5 폴드 세트의 평균 RMSE: 5.526
alpha 0.5 일 때 5 폴드 세트의 평균 RMSE: 5.467
alpha 1 일 때 5 폴드 세트의 평균 RMSE: 5.597
alpha 3 일 때 5 폴드 세트의 평균 RMSE: 6.068
```

	alpha:0.07	alpha:0.1	alpha:0.5	alpha:1	alpha:3
RM	3.574162	3.414154	1.918419	0.938789	0.000000
CHAS	1.330724	0.979706	0.000000	0.000000	0.000000
RAD	0.278880	0.283443	0.300761	0.289299	0.146846
ZN	0.050107	0.050617	0.052878	0.052136	0.038268
B	0.010122	0.010067	0.009114	0.008320	0.007020
AGE	-0.010116	-0.008276	0.007760	0.020348	0.043446
TAX	-0.014522	-0.014814	-0.016046	-0.016218	-0.011417
INDUS	-0.044855	-0.042719	-0.023252	-0.000000	-0.000000
CRIM	-0.099468	-0.099213	-0.089070	-0.073577	-0.019058
NOX	-0.175072	-0.000000	-0.000000	-0.000000	-0.000000
LSTAT	-0.574822	-0.587702	-0.693861	-0.760457	-0.800368
PTRATIO	-0.779498	-0.784725	-0.790969	-0.738672	-0.423065
DIS	-1.189438	-1.173647	-0.975902	-0.725174	-0.031208

alpha 0.5일 때 RMSE가 5.467로 가장 좋은 예측 성능을 보이고 있다. alpha 값에 따른 피처들의 회귀 계수들 값이 라쏘보다는 상대적으로 0이 되는 값이 적음을 알 수 있다.

→ 선형 회귀의 경우 최적의 하이퍼 파라미터를 찾아내는 것 못지않게 먼저 데이터 분포도의 정규화와 인코딩 방법이 매우 중요합니다.

선형 회귀 모델을 위한 데이터 변환

1. `StandardSclaer` **클래스를 이용해 평균이 0, 분산이 1인 표준 정규 분포를 가진 데이터 세트로 변환하거나** `MinMaxScaler` **클래스를 이용해 최솟값이 0 이고 최댓값이 1인 값으로 정규화를 수행한다.**
2. **스케일링/정규화를 수행한 데이터 세트에 다시 다항 특성을 적용하여 변환하는 방법이다.**
3. **원래 값에 log 함수를 적용하면 보다 정규 분포에 가까운 형태로 값이 분포된다. 이러한 변환을 로그 변환이라고 부른다. 1,2 방법보다 로그 변환이 훨씬 많이 사용되는 변환 방법이다.**

```
 변환 유형 : None, Polynomial Degree : None
alpha 0.1 일 때 5 폴드 세트의 평균 RMSE: 5.788
alpha 1 일 때 5 폴드 세트의 평균 RMSE: 5.653
alpha 10 일 때 5 폴드 세트의 평균 RMSE: 5.518
alpha 100 일 때 5 폴드 세트의 평균 RMSE: 5.330

 변환 유형 : Stanard, Polynomial Degree : None
alpha 0.1 일 때 5 폴드 세트의 평균 RMSE: 5.788
alpha 1 일 때 5 폴드 세트의 평균 RMSE: 5.653
alpha 10 일 때 5 폴드 세트의 평균 RMSE: 5.518
alpha 100 일 때 5 폴드 세트의 평균 RMSE: 5.330

 변환 유형 : Stanard, Polynomial Degree : 2
alpha 0.1 일 때 5 폴드 세트의 평균 RMSE: 9.141
alpha 1 일 때 5 폴드 세트의 평균 RMSE: 8.938
alpha 10 일 때 5 폴드 세트의 평균 RMSE: 10.556
alpha 100 일 때 5 폴드 세트의 평균 RMSE: 10.566

 변환 유형 : MinMax, Polynomial Degree : None
alpha 0.1 일 때 5 폴드 세트의 평균 RMSE: 5.764
alpha 1 일 때 5 폴드 세트의 평균 RMSE: 5.465
alpha 10 일 때 5 폴드 세트의 평균 RMSE: 5.754
alpha 100 일 때 5 폴드 세트의 평균 RMSE: 7.635

 변환 유형 : MinMax, Polynomial Degree : 2
alpha 0.1 일 때 5 폴드 세트의 평균 RMSE: 5.298
alpha 1 일 때 5 폴드 세트의 평균 RMSE: 4.323
alpha 10 일 때 5 폴드 세트의 평균 RMSE: 5.185
alpha 100 일 때 5 폴드 세트의 평균 RMSE: 6.538

 변환 유형 : Log, Polynomial Degree : None
alpha 0.1 일 때 5 폴드 세트의 평균 RMSE: 4.770
alpha 1 일 때 5 폴드 세트의 평균 RMSE: 4.676
alpha 10 일 때 5 폴드 세트의 평균 RMSE: 4.836
alpha 100 일 때 5 폴드 세트의 평균 RMSE: 6.241
```

결론적으로, 선형 회귀를 적용하려는 데이터 세트에 데이터 값의 **분포가 심하게 왜곡**되어 있을 경우, **로그 변환을 적용하는 것이 좋은 결과를 기대할 수 있다.**

07 로지스틱 회귀

로지스틱 회귀는 선형 회귀 방식을 분류에 적용한 알고리즘입니다. 그러나, 학습을 통해 선형 함수의 회귀 최적선을 찾는 것이 아니라 **시그모이드(Sigmoid) 함수 최적선을 찾고, 이 시그모이드 함수의 반환 값을 확률로 간주에 확률에 따라 분류를 결정한다는 것**이다.

표준 스케일링을 적용한 후에, train_test_split()을 이용해 데이터를 분리하고 정확도 및 ROC_AUC 값을 구하자.

```
accuracy: 0.982
roc_auc: 0.979
```

`LogisticRegression` **클래스의 주요 하이퍼 파라미터로** `penalty` **와** `C` **가 있다.** `penalty` **는 규제의 유형을 설정하며** `l2` **설정 시 I2 규제를,** `l1` **으로 설정 시 I1 규제를 뜻한다.** $C = 1/alpha$ **이며, C 값이 작을수록, 규제 강도가 크다.**

```
최적 하이퍼 파라미터 : {'C': 0.1, 'penalty': 'l2'}, 최적 평균 정확도 : 0.979
```

08 회귀 트리

트리 기반의 회귀는 회귀 트리를 이용하는 것이다. 즉, 회귀를 위한 트리를 생성하고 이를 기반으로 회귀 예측을 하는 것이다.

회귀 트리는 리프 노드에서 예측 결정 값을 만드는 과정에 차이가 있는데, 분류 트리가 특정 클래스 레이블을 결정하는 것과는 달리, **리프 노드에 속한 데이터 값의 평균값을 구해 회귀 예측값을 계산한다.**

CART(Classification and Regression Tree)

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

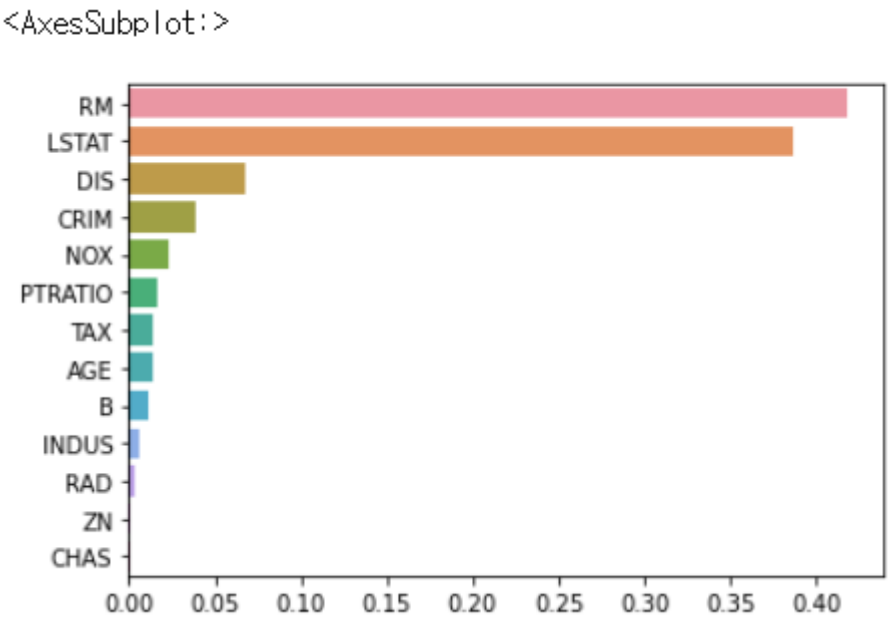
랜덤포레스트의 회귀 트리를 이용해 구한 결과

```
5 교차 검증의 개별 Negative MSE Scores: [ -7.88 -13.14 -20.57 -46.23 -18.88]
5 교차 검증의 개별 RMSE scores: [2.81 3.63 4.54 6.8  4.34]
5 교차 검증의 평균 RMSE:4.423
```

`get_model_cv_prediction()` **(입력 모델과 데이터 세트를 입력 받아 교차 검증으로 평균 RMSE를 계산해주는 함수)를 생성해서 결정트리, GBM, XGBoost, LightGBM의 Regressor를 모두 이용해보자.**

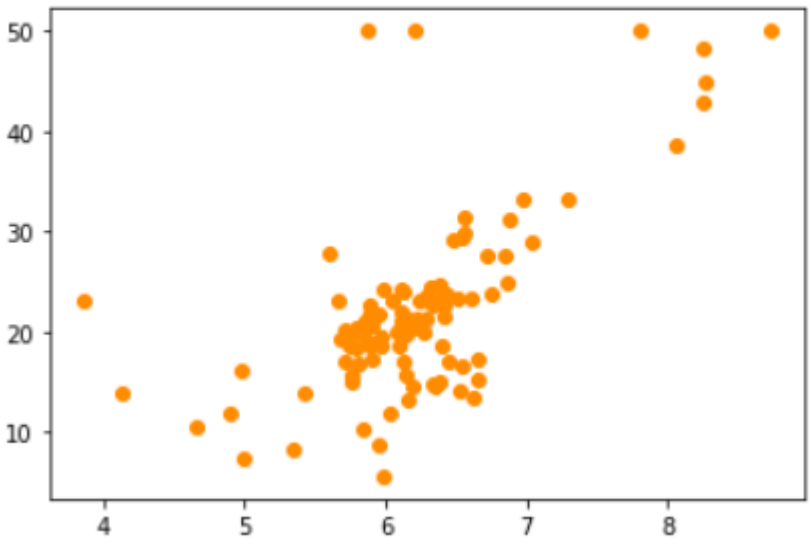
```
####  DecisionTreeRegressor  #####
 5 교차 검증의 평균 RMSE : 5.978
####  RandomForestRegressor  #####
 5 교차 검증의 평균 RMSE : 4.423
####  GradientBoostingRegressor  #####
 5 교차 검증의 평균 RMSE : 4.269
####  XGBRegressor  #####
 5 교차 검증의 평균 RMSE : 4.251
####  LGBMRegressor  #####
 5 교차 검증의 평균 RMSE : 4.646
```

`feature_importance()` **를 이용해 나타낸 피처별 중요도 값**



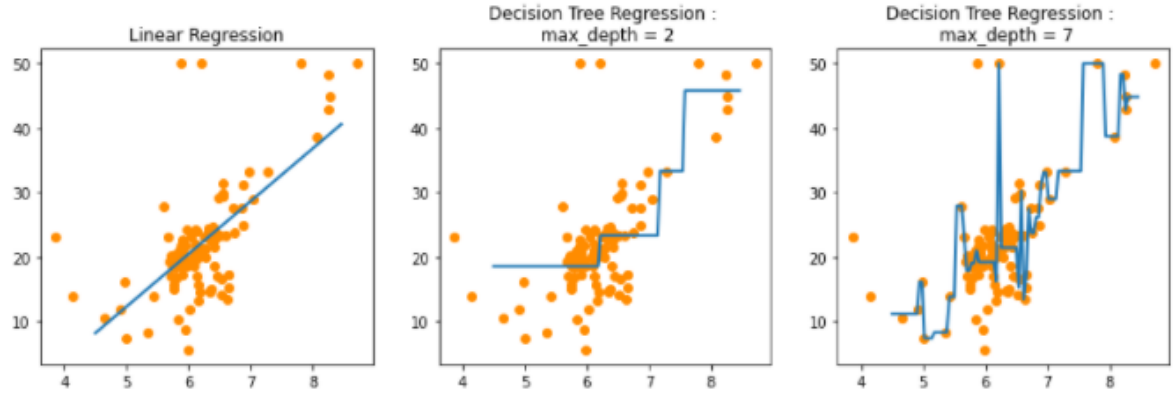
보스턴 데이터 세트에서 RM과 PRICE 칼럼만 추출한 그래프

```
<matplotlib.collections.PathCollection at 0x23a69acd760>
```



`LineRegression`, `DecisionTreeRegressor`**를 max_depth :2, 7 로 학습한 결과**

```
[<matplotlib.lines.Line2D at 0x23a69dba550>]
```



선형 회귀는 직선으로 예측 회귀선을 표현 하는데에 반해, **회귀 트리는 분할되는 지점마다 브랜치를 만들어 계단 형태로 회귀선을 만드는 것**을 확인할 수 있다. max_depth = 7인 경우, **학습데이터의 outlier도 학습**하면서 회귀선을 만드는 것을 확인할 수 있다. **이는 과적합이 되기 쉬운 모델이 되었음을 알 수 있다.**