

7th Meet

□ 오늘 공부할 것은

04

분류

11. 스택킹 앙상블	295
기본 스택킹 모델	297
CV 세트 기반의 스택킹	300
12. 정리	306

05

회귀

01. 회귀 소개	308
02. 단순 선형 회귀를 통한 회귀 이해	310
03. 비용 최소화하기 - 경사 하강법(Gradient Descent) 소개	312
04. 사이킷런 LinearRegression을 이용한 보스턴 주택 가격 예측	321
LinearRegression 클래스 - Ordinary Least Squares	321
회귀 평가 지표	322
LinearRegression을 이용해 보스턴 주택 가격 회귀 구현	324
05. 다항 회귀와 과(대)적합/과소적합 이해	329
다항 회귀 이해	329
다항 회귀를 이용한 과소적합 및 과적합 이해	332
편향-분산 트레이드오프(Bias-Variance Trade off)	336

Now, It's your turn !!!

40min ~ 50min

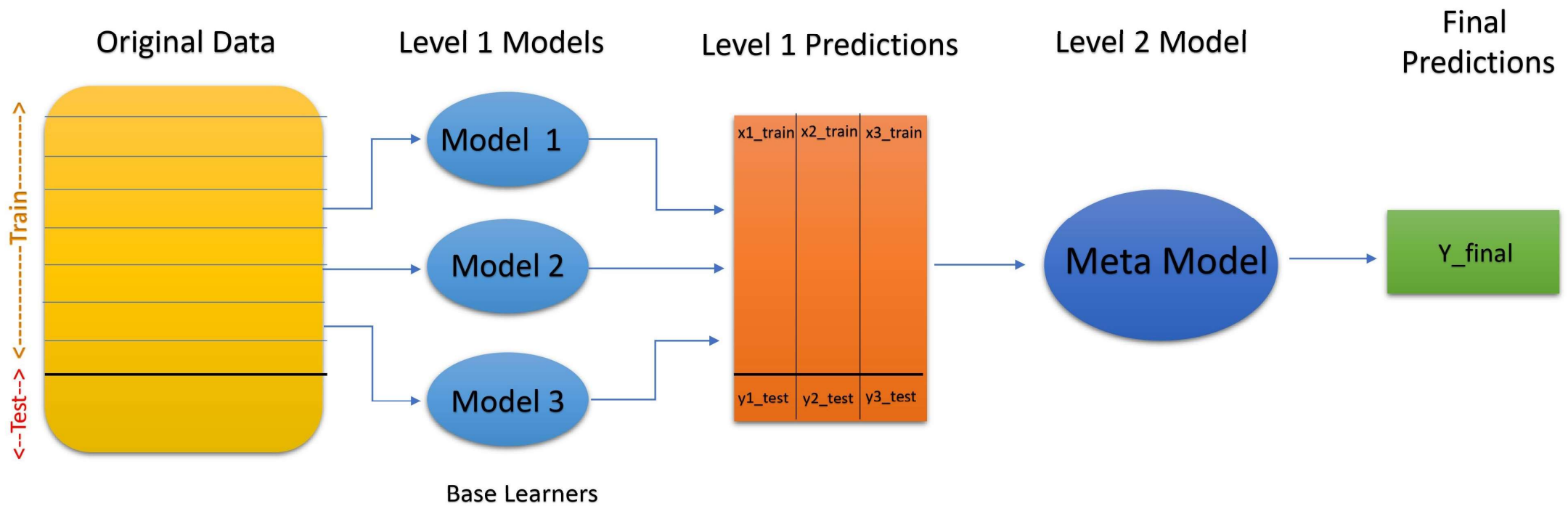
Chapter 04

분류 (Classification)

[04-11]

스태킹 앙상블

(Stacking Ensemble)



<https://iq.opengenus.org/stacking-in-machine-learning/>

https://github.com/whatwant-school/python-ml/blob/main/07-week/07-week_01-Stacking.ipynb

```
knn_train, knn_test = get_stacking_base_datasets(knn_clf, train_X, train_y, test_X, 7)
rf_train, rf_test = get_stacking_base_datasets(rf_clf, train_X, train_y, test_X, 7)
ada_train, ada_test = get_stacking_base_datasets(ada_clf, train_X, train_y, test_X, 7)
dt_train, dt_test = get_stacking_base_datasets(dt_clf, train_X, train_y, test_X, 7)
```

KNeighborsClassifier model 시작

폴드 세트: 0 시작
폴드 세트: 1 시작
폴드 세트: 2 시작
폴드 세트: 3 시작
폴드 세트: 4 시작
폴드 세트: 5 시작
폴드 세트: 6 시작

RandomForestClassifier model 시작

폴드 세트: 0 시작
폴드 세트: 1 시작
폴드 세트: 2 시작
폴드 세트: 3 시작
폴드 세트: 4 시작
폴드 세트: 5 시작
폴드 세트: 6 시작

AdaBoostClassifier model 시작

폴드 세트: 0 시작
폴드 세트: 1 시작
폴드 세트: 2 시작
폴드 세트: 3 시작

Chapter 05

회귀 (Regression)

[05-01]

회귀 소개

□ 회귀

- 여러 개의 독립변수와 한 개의 종속변수 간의 상관관계를 모델링하는 기법

선형
비선형

일반 선형 회귀: RSS (Residual Sum of Squares) 최소화할 수 있도록 회귀 계수 최적화
릿지(Ridge): L2 Regularization (규제) 추가. 회귀 계수 값을 작게 만드는 규제 모델
라쏘(Lasso): L1 Regularization. 예측 영향력이 작은 회귀 계수 0 만들기 → 피쳐 선택
엘라스틱넷(ElasticNet): L2 L1 결합 모델. 피쳐가 많은 Dataset에 주로 적용
로지스틱 회귀(Logistic Regression): 회귀 명칭이지만, 실제로는 분류

회귀 계수 (Regression coefficients)

$$Y = W_1 * X_1 + W_2 * X_2 + W_3 * X_3 + \dots + W_n * X_n$$

종속변수 (label)

이산값 : Classification
연속값 : Regression

독립변수 (features)

단일 회귀
다중 회귀

[05-02]

단순 선형 회귀를 통한 회귀 이해

□ 단순 선형 회귀

$$Y = W_1 * X_1$$

회귀 계수 (Regression coefficients) W_1

종속변수 (label) Y

독립변수 (features) X_1

오류 절대값의 합

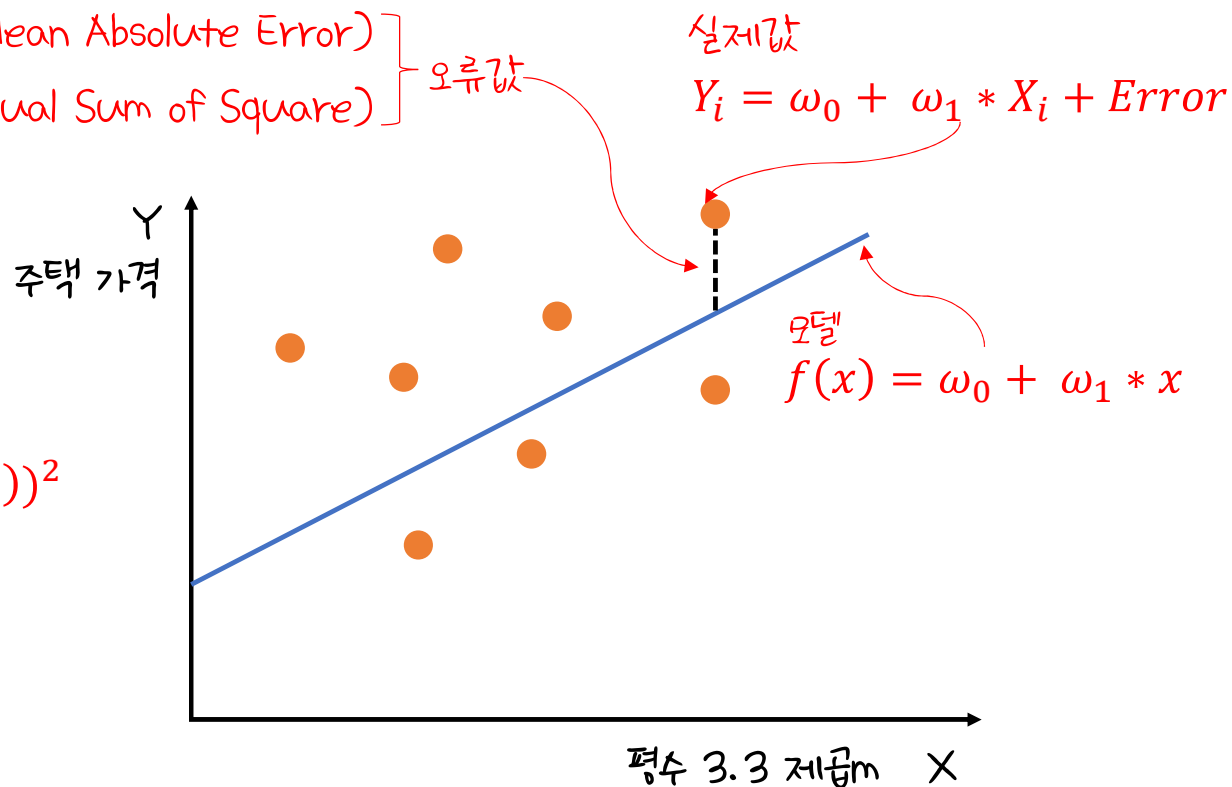
MAE (Mean Absolute Error)

RSS (Residual Sum of Square)

오류 제곱의 합

cost function / loss function

$$RSS(\omega_0, \omega_1) = \frac{1}{N} \sum_{i=1}^N (y_i - (\omega_0 + \omega_1 * x_i))^2$$

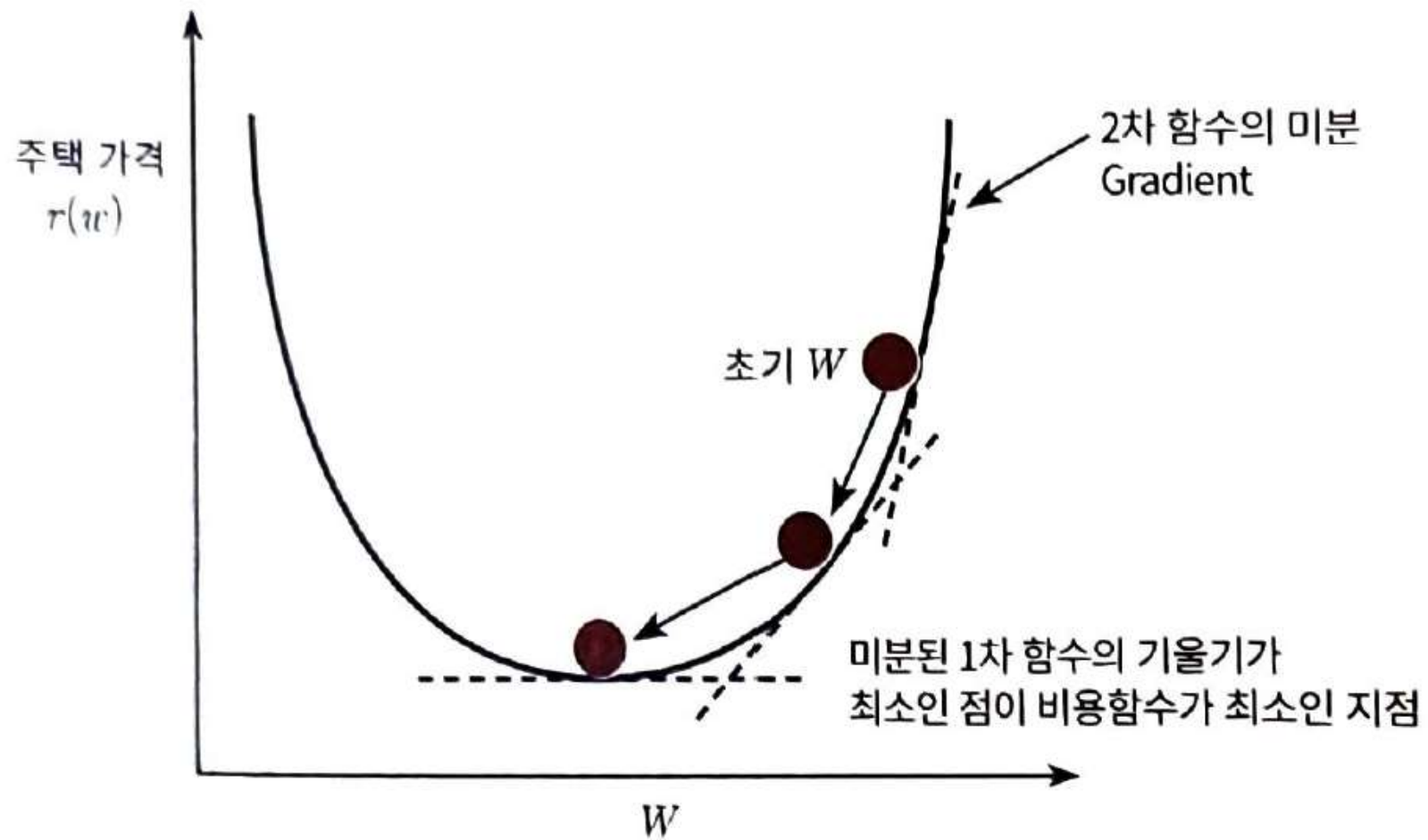


[05-03]

비용 최소화하기

경사 하강법 (Gradient Descent) 소개

□ Gradient Descent



□ Gradient Descent

$$RSS(\omega_0, \omega_1) = \frac{1}{N} \sum_{i=1}^N (y_i - (\omega_0 + \omega_1 * x_i))^2$$

← 2번의 편미분 필요

$$\begin{aligned} \frac{\partial RSS(\omega_0, \omega_1)}{\partial \omega_1} &= \frac{1}{N} \sum_{i=1}^N (y_i - (\omega_0 + \omega_1 * x_i))(-2 x_i) \\ &= -\frac{2}{N} \sum_{i=1}^N x_i (y_i - (\omega_0 + \omega_1 * x_i)) \\ &= -\frac{2}{N} \sum_{i=1}^N x_i (\text{실제값}_i - \text{예측값}_i) \end{aligned}$$

$$\begin{aligned} \frac{\partial RSS(\omega_0, \omega_1)}{\partial \omega_0} &= \frac{1}{N} \sum_{i=1}^N (y_i - (\omega_0 + \omega_1 * x_i))(-2) \\ &= -\frac{2}{N} \sum_{i=1}^N (y_i - (\omega_0 + \omega_1 * x_i)) \\ &= -\frac{2}{N} \sum_{i=1}^N (\text{실제값}_i - \text{예측값}_i) \end{aligned}$$

□ Gradient Descent

$$\begin{aligned}\omega_{i+1} &= \omega_i - \overset{\text{학습률 (learning rate)}}{\eta} * \left(-\frac{2}{N} \sum_{i=1}^N x_i (\text{실제값}_i - \text{예측값}_i)\right) \\ &= \omega_i + \eta \frac{2}{N} \sum_{i=1}^N x_i (\text{실제값}_i - \text{예측값}_i)\end{aligned}$$

Step 1 : w_0, w_1 을 임의의 값으로 설정하고 첫 비용 함수의 값을 계산

Step 2 : w_0, w_1 을 업데이트한 후 다시 비용 함수의 값을 계산

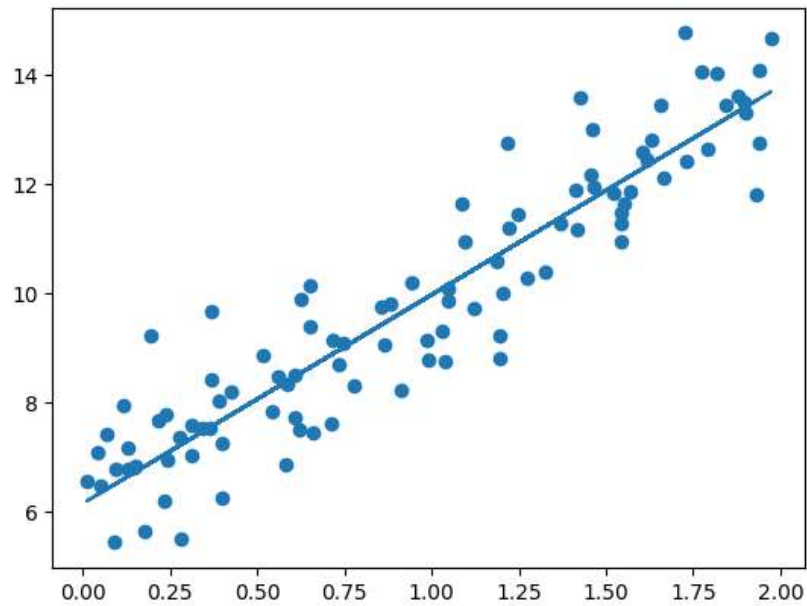
Step 3 : 비용 함수가 감소하는 방향으로 주어진 횟수만큼 Step 2 반복

□ Gradient Descent

https://github.com/whatwant-school/python-ml/blob/main/07-week/07-week_02-GradientDescent.ipynb

```
y_pred = w1[0,0] * X + w0  
  
print('Gradient Descent Total Cost:{0:.4f}'.format(get_cost(y, y_pred)))  
Gradient Descent Total Cost:0.8075  
  
plt.scatter(X, y)  
plt.plot(X, y_pred)
```

[<matplotlib.lines.Line2D at 0x7f8706d2b880>]



□ Stochastic Gradient Descent (확률적 경사 하강법)

https://github.com/whatwant-school/python-ml/blob/main/07-week/07-week_03-StochasticGradientDescent.ipynb

```
def stochastic_gradient_descent_steps(X, y, batch_size=10, iters=1000):
    w0 = np.zeros((1,1))
    w1 = np.zeros((1,1))

    prev_cost = 100000
    iter_index = 0

    for ind in range(iters):
        np.random.seed(ind)
        stochastic_random_index = np.random.permutation(X.shape[0])

        sample_X = X[stochastic_random_index[0:batch_size]]
        sample_y = y[stochastic_random_index[0:batch_size]]

        w1_update, w0_update = get_weight_updates(w1, w0, sample_X, sample_y, learning_rate=0.01)
        w1 = w1 - w1_update
        w0 = w0 - w0_update

    return w1, w0
```

[05-04]

사이킷런 LinearRegression을 이용한
보스턴 주택 가격 예측

□ OLS (Ordinary Least Squares, 최소제곱법/최소자승법)

- 근사적으로 구하려는 해와 실제 해의 오차의 제곱의 합이 최소가 되는 해를 구하는 방법
 - 잔차제곱합(RSS: Residual Sum of Squares)을 최소화 하는 가중치 벡터를 구하는 방법
 - RSS 공식을 미분한 식의 결과가 0이 되도록 하는 벡터 값을 구하는 방법
- But 다중공선성 상황(상관관계가 높은 경우)에서는 오류에 매우 민감해짐

□ 다중공선성 (Multicollinearity)

- 어떤 독립변수가 다른 독립변수들과 완벽한 선형 독립이 아닌 경우
- 독립변수 간 상관관계가 높은 경우

□ 피어슨 상관 계수 (Pearson Correlation Coefficient ,PCC)

- 두 변수 간의 선형 상관 관계를 계량화한 수치

□ 회귀 평가 지표

평가 방법	사이킷런 API	Scoring 함수 적용 값	공식
MAE (Mean Absolute Error)	metrics.mean_absolute_error	'neg_mean_absolute_error'	$MAE = \frac{1}{n} \sum_{i=1}^n Y_i - \hat{Y}_i $
MSE (Mean Squared Error)	metrics.mean_squared_error	'neg_mean_squared_error'	$MSE = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$
RMSE (Root Mean Squared Error)	metrics.mean_squared_error (단, squared=False)	'neg_root_mean_squared_error'	$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2}$

□ Boston Housing (Housing Values in Suburbs of Boston)

<https://www.kaggle.com/c/boston-housing/overview>

The screenshot shows the Kaggle interface for the Boston Housing competition. On the left is a sidebar with navigation links: Home, Competitions (selected), Datasets, Models, Code, Discussions, Learn, More, Your Work, and a VIEWED section containing Boston Housing. The main content area has a search bar and a user profile icon. Below the search bar is a large blue banner with the text 'Late Submission' and a three-dot menu. The title 'Boston Housing' is prominently displayed, followed by the subtitle 'Boston Housing'. Below the title are tabs for Overview, Data, Discussion, Leaderboard, and Rules. The Overview tab is active, showing a timeline from 'Start' (Jun 3, 2016) to 'Close' (Jan 1, 2017). To the right of the timeline are two sections: 'Prizes & Awards' and 'Participation'. The 'Prizes & Awards' section states 'Knowledge' and 'Does not award Points or Medals'. The 'Participation' section lists '83 Competitors', '75 Teams', and '261 Entries'.

Boston Housing
Boston Housing

Overview Data Discussion Leaderboard Rules

Overview

Start
Jun 3, 2016

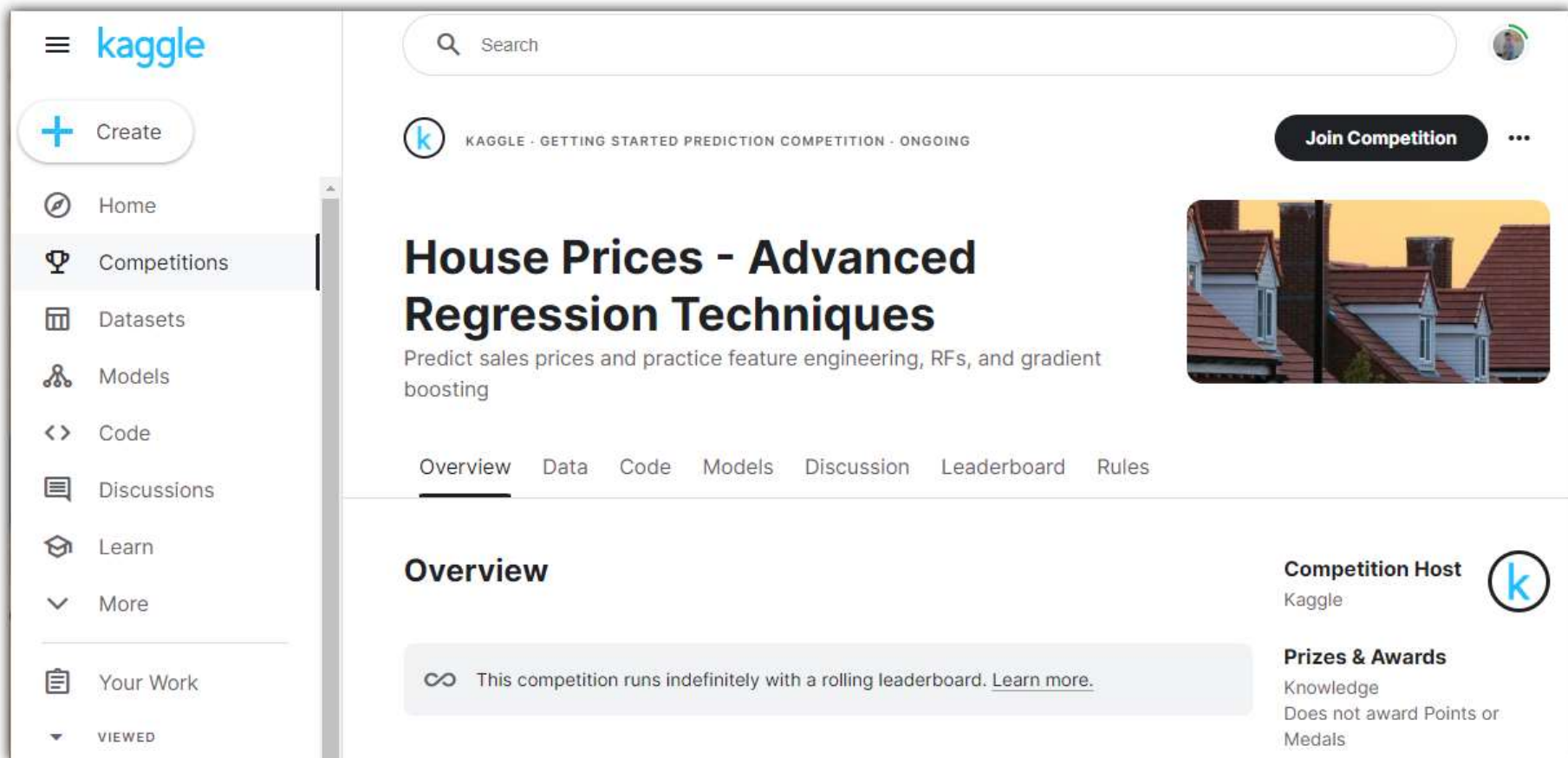
Close
Jan 1, 2017

Prizes & Awards
Knowledge
Does not award Points or Medals

Participation
83 Competitors
75 Teams
261 Entries

□ Boston Housing (Housing Values in Suburbs of Boston)

<https://www.kaggle.com/c/house-prices-advanced-regression-techniques>



The image shows the Kaggle interface for the 'House Prices - Advanced Regression Techniques' competition. On the left is a sidebar with navigation links: Create, Home, Competitions (selected), Datasets, Models, Code, Discussions, Learn, More, Your Work, and VIEWED. The main content area has a search bar and a user profile icon. Below the search bar, it says 'KAGGLE · GETTING STARTED PREDICTION COMPETITION · ONGOING' and features a 'Join Competition' button. The competition title 'House Prices - Advanced Regression Techniques' is prominently displayed, followed by a subtitle 'Predict sales prices and practice feature engineering, RFs, and gradient boosting' and a thumbnail image of houses. A navigation bar below the title includes links for Overview (selected), Data, Code, Models, Discussion, Leaderboard, and Rules. The 'Overview' section contains a note that the competition runs indefinitely with a rolling leaderboard. On the right, it identifies the 'Competition Host' as Kaggle and lists 'Prizes & Awards' as Knowledge, noting that it does not award Points or Medals.

Kaggle

Search

KAGGLE · GETTING STARTED PREDICTION COMPETITION · ONGOING

Join Competition

House Prices - Advanced Regression Techniques

Predict sales prices and practice feature engineering, RFs, and gradient boosting

Overview Data Code Models Discussion Leaderboard Rules

Overview

∞ This competition runs indefinitely with a rolling leaderboard. [Learn more.](#)

Competition Host
Kaggle

Prizes & Awards
Knowledge
Does not award Points or Medals

□ Boston Housing (Housing Values in Suburbs of Boston)

https://github.com/whatwant-school/python-ml/blob/main/07-week/07-week_04-BostonHousing.ipynb

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import numpy as np

target_y = df['PRICE']
data_X = df.drop(['PRICE'],axis=1,inplace=False)

train_X, test_X, train_y, test_y = train_test_split(data_X, target_y, test_size=0.3, random_state=42)

lr = LinearRegression()
lr.fit(train_X, train_y)

predicts = lr.predict(test_X)
mse = mean_squared_error(test_y, predicts)
rmse = np.sqrt(mse)

print('MSE : {0:.3f} , RMSE : {1:.3F}'.format(mse , rmse))
print('Variance score : {0:.3f}'.format(r2_score(test_y, predicts)))

MSE : 21.100 , RMSE : 4.593
Variance score : 0.725

print('절편 값:', lr.intercept_)
print('회귀 계수값:', np.round(lr.coef_, 1))

절편 값: 37.421623169243276
회귀 계수값: [ -0.   -0.1   0.1  -0.   4.8 -15.9  3.7  -0.   -2.   0.4  -0.   -0.7
 0.   -0.6]
```

[05-05]

다항 회귀와

과(대)적합/과소적합

이해

□ 다항 회귀

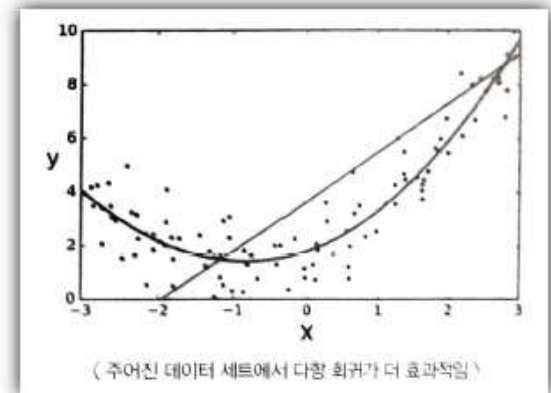
$$y = \omega_0 + \omega_1 * x_1 + \omega_2 * x_2 + \omega_3 * x_3 + \dots + \omega_n * x_n$$



$$y = \omega_0 + \omega_1 * x_1 + \omega_2 * x_2 + \omega_3 * x_2 * x_3 + \omega_4 * x_1^2$$

단항식

다항식 (Polynomial)



https://github.com/whatwant-school/python-ml/blob/main/07-week/07-week_05-polynomial.ipynb

```
from sklearn.pipeline import Pipeline
import numpy as np

def polynomial_func(X):
    y = 1 + 2*X[:,0] + 3*X[:,0]**2 + 4*X[:,1]**3
    return y

model = Pipeline([('poly', PolynomialFeatures(degree=3)),
                  ('linear', LinearRegression())])

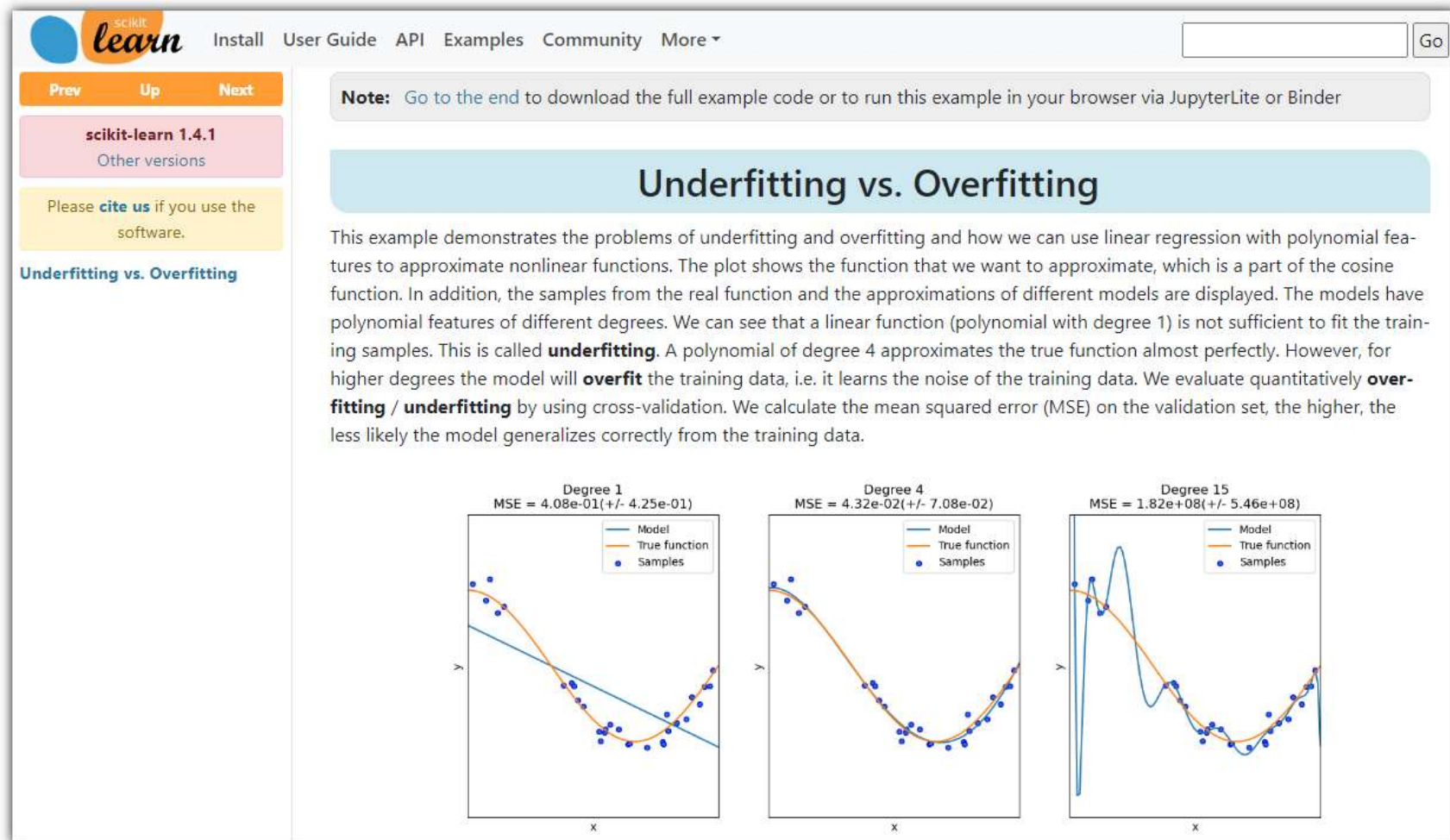
X = np.arange(4).reshape(2,2)
y = polynomial_func(X)

model = model.fit(X, y)
print('Polynomial 회귀 계수\n', np.round(model.named_steps['linear'].coef_, 2))

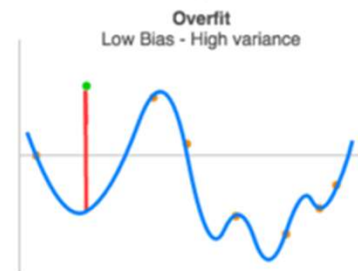
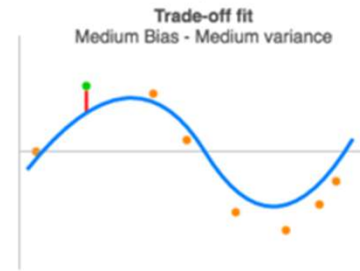
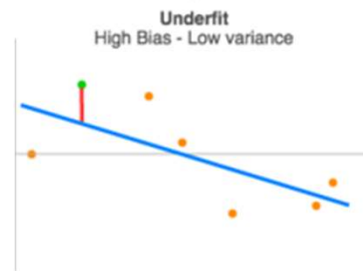
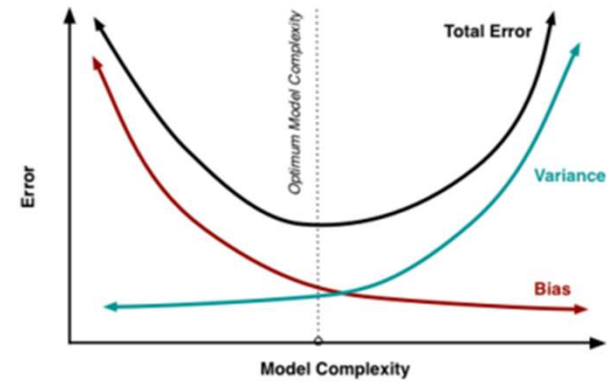
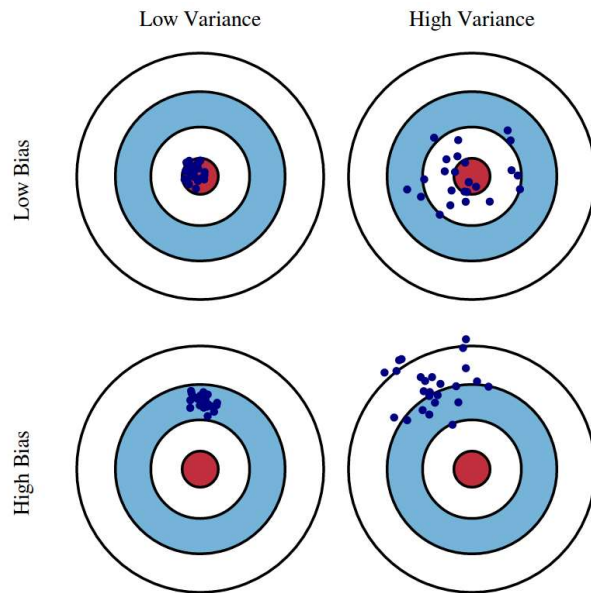
Polynomial 회귀 계수
[0.  0.18 0.18 0.36 0.54 0.72 0.72 1.08 1.62 2.34]
```

□ Underfitting vs. Overfitting

https://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html



□ 편향-분산 트레이드오프 (Bias-Variance Trade off)



<http://scott.fortmann-roe.com/docs/BiasVariance.html>

<https://m.blog.naver.com/ckdgus1433/221594203319>

You've really worked hard today

Next Week ~ ?

05

회귀

11. 스택킹 앙상블	295
기본 스택킹 모델	297
CV 세트 기반의 스택킹	300
12. 정리	306
01. 회귀 소개	308
02. 단순 선형 회귀를 통한 회귀 이해	310
03. 비용 최소화하기 - 경사 하강법(Gradient Descent) 소개	312
04. 사이킷런 LinearRegression을 이용한 보스턴 주택 가격 예측	321
LinearRegression 클래스 - Ordinary Least Squares	321
회귀 평가 지표	322
LinearRegression을 이용해 보스턴 주택 가격 회귀 구현	324
05. 다항 회귀와 과(대)적합/과소적합 이해	329
다항 회귀 이해	329
다항 회귀를 이용한 과소적합 및 과적합 이해	332
편향-분산 트레이드오프(Bias-Variance Trade off)	336
06. 규제 선형 모델 - 릿지, 라쏘, 엘라스틱넷	337
규제 선형 모델의 개요	337
릿지 회귀	339
라쏘 회귀	342
엘라스틱넷 회귀	345
선형 회귀 모델을 위한 데이터 변환	347
07. 로지스틱 회귀	350

08. 회귀 트리	355
09. 회귀 실습 - 자전거 대여 수요 예측	362
데이터 클렌징 및 가공과 데이터 시각화	363
로그 변환, 피쳐 인코딩과 모델 학습/예측/평가	368
10. 회귀 실습 - 캐글 주택 가격: 고급 회귀 기법	375
데이터 사전 처리(Preprocessing)	375
선형 회귀 모델 학습/예측/평가	380
회귀 트리 모델 학습/예측/평가	391
회귀 모델의 예측 결과 혼합을 통한 최종 예측	392
스태킹 앙상블 모델을 통한 회귀 예측	394
11. 정리	397

06

차원 축소

01. 차원 축소(Dimension Reduction) 개요	399
02. PCA(Principal Component Analysis)	401
PCA 개요	401
03. LDA(Linear Discriminant Analysis)	415
LDA 개요	415
04. SVD(Singular Value Decomposition)	418
SVD 개요	418
사이킷런 TruncatedSVD 클래스를 이용한 변환	424
05. NMF(Non-Negative Matrix Factorization)	427
NMF 개요	427
06. 정리	429

Who ~ ?

See you Next Weekend ~ ?