T-Academy

Lab_03) Regression

2019. Apr. 25.
SK플래닛 T아카데미
캐글 코리아
강천성

Linear Regression

선형 회귀는 종속 변수와 한개 이상의 독립 변수와의 선형 상관 관계를 모델링하는 회귀 분석 기법입니다.
용어를 종속 변수, 독립 변수로 표현하면 이해하기 어려우니 다음 수식에서의 y, x 로 표현하겠습니다.

$$y = wx + b$$
$$y = w_0x_0 + w_1x_1 + w_2x_2 + \ldots . w_nx_n + b$$
$$w : \text{계수(가중치)}$$
$$b : \text{절편(편향)}$$

간단하게 생각해보면 선형 회귀는 데이터가 분포되어 있는 공간에서 데이터를 가장 잘 표현하는 선을 하나 긋는다고 생각할 수 있습니다.
선형 회귀의 비용 함수는 다음과 같이 표현될 수 있습니다.

$$Cost_{lr} = \sum_i (y_i - \hat{y}_i)^2$$
$$\hat{y}_i = b + wx_i$$

결국 실제 참값 y_i와 회귀 모델이 출력한 $\hat{y}$ 사이의 잔차의 제곱의 합을 최소화하는 w(계수)를 구하는 것이 목적입니다. -> Least Square, 최소 제곱법
선형 회귀는 출력되는 y가 1개 또는 2개 이상인지의 유무에 따라 단변량, 다변량이라는 말이 붙는데, 이번 수업에서는 출력값인 y가 1개(단변량)라고 가정하겠습니다.
또한, 입력으로 들어가는 x가 1개 또는 2개 이상인지의 유무에 따라 단순(Simple), 다중(Multiple)이라는 말이 붙는데, 이번 실습에서는 단순, 다중 선형 회귀 분석에 대해 모두 알아보겠습니다.

선형 회귀분석의 4가지 기본 가정

선형 회귀에는 4가지 가정이 필요합니다. 우리 수업에서는 이론적인 내용을 다루지 않으므로, 추후에 살펴보시면 좋겠습니다.
맨 아래 참조 목록에 4가지 가정에 대해 잘 설명해준 페이지의 링크를 달아두었습니다.

1. 선형성
2. 독립성
3. 등분산성
4. 정규성

• Simple Linear Regression

1. 모델 불러오기 및 정의

```
from sklearn.linear_model import LinearRegression
sim_lr = LinearRegression()
```

2. fit

```
sim_lr.fit(x_train['RM'].values.reshape((-1, 1)), y_train)
```

3. predict

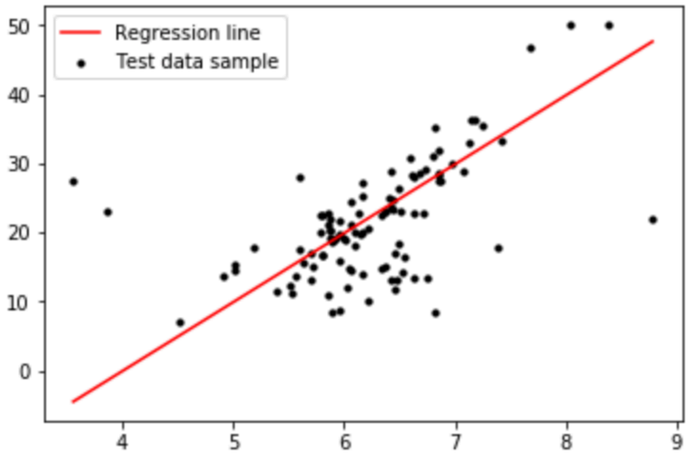
```
y_pred = sim_lr.predict(x_test['RM'].values.reshape((-1, 1)))
```

4. 결과 확인

```
from sklearn.metrics import r2_score
```

```
print('단순 선형 회귀, R2 : {:.4f}'.format(r2_score(y_test, y_pred)))
```

```
단순 선형 회귀, R2 : 0.1795
```



회귀 모델의 계수 w, 절편 b 살펴보기

어떤 변수에 얼마 만큼의 가중치가 할당되고, 절편 값은 얼마나 할당되는지 살펴볼 수 있습니다.

```
print('단순 선형 회귀, 계수(w) : {:.4f}, 절편(b) : {:.4f}'.format(sim_lr.coef_[0], sim_lr.intercept_))
```

```
단순 선형 회귀, 계수(w) : 9.9900, 절편(b) : -40.0941
```

• Multiple Linear Regression

1. 모델 불러오기 및 정의

```
from sklearn.linear_model import LinearRegression
mul_lr = LinearRegression()
```

2. fit

```
mul_lr.fit(x_train.values, y_train)
```

3. predict

```
y_pred = mul_lr.predict(x_test.values)
```

4. 결과 확인

```
print('다중 선형 회귀, R2 : {:.4f}'.format(r2_score(y_test, y_pred)))
```

```
다중 선형 회귀, R2 : 0.6174
```

회귀 모델의 계수 w, 절편 b 살펴보기

어떤 변수에 얼마 만큼의 가중치가 할당되고, 절편 값은 얼마나 할당되는지 살펴볼 수 있습니다.

```
print('다중 선형 회귀, 계수(w) : {}, 절편(b) : {:.4f}'.format(mul_lr.coef_, mul_lr.intercept_))
```

```
다중 선형 회귀, 계수(w) : [-1.39521123e-01  4.17817156e-02 -4.57312740e-03  3.78506106e+00
 -1.46255552e+01  4.52548061e+00  1.49683102e-04 -1.38217694e+00
  2.78132923e-01 -1.03183306e-02 -8.42539713e-01  1.05460752e-02
 -5.19900681e-01], 절편(b) : 27.2753
```

Machine Learning Algorithm Based Regression

이번에는 머신러닝 알고리즘을 기반으로한 회귀 모델에 대해 알아보겠습니다.
Sklearn이 지원하는 머신러닝 기반 회귀 모델로는 결정 트리, 랜덤 포레스트, 서포트 벡터 머신, MLP, AdaBoost, Gradient Boosting 등이 있습니다.
그 중 결정 트리, 서포트 벡터 머신, MLP 회귀 모델을 살펴보겠습니다.

• Decision Tree Regression

트리 모델은 데이터의 불순도(impurity, Entropy)를 최소화 하는 방향으로 트리를 분기하여 모델을 생성합니다. 자세한 내용은 분류 수업에서 설명 드리겠습니다.
결정 트리 회귀 모델은 Sklearn의 tree 패키지에 있습니다.

1. 모델 불러오기 및 정의

```
from sklearn.tree import DecisionTreeRegressor
dt_regr = DecisionTreeRegressor(max_depth=4, random_state=2019)
```

2. fit

```
dt_regr.fit(x_train['RM'].values.reshape((-1, 1)), y_train)
```

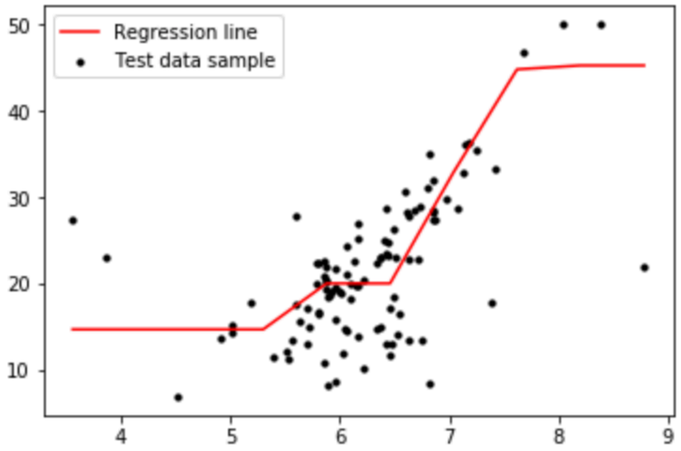
3. predict

```
y_pred = dt_regr.predict(x_test['RM'].values.reshape((-1, 1)))
```

4. 결과 확인

```
print('단순 결정 트리 회귀, R2 : {:.4f}'.format(r2_score(y_test, y_pred)))
```

```
단순 결정 트리 회귀, R2 : 0.4643
```



13개의 변수를 모두 사용해 결정 트리 회귀 모델을 사용해 보세요. (5분)

```
dt_regr = DecisionTreeRegressor(max_depth=4, random_state=2019)
dt_regr.fit(x_train, y_train)
y_pred = dt_regr.predict(x_test)
print('다중 결정 트리 회귀, R2 : {:.4f}'.format(r2_score(y_test, y_pred)))
```

```
다중 결정 트리 회귀, R2 : 0.6494
```

• Support Vector Machine Regression

서포트 벡터 머신의 기본 개념은 결정 경계와 가장 가까운 데이터 샘플의 거리(Margin)을 최대화 하는 방식으로 모델을 조정합니다.
자세한 내용은 분류 파트에서 설명드리겠습니다.
서포트 벡터 머신 회귀 모델은 Sklearn의 svm 패키지에 있습니다.

1. 모델 불러오기 및 정의

```
from sklearn.svm import SVR
svm_regr = SVR(C=1.0, kernel='rbf')
```

2. fit

```
svm_regr.fit(x_train['RM'].values.reshape((-1, 1)), y_train)
```

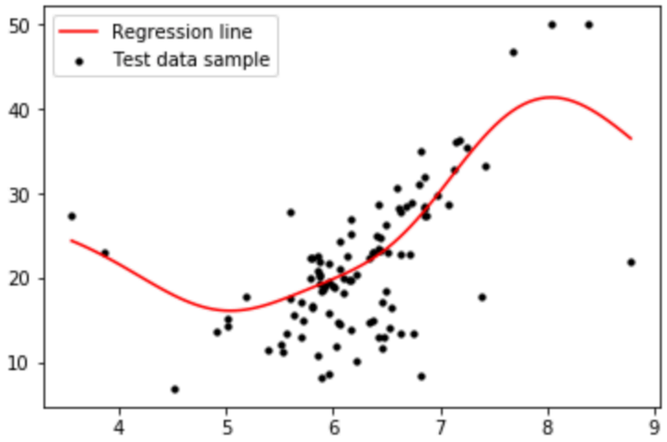
3. predict

```
y_pred = svm_regr.predict(x_test['RM'].values.reshape((-1, 1)))
```

4. 결과 확인

```
print('단순 서포트 벡터 머신 회귀, R2 : {:.4f}'.format(r2_score(y_test, y_pred)))
```

```
단순 서포트 벡터 머신 회귀, R2 : 0.4906
```



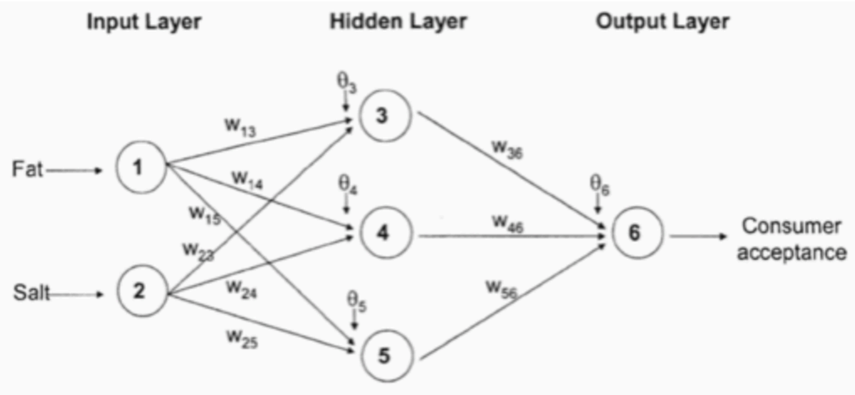
13개의 변수를 모두 사용해 서포트 벡터 머신 회귀 모델을 사용해 보세요. (5분)

```
svm_regr = SVR(C=1.0, kernel='rbf')
svm_regr.fit(x_train, y_train)
y_pred = svm_regr.predict(x_test)
print('다중 서포트 벡터 머신 회귀, R2 : {:.4f}'.format(r2_score(y_test, y_pred)))
```

```
다중 서포트 벡터 머신 회귀, R2 : 0.0359
```

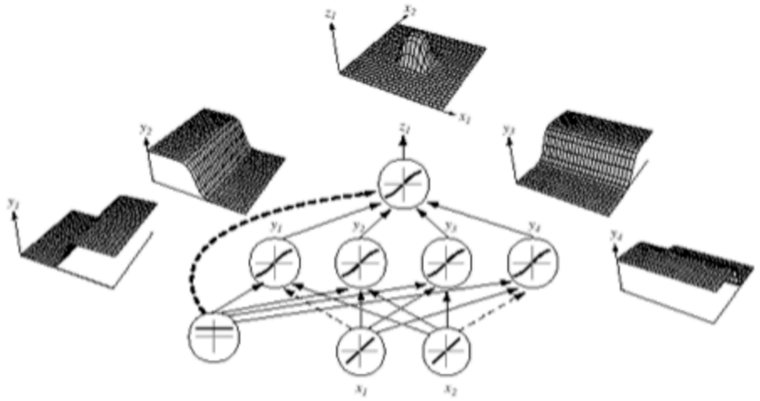
3. Multi Layer Perceptron Regressor

딥러닝의 기본 모델인 뉴럴 네트워크를 기반으로 한 회귀 모델입니다. 기본적으로 MLP라 하면, 입력층-은닉층-출력층 3개로 이루어진 뉴럴 네트워크를 의미합니다.



어떻게 뉴럴 네트워크가 비선형 문제를 해결할 수 있을까?

은닉층에 존재하는 하나하나의 노드는 기본 선형 회귀 모델과 동일하게 $wx + b$로 이루어져 있습니다.
하지만 이런 선형 분리를 할 수 있는 모델을 여러개를 모아 비선형 분리를 수행하는 것이 뉴럴 네트워크 입니다.



아래 그림을 보면 4개의 벡터 공간을 선형 분리하는 퍼셉트론들이 하나의 비선형 공간을 분류할 수 있는 벡터 공간을 형성하는 것을 확인할 수 있습니다.
직관적으로는 이해하기 어려우시겠지만, 우리가 케익을 4개의 퍼셉트론들이 분할하는 대로 잘라 가운데 부분을 남기는 것을 생각해보시면 되겠습니다.

• Multi Layer Perceptron Regression

1. 모델 불러오기 및 정의

```
from sklearn.neural_network import MLPRegressor
mlp_regr = MLPRegressor(hidden_layer_sizes=(100,), random_state=2019)
```

2. fit

```
mlp_regr.fit(x_train['RM'].values.reshape((-1, 1)), y_train)
```

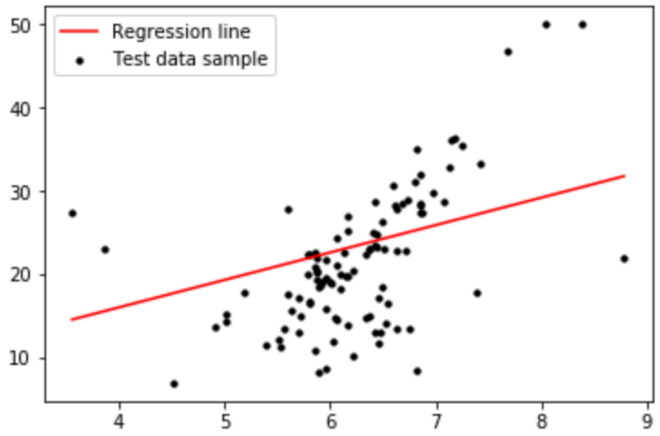
3. predict

```
y_pred = mlp_regr.predict(x_test['RM'].values.reshape((-1, 1)))
```

4. 결과 확인

```
print('단순 MLP 회귀, R2 : {:.4f}'.format(r2_score(y_test, y_pred)))
```

```
단순 MLP 회귀, R2 : 0.2045
```



13개의 변수를 모두 사용해 MLP 회귀 모델을 사용해 보세요. (5분)

```
mlp_regr = MLPRegressor(hidden_layer_sizes=(100,), random_state=2019)
mlp_regr.fit(x_train, y_train)
y_pred = mlp_regr.predict(x_test)
print('다중 MLP 회귀, R2 : {:.4f}'.format(r2_score(y_test, y_pred)))
```

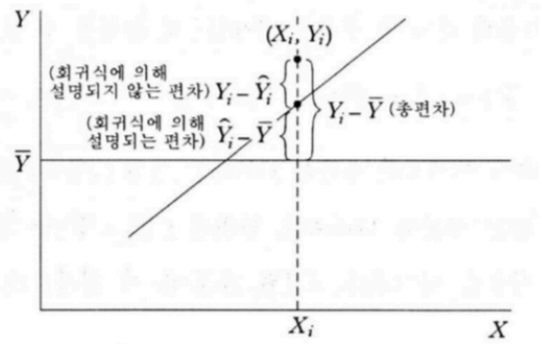
```
다중 MLP 회귀, R2 : 0.4824
```

Evaluation

R^2

Scikit-Learn에서 지원하는 회귀 모델의 평가 방법으로는 R^2가 있습니다.

학습한 회귀 모델이 얼마나 데이터를 잘 표현하는지에 대한 정도를 나타내는 통계적인 척도이며, $0 < R^2 < 1$ 범위의 값을 갖습니다.



$$SST = \sum_{i=1}^{n}(Y_i - \bar{Y})^2 \quad SSR = \sum_{i=1}^{n}(\hat{Y}_i - \bar{Y})^2 \quad SSE = \sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

$$\boldsymbol{SST = SSR + SSE}$$

$$R^2 = 1 - \frac{SSR}{SST}$$

- R^2 = 1, 모델이 데이터를 완벽하게 표현함 (Fits perfectly)
- R^2 = 0, 모델이 데이터를 전혀 표현하지 못함 (Does not explain anything)

• Reference

- 선형 회귀의 기본 가정 : https://kkokkilkon.tistory.com/175
- Wikipedia, Linear Regression : https://ko.wikipedia.org/wiki/선형_회귀
- Wikipedia, R-Square : https://en.wikipedia.org/wiki/Coefficient_of_determination
- Sklearn, Boston dataset : https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_boston.html
- Sklearn, Linear Regression : https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
- Sklearn, Decision Tree Regression : https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html
- Sklearn, Support Vector Machine Regression : https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html
- Sklearn, MLP Regression : https://www.google.com/url?q=http://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html&sa=U&ved=0ahUKEwjsntScreDhAhWMfbwKHWBXAbIQFggOMAU&client=internal-uds-cse&cx=016639176250731907682:tjtqbvtvij0&usg=AOvVaw3lJrRpqB9lWv_Fd98pGuJD
- Sklearn, R-Square : https://www.google.com/url?q=http://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html&sa=U&ved=0ahUKEwicgu3vueDhAhUI9LwKHeLDD3UQFggEMAA&client=internal-uds-cse&cx=016639176250731907682:tjtqbvtvij0&usg=AOvVaw3JYUuCpR-KNsPU189XgvWR