회귀 알고리즘

Kyungsik Han

본 영상에서 다룰 내용

- 회귀 알고리즘 구체적 내용 학습

회귀 분석

$$\boldsymbol{y}^{\wedge} = \boldsymbol{\theta_0} + \boldsymbol{\theta_1 x_1}$$

- $\boldsymbol{x}$: 설명 변수 (explanatory variable)
- $\boldsymbol{y}^{\wedge}$: 반응 변수 (response or target variable)
- $\boldsymbol{\theta_0}$: y 절편 (편향)
- $\boldsymbol{\theta_1}$: 설명 변수 계수

선형 회귀 모델의 예측

$$\boldsymbol{y}^{\wedge} = \boldsymbol{\theta_0} + \boldsymbol{\theta_1 x_1} + \boldsymbol{\theta_2 x_2} + \boldsymbol{\theta_3 x_3} + \cdots + \boldsymbol{\theta_n x_n}$$

선형 회귀 모델의 예측 (벡터 형태)

$$\boldsymbol{y}^{\wedge} = \boldsymbol{h_\theta}(\boldsymbol{x}) = \ \boldsymbol{\theta} \cdot \boldsymbol{X}$$

- $\boldsymbol{\theta}$: θ_0 와 θ_1~ θ_n까지의 특성 가중치를 담은 모델의 파라미터 벡터
- $\boldsymbol{X}$: x_0 ~ x_n 까지 담은 샘플의 특성 벡터 (x_0 = 1)
- $\boldsymbol{\theta} \cdot \boldsymbol{X}$: $\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$
- $\boldsymbol{h_\theta}$: 모델 파라미터 θ 를 사용한 가설(hypothesis) 함수

회귀 분석

$$offset = y^{\wedge} - y$$

- Offset은 반응 값($y^{\wedge}$)과 실제 값(y)의 차이

$$\sum_{i=1}^{n} (y^{\wedge(i)} - y^{(i)})^2$$

- 최소제곱법은 모든 데이터에 대해서 오프셋을 제곱하고 모두 더한 값이 최소가 되는 것이므로 **위의 값이 최소가 되는 회귀 모델을 구하는 것이 회귀 모델의 목표**

아래 두 식은 같은 내용

$$\sum_{i=1}^{n} (h(x^{(i)}) - y^{(i)})^2 \qquad \sum_{i=1}^{n} (y^{\wedge(i)} - y^{(i)})^2$$

RMSE and MAE

CODING X

평균 제곱 오차
(Mean Square Error: MSE)

$$MSE(\boldsymbol{X},\boldsymbol{h}) = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{h}(\boldsymbol{x}^{(i)}) - y^{(i)})^2$$

훈련세트 X에 대한 선형회귀가설 $\boldsymbol{h_\theta}$의 MSE

$$MSE(\boldsymbol{X},\boldsymbol{h_\theta}) = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{x}^{(i)} - y^{(i)})^2$$

평균 제곱근 오차
(Root Mean Square Error: RMSE)

$$RMSE(X,h) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(h(x^{(i)}) - y^{(i)})^2}$$

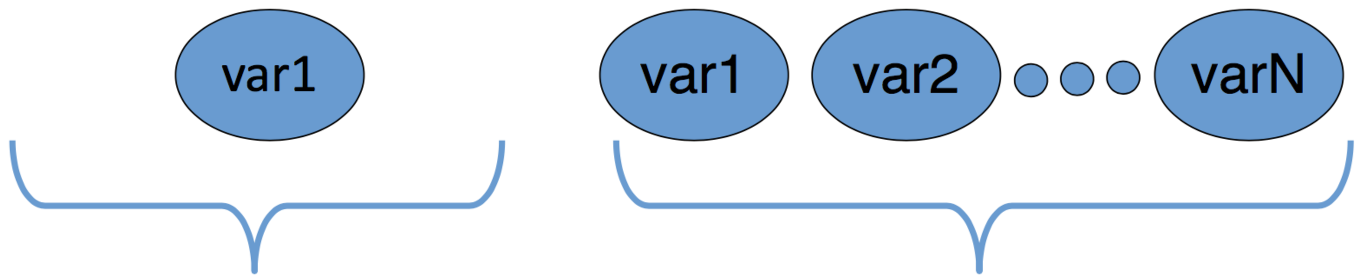
평균 절대 오차
(Mean Absolute Error: MAE)

$$MAE(X,h) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left|h(x^{(i)}) - y^{(i)}\right|}$$

선형 회귀 종류

Simple Linear Regression

Multiple Linear Regression



Input has one variable

Input has >1 variables

선형 회귀 평가

- *F-statistic*
  - 도출된 회귀식이 회귀분석 모델 전체에 대해 통계적으로 의미가 있는지 파악
- *P-value*
  - 각 변수가 종속변수에 미치는 영향이 유의한지 파악
- *R^2 score*
  - 회귀직선에 의하여 설명되는 변동이 총 변동 중에서 차지하고 있는 상대적인 비율이 얼마인지 파악
  - 회귀직선이 종속변수 몇% 를 설명하고 있는지 확인

scikit-learn (or sklearn) library

회귀 코드

In [1]:
```
import numpy as np
from sklearn.linear_model import LinearRegression

x = np.array([[0.0],[1.0],[2.0]])
y = np.array([1.0, 2.0, 2.9])
```

In [2]:
```
lm = LinearRegression()
lm.fit(x, y)

print(lm)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

In [3]:
```
lm.coef_
```

Out[3]:
```
array([ 0.95])
```

In [4]:
```
lm.intercept_
```

Out[4]:
```
1.01666666666671
```

다항 회귀 (Polynomial Regression)

선형 모델 사용

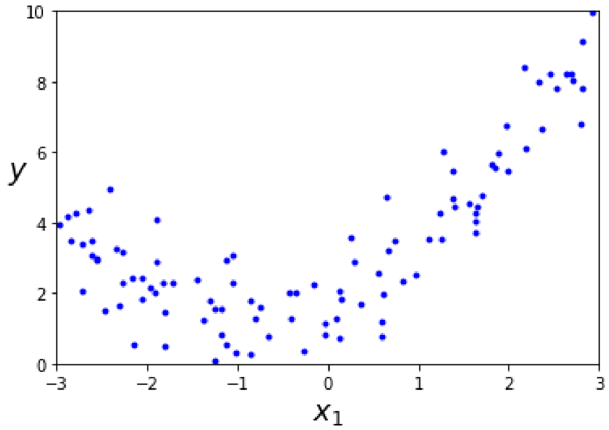
- Polynomial Regression
- 비선형 데이터에 선형 모델 사용

랜덤 데이터 생성

In [16]:
```
m = 100
X = 6 * np.random.rand(m,1) - 3
y = 0.5 * X**2 + X + 2 + np.random.randn(m,1)
```

$$y = 0.5x^2 + x + constant$$

In [18]:
```
plt.plot(X, y, "b.")
plt.xlabel("$x_1$", fontsize=18)
plt.ylabel("$y$", rotation=0, fontsize=18)
plt.axis([-3, 3, 0, 10])
plt.show()
```



다항 회귀 예제

CODING X

다항식 변환

In [7]:
```
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model  import LinearRegression

poly_features = PolynomialFeatures(degree=2, include_bias=False)
X_poly = poly_features.fit_transform(X) # X_poly is y_pred
```

임의의 2차원 다항식을 만들고,
X를 대입하여 X_ploy 생성

In [8]:
```
X[0]
```
Out[8]:
```
array([2.01206324])
```

In [9]:
```
X_poly[0]
```
Out[9]:
```
array([2.01206324, 4.04839848])
```

In [10]:
```
## Linear Regression
lin_reg = LinearRegression()
lin_reg.fit(X_poly, y)
```
Out[10]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

In [11]:
```
lin_reg.intercept_, lin_reg.coef_
```
Out[11]:
```
(array([1.83388505]), array([[1.04598013, 0.51895615]]))
```

다음 영상에서 배울 내용

- 지도학습 회귀(regression) 실습

CODINGX

수고하셨습니다