非線形回帰モデル

$\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ をラベル付けされたデータとし、N をデータの数、$\mathbf{x}_i$ を D 次元特徴ベクトル、y_i を $\mathbf{x}_i$ のラベルとする。$\mathbf{w}$ を D' 次元ベクトル、b を実数、$\phi : \mathbb{R}^D \to \mathbb{R}^{D'}$ を非線形関数とする。

$$f_{\mathbf{w},b}(\mathbf{x}) := \mathbf{w}\phi(\mathbf{x}) + b$$

とおく。この式を用いて、未知の D 次元特徴ベクトル $\mathbf{x}$ に対して、ラベル $y = f_{\mathbf{w},b}(\mathbf{x})$ を予測する。最適な $\mathbf{w}, b$ は

$$\min_{\mathbf{w},b} \frac{1}{N} \sum_{i=1}^{N} (f_{\mathbf{w},b}(\mathbf{x}_i) - y_i)^2$$

で求められる。ϕ は**基底関数**と呼ばれる。

例 0.1. (コードは非線形回帰モデル.ipynb) $D = 1, \phi(x) = (x, x^2)$ である場合を考える。

```
#データを生成
import numpy as np
m = 100
X = 6 * np.random.rand(m, 1) - 3
y = 0.5 * X**2 + X + 2 + np.random.randn(m, 1)

from sklearn.preprocessing import PolynomialFeatures
poly_features = PolynomialFeatures(degree=2, include_bias=False)
X_poly = poly_features.fit_transform(X) # φによって、データ x を (x,x^{2}) に変換

X[0] #変換前のデータ
> array([1.32268224])

X_poly[0] #変換後のデータ
>array([1.32268224, 1.7494883 ])

from sklearn.linear_model import LinearRegression
lin_reg = LinearRegression() #線形回帰モデルを選択
lin_reg.fit(X_poly, y) #最適解を求める

lin_reg.coef_ #w の最適解
```

```
>array([[0.94041935, 0.49876073]])

lin_reg.intercept_ #b の最適解
>array([1.98435621])
```

参考文献

[1] Andriy Burkov. (2019). The hundred-page machine learning book.

[2] Marc Peter Deisenroth., A. Aldo Faisal., Cheng Soon Ong. (2020). Mathematics for machine learning. Cambridge University Press.

[3] Aurëlien Gëron. (2019). Hands-on machine learning with Scikit-Learn, Keras & TensorFlow. 2nd Edition. Oreilly.

[4] 小縣信也., 斎藤翔汰., 溝口聡., 若杉一幸. (2021). ディープラーニング E 資格エンジニア問題集 インプレス.

[5] Sebastian Raschka., Vahid Mirjalili. (2019). Python machine learning. Third Edition. Packt.