非線形回帰モデル (NumPy)

In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
plt.style.use('ggplot')
```

多項式回帰

訓練データ生成

In [2]:

```
# パラメータ設定
n_samples = 10
var = .25
```

In [3]:

```
def sin_func(x):
    """ sin波のデータ作成 """
    return np.sin(2 * np.pi * x)
```

In [4]:

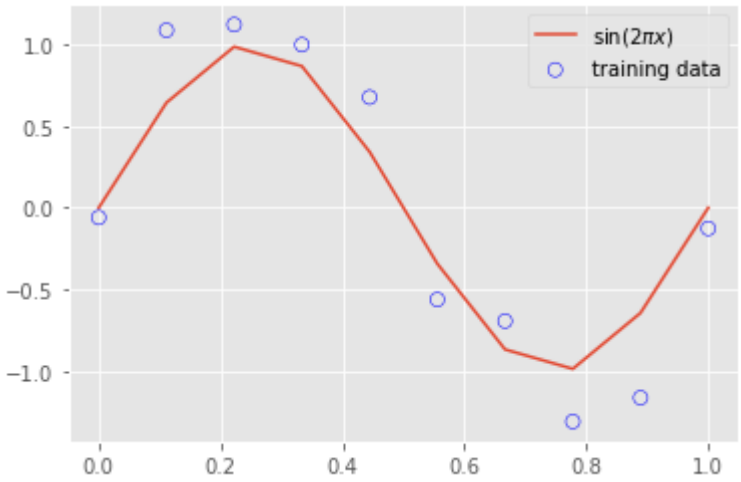
```
def add_noise(y_true: np.ndarray, var: float) -> np.ndarray:
    """ 正規分布に従うノイズを追加する """
    noise = np.random.normal(scale=var, size=y_true.shape)  # scale : 標準偏差
    y_noise = y_true + noise
    return y_noise
```

In [5]:

```
xs = np.linspace(0, 1, n_samples)
ys_true = sin_func(xs)  # 元データ
ys = add_noise(ys_true, var)  # ノイズ追加
```

In [6]:

```
plt.scatter(xs, ys, facecolor="none", edgecolor="b", s=50, label="training data")
plt.plot(xs, ys_true, label="$\sin(2\pi x)$")
plt.legend()
plt.show()
```



学習

求める回帰係数 $\hat{w}$ は以下の式となる。

$\hat{w} = \left(\Phi^{\mathrm{T}}\Phi\right)^{-1}\Phi^{\mathrm{T}}y$

In [7]:

```
def polynomial_features(xs, degree=3):
    """
    入力されたNumpy配列を多項式特徴ベクトルΦに変換
    X = [[1, x1, x1^2, x1^3],
         [1, x2, x2^2, x2^3],
         ...
         [1, xn, xn^2, xn^3]]
    """
    X = np.ones((len(xs), degree + 1))  # すべての要素が1の行列を作成
    X_t = X.T

    for i in range(1, degree + 1):
        X_t[i] = X_t[i - 1] * xs

    return X_t.T
```

In [8]:

```
def regression_coefficient(phi: np.ndarray, ys: np.ndarray) -> np.ndarray:
    """ 回帰係数wを求める"""
    tmp = np.dot(phi.T, phi)
    tmp = np.linalg.inv(tmp)
    tmp = np.dot(tmp, phi.T)
    result = np.dot(tmp, ys)
    return result
```

In [9]:

```
phi = polynomial_features(xs)
w = regression_coefficient(phi, ys)
```

In [13]:

```
print(phi)
print(w)
```

```
[[1.         0.         0.         0.        ]
 [1.         0.11111111 0.01234568 0.00137174]
 [1.         0.22222222 0.04938272 0.01097394]
 [1.         0.33333333 0.11111111 0.03703704]
 [1.         0.44444444 0.19753086 0.0877915 ]
 [1.         0.55555556 0.30864198 0.17146776]
 [1.         0.66666667 0.44444444 0.2962963 ]
 [1.         0.77777778 0.60493827 0.47050754]
 [1.         0.88888889 0.79012346 0.70233196]
 [1.         1.         1.         1.        ]]
[ -0.05122573  13.42780129 -39.15716865  25.60304391]
```

予測

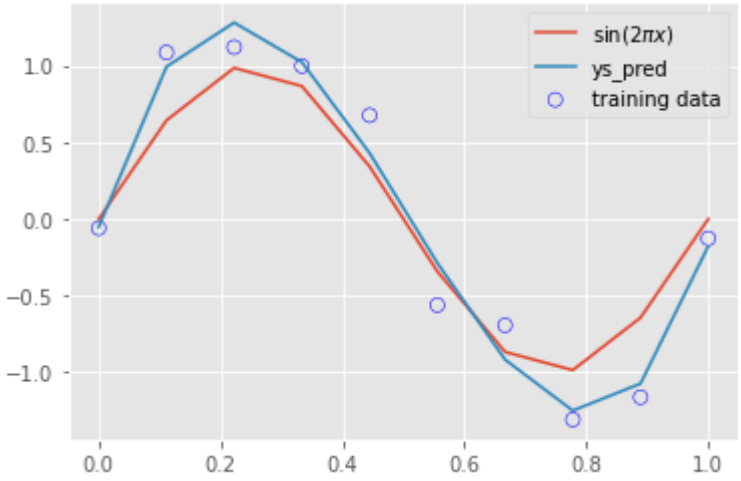
上で求めた $\hat{w}$ を使用して、$y_{pred} = \hat{w}\phi(x)$ $(y(x) = \Phi\hat{w})$ で y_{pred} を予測する。

In [11]:

```
ys_pred = np.dot(phi, w)
```

In [12]:

```
plt.scatter(xs, ys, facecolor='none', edgecolor='b', s=50, label='training data')
plt.plot(xs, ys_true, label='$\sin(2\pi x)$')
plt.plot(xs, ys_pred, label='ys_pred')
plt.legend()
plt.show()
```



考察：
予測値は元データのsin波に近いグラフとなっているが、ノイズを加えた値（学習データ）のほうに近くなっている。

In [ ]:

```
```