Artificial Intelligence (EI06024001)

Assignment 3:
Linear Regression (Inverse Mat)

Reminder: Computing the MLE

$$\widehat{\boldsymbol{\theta}}_{MLE} = \left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{Y}$$

Computing tools: Numpy

$\boldsymbol{X^T}$	np.transpose(X)	
$\boldsymbol{AB}$	np.matmul(A,B)	
$\boldsymbol{X^{-1}}$	inv(X)	from numpy.linalg import inv

main_linea_reg_mat.py (1)

```python
if __name__ == '__main__':
    # STEP 1: SET DATA ---------------------------------------------------------
    data_x = np.linspace(1.0, 10.0, 100)[:, np.newaxis]
    data_y = np.sin(data_x) + 0.1 * np.power(data_x, 2) + 0.5 * np.random.randn(100, 1)
    data_x /= np.max(data_x)

    data_x = np.hstack((np.ones_like(data_x), data_x))

    # Set train & test data
    order = np.random.permutation(len(data_x))
    portion = 20
    x_test = data_x[order[:portion]]
    y_test = data_y[order[:portion]]
    x_train = data_x[order[portion:]]
    y_train = data_y[order[portion:]]
```

data_x (N, 2)			data_y (N, 1)
	1	0.234	0.414
	1	0.148	0.245
	1	0.532	0.711
	…		…

main_linea_reg_mat.py (2)

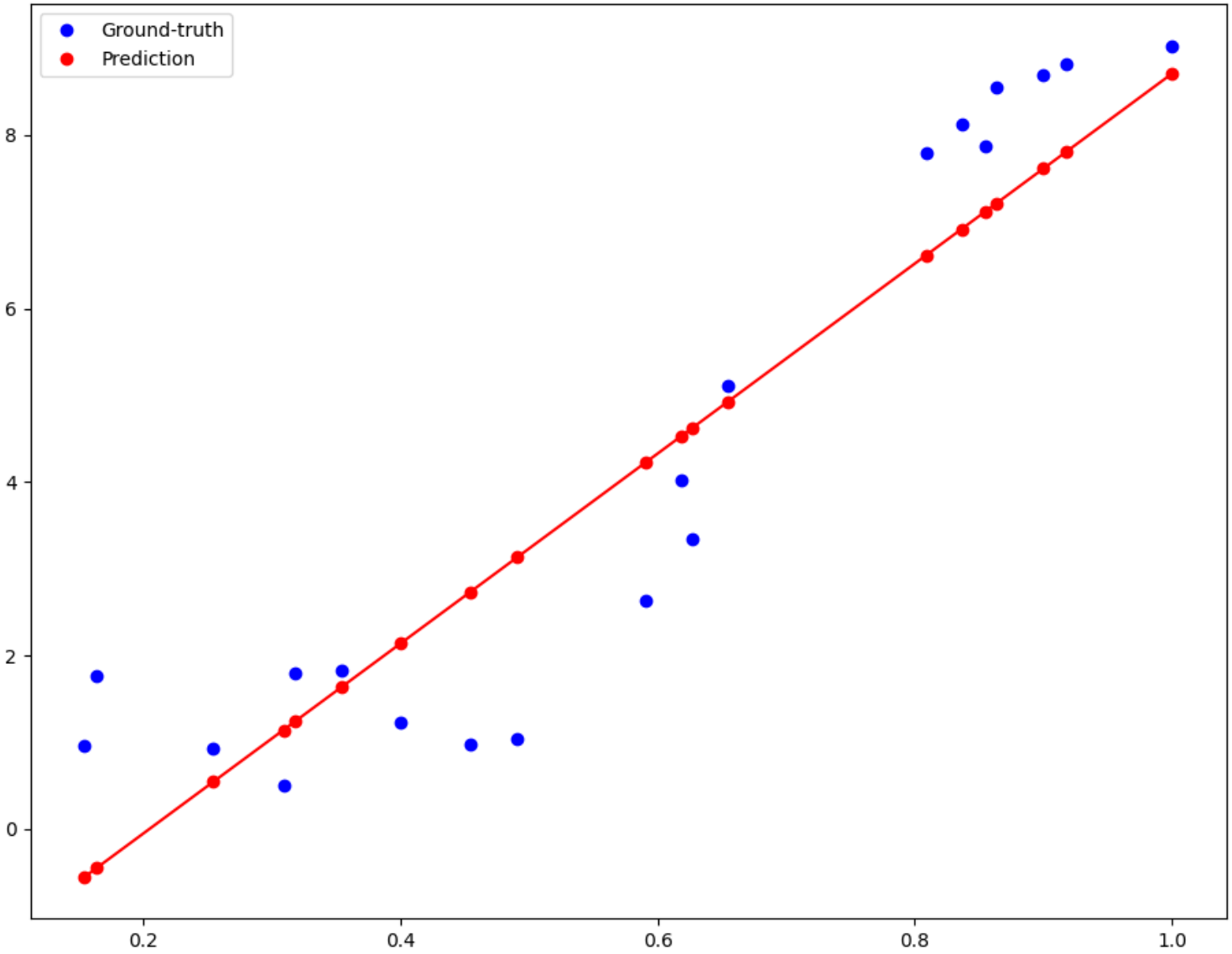
```
# STEP 2: DO PREDICTION ----------------------------------------------
# Write code here!
# Find theta_hat.
theta_hat = np.ones((2, 1))  # 코드 작성시 이부분을 지우세요!
y_pred = np.matmul(x_test, theta_hat)
```

$$\hat{\theta}_{\mathrm{MLE}} = (X^T X)^{-1} X^T Y$$

X → x_train

Y → y_train

Expected Result



Additional Work: Using Nonlinear Basis

Apply nonlinear basis!

$$\hat{\theta}_{MLE} = \left(\phi^{\mathrm{T}}\phi\right)^{-1}\phi^{\mathrm{T}}Y$$

Use Gaussian basis

$$\phi_j(x) = \exp\left(-\frac{(x-\mu_j)^2}{2s^2}\right),$$

where $\mu_1 = 0.0, \mu_2 = 0.2, \mu_3 = 0.4, \mu_4 = 0.6$ and $s = 0.5$.

Basis matrix:

$$\boldsymbol{\phi} = \begin{bmatrix} \phi(x_1) \\ \phi(x_2) \\ \vdots \\ \vdots \\ \phi(x_N) \end{bmatrix}$$

$\phi(x_1) \rightarrow 1, \phi_1(x_1), \phi_2(x_1), \phi_3(x_1), \phi_4(x_1)$

$\rightarrow N \times 5$

main_linea_reg_mat_basis.py

```python
# STEP 2: DO PREDICTION ----------------------------------------------
# Nonlinear basis function
def apply_exp_basis(X, s=0.5):
    # Write code here (1)!
    Phi = np.copy(X)  # 코드 작성시 이부분을 지우세요!
    return Phi

phi_train = apply_exp_basis(x_train)

# Write code here (2)!
# Find theta_hat.
theta_hat = np.ones((2, 1))  # 코드 작성시 이부분을 지우세요!

phi_test = apply_exp_basis(x_test)
y_pred = np.matmul(phi_test, theta_hat)
```

Expected Result: Using Nonlinear Basis