Fundamentals of Machine Learning (Fall 2022)

Homework #1 (80 Pts, Due date: Sep 21, 2022)

Student ID ______________________

Name ______________________

(1) Given n training samples, $x_1, x_2, .., x_n \in \mathbb{R}$, we want to find a constant w_0 that minimizes the following error function.

$$E(w_0) = \frac{1}{n}\sum_{i=1}^{n}(x_i - w_0)^2$$

Assume that we have five training samples ($n = 5$) such that $x_1 = 1, x_2 = 3, x_3 = 4, x_4 = 5, x_5 = 9$.

(a) [10 pts] Calculate the optimal w_0 using an analytical solution.

Answer:

(b) [10 pts] Explain the meaning of the optimal solution in terms of normal distribution $\mathcal{N}(\mu, \sigma^2)$.

Answer:

(2) We provide all template code and datasets in Python. Write your code to implement linear regression. You need to install NumPy and Matplotlib libraries.

(a) [5 pts] Implement the util function "add_bias" in 'models/LinearRegression.py.' You should add a column of ones for bias after the last column of the input matrix.

Note: Fill in your code (EDIT HERE part). You also have to submit your code to i-campus.

Answer:

(b) [5 pts] Implement the training function "numerical_solution" in 'models/LinearRegression.py' using the **batch gradient descent method**. The error function is defined as follows:

$$\hat{y}_i = w_0 + \sum_j w_j x_{ij}$$

$$E(\mathbf{w}) = \frac{1}{2n} \sum_i (y_i - \hat{y}_i)^2$$, where n is the number of samples.

Note: Fill in your code (EDIT HERE part). You also have to submit your code to i-campus.

Answer:

(c) [10 pts] Implement training function "numerical_solution" in 'models/LinearRegression.py' using the **mini-batch stochastic gradient descent method**. The error function is defined as follows:

$$E(w) = \frac{1}{2|\mathcal{B}|}\sum_i (y_i - \hat{y_i})^2 \text{ , where } |\mathcal{B}| \text{ is the number of the minibatch samples.}$$

Note: Fill in your code (EDIT HERE part). You also have to submit your code to i-campus.

Answer:

(d) [10 pts] Implement the training function "analytical_solution" in 'models/LinearRegression.py' using the **normal equation**.

Note: Fill in your code (EDIT HERE part). You also have to submit your code to i-campus.

Answer:

(3) Evaluate your code using two datasets, "Wave" and "Diabetes."

(a) [10 pts] After training your model on the "Diabetes" dataset, fill the blank using the following metrics.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y^{(i)} - f\left(x^{(i)}\right)\right)^2}$$

Write your opinion briefly on two solutions, the analytical solution and the gradient descent method. (Hyperparameter for gradient: Epoch = 10,000, Batch size = 32, learning rate = 0.01, optimizer='SGD')

Answer: Fill the blank in the table.

	$\boldsymbol{w_0}$	$\boldsymbol{w_1}$	$\boldsymbol{w_2}$	**RMSE**
Initial value	0.0	0.0	0.0	168.7240
Gradient Descent				
Analytic solution				

(b) [10 pts] For the "Wave" dataset, draw the plots by adjusting learning rates, where the other hyperparameters are the same as (a). For each plot, the x-axis is # of iterations, and the y-axis is the error value. Try at least five different learning rates.

Answer:

(c) [10 pts] For the "Wave" dataset, draw the plots by adjusting batch sizes, where the other hyperparameters are the same as (a). For each plot, the x-axis is # of iterations, and the y-axis is the error value. Try at least five different values.

Answer: