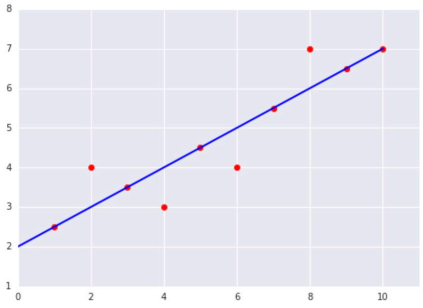
ASSIGNMENT 2

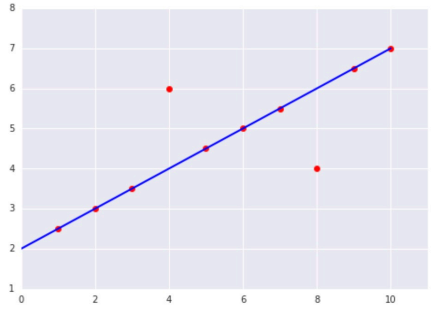
Deadline: 11:59 pm, March 10, 2023

Submit via Blackboard with VeriGuide receipt.

Please follow the course policy and the school's academic honesty policy.
Each question is worth ten points.

1. Explore the options below. Which of the two data sets shown in the preceding plots has the higher Mean Squared Error (MSE)? The dataset is on the left or on the right? Please state your reasons.





2. Answer the following questions based on the hyperbolic tangent.

$$\tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}}$$

(1) How is $\tanh$ related to the logistic function θ? *Hint: shift and scale*

(2) Show that $\tanh(s)$ converges to a hard threshold for large $|s|$, and converges to no threshold for small $|s|$ *Hint: Formalize the figure below.*

3. Consider the scenario where we are learning from data where the target variable y can take on values ± 1. Given a noisy target $P(y \mid x)$ and a candidate hypothesis h, your task is to demonstrate that the maximum likelihood method can be equivalently viewed as the task of finding the hypothesis h that minimizes the cross-entropy error measure.

(1) Formulate the cross-entropy error measure that is to be minimized. Your formulation should be expressed in terms of the hypothesis h, the data points x_n, and the corresponding target values y_n.

$$E_{in}(w) = \sum_{n=1}^{N} \left[(y_n = +1) \ln \frac{1}{h(x_n)} + (y_n = -1) \ln \frac{1}{1 - h(x_n)} \right]$$

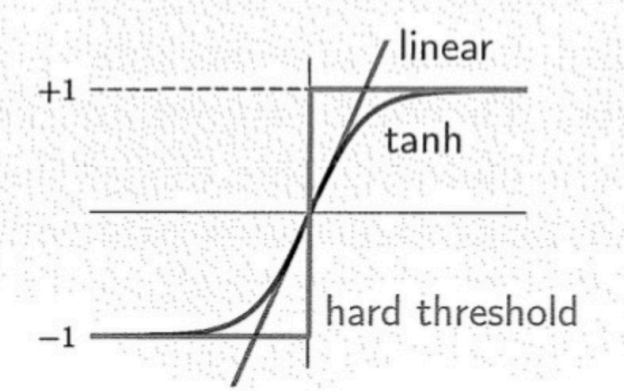


Figure 1: A graph showing the tanh function and its relation to the hard threshold.

4. We define the function $f(x)$ as

$$f(x) = \frac{a}{1 + e^{kx+b}},$$

where a, k, b are constants.

(1) Discuss the monotonicity of $f(x)$.

(2) Determine the value range of $f(x)$ based on question (1).

(3) Prove the equation $\frac{d}{dx} f(x) = \frac{k}{a} f(x)(f(x) - a)$.

5. For a given matrix X, where X is an N by $d + 1$ matrix, and $\mathrm{X^T X}$ is invertible. We define the matrix $\mathrm{H = X(X^T X)^{-1} X^T}$.

(1) Prove that H is symmetric. A symmetric matrix is a square matrix that is equal to its transpose.

Hint: $\mathrm{(AB)^T = B^T A^T}$.

(2) Prove that $\mathrm{H^K = H}$ for any positive integer K.

(3) if I is the identity matrix of size N, show that $\mathrm{(I - H)^K = I - H}$ for any positive integer K.

(4) Show that $\mathrm{trace(H) = d + 1}$, where the trace is the sum of diagonal elements.

Hint: $\mathrm{trace(AB) = trace(BA)}$.

6. Logistic regression algorithm implementation via Python. Consider a logistic regression method in a two-dimensional case ($d = 2$). The data is given in the Table 1. Please compress the related codes into a ZIP file and submit it along with your PDF answer.

a	4	5	12	29	30	36	36	54	58	70	72	76	78	82	87	90	90	92	95	98
b	49	4	28	18	65	32	1	29	76	12	26	55	4	15	95	70	55	84	14	21
label	0	1	1	1	1	1	1	1	0	1	0	0	1	1	0	0	0	0	0	0

Table 1: The dataset has 20 data points with two attributes, namely a and b, and a label. Each column represents a data point.

(1) Visualize the 2-D data with a scatter plot, where the positive samples (label=1) are marked in red and the negative samples (label=0) are marked in blue (Please paste the figure to the answer sheet).

(2) Run the logistic regression method on the given dataset (Please copy the Python code to the answer sheet), and calculate the accuracy on the training dataset. Please be careful not to use the encapsulated

logistic regression method. You should use Numpy to implement forward propagation and backward propagation here.

(3) Try at least 5 different learning rates and report the errors with different learning rates.

(4) Choose the learning rate with the smallest loss, and then visualize the plot of training errors as the training step increases, where the x-axis and y-axis present the training step and loss, respectively (Please paste the figures to the answer sheet). And estimate how many steps would it take for the model to reach convergence according to the figure.

Hint:

1. **matplotlib.pyplot.scatter** draws a scatter plot of y vs. x with varying marker size and/or color.

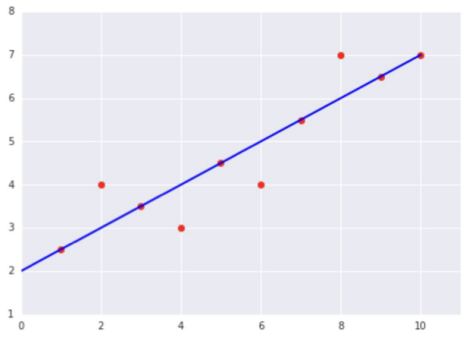
2. **matplotlib.pyplot.plot** plots y versus x as lines and/or markers.

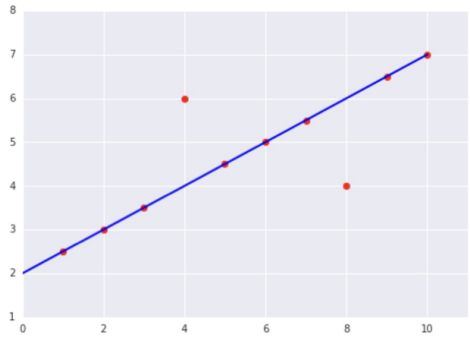
3. **Numpy.dot** calculates the dot product of two arrays.

4. **Numpy.exp** and **Numpy.log** calculate the exponential and logarithm of all elements in the input array.

*** END ***

1. Explore the options below. Which of the two data sets shown in the preceding plots has the higher Mean Squared Error (MSE)? The dataset is on the left or on the right? Please state your reasons.





data on the right has higher MSE

in both graphs, there are in total 10 points,

in the right graph. 4 points has a difference of 1, the others have no error

in the left graph. 2 points has a difference of 2, the others have no error

by definition of MSE,

error of left $E_{left} = \frac{1}{10} \times (1^2 \times 4) = 0.4$

error of right. $E_{right} = \frac{1}{10} \times (2^2 \times 2) = 0.8$

$\therefore E_{left} < E_{right}$

So data on the right has higher MSE

2. Answer the following questions based on the hyperbolic tangent.

$$\tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}}$$

(1) How is $\tanh$ related to the logistic function θ? *Hint: shift and scale*

(2) Show that $\tanh(s)$ converges to a hard threshold for large $|s|$, and converges to no threshold for small $|s|$ *Hint: Formalize the figure below.*

(1).

$$\tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}} = \frac{2e^s - e^s - e^{-s}}{e^s + e^{-s}} - 1 = 2\frac{e^s}{e^s + e^{-s}} - 1$$

$$= 2\frac{e^{2s}}{e^{2s}+1} - 1$$

$$= 2\theta(2s) - 1$$

so if we plot1 $y_1 = \theta(s)$; plot2: $y_2 = \tanh(s)$,

then plot2 can be obtained by

first scale plot1 doubled along the y-axis

and scale plot1 half along the s- axis,

finally shift plot1 downward by 1 unit.

(2):

$$\tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}} = \frac{1 - e^{-2s}}{1 + e^{-2s}}$$

$$\because \lim_{s\to\infty} e^{-2s} = 0$$

$$\Rightarrow \lim_{s\to\infty} \tanh(s) = \frac{1-0}{1+0} = 1$$

$$\tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}} = \frac{e^{2s} - 1}{e^{2s} + 1}$$

$$\because \lim_{s\to-\infty} e^{2s} = 0$$

$$\Rightarrow \lim_{s\to-\infty} \tanh(s) = \frac{0-1}{0+1} = -1$$

also, from (1): $\tanh(s) = 2\frac{e^{2s}}{e^{2s}+1} - 1 = 2\frac{1}{1+\frac{1}{e^{2s}}} - 1$

$\therefore$ $\tanh(s)$ is increasing $\Rightarrow$ $-1 < \tanh(s) < 1$

$\therefore$ when $|s|$ is big, $-1, 1$ are hard threshold for $\tanh(s)$

when $|s|$ is small,

by taylor expension, $e^s = 1 + s + \frac{s^2}{2} + O(s^3)$

$$\Rightarrow e^{-s} = 1 - s + \frac{s^2}{2} - O(s^3)$$

$$\Rightarrow \tanh(s) = \frac{e^s - e^{-s}}{e^s + e^{-s}} \approx \frac{2s + 2O(s^3)}{2 + s^2}$$

when $|s| \to 0$, we can ignore the higher order terms

$$\Rightarrow \tanh(s) \approx \frac{2s}{2} = s$$

$\Rightarrow$ tanh(s) is approximately linearly for small $|s|$, and therefore no threshold

3. Consider the scenario where we are learning from data where the target variable y can take on values ± 1. Given a noisy target $P(y \mid x)$ and a candidate hypothesis h, your task is to demonstrate that the maximum likelihood method can be equivalently viewed as the task of finding the hypothesis h that minimizes the cross-entropy error measure.

(1) Formulate the cross-entropy error measure that is to be minimized. Your formulation should be expressed in terms of the hypothesis h, the data points x_n, and the corresponding target values y_n.

$$E_{in}(w) = \sum_{n=1}^{N} \left[(y_n = +1) \ln \frac{1}{h(x_n)} + (y_n = -1) \ln \frac{1}{1 - h(x_n)} \right]$$

to find the optimized h, we are looking for h that maximizes the probability

$P = PC(x_1, y_1), (x_2, y_2), \cdots (x_N, y_N))$

since data are independent:

$$P = \prod_{n=1}^{N} P(y_n | x_n) = \prod_{n=1}^{N} \left[(y_n = +1) \cdot h(x_n) + (y_n = -1) \cdot (1 - h(x_n)) \right]$$

which is the same as maximizing $\ln P$ (since $f(x) = \ln x$ is monotonically increasing)

$$\ln P = \ln \prod_{n=1}^{N} \left[(y_n = +1) \cdot h(x_n) + (y_n = -1) \cdot (1 - h(x_n)) \right]$$

$$= \sum_{n=1}^{N} \ln \left[(y_n = +1) \cdot h(x_n) + (y_n = -1) \cdot (1 - h(x_n)) \right]$$

$$= \sum_{n=1}^{N} \left[(y_n = +1) \ln h(x_n) + (y_n = -1) \ln (1 - h(x_n)) \right]$$

(since either $\begin{cases} (y_n = +1) = 1 \\ (y_n = -1) = 0 \end{cases}$ or $\begin{cases} (y_n = +1) = 0 \\ (y_n = -1) = 1 \end{cases}$)

maximizing $\ln P$ is the same with minimizing $-\ln P$

$$-\ln P = -\sum_{n=1}^{N} \left[(y_n = +1) \ln h(x_n) + (y_n = -1) \ln (1 - h(x_n)) \right]$$

$$= \sum_{n=1}^{N} - \left[(y_n = +1) \ln h(x_n) + (y_n = -1) \ln (1 - h(x_n)) \right]$$

$$= \sum_{n=1}^{N} \left[(y_n = +1) \cdot \ln \frac{1}{h(x_n)} + (y_n = -1) \cdot \ln \frac{1}{(1 - h(x_n))} \right]$$

(since either $\begin{cases} (y_n = +1) = 1 \\ (y_n = -1) = 0 \end{cases}$ or $\begin{cases} (y_n = +1) = 0 \\ (y_n = -1) = 1 \end{cases}$)

$$= E_{in}(w)$$

So to maximize the likelihood $P = PC(x_1, y_1), (x_2, y_2), \cdots (x_N, y_N))$ can be inferred as minimizing the cross entropy error

$$E_{in}(w) = \sum_{n=1}^{N} \left[(y_n = +1) \cdot \ln \frac{1}{h(x_n)} + (y_n = -1) \cdot \ln \frac{1}{(1 - h(x_n))} \right]$$

4. We define the function $f(x)$ as

$$f(x) = \frac{a}{1 + e^{kx+b}},$$

where a, k, b are constants.

(1) Discuss the monotonicity of $f(x)$.

(2) Determine the value range of $f(x)$ based on question (1).

(3) Prove the equation $\frac{d}{dx} f(x) = \frac{k}{a} f(x)(f(x) - a)$.

(1).

$$\frac{d}{dx} f(x) = a \cdot \frac{d}{dx}(1 + e^{kx+b}) \cdot \left[- \frac{1}{(1+e^{kx+b})^2} \right]$$

$$= - \frac{e^{kx+b}}{(1+e^{kx+b})^2} \cdot ak$$

Since $- \frac{e^{kx+b}}{(1+e^{kx+b})^2} < 0$

∴ ①: $ak \geq 0$, ie a, k have same sign or at least one is 0,

$\frac{d}{dx} f(x) \leq 0$, $f(x)$ monotonically decreases

since constant is also considered as monotonically decreasing

②: $ak < 0$, ie a, k have different sign (and none is 0)

$\frac{d}{dx} f(x) > 0$ $f(x)$ monotonically increases

(2)

from (1), since $f(x)$ monotonically decreases or increases, $f(x)$ approaches its max or min when x approaches infinity

when $k > 0$

$$\lim_{x \to \infty} f(x) = 0 \quad , \quad \lim_{x \to -\infty} = a$$

when $k = 0$

$$\lim_{x \to \infty} f(x) = \lim_{x \to -\infty} f(x) = \frac{a}{1+e^b}$$

when $k < 0$

$$\lim_{x \to \infty} f(x) = a, \quad \lim_{x \to -\infty} f(x) = 0$$

⇒ ①: $k = 0$. or $a = 0$ $f(x)$ can only be $\frac{a}{1+e^b}$

②: $k \neq 0$, $a > 0$ $f(x) \in (0, a)$

③. $k \neq 0$, $a < 0$ $f(x) \in (a, 0)$

(3) proof:

$$\frac{k}{a} f(x)\,(f(x)-a)$$

$$= \frac{k}{a} \cdot \frac{a}{1+e^{kx+b}} \cdot \left(\frac{a}{1+e^{kx+b}} - a\right)$$

$$= \frac{k}{1+e^{kx+b}} \cdot \frac{a - a - ae^{kx+b}}{1+e^{kx+b}}$$

$$= \frac{k \cdot}{1+e^{kx+b}} \cdot \frac{-ae^{kx+b}}{1+e^{kx+b}}$$

$$= -ak \cdot \frac{e^{kx+b}}{(1+e^{kx+b})^2}$$

Since from (1)

$$\frac{d}{dx} f(x) = -\frac{e^{kx+b}}{(1+e^{kx+b})^2} \cdot ak$$

$$\Rightarrow \frac{d}{dx} f(x) = \frac{k}{a} f(x)\,(f(x)-a)$$

5. For a given matrix X, where X is an N by $d+1$ matrix, and $\mathrm{X^TX}$ is invertible. We define the matrix $\mathrm{H = X(X^TX)^{-1}X^T}$.

(1) Prove that H is symmetric. A symmetric matrix is a square matrix that is equal to its transpose.
Hint: $\mathrm{(AB)^T = B^TA^T}$.

(2) Prove that $\mathrm{H^K = H}$ for any positive integer K.

(3) if I is the identity matrix of size N, show that $\mathrm{(I-H)^K = I-H}$ for any positive integer K.

(4) Show that $\mathrm{trace(H) = d+1}$, where the trace is the sum of diagonal elements.
Hint: $\mathrm{trace(AB) = trace(BA)}$.

(1). proof:

$$H^T = (X(X^TX)^{-1}X^T)^T = (X^T)^T\left[(X^TX)^{-1}\right]^T \cdot X^T$$

since $(X^T)^T = X$

$$\left[(X^TX)^{-1}\right]^T = \left[(X^TX)^T\right]^{-1}$$

$$= \left[X^T(X^T)^T\right]^{-1}$$

$$= (X^TX)^{-1}$$

$$\Rightarrow H^T = (X^T)^T\left[(X^TX)^{-1}\right]^T \cdot X^T$$

$$= X(X^TX)^{-1}X^T$$

$$= H$$

$\therefore$ H is symmetric

(2) proof by induction:

let hypothesis $P(K): H^K = H$

① when $K=1$: $H^1 = H$ true

② when $K \geq 1$, assume $P(K)$ true: $H^K = H$

then $H^{K+1} = H^K \cdot H = H \cdot H$

$$= X(X^TX)^{-1}X^TX(X^TX)^{-1}X^T$$

$$= X\left[(X^TX)^{-1}X^TX\right](X^TX)^{-1}X^T$$

$$= X(X^TX)^{-1}X^T$$

$$= H$$

$\Rightarrow P(K+1)$ true

③ by induction $H^K = H$ true for all positive integer

(3) proof by induction:

let hypothesis $P(k)$: $(I-H)^k = I-H$

①: when $k=1$ $(I-H)^1 = I-H$ $\therefore$ $P(k)$ true for $k=1$

②: when $k \geq 1$, assume $P(k)$ true: $(I-H)^k = I-H$

then $(I-H)^{k+1} = (I-H)^k (I-H)$

$= (I-H)(I-H)$

$= I^2 - HI - IH + H^2$

$= I - 2H + H^2$

Since from (2) $H^k = H$ for any positive integer

$\Rightarrow H^2 = H$

$\Rightarrow I - 2H + H^2 = I - 2H + H$

$= I - H$

$\Rightarrow P(k+1)$ true

③. by induction $H^k = H$ true for all positive integer

(4). proof:

$\text{trace}(H) = \text{trace}(X(X^TX)^{-1}X^T)$

$= \text{trace}([X(X^TX)^{-1}]X^T)$

$= \text{trace}(X^T[X(X^TX)^{-1}])$

$= \text{trace}((X^TX)(X^TX)^{-1})$

$= \text{trace}(I_{d+1})$ (where I_{d+1} means $d+1$ by $d+1$ identity matrix

$= \sum_{i=1}^{d+1} (I_{d+1})_{ii}$

$= d+1$

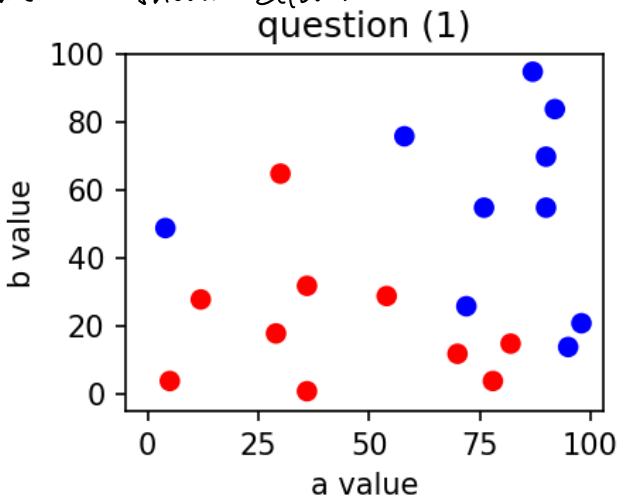
6. Logistic regression algorithm implementation via Python. Consider a logistic regression method in a two-dimensional case ($d = 2$). The data is given in the Table 1. Please compress the related codes into a ZIP file and submit it along with your PDF answer.

a	4	5	12	29	30	36	36	54	58	70	72	76	78	82	87	90	90	92	95	98
b	49	4	28	18	65	32	1	29	76	12	26	55	4	15	95	70	55	84	14	21
label	0	1	1	1	1	1	1	1	0	1	0	0	1	1	0	0	0	0	0	0

Table 1: The dataset has 20 data points with two attributes, namely a and b, and a label. Each column represents a data point.

(1) Visualize the 2-D data with a scatter plot, where the positive samples (label=1) are marked in red and the negative samples (label=0) are marked in blue (Please paste the figure to the answer sheet).

the python file is attached with this pdf.
the figure is shown below:



(2) Run the logistic regression method on the given dataset (Please copy the Python code to the answer sheet), and calculate the accuracy on the training dataset. Please be careful not to use the encapsulated logistic regression method. You should use Numpy to implement forward propagation and backward propagation here.

note: about stopping criterion: I tested the total-steps of the training model for different numbers

(eg. 1,000; 10,000; 100,000; 500,000; 1,000,000),

and found that total-steps = 100,000 will allow most models to

sufficiantly converge (as shown in the visualization in question (4))

so I set total-steps to be 100,000 as the hard stopping criterion

the python code is shown below

(1): gradient descent method.

```python
def grad_descent(X, Y, lr = 0.001, total_steps = 100000, prsc_report=False, precise=100):
    w = np.zeros(3)
    error_list = []

    for step in tqdm(range(total_steps)):
        total_error = 0
        total_grad = 0

        # forward
        for x, y in zip(X, Y):
            # cal error
            h = logistic_func(np.dot(x, w))
            total_error += - np.sum(y * np.log(h) + (1 - y) * np.log(1 - h))

            # cal grad
            grad = np.dot(x.T, (h-y))
            total_grad += grad

        total_error /= X.shape[0]
        total_grad /= X.shape[0]

        # backward propagation
        w = w - lr * total_grad

        if prsc_report and (step+1) % precise == 0:
            error_list.append([step, total_error])


    if prsc_report:
        steps = [points[0] for points in error_list]
        errors = [points[1] for points in error_list]
        plt.figure()
        plt.xlabel('step')
        plt.ylabel('loss')
        plt.title('question (4)')
        plt.plot(steps, errors, color ='blue')
        plt.savefig(f'question (4).jpg')
        print("The plot is stored in file question (4).jpg")


    return w, total_error
```

② data definition:

```
a = [4, 5, 12, 29, 30, 36, 36, 54, 58, 70, 72, 76, 78, 82, 87, 90, 90, 92, 95, 98]
b = [49, 4, 28, 18, 65, 32, 1, 29, 76, 12, 26, 55, 4, 15, 95, 70, 55, 84, 14, 21]
label = [0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0]
```

③ training and evaluating:

```
'''
question (2)
'''
print("\n\nQuestion(2)")

X = np.array([[1, a_itm, b_itm] for a_itm, b_itm in zip(a, b)]).reshape((len(a), 3))     # add bias term

Y = np.array(label)


# report
w, total_error = grad_descent(X, Y)

print(f"\nTraining finished.\n Final w = {w}\n Final error {total_error}")

# calculate accuracy
correct_cnt = 0
for x, y in zip(X, Y):
    predicted = logistic_func(np.dot(x, w))
    if np.round(predicted).astype(int) == y:
        correct_cnt += 1
print(f"Accuracy is {correct_cnt / X.shape[0]}")
```

output:

```
Question(2)
100%|██████████| 100000/100000 [00:16<00:00, 6095.43it/s]

Training finished.
 Final w = [ 4.3937461  -0.0392735  -0.06264589]
 Final error 0.3457960740110247
Accuracy is 0.9
```

so the accuracy is 0.9

(3) Try at least 5 different learning rates and report the errors with different learning rates.

try with learning rate: 0.0005, 0.001, 0.0025, 0.005, 0.0075

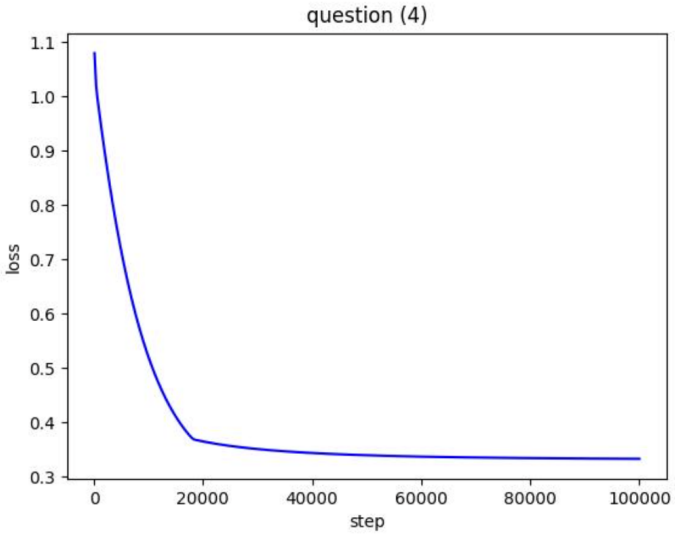
the output is

```
Question(3)
100%|██████████| 100000/100000 [00:16<00:00, 6150.43it/s]

For learning rate = 0.0005:
 Final w = [ 3.25537692 -0.028547   -0.04985973]
 Final error 0.37282798311132825
100%|██████████| 100000/100000 [00:16<00:00, 5950.92it/s]

For learning rate = 0.001:
 Final w = [ 4.3937461  -0.0392735  -0.06264589]
 Final error 0.3457960740110247
100%|██████████| 100000/100000 [00:16<00:00, 5995.59it/s]

For learning rate = 0.0025:
 Final w = [ 5.70927961 -0.05165735 -0.07833275]
 Final error 0.3331829008236791
100%|██████████| 100000/100000 [00:15<00:00, 6279.89it/s]

For learning rate = 0.005:
 Final w = [ 9.14048024 -0.05662365 -0.12062942]
 Final error 0.5270197012266905
100%|██████████| 100000/100000 [00:15<00:00, 6269.07it/s]

For learning rate = 0.0075:
 Final w = [13.38234566 -0.16640852 -0.20966006]
 Final error 0.8109174287252097
```

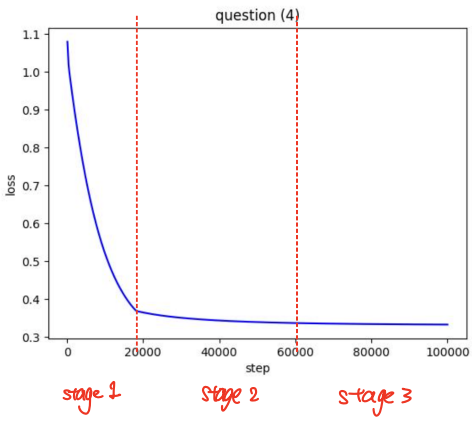
so the error are 0.372, 0.345, 0.333, 0.527, 0.810 respectively

(4) Choose the learning rate with the smallest loss, and then visualize the plot of training errors as the training step increases, where the x-axis and y-axis present the training step and loss, respectively (Please paste the figures to the answer sheet) And estimate how many steps would it take for the model to reach convergence according to the figure.

according to (3). the best learning rate is 0.0025
the plot of trainin loss is:



based on the plot, I think there are 3 stages of loss



in stage 1.(step 0~18,000) the loss is quickly decreasing
in stage 2(step 18,000 ~ 60,000) the loss is slowing decreasing
in stage 3(step 60,000~100,000) the loss remains almost unchanged.

So based on the plot, the estimated steps for model to converge is 60,000