In [1]:
```
import numpy as np
import pandas as pd
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
data = load_iris()
```

In [3]:
```
x = data.data
```

In [4]:
```
y = data.target
```

In [5]:
```
y
```

Out[5]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

In [6]:
```
res=np.zeros((y.size,3),dtype=int)
res[np.arange(y.size),y]=1
res
```

Out[6]:
```
array([[1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
       [1, 0, 0],
```

In [7]:
```
x_train,x_test,y_train,y_test=train_test_split(x,res,test_size=20,random_state
```

In [8]:
```
learning_rate = 0.1
iteration=5000
n=y_train.size
input_size=4
hidden_size = 2
output_size = 3
results = pd.DataFrame(columns=['mse'])
```

In [9]:
```
results
```

Out[9]:

	mse

In [10]:
```
np.random.seed(10)
w1=np.random.normal(scale=0.5,size=(input_size,hidden_size))
```

In [11]:
```
w1
```

Out[11]:
```
array([[ 0.66579325,  0.35763949],
       [-0.77270015, -0.00419192],
       [ 0.31066799, -0.36004278],
       [ 0.13275579,  0.05427426]])
```

In [12]:
```
w2=np.random.normal(scale=0.5,size=(hidden_size,output_size))
w2
```

Out[12]:
```
array([[ 0.00214572, -0.08730011,  0.21651309],
       [ 0.60151869, -0.48253284,  0.51413704]])
```

In [13]:
```
def sigmoid(x):
    return 1/(1+np.exp(-x))
```

In [14]:
```
def mean_squared_error(y_pred,y_True):
    return ((y_pred-y_True)**2).sum()/(2*y_pred.size)
```

In [15]:
```
for itr in range(iteration):
    z1=np.dot(x,w1)
    a1=sigmoid(z1)

    z2=np.dot(a1,w2)
    a2=sigmoid(z2)

    mse=mean_squared_error(a2,res)
    results=results.append({"mse":mse},ignore_index=True)


    #back propogation
    e1=a2-res
    dw1=e1*a2*(1-a2)

    e2=np.dot(dw1,w2.T)
    dw2=e2*a1*(1-a1)

    #Updating the weights
    w2_update=np.dot(a1.T,dw1)/n
    w1_update=np.dot(x.T,dw2)/n

    w2=w2-learning_rate*w2_update
    w1=w1-learning_rate*w1_update
```

In [16]:
```
results
```

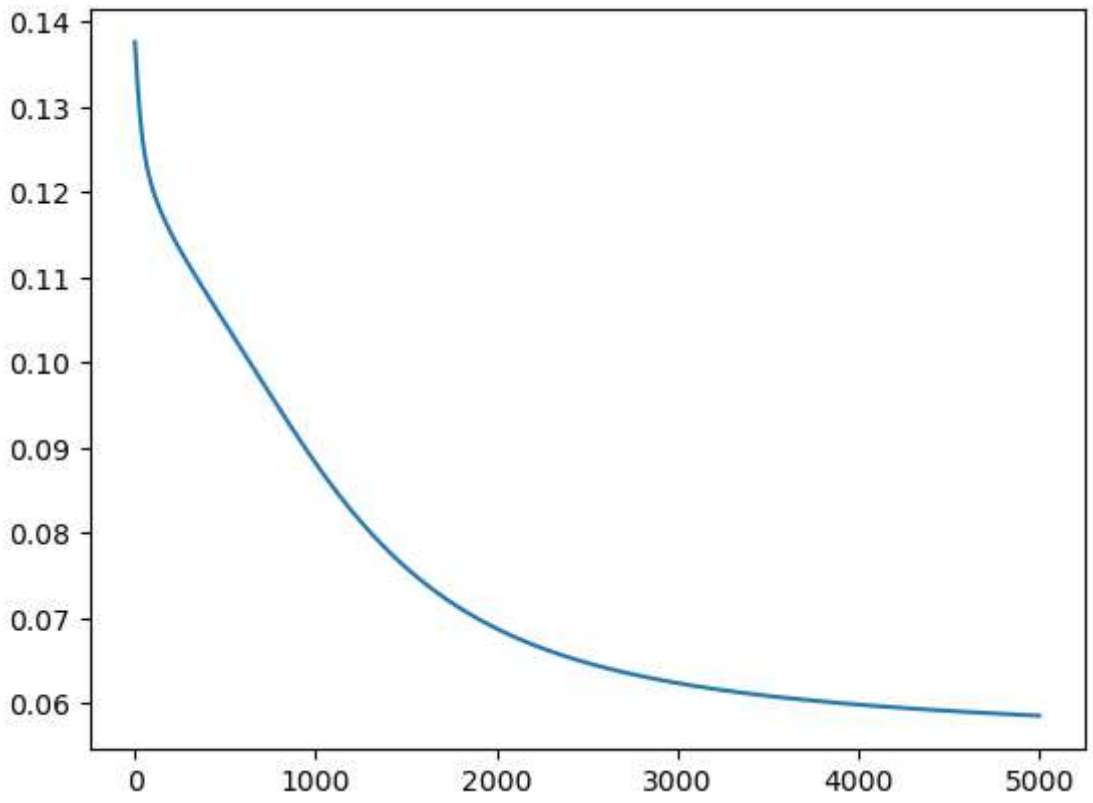
Out[16]:

	mse
0	0.137620
1	0.137258
2	0.136898
3	0.136541
4	0.136186
...	...
4995	0.058558
4996	0.058557
4997	0.058556
4998	0.058555
4999	0.058554

5000 rows × 1 columns

In [17]:
```
results.mse.plot()
```

Out[17]: <AxesSubplot:>



In [ ]:
```

```