دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

دانشکده مهندسی برق

گزارشکار اول آزمایشگاه مقدمه ای بر هوش محاسباتی

نگارش
مهدیه سادات بنیس

استاد درس
دکتر فرزانه عبدالهی

استاد آزمایشگاه
الهه السادات احمدی موسوی

نیم سال اول ۱۴۰۱

فهرست مطالب

عنوان **صفحه**

فصل اول
آزمایش ۱

۱-۱- شرح آزمایش

در این قسمت به پیاده سازی یک تابع ساده در محیط متلب و پایتون پرداخته می شود. تابع یک بار در محیط پایتون و یک بار در محیط متلب باید پیاده سازی شود.

تابع sigmoid و مشتق آن از پرکاربرد ترین توابع در شبکه عصبی هستند. تابع sigmoid و مشتق آن توسط معادلات زیر توصیف شده اند:

$$f(x) = \frac{2}{1 + exp(-x)} + 1$$

$$\dot{f}(x) = \frac{2exp(-x)}{(1 + exp(-x))^2} = \frac{1 - (f(x))^2}{2}$$

تابعی به اسم sigmatrix بنویسید که یک ماتریس و ابعاد آن را از ورودی دریافت کند و دارای۲ خروجی به صورت زیر باشد(A_{ij} اعضای ماتریس ورودی را نشان می دهد و $\dot{sigmoid}$ مشتق تابع sigmoid است. n و m نیز به ترتیب برابر با تعداد سطرها و ستون های ماتریس ورودی است) :

خروجی اول: $\sum_{i=1}^{n}\sum_{j=1}^{m} sigmoid(A_{ij})$

خروجی دوم: $\sum_{i=1}^{n}\sum_{j=1}^{m} \dot{sigmoid}(A_{ij})$

برای پیاده سازی تابع خواسته شده مراحل زیر را دنبال کنید:

۱. یک تابع بنویسید که یک عدد را از ورودی دریافت کند و مقدار تابع sigmoid را در آن نقطه باز گرداند.

۲. تابعی بنویسید که یک عدد را از ورودی دریافت کند و مقدار مشتق تابع sigmoid در آن نقطه را بازگرداند.

۳. یک تابع به اسم sigmatrix بنویسید که ماتریس و ابعاد آن را از ورودی دریافت می کند و برای تمام اعضای ماتریس

توابع نوشته شده در قسمت ۱و ۲ را صدا می زند (توسط یک حلقه for)سپس توسط عمل جمع خروجی های ۱و۲ خواسته شده را تولید کنید.

بعد از پیاده سازی تابع ماتریس M را که توسط معادلات زیر توصیف شده است به همراه تعداد سطرها و ستون های آن به تابع نوشته شده بدهید و خروجی را مشاهده کنید.

$$M = \begin{bmatrix} 1 & 0 & sin(pi/4) \\ 0 & 1 & sin(pi/2) \\ 1 & 0 & 1 \end{bmatrix}$$

۱-۲- توضیح کد به زبان پایتون

ابتدا کتابخانه های مورد نظر را اضافه میکنیم.(numpy برای استفاده از آرایه ها و math برای عبارت های ریاضی)

Experiment 1: Sigmoid Function

Importing Libraries

```
import numpy as np
import math
```

باتوجه به شرح آزمایش و فرمول ذکر شده تابع sigmoid را تعریف میکنیم:

Sigmoid Function

```
def sigmoid(x):

    f_x = 2 / (1 + math.exp(-x)) - 1
    return f_x
```

باتوجه به شرح آزمایش و فرمول ذکر شده مشتق تابع sigmoid را تعریف میکنیم:

Derivative of the Sigmoid function

```
def diff_sigmoid(x):

    df_x = 2 * math.exp(-x) / ((1 + math.exp(-x)) ** 2)
    return df_x
```

تابعsigmatrix را به این صورت تعریف میکنیم که در ورودی ابعاد ماتریس و ماتریس را دریافت میکند

out1 , out2 خروجی های تابع می باشند که با توجه به شرح آزمایش آن را از از روی ماتریس ورودی بدست می آوریم(در ابتدا هر دو مقدار ۰ را دارند)

Sigmatrix

```
def sigmatrix(n, m, A):

    out1 = 0
    out2 = 0

    for i in range(n):
        for j in range(m):
            out1 += sigmoid(A[i][j])
            out2 += diff_sigmoid(A[i][j])
    return out1, out2
```

در قسمت ورودی ابعاد ماتریس و ماتریس را از ورودی دریافت میکنیم(ماتریس را سطر به سطر دریافت میکنیم به طوری که المان ها با فاصله جدا شدند)

Input

```
R = int(input("Please enter number of rows: "))
C = int(input("Please enter number of columns: "))
```

```
Please enter number of rows: 3
Please enter number of columns: 3
```

```
matrix = []
for i in range(R):
    a = list(map(int, input(f"Enter {i + 1}th row: ").split()))
    matrix.append(a)
```

```
Enter 1th row: 0 0 0
Enter 2th row: 1 1 1
Enter 3th row: 2 2 2
```

در آخر خروجی تابع را به ازای ورودی داده شده و ماتریس M بدست می آوریم:

Output

```
print("First Output & Second Output: ", sigmatrix(R, C, matrix))
```

```
First Output & Second Output:  (3.671133939647323, 3.30963311186993)
```

```
M = np.array([[1, 0, math.sin(math.pi / 4)],
     [0, 1, math.sin(math.pi / 2)],
     [1, 0, 1]])
M
```

```
array([[1.        , 0.        , 0.70710678],
       [0.        , 1.        , 1.        ],
       [1.        , 0.        , 1.        ]])
```

```
sigmatrix(3, 3, M)
```

```
(2.6501088849533625, 3.9084813651552452)
```

۱-۳- توضیح کد به زبان متلب

در قسمت ورودی ابعاد ماتریس و ماتریس را از ورودی دریافت میکنیم(ماتریس را سطر به سطر دریافت میکنیم به طوری که المان ها با فاصله جدا شدند)

```
clc
clear all
close all
warning off
```

getting input

```
%--------------------------------------------------------------------
R = input("Please enter number of rows: ");
C = input("Please enter number of columns: ");
matrix = input("Please enter matrix: ");
```

در خروجی تابع را به ازای ورودی داده شده و ماتریس M بدست می آوریم:

print output of sigmatrix

```
%--------------------------------------------------------------------
[out1, out2] = sigmatrix(R, C, matrix);
fprintf('out1: %d \nout2: %d\n\n', out1, out2);

M = [1 0 sin(pi / 4); 0 1 sin(pi / 2); 1 0 1];
[out1, out2] = sigmatrix(3, 3, M);
disp(M);
fprintf('out1: %d \nout2: %d\n', out1, out2);
```

باتوجه به شرح آزمایش و فرمول ذکر شده تابع sigmoid را تعریف میکنیم:

sigmoid function

```
%--------------------------------------------------------------------
function [f_x] = sigmoid(x)
    f_x = 2 / (1 + exp(-x)) - 1;
end
```

باتوجه به شرح آزمایش و فرمول ذکر شده مشتق تابع sigmoid را تعریف میکنیم:

Derivative of the Sigmoid function

```
%--------------------------------------------------------------------
function [df_x] = diffsigmoid(x)
    df_x = (2 * exp(-x)) / ((1 + exp(-x))^2);
end
```

تابعsigmatrix را به این صورت تعریف میکنیم که در ورودی ابعاد ماتریس و ماتریس را دریافت میکند

out1 , out2 خروجی های تابع می باشند که با توجه به شرح آزمایش آن را از از روی ماتریس ورودی بدست می آوریم(در ابتدا هر دو مقدار ۰ را دارند)

Sigmatrix function

```
%--------------------------------------------------------------------
function [out1, out2] = sigmatrix(n, m, A)
    out1 = 0;
    out2 = 0;
    for i = 1:1:n
        for j = 1:1:m
             out1 = out1 + sigmoid(A(i, j));
             out2 = out2 + diffsigmoid(A(i, j));
        end
    end
end
```

Command Window

```
Please enter number of rows: 3
Please enter number of columns: 3
Please enter matrix: [0 0 0; 1 1 1; 2 2 2]
out1: 3.671134e+00
out2: 3.309633e+00

    1.0000         0    0.7071
         0    1.0000    1.0000
    1.0000         0    1.0000

out1: 2.650109e+00
out2: 3.908481e+00
```

فصل دوم
تمرین

۲-۱- شرح تمرین

تابعی بنویسید که یک ماتریس و ابعاد آن را از ورودی دریافت کند و $\sum_{i=1}^{n}\sum_{j=1}^{m} tanh(A_{ij})$ را باز گرداند

(A نمایش دهنده ماتریس ورودی تابع و n و m تعداد سطرها و ستون های آن هستند.) تابع $tanh(x)$ به صورت زیر تعریف می شود.

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

۲-۲- توضیح کد به زبان پایتون

ابتدا کتابخانه های مورد نظر را اضافه میکنیم.(numpy برای استفاده از آرایه ها و math برای عبارت های ریاضی)

Experiment 1:

practice: tanh Function

Importing Libraries

```
import numpy as np
import math
```

باتوجه به شرح تمرین و فرمول ذکر شده تابع tanh را تعریف میکنیم:

tanh Function

```
def tanh(x):

    f_x = (math.exp(x) - math.exp(-x)) / (math.exp(x) + math.exp(-x))
    return f_x
```

تابعtanhmatrix را به این صورت تعریف میکنیم که در ورودی ابعاد ماتریس و ماتریس را دریافت میکند

out خروجی تابع می باشد که با توجه به شرح آزمایش آن را از از روی ماتریس ورودی بدست می آوریم(در ابتدا هر آن مقدار ۰ را دارند)

tanhmatrix

```
def tanhmatrix(n, m, A):

    out = 0

    for i in range(n):
        for j in range(m):
            out += math.tanh(A[i][j])
    return out
```

در قسمت ورودی ابعاد ماتریس و ماتریس را از ورودی دریافت میکنیم(ماتریس را سطر به سطر دریافت میکنیم به طوری که المان ها با فاصله جدا شدند)

Input

```
R = int(input("Please enter number of rows: "))
C = int(input("Please enter number of columns: "))
```

```
Please enter number of rows: 3
Please enter number of columns: 3
```

```
matrix = []
for i in range(R):
    a = list(map(int, input(f"Enter {i + 1}th row: ").split()))
    matrix.append(a)
```

```
Enter 1th row: 0 0 0
Enter 2th row: 1 1 1
Enter 3th row: 2 2 2
```

در آخر خروجی تابع را به ازای ورودی داده شده و ماتریس M بدست می آوریم:

Output

```
print("First Output: ", tanhmatrix(R, C, matrix))
```

```
First Output:  5.176865208094744
```

```
M = np.array([[1, 0, math.sin(math.pi / 4)],
     [0, 1, math.sin(math.pi / 2)],
     [1, 0, 1]])
M
```

```
array([[1.        , 0.        , 0.70710678],
       [0.        , 1.        , 1.        ],
       [1.        , 0.        , 1.        ]])
```

```
tanhmatrix(3, 3, M)
```

```
4.416830144792739
```

۲-۳- توضیح کد به زبان متلب

در قسمت ورودی ابعاد ماتریس و ماتریس را از ورودی دریافت میکنیم(ماتریس را سطر به سطر دریافت میکنیم به طوری که المان ها با فاصله جدا شدند)

```
clc
clear all
close all
warning off
```

getting input

```
%--------------------------------------------------------------------------
R = input("Please enter number of rows: ");
C = input("Please enter number of columns: ");
matrix = input("Please enter matrix: ");
```

در خروجی تابع را به ازای ورودی داده شده و ماتریس M بدست می آوریم:

print output of sigmatrix

```
%--------------------------------------------------------------------------
[out] = tanhmatrix(R, C, matrix);
fprintf('out: %d\n\n', out);

M = [1 0 sin(pi / 4); 0 1 sin(pi / 2); 1 0 1];
[out] = tanhmatrix(3, 3, M);
disp(M);
fprintf('out: %d\n', out);
```

باتوجه به شرح تمرین و فرمول ذکر شده تابع tanh را تعریف میکنیم:

tanh function

```
%--------------------------------------------------------------------------
function [f_x] = tanh(x)
    f_x = (exp(x) - exp(-x)) / (exp(x) + exp(-x));
end
```

تابعtanhmatrix را به این صورت تعریف میکنیم که در ورودی ابعاد ماتریس و ماتریس را دریافت میکند

out خروجی تابع می باشد که با توجه به شرح آزمایش آن را از از روی ماتریس ورودی بدست می آوریم(در ابتدا هر آن مقدار ۰ را دارند)

tanhmatrix function

```
%--------------------------------------------------------------------
function [out] = tanhmatrix(n, m, A)
    out = 0;
    for i = 1:1:n
        for j = 1:1:m
             out = out + tanh(A(i, j));
        end
    end
end
```

Command Window

```
Please enter number of rows: 3
Please enter number of columns: 3
Please enter matrix: [0 0 0; 1 1 1; 2 2 2]
out: 5.176865e+00

    1.0000         0    0.7071
         0    1.0000    1.0000
    1.0000         0    1.0000

out: 4.416830e+00
```

فصل سوم
تمرین امتیازی

چرا پایتون یک زبان شی گرا است؟

«برنامه‌نویسی شی‌گرا»[۱] یا به اختصار OOP یک الگو یا شیوه تفکر در برنامه‌نویسی است که برگرفته از دنیای واقعی بوده و از دهه ۱۹۶۰ میلادی مطرح گشته است. به زبانی که از این الگو پشتیبانی کند، «زبان شی‌گرا» گفته می‌شود؛ ایده شی‌گرایی در پاسخ به برخی از نیازها که الگوهای موجود پاسخ‌گو آن‌ها نبودند به وجود آمد؛ نیازهایی مانند: توانایی حل تمامی مسائل پیچیده[۲]، «پنهان‌سازی داده»[۳]«قابلیت استفاده مجدد[۴]» بیشتر، وابستگی کمتر به توابع، انعطاف بالا و...

رویکرد برنامه‌نویسی شی‌گرا «از پایین به بالا»[۵] است؛ یعنی ابتدا واحدهایی کوچک از برنامه ایجاد می‌شوند و سپس با پیوند این واحدها، واحدهایی بزرگتر و در نهایت شکلی کامل از برنامه به وجود می‌آید. برنامه‌نویسی شی‌گرا در قالب دو مفهوم «کلاس»[۶] و «شی»[۷] ارایه می‌گردد. هر کلاس واحدی از برنامه است که تعدادی داده و عملیات را در خود نگه‌داری می‌کند و هر شی نیز حالتی[۸] مشخص از یک کلاس می‌باشد.

برنامه‌نویسی شی گرایی شیوه‌ای از برنامه‌نویسی و مدل کردن است که توابع و متغیرهای مختلف کلاس‌ها را می‌سازند و کلاس‌ها هم اشیا را تشکیل می‌دهند. داده‌ها و پردازش آن‌ها که در قالب اشیا بیان می‌شوند امکان دسترسی و کنترل بهتری را برای برنامه‌نویسان فراهم می‌کنند تا از پیچیدگی‌های ایجادشده کم کنند.

پایتون هم از زبان برنامه نویسی شی گرا و هم از زبان برنامه نویسی رویه ای پشتیبانی می کند زیرا یک زبان برنامه نویسی سطح بالایی است که برای برنامه نویسی با هدف عمومی طراحی شده است . ... مفاهیم OOP مانند کلاس ها، کپسولاسیون، چندشکلی، ارث بردن و غیره در پایتون آن را به عنوان یک زبان برنامه نویسی شی گرا تبدیل می کند.

مانند سایر زبان های برنامه نویسی همه منظوره، پایتون نیز از همان ابتدا یک زبان شی گرا است. ... در پایتون به راحتی می توانیم کلاس ها و آبجکت ها را ایجاد و استفاده کنیم. یک پارادایم شی گرا طراحی برنامه با استفاده از کلاس ها و اشیاء است.

---

۱ Object-Oriented Programming

۲ Complex

۳ Data Hiding

۴ Reusability

۵ Bottom-Up

۶ Class

۷ Object

۸ State