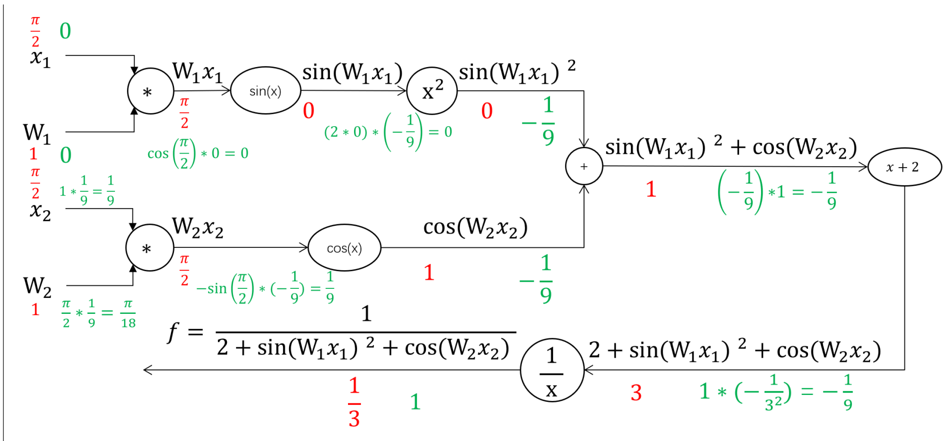
Homework Assignment 2

Name: Linqi Xiao NetID: lx130

Problem 1 (Practice of scalar-based backpropagation)

(a)



$\frac{\pi}{2}$ 0
x_1
W_1
1 0
$\frac{\pi}{2}$ $1*\frac{1}{9}=\frac{1}{9}$
x_2
W_2
1 $\frac{\pi}{2}*\frac{1}{9}=\frac{\pi}{18}$
*
W_1x_1
$\frac{\pi}{2}$
$\cos\left(\frac{\pi}{2}\right)*0=0$
sin(x)
$\sin(W_1x_1)$
0
$(2*0)*\left(-\frac{1}{9}\right)=0$
x^2
$\sin(W_1x_1)^2$
0 $-\frac{1}{9}$
+
$\sin(W_1x_1)^2+\cos(W_2x_2)$
1
$\left(-\frac{1}{9}\right)*1=-\frac{1}{9}$
$x+2$
*
W_2x_2
$\frac{\pi}{2}$
$-\sin\left(\frac{\pi}{2}\right)*(-\frac{1}{9})=\frac{1}{9}$
cos(x)
$\cos(W_2x_2)$
1 $-\frac{1}{9}$
$f=\frac{1}{2+\sin(W_1x_1)^2+\cos(W_2x_2)}$
$\frac{1}{3}$ 1
$\frac{1}{x}$
$2+\sin(W_1x_1)^2+\cos(W_2x_2)$
3 $1*(-\frac{1}{3^2})=-\frac{1}{9}$

Forward
Backward

$f(x)=\frac{1}{x}$ $\frac{df}{dx}=-\frac{1}{x^2}$

$f(x)=2+x$ $\frac{df}{dx}=1$

$f(x,y)=x+y$ $\frac{df}{dx}=1$ $\frac{df}{dy}=1$

$f(x)=x^2$ $\frac{df}{dx}=2x$

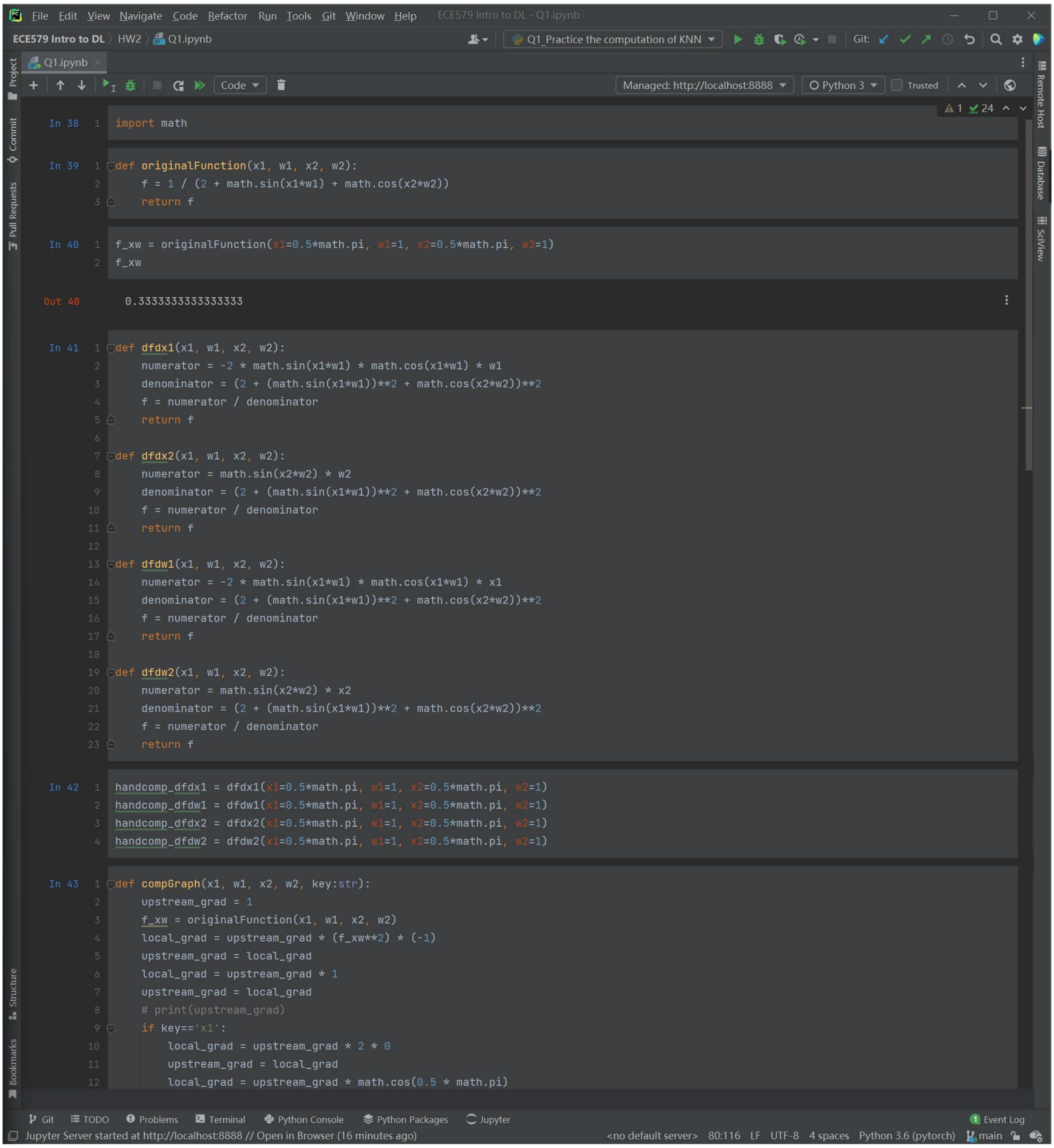
$f(x)=\sin(x)$ $\frac{df}{dx}=\cos(x)$

$f(x)=\cos(x)$ $\frac{df}{dx}=-\sin(x)$

$f(x)=W_1x_1$ $\frac{df}{dx}=W_1$

$f(x)=W_2x_2$ $\frac{df}{dx}=W_2$

(b)



File Edit View Navigate Code Refactor Run Tools Git Window Help
ECE579 Intro to DL - Q1.ipynb
ECE579 Intro to DL 〉 HW2 〉 Q1.ipynb
Q1_Practice the computation of KNN
Git:
Q1.ipynb
Code
Managed: http://localhost:8888
Python 3
Trusted
Project
Commit
Pull Requests
Structure
Bookmarks
Remote Host
Database
SciView
Git
TODO
Problems
Terminal
Python Console
Python Packages
Jupyter
Event Log

In 38

```python
import math
```

In 39

```python
def originalFunction(x1, w1, x2, w2):
    f = 1 / (2 + math.sin(x1*w1) + math.cos(x2*w2))
    return f
```

In 40

```python
f_xw = originalFunction(x1=0.5*math.pi, w1=1, x2=0.5*math.pi, w2=1)
f_xw
```

Out 40

```
0.3333333333333333
```

In 41

```python
def dfdx1(x1, w1, x2, w2):
    numerator = -2 * math.sin(x1*w1) * math.cos(x1*w1) * w1
    denominator = (2 + (math.sin(x1*w1))**2 + math.cos(x2*w2))**2
    f = numerator / denominator
    return f

def dfdx2(x1, w1, x2, w2):
    numerator = math.sin(x2*w2) * w2
    denominator = (2 + (math.sin(x1*w1))**2 + math.cos(x2*w2))**2
    f = numerator / denominator
    return f

def dfdw1(x1, w1, x2, w2):
    numerator = -2 * math.sin(x1*w1) * math.cos(x1*w1) * x1
    denominator = (2 + (math.sin(x1*w1))**2 + math.cos(x2*w2))**2
    f = numerator / denominator
    return f

def dfdw2(x1, w1, x2, w2):
    numerator = math.sin(x2*w2) * x2
    denominator = (2 + (math.sin(x1*w1))**2 + math.cos(x2*w2))**2
    f = numerator / denominator
    return f
```

In 42

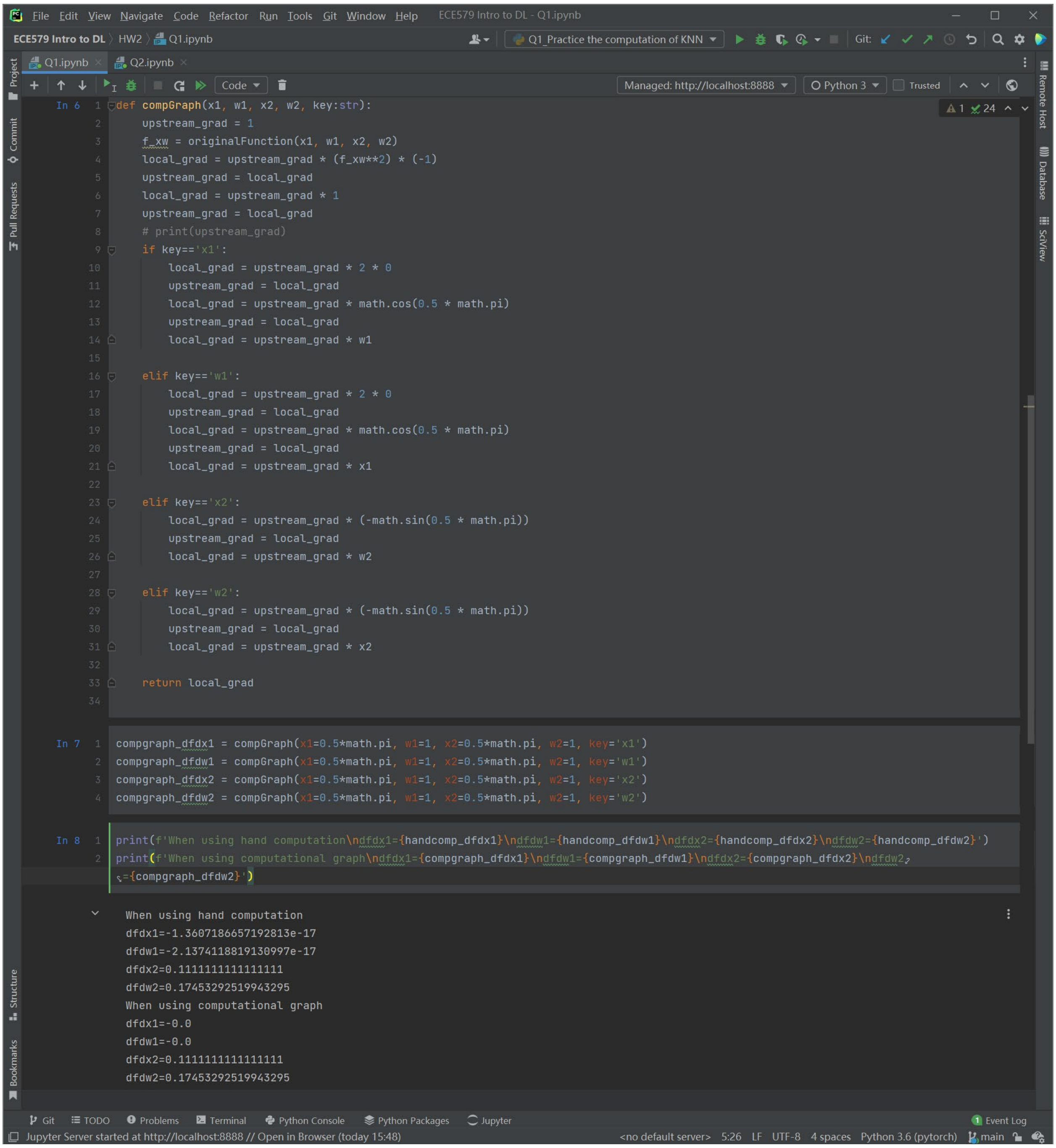
```python
handcomp_dfdx1 = dfdx1(x1=0.5*math.pi, w1=1, x2=0.5*math.pi, w2=1)
handcomp_dfdw1 = dfdw1(x1=0.5*math.pi, w1=1, x2=0.5*math.pi, w2=1)
handcomp_dfdx2 = dfdx2(x1=0.5*math.pi, w1=1, x2=0.5*math.pi, w2=1)
handcomp_dfdw2 = dfdw2(x1=0.5*math.pi, w1=1, x2=0.5*math.pi, w2=1)
```

In 43

```python
def compGraph(x1, w1, x2, w2, key:str):
    upstream_grad = 1
    f_xw = originalFunction(x1, w1, x2, w2)
    local_grad = upstream_grad * (f_xw**2) * (-1)
    upstream_grad = local_grad
    local_grad = upstream_grad * 1
    upstream_grad = local_grad
    # print(upstream_grad)
    if key=='x1':
        local_grad = upstream_grad * 2 * 0
        upstream_grad = local_grad
        local_grad = upstream_grad * math.cos(0.5 * math.pi)
```

Jupyter Server started at http://localhost:8888 // Open in Browser (16 minutes ago)

<no default server> 80:116 LF UTF-8 4 spaces Python 3.6 (pytorch) main


ECE579 Intro to DL - Q1.ipynb
ECE579 Intro to DL 〉 HW2 〉 Q1.ipynb
Q1_Practice the computation of KNN
Q1.ipynb
Q2.ipynb
Code
Managed: http://localhost:8888
Python 3
Trusted
Git
TODO
Problems
Terminal
Python Console
Python Packages
Jupyter
Event Log
Jupyter Server started at http://localhost:8888 // Open in Browser (today 15:48)
<no default server> 5:26 LF UTF-8 4 spaces Python 3.6 (pytorch) main

In 6

```python
def compGraph(x1, w1, x2, w2, key:str):
    upstream_grad = 1
    f_xw = originalFunction(x1, w1, x2, w2)
    local_grad = upstream_grad * (f_xw**2) * (-1)
    upstream_grad = local_grad
    local_grad = upstream_grad * 1
    upstream_grad = local_grad
    # print(upstream_grad)
    if key=='x1':
        local_grad = upstream_grad * 2 * 0
        upstream_grad = local_grad
        local_grad = upstream_grad * math.cos(0.5 * math.pi)
        upstream_grad = local_grad
        local_grad = upstream_grad * w1

    elif key=='w1':
        local_grad = upstream_grad * 2 * 0
        upstream_grad = local_grad
        local_grad = upstream_grad * math.cos(0.5 * math.pi)
        upstream_grad = local_grad
        local_grad = upstream_grad * x1

    elif key=='x2':
        local_grad = upstream_grad * (-math.sin(0.5 * math.pi))
        upstream_grad = local_grad
        local_grad = upstream_grad * w2

    elif key=='w2':
        local_grad = upstream_grad * (-math.sin(0.5 * math.pi))
        upstream_grad = local_grad
        local_grad = upstream_grad * x2

    return local_grad

```

In 7

```python
compgraph_dfdx1 = compGraph(x1=0.5*math.pi, w1=1, x2=0.5*math.pi, w2=1, key='x1')
compgraph_dfdw1 = compGraph(x1=0.5*math.pi, w1=1, x2=0.5*math.pi, w2=1, key='w1')
compgraph_dfdx2 = compGraph(x1=0.5*math.pi, w1=1, x2=0.5*math.pi, w2=1, key='x2')
compgraph_dfdw2 = compGraph(x1=0.5*math.pi, w1=1, x2=0.5*math.pi, w2=1, key='w2')
```

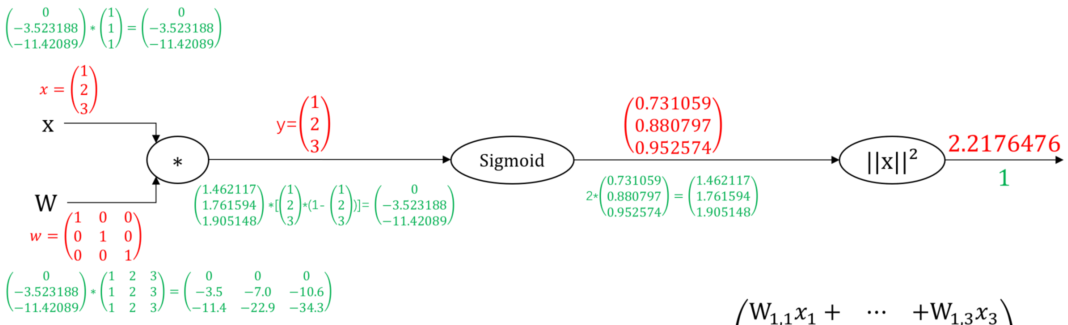
In 8

```python
print(f'When using hand computation\ndfdx1={handcomp_dfdx1}\ndfdw1={handcomp_dfdw1}\ndfdx2={handcomp_dfdx2}\ndfdw2={handcomp_dfdw2}')
print(f'When using computational graph\ndfdx1={compgraph_dfdx1}\ndfdw1={compgraph_dfdw1}\ndfdx2={compgraph_dfdx2}\ndfdw2={compgraph_dfdw2}')
```

```
When using hand computation
dfdx1=-1.3607186657192813e-17
dfdw1=-2.1374118819130997e-17
dfdx2=0.1111111111111111
dfdw2=0.17453292519943295
When using computational graph
dfdx1=-0.0
dfdw1=-0.0
dfdx2=0.1111111111111111
dfdw2=0.17453292519943295
```

Problem 2 (Practice of vector-based backpropagation)

(a)



$\begin{pmatrix}0\\-3.523188\\-11.42089\end{pmatrix}*\begin{pmatrix}1\\1\\1\end{pmatrix}=\begin{pmatrix}0\\-3.523188\\-11.42089\end{pmatrix}$
$x=\begin{pmatrix}1\\2\\3\end{pmatrix}$
x
$y=\begin{pmatrix}1\\2\\3\end{pmatrix}$
*
Sigmoid
$\begin{pmatrix}0.731059\\0.880797\\0.952574\end{pmatrix}$
||x||^2
2.2176476
1
W
$\begin{pmatrix}1.462117\\1.761594\\1.905148\end{pmatrix}*[\begin{pmatrix}1\\2\\3\end{pmatrix}*(1-\begin{pmatrix}1\\2\\3\end{pmatrix})]=\begin{pmatrix}0\\-3.523188\\-11.42089\end{pmatrix}$
$2*\begin{pmatrix}0.731059\\0.880797\\0.952574\end{pmatrix}=\begin{pmatrix}1.462117\\1.761594\\1.905148\end{pmatrix}$
$w=\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}$
$\begin{pmatrix}0\\-3.523188\\-11.42089\end{pmatrix}*\begin{pmatrix}1&2&3\\1&2&3\\1&2&3\end{pmatrix}=\begin{pmatrix}0&0&0\\-3.5&-7.0&-10.6\\-11.4&-22.9&-34.3\end{pmatrix}$

$$W\cdot\mathrm{x}=\begin{pmatrix}\mathrm{W}_{1,1}x_1+&\cdots&+\mathrm{W}_{1,3}x_3\\\vdots&\vdots&\vdots\\\mathrm{W}_{3,1}x_1+&\cdots&+\mathrm{W}_{3,3}x_3\end{pmatrix}$$

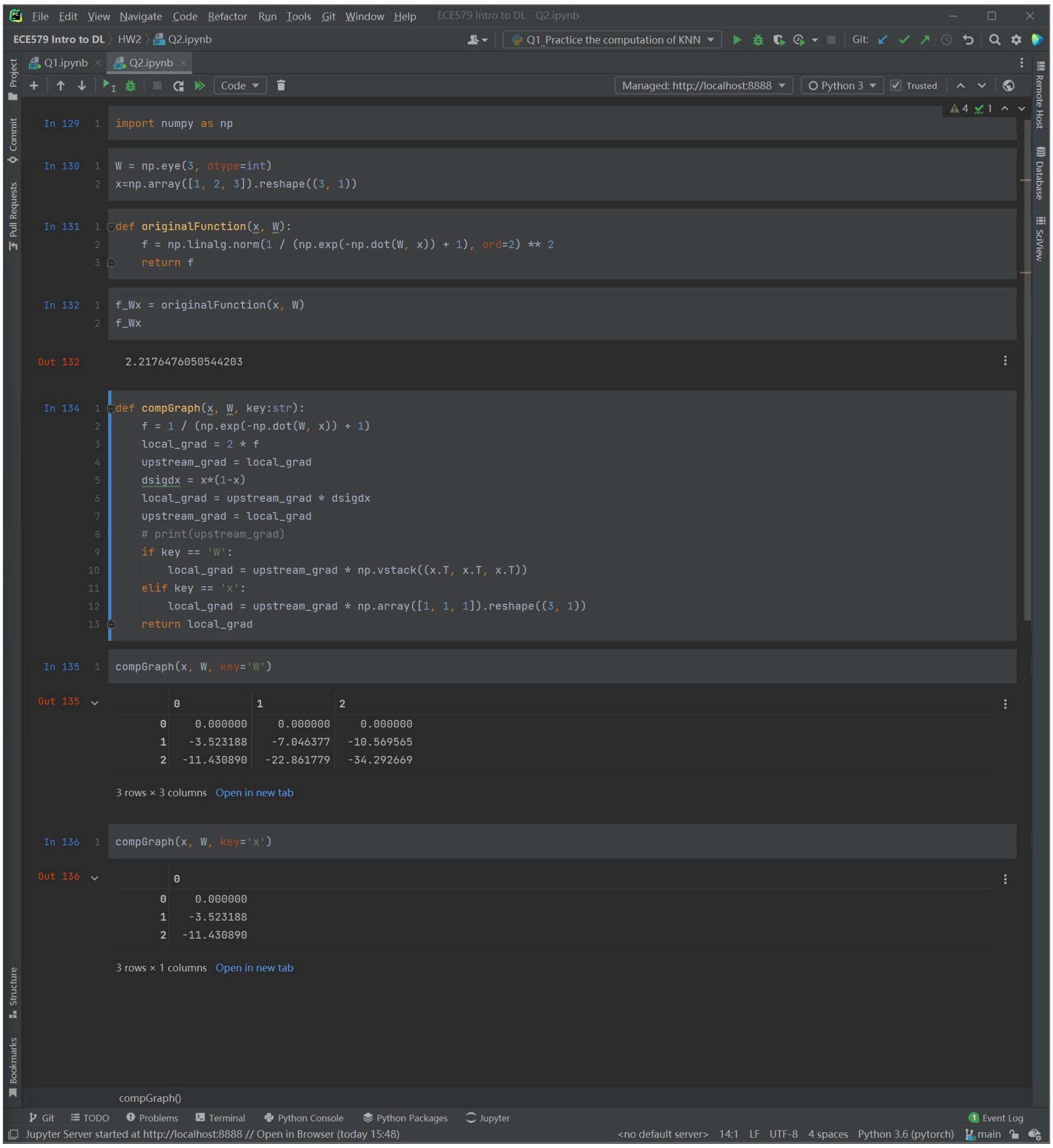
$$\frac{dL}{dx_{1,1}}=\frac{dy}{dx_{1,1}}*\frac{dL}{dy}=\begin{pmatrix}1\\0\\0\end{pmatrix}*\begin{pmatrix}0\\-3.523188\\-11.42089\end{pmatrix}=0$$

$$\frac{dL}{dx_{2,1}}=\frac{dy}{dx_{2,1}}*\frac{dL}{dy}=\begin{pmatrix}0\\1\\0\end{pmatrix}*\begin{pmatrix}0\\-3.523188\\-11.42089\end{pmatrix}=-3.523188$$

$$\frac{dL}{dx_{3,1}}=\frac{dy}{dx_{3,1}}*\frac{dL}{dy}=\begin{pmatrix}0\\0\\1\end{pmatrix}*\begin{pmatrix}0\\-3.523188\\-11.42089\end{pmatrix}=-11.42089$$

f(x)= Sig(x)=h $\frac{df}{dx}=h(1-h)$

f(x)=||x||^2=$\mathrm{x}_{1,1}^2+\mathrm{x}_{2,1}^2+\mathrm{x}_{3,1}^2$ $\frac{df}{d\mathrm{x}_{i,1}}=2x_{i,1}$


File Edit View Navigate Code Refactor Run Tools Git Window Help
ECE579 Intro to DL - Q2.ipynb
ECE579 Intro to DL 〉 HW2 〉 Q2.ipynb
Q1_Practice the computation of KNN
Git:
Q1.ipynb
Q2.ipynb
Code
Managed: http://localhost:8888
Python 3
☑ Trusted
Project
Commit
Pull Requests
Structure
Bookmarks
Remote Host
Database
SciView
compGraph()
Git
TODO
Problems
Terminal
Python Console
Python Packages
Jupyter
Event Log
<no default server> 14:1 LF UTF-8 4 spaces Python 3.6 (pytorch) main

In 129
```python
import numpy as np
```

In 130
```python
W = np.eye(3, dtype=int)
x=np.array([1, 2, 3]).reshape((3, 1))
```

In 131
```python
def originalFunction(x, W):
    f = np.linalg.norm(1 / (np.exp(-np.dot(W, x)) + 1), ord=2) ** 2
    return f
```

In 132
```python
f_Wx = originalFunction(x, W)
f_Wx
```

Out 132
```
2.2176476050544203
```

In 134
```python
def compGraph(x, W, key:str):
    f = 1 / (np.exp(-np.dot(W, x)) + 1)
    local_grad = 2 * f
    upstream_grad = local_grad
    dsigdx = x*(1-x)
    local_grad = upstream_grad * dsigdx
    upstream_grad = local_grad
    # print(upstream_grad)
    if key == 'W':
        local_grad = upstream_grad * np.vstack((x.T, x.T, x.T))
    elif key == 'x':
        local_grad = upstream_grad * np.array([1, 1, 1]).reshape((3, 1))
    return local_grad
```

In 135
```python
compGraph(x, W, key='W')
```

Out 135

	0	1	2
0	0.000000	0.000000	0.000000
1	-3.523188	-7.046377	-10.569565
2	-11.430890	-22.861779	-34.292669

3 rows × 3 columns Open in new tab

In 136
```python
compGraph(x, W, key='x')
```

Out 136

	0
0	0.000000
1	-3.523188
2	-11.430890

3 rows × 1 columns Open in new tab

Jupyter Server started at http://localhost:8888 // Open in Browser (today 15:48)

The answer is as same as I calculate by my hand.