TSM DEEP LEARNING

Practical work 02

Gregory Banfi
Benjamin Kühnis
October 10, 2018

EXERCICE 2 – SIGMOID FUNCTION

(a) Compute the derivative of the sigmoid function

$$
\begin{aligned}
\frac{d}{dz}\left[\frac{1}{1+e^{-z}}\right] &= -\frac{\frac{d}{dz}\left[1+e^{-z}\right]}{(1+e^{-z})^2} && \textit{Reciprocal rule} \\
&= -\frac{\frac{d}{dz}[1]+\frac{d}{dz}\left[e^{-z}\right]}{(1+e^{-z})^2} && \textit{Sum rule} \\
&= -\frac{0+e^{-z}\cdot\frac{d}{dz}[-z]}{(1+e^{-z})^2} && \textit{Chain, Exponential rule} \\
&= -\frac{e^{-z}\cdot -1}{(1+e^{-z})^2} \\
&= \frac{e^{-z}}{(1+e^{-z})^2}
\end{aligned} \tag{0.1}
$$

(b) Show that the derivative fulfills the equation

$$\begin{aligned}\sigma(z)' &= \sigma(z)\cdot(1-\sigma(z))\\&= \frac{1}{1+e^{-z}}\cdot 1-\frac{1}{1+e^{-z}}\\&= \frac{1}{1+e^{-z}}\cdot\frac{1+e^{-z}-1}{1+e^{-z}}\\&= \frac{1}{1+e^{-z}}\cdot\frac{e^{-z}}{1+e^{-z}}\\&= \frac{e^{-z}}{(1+e^{-z})^2}\end{aligned} \tag{0.2}$$

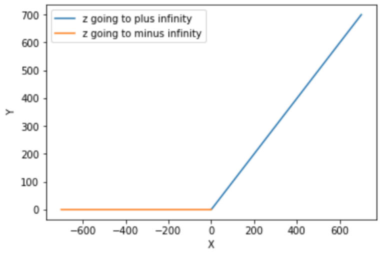
(c) Compute the first and second derivative
First Derivative

$$\begin{aligned}\zeta(z)' &= \frac{d}{dz}[-\ln(\sigma(-z))]\\&= \frac{d}{dz}[-\ln(\frac{1}{1+e^x})] && \textit{Log rule to simplify}\\&= \frac{d}{dz}[\ln(e^x+1)] && \textit{Chain, Log rule}\\&= \frac{1}{e^x+1}\cdot\frac{d}{dz}[e^x+1] && \textit{Sum rule}\\&= \frac{1}{e^x+1}\cdot\frac{d}{dz}[e^x]+\frac{d}{dz}[1] && \textit{Chain, Exponential rule}\\&= \frac{1}{e^x+1}\cdot e^x+0\\&= \frac{e^x}{e^x+1}\end{aligned} \tag{0.3}$$

Second Derivative

$$\begin{aligned}\frac{d}{dz}[\zeta(z)'] &= \frac{d}{dz}\left[\frac{e^x}{e^x+1}\right] && \textit{Quotient rule}\\&= \frac{\frac{d}{dz}[e^x]\cdot(e^x+1)-e^x\cdot(\frac{d}{dz}[e^x+1])}{(e^x+1)^2} && \textit{Sum, Exponential rule}\\&= \frac{e^x\cdot(e^x+1)-e^x\cdot(\frac{d}{dz}[e^x]+\frac{d}{dz}[1])}{(e^x+1)^2} && \textit{Constant, Exponential rule}\\&= \frac{e^x\cdot(e^x+1)-e^x\cdot(e^x+0)}{(e^x+1)^2}\\&= \frac{e^{2x}+e^x-e^{2x}}{(e^x+1)^2}\\&= \frac{e^x}{(e^x+1)^2}\end{aligned} \tag{0.4}$$

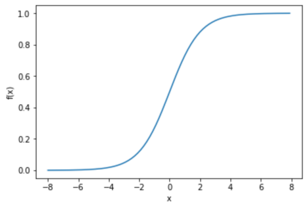
Plot of the function



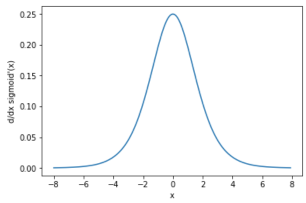
(d) Implement the sigmoid function in a iPython Notebook.
Sigmoid Function and Sigmoid Function Derivative

In [5]:

```
import matplotlib.pylab as plt
import numpy as np

x = np.arange(-8, 8, 0.1)
f = 1 / (1 + np.exp(-x))
plt.plot(x, f)
plt.xlabel('x')
plt.ylabel('f(x)')
plt.show()
```




```
d1_f = (np.exp(-x) / (1 + np.exp(-x))**2)
plt.plot(x, d1_f)
plt.xlabel('x')
plt.ylabel('d/dx sigmoid\'(x)')
plt.show()
```



(e) Show that the function First Derivative

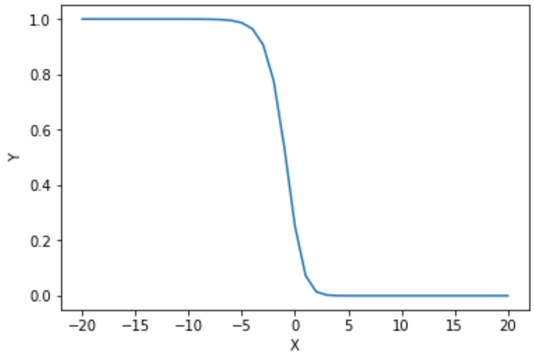
$$
\begin{aligned}
c_1(x)' &= \frac{d}{dx}[(\sigma(x)-1)^2] && \textit{Power, Chain rule} \\
&= 2(\sigma(x)-1)\cdot\frac{d}{dx}[\sigma(x)-1] && \textit{Sum rule} \\
&= 2(\sigma(x)-1)\cdot\frac{d}{dx}[\sigma(x)]-\frac{d}{dx}[1] && \textit{Result from 0.2 equation, Constant rule} \\
&= 2(\sigma(x)-1)\cdot(\sigma(x)\cdot(1-\sigma(x)))-0 \\
&= (2\sigma(x)-2)\cdot(\sigma(x)-\sigma(x)^2) \\
&= (2\sigma(x)^2-2\sigma(x)^3-2\sigma(x)+2\sigma(x)^2) \\
&= -2\sigma(x)^3+4\sigma(x)^2-2\sigma(x)
\end{aligned}
\tag{0.5}
$$

Second Derivative

$$
\begin{aligned}
c_1(x)''] &= \frac{d}{dx}[-2\sigma(x)^3 + 4\sigma(x)^2 - 2\sigma(x)] && \textit{Sum rule} \\
&= \frac{d}{dx}[-2\sigma(x)^3] + \frac{d}{dx}[4\sigma(x)^2] - \frac{d}{dx}[2\sigma(x)] && \textit{Power, Chain rule} \\
&= -6\sigma(x)^2 \cdot \frac{d}{dx}[\sigma(x)] + 8\sigma(x) \cdot \frac{d}{dx}[\sigma(x)] - 2\frac{d}{dx}[\sigma(x)] && \textit{Result from 0.2 equation} \\
&= -6\sigma(x)^2 \cdot (\sigma(x) \cdot (1-\sigma(x))) + 8\sigma(x) \cdot (\sigma(x) \cdot (1-\sigma(x))) - 2(\sigma(x) \cdot (1-\sigma(x))) \\
&= -6\sigma(x)^2 \cdot (\sigma(x) - \sigma(x)^2) + 8\sigma(x) \cdot (\sigma(x) - \sigma(x)^2) - 2(\sigma(x) - \sigma(x)^2) \\
&= -6\sigma(x)^3 + 6\sigma(x)^4 + 8\sigma(x)^2 - 8\sigma(x)^3 - 2\sigma(x) - 2\sigma(x)^2 \\
&= 6\sigma(x)^4 - 14\sigma(x)^3 + 6\sigma(x)^2 - 2\sigma(x)
\end{aligned}
\tag{0.6}
$$

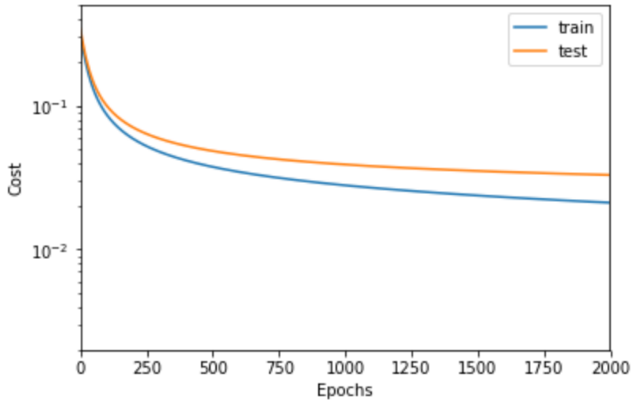
The problem is that initial settings can set us in the upper flat zone. There the slope of the function will be 0 and also the derivative, this will be problematic because the update rule will not update the value and we will be stack at the same point without finding the minimum.

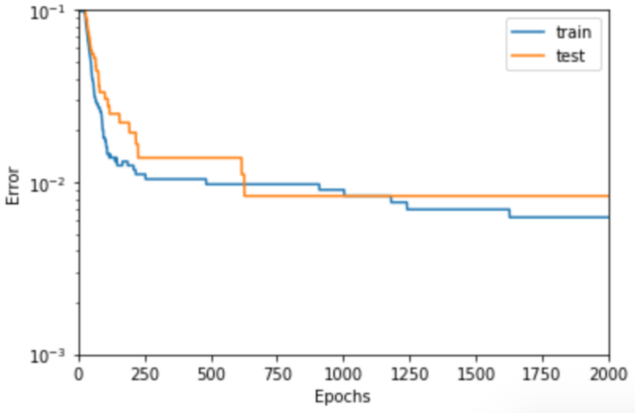
```
def nonConvex(x):
    return (sigmoid(x)-1)**2

x = np.arange(-20,21)
y = nonConvex(x)
plt.plot(x,y)
plt.ylabel('Y')
plt.xlabel('X')
plt.show()
```

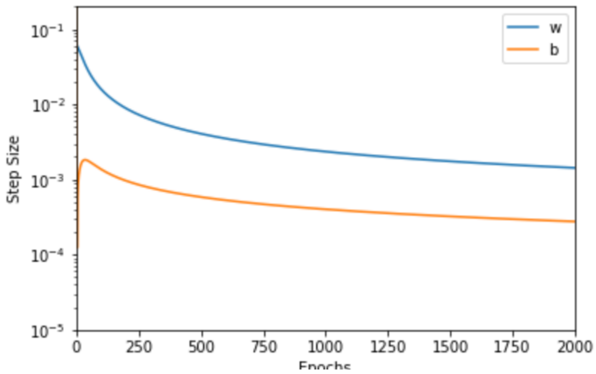


EXERCICE 3 – GRADIENT DESCENT FOR PERCEPTRON

(a) Plot Cost, Error Rate and Learning Speed.







(b) Analyze the dependency of the final error rate on the number of epochs. With a larger learning rate we could decrease the error with less epochs, but we will also get faster to a point where the error gets stable and there is no more improvement.

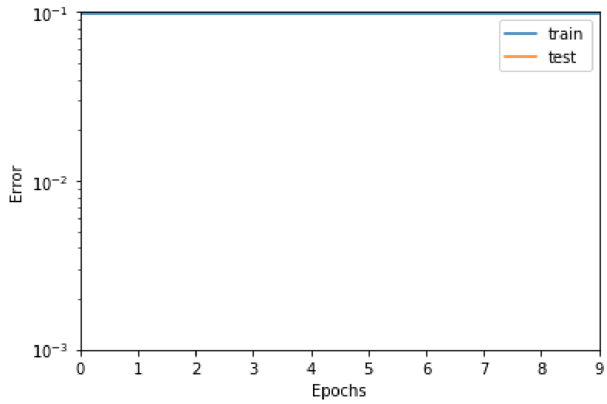


Figure 0.1: With 10 epochs we do not get any improvement on the error.

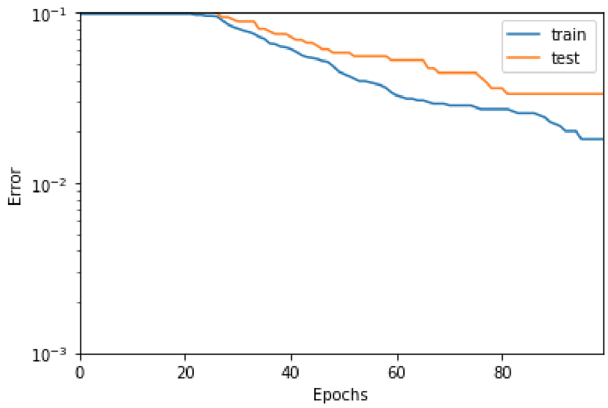


Figure 0.2: With 100 epochs we can notice that the error starts decreasing after around 25-30 epochs.

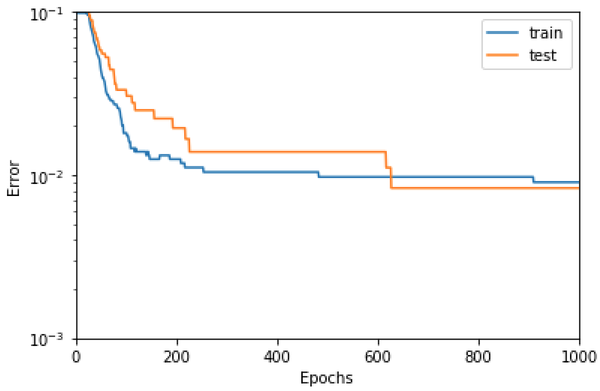


Figure 0.3: With 1000 epochs we notice that the error decreases very fast at the beginning and then it slows down and the error is more or less stable.

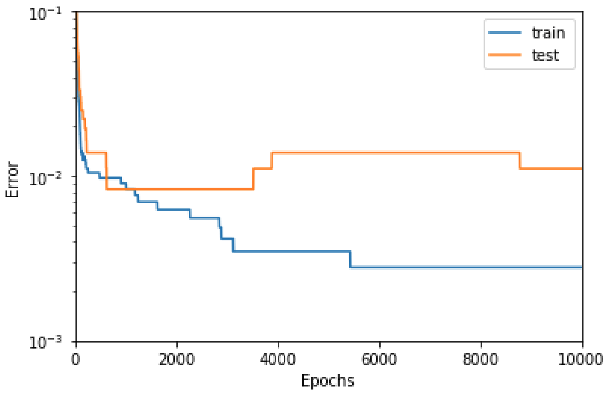


Figure 0.4: With 10000 epochs we notice easier what we said above, that after about 600 epochs the error gets stable and does not decrease any more. Even worst on the test set we see that the error increases. This could may be explained because we found the local/global minimum and we are trying to over take it. We are going "uphill".

(c) Analyze the dependency on the learning rate. As the learning rate gets larger we need less epochs to reach the same error.

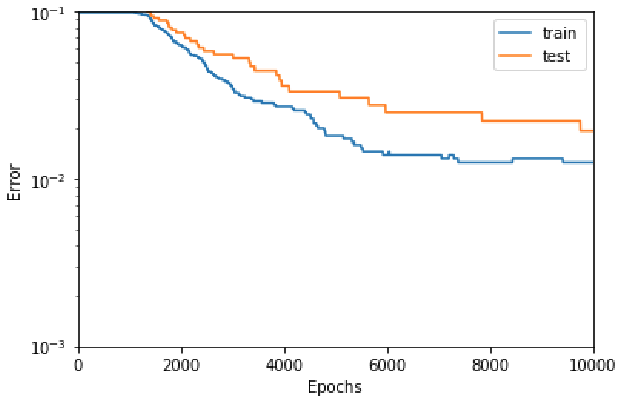


Figure 0.5: Learning rate = 0.01, Epochs = 10000

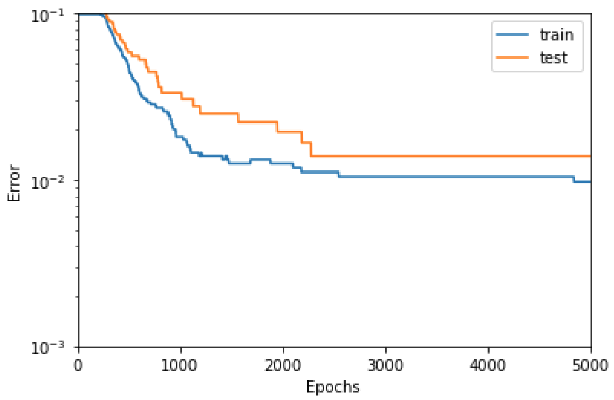


Figure 0.6: Learning rate = 0.05, Epochs = 5000

(d) Which digit can be predicted very well - for which is the prediction rather bad?

Digit	0	1	2	3	4	5	6	7	8	9
Test error	**0.0000**	0.0194	0.0111	0.0111	0.0056	0.0083	0.0111	0.0111	**0.0333**	0.0222
Test cost	0.0082	0.0609	0.0265	0.0339	0.0180	0.0330	0.0254	0.0294	0.0988	0.0627

(e) Plot the Trained Weights as 8x8 Image

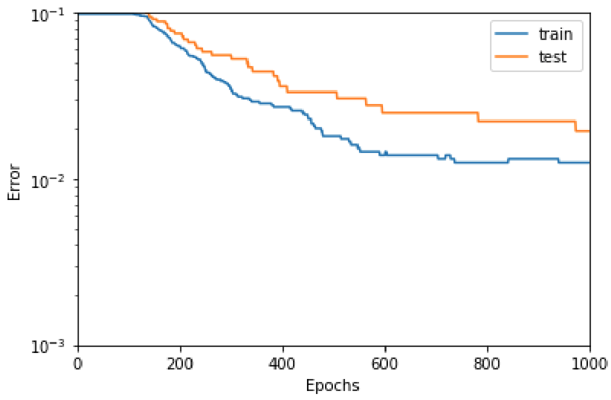


Figure 0.7: Learning rate = 0.1, Epochs = 1000

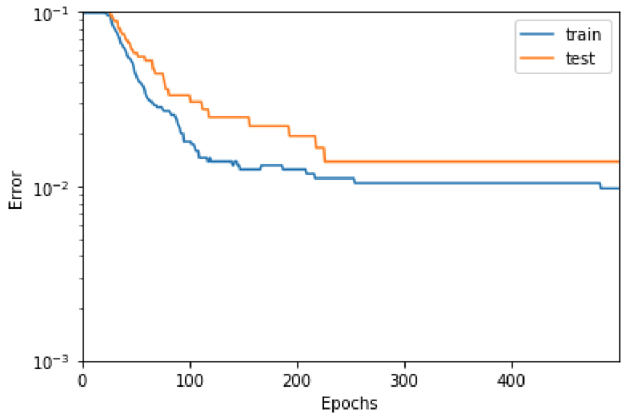


Figure 0.8: Learning rate = 0.5, Epochs = 500

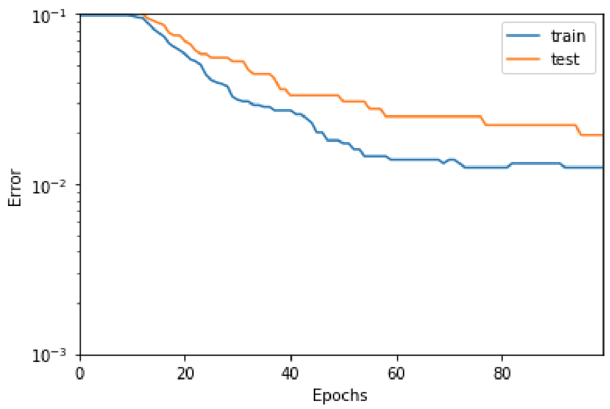


Figure 0.9: Learning rate = 1, Epochs = 100

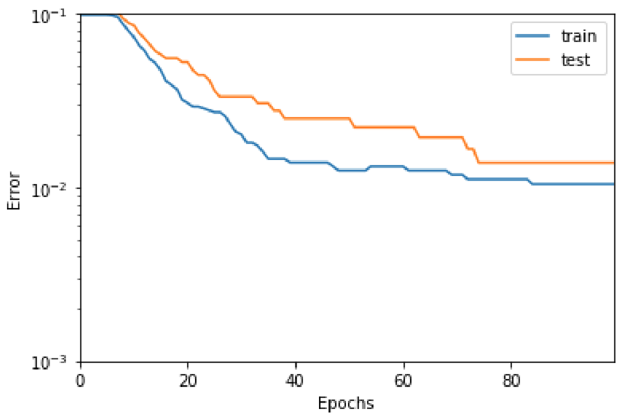


Figure 0.10: Learning rate = 1.5, Epochs = 100

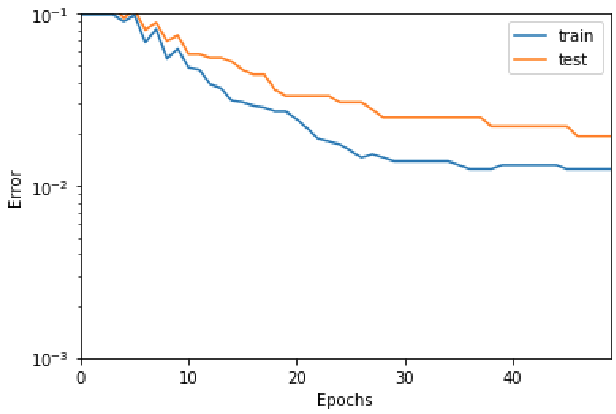


Figure 0.11: Learning rate = 2, Epochs = 50

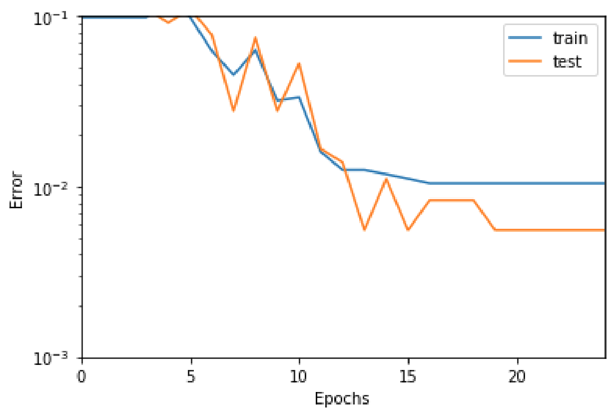


Figure 0.12: Learning rate = 5, Epochs = 25

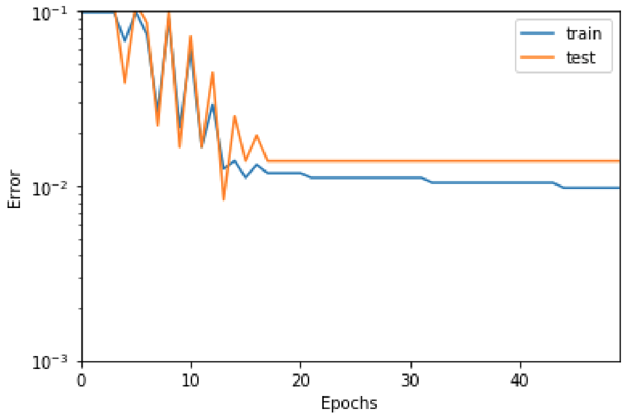


Figure 0.13: Learning rate = 10, Epochs = 50