TDT4171 - Assignment 4

Martin Bjerke

April 9, 2018

I Gradient Descent

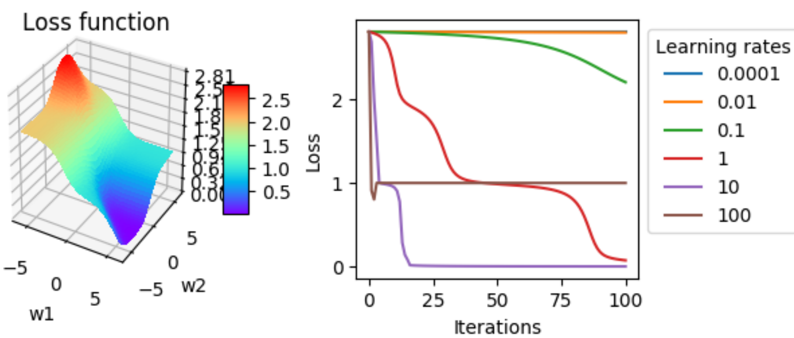


Figure 1: Loss function & learningrate

1. From figure 1 its clear that the minimum of the loss function(within the range of $w_i \in (6, -6)$) is $(w_1, w_2) = (6, -3)$.

2. Since the derivative of $\frac{\partial L_{simple}(w)}{\partial w_i}$ will be indifferent of i I'll show the calculations of w_i:

$$
\begin{aligned}
\frac{\partial L_{simple}(w)}{\partial w_i} &= \frac{[\sigma(w,[1,0]) - 1]^2 + [\sigma(w,[0,1])]^2 + [\sigma(w,[1,1]) - 1]^2}{\partial w_i} \\
&= \frac{[\sigma(w,[1,0]) - 1]^2}{\partial w_i} + \frac{[\sigma(w,[0,1])]^2}{\partial w_i} + \frac{[\sigma(w,[1,1]) - 1]^2}{\partial w_i} \\
&= \frac{1}{2}\left[\sigma(w,[1,0]) - 1\right] \frac{[\sigma(w,[1,0]) - 1]}{\partial w_i} + \frac{1}{2}\left[\sigma(w,[0,1])\right] \frac{[\sigma(w,[0,1])]}{\partial w_i} \\
&+ \frac{1}{2}\left[\sigma(w,[1,1]) - 1\right] \frac{[\sigma(w,[1,1]) - 1]}{\partial w_i}
\end{aligned}
$$

Next I find the derivative of $\sigma(w,x)$:

$$
\begin{aligned}
\frac{\partial \sigma(w,x)}{\partial w_i} &= \frac{\partial (1+e^{-w^T x})^{-1}}{\partial w_i} \\
&= -\left(\frac{1}{1+e^{-w^T x}}\right)^{-2} \cdot \frac{\partial (1+e^{-w^T x})}{\partial w_i} \\
&= -\left(\frac{1}{1+e^{-w^T x}}\right)^{-2} \cdot (-x_i e^{-w^T x}) \\
&= x_i \frac{e^{-w^T x}}{1+e^{-w^T x}} \cdot \frac{1}{1+e^{-w^T x}} \\
&= x_i \frac{1+e^{-w^T x}-1}{1+e^{-w^T x}} \cdot \frac{1}{1+e^{-w^T x}} \\
&= x_i \left[\frac{1+e^{-w^T x}}{1+e^{-w^T x}} - \frac{1}{1+e^{-w^T x}}\right] \cdot \frac{1}{1+e^{-w^T x}} \\
&= x_i \left[1 - \frac{1}{1+e^{-w^T x}}\right] \cdot \frac{1}{1+e^{-w^T x}} = x_i \sigma(w,x)\left[1-\sigma(w,x)\right]
\end{aligned}
$$

Inserting this:

$$
\begin{aligned}
\frac{\partial L_{simple}(w)}{\partial w_i} &= \frac{x_i}{2}\left[\sigma(w,[1,0])-1\right]\sigma(w,[1,0])\left[1-\sigma(w,[1,0])\right] \\
&+ \frac{x_i}{2}\left[\sigma(w,[0,1])\right]\sigma(w,[0,1])\left[1-\sigma(w,[0,1])\right] \\
&+ \frac{x_i}{2}\left[\sigma(w,[1,1])-1\right]\sigma(w,[1,1])\left[1-\sigma(w,[1,1])\right]
\end{aligned}
$$

This derivative results in the following derivative of the loss function:

$$
\begin{aligned}
\nabla_w L_{simple}(w) &= \left[\frac{\partial L_{simple}(w)}{\partial w_1}, \frac{\partial L_{simple}(w)}{\partial w_1}\right] \\
\frac{\partial L_{simple}(w)}{\partial w_1} &= -\frac{1}{2}\left(\sigma(w,[1,0])\left[\sigma(w,[1,0]-1)\right]^2 + \sigma(w,[1,1])\left[\sigma(w,[1,1]-1)\right]^2\right) \\
\frac{\partial L_{simple}(w)}{\partial w_1} &= -\frac{1}{2}\left(\sigma(w,[0,1])^2\left[\sigma(w,[0,1])\right] + \sigma(w,[1,1])\left[\sigma(w,[1,1]-1)\right]^2\right)
\end{aligned}
$$

3. From figure 1 you can see that with a learningrate of 1 and 10 minimizes L_{simple} after 100 and 15 iterations each, while the smaller iterations bearly starts to minimize after 100 iterations.

II Perceptron

1. Continuing the calculations from above:

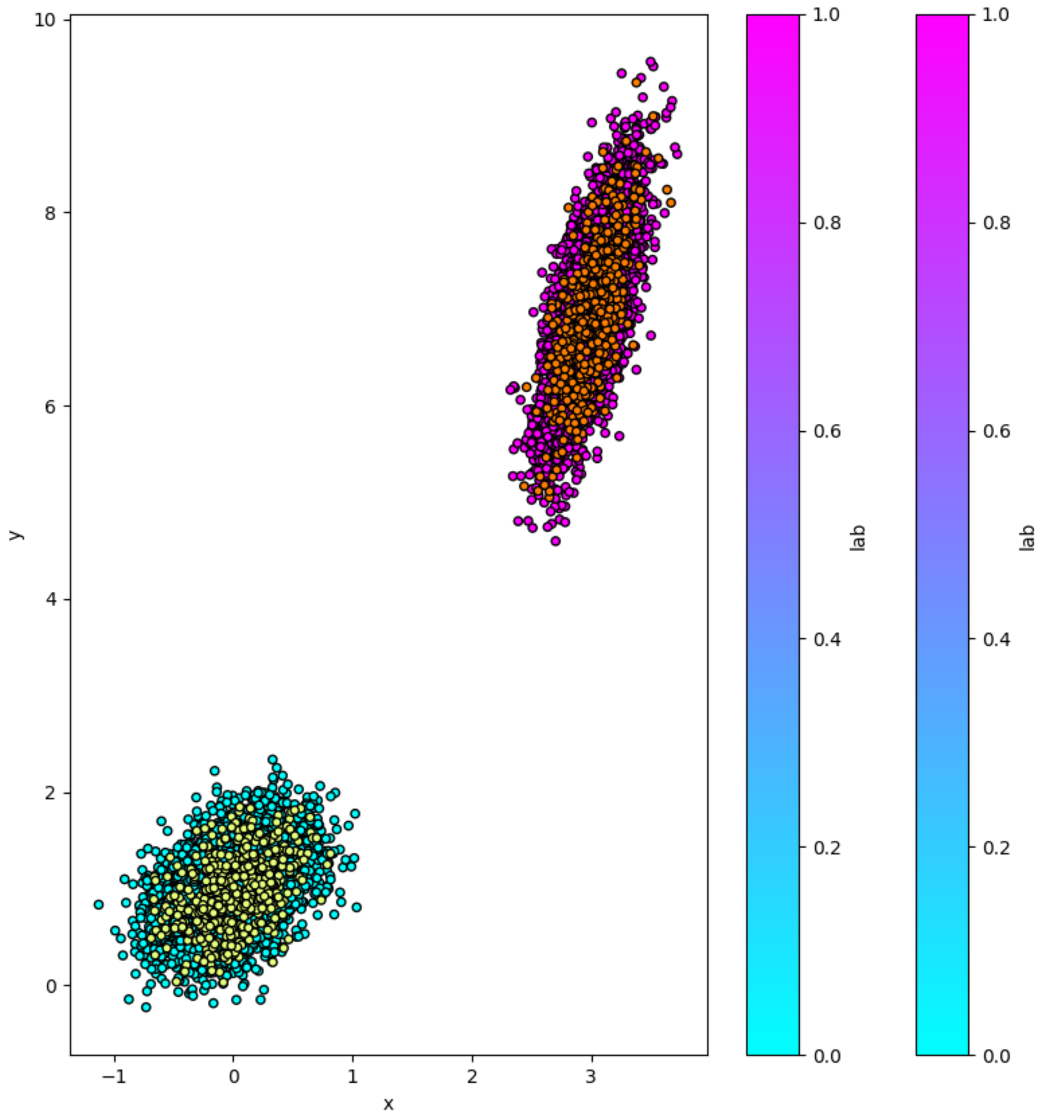
$$
\frac{\partial L_n(w,x_n,y_n)}{\partial w_i} = x_i \cdot \left[\sigma(w,x_n)-y_n\right]\left[1-\sigma(w,x_n)\right]\sigma(w,x_n)
$$

2. The table below show the avgarge when the perceptron was trained 10 times with 200 iterations, time is given in seconds and errors are given in percent where 0% meaning everything was classified correctly.

dataset	Batch		Stochastic	
	Training time(s)	Avg. error(%)	Training time(s)	Avg. error(%)
Big_nonsep	57.86	19.3	0.01	21.4
Big_sep	59.31	00.9	0.02	1.5
Small_nonsep	12.07	20.6	0.01	21.3
Small_sep	11.89	00.8	0.01	2.6

From the table above it is clear that batch gradient descent training is slower, but slightly more accurate on these datasets. This result concurces with the algorithm as batch gradient descent calculates $T \cdot (d) \cdot |D_{train}|$ gradients while stochastic only calculates $T \cdot (d)$. By looking further on the number of gradients calculated by the methods the correlation between the times also becomes apparent; Stochastic doesn't depend on the size of the dataset and calculates $200 \cdot 2 = 400$ gradients, while batch calculates $400 \cdot 11000$ and $400 \cdot 2200$ gradients on 200 iterations. As the big datasets are 5 times as large as the small, the difference in training time correlates with this by beeing 5 times as large. As one would believe the non separable datasets are more inacurate and the seperability of the dataset has no effect on the time used by the different methods since the code does not take this into consideration when calculating the gradients.

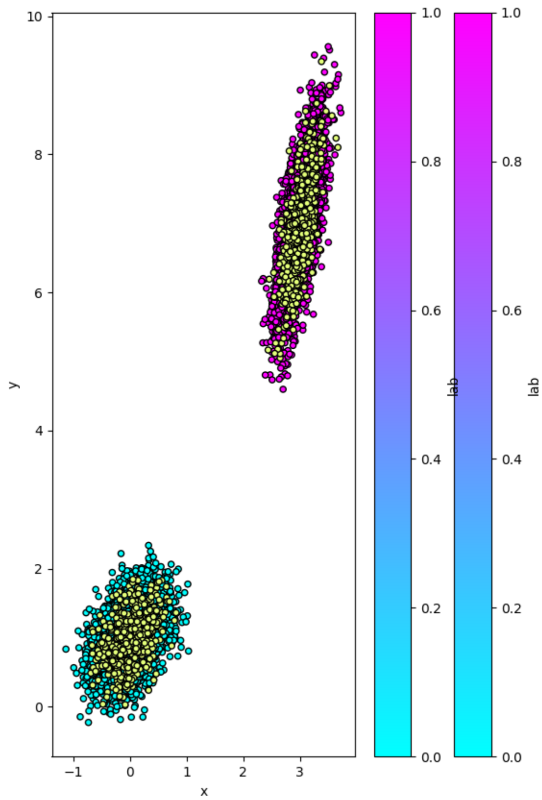
3. Training the perceptron using stochastic gradients descent on the the small seperatable dataset and ploting the results gives the following plot.

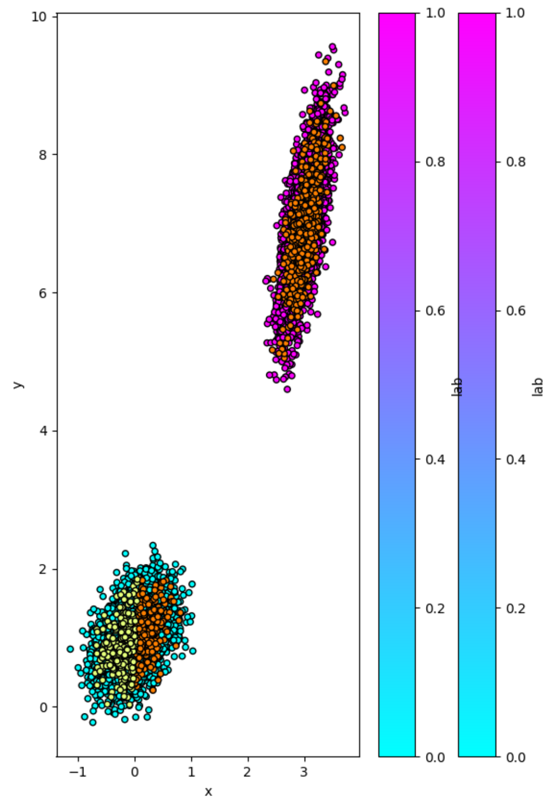


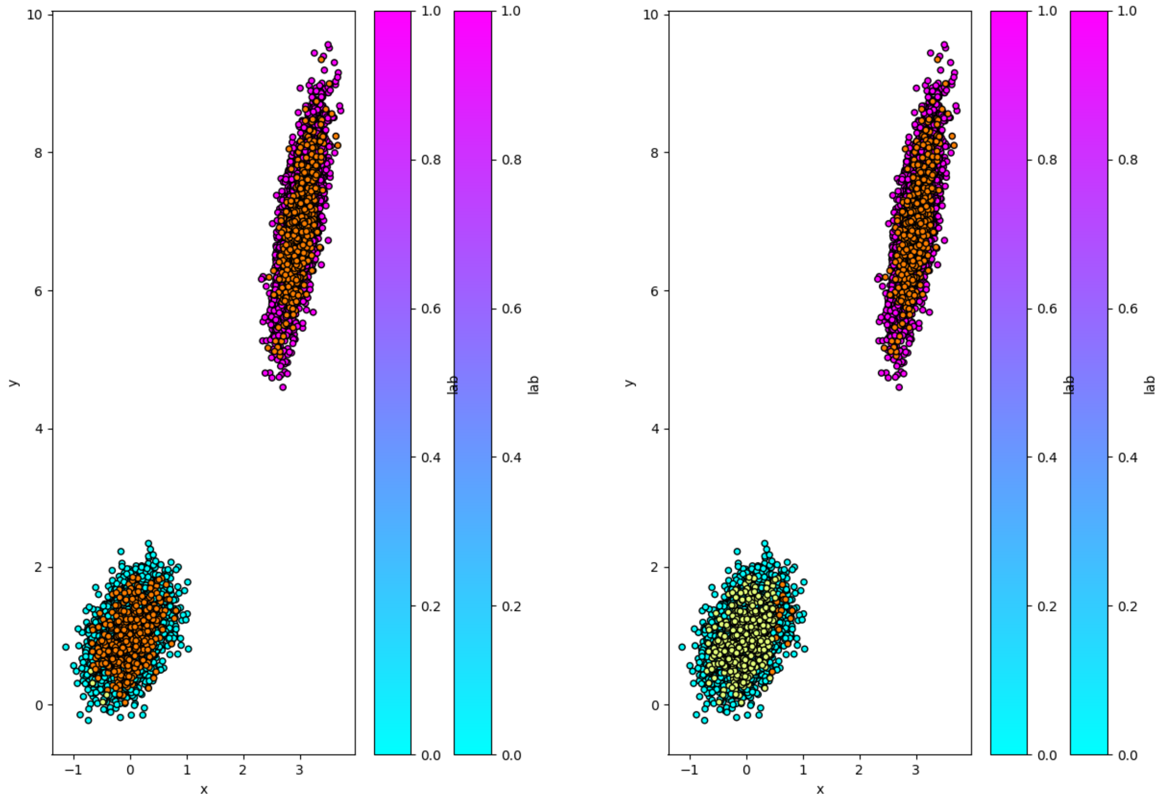
4. The same dataset and algorithm as above gives the following results(plots are placed after the discussion):

Iterations	Error(%)	Time
10	50.0	0.008
20	21.8	0.009
50	49.6	0.009
100	1.8	0.010
500	0.0	0.019

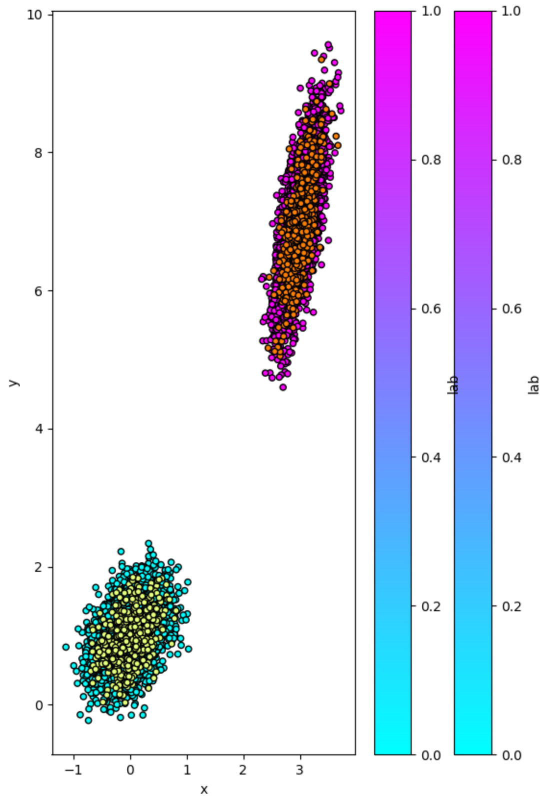
The first mention should be the 49.6% error rate on 50 iterations. The results are for 5 different sets of initial weights, so the error increase from 20 to 50 iterations is purly random. Other then this the general trend is a decreased error rate for each iteration, concuring with results gotten on the previous tasks and experiments done during this assignment(where I didn't store the results). The reason of this decrease in error rate is the correction of the weights, which in this case(a 2-d fature vector) can be viewed as trying to place a line between the 2 classifications of the dataset, for each iteration the weights are corrected to minimize the loss function, translating to moving the line in the graph.







y
x
lab
lab
y
x
lab
lab


10
8
6
4
2
0
y
−1
0
1
2
3
x
1.0
0.8
0.6
0.4
0.2
0.0
lab
1.0
0.8
0.6
0.4
0.2
0.0
lab