The Perceptron

1. Consider the function $f(x)=x^3+2x^2$ and the current solution $x_t=\langle 2\rangle$ compute one step of gradient descent with learning rate $\eta=0.1$.

   $\nabla f(x)=3x^2+4x$
   $\nabla f(x_t)=3(2)^2+4(2)=20$
   $x_{t+1}=x_t-\eta\nabla f(x_t)=2-0.1\cdot 20=0$

   The new point is $x_{t+1}=0$. We can verify that it is an improvement over the previous point, because the value of the function is lower: $f(2)=16$, while $f(0)=0$. It is also a minimum, since $\nabla f(0)=0$.

2. Consider the function $f(x,y,z)=x^2+yz+yz^2$ and the current solution $x_t=\langle 1,1,-1\rangle$ compute one step of gradient descent with learning rate $\eta=0.1$.

   $\nabla f(x)=\langle 2x, z+z^2, y+2yz\rangle$
   $\nabla f(x_t)=\langle 2,0,-1\rangle$
   $x_{t+1}=\langle 1,1,-1\rangle-0.1\cdot\langle 2,0,-1\rangle=\langle 0.8,1,-0.9\rangle$.

3. The threshold of the activation function of the perceptron is a parameter to learn, as much as the weights. How can we put the threshold in the same form as the weights, so that the same update rule can be used for it?

   The equation of the output of the perceptron is:

   $$o(\boldsymbol{x})=\begin{cases}1 \text{ if } \boldsymbol{w}^T\boldsymbol{x}>k\\ 0 \text{ if } \boldsymbol{w}^T\boldsymbol{x}\leq k\end{cases}$$

   both inequalities can be rewritten by moving the threshold k to the other side, and incorporating it into the vector of weights, by adding a constant input:

   $$\boldsymbol{w}^T\boldsymbol{x}-k>0=\sum_{i=1} w_i x_i+k(-1)=0$$

   where -1 can be added to the inputs (forming the *bias* input), and k to the weights.

4. What is the equation of the decision boundary for a perceptron? What does it represent?

   $\boldsymbol{w}^T\boldsymbol{x}+w_0=0$. It is the equation of an hyperplane.

5. Given the equation of the decision boundary of a perceptron in 2D (a straight line), what is the slope and what is the intercept?

In 2D the equation is: $w_0+w_1x_1+w_2x_2=0$. From which: $x_2=-\frac{w_1}{w_2}x_1-\frac{w_0}{w_2}$, where $-\frac{w_1}{w_2}$ is the slope, and $-\frac{w_0}{w_2}$ is the intercept.

6. What does it mean for two classes to be linearly separable in D dimensions?

   That there exists a hyperplane in D dimensions that separates the points of the two classes.

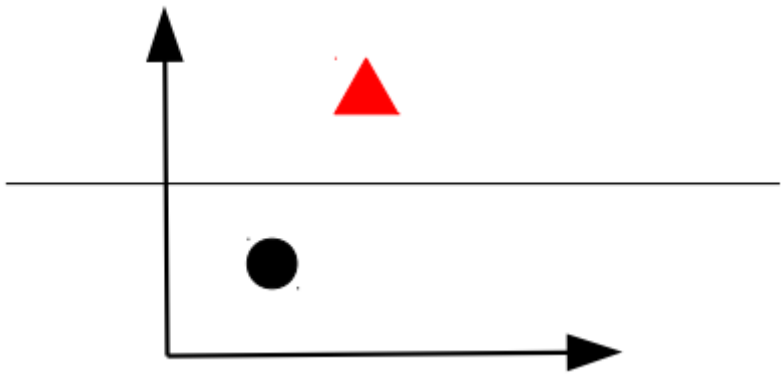
7. Will the perceptron algorithm always converge to a point with zero error?

   Only if the dataset is linearly separable.

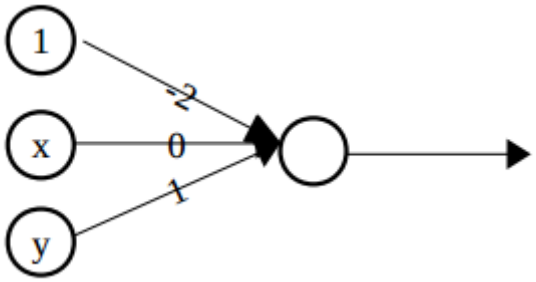
8. What error function do we use to derive the update rule for the perceptron? Why not the number of errors on a dataset? What is the advantage of the function we use over the number of errors?

   We use the error function: $E(\boldsymbol{X})=\sum_{x_n\in\boldsymbol{X}}\boldsymbol{w}^T\boldsymbol{x}_n(y_n-t_n)$ where X is the dataset, y_n is the output of the perceptron on point x_n, and t_n is the desired output for point x_n . We prefer this function to the number of errors because it is proportional to the distance of the misclassified points from the decision boundary and is differentiable. The number of errors, on the other hand, does not provide any direction for improvement, since the error does not change smoothly as the hyperplane changes.

9. Construct a perceptron able to separate the points: <1,1,0>, <2,3,1> where the last element is the class.



   One possible solution is given by the perceptron implementing the boundary y=2 which corresponds to the weight vector: $\langle -2,0,1\rangle$. Note that any line separating the classes is a valid solution! The resulting perceptron is:



10. Add a new point to the dataset from the previous question, so that the perceptron you computed missclassifies it. Perform one step of gradient descent on the perceptron error to improve the error on the new data point.

    The point <2,1> of class 1 is misclassified by the perceptron above. The error is:

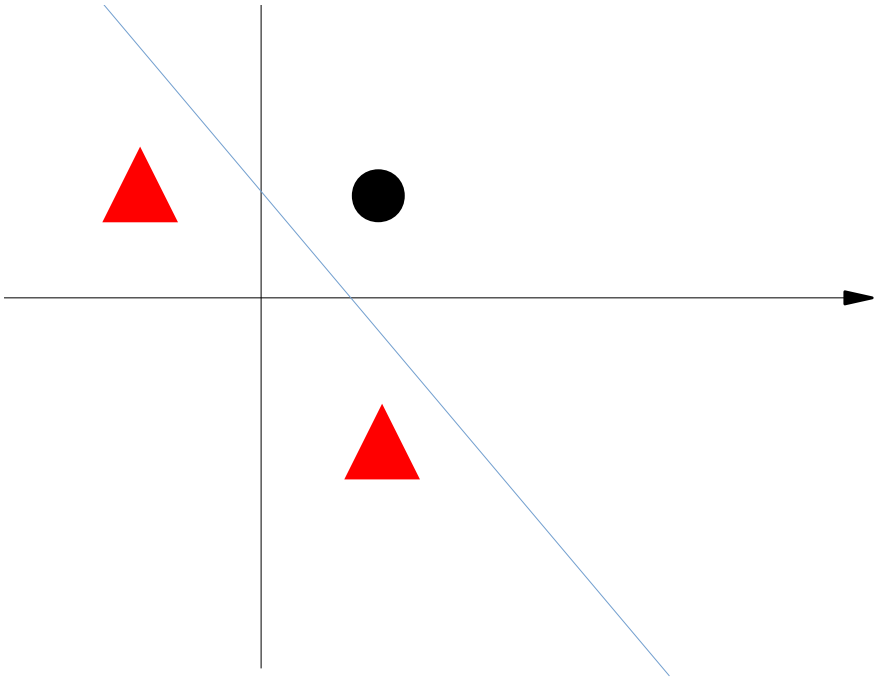
    $$E(\boldsymbol{X})=\sum_{x_n\in\boldsymbol{X}} \boldsymbol{w}^T \boldsymbol{x}_n(y_n-t_n)=[-2,0,1]\cdot\begin{bmatrix}1\\2\\1\end{bmatrix}(0-1)=1 \quad .$$

    Stochastic gradient descent performs the update $\boldsymbol{w}_{k+1}=\boldsymbol{w}_k-\eta\boldsymbol{x}(y-t)$, in our case:

    $\boldsymbol{w}_{k+1}=\langle -2,0,1\rangle-0.1\cdot\langle 1,2,1\rangle\cdot(0-1)=\langle -1.9,0.2,1.1\rangle$.

    The error of the new vector is: $[-1.9,0.2,1.1]\cdot\begin{bmatrix}1\\2\\1\end{bmatrix}(0-1)=0.4$ which is lower than before.

11. Construct a perceptron able to separate the points <1,1,0>, <1, -1, 1>, and <-1, 1, 1> where the last element is the class.



    An equation that separates the points is $y=-x+1$ which is equivalent to $x+y-1=0$ which leads to the weight vector: $\langle -1,1,1\rangle$. The part of the vector that is multiplied by

the variables (note how this is the gradient of the line…) is $\langle 1,1\rangle$ and points towards the point of class 0, while we want the perceptron to return 1 in the other half-plane. Therefore we need to multiply the vector by -1, and obtain the weight vector: $\boldsymbol{w}=\langle 1,-1,-1\rangle$ and the corresponding perceptron: