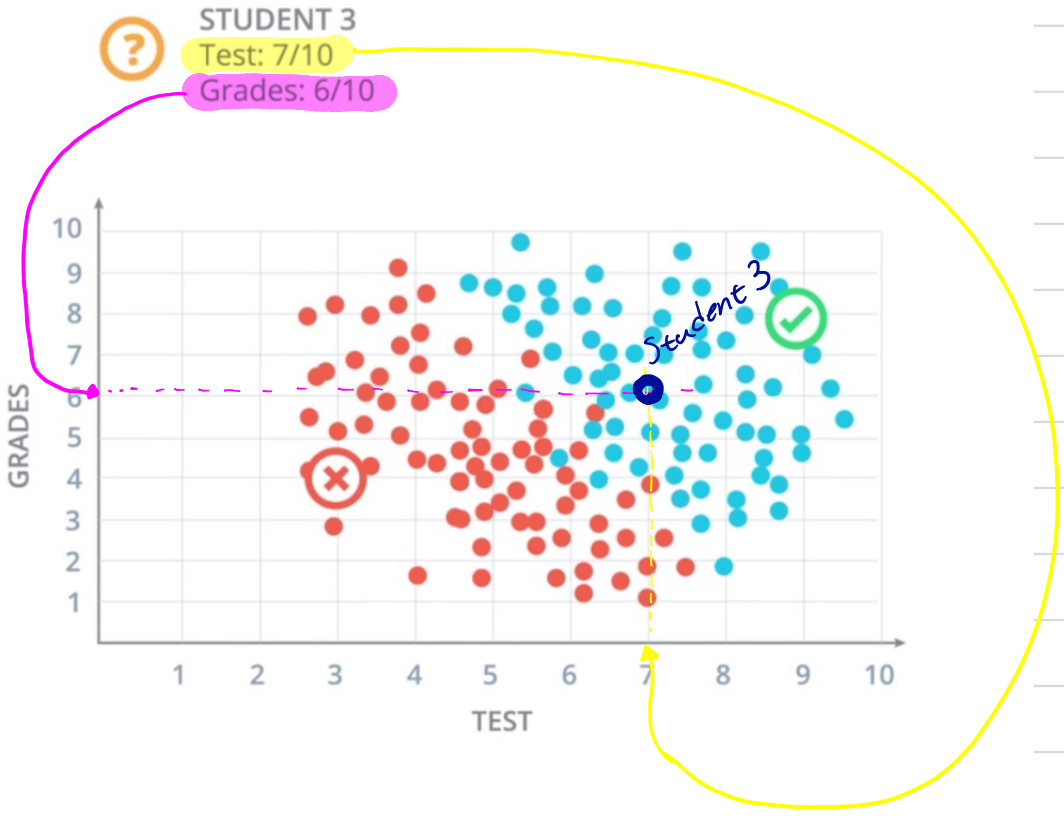
# Lesson 2

## Questions - Explanations

# Lesson 2-2: Classification Problems I

## Question 1: Does the student get accepted?

We should find out if the student gets accepted by trying to "draw" him at the chart.




As we notice, the student is surrounded by "accepted" students. So it's safe to assume that himself to will be accepted.

# Lesson 2-4: Linear Boundaries

Question 1: now that you know the equation for the line (2x1 + x2 - 18=0), and similarly the "score" (2x1 + x2 - 18), what is the score of the student who got 7 in the test and 6 for grades?

We can easily calculate the students score by replacing the variables in the equation with his test/grades numbers.

We remember that $x_1$ represents the test and $x_2$ represents the grades.

Therefore: $\text{score} = 2x_1 + x_2 - 18 =$

$= 2 \cdot 7 + 6 - 18 =$

$= 14 - 12 =$

$= 2$

P.S: Our previous assumption was right. The student should be accepted, as his score is $>0$.

# Lesson 2-5: Higher Dimensions

Question 1: Given the table in the video above, what would the dimensions be for input features (x), the weights (W), and the bias (b) to satisfy (Wx + b)?

## Acceptance at a University

| | $x_1$ EXAM 1 | $x_2$ EXAM 2 | $x_3$ GRADES | ... | $x_n$ ESSAY | y PASS? |
|---|---|---|---|---|---|---|
| STUDENT 1 | 9 | 6 | 5 | ... | 6 | 1(yes) |
| STUDENT 2 | 8 | 4 | 8 | ... | 3 | 0(no) |
| ... | ... | ... | ... | ... | ... | |
| STUDENT n | 6 | 7 | 2 | ... | 8 | 1(yes) |

n-dimensional space
$x_1, x_2, \ldots, x_n$

**BOUNDARY:**
n-1 dimensional hyperplane
$w_1x_1 + w_2x_2 + w_nx_n + b = 0$
$Wx + b = 0$

**PREDICTION:**

$$\hat{y} = \begin{cases} 1 \text{ if } Wx + b \geq 0 \\ 0 \text{ if } Wx + b < 0 \end{cases}$$

We want the dimensions to be that way so $Wx + b = 0$ is equilivant with $w_1x_1 + w_2x_2 + \ldots + w_nx_n + b = 0$

For that, we need a row vector $n \times 1$ for the inputs and a column vector $1 \times n$ for the weights

The bias $b$ is just $1 \times 1$ so it can be added to the result of the matrix multiplication $Wx$

Example: $Wx + b = [x_1\ x_2]\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} + b = [7\ \ 6]\begin{bmatrix} 2 \\ 1 \end{bmatrix} + (-18) = 7 \cdot 2 + 6 \cdot 1 - 18 =$

$= 14 - 12 = 2$

# Lesson 2-6: Perceptrons

Question 1: Given Score = 2*Test + 1*Grade - 18, suppose w1 was 1.5 instead of 2. Would the student who got 7 on the test and 6 on the grades be accepted or rejected?

At first, the equation was:

$$\overset{w_1}{2} * \text{Test} + \overset{w_2}{1} * \text{Grade} - 18$$

By changing $w_1$ to 1.5, the equation becomes:

$$1{,}5 * \text{Test} + 1 * \text{Grade} - 18$$

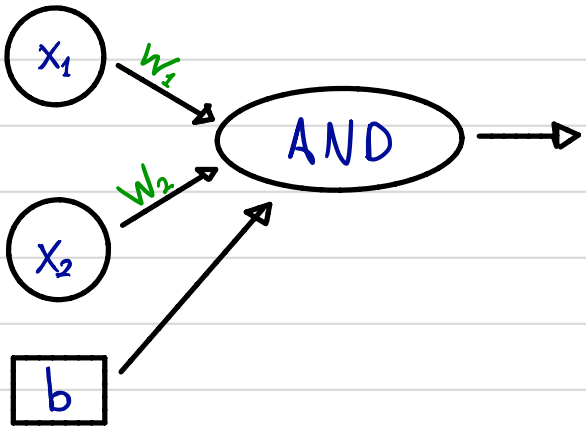
The only thing we need to do now is to replace test/grade with the students numbers and calculate the result.

$$\begin{aligned} \text{score} &= 1{,}5 * 7 + 1 * 6 - 18 = \\ &= 10{,}5 - 12 \\ &= -1{,}5 \end{aligned}$$

The result is $< 0$, so the student is rejected

# Lesson 2-8: Perceptrons as Logical Operators.

Question 1: What are the weights and bias for the AND perceptron?
Set the weights (weight1, weight2) and bias (bias) to values that will correctly determine the AND operation as shown above.
More than one set of values will work!




①

| $x_1$ | $x_2$ | AND |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

$w_1 = ?$
$w_2 = ?$
$b = ?$

To determine the weights and bias, ... must remember that the result comes from:

$$w_1 x_1 + w_2 x_2 + b \quad ②$$

Here, $x_1$ and $x_2$ can be either 0 or 1 and we want the result of the equation ② to be according the table ①.
We remember that the equation will result to: $\begin{cases} 0 & \text{when result} < 0 \\ 1 & \text{when result} \geq 0 \end{cases}$

It's easy to start with weights equal to 1 and see if we can achieve the desired outcome by changing the bias b.

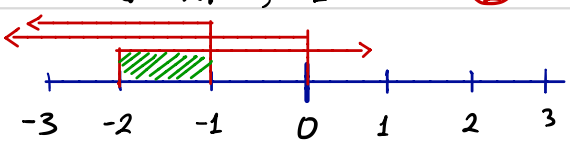
we want $< 0$ so the outcome becomes 0.

For $w_1 = w_2 = 1$ and $x_1 = x_2 = 0$: ② $\Rightarrow 1 \cdot 0 + 1 \cdot 0 + b < 0 \Rightarrow b < 0$

–//– and $x_1 = 0, x_2 = 1$: ② $\Rightarrow 1 \cdot 0 + 1 \cdot 1 + b < 0 \Rightarrow b < -1$

–//– and $x_1 = 1, x_2 = 0$: ② $\Rightarrow 1 \cdot 1 + 1 \cdot 0 + b < 0 \Rightarrow b < -1$

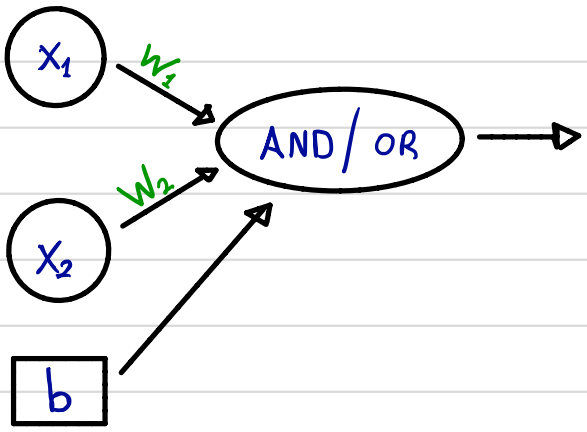
–//– and $x_1 = 1, x_2 = 1$: ② $\Rightarrow 1 \cdot 1 + 1 \cdot 1 + b \geq 0 \Rightarrow b \geq -2$




Any values between $[-2, -1)$ will work for our case.
We can choose $b = -2$

# Lesson 2-8: Perceptrons as Logical Operators. (continue)

Question 2: What are two ways to go from an AND perceptron to an OR perceptron?




| $X_1$ | $X_2$ | AND | OR |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 |

**OPTIONS**

1. Increase the weights
2. Decrease the weights
3. Increase a single weight
4. Decrease a single weight
5. Increase the bias
6. Decrease the bias

We know that the outcome depends on:

$$w_1 \cdot x_1 + w_2 \cdot x_2 + b$$

And with our previously calculated weights ($w_1 = w_2 = 1$) and bias ($b = -2$)

$$1 \cdot x_1 + 1 \cdot x_2 - 2 \quad (1)$$

The equation (1) is correctly $\geq 0$ only when both $x_1 = x_2 = 1$.
For OR we want it to be $\geq 0$ if at least one of $x_1$ or $x_2$ is 1.

We notice that the equations result can increase (to a point that it will get $>0$ if only $x_1$ or $x_2$ is 1) if we:

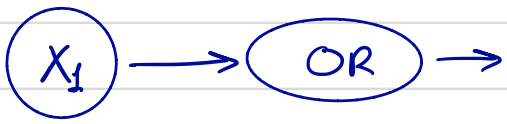
- increase the weights – ex: $\left.\begin{matrix} w_1 = 2 \\ w_2 = 2 \end{matrix}\right\}$ (1) $\Rightarrow 2 \cdot 0 + 2 \cdot 1 - 2 = 0 \geq 0$ ✓
- decrease the bias – ex: $b = 1$ } (1) $\Rightarrow 1 \cdot 0 + 1 \cdot 1 - 1 = 0 \geq 0$ ✓

# Lesson 2-8: Perceptrons as Logical Operators. (continue)

Question 3: NOT Perceptron

Unlike the other perceptrons we looked at, the NOT operation only cares about one input. The operation returns a 0 if the input is 1 and a 1 if it's a 0. The other inputs to the perceptron are ignored.

In this quiz, you'll set the weights (weight1, weight2) and bias bias to the values that calculate the NOT operation on the second input and ignores the first input.




| $x_1$ | OR |
|---|---|
| 0 | 1 |
| 1 | 0 |

The outcome is calculated by: $w_1 \cdot x_1 + w_2 \cdot x_2 + b$ (1)

We only care about the second input, according to description, so we can have $w_1 = 0$ and $b = 0$

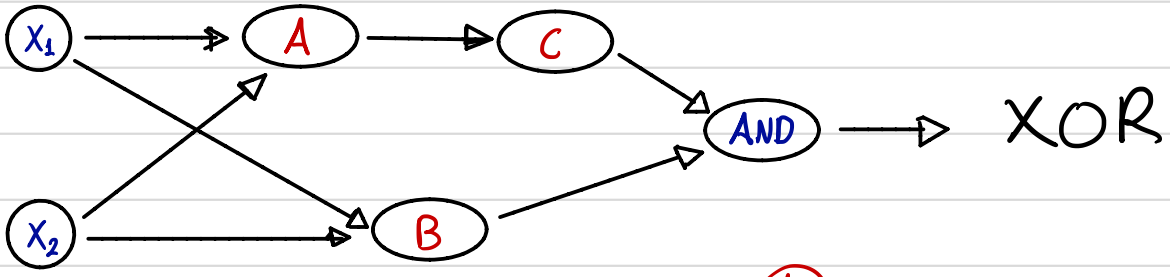
From there: (1) $\Rightarrow$ $w_2 \cdot x_2$ should be $\geq 0$ when $x_2 = 0$
and $< 0$ when $x_2 = 1$

This is easily done with $w_2 = -1$

Example: $x_2 = 0$ : $-1 \cdot 0 = 0 \geq 0$ ✓
$x_2 = 1$ : $-1 \cdot 1 = -1 < 0$ ✓

# Lesson 2-8: Perceptrons as Logical Operators. (continue)

Question 4: Quiz: Build an XOR Multi-Layer Perceptron
Now, let's build a multi-layer perceptron from the AND, NOT, and OR perceptrons to create XOR logic!




(1)

| $X_1$ | $X_2$ | XOR |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

I am sure there is a more "scientific" way to solve this, but this is a solution too.

C is the only one with one input and we know that this means it can only be a NOT.

Now we have to determine if the correct is:

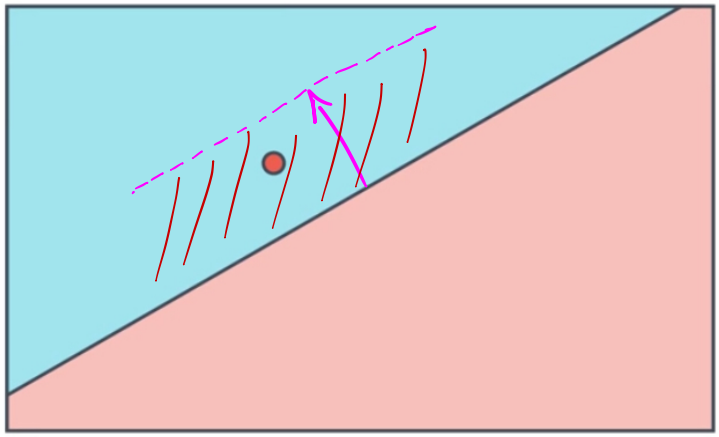
A is OR
B is AND

A is AND
B is OR

We follow the diagram above by replacing $X_1, X_2$ with the 4 possible pairs and compare the outcome with the table (1). The structure that matches on all outcomes is the correct one.

# Lesson 2-9: Perceptron Trick

Question 1: Does the misclassified point want the line to be closer or farther?



**QUIZ**

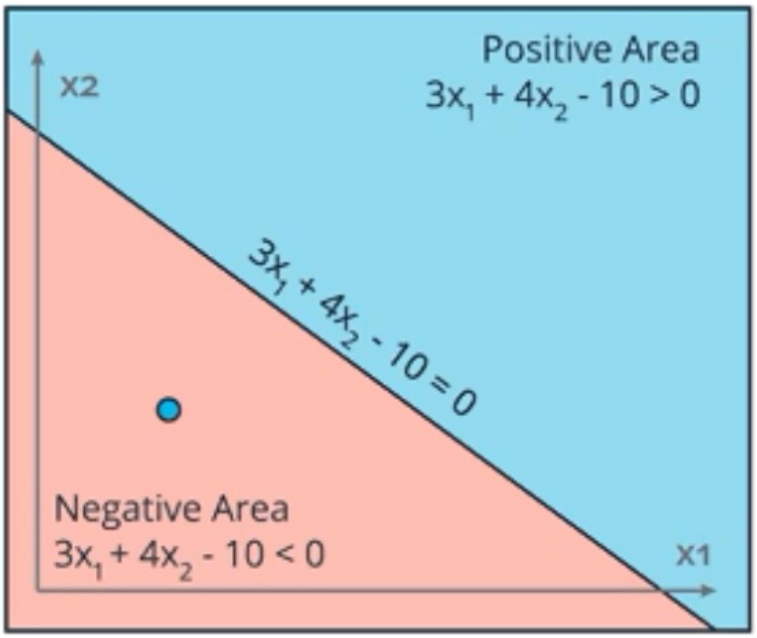
Does the misclassified point want the line to be close or farther?

- ○ Closer
- ○ Farther

We want the red point to be placed in the red area. For that to happen, the line has to move closer and above the point.

# Lesson 2-9: Perceptron Trick (continue)

Question 2: For the second example, where the line is described by 3x1+ 4x2 - 10 = 0, if the learning rate was set to 0.1, how many times would you have to apply the perceptron trick to move the line to a position where the blue point, at (1, 1), is correctly classified?




LINE: $3x_1 + 4x_2 - 10 = 0$
POINT: (1,1)

As the point is in the negative area, we need to add to the line.

Line Point

1 time: $3x_1 + 4x_2 - 10 = 0$ $(1,1)$ 1 (bias)

$+0.1 \quad 0.1 \quad 0.1$ ← ×0.1 learning rate

new line: $3,1x_1 + 4,1x_2 - 9,9 = 0$

For the new line, we check if the point (1,1) makes it true ($\geq 0$)

$3,1 \cdot 1 + 4,1 \cdot 1 - 9,9 = 7,2 - 9,9 = -2,7 < 0$ ✗

We need to repeat the procedure, until the outcome is $\geq 0$.

After 10 times the line becomes: $4x_1 + 5x_2 - 9 = 0 \xrightarrow{(1,1)} 4 + 5 - 9 = 0 \geq 0$ ✓

# Lesson 2-10: Perceptron Algorithm

Question 2: Coding the Perceptron algorithm.

```
def perceptronStep(X, y, W, b, learn_rate = 0.01):
```

First, we should understand what each parameter is:

X: Our data. example:
y: The correct label (0 or 1) of a point
W: The weights of our function
$W = [w_1, w_2]$
b: The bias, ex: b = -18
learn_rate: The number we multiply our changes with on every repetition.

In Python:
examples:

```
X = [[0.1, 0.5, 1]
     [0.2, 0.4, 1]
     [0.7, 0.3, 0]]

W = [3.2, 1.7]
```

Now, we want to:

- Loop through our data X:

```
for i in range(len(X)):
```

- Calculate the prediction:

```
y_hat = prediction(X[i], W, b)
```

- Do nothing if the point is: correctly classified

```
if y[i] == y_hat:
    continue
```

- Update the weights if the: point is wrong classified negative (and the bias)

```
if y_hat == 0:
    W[0] += X[i][0] * learn_rate
    W[1] += X[i][1] * learn_rate
    b    += learn_rate
```

# Lesson 2-10: Perceptron Algorithm (continue)

- update the weights if the point is wrong classified positive (and the bias)

```
elif y_hat == 1:
    W[0] -= X[i][0] * learn_rate
    W[1] -= X[i][1] * learn_rate
    b    -= learn_rate
```

- return the updated:

```
return W, b
```

# Lesson 2-13: Log-loss Error Function

Question 1: Which of the following conditions should be met in order to apply gradient descend?

- Should be discrete ✗ No, as then we wouldn't be able to understand small changes in our function.
- Should contain only positive values ✗ No, it was never mentioned as a requirement. It's the sum of all errors that matters.
- Should be differentiable ✓ Yes, it's mentioned that will be explained later.
- Should be normalized ✗ Never mentioned
- Should be continuous ✓ Yes, so small changes affect the error, therefore we know if we are doing better than before.

# Lesson 2-13: Log-loss Error Function

Question 1: The sigmoid function is defined as ...
if the score is defined by 4x1 + 5x2 - 9 = score, then which of the following points has exactly 50% probability of being blue or red?

We first calculate the score for a point:

example: $(1,1) \Rightarrow$ $\text{score} = 4x_1 + 5x_2 - 9 = 4 \cdot 1 + 5 \cdot 1 - 9 = 0$

Then we pass it through the sigmoid function: $\text{sigmoid}(0) = \frac{1}{1+e^0} = \frac{1}{1+1} = 0.5$ or 50%

Similar for $(2,4) \Rightarrow$ $\text{score} = 4 \cdot 2 + 5 \cdot 4 - 9 = 8 + 20 - 9 = 19$

Again, through the sigmoid: $\text{sigmoid}(19) = \frac{1}{1+e^{-19}} = \frac{1}{1+0,000...05} \simeq 0.99$ or 99%

And we repeat for the rest:

$(5,5) \Rightarrow$ $\text{score} = 4 \cdot 5 + 5(-5) - 9 = 20 - 25 - 9 = -14$

$\text{sigmoid}(-14) = \frac{1}{1+e^{14}} = \frac{1}{1+1202604} \simeq 0,01$ or 1%

$(-4,5) \Rightarrow$ $\text{score} = 4 \cdot (-4) + 5 \cdot 5 - 9 = -16 + 25 - 9 = 0$

$\text{sigmoid}(0) = \ldots 0.5$ or 50%

# Lesson 2-13: Log-loss Error Function

Question 1: What function turn every number into a positive number?

That will be exp

example:

$e^{1} = 2.74...$

$e^{0} = 1$

$e^{-1} = 0.36...$

$e^{-99} = 1 \cdot 10^{-43}$ small but still positive!

Question 2: Coding Softmax

We have the scores: $Z_1, Z_2, \ldots, Z_n$

With softmax: $P(\text{class } i) = \frac{e^{Z_i}}{e^{Z_1} + \ldots + e^{Z_n}}$

In python:

- calculate the exp:
- return the result of the: exp, divided by the sum of all exp of our data.

```
def softmax(L):
    expL = np.exp(L)
    return np.divide(expL, expL.sum())
```