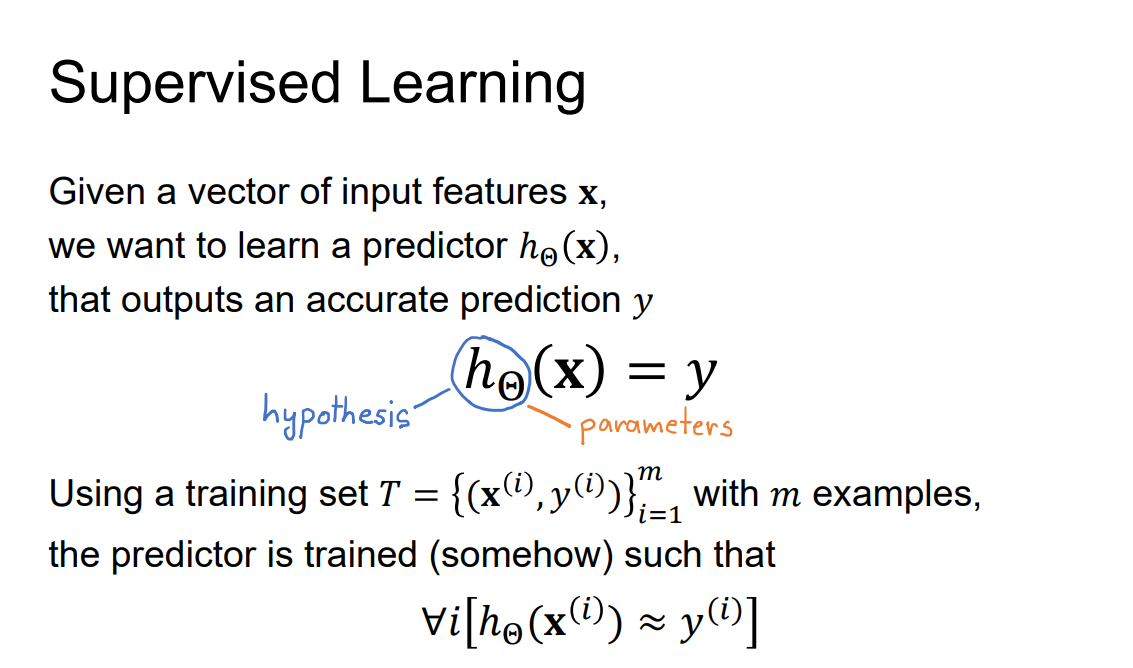
1A.

i)

0

ii)

hθ(x) = (θTx) <- X0 = 1

iii)

hθ(x) = g(θTx)

X0 = 1

g(z) = 1 / (1 + e-z)

1b.

i)

hθ(x) = g(θTx) # If this is linear regression g(x) = pass-through

Loss = 1/2m

1) Pass in the full dataset through the model

2) compute the loss

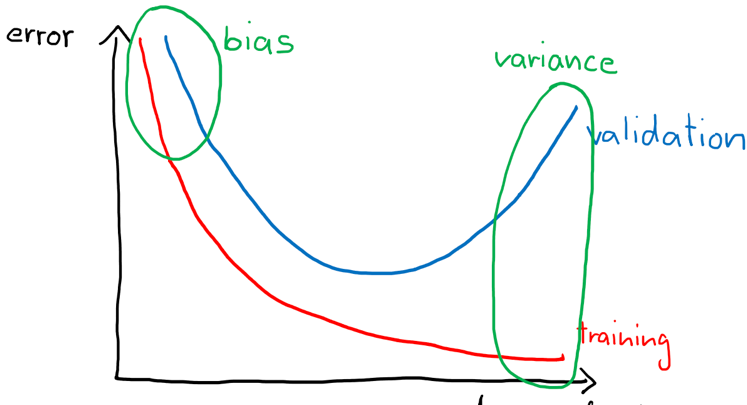
3) backpropagate and compute derivatives of loss wrt each parameter

4) set θ= θ-α \* derivative

ii)

Same but now for sampled minibatch instead of full dataset

III)



On a simple model the bias is high as the model hasn’t learnt any trends yet (underfitting). As the complexity increases, the model tends to fit more closely to the training data. As the model begins to overfit, the variance increases and while the error on the training data reduces, the error on the validation set increases.

IV)

* Add more features
* Decrease regularisation
* Increase model flexibility (“complexity” of the function it can represent)
* Cross validation?
* Train model for more epochs / longer?

C)

I) TP: 9, FP: 2, TN: 17 FN: 8

II) Precision: 9/11

Recall: 9/17

Specificity: 17/19

Dice: 18/28=9/14

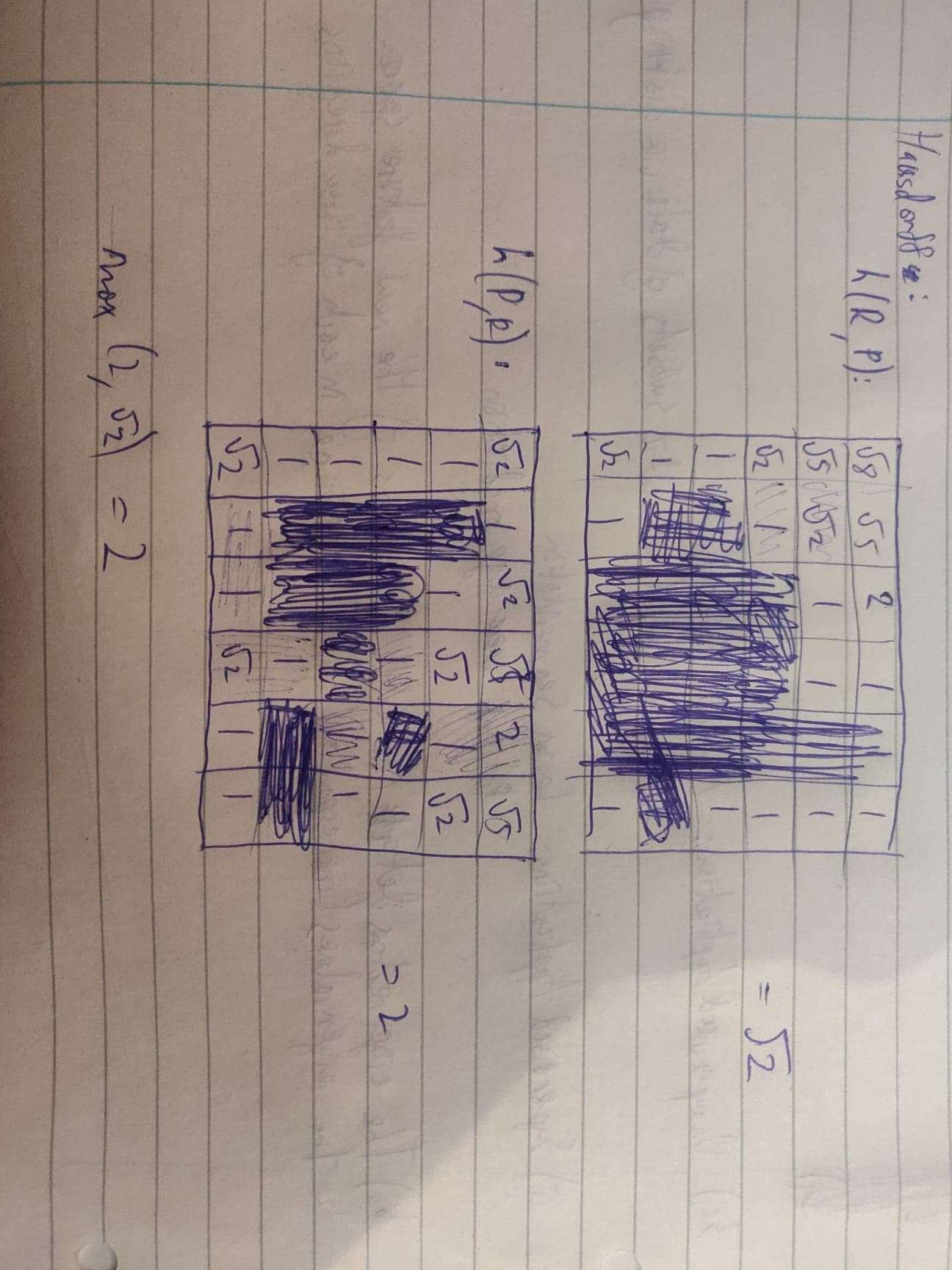
Hausdorff distance: 2 (the top 1 on the left image to the closest 1 on the right image) (confirmed by sample answers)

-----------------------------------------------------------------------------------

Here is my solution:

You take the Euclidean (can use chessboard IF STATED but they use Euclidean) distance of all squares away from each image.  
Then you overlap the other image onto it (the lightly scribbled out cells (only 2 in first image)) and take the largest value in the overlap

Repeat for the other image and then take the max of the 2 values.



III) No, as it judges the accuracy of background pixels. More background = better specificity

2

A

(Not examinable anymore)

B

I)

1x64x64 -> 2x60x60 -> 4x60x60 -> 8x30x30 -> 4x28x28 -> 2x14x14 -> 2x12x12

II)

1x128x32 -> 2x124x28 -> 4x124x28 -> 8x62x14 -> 4x60x12 -> 2x30x6 -> 2x28x4

C

I)

1\*112\*112 -> 2\*108\*108 -> 4\*108\*108 -> 8\*54\*54 -> 4\*52\*52 -> 2\*26\*26 -> 2\*24\*24

M = 24\*24

II)

C7 = Conv2d(in=2, out =16, kernel = 24, stride = 1, pad = 0)

C8 = Conv2d(in=16, out =10, kernel = 1, stride = 1, pad = 0)

D

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | | 0 | 0 | | 0 | | 0 |
| 0 | -3 | | 0 | 2 | | 0 | | 0 |
| 0 | 0 | | 0 | 0 | | 0 | | 0 |
| 0 | 4 | | 0 | -1 | | 0 | | 0 |
| 0 | 0 | | 0 | 0 | | 0 | | 0 |
| 0 | 0 | | 0 | 0 | | 0 | | 0 |
| -3 | | -0.5 | | | 2 | | 1 | | |
| 0.5 | | 0.5 | | | 0.5 | | 0.25 | | |
| 4 | | 1.5 | | | -1 | | -0.5 | | |
| 2 | | 0.75 | | | -0.5 | | -0.25 | | |

E

Collect/augment more data, weighted loss (penalize minority class more), different models for classes.

[8 Tactics to Combat Imbalanced Classes in Your Machine Learning Dataset (machinelearningmastery.com)](https://machinelearningmastery.com/tactics-to-combat-imbalanced-classes-in-your-machine-learning-dataset/)

3

A

i)

(They are all base 2 logarithms)

H1 = -5/9log (5/9) - 4/9log (4/9) = 0.991

H2 = -1/4log (1/4) - 3/4log (3/4) = 0.811

H3 = -2/5log (2/5) - 3/5log (3/5) = 0.971

H4 = 1

H5 = -2/3log (2/3) - 1/3log (1/3) = 0.918

ii)

IG3 = 0.971 - 2/5\*1 - 3/5\*0.918 = 0.02

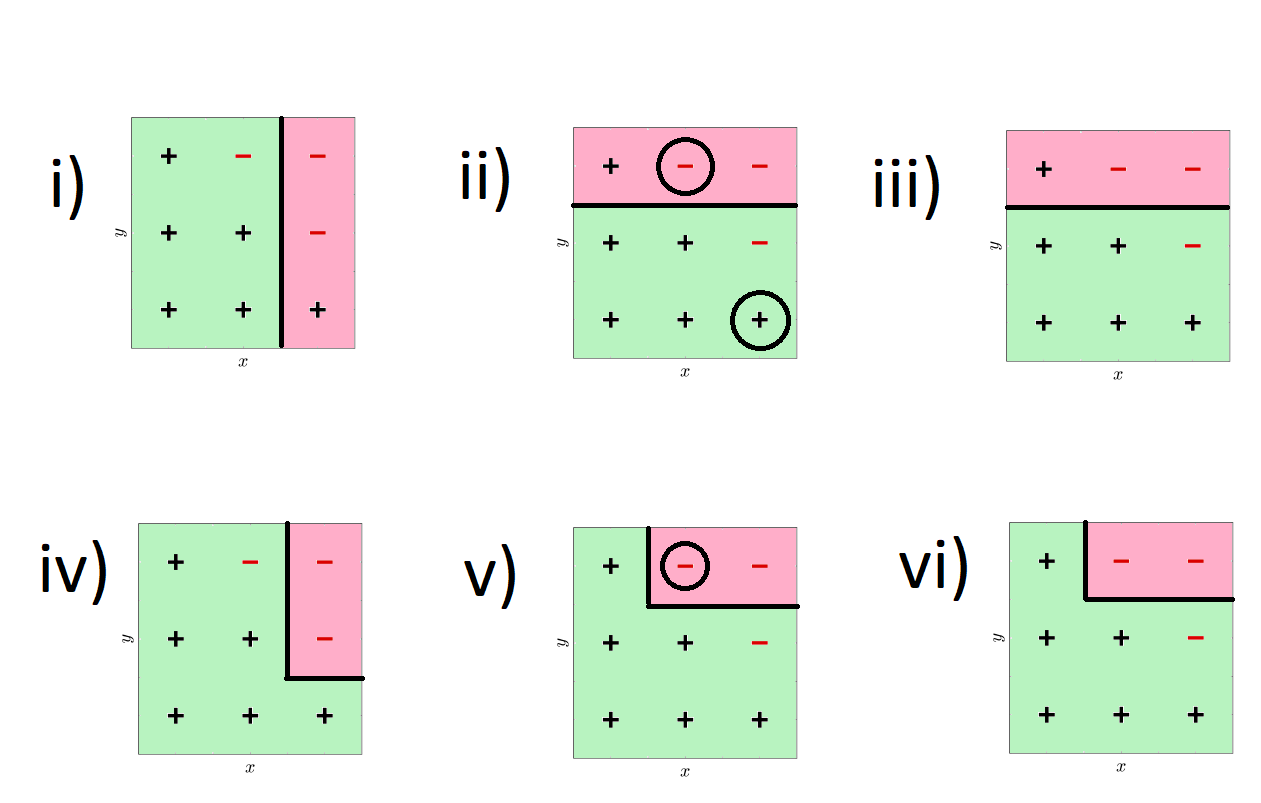
iii)

IG\_total = 0.991 - 2/9\*1 – 3/9\*0.918 - 4/9\*0.813 = 0.10

B

DEFINITIVE ANSWER WITH CALCULATED ALPHAS:

Please comment if you see any wrong calculations.

o

i)

ii)

iii)

As h2 has the higher weight, H will always go with the decision of h2.

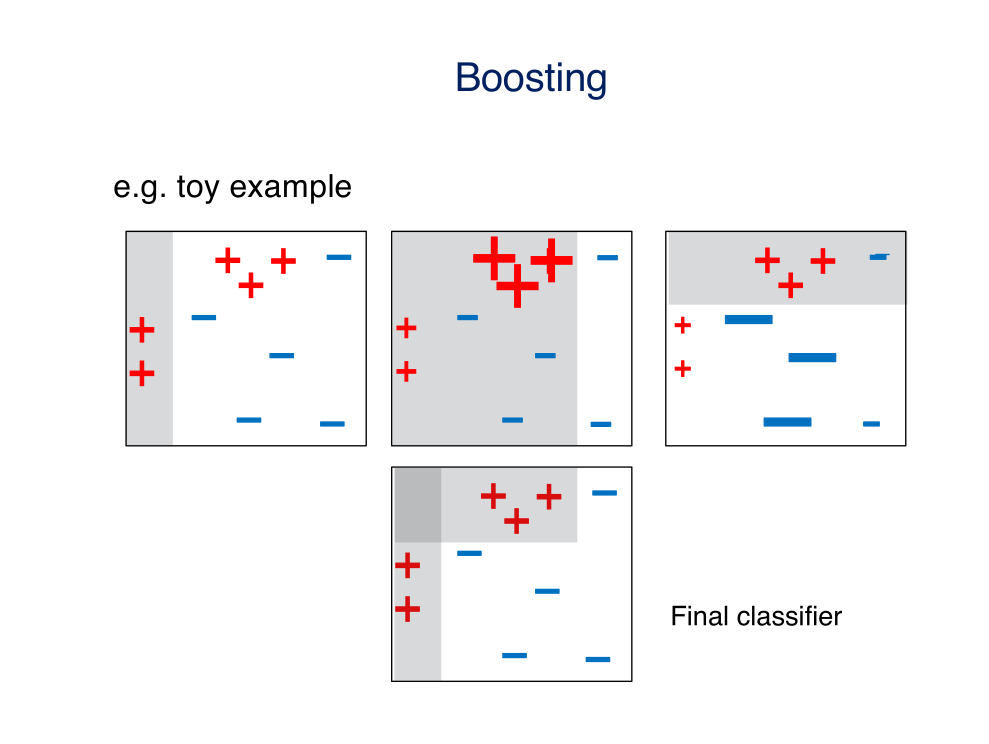
iv)

v)

vi)

As h2 has the higher weight, H will always go with the decision of h2.

H1 will either give 1 or –1, so will h2. If h2 gives –1, H will be negative no matter what, since h2 is multiplied by a higher number than h1. If h2 gives 1, H will be positive no matter what, since h2 is multiplied by a higher number than hq1.

(I disagree with iii) and vi) because looking at the examples in the notes the ‘final classifier’ is a combinatiton of all decision boundaries: )

However, the above uses 3 classifiers. From one of the 19-20 questions which asked how many classifiers are needed, it says 3 because 2 classifiers can’t be combined. The weight of one would always outweigh the other.

C

Decreases bias, mislabeled examples are upweighted each time reducing the mean error.

4

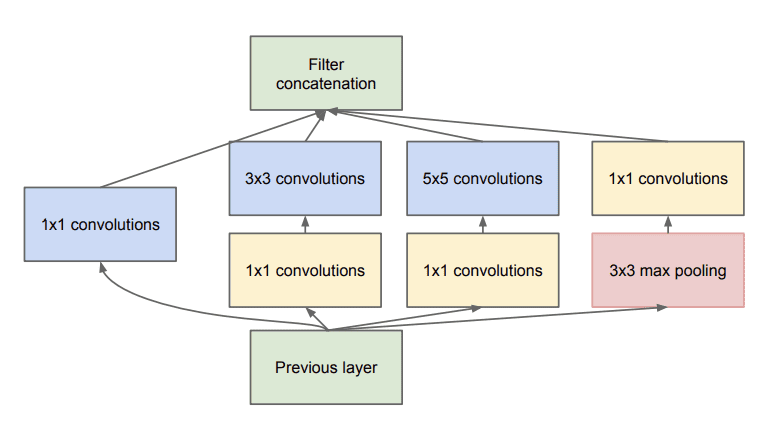
A

I) 6x5x5x8 + 8 biases

II) (6x20x20 + 1 bias) \* 8x10x10

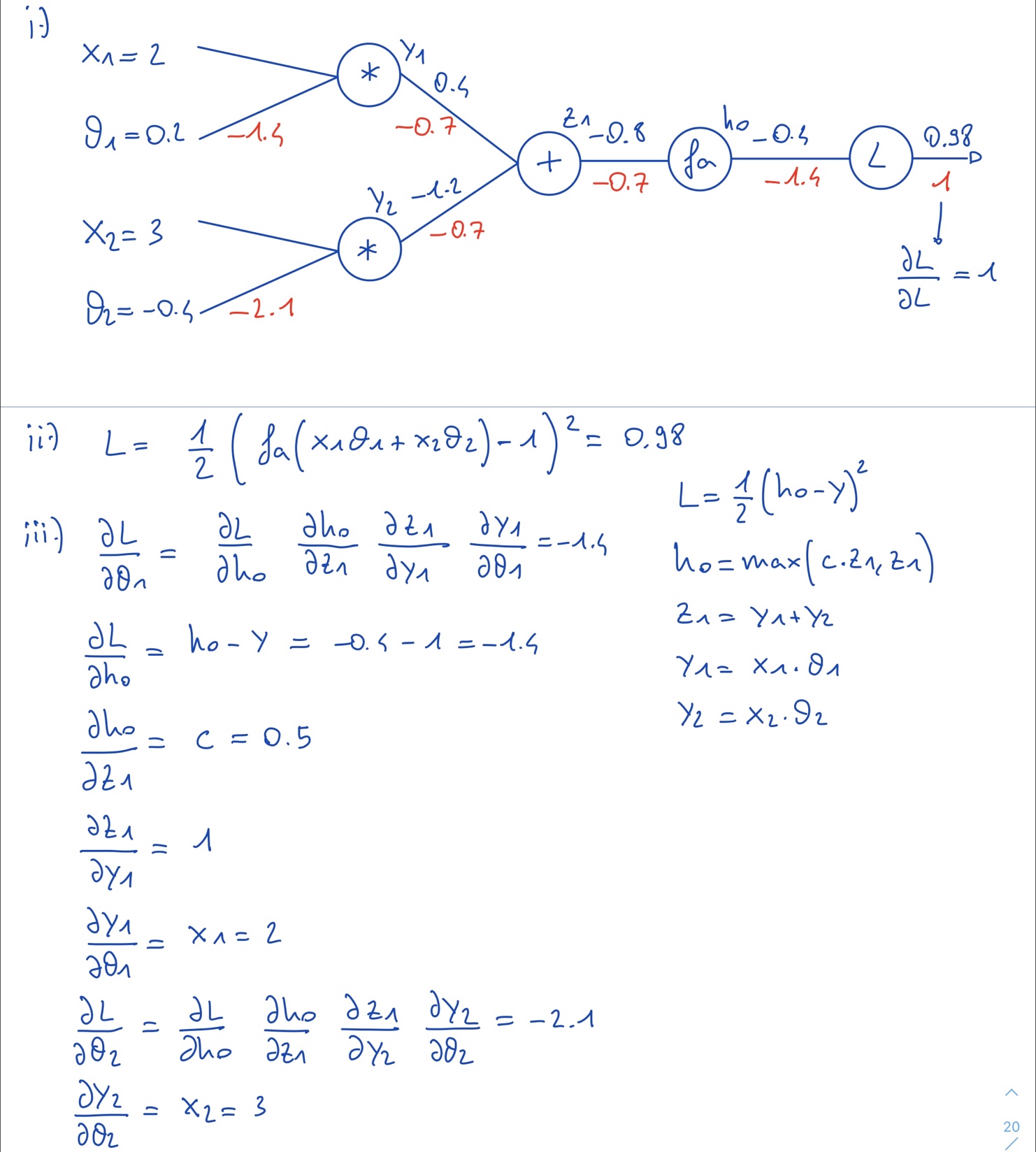
B

I)



II) Used to alter the number of feature maps in a layer without losing any information by effectively doing channel-wise pooling / acting like a perceptron. Decrease before expensive 3x3 and 5x5 convolutions to reduce computational complexity, increase after max-pooling. Because they are also followed by ReLU activation, they may also learn new features.

C



D

L1/L2 norm, gradient clipping, early stopping, dropout, batch-norm.