01) Neural Networks are a model of simulation of human nervous system. Nervous system is composed of neurons.

Perceptron is a model of single neuron. The perception algorithm is a two class (Binary) classification algorithm. Perceptoon In this algorithm we minic behaviour of neuron. We consider an activation score based on the synaphic connections and this activation score is then this activation sore is compared against some threshold value. If the activation score is larger than threshold value it gives output as +1 else negative -1.

Mathematical cotation Paper object $X = (x, \tau_2, t_3,, \tau_d)$ Weight $W = (w, w_1, w_3,, w_d)$ Activation score $a = (w, t_1) = W(x)$ output 1 If $a > 0$, and -1 if $a < 0$ Perceptron Train (D, Max Plea) 1. $wd = 0$, for all $d = 1 $	Harry Millians on special
Activation score a = Z w: 1: = NT & output 1 if a > 0, and -1 if a < 0 * Pseudocade Pexceptron Train (D, Max Ites) 1. wd ~ 0, for all d=1D !! initialize we 2. be ~ 0 !! initialize bial 3. for iter=1 Max Ites do 4. for all (any) E) do \$ a ~ Zd=1 wxd+b !! comple activation to G. if ya < 0 then 7. wd ~ wd + yxd for all d=1D !! up 8. b ~ b + y 9. end if 10. end for 11. end for 12. return wo, v1, wo, b	-
Activation score a = 2 w: 1: = NT & output 1 if a > 0, and -1 if a < 0 ** Pseudocade ** Perceptron Train (D, Max Ster) 1. wd ~ 0, for all d=1D !! initialize we 2. be ~ 0	
Activation score a = Z w: 1: = NT & output 1 if a > 0, and -1 if a < 0 * Pseudocade Pexceptron Train (D, Max Ites) 1. wd ~ 0, for all d=1D !! initialize we 2. be ~ 0 !! initialize bial 3. for iter=1 Max Ites do 4. for all (any) E) do \$ a ~ Zd=1 wxd+b !! comple activation to G. if ya < 0 then 7. wd ~ wd + yxd for all d=1D !! up 8. b ~ b + y 9. end if 10. end for 11. end for 12. return wo, v1, wo, b	Jor namentonic removings
Pseudocode Perceptron Train (D, Max Pter) 1. wd & O, for all d=1D	
Pseudocode Pexceptron Train (D, Max Pter) 1. wd = 0, for all d=1D !!initialize we 2. be = 0 3. for Her=1 Max Pter do 4. for all (ary) ED do 5. a = 2d=1 waxd + b !! compute activation to 6. if ya < 0 then 7. wd < wd + yxd for all d=1D !! up 8. b = b + y 9. end if 10. end for 11. end for 12. return wo, w1, w2, b	
Perceptron Train (D, Max ster) 1. wd = 0, for all d=1D !! initialize we on the start of	
Perceptron Train (D, Max Iter) 1. wd \(\infty \) o, for all d=1D !! initialize we get all d=1D !! initialize we get all (\(\text{ary} \)) \(\text{for all (ary)} \) \(\text{for all (ary)} \) \(\text{for all (ary)} \) \(\text{for all delD} \) !! compute activation to eram 6. If ya <0 then 7. \(\text{vd} \) = \(\text{vd} \) + \(\text{y} \) \(\text{for all delD} \) !! up 8. \(\text{b} \) \(\text{b} \) \(\text{b} \) \(\text{compute activation to eram 9. \(\text{end for } \) 11. \(\text{end for } \) 12. \(\text{return wo , \(\text{v}_1, \) wo , \(\text{b} \)	
Perceptron Train (D, Max Iter) 1. wd \(\infty \) o, for all d=1D !! initialize we get all d=1D !! initialize we get all (\(\text{ary} \)) \(\text{for all (ary)} \) \(\text{for all (ary)} \) \(\text{for all (ary)} \) \(\text{for all delD} \) !! compute activation to eram 6. If ya <0 then 7. \(\text{vd} \) = \(\text{vd} \) + \(\text{y} \) \(\text{for all delD} \) !! up 8. \(\text{b} \) \(\text{b} \) \(\text{b} \) \(\text{compute activation to eram 9. \(\text{end for } \) 11. \(\text{end for } \) 12. \(\text{return wo , \(\text{v}_1, \) wo , \(\text{b} \)	
2. be $\angle O$ 3. for iter=1 Maxsex do 4. for all (ary) \in) do 5. $q \leftarrow Zd-1$ which to $= Zd-1$ which for $= Zd-1$ which for $= Zd-1$ which then 6. if ya $\in O$ then 7. $= Zd-1$ which then 8. $= Zd-1$ which then 9. $= Zd-1$ which then 10. $= Zd-1$ which then 11. $= Zd-1$ which then $= Zd-1$ which	
2. be $\angle O$ 3. for iler = 1 Marster do 4. for all (ary) \in D do 5. $q \leftarrow Zd = U \times d + b$ (Compute activation to eran 6. if ya < 0 then 7. $U = U + U \times d = 0$ for all $d = 1 = 0$ If $U = 0$ 8. $U = U + U \times d = 0$ for all $d = 1 = 0$ If $U = 0$ 9. end if $U = 0$ 11. end for $U = 0$ 12. $U = 0$ 13. $U = 0$ 14. $U = 0$ 15. $U = 0$ 16. $U = 0$ 17. $U = 0$ 18. $U = 0$ 19. $U = 0$ 10. $U = 0$ 11. $U = 0$ 12. $U = 0$ 13. $U = 0$ 14. $U = 0$ 15. $U = 0$ 16. $U = 0$ 17. $U = 0$ 18. $U = 0$ 19. $U = 0$ 19. $U = 0$ 10. $U = 0$ 10. $U = 0$ 11. $U = 0$ 12. $U = 0$ 13. $U = 0$ 14. $U = 0$ 15. $U = 0$ 16. $U = 0$ 17. $U = 0$ 18. $U = 0$ 19. $U = 0$ 19. $U = 0$ 10. $U = 0$ 10. $U = 0$ 11. $U = 0$ 12. $U = 0$ 13. $U = 0$ 14. $U = 0$ 15. $U = 0$ 16. $U = 0$ 17. $U = 0$ 18. $U = 0$ 19. $U = 0$ 19. $U = 0$ 19. $U = 0$ 10. $U = 0$ 10. $U = 0$ 10. $U = 0$ 11. $U = 0$ 12. $U = 0$ 13. $U = 0$ 14. $U = 0$ 15. $U = 0$ 16. $U = 0$ 17. $U = 0$ 18. $U = 0$ 19.	
2. be $\angle O$ 3. for iter=1 Maxidex do 4. for all (ary) \in) do 5. $a \leftarrow Zd=1$ which then 6. if ya $\in O$ then 7. $ed \leftarrow ed + yd=1$ for all $d=1$) 8. $ed \leftarrow ed + yd=1$ for all $d=1$) 9. $ed \in ed$ if 10. $ed \in ed$ for 11. $ed \in ed$ 12. $ed \in ed$ 11. $ed \in ed$	ights
3. for Harzl Marker do 4. for all (dry) E) do 5. q ~ Eda ward + b // Compute activation to G. If ya < 0 then 7. vd ~ vd + yad for all del D // up 8. b ~ b + y 9. end if 10. end for 11. end for 12. return wo, vi, wo, b	U
4. for all (ary) ED do S a Zd-1 ward + b // Comprete activation to G. if ya < 0 then 7. wd < wd + yad for all del D // up 8. b < b + y 9. end if 10. end for 12. return wo, w, , , , , , , wo, b	
S q ~ Zd-1 which to (Compute activation to eram G. If ya < 0 then 7.	
6. If ya < 0 then 7. Ud < Ud + yad for all del D Il up 8. b < b + y 9. end if 10. end for 11. end for 12. return wo, w, wo, b	· th
2. 6d = 6d + yad for all del D // Up 8. 6 + y 9. end if 10. end for 12. return wo, v1, wo, b	ple
8. be b + y 9. end if 10. end for 11. end for 12. return wo, v, v, v, wo, b	del
9. end for 11. end for 12. return wo, w, w, wo, b	7
10. end for 11. end for 12. return wo, v,,,,, ws, b	
11. end for 12. return wo, v, ws, b	
12. neturn wo, v,,,,, ws, b	
12. octum wo, v,, wo, b	
Para otoro fort (and us	
regardent les control con les controls con les controls control controls controls control contro	
1. a - Eder wod 3d +b // compute activation for	KIL
2. return SEGN (a)	