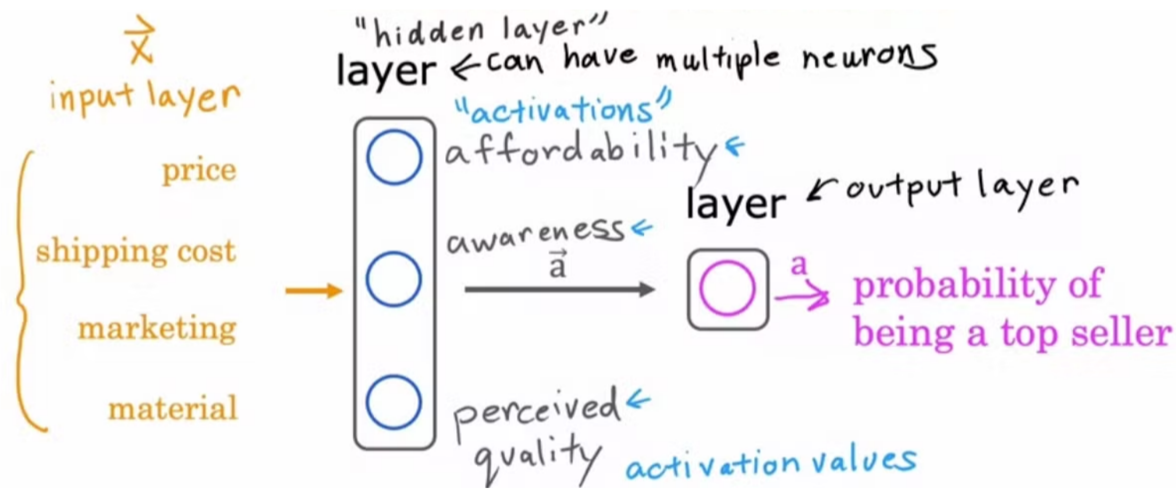
Advanced Learning Algorithms

Origins: algorithms that try to mimic the brain

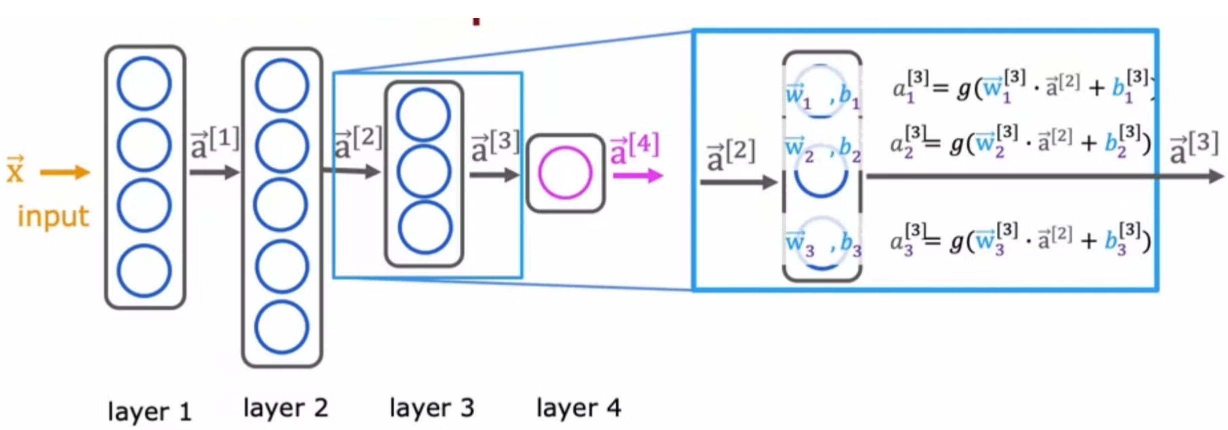
Ressurge from around 2005 under the term *deep learning*, as the amount of data rapidly increased. The rise of GPUs was also a major force.

Applications: speech recognition → computer vision → NLP ...

Example: predict if t-shirt will become a top-seller



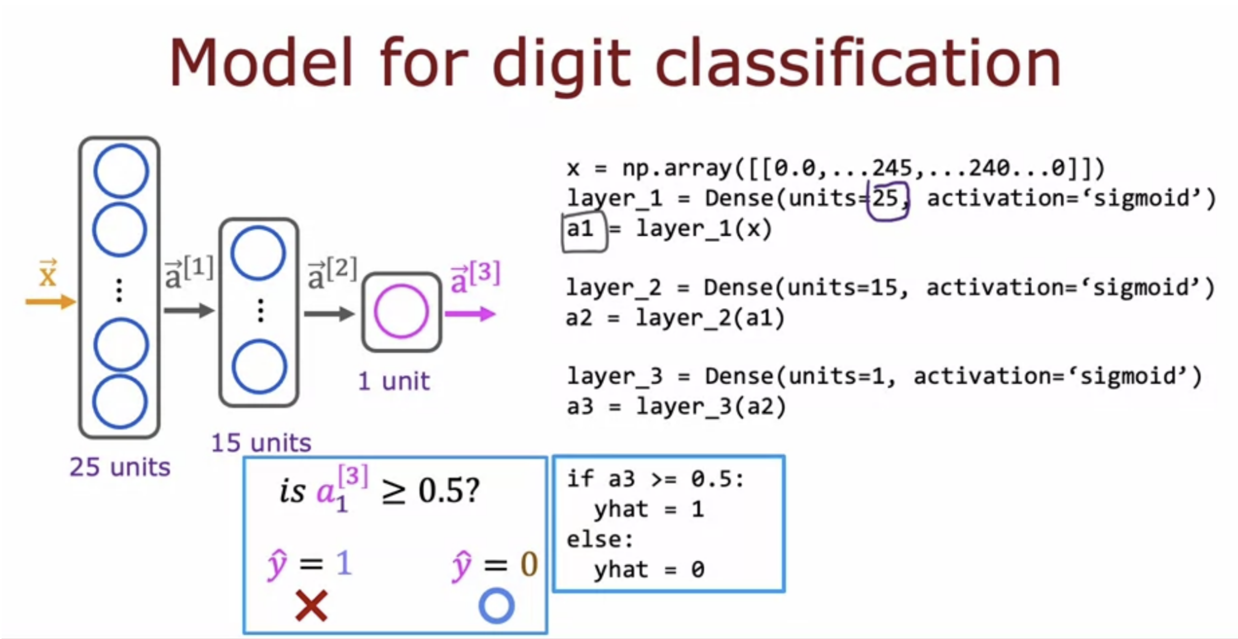
It's like a version of logistic regression that learns its own features layer by layer. That replaces manual feature engineering.

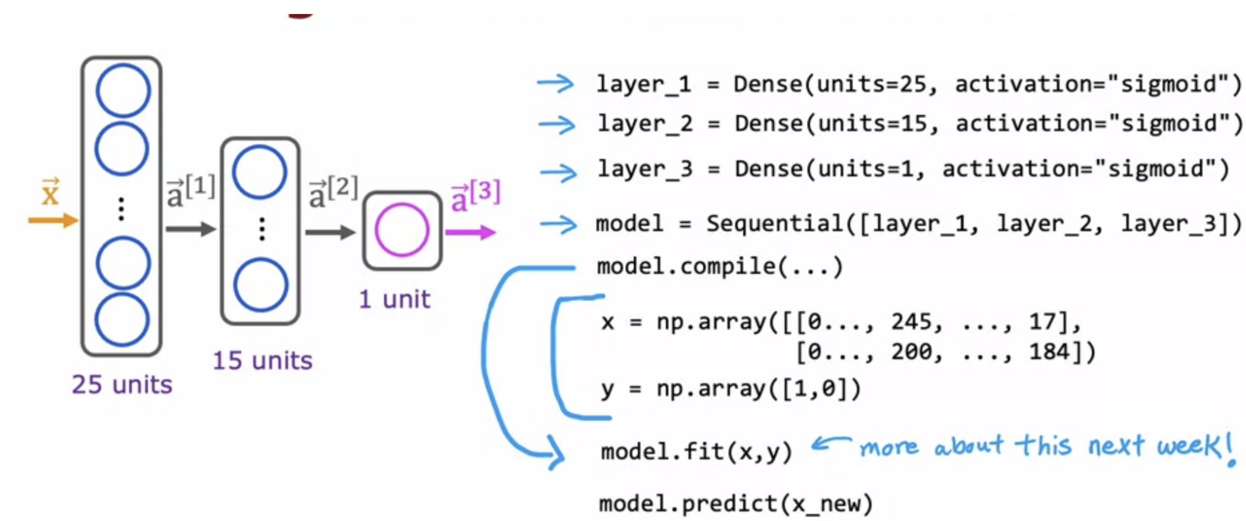


The activation function g could be the sigmoid function or other functions.

Forward propagation to make a prediction (inference).

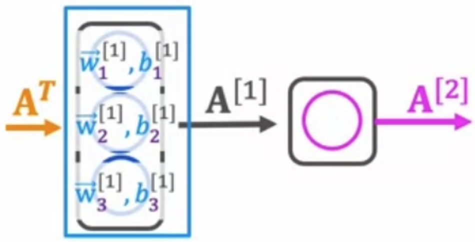
TensorFlow





Vectorized implementation makes neural networks run much faster!

Dense layer vectorized



$\mathbf{A}^T = [200 \quad 17]$
1 x 2

$\mathbf{W} = \begin{bmatrix} \boxed{\begin{matrix} 1 \\ -2 \end{matrix}} & \boxed{\begin{matrix} -3 \\ 4 \end{matrix}} & \boxed{\begin{matrix} 5 \\ -6 \end{matrix}} \end{bmatrix}$
2 x 3

$\mathbf{B} = [-1 \quad 1 \quad 2]$
1 x 3

$\mathbf{Z} = \mathbf{A}^T\mathbf{W} + \mathbf{B}$

$[\boxed{165} \quad \boxed{-531} \quad \boxed{900}]$

$z_1^{[1]} \quad z_2^{[1]} \quad z_3^{[1]}$

$\mathbf{A} = g(\mathbf{Z})$

$[1 \quad 0 \quad 1]$

A
```
AT = np.array([[200, 17]])
W = np.array([[1, -3, 5],
              [-2, 4, -6]])
b = np.array([[-1, 1, 2]])
```
a_in
```
def dense(AT,W,b):
  z = np.matmul(AT,W) + b
  a_out = g(z)
  return a_out
```
```
[[1,0,1]]
```