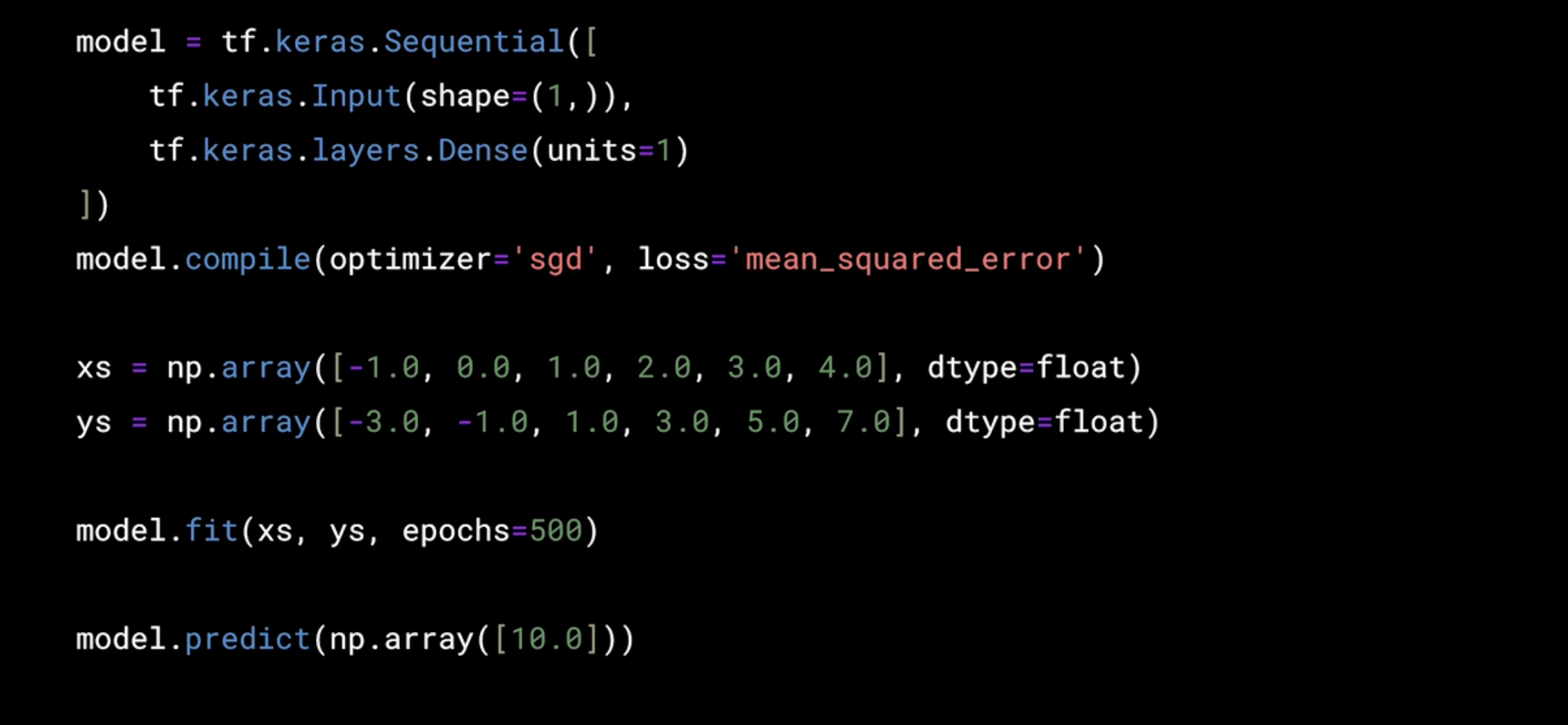
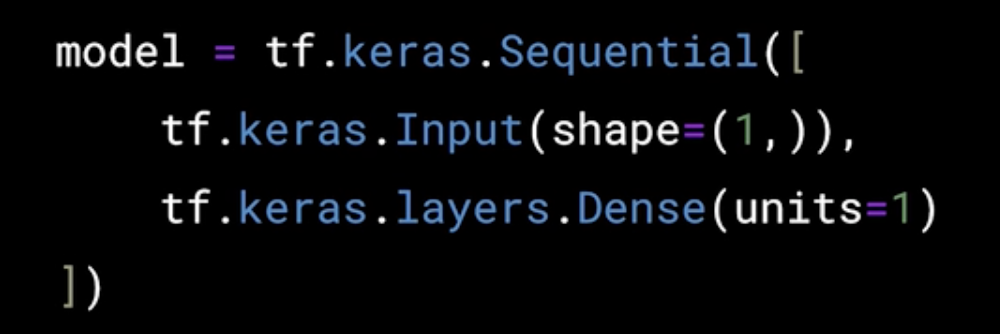


Formula here : 2X - 1

pip install --upgrade keras

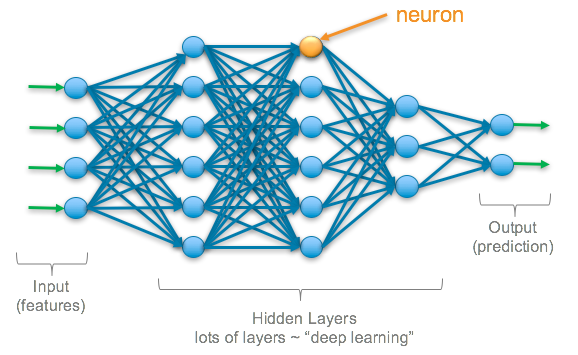




This is a neural network :

tf.keras.Sequential([...])

- This creates a Sequential model, meaning layers are stacked one after another in a linear fashion.



tf.keras.layers.Dense(units=1) means:

- You’re adding 1 neuron (just one circle) in that layer.

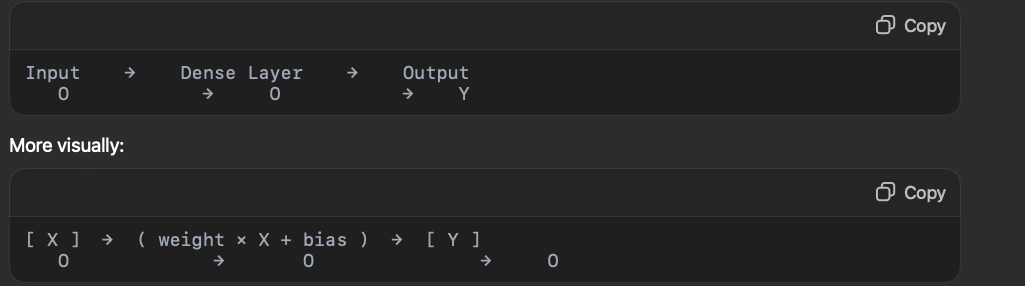
- Since it’s a Dense layer, this one neuron is fully connected to all previous inputs.

**model = tf.keras.Sequential([**

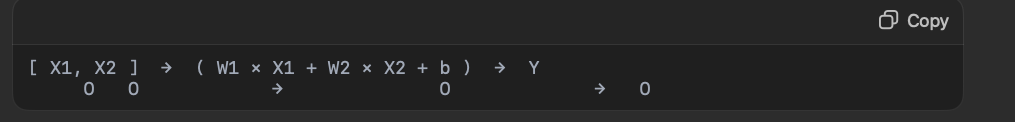
**tf.keras.Input(shape=(1,)), # One input feature → 1 green arrow**

**tf.keras.layers.Dense(units=1) # One output neuron → 1 blue circle**

**])**



If tf.keras.Input(shape=(2,)),





“Hey model, when you’re learning, use this method to improve (sgd) and use this formula (mean\_squared\_error) to see how wrong you are.”

optimizer='sgd' (Stochastic Gradient Descent)

- Tells the model how to learn.

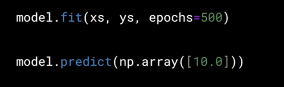
- SGD = one of the simplest ways: guess → check → adjust.

- Think of it like trial and error, but smart.

loss='mean\_squared\_error'

- Tells the model how to measure mistakes.

- If the model guesses 4 but the answer is 6, the error is 2 — this squares it to punish big mistakes more.



model.fit(xs, ys, epochs=500)

- Train the model using the data xs and ys

- Do it 500 times (called “epochs”) to learn the pattern well

- After training, the model will figure out the best math formula (like Y = 2X - 1)

model.predict(np.array([10.0]))

- Ask the model: “If X is 10, what’s Y?”

- The model uses what it learned to predict Y