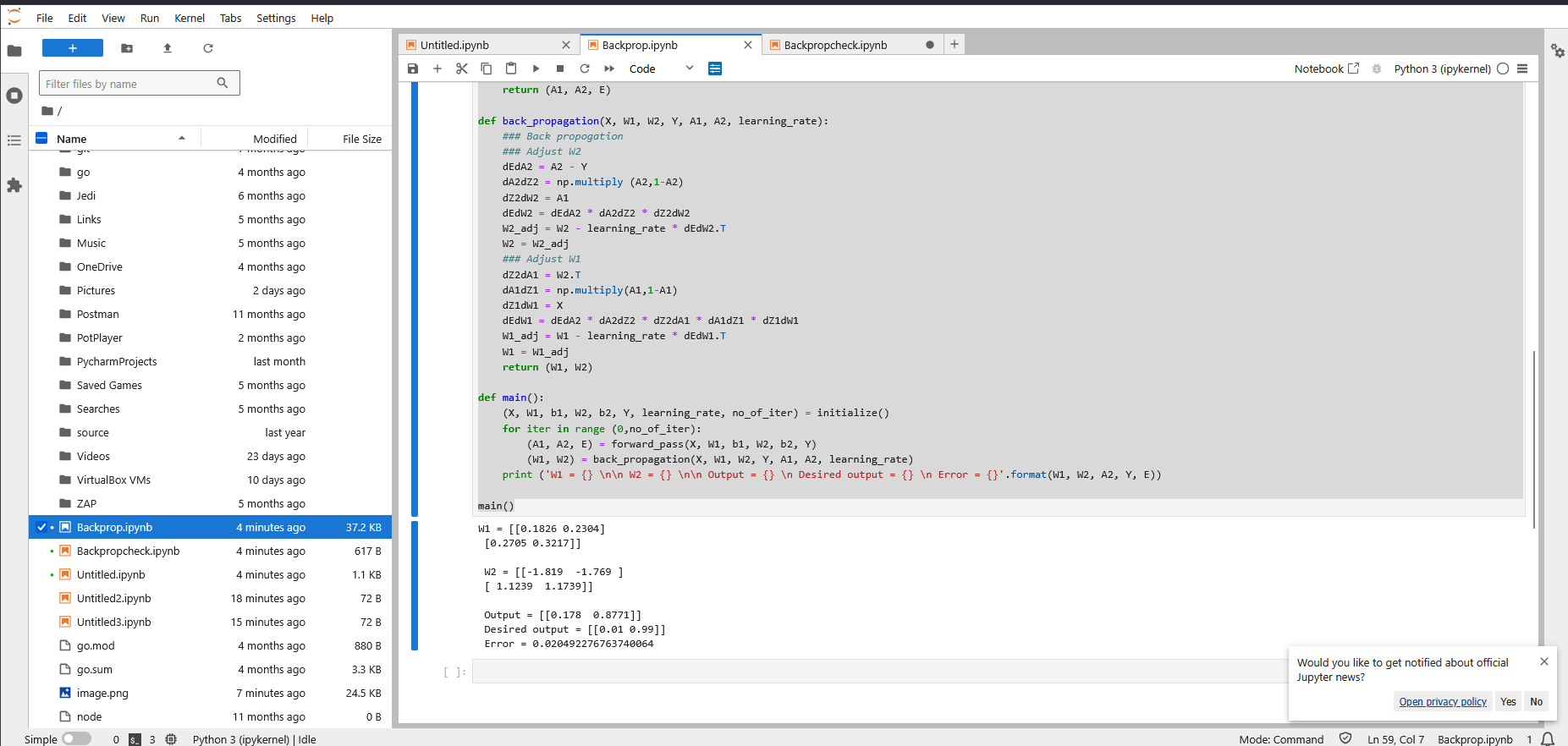
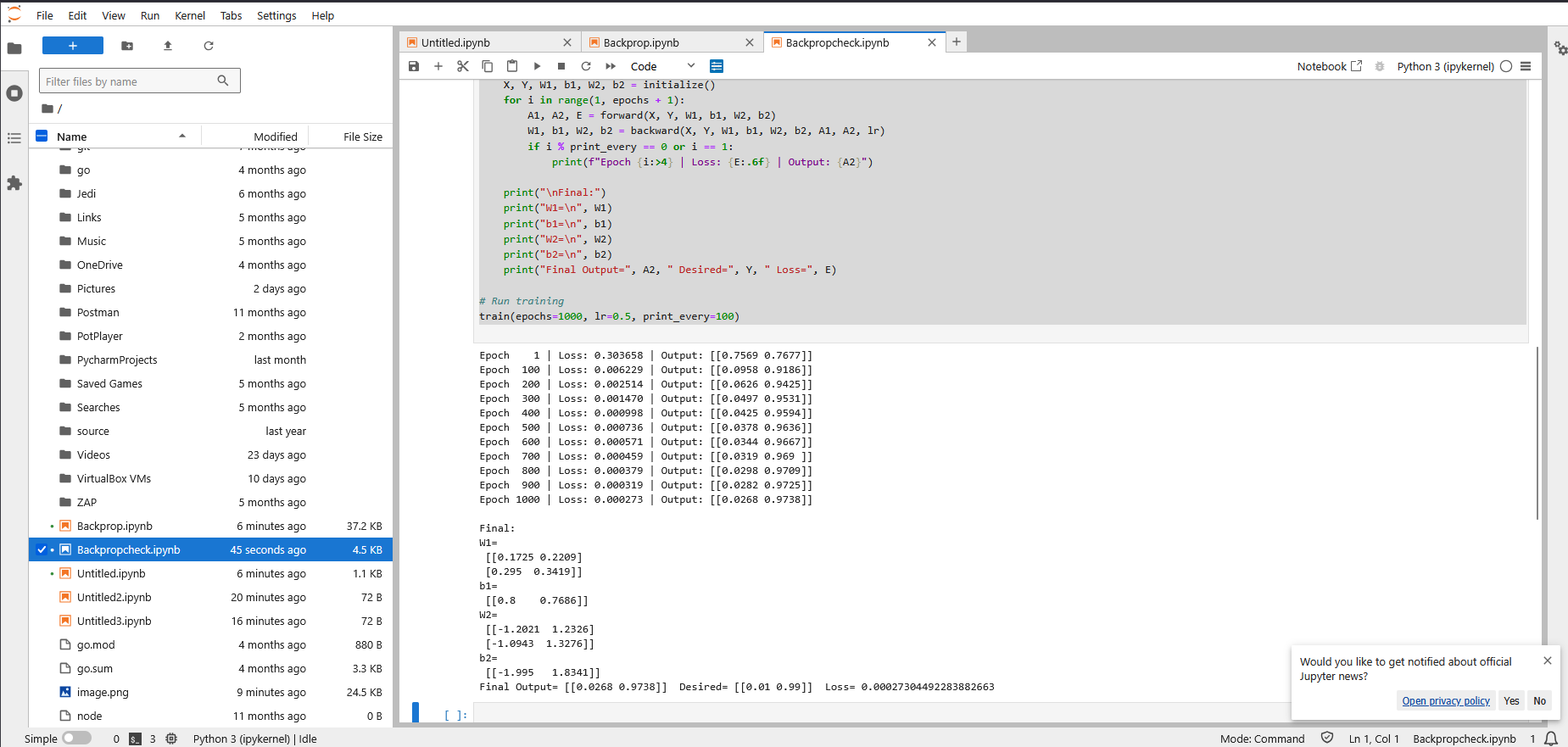
**Deep Learning Labs**

1. **Increase the number of iterations (epochs) and see whether it improves the prediction accuracy.**

Before

****

After changing number of epochs



**Increase epochs** (e.g., 100 → 1,000 → 10,000) and observe:

* + The **loss** should monotonically decrease (with this tiny network and learning rate 0.5 it will converge quickly).
  + The **output** should approach the target [0.01, 0.99].

1. **Add the following text cell and the code cell to the notebook and run it again.**

**Forward propagation → *make predictions* .**Pass the input data through the network to compute the outputs and the loss.

**Backward propagation → *learn from mistakes*.**Figure out how to change the weights and biases to make the predictions better.

**Cross-entropy loss** is a way to measure how well a classification model’s predicted probability distribution matches the actual labels

1. **What happens when the number of hidden nodes increase?**

### **The positive side**

* **More capacity to learn complex patterns**  
  More neurons mean the network can represent more complex decision boundaries.
* **Better fit for complex data**  
  If your dataset has nonlinear relationships, more hidden units can capture those patterns.

### **The risks / drawbacks**

* **Overfitting**  
  Too many neurons can memorize the training data, performing poorly on unseen data.
* **Longer training time**  
  More parameters → more computations per epoch.
* **Higher memory use**  
  Each extra neuron adds weights and biases to store and update.
* **Vanishing/exploding gradients** (in deep nets)  
  With large hidden layers, numerical instability can be worse if not managed.

### **Intuition with an example**

* **Small hidden layer** → might underfit (too simple, can’t capture the data’s complexity).
* **Moderate hidden layer** → good generalization.
* **Huge hidden layer** → might overfit (memorize instead of generalizing).

1. **Can you explain the pattern of the accuracy when the hidden nodes increase?**

**Typical pattern of accuracy change**

1. **Small number of hidden nodes**
   * The network is **underpowered** → can’t capture the complexity of the data.
   * Accuracy is low because it **underfits** (high bias).
2. **Moderate number of hidden nodes**
   * The network has enough capacity to learn relevant patterns.
   * Accuracy on training and test data **increases**.
   * This is usually the **sweet spot** for generalization.
3. **Too many hidden nodes**
   * Training accuracy may approach **100%** (model memorizes training data).
   * **Test accuracy may drop** due to **overfitting** (high variance).
   * The model is fitting noise and irrelevant details in the training set.

**Why this happens**

* **Capacity vs. generalization**: More hidden nodes = more capacity → better training fit but higher risk of overfitting.
* **Bias-variance trade-off**:
  + Low hidden nodes → high bias (underfitting).
  + High hidden nodes → high variance (overfitting)

**Visual pattern**

If you plotted **test accuracy** vs. **number of hidden nodes**, you’d often see:

Accuracy

^

| /\ Test accuracy peaks, then falls

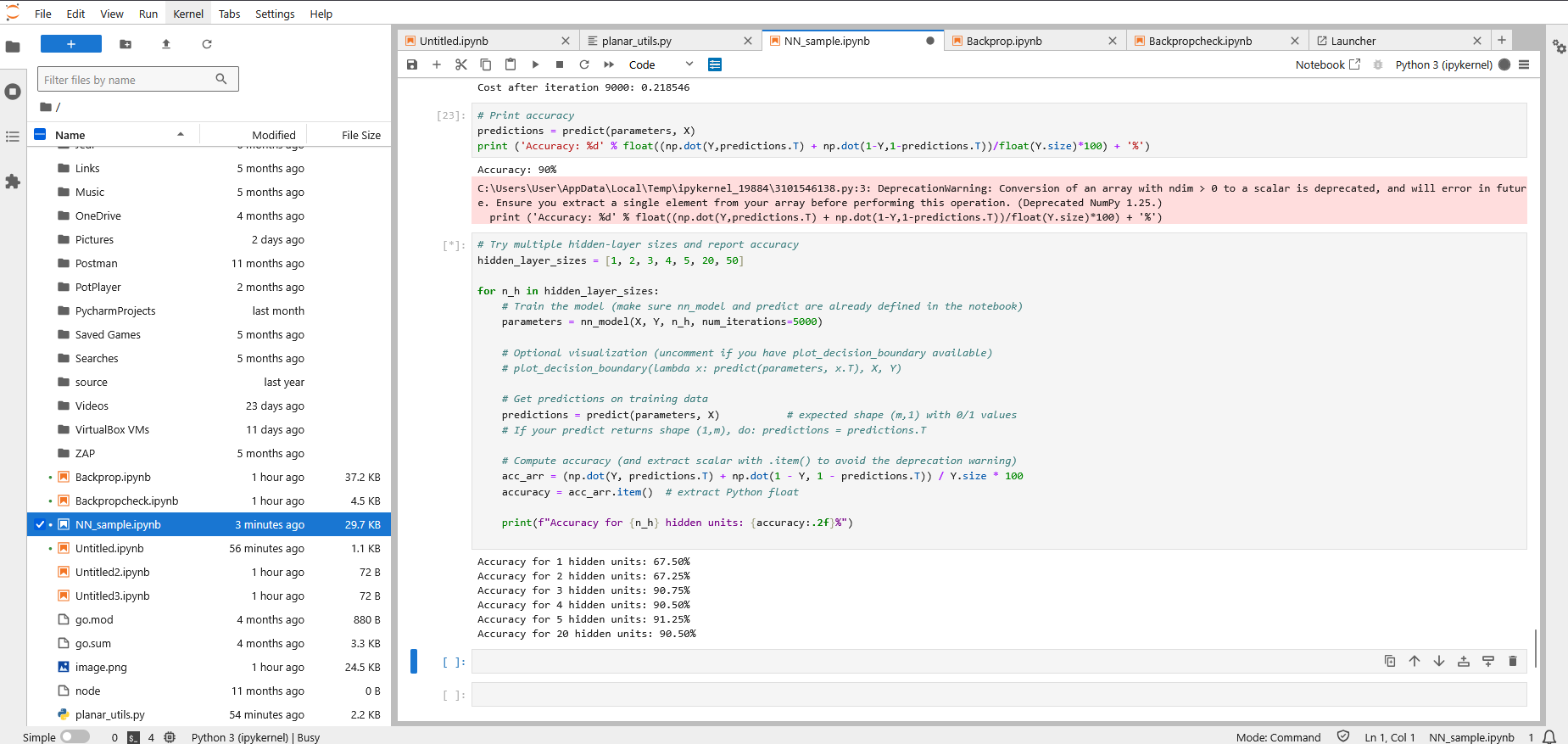
| / \

| / \

|\_\_\_\_/ \\_\_\_\_\_\_\_\_\_\_\_\_\_

Hidden nodes →

* Training accuracy: rises and stays high as hidden nodes increase.
* Test accuracy: rises, peaks, then declines.

****