Iteration 0: Cost nan Iteration 1000: Cost nan Iteration 2000: Cost nan Iteration 3000: Cost nan getting such values

ChatGPT

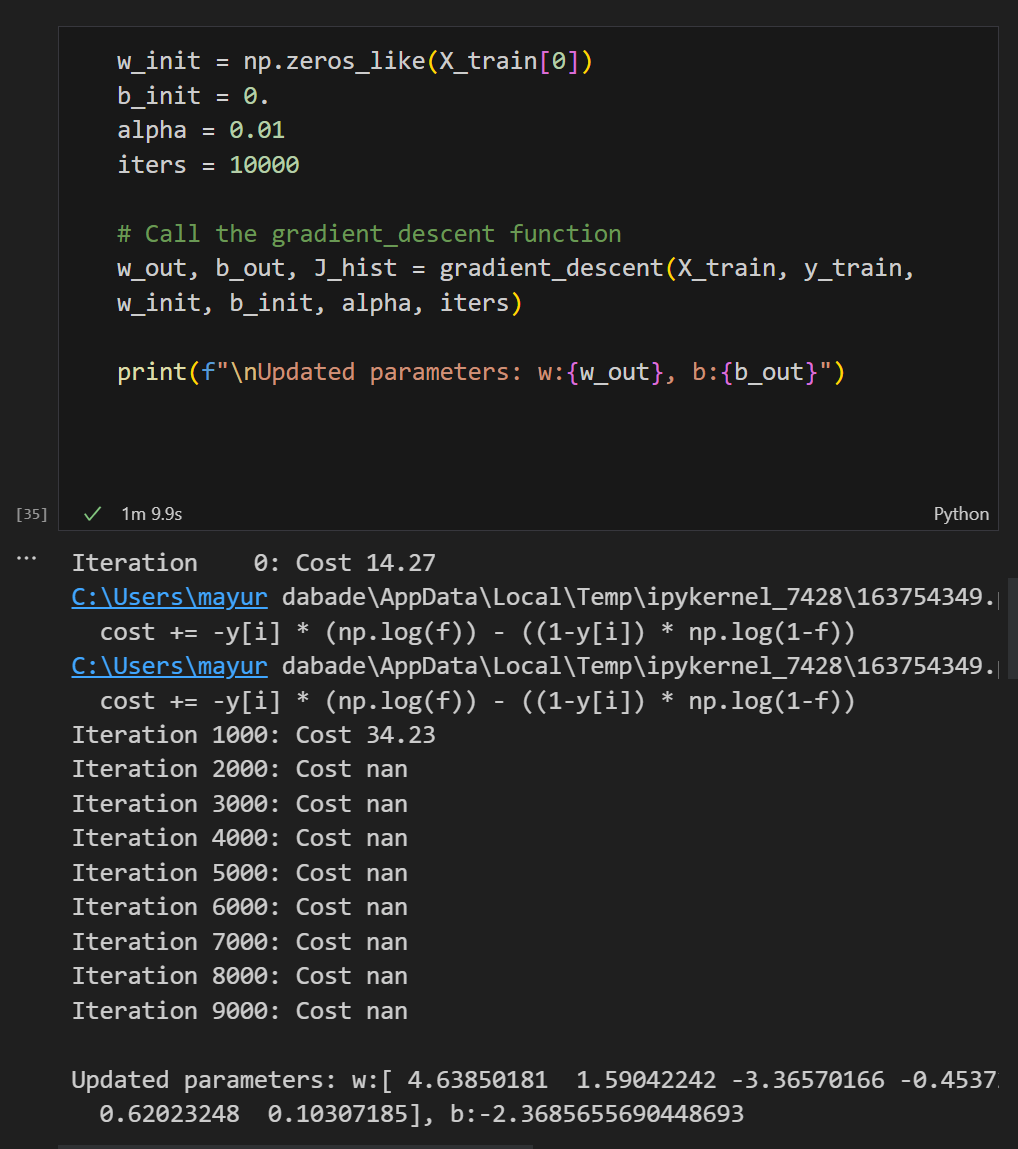
The issue you're facing with NaN (Not-a-Number) values in your cost function during gradient descent typically indicates a problem with your implementation. NaN values can arise due to various reasons such as incorrect calculations, division by zero, or improper handling of the sigmoid function.

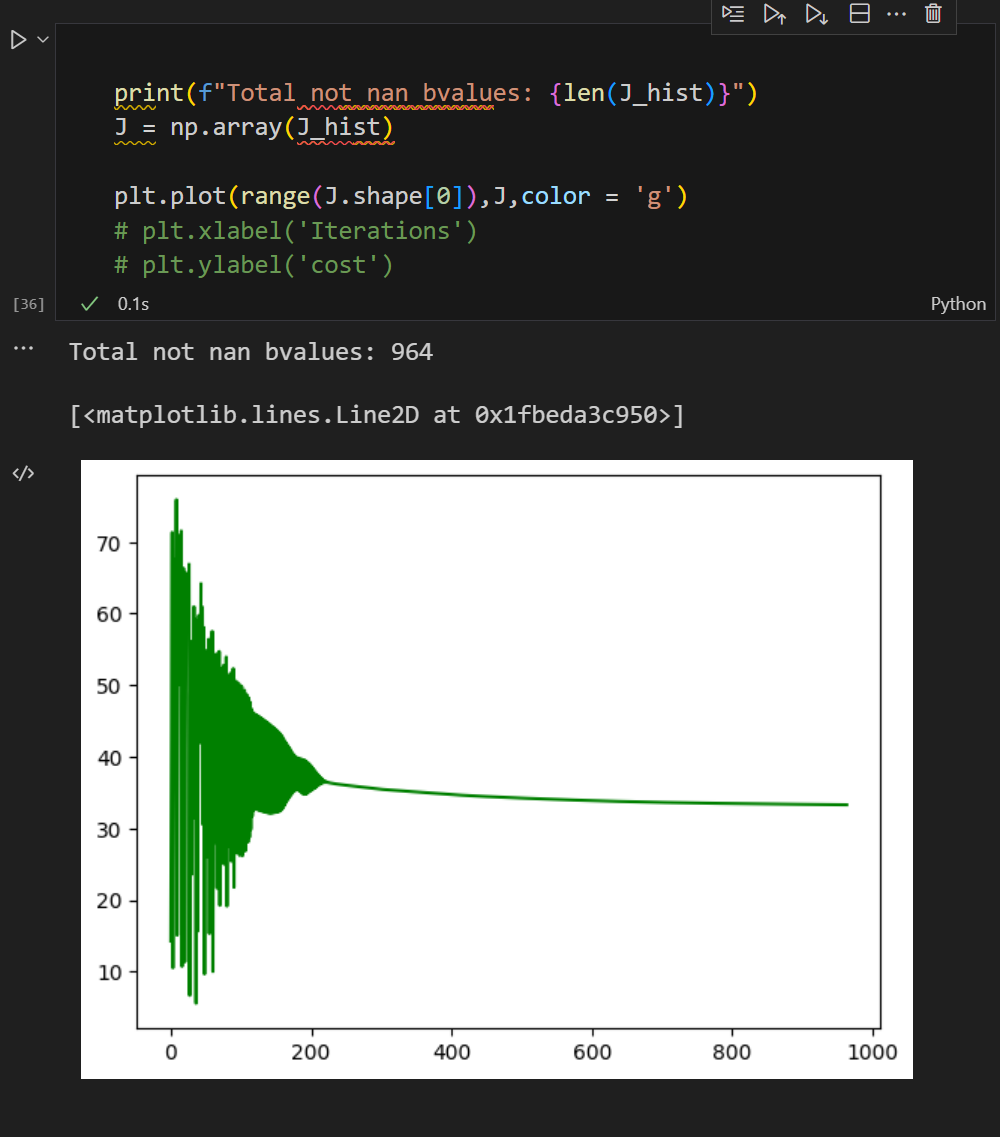
Here are a few things you can check and troubleshoot:

1. **Sigmoid Function:** Ensure that your sigmoid function is correctly implemented and handles values within a valid range. Make sure it doesn't return NaN or infinity for any input.
2. **Data Preprocessing:** Check if your input data **X** has been properly preprocessed and scaled. Incorrect preprocessing can lead to numerical instability and NaN values.
3. **Learning Rate (Alpha):** If the learning rate (**alpha**) is too large, it can cause your gradient descent to diverge, resulting in NaN values. Try reducing the learning rate and see if the issue persists.
4. **Initial Parameters:** Make sure that your initial weights **w** and bias **b** are reasonable values and not too large, which could lead to numerical instability.
5. **Check for Division by Zero:** Look for any instances where you might be dividing by zero. This can often cause NaN values.
6. **Overflow or Underflow:** Be cautious of numerical overflow or underflow issues that can occur with large or small values in exponentials or other calculations.
7. **Data Issues:** Check if your training data contains any NaN or missing values that could propagate through your calculations.
8. **Print Debugging:** Add print statements to your code to print intermediate values and identify where the NaN values might be coming from.
9. **Review the Sigmoid Derivative:** If you are using the derivative of the sigmoid function in your code, make sure that it is implemented correctly and not causing any issues.

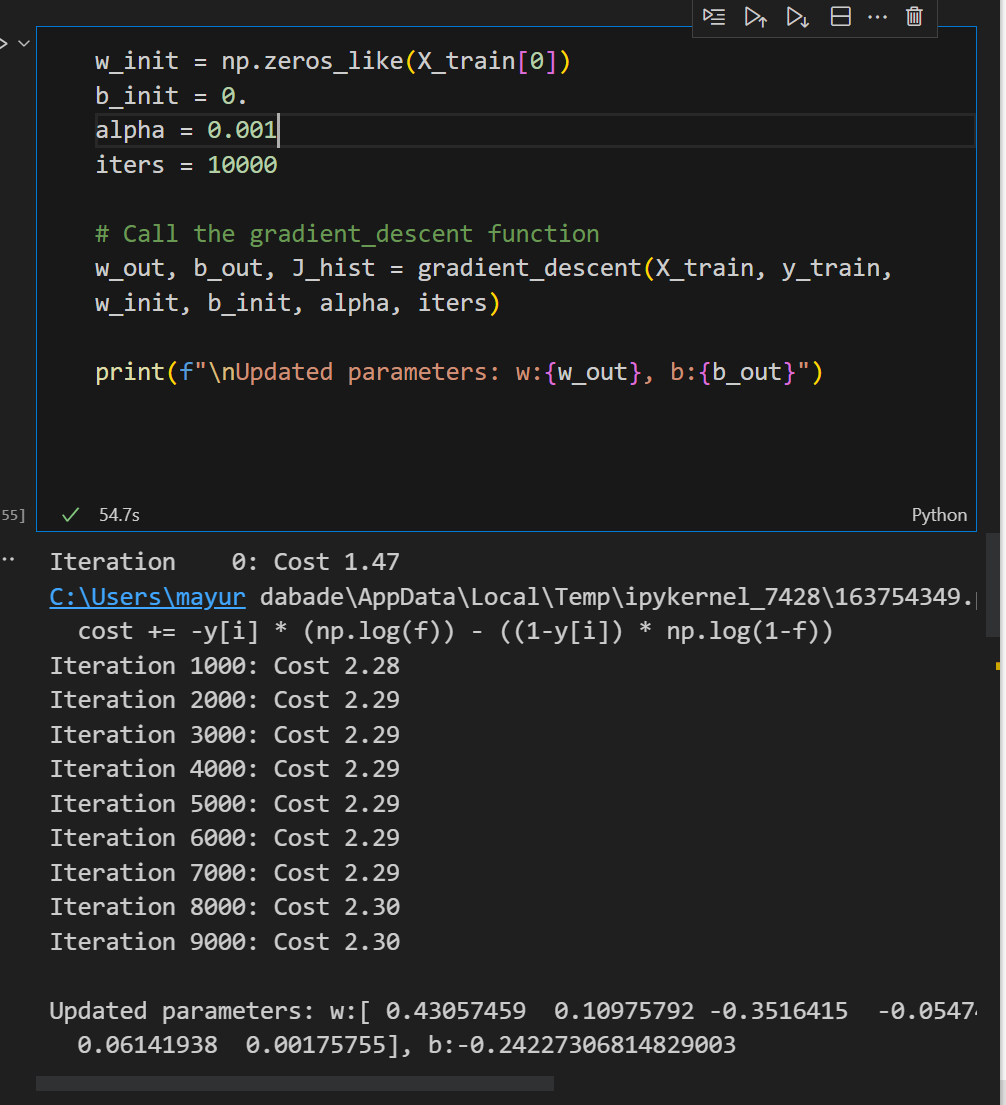
By carefully checking and addressing these potential issues, you should be able to identify and resolve the cause of the NaN values in your cost function during gradient descent.

**Learning rate (alpha) = 0.01**





**Learning rate (alpha) = 0.001**





**Learning rate (alpha) = 0.0001**

