Pseudo Code

```
# Input Data
Initialize age array with propellant ages
Initialize temperature array with storage temperatures
Initialize pass_fail array with pass/fail outcomes

# Data Preparation
Create feature matrix X by combining age and temperature arrays,
adding a column of ones for bias
Set target variable y to pass_fail array

# Define Functions
Function sigmoid(z):
    Return 1 / (1 + exp(-z))

Function cost_function(w):
    Compute linear combination z = X * w
    Compute predicted probabilities h = sigmoid(z)
    Clip h to avoid extreme values
    Calculate cost using cross-entropy loss formula
    Return cost

Function gradient_function(w):
    Compute linear combination z = X * w
    Compute predicted probabilities h = sigmoid(z)
    Calculate gradient using the derivative of the cost function
    Return gradient

Function line_search(cost_function, gradient_function, w):
    Initialize step size
    For each trial:
        Update weights using gradient descent
        Check if cost_function with updated weights is improved
        If improved, return step size
        Otherwise, reduce step size and try again
    Return optimal step size

Function gradient_descent(max_iter, epsilon):
    Initialize weights to zeros
    For each iteration up to max_iter:
        Compute gradient
```

```
        Check if gradient norm is below epsilon
        Perform line search to find optimal step size
        Update weights using gradient descent
        Print progress every 1000 iterations
    Return final weights, minimum cost, and number of iterations

# Run Optimization
Execute gradient descent to get optimal weights
Print optimal weights, minimum cost, and number of iterations

# Visualization
Generate grid of age and temperature values
Compute probabilities for grid using sigmoid function
Create a colormap from red (fail) to green (pass)
Plot filled contour of probabilities
Overlay scatter plot of individual samples
Add color bar, labels, legend, and title
Display plot
```