notes_07

December 4, 2024

1 Previous Class Definitions

The previously defined Layer_Dense, Activation_ReLU, and Activation_Softmax

[5]:
```
# imports
import matplotlib.pyplot as plt
import numpy as np
import nnfs
from nnfs.datasets import spiral_data, vertical_data
nnfs.init()
```

[6]:
```
class Layer_Dense:
    def __init__(self, n_inputs, n_neurons):
        # Initialize the weights and biases
        self.weights = 0.01 * np.random.randn(n_inputs, n_neurons)  # Normal␣
 ↪distribution of weights
        self.biases = np.zeros((1, n_neurons))

    def forward(self, inputs):
        # Calculate the output values from inputs, weights, and biases
        self.output = np.dot(inputs, self.weights) + self.biases        #␣
 ↪Weights are already transposed

class Activation_ReLU:
    def forward(self, inputs):
        self.output = np.maximum(0, inputs)

class Activation_Softmax:
    def forward(self, inputs):
        # Get the unnormalized probabilities
        # Subtract max from the row to prevent larger numbers
        exp_values = np.exp(inputs - np.max(inputs, axis=1, keepdims=True))

        # Normalize the probabilities with element wise division
        probabilities = exp_values / np.sum(exp_values, axis=1,keepdims=True)
        self.output = probabilities
```

2 Forward Pass with No Loss Consideration

2 input neural network with 2 layers of 3 neurons each. ReLU activation in the first layer with Softmax in the second layer to normalize the outputs.

[4]:
```
# Create dataset
X, y = spiral_data(samples=100, classes=3)
# Create Dense layer with 2 input features and 3 output values
dense1 = Layer_Dense(2, 3)
# Create ReLU activation (to be used with Dense layer):
activation1 = Activation_ReLU()
# Create second Dense layer with 3 input features (as we take output
# of previous layer here) and 3 output values
dense2 = Layer_Dense(3, 3)
# Create Softmax activation (to be used with Dense layer):
activation2 = Activation_Softmax()

# Make a forward pass of our training data through this layer
dense1.forward(X)

# Make a forward pass through activation function
# it takes the output of first dense layer here
activation1.forward(dense1.output)
# Make a forward pass through second Dense layer
# it takes outputs of activation function of first layer as inputs
dense2.forward(activation1.output)
# Make a forward pass through activation function
# it takes the output of second dense layer here
activation2.forward(dense2.output)
# Let's see output of the first few samples:
print(activation2.output[:5])
```

```
[[0.33333334 0.33333334 0.33333334]
 [0.33333316 0.3333332  0.33333364]
 [0.33333287 0.3333329  0.33333418]
 [0.3333326  0.33333263 0.33333477]
 [0.33333233 0.3333324  0.33333528]]
```

3 Calculating Network Error with Categorical Cross Entropy Loss

loss = negative sum of the expected output * log(neural network output) loss = - sum(expected_i * log(nn_output_i)) for all i in outputs

In the classification case, incorrect outputs do not end up mattering as the expected_i for the wrong class is 0.

[6]:
```
nn_outputs = np.array([
    [0.7, 0.1, 0.2],
```

```
    [0.1, 0.5, 0.4],
    [0.02, 0.9, 0.08]])
class_targets = [0, 1, 1]
losses = -np.log(nn_outputs[range(len(nn_outputs)), class_targets])
print(f"Losses: {losses}")
print(f"Average Loss: {np.average(losses)}")
```

```
Losses: [0.35667494 0.69314718 0.10536052]
Average Loss: 0.38506088005216804
```

3.1 Loss with One Hot Encoding

Classification typically has the expected output to be all zero except for the class the inputs belong too. This leads to simplfiying the cross entropy loss calculation.

[7]:
```
true_output = np.array([
    [1, 0, 0],
    [0, 1, 0],
    [0, 1, 0]
])

nn_output = np.array([
    [0.7, 0.2, 0.1],
    [0.1, 0.5, 0.4],
    [0.02, 0.9, 0.08]
])

# Element by element multiplication "erases" the output terms corresponding␣
 ↪with 0
A = true_output*nn_output

# Sum the columns (ie, sum every element in row 0, then row 1, etc) because␣
 ↪each row is a batch of output
B = np.sum(A, axis = 1)

# Get the cross entropy loss
C = -np.log(B)

print(f"Losses: {C}")
print(f"Average Loss: {np.mean(C)}")
```

```
Losses: [0.35667494 0.69314718 0.10536052]
Average Loss: 0.38506088005216804
```

3.2 Implementing the Loss Class

[2]:
```
# Base class for Loss functions
class Loss:
    '''Calculates the data and regularization losses given
    model output and ground truth values'''
    def calculate(self, output, y):
        sample_losses = self.forward(output, y)
        data_loss = np.average(sample_losses)
        return data_loss
```

3.3 Implementing the Categorical Cross Entropy Loss Class

[3]:
```
class Loss_CategoricalCrossEntropy(Loss):
    def forward(self, y_pred, y_true):
        '''y_pred is the neural network output
        y_true is the ideal output of the neural network'''
        samples = len(y_pred)
        # Bound the predicted values
        y_pred_clipped = np.clip(y_pred, 1e-7, 1-1e-7)

        if len(y_true.shape) == 1:      # Categorically labeled
            correct_confidences = y_pred_clipped[range(samples), y_true]
        elif len(y_true.shape) == 2:    # One hot encoded
            correct_confidences = np.sum(y_pred_clipped*y_true, axis=1)

        # Calculate the losses
        negative_log_likelihoods = -np.log(correct_confidences)
        return negative_log_likelihoods
```

[17]:
```
nn_outputs = np.array([
    [0.7, 0.1, 0.2],
    [0.1, 0.5, 0.4],
    [0.02, 0.9, 0.08]])
class_targets = np.array([
    [1, 0, 0],
    [0, 1, 0],
    [0, 1, 0]])

loss_function = Loss_CategoricalCrossEntropy()
losses = loss_function.calculate(nn_outputs, class_targets)
print(f"Losses: {losses}")
print(f"Average Loss: {np.average(losses)}")
```

```
Losses: 0.38506088005216804
Average Loss: 0.38506088005216804
```

4 Introducing Accuracy

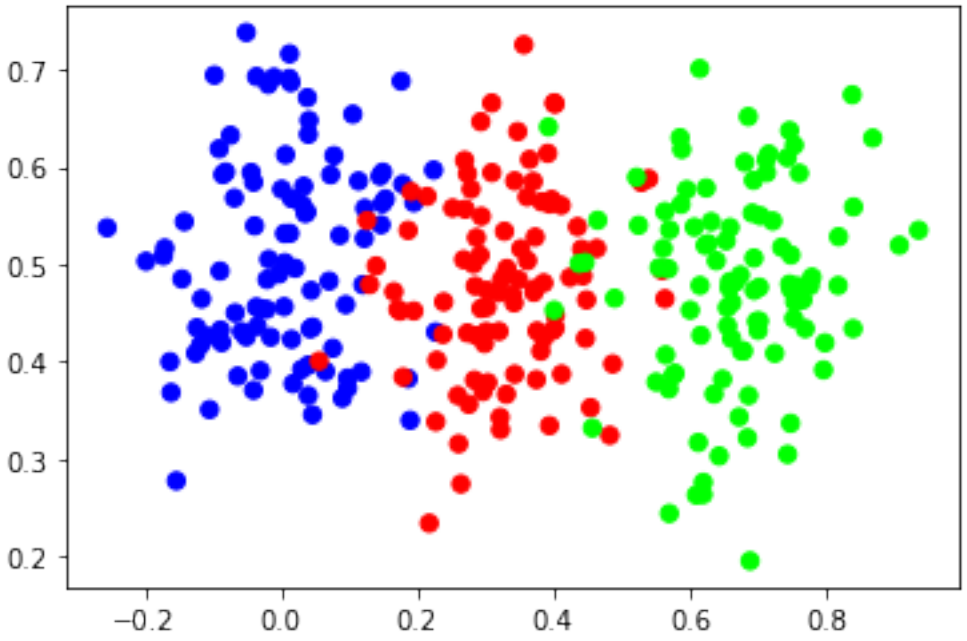
In the simple example, if the highest value in the outputs align with the correct classification, then that accuracy is 1. Even if it was 51% red and 49% blue, and the true output is red, it would be considered fully accurate.

[18]:
```
nn_outputs = np.array([
    [0.7, 0.1, 0.2],
    [0.1, 0.5, 0.4],
    [0.02, 0.9, 0.08]])
class_targets = np.array([
    [1, 0, 0],
    [0, 1, 0],
    [0, 1, 0]])

# Calculate the losses
loss_function = Loss_CategoricalCrossEntropy()
losses = loss_function.calculate(nn_outputs, class_targets)
print(f"Losses: {losses}")
print(f"Average Loss: {np.average(losses)}")

# Calculate the accuracy
predictions = np.argmax(nn_outputs, axis=1)
# If targets are one-hot encoded - convert them
if len(class_targets.shape) == 2:
    class_targets = np.argmax(class_targets, axis=1)
# True evaluates to 1; False to 0
accuracy = np.mean(predictions == class_targets)
print(f"Accuracy: {accuracy}")
```

```
Losses: 0.38506088005216804
Average Loss: 0.38506088005216804
Accuracy: 1.0
```

5 The Need for Optimization

[5]:
```
#SIMPLER DATASET
nnfs.init()
X, y = vertical_data(samples=100, classes=3)
plt.scatter(X[:, 0], X[:, 1], c=y, s=40, cmap='brg')
plt.show()
```



6 Test Strategy 1: Randomly Select Weights and Biases

For a large number of tests, randomly set weights and biases and look at accuracy.

[ ]:
```
# Create dataset
X, y = vertical_data(samples=100, classes=3)

# Create model
dense1 = Layer_Dense(2, 3) # first dense layer, 2 inputs
activation1 = Activation_ReLU()
dense2 = Layer_Dense(3, 3) # second dense layer, 3 inputs, 3 outputs
activation2 = Activation_Softmax()

# Create loss function
loss_function = Loss_CategoricalCrossEntropy()

# Helper variables
lowest_loss = 9999999 # some initial value
best_dense1_weights = dense1.weights.copy()
best_dense1_biases = dense1.biases.copy()
best_dense2_weights = dense2.weights.copy()
best_dense2_biases = dense2.biases.copy()

for iteration in range(10000):
 # Generate a new set of weights for iteration
```

```
dense1.weights = 0.05 * np.random.randn(2, 3)
dense1.biases = 0.05 * np.random.randn(1, 3)
dense2.weights = 0.05 * np.random.randn(3, 3)
dense2.biases = 0.05 * np.random.randn(1, 3)

# Perform a forward pass of the training data through this layer
dense1.forward(X)
activation1.forward(dense1.output)
dense2.forward(activation1.output)
activation2.forward(dense2.output)

# Perform a forward pass through activation function
# it takes the output of second dense layer here and returns loss
loss = loss_function.calculate(activation2.output, y)

# Calculate accuracy from output of activation2 and targets
# calculate values along first axis
predictions = np.argmax(activation2.output, axis=1)
accuracy = np.mean(predictions == y)

# If loss is smaller - print and save weights and biases aside
if loss < lowest_loss:
  print('New set of weights found, iteration:', iteration,'loss:', loss, 'acc:
↪', accuracy)
  best_dense1_weights = dense1.weights.copy()
  best_dense1_biases = dense1.biases.copy()
  best_dense2_weights = dense2.weights.copy()
  best_dense2_biases = dense2.biases.copy()
  lowest_loss = loss
```

```
New set of weights found, iteration: 0 loss: 1.0986564 acc: 0.3333333333333333
New set of weights found, iteration: 3 loss: 1.098138 acc: 0.3333333333333333
New set of weights found, iteration: 117 loss: 1.0980115 acc: 0.3333333333333333
New set of weights found, iteration: 124 loss: 1.0977516 acc: 0.6
New set of weights found, iteration: 165 loss: 1.097571 acc: 0.3333333333333333
New set of weights found, iteration: 552 loss: 1.0974693 acc: 0.34
New set of weights found, iteration: 778 loss: 1.0968257 acc: 0.3333333333333333
New set of weights found, iteration: 4307 loss: 1.0965533 acc:
0.3333333333333333
New set of weights found, iteration: 4615 loss: 1.0964499 acc:
0.3333333333333333
New set of weights found, iteration: 9450 loss: 1.0964295 acc:
0.3333333333333333
```

7 Test Strategy 2: Randomly Adjust Weights and Biases

For a large number of tests with a starting weight and bias, update the weights and biases by some small, random value. If the new accuracy is higher, keep the weights and biases. If the new accuracy is lower, revert back to the last weights and biases.

[8]:
```
# Create dataset
X, y = vertical_data(samples=100, classes=3)

# Create model
dense1 = Layer_Dense(2, 3) # first dense layer, 2 inputs
activation1 = Activation_ReLU()
dense2 = Layer_Dense(3, 3) # second dense layer, 3 inputs, 3 outputs
activation2 = Activation_Softmax()

# Create loss function
loss_function = Loss_CategoricalCrossEntropy()

# Helper variables
lowest_loss = 9999999 # some initial value
best_dense1_weights = dense1.weights.copy()
best_dense1_biases = dense1.biases.copy()
best_dense2_weights = dense2.weights.copy()
best_dense2_biases = dense2.biases.copy()
for iteration in range(10000):
 # Update weights with some small random values
 dense1.weights += 0.05 * np.random.randn(2, 3)
 dense1.biases += 0.05 * np.random.randn(1, 3)
 dense2.weights += 0.05 * np.random.randn(3, 3)
 dense2.biases += 0.05 * np.random.randn(1, 3)

 # Perform a forward pass of our training data through this layer
 dense1.forward(X)
 activation1.forward(dense1.output)
 dense2.forward(activation1.output)
 activation2.forward(dense2.output)

 # Perform a forward pass through activation function
 # it takes the output of second dense layer here and returns loss
 loss = loss_function.calculate(activation2.output, y)

 # Calculate accuracy from output of activation2 and targets
 # calculate values along first axis
 predictions = np.argmax(activation2.output, axis=1)
 accuracy = np.mean(predictions == y)

 # If loss is smaller - print and save weights and biases aside
 if loss < lowest_loss:
```

```
 print('New set of weights found, iteration:', iteration,'loss:', loss, 'acc:
↪', accuracy)
 best_dense1_weights = dense1.weights.copy()
 best_dense1_biases = dense1.biases.copy()
 best_dense2_weights = dense2.weights.copy()
 best_dense2_biases = dense2.biases.copy()
 lowest_loss = loss
# Revert weights and biases
else:
 dense1.weights = best_dense1_weights.copy()
 dense1.biases = best_dense1_biases.copy()
 dense2.weights = best_dense2_weights.copy()
 dense2.biases = best_dense2_biases.copy()
```

```
New set of weights found, iteration: 0 loss: 1.1004413 acc: 0.3333333333333333
New set of weights found, iteration: 1 loss: 1.1003714 acc: 0.3333333333333333
New set of weights found, iteration: 2 loss: 1.0999109 acc: 0.3333333333333333
New set of weights found, iteration: 6 loss: 1.098478 acc: 0.3333333333333333
New set of weights found, iteration: 7 loss: 1.0979133 acc: 0.3333333333333333
New set of weights found, iteration: 10 loss: 1.0962688 acc: 0.3333333333333333
New set of weights found, iteration: 11 loss: 1.0956886 acc: 0.3333333333333333
New set of weights found, iteration: 18 loss: 1.0933328 acc: 0.3333333333333333
New set of weights found, iteration: 27 loss: 1.0928771 acc: 0.3333333333333333
New set of weights found, iteration: 35 loss: 1.0894114 acc: 0.64
New set of weights found, iteration: 38 loss: 1.0819336 acc: 0.6666666666666666
New set of weights found, iteration: 42 loss: 1.0804778 acc: 0.5866666666666667
New set of weights found, iteration: 48 loss: 1.0791433 acc: 0.6
New set of weights found, iteration: 50 loss: 1.076686 acc: 0.66
New set of weights found, iteration: 57 loss: 1.0729789 acc: 0.6333333333333333
New set of weights found, iteration: 62 loss: 1.0626912 acc: 0.6666666666666666
New set of weights found, iteration: 64 loss: 1.062278 acc: 0.6266666666666667
New set of weights found, iteration: 66 loss: 1.0609949 acc: 0.3333333333333333
New set of weights found, iteration: 71 loss: 1.0521731 acc: 0.3333333333333333
New set of weights found, iteration: 76 loss: 1.0494336 acc: 0.3333333333333333
New set of weights found, iteration: 79 loss: 1.0357945 acc: 0.39
New set of weights found, iteration: 81 loss: 1.0338205 acc: 0.39
New set of weights found, iteration: 83 loss: 1.0324905 acc: 0.3433333333333333
New set of weights found, iteration: 84 loss: 1.0297213 acc: 0.39666666666666667
New set of weights found, iteration: 86 loss: 1.0251579 acc: 0.5866666666666667
New set of weights found, iteration: 91 loss: 1.021905 acc: 0.64
New set of weights found, iteration: 94 loss: 1.0156672 acc: 0.44333333333333336
New set of weights found, iteration: 97 loss: 1.0146924 acc: 0.6366666666666667
New set of weights found, iteration: 101 loss: 1.0104957 acc: 0.6566666666666666
New set of weights found, iteration: 104 loss: 1.0020038 acc: 0.6433333333333333
New set of weights found, iteration: 105 loss: 0.99834555 acc:
0.5933333333333334
New set of weights found, iteration: 106 loss: 0.9966272 acc: 0.6166666666666667
```

```
New set of weights found, iteration: 119 loss: 0.99438596 acc: 0.65
New set of weights found, iteration: 125 loss: 0.993829 acc: 0.6633333333333333
New set of weights found, iteration: 127 loss: 0.98677754 acc:
0.6666666666666666
New set of weights found, iteration: 130 loss: 0.98621744 acc:
0.6666666666666666
New set of weights found, iteration: 132 loss: 0.9835827 acc: 0.6533333333333333
New set of weights found, iteration: 135 loss: 0.97278476 acc:
0.6666666666666666
New set of weights found, iteration: 140 loss: 0.96625113 acc:
0.6533333333333333
New set of weights found, iteration: 141 loss: 0.9602833 acc: 0.6266666666666667
New set of weights found, iteration: 142 loss: 0.960012 acc: 0.6633333333333333
New set of weights found, iteration: 148 loss: 0.9562915 acc: 0.6666666666666666
New set of weights found, iteration: 149 loss: 0.9523678 acc: 0.6633333333333333
New set of weights found, iteration: 152 loss: 0.95160383 acc:
0.6266666666666667
New set of weights found, iteration: 153 loss: 0.9451301 acc: 0.6433333333333333
New set of weights found, iteration: 154 loss: 0.9373847 acc: 0.3566666666666667
New set of weights found, iteration: 161 loss: 0.9350953 acc:
0.39666666666666667
New set of weights found, iteration: 165 loss: 0.93221325 acc:
0.3566666666666667
New set of weights found, iteration: 170 loss: 0.9305483 acc: 0.39
New set of weights found, iteration: 172 loss: 0.9251562 acc: 0.5133333333333333
New set of weights found, iteration: 175 loss: 0.92369926 acc:
0.5133333333333333
New set of weights found, iteration: 178 loss: 0.9234428 acc: 0.5466666666666666
New set of weights found, iteration: 179 loss: 0.922442 acc: 0.45
New set of weights found, iteration: 181 loss: 0.9147015 acc: 0.3566666666666667
New set of weights found, iteration: 183 loss: 0.90779805 acc:
0.7166666666666667
New set of weights found, iteration: 184 loss: 0.9027177 acc: 0.7266666666666667
New set of weights found, iteration: 187 loss: 0.89574933 acc: 0.69
New set of weights found, iteration: 193 loss: 0.875872 acc: 0.49333333333333335
New set of weights found, iteration: 196 loss: 0.8563512 acc: 0.77
New set of weights found, iteration: 211 loss: 0.8544111 acc: 0.73
New set of weights found, iteration: 213 loss: 0.8524884 acc: 0.7533333333333333
New set of weights found, iteration: 216 loss: 0.8505173 acc: 0.66
New set of weights found, iteration: 221 loss: 0.8498888 acc: 0.6566666666666666
New set of weights found, iteration: 229 loss: 0.8474012 acc: 0.6666666666666666
New set of weights found, iteration: 231 loss: 0.8292791 acc: 0.6966666666666667
New set of weights found, iteration: 237 loss: 0.8153417 acc: 0.73
New set of weights found, iteration: 239 loss: 0.8059437 acc: 0.6666666666666666
New set of weights found, iteration: 240 loss: 0.8048912 acc: 0.6666666666666666
New set of weights found, iteration: 242 loss: 0.80074865 acc:
0.6666666666666666
New set of weights found, iteration: 245 loss: 0.7959438 acc: 0.6666666666666666
```

```
New set of weights found, iteration: 246 loss: 0.7937235 acc: 0.6666666666666666
New set of weights found, iteration: 247 loss: 0.7921869 acc: 0.7133333333333334
New set of weights found, iteration: 248 loss: 0.7904797 acc: 0.6666666666666666
New set of weights found, iteration: 249 loss: 0.7902046 acc: 0.6666666666666666
New set of weights found, iteration: 251 loss: 0.7724022 acc: 0.6666666666666666
New set of weights found, iteration: 255 loss: 0.7669011 acc: 0.6666666666666666
New set of weights found, iteration: 256 loss: 0.75499004 acc:
0.6666666666666666
New set of weights found, iteration: 257 loss: 0.74510765 acc:
0.7166666666666667
New set of weights found, iteration: 259 loss: 0.73144156 acc:
0.7033333333333334
New set of weights found, iteration: 260 loss: 0.729851 acc: 0.7033333333333334
New set of weights found, iteration: 264 loss: 0.72137403 acc:
0.6666666666666666
New set of weights found, iteration: 265 loss: 0.7202212 acc: 0.77
New set of weights found, iteration: 267 loss: 0.71849644 acc:
0.7966666666666666
New set of weights found, iteration: 271 loss: 0.7111821 acc: 0.8466666666666667
New set of weights found, iteration: 272 loss: 0.6996752 acc: 0.8233333333333334
New set of weights found, iteration: 280 loss: 0.69436175 acc: 0.85
New set of weights found, iteration: 286 loss: 0.6916804 acc: 0.8233333333333334
New set of weights found, iteration: 295 loss: 0.69114727 acc: 0.85
New set of weights found, iteration: 296 loss: 0.6862494 acc: 0.84
New set of weights found, iteration: 300 loss: 0.6840637 acc: 0.8366666666666667
New set of weights found, iteration: 301 loss: 0.66635275 acc:
0.8866666666666667
New set of weights found, iteration: 316 loss: 0.66121036 acc:
0.8333333333333334
New set of weights found, iteration: 318 loss: 0.6531562 acc: 0.8266666666666667
New set of weights found, iteration: 319 loss: 0.65067077 acc:
0.8133333333333334
New set of weights found, iteration: 323 loss: 0.64174473 acc: 0.9
New set of weights found, iteration: 327 loss: 0.64135444 acc:
0.8766666666666667
New set of weights found, iteration: 330 loss: 0.634816 acc: 0.9
New set of weights found, iteration: 331 loss: 0.63439554 acc: 0.83
New set of weights found, iteration: 337 loss: 0.61878663 acc:
0.8766666666666667
New set of weights found, iteration: 338 loss: 0.61758184 acc:
0.8366666666666667
New set of weights found, iteration: 340 loss: 0.6141628 acc: 0.84
New set of weights found, iteration: 350 loss: 0.60532016 acc:
0.8333333333333334
New set of weights found, iteration: 353 loss: 0.5960431 acc: 0.9
New set of weights found, iteration: 354 loss: 0.5949759 acc: 0.8966666666666666
New set of weights found, iteration: 356 loss: 0.5906401 acc: 0.8433333333333334
New set of weights found, iteration: 359 loss: 0.5798522 acc: 0.8466666666666667
```

```
New set of weights found, iteration: 360 loss: 0.5785292 acc: 0.86
New set of weights found, iteration: 362 loss: 0.57752687 acc: 0.86
New set of weights found, iteration: 363 loss: 0.5740031 acc: 0.8633333333333333
New set of weights found, iteration: 364 loss: 0.57001764 acc:
0.8466666666666667
New set of weights found, iteration: 366 loss: 0.5677138 acc: 0.8333333333333334
New set of weights found, iteration: 367 loss: 0.5671378 acc: 0.8433333333333334
New set of weights found, iteration: 368 loss: 0.5650589 acc: 0.8333333333333334
New set of weights found, iteration: 369 loss: 0.55381805 acc:
0.8933333333333333
New set of weights found, iteration: 378 loss: 0.553261 acc: 0.9266666666666666
New set of weights found, iteration: 379 loss: 0.55210435 acc:
0.9066666666666666
New set of weights found, iteration: 380 loss: 0.55189973 acc: 0.85
New set of weights found, iteration: 381 loss: 0.54758984 acc:
0.8433333333333334
New set of weights found, iteration: 384 loss: 0.53211606 acc: 0.85
New set of weights found, iteration: 404 loss: 0.5225475 acc: 0.9
New set of weights found, iteration: 409 loss: 0.5161618 acc: 0.93
New set of weights found, iteration: 410 loss: 0.5132672 acc: 0.92
New set of weights found, iteration: 425 loss: 0.5113815 acc: 0.9166666666666666
New set of weights found, iteration: 443 loss: 0.51136065 acc:
0.8933333333333333
New set of weights found, iteration: 444 loss: 0.50422305 acc:
0.8866666666666667
New set of weights found, iteration: 445 loss: 0.5036243 acc: 0.9
New set of weights found, iteration: 448 loss: 0.501374 acc: 0.89
New set of weights found, iteration: 449 loss: 0.49592137 acc:
0.9166666666666666
New set of weights found, iteration: 453 loss: 0.4913305 acc: 0.9066666666666666
New set of weights found, iteration: 454 loss: 0.49017328 acc:
0.8933333333333333
New set of weights found, iteration: 455 loss: 0.47922668 acc: 0.92
New set of weights found, iteration: 462 loss: 0.47773126 acc: 0.92
New set of weights found, iteration: 465 loss: 0.4757397 acc: 0.9133333333333333
New set of weights found, iteration: 466 loss: 0.46511835 acc:
0.9133333333333333
New set of weights found, iteration: 468 loss: 0.45972347 acc:
0.9233333333333333
New set of weights found, iteration: 474 loss: 0.45767045 acc:
0.9166666666666666
New set of weights found, iteration: 475 loss: 0.4531248 acc: 0.9133333333333333
New set of weights found, iteration: 477 loss: 0.45079032 acc:
0.9166666666666666
New set of weights found, iteration: 493 loss: 0.44731975 acc:
0.9266666666666666
New set of weights found, iteration: 501 loss: 0.44255528 acc:
0.9166666666666666
```

```
New set of weights found, iteration: 504 loss: 0.43813455 acc:
0.9233333333333333
New set of weights found, iteration: 509 loss: 0.43588492 acc:
0.9133333333333333
New set of weights found, iteration: 514 loss: 0.4349694 acc: 0.91
New set of weights found, iteration: 522 loss: 0.42528915 acc:
0.9333333333333333
New set of weights found, iteration: 525 loss: 0.4251014 acc: 0.9166666666666666
New set of weights found, iteration: 527 loss: 0.42076018 acc:
0.9133333333333333
New set of weights found, iteration: 529 loss: 0.42038155 acc:
0.9233333333333333
New set of weights found, iteration: 532 loss: 0.42007548 acc: 0.93
New set of weights found, iteration: 537 loss: 0.417006 acc: 0.93
New set of weights found, iteration: 541 loss: 0.41556618 acc: 0.93
New set of weights found, iteration: 545 loss: 0.41544887 acc:
0.9266666666666666
New set of weights found, iteration: 547 loss: 0.413486 acc: 0.92
New set of weights found, iteration: 552 loss: 0.4105945 acc: 0.9166666666666666
New set of weights found, iteration: 554 loss: 0.41028866 acc: 0.93
New set of weights found, iteration: 558 loss: 0.40889078 acc: 0.91
New set of weights found, iteration: 567 loss: 0.4067462 acc: 0.9033333333333333
New set of weights found, iteration: 572 loss: 0.40230143 acc: 0.93
New set of weights found, iteration: 584 loss: 0.4005359 acc: 0.9366666666666666
New set of weights found, iteration: 589 loss: 0.39260605 acc:
0.9266666666666666
New set of weights found, iteration: 590 loss: 0.3811324 acc: 0.9333333333333333
New set of weights found, iteration: 596 loss: 0.38065374 acc:
0.9333333333333333
New set of weights found, iteration: 597 loss: 0.37342408 acc: 0.93
New set of weights found, iteration: 602 loss: 0.37175146 acc: 0.93
New set of weights found, iteration: 606 loss: 0.37065893 acc: 0.93
New set of weights found, iteration: 613 loss: 0.3700704 acc: 0.9266666666666666
New set of weights found, iteration: 615 loss: 0.3681412 acc: 0.9333333333333333
New set of weights found, iteration: 617 loss: 0.3666133 acc: 0.9333333333333333
New set of weights found, iteration: 620 loss: 0.3613251 acc: 0.9333333333333333
New set of weights found, iteration: 622 loss: 0.35900766 acc:
0.9333333333333333
New set of weights found, iteration: 629 loss: 0.35549423 acc:
0.9266666666666666
New set of weights found, iteration: 637 loss: 0.35331622 acc:
0.9233333333333333
New set of weights found, iteration: 640 loss: 0.35213044 acc:
0.9233333333333333
New set of weights found, iteration: 642 loss: 0.34433836 acc:
0.9266666666666666
New set of weights found, iteration: 648 loss: 0.34404942 acc:
0.9233333333333333
```

```
New set of weights found, iteration: 651 loss: 0.33547923 acc:
0.9166666666666666
New set of weights found, iteration: 653 loss: 0.3346703 acc: 0.9233333333333333
New set of weights found, iteration: 655 loss: 0.3344787 acc: 0.92
New set of weights found, iteration: 658 loss: 0.33275664 acc: 0.92
New set of weights found, iteration: 660 loss: 0.33186576 acc:
0.9333333333333333
New set of weights found, iteration: 661 loss: 0.32966954 acc: 0.92
New set of weights found, iteration: 666 loss: 0.32630792 acc:
0.9266666666666666
New set of weights found, iteration: 689 loss: 0.32379147 acc: 0.93
New set of weights found, iteration: 692 loss: 0.32295424 acc:
0.9266666666666666
New set of weights found, iteration: 694 loss: 0.3228055 acc: 0.9366666666666666
New set of weights found, iteration: 700 loss: 0.3196157 acc: 0.9266666666666666
New set of weights found, iteration: 709 loss: 0.3168142 acc: 0.9233333333333333
New set of weights found, iteration: 710 loss: 0.31664997 acc:
0.9233333333333333
New set of weights found, iteration: 716 loss: 0.31424063 acc:
0.9233333333333333
New set of weights found, iteration: 717 loss: 0.31154537 acc:
0.9133333333333333
New set of weights found, iteration: 719 loss: 0.31104082 acc:
0.9266666666666666
New set of weights found, iteration: 722 loss: 0.30798188 acc: 0.92
New set of weights found, iteration: 726 loss: 0.30246973 acc: 0.93
New set of weights found, iteration: 732 loss: 0.3008993 acc: 0.9366666666666666
New set of weights found, iteration: 735 loss: 0.2962119 acc: 0.92
New set of weights found, iteration: 737 loss: 0.29480347 acc:
0.9233333333333333
New set of weights found, iteration: 739 loss: 0.28837058 acc: 0.93
New set of weights found, iteration: 740 loss: 0.287716 acc: 0.9266666666666666
New set of weights found, iteration: 758 loss: 0.28682068 acc: 0.92
New set of weights found, iteration: 768 loss: 0.2837636 acc: 0.9266666666666666
New set of weights found, iteration: 774 loss: 0.28291276 acc:
0.9366666666666666
New set of weights found, iteration: 792 loss: 0.28240937 acc:
0.9233333333333333
New set of weights found, iteration: 803 loss: 0.28232166 acc:
0.9233333333333333
New set of weights found, iteration: 818 loss: 0.28129843 acc:
0.9233333333333333
New set of weights found, iteration: 822 loss: 0.27883106 acc:
0.9266666666666666
New set of weights found, iteration: 836 loss: 0.2774233 acc: 0.9333333333333333
New set of weights found, iteration: 839 loss: 0.27680957 acc:
0.9266666666666666
New set of weights found, iteration: 844 loss: 0.27618676 acc: 0.93
```

```
New set of weights found, iteration: 846 loss: 0.27285808 acc: 0.92
New set of weights found, iteration: 848 loss: 0.26968768 acc:
0.9233333333333333
New set of weights found, iteration: 858 loss: 0.26904428 acc:
0.9333333333333333
New set of weights found, iteration: 884 loss: 0.268827 acc: 0.9233333333333333
New set of weights found, iteration: 889 loss: 0.26809755 acc: 0.92
New set of weights found, iteration: 897 loss: 0.26714522 acc:
0.9266666666666666
New set of weights found, iteration: 898 loss: 0.26399252 acc:
0.9233333333333333
New set of weights found, iteration: 900 loss: 0.26366332 acc: 0.93
New set of weights found, iteration: 908 loss: 0.26240122 acc:
0.9233333333333333
New set of weights found, iteration: 925 loss: 0.26182953 acc:
0.9266666666666666
New set of weights found, iteration: 937 loss: 0.26113904 acc:
0.9233333333333333
New set of weights found, iteration: 942 loss: 0.2595808 acc: 0.9333333333333333
New set of weights found, iteration: 945 loss: 0.25693676 acc:
0.9266666666666666
New set of weights found, iteration: 948 loss: 0.25630182 acc:
0.9266666666666666
New set of weights found, iteration: 953 loss: 0.2554938 acc: 0.9366666666666666
New set of weights found, iteration: 955 loss: 0.25241867 acc:
0.9366666666666666
New set of weights found, iteration: 956 loss: 0.24921493 acc: 0.93
New set of weights found, iteration: 959 loss: 0.24579681 acc:
0.9333333333333333
New set of weights found, iteration: 960 loss: 0.24326381 acc:
0.9233333333333333
New set of weights found, iteration: 963 loss: 0.24075015 acc: 0.93
New set of weights found, iteration: 968 loss: 0.24000326 acc:
0.9333333333333333
New set of weights found, iteration: 973 loss: 0.2382134 acc: 0.92
New set of weights found, iteration: 991 loss: 0.23589782 acc:
0.9166666666666666
New set of weights found, iteration: 992 loss: 0.23532195 acc:
0.9233333333333333
New set of weights found, iteration: 1003 loss: 0.23462674 acc:
0.9133333333333333
New set of weights found, iteration: 1006 loss: 0.2343067 acc:
0.9266666666666666
New set of weights found, iteration: 1011 loss: 0.23375021 acc:
0.9233333333333333
New set of weights found, iteration: 1017 loss: 0.23200704 acc: 0.92
New set of weights found, iteration: 1027 loss: 0.23154141 acc: 0.92
New set of weights found, iteration: 1038 loss: 0.23015086 acc:
```

```
0.9166666666666666
New set of weights found, iteration: 1042 loss: 0.2283845 acc:
0.9166666666666666
New set of weights found, iteration: 1044 loss: 0.22817597 acc:
0.9333333333333333
New set of weights found, iteration: 1045 loss: 0.22744556 acc: 0.92
New set of weights found, iteration: 1048 loss: 0.22593918 acc:
0.9333333333333333
New set of weights found, iteration: 1052 loss: 0.22136909 acc: 0.92
New set of weights found, iteration: 1058 loss: 0.22063312 acc:
0.9233333333333333
New set of weights found, iteration: 1062 loss: 0.21697378 acc: 0.92
New set of weights found, iteration: 1063 loss: 0.2154923 acc: 0.92
New set of weights found, iteration: 1081 loss: 0.21519203 acc: 0.92
New set of weights found, iteration: 1088 loss: 0.21324503 acc: 0.92
New set of weights found, iteration: 1089 loss: 0.21294884 acc:
0.9233333333333333
New set of weights found, iteration: 1107 loss: 0.21196988 acc: 0.92
New set of weights found, iteration: 1123 loss: 0.21161827 acc: 0.92
New set of weights found, iteration: 1132 loss: 0.20942998 acc:
0.9233333333333333
New set of weights found, iteration: 1134 loss: 0.20683934 acc: 0.92
New set of weights found, iteration: 1152 loss: 0.20673288 acc: 0.92
New set of weights found, iteration: 1158 loss: 0.20602302 acc: 0.92
New set of weights found, iteration: 1196 loss: 0.20581451 acc:
0.9166666666666666
New set of weights found, iteration: 1211 loss: 0.20252144 acc:
0.9233333333333333
New set of weights found, iteration: 1219 loss: 0.20165876 acc: 0.92
New set of weights found, iteration: 1222 loss: 0.20161158 acc:
0.9166666666666666
New set of weights found, iteration: 1227 loss: 0.20160003 acc: 0.92
New set of weights found, iteration: 1238 loss: 0.2002937 acc: 0.92
New set of weights found, iteration: 1242 loss: 0.19846326 acc:
0.9166666666666666
New set of weights found, iteration: 1253 loss: 0.19829592 acc:
0.9233333333333333
New set of weights found, iteration: 1258 loss: 0.19702826 acc:
0.9166666666666666
New set of weights found, iteration: 1269 loss: 0.19549341 acc: 0.92
New set of weights found, iteration: 1274 loss: 0.1936545 acc:
0.9166666666666666
New set of weights found, iteration: 1295 loss: 0.19341804 acc:
0.9166666666666666
New set of weights found, iteration: 1310 loss: 0.19328523 acc: 0.92
New set of weights found, iteration: 1340 loss: 0.19306006 acc:
0.9166666666666666
New set of weights found, iteration: 1343 loss: 0.1926579 acc:
```

```
0.9166666666666666
New set of weights found, iteration: 1369 loss: 0.19204491 acc: 0.92
New set of weights found, iteration: 1370 loss: 0.19107494 acc:
0.9166666666666666
New set of weights found, iteration: 1386 loss: 0.19073413 acc:
0.9166666666666666
New set of weights found, iteration: 1416 loss: 0.19026798 acc:
0.9233333333333333
New set of weights found, iteration: 1420 loss: 0.19009912 acc:
0.9166666666666666
New set of weights found, iteration: 1430 loss: 0.1890326 acc: 0.92
New set of weights found, iteration: 1461 loss: 0.1889198 acc:
0.9233333333333333
New set of weights found, iteration: 1482 loss: 0.18784311 acc:
0.9266666666666666
New set of weights found, iteration: 1483 loss: 0.18726717 acc:
0.9266666666666666
New set of weights found, iteration: 1497 loss: 0.18642144 acc: 0.92
New set of weights found, iteration: 1504 loss: 0.18634032 acc:
0.9166666666666666
New set of weights found, iteration: 1535 loss: 0.18609515 acc:
0.9166666666666666
New set of weights found, iteration: 1572 loss: 0.18588851 acc: 0.92
New set of weights found, iteration: 1601 loss: 0.18577391 acc:
0.9166666666666666
New set of weights found, iteration: 1605 loss: 0.18567057 acc:
0.9233333333333333
New set of weights found, iteration: 1610 loss: 0.18486346 acc:
0.9166666666666666
New set of weights found, iteration: 1616 loss: 0.18467394 acc: 0.92
New set of weights found, iteration: 1622 loss: 0.18462817 acc: 0.92
New set of weights found, iteration: 1636 loss: 0.18456662 acc:
0.9166666666666666
New set of weights found, iteration: 1704 loss: 0.18404384 acc: 0.92
New set of weights found, iteration: 1706 loss: 0.1837163 acc: 0.92
New set of weights found, iteration: 1780 loss: 0.18321306 acc:
0.9233333333333333
New set of weights found, iteration: 1796 loss: 0.18313819 acc: 0.92
New set of weights found, iteration: 1797 loss: 0.18233562 acc:
0.9233333333333333
New set of weights found, iteration: 1858 loss: 0.18162519 acc: 0.92
New set of weights found, iteration: 1879 loss: 0.18157774 acc:
0.9233333333333333
New set of weights found, iteration: 1916 loss: 0.18105999 acc: 0.92
New set of weights found, iteration: 1928 loss: 0.18020274 acc:
0.9233333333333333
New set of weights found, iteration: 1974 loss: 0.1798982 acc:
0.9233333333333333
```

```
New set of weights found, iteration: 1994 loss: 0.17920457 acc:
0.9233333333333333
New set of weights found, iteration: 2006 loss: 0.17876633 acc:
0.9233333333333333
New set of weights found, iteration: 2088 loss: 0.17826015 acc:
0.9233333333333333
New set of weights found, iteration: 2273 loss: 0.1780754 acc:
0.9233333333333333
New set of weights found, iteration: 2278 loss: 0.17789875 acc:
0.9233333333333333
New set of weights found, iteration: 2349 loss: 0.17731167 acc:
0.9233333333333333
New set of weights found, iteration: 2439 loss: 0.1767629 acc:
0.9233333333333333
New set of weights found, iteration: 2580 loss: 0.17660397 acc:
0.9233333333333333
New set of weights found, iteration: 2648 loss: 0.1765489 acc:
0.9233333333333333
New set of weights found, iteration: 2873 loss: 0.17626359 acc:
0.9233333333333333
New set of weights found, iteration: 2975 loss: 0.17622189 acc:
0.9233333333333333
New set of weights found, iteration: 3036 loss: 0.17601877 acc:
0.9233333333333333
New set of weights found, iteration: 3158 loss: 0.17592181 acc:
0.9233333333333333
New set of weights found, iteration: 3175 loss: 0.17590967 acc:
0.9233333333333333
New set of weights found, iteration: 3240 loss: 0.17587931 acc:
0.9233333333333333
New set of weights found, iteration: 3251 loss: 0.17557663 acc:
0.9233333333333333
New set of weights found, iteration: 3252 loss: 0.17530455 acc:
0.9233333333333333
New set of weights found, iteration: 3293 loss: 0.17516907 acc:
0.9233333333333333
New set of weights found, iteration: 3308 loss: 0.17511182 acc:
0.9233333333333333
New set of weights found, iteration: 3338 loss: 0.17484583 acc:
0.9233333333333333
New set of weights found, iteration: 3360 loss: 0.17471306 acc:
0.9233333333333333
New set of weights found, iteration: 3393 loss: 0.17469047 acc:
0.9233333333333333
New set of weights found, iteration: 3395 loss: 0.1743806 acc:
0.9233333333333333
New set of weights found, iteration: 3642 loss: 0.17433934 acc:
0.9233333333333333
```

```
New set of weights found, iteration: 3895 loss: 0.17422049 acc:
0.9233333333333333
New set of weights found, iteration: 3922 loss: 0.17404793 acc:
0.9233333333333333
New set of weights found, iteration: 4161 loss: 0.17401835 acc:
0.9233333333333333
New set of weights found, iteration: 4186 loss: 0.17398895 acc:
0.9233333333333333
New set of weights found, iteration: 4194 loss: 0.17370409 acc:
0.9233333333333333
New set of weights found, iteration: 4392 loss: 0.17358227 acc:
0.9233333333333333
New set of weights found, iteration: 5189 loss: 0.17355132 acc:
0.9233333333333333
New set of weights found, iteration: 5918 loss: 0.17352708 acc:
0.9233333333333333
New set of weights found, iteration: 6323 loss: 0.17347664 acc:
0.9233333333333333
New set of weights found, iteration: 6503 loss: 0.17335322 acc:
0.9233333333333333
New set of weights found, iteration: 6539 loss: 0.17328598 acc:
0.9233333333333333
New set of weights found, iteration: 7013 loss: 0.17325447 acc:
0.9233333333333333
New set of weights found, iteration: 7724 loss: 0.17321768 acc:
0.9233333333333333
New set of weights found, iteration: 7816 loss: 0.17312418 acc:
0.9233333333333333
New set of weights found, iteration: 7890 loss: 0.17311428 acc:
0.9233333333333333
New set of weights found, iteration: 8234 loss: 0.1730845 acc:
0.9233333333333333
New set of weights found, iteration: 8588 loss: 0.173055 acc: 0.9233333333333333
New set of weights found, iteration: 8671 loss: 0.17292134 acc:
0.9233333333333333
New set of weights found, iteration: 8800 loss: 0.17289625 acc:
0.9233333333333333
New set of weights found, iteration: 8953 loss: 0.1728954 acc:
0.9233333333333333
New set of weights found, iteration: 8970 loss: 0.17283815 acc:
0.9233333333333333
```

8 Test Strategy 2 on Spiral Dataset

[9]:
```
# Create dataset
X, y = spiral_data(samples=100, classes=3)

# Create model
dense1 = Layer_Dense(2, 3) # first dense layer, 2 inputs
activation1 = Activation_ReLU()
dense2 = Layer_Dense(3, 3) # second dense layer, 3 inputs, 3 outputs
activation2 = Activation_Softmax()

# Create loss function
loss_function = Loss_CategoricalCrossEntropy()

# Helper variables
lowest_loss = 9999999 # some initial value
best_dense1_weights = dense1.weights.copy()
best_dense1_biases = dense1.biases.copy()
best_dense2_weights = dense2.weights.copy()
best_dense2_biases = dense2.biases.copy()
for iteration in range(10000):
 # Update weights with some small random values
 dense1.weights += 0.05 * np.random.randn(2, 3)
 dense1.biases += 0.05 * np.random.randn(1, 3)
 dense2.weights += 0.05 * np.random.randn(3, 3)
 dense2.biases += 0.05 * np.random.randn(1, 3)

 # Perform a forward pass of our training data through this layer
 dense1.forward(X)
 activation1.forward(dense1.output)
 dense2.forward(activation1.output)
 activation2.forward(dense2.output)

 # Perform a forward pass through activation function
 # it takes the output of second dense layer here and returns loss
 loss = loss_function.calculate(activation2.output, y)

 # Calculate accuracy from output of activation2 and targets
 # calculate values along first axis
 predictions = np.argmax(activation2.output, axis=1)
 accuracy = np.mean(predictions == y)

 # If loss is smaller - print and save weights and biases aside
 if loss < lowest_loss:
  print('New set of weights found, iteration:', iteration,'loss:', loss, 'acc:
 ↪', accuracy)
  best_dense1_weights = dense1.weights.copy()
```

```
 best_dense1_biases = dense1.biases.copy()
 best_dense2_weights = dense2.weights.copy()
 best_dense2_biases = dense2.biases.copy()
 lowest_loss = loss
# Revert weights and biases
else:
 dense1.weights = best_dense1_weights.copy()
 dense1.biases = best_dense1_biases.copy()
 dense2.weights = best_dense2_weights.copy()
 dense2.biases = best_dense2_biases.copy()
```

```
New set of weights found, iteration: 0 loss: 1.0986983 acc: 0.3333333333333333
New set of weights found, iteration: 42 loss: 1.0984432 acc: 0.3333333333333333
New set of weights found, iteration: 48 loss: 1.0983725 acc: 0.3333333333333333
New set of weights found, iteration: 54 loss: 1.097728 acc: 0.38666666666666666
New set of weights found, iteration: 55 loss: 1.0976882 acc: 0.3333333333333333
New set of weights found, iteration: 56 loss: 1.0973428 acc: 0.38333333333333336
New set of weights found, iteration: 57 loss: 1.0970833 acc: 0.3333333333333333
New set of weights found, iteration: 64 loss: 1.0964628 acc: 0.3566666666666667
New set of weights found, iteration: 65 loss: 1.0957834 acc: 0.34
New set of weights found, iteration: 84 loss: 1.0957702 acc: 0.34
New set of weights found, iteration: 90 loss: 1.0955024 acc: 0.3566666666666667
New set of weights found, iteration: 95 loss: 1.0942755 acc: 0.39
New set of weights found, iteration: 100 loss: 1.0938662 acc: 0.3466666666666667
New set of weights found, iteration: 101 loss: 1.091843 acc: 0.33666666666666667
New set of weights found, iteration: 104 loss: 1.0912626 acc: 0.34
New set of weights found, iteration: 105 loss: 1.0882009 acc:
0.36666666666666664
New set of weights found, iteration: 106 loss: 1.0867509 acc: 0.41
New set of weights found, iteration: 110 loss: 1.0861986 acc:
0.38333333333333336
New set of weights found, iteration: 111 loss: 1.0858816 acc: 0.3433333333333333
New set of weights found, iteration: 114 loss: 1.0845512 acc:
0.32666666666666666
New set of weights found, iteration: 115 loss: 1.0842649 acc: 0.3433333333333333
New set of weights found, iteration: 124 loss: 1.0840762 acc:
0.32666666666666666
New set of weights found, iteration: 125 loss: 1.0813359 acc: 0.39
New set of weights found, iteration: 133 loss: 1.0780971 acc: 0.37
New set of weights found, iteration: 137 loss: 1.077851 acc: 0.38666666666666666
New set of weights found, iteration: 138 loss: 1.0777876 acc: 0.4066666666666667
New set of weights found, iteration: 143 loss: 1.0771211 acc:
0.39666666666666667
New set of weights found, iteration: 144 loss: 1.0768937 acc:
0.38333333333333336
New set of weights found, iteration: 146 loss: 1.0742698 acc:
0.38333333333333336
```

```
New set of weights found, iteration: 148 loss: 1.0733455 acc: 0.41
New set of weights found, iteration: 162 loss: 1.0730222 acc: 0.4033333333333333
New set of weights found, iteration: 179 loss: 1.0726937 acc: 0.4166666666666667
New set of weights found, iteration: 191 loss: 1.0725039 acc: 0.42
New set of weights found, iteration: 222 loss: 1.0716708 acc: 0.4033333333333333
New set of weights found, iteration: 253 loss: 1.0708596 acc: 0.39
New set of weights found, iteration: 272 loss: 1.0706216 acc: 0.4066666666666667
New set of weights found, iteration: 290 loss: 1.0698603 acc: 0.44
New set of weights found, iteration: 300 loss: 1.0697052 acc: 0.4166666666666667
New set of weights found, iteration: 325 loss: 1.069674 acc: 0.43666666666666665
New set of weights found, iteration: 381 loss: 1.0691994 acc: 0.3933333333333333
New set of weights found, iteration: 398 loss: 1.0687411 acc: 0.4266666666666667
New set of weights found, iteration: 406 loss: 1.0684437 acc: 0.43
New set of weights found, iteration: 550 loss: 1.0684316 acc:
0.43333333333333335
New set of weights found, iteration: 570 loss: 1.0684133 acc: 0.41
New set of weights found, iteration: 594 loss: 1.068293 acc: 0.4
New set of weights found, iteration: 596 loss: 1.0681537 acc: 0.4066666666666667
New set of weights found, iteration: 597 loss: 1.0677991 acc: 0.42
New set of weights found, iteration: 642 loss: 1.0676459 acc:
0.39666666666666667
New set of weights found, iteration: 661 loss: 1.0675713 acc: 0.3933333333333333
New set of weights found, iteration: 681 loss: 1.0674102 acc:
0.38333333333333336
New set of weights found, iteration: 695 loss: 1.0673658 acc: 0.4033333333333333
New set of weights found, iteration: 701 loss: 1.0666102 acc: 0.4066666666666667
New set of weights found, iteration: 719 loss: 1.0663345 acc: 0.42
New set of weights found, iteration: 737 loss: 1.066033 acc: 0.4033333333333333
New set of weights found, iteration: 752 loss: 1.0657896 acc: 0.4266666666666667
New set of weights found, iteration: 903 loss: 1.0655118 acc: 0.4166666666666667
New set of weights found, iteration: 981 loss: 1.065493 acc: 0.41333333333333333
New set of weights found, iteration: 1006 loss: 1.0654801 acc: 0.41
New set of weights found, iteration: 1048 loss: 1.0651859 acc:
0.39666666666666667
New set of weights found, iteration: 1175 loss: 1.064625 acc: 0.4266666666666667
New set of weights found, iteration: 1209 loss: 1.0643268 acc:
0.41333333333333333
New set of weights found, iteration: 1245 loss: 1.0643263 acc:
0.43666666666666665
New set of weights found, iteration: 1302 loss: 1.0640283 acc: 0.4
New set of weights found, iteration: 1303 loss: 1.0634205 acc:
0.44333333333333336
New set of weights found, iteration: 1352 loss: 1.0630084 acc: 0.43
New set of weights found, iteration: 1577 loss: 1.0626279 acc:
0.42333333333333334
New set of weights found, iteration: 1594 loss: 1.0625374 acc:
0.43333333333333335
New set of weights found, iteration: 1600 loss: 1.0623267 acc:
```

```
0.44333333333333336
New set of weights found, iteration: 1794 loss: 1.0622777 acc: 0.41
New set of weights found, iteration: 1851 loss: 1.0618818 acc:
0.43333333333333335
New set of weights found, iteration: 1877 loss: 1.0616083 acc:
0.43333333333333335
New set of weights found, iteration: 1958 loss: 1.0614555 acc:
0.43666666666666665
New set of weights found, iteration: 1998 loss: 1.0613961 acc:
0.4066666666666667
New set of weights found, iteration: 2031 loss: 1.0606906 acc: 0.46
New set of weights found, iteration: 2130 loss: 1.0606595 acc: 0.43
New set of weights found, iteration: 2431 loss: 1.06059 acc: 0.4066666666666667
New set of weights found, iteration: 3294 loss: 1.0603732 acc: 0.4
New set of weights found, iteration: 3492 loss: 1.0603614 acc:
0.4266666666666667
New set of weights found, iteration: 3662 loss: 1.0598251 acc: 0.4
New set of weights found, iteration: 3756 loss: 1.0595479 acc: 0.39
New set of weights found, iteration: 3769 loss: 1.0593852 acc:
0.42333333333333334
New set of weights found, iteration: 3875 loss: 1.0583456 acc:
0.4033333333333333
New set of weights found, iteration: 3981 loss: 1.0582583 acc:
0.42333333333333334
New set of weights found, iteration: 4146 loss: 1.0579673 acc:
0.4166666666666667
New set of weights found, iteration: 4153 loss: 1.0578284 acc:
0.4166666666666667
New set of weights found, iteration: 4301 loss: 1.0575745 acc: 0.41
New set of weights found, iteration: 4405 loss: 1.057048 acc:
0.43333333333333335
New set of weights found, iteration: 4498 loss: 1.056719 acc: 0.4066666666666667
New set of weights found, iteration: 4594 loss: 1.0565504 acc:
0.43666666666666665
New set of weights found, iteration: 5092 loss: 1.0562842 acc:
0.4266666666666667
New set of weights found, iteration: 5117 loss: 1.0557985 acc: 0.4
New set of weights found, iteration: 5497 loss: 1.0555316 acc: 0.44
New set of weights found, iteration: 6021 loss: 1.0554525 acc:
0.3933333333333333
New set of weights found, iteration: 6154 loss: 1.0551611 acc:
0.4033333333333333
New set of weights found, iteration: 6168 loss: 1.0548483 acc: 0.42
New set of weights found, iteration: 6210 loss: 1.0546328 acc:
0.44666666666666666
New set of weights found, iteration: 6233 loss: 1.0541582 acc: 0.44
New set of weights found, iteration: 6323 loss: 1.0541245 acc:
0.4533333333333333
```

```
New set of weights found, iteration: 6386 loss: 1.0537696 acc:
0.4633333333333333
New set of weights found, iteration: 6702 loss: 1.0534701 acc:
0.4533333333333333
New set of weights found, iteration: 6997 loss: 1.0533447 acc:
0.4166666666666667
New set of weights found, iteration: 7101 loss: 1.0529538 acc: 0.41
New set of weights found, iteration: 7182 loss: 1.0524737 acc: 0.42
New set of weights found, iteration: 7476 loss: 1.0522219 acc:
0.44333333333333336
New set of weights found, iteration: 7719 loss: 1.0521553 acc:
0.44666666666666666
New set of weights found, iteration: 7858 loss: 1.0520765 acc:
0.4266666666666667
New set of weights found, iteration: 7877 loss: 1.0507878 acc: 0.41
New set of weights found, iteration: 7953 loss: 1.0506427 acc:
0.41333333333333333
New set of weights found, iteration: 8026 loss: 1.0503834 acc: 0.42
New set of weights found, iteration: 8763 loss: 1.0503162 acc:
0.41333333333333333
New set of weights found, iteration: 9308 loss: 1.0501956 acc: 0.41
New set of weights found, iteration: 9399 loss: 1.0493395 acc:
0.4066666666666667
New set of weights found, iteration: 9529 loss: 1.0491025 acc:
0.4166666666666667
New set of weights found, iteration: 9822 loss: 1.0488548 acc:
0.4533333333333333
```