

# Running data through a network

## Function Signature

```
def run(  
    self,  
    inputs: np.ndarray  
) -> np.ndarray:
```

## Parameters

- `inputs` (`np.ndarray`): The input data to run through the network. This should be a numpy array of shape `(input_shape,)`.

## Return Value

- `output` (`np.ndarray`): The output of the network after running the given input through it. This should be a numpy array of shape `(output_shape,)`.

## Description

The `run` function takes a single input and runs it through the network, returning the output of the network.

The `inputs` parameter should be a numpy array of shape `(input_shape,)`, where `input_shape` is the shape of the input to the network.

The `output` parameter is a numpy array of shape `(output_shape,)`, where `output_shape` is the shape of the output of the network.

# Examples

Here's an example of how to use the `run` function:

```
from deeprai.models import FeedForward

model = FeedForward()
model.add_dense(784)
model.add_dense(128, activation='relu')
model.add_dense(64, activation='relu')
model.add_dense(10, activation='softmax')
model.config(loss='categorical cross entropy')

input_data = np.random.rand(784)
output_data = model.run(input_data)
```

This code creates a `FeedForward` model with a single dense layer of size `784`, followed by two additional dense layers with ReLU activation functions, and a final dense layer with a softmax activation function. The `config` function sets the optimizer to `gradient descent` and the loss function to `categorical cross entropy`.

The `run` function takes a single input data of shape `(784,)` and returns the output of the network as a numpy array of shape `(10,)`.

---

Revision #1

Created 7 April 2023 07:47:48 by Kieran Carter

Updated 7 April 2023 07:49:49 by Kieran Carter