Lecture_17_TensorFlow_code

October 7, 2024

Introduction to TensorFlow

1 1. Tensors

1.0.1 Checking the version of Tensorflow

[1]:
```
import tensorflow as tf  # now import the tensorflow module
print(tf.__version__)  # make sure the version is 2.x
```

```
2.13.0
```

[2]:
```
# you would also need numpy for tensorflow programming. Use pip install numpy
import numpy as np
from matplotlib import pyplot as plt
```

Tensorflow programming starts with **Tensors**. A tensor has a data type (e.g., float32, int32, string, etc.), a rank, and a shape (dimensionality of the matrix or array).

You simply define the value of the tensor and the datatype and you are good to go! It's worth mentioning that usually we deal with tensors of numeric data, it is quite rare to see string tensors.

1.0.2 Scalar tensors, or rank 0 tensors (i.e. they don't have dimensions)

[3]:
```
number_tensor = tf.constant(324, dtype = tf.int32)
float_tensor = tf.constant(3.567, dtype = tf.float64)

print("number_tensor", number_tensor)
print("float_tensor",float_tensor)
```

```
number_tensor tf.Tensor(324, shape=(), dtype=int32)
float_tensor tf.Tensor(3.567, shape=(), dtype=float64)
```

1.0.3 Vector tensors, or rank 1 tensors

[4]:
```
rank1_tensor = tf.constant([1.0,2,3])
print(rank1_tensor)
```

```
tf.Tensor([1. 2. 3.], shape=(3,), dtype=float32)
```

1.0.4 Matrix tensors, or rank 2 tensors

[5]:
```
rank2_tensor = tf.constant([[1,2], [3,4]])
print(rank2_tensor)
```

```
tf.Tensor(
[[1 2]
 [3 4]], shape=(2, 2), dtype=int32)
```

1.0.5 Tensor Shape and Rank

The rank of a tensor is the number of dimensions the tensor has, e.g. vector has 1 axis and is rank 1, matrix is rank 2, etc. The rank of a tensor is direclty related to the deepest level of nested lists. You can see that number_tensor doesn't have any depth and therefore, its rank is 0. Also, you can see the rank1_tensor `[1.0,2,3]` is a rank 1 tensor as the deepest level of nesting is 1 where the rank2_tensor `[[1, 2], [3, 4]]` is a rank 2 tensor as the deepest level of nesting is 2

To determine the rank of a tensor, you can call the rank function.

[6]:
```
print(tf.rank(rank1_tensor))
```

```
tf.Tensor(1, shape=(), dtype=int32)
```

[7]:
```
print(tf.rank(rank2_tensor))
```

```
tf.Tensor(2, shape=(), dtype=int32)
```

The **shape of a tensor** is the number of elements that exist in each dimension. TensorFlow will try to determine the shape of a tensor but sometimes it may be unknown. To **get the shape** of a tensor we use the shape attribute.

[8]:
```
rank2_tensor.shape
```

[8]:
```
TensorShape([2, 2])
```

1.0.6 Rank 3 tensor

[9]:
```
rank_3_tensor = tf.constant([
  [[0, 1, 2, 3, 4],
   [5, 6, 7, 8, 9]],
  [[10, 11, 12, 13, 14],
   [15, 16, 17, 18, 19]],
  [[20, 21, 22, 23, 24],
   [25, 26, 27, 28, 29]],])

print(rank_3_tensor)
```

```
tf.Tensor(
[[[ 0  1  2  3  4]
  [ 5  6  7  8  9]]
```

```
[[10 11 12 13 14]
 [15 16 17 18 19]]

[[20 21 22 23 24]
 [25 26 27 28 29]]], shape=(3, 2, 5), dtype=int32)
```

1.0.7 Tensor Creation

[10]:
```
# Creating an tensor where all entries is 1. Note here [2,2] is the shape of␣
 ↪the tensor
all_one_tensor = tf.ones([2,2])

# Creating an tensor where all entries is 0. Note here [2,2] is the shape of␣
 ↪the tensor
all_zero_tensor = tf.zeros([2,2])

# Creating an tensor from nparray, (nested) lists.
nparray = np.array([1,2])
tensor_from_nparray = tf.constant(nparray)
```

1.0.8 Changing the Tensor Shape

The number of elements of a tensor is the product of the sizes of all its shapes. There are often many shapes that have the same number of elements, making it convient to be able to change the shape of a tensor.

The following example shows how to change the shape of a tensor.

[11]:
```
# tf.ones() creates a shape [1,2,3] tensor full of ones
tensor1 = tf.ones([1,2,3])
# reshape existing data to shape [2,3,1]
tensor2 = tf.reshape(tensor1, [2,3,1])

# -1 tells the tensor to calculate the size of the dimension in that place
# this will reshape the tensor to [3,2]
tensor3 = tf.reshape(tensor2, [3, -1])

# The numer of elements in the reshaped tensor MUST match the number in the␣
 ↪original
```

[12]:
```
print(tensor1)
print ('\n')
print(tensor2)
print ('\n')
print(tensor3)
# Notice the changes in shape
```

```
tf.Tensor(
[[[1. 1. 1.]
  [1. 1. 1.]]], shape=(1, 2, 3), dtype=float32)


tf.Tensor(
[[[1.]
  [1.]
  [1.]]

 [[1.]
  [1.]
  [1.]]], shape=(2, 3, 1), dtype=float32)


tf.Tensor(
[[1. 1.]
 [1. 1.]
 [1. 1.]], shape=(3, 2), dtype=float32)
```

1.0.9 Slicing Tensors

You may be familiar with the term slice in python and its use on lists, tuples etc. Well the slice operator can be used on tensors to select specific axes or elements.

When we slice or select elements from a tensor, we can use comma seperated values inside the set of square brackets. Each subsequent value refrences a different dimension of the tensor.

Ex: tensor[dim1, dim2, dim3]

[13]:
```
# Creating a 2D tensor
matrix = [[1,2,3,4,5],
          [6,7,8,9,10],
          [11,12,13,14,15],
          [16,17,18,19,20]]

tensor = tf.constant(matrix, dtype=tf.int32)
print(tf.rank(tensor))
print(tensor.shape)
```

```
tf.Tensor(2, shape=(), dtype=int32)
(4, 5)
```

[14]:
```
# Now lets select some different rows and columns from our tensor

three = tensor[0,2]  # selects the 3rd element from the 1st row
print(three)  # -> 3

row1 = tensor[0]  # selects the first row
```

```
print(row1)

column1 = tensor[:, 0]  # selects the first column
print(column1)
```

```
tf.Tensor(3, shape=(), dtype=int32)
tf.Tensor([1 2 3 4 5], shape=(5,), dtype=int32)
tf.Tensor([ 1  6 11 16], shape=(4,), dtype=int32)
```

[15]:
```
row_2_and_4 = tensor[1::2]  # selects second and fourth row
print(row_2_and_4)

column_1_in_row_2_and_3 = tensor[1:3, 0]
print(column_1_in_row_2_and_3)
```

```
tf.Tensor(
[[ 6  7  8  9 10]
 [16 17 18 19 20]], shape=(2, 5), dtype=int32)
tf.Tensor([ 6 11], shape=(2,), dtype=int32)
```

1.0.10 Tensor Operations

[16]:
```
# We can also have common operations between tensors
vector_1 = tf.constant([1,2])
vector_2 = tf.constant([3,4])
sum = vector_1+vector_2
print('sum = ', sum)

# elementwise_product
elementwise_product = vector_1*vector_2
print('elementwise_product = ', elementwise_product)

# dot product with tensordot
dot_product = tf.tensordot(vector_1,vector_2,axes = 1)
print('dot_product = ', dot_product) # 1*3 + 2*4

# another way to do dot product
dot_product_2 = tf.squeeze( tf.expand_dims(vector_1,0) @ tf.
 ↪expand_dims(vector_2,1) )
print('dot_product_2 = ', dot_product_2) # 1*3 + 2*4

# matrix vector product
matrix_1 = tf.constant( [[1,1],[2,2]])
matrix_vector_product = tf.tensordot(matrix_1,vector_1, axes=1)
print("matrix_vector_product = ",matrix_vector_product) # [1+2, 2+4]

# Alternative way to do matrix vector product
```

```
matrix_vector_product_2 = tf.squeeze(matrix_1@tf.expand_dims(vector_1,1))
print("matrix_vector_product_2 = ",matrix_vector_product_2) # [1+2, 2+4]

# matrix matrix product
matrix_2 = tf.constant([[1,2],[1,2]])
matrix_product = matrix_1@matrix_2
print("matrix_product = ",matrix_product)
```

```
sum =  tf.Tensor([4 6], shape=(2,), dtype=int32)
elementwise_product =  tf.Tensor([3 8], shape=(2,), dtype=int32)
dot_product =  tf.Tensor(11, shape=(), dtype=int32)
dot_product_2 =  tf.Tensor(11, shape=(), dtype=int32)
matrix_vector_product =  tf.Tensor([3 6], shape=(2,), dtype=int32)
matrix_vector_product_2 =  tf.Tensor([3 6], shape=(2,), dtype=int32)
matrix_product =  tf.Tensor(
[[2 4]
 [4 8]], shape=(2, 2), dtype=int32)
```

2 2. Building ML Models with Keras API

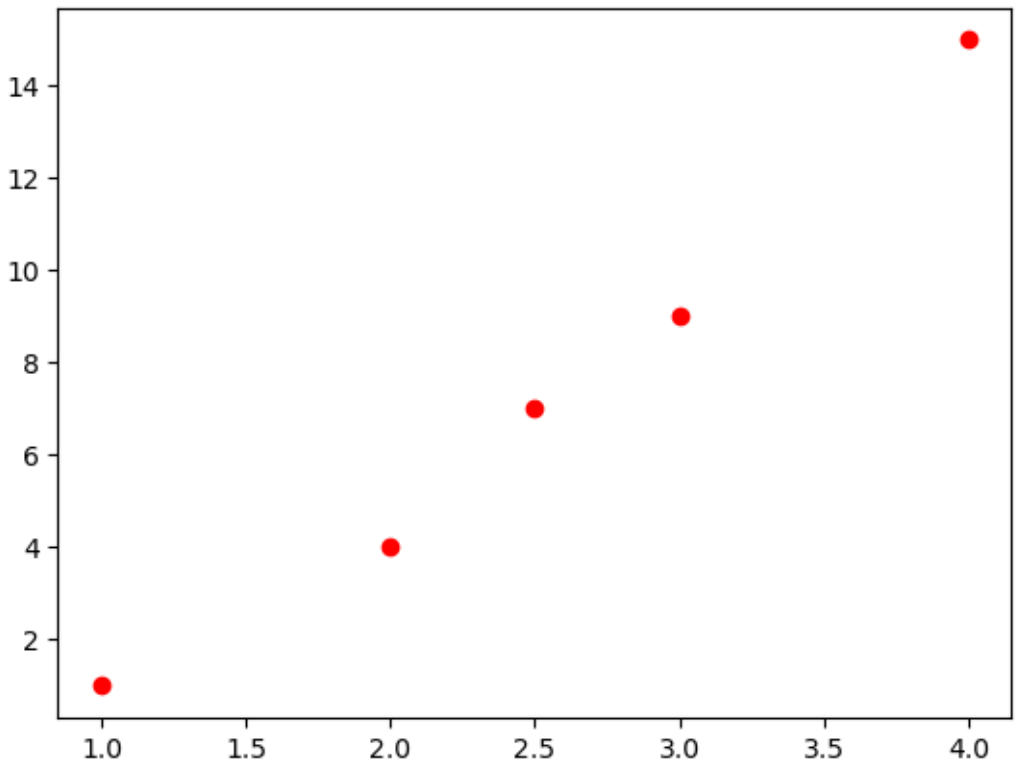
2.1 2.1 Linear Regression Model

2.1.1 Prepare training data

Consider fitting a linear curve to a data set where there is one feature of numeric type, and the output (target) is also a numeric type. Let's use the following synthetic data as an example.

[17]:
```
import matplotlib.pyplot as plt
import numpy as np

x = np.array([ 1, 2, 2.5, 3, 4],dtype=float)
x = tf.constant(x[:,np.newaxis],dtype=float)
y = np.array([1, 4, 7, 9, 15],dtype=float)
y = tf.constant(y[:,np.newaxis],dtype=float)
plt.plot(x, y, 'ro')
```

[17]:
```
[<matplotlib.lines.Line2D at 0x2822824f0>]
```



2.1.2 Creating keras.Sequential() model

One simple way to create a keras model is the `keras.Sequential()` API. We will first create an empty `Sequential` model. For the purpose of linear regression, we will add a `Dense(1)` layer, here the `1` means the output of this layer is only 1 dimensional (as our target output is only a scalar). We don't need to specify the input dimension - the input dimension will be automatically inferred at runtime.

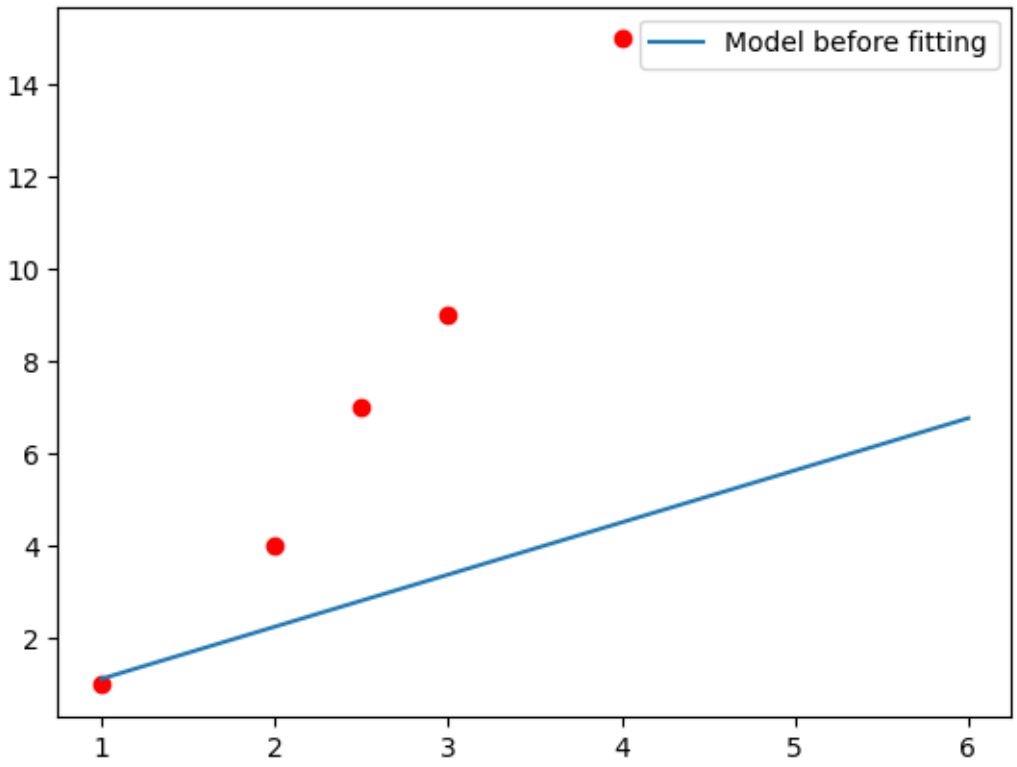
[18]:
```
from tensorflow import keras
# create a sequential layer to conduct linear regression

model = keras.Sequential()
model.add(keras.layers.Dense(1))
```

[19]:
```
# You shouldn't run model.summary() until you have pass a data to the model.
# Without passing the data to the model, the model does not know its input␣
 ↪dimension and is not 'built'.
# model.summary()
```

[19]:
```
# Let's check out what the prediction of the model looks like (Note: we haven't␣
 ↪trained the model yet)
```

```
x_mesh = np.linspace(1,6,100) # generate 100 input values between 1 and 6
x_mesh = tf.constant( x_mesh[:,np.newaxis])
y_pred_mesh = model(x_mesh) # get the output of our model
plt.plot(x, y, 'ro')
plt.plot(x_mesh, y_pred_mesh , label = "Model before fitting")
plt.legend()
```

[19]: <matplotlib.legend.Legend at 0x2824e0df0>



[21]:
```
# model.summary() is a useful tool to visualize the structure of the model

model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (100, 1)                  2

=================================================================
Total params: 2 (8.00 Byte)
Trainable params: 2 (8.00 Byte)
```

```
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

2.1.3 Train the model

Training a `keras` model consists of two steps. The first step is `compile`, in which you need to specify the optimizer and the loss. Then, we call the `fit` method.

[20]:
```
model.compile(optimizer = keras.optimizers.SGD(), loss = keras.losses.
  ↪MeanSquaredError())
model.fit(x,y,epochs=30)
```

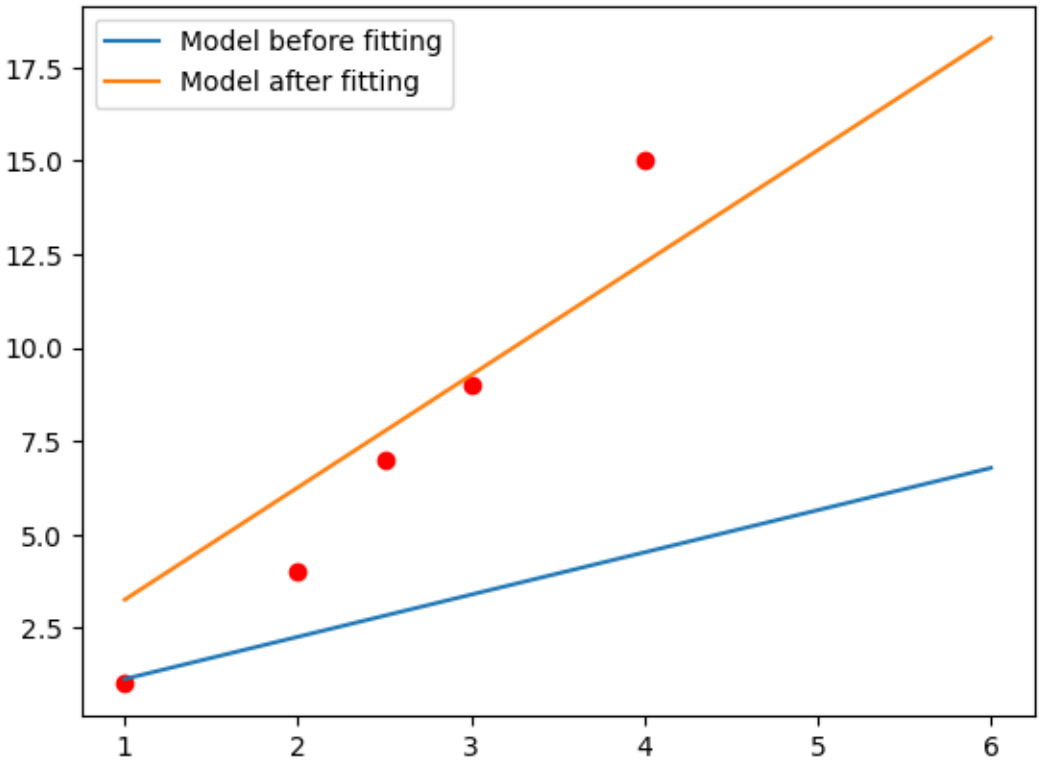
```
WARNING:absl:At this time, the v2.11+ optimizer `tf.keras.optimizers.SGD` runs
slowly on M1/M2 Macs, please use the legacy Keras optimizer instead, located at
`tf.keras.optimizers.legacy.SGD`.
WARNING:absl:There is a known slowdown when using v2.11+ Keras optimizers on
M1/M2 Macs. Falling back to the legacy Keras optimizer, i.e.,
`tf.keras.optimizers.legacy.SGD`.

Epoch 1/30
1/1 [==============================] - 0s 109ms/step - loss: 32.3518
Epoch 2/30
1/1 [==============================] - 0s 1ms/step - loss: 23.9135
Epoch 3/30
1/1 [==============================] - 0s 1ms/step - loss: 17.9901
Epoch 4/30
1/1 [==============================] - 0s 1ms/step - loss: 13.8304
Epoch 5/30
1/1 [==============================] - 0s 1ms/step - loss: 10.9078
Epoch 6/30
1/1 [==============================] - 0s 1ms/step - loss: 8.8528
Epoch 7/30
1/1 [==============================] - 0s 997us/step - loss: 7.4064
Epoch 8/30
1/1 [==============================] - 0s 1ms/step - loss: 6.3867
Epoch 9/30
1/1 [==============================] - 0s 2ms/step - loss: 5.6664
Epoch 10/30
1/1 [==============================] - 0s 2ms/step - loss: 5.1561
Epoch 11/30
1/1 [==============================] - 0s 1ms/step - loss: 4.7930
Epoch 12/30
1/1 [==============================] - 0s 1ms/step - loss: 4.5333
Epoch 13/30
1/1 [==============================] - 0s 1ms/step - loss: 4.3461
Epoch 14/30
1/1 [==============================] - 0s 1ms/step - loss: 4.2098
Epoch 15/30
1/1 [==============================] - 0s 1ms/step - loss: 4.1091
```

```
Epoch 16/30
1/1 [==============================] - 0s 1ms/step - loss: 4.0336
Epoch 17/30
1/1 [==============================] - 0s 1ms/step - loss: 3.9756
Epoch 18/30
1/1 [==============================] - 0s 1ms/step - loss: 3.9300
Epoch 19/30
1/1 [==============================] - 0s 1ms/step - loss: 3.8931
Epoch 20/30
1/1 [==============================] - 0s 2ms/step - loss: 3.8623
Epoch 21/30
1/1 [==============================] - 0s 2ms/step - loss: 3.8358
Epoch 22/30
1/1 [==============================] - 0s 2ms/step - loss: 3.8124
Epoch 23/30
1/1 [==============================] - 0s 1ms/step - loss: 3.7911
Epoch 24/30
1/1 [==============================] - 0s 1000us/step - loss: 3.7714
Epoch 25/30
1/1 [==============================] - 0s 959us/step - loss: 3.7528
Epoch 26/30
1/1 [==============================] - 0s 1ms/step - loss: 3.7350
Epoch 27/30
1/1 [==============================] - 0s 1ms/step - loss: 3.7178
Epoch 28/30
1/1 [==============================] - 0s 1ms/step - loss: 3.7010
Epoch 29/30
1/1 [==============================] - 0s 1ms/step - loss: 3.6846
Epoch 30/30
1/1 [==============================] - 0s 1ms/step - loss: 3.6684
```

[20]:
```
<keras.src.callbacks.History at 0x282634ac0>
```

[21]:
```
### Let's see how the model looks like after training

y_pred_mesh_afterfitting = model(x_mesh)
plt.plot(x, y, 'ro')
plt.plot(x_mesh, y_pred_mesh , label = "Model before fitting")
plt.plot(x_mesh, y_pred_mesh_afterfitting , label = "Model after fitting")
plt.legend()
```

[21]:
```
<matplotlib.legend.Legend at 0x2826dbeb0>
```

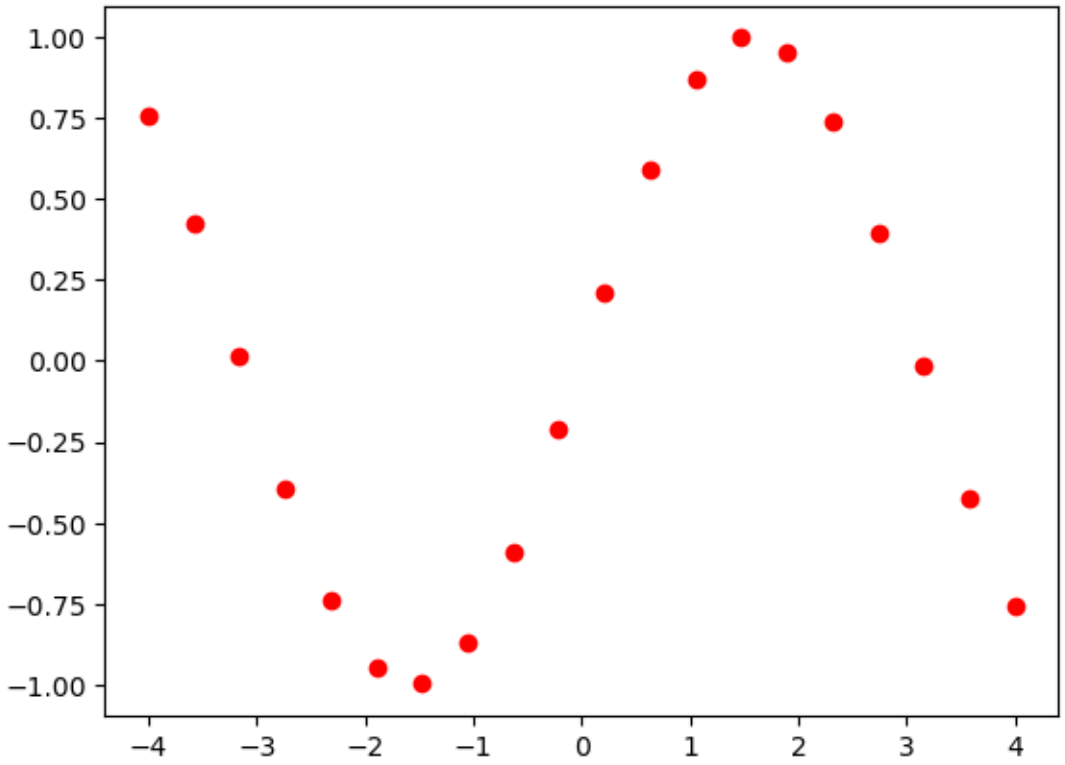


2.2 2.2 Building Simple Neural Network with Keras

2.2.1 Prepare training data

Let's use a neural network to fit a nonlinear function. We create the training data first which is a sinusoidal curve.

[22]:
```
x  = np.linspace(-4,4,20)
x = x[:,np.newaxis]
y = np.sin(x)

plt.figure()
plt.plot(x,y,'ro')

x = tf.constant(x)
y = tf.constant(y)
```



2.2.2 Creating keras.Sequential() model

To create a neural network, you just need to keep adding layers to the sequential model. In each layer, you need to specify the dimension of the output (in other words, the width of this layer), and the activation, which we use 'relu'.

[23]:
```
model = keras.Sequential()

model.add(keras.layers.Dense(20,activation='relu'))
model.add(keras.layers.Dense(20,activation='relu'))
model.add(keras.layers.Dense(20,activation='relu'))

model.add(keras.layers.Dense(1))
```

2.2.3 Creating keras.Sequential() model - method 2

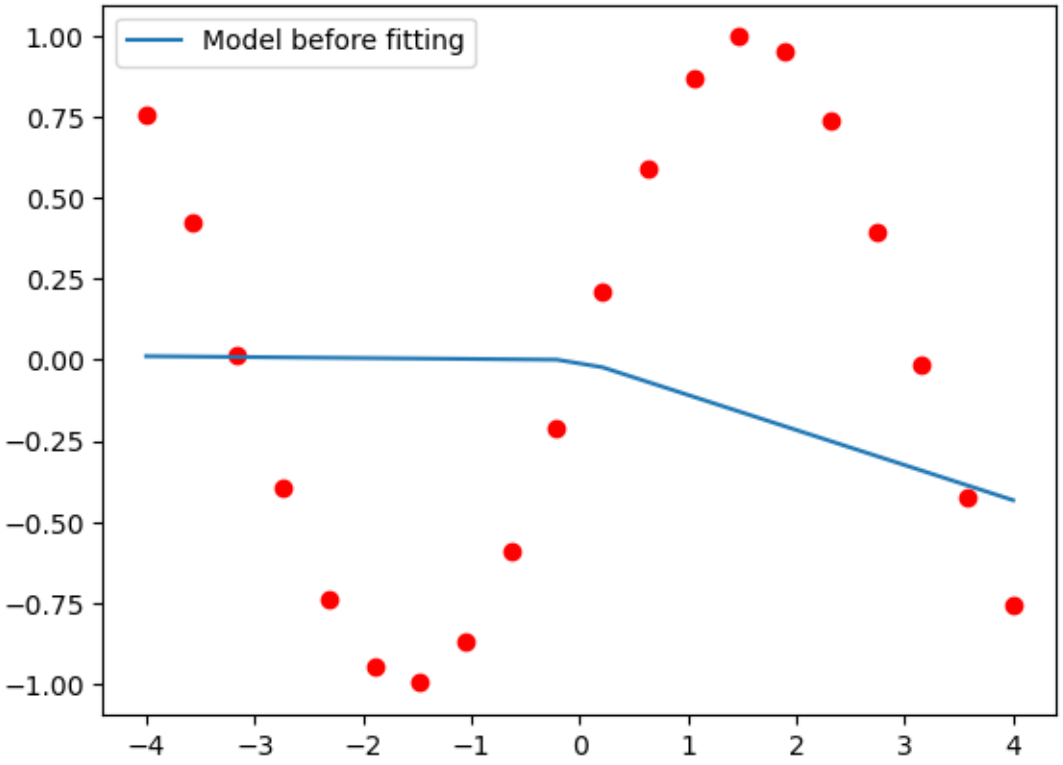
[24]:
```
model2 = keras.Sequential(layers=[keras.layers.Dense(20,activation='relu'),
            keras.layers.Dense(20,activation='relu'),
            keras.layers.Dense(20,activation='relu'),
            keras.layers.Dense(1)])
```

2.2.4 Creating keras model with functional API

[25]:
```
input = keras.Input(shape = (1))

intermediate_1 = keras.layers.Dense(20,activation='relu')(input)
intermediate_2 = keras.layers.Dense(20,activation='relu')(intermediate_1)
intermediate_3 = keras.layers.Dense(20,activation='relu')(intermediate_2)

output = keras.layers.Dense(1)(intermediate_3)

model3 = keras.Model(inputs = input,outputs = output)
```

[26]:
```
y_pred_before_training = model(x) # get the output of our model
plt.plot(x, y, 'ro')
plt.plot(x, y_pred_before_training , label = "Model before fitting")
plt.legend()
```

[26]:
```
<matplotlib.legend.Legend at 0x28380abb0>
```



[27]:
```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
```

```
 Layer (type)                Output Shape              Param #
=================================================================
 dense_1 (Dense)             (20, 20)                  40

 dense_2 (Dense)             (20, 20)                  420

 dense_3 (Dense)             (20, 20)                  420

 dense_4 (Dense)             (20, 1)                   21

=================================================================
Total params: 901 (3.52 KB)
Trainable params: 901 (3.52 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

[28]:
```
# training the model by calling the compile and the fit function
model.compile(optimizer = keras.optimizers.SGD(),loss=keras.losses.
  ↪MeanSquaredError())
model.fit(x,y,epochs=1000)
```

```
WARNING:absl:At this time, the v2.11+ optimizer `tf.keras.optimizers.SGD` runs
slowly on M1/M2 Macs, please use the legacy Keras optimizer instead, located at
`tf.keras.optimizers.legacy.SGD`.
WARNING:absl:There is a known slowdown when using v2.11+ Keras optimizers on
M1/M2 Macs. Falling back to the legacy Keras optimizer, i.e.,
`tf.keras.optimizers.legacy.SGD`.

Epoch 1/1000
1/1 [==============================] - 0s 105ms/step - loss: 0.5137
Epoch 2/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.5028
Epoch 3/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4931
Epoch 4/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.4844
Epoch 5/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4767
Epoch 6/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.4697
Epoch 7/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.4636
Epoch 8/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.4584
Epoch 9/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.4538
Epoch 10/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4498
```

```
Epoch 11/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4467
Epoch 12/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4440
Epoch 13/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4416
Epoch 14/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4397
Epoch 15/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4379
Epoch 16/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.4361
Epoch 17/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4346
Epoch 18/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.4332
Epoch 19/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4319
Epoch 20/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4306
Epoch 21/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4294
Epoch 22/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4282
Epoch 23/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4268
Epoch 24/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.4255
Epoch 25/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.4244
Epoch 26/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.4233
Epoch 27/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4222
Epoch 28/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4212
Epoch 29/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.4201
Epoch 30/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4191
Epoch 31/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4181
Epoch 32/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4172
Epoch 33/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.4162
Epoch 34/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.4153
```

```
Epoch 35/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4143
Epoch 36/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4134
Epoch 37/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4124
Epoch 38/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4114
Epoch 39/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4104
Epoch 40/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.4093
Epoch 41/1000
1/1 [==============================] - 0s 10ms/step - loss: 0.4083
Epoch 42/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.4074
Epoch 43/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4066
Epoch 44/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4058
Epoch 45/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.4051
Epoch 46/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.4043
Epoch 47/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4035
Epoch 48/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4027
Epoch 49/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4019
Epoch 50/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4011
Epoch 51/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.4004
Epoch 52/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3996
Epoch 53/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.3989
Epoch 54/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3981
Epoch 55/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3974
Epoch 56/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3966
Epoch 57/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3959
Epoch 58/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3951
```

```
Epoch 59/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3944
Epoch 60/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3936
Epoch 61/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3929
Epoch 62/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3922
Epoch 63/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3916
Epoch 64/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3908
Epoch 65/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3902
Epoch 66/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3895
Epoch 67/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3888
Epoch 68/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3881
Epoch 69/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3874
Epoch 70/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3867
Epoch 71/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3860
Epoch 72/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3852
Epoch 73/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3845
Epoch 74/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3838
Epoch 75/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3831
Epoch 76/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3824
Epoch 77/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3816
Epoch 78/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3808
Epoch 79/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3801
Epoch 80/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3793
Epoch 81/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3785
Epoch 82/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3776
```

```
Epoch 83/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3768
Epoch 84/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3760
Epoch 85/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3752
Epoch 86/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3745
Epoch 87/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3738
Epoch 88/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3731
Epoch 89/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3724
Epoch 90/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3717
Epoch 91/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3710
Epoch 92/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3703
Epoch 93/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3696
Epoch 94/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3688
Epoch 95/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3681
Epoch 96/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3674
Epoch 97/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3667
Epoch 98/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.3660
Epoch 99/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3653
Epoch 100/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3646
Epoch 101/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3639
Epoch 102/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3632
Epoch 103/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3625
Epoch 104/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3618
Epoch 105/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3611
Epoch 106/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3603
```

```
Epoch 107/1000
1/1 [==============================] - 0s 10ms/step - loss: 0.3596
Epoch 108/1000
1/1 [==============================] - 0s 4ms/step - loss: 0.3589
Epoch 109/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.3582
Epoch 110/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3575
Epoch 111/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3568
Epoch 112/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3561
Epoch 113/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3554
Epoch 114/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3547
Epoch 115/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3540
Epoch 116/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3533
Epoch 117/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3527
Epoch 118/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3520
Epoch 119/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3513
Epoch 120/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3506
Epoch 121/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3499
Epoch 122/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3493
Epoch 123/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3486
Epoch 124/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3479
Epoch 125/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3472
Epoch 126/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3466
Epoch 127/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3459
Epoch 128/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3452
Epoch 129/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3445
Epoch 130/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3438
```

```
Epoch 131/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3431
Epoch 132/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3424
Epoch 133/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3418
Epoch 134/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3411
Epoch 135/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3404
Epoch 136/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3397
Epoch 137/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3390
Epoch 138/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3383
Epoch 139/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3376
Epoch 140/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3369
Epoch 141/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3362
Epoch 142/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3355
Epoch 143/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3348
Epoch 144/1000
1/1 [==============================] - 0s 4ms/step - loss: 0.3341
Epoch 145/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3334
Epoch 146/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3327
Epoch 147/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3320
Epoch 148/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3313
Epoch 149/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3306
Epoch 150/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3298
Epoch 151/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.3291
Epoch 152/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3284
Epoch 153/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3277
Epoch 154/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3270
```

```
Epoch 155/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3263
Epoch 156/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3255
Epoch 157/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3248
Epoch 158/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3241
Epoch 159/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3233
Epoch 160/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3226
Epoch 161/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3219
Epoch 162/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3211
Epoch 163/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3204
Epoch 164/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3197
Epoch 165/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3189
Epoch 166/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3182
Epoch 167/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3175
Epoch 168/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3167
Epoch 169/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3160
Epoch 170/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3152
Epoch 171/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3145
Epoch 172/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3137
Epoch 173/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3130
Epoch 174/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3122
Epoch 175/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.3115
Epoch 176/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3107
Epoch 177/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3100
Epoch 178/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3092
```

```
Epoch 179/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3085
Epoch 180/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3077
Epoch 181/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.3069
Epoch 182/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.3062
Epoch 183/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.3054
Epoch 184/1000
1/1 [==============================] - 0s 5ms/step - loss: 0.3047
Epoch 185/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3039
Epoch 186/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3032
Epoch 187/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.3024
Epoch 188/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.3016
Epoch 189/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.3008
Epoch 190/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.3000
Epoch 191/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2992
Epoch 192/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2984
Epoch 193/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2976
Epoch 194/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2969
Epoch 195/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2960
Epoch 196/1000
1/1 [==============================] - 0s 25ms/step - loss: 0.2953
Epoch 197/1000
1/1 [==============================] - 0s 4ms/step - loss: 0.2945
Epoch 198/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2937
Epoch 199/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2929
Epoch 200/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2921
Epoch 201/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2912
Epoch 202/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2905
```

```
Epoch 203/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2896
Epoch 204/1000
1/1 [==============================] - 0s 5ms/step - loss: 0.2888
Epoch 205/1000
1/1 [==============================] - 0s 5ms/step - loss: 0.2880
Epoch 206/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2872
Epoch 207/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2863
Epoch 208/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2855
Epoch 209/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2846
Epoch 210/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2838
Epoch 211/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2829
Epoch 212/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2820
Epoch 213/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2812
Epoch 214/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2803
Epoch 215/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2795
Epoch 216/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2785
Epoch 217/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2776
Epoch 218/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2767
Epoch 219/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2758
Epoch 220/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2749
Epoch 221/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2740
Epoch 222/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2730
Epoch 223/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2721
Epoch 224/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2711
Epoch 225/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2702
Epoch 226/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2693
```

```
Epoch 227/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2683
Epoch 228/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2674
Epoch 229/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2665
Epoch 230/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2656
Epoch 231/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2647
Epoch 232/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2639
Epoch 233/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2630
Epoch 234/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2622
Epoch 235/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2613
Epoch 236/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2605
Epoch 237/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2597
Epoch 238/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2589
Epoch 239/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2580
Epoch 240/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2572
Epoch 241/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2564
Epoch 242/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2555
Epoch 243/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2547
Epoch 244/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2539
Epoch 245/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2530
Epoch 246/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2522
Epoch 247/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2514
Epoch 248/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2506
Epoch 249/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2497
Epoch 250/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2489
```

```
Epoch 251/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2481
Epoch 252/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2472
Epoch 253/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2464
Epoch 254/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2455
Epoch 255/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2447
Epoch 256/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2439
Epoch 257/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2430
Epoch 258/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2422
Epoch 259/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2413
Epoch 260/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2405
Epoch 261/1000
1/1 [==============================] - 0s 4ms/step - loss: 0.2397
Epoch 262/1000
1/1 [==============================] - 0s 26ms/step - loss: 0.2388
Epoch 263/1000
1/1 [==============================] - 0s 13ms/step - loss: 0.2380
Epoch 264/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.2371
Epoch 265/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.2363
Epoch 266/1000
1/1 [==============================] - 0s 5ms/step - loss: 0.2354
Epoch 267/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2346
Epoch 268/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2338
Epoch 269/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2329
Epoch 270/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2321
Epoch 271/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2312
Epoch 272/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2304
Epoch 273/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2295
Epoch 274/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2287
```

```
Epoch 275/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2278
Epoch 276/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2270
Epoch 277/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2262
Epoch 278/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2253
Epoch 279/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2245
Epoch 280/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2236
Epoch 281/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.2228
Epoch 282/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2220
Epoch 283/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2211
Epoch 284/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2203
Epoch 285/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2194
Epoch 286/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2186
Epoch 287/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2177
Epoch 288/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2169
Epoch 289/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2160
Epoch 290/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2152
Epoch 291/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2143
Epoch 292/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2135
Epoch 293/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2126
Epoch 294/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2118
Epoch 295/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2109
Epoch 296/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2100
Epoch 297/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2092
Epoch 298/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.2083
```

```
Epoch 299/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2075
Epoch 300/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2066
Epoch 301/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2057
Epoch 302/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2048
Epoch 303/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2040
Epoch 304/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2031
Epoch 305/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.2022
Epoch 306/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.2014
Epoch 307/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.2005
Epoch 308/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1997
Epoch 309/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1988
Epoch 310/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1979
Epoch 311/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1971
Epoch 312/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1963
Epoch 313/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1954
Epoch 314/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1946
Epoch 315/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1937
Epoch 316/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1929
Epoch 317/1000
1/1 [==============================] - 0s 18ms/step - loss: 0.1920
Epoch 318/1000
1/1 [==============================] - 0s 4ms/step - loss: 0.1912
Epoch 319/1000
1/1 [==============================] - 0s 6ms/step - loss: 0.1904
Epoch 320/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1895
Epoch 321/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1887
Epoch 322/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1878
```

```
Epoch 323/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1870
Epoch 324/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1862
Epoch 325/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.1853
Epoch 326/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1845
Epoch 327/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1837
Epoch 328/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1829
Epoch 329/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1820
Epoch 330/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1812
Epoch 331/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1804
Epoch 332/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1796
Epoch 333/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1787
Epoch 334/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1779
Epoch 335/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1771
Epoch 336/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1763
Epoch 337/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1755
Epoch 338/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1746
Epoch 339/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1738
Epoch 340/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1730
Epoch 341/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1722
Epoch 342/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1714
Epoch 343/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1706
Epoch 344/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1698
Epoch 345/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1690
Epoch 346/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1682
```

```
Epoch 347/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1674
Epoch 348/1000
1/1 [==============================] - 0s 6ms/step - loss: 0.1666
Epoch 349/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.1658
Epoch 350/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1650
Epoch 351/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1642
Epoch 352/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1634
Epoch 353/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1626
Epoch 354/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1618
Epoch 355/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.1610
Epoch 356/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1602
Epoch 357/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1594
Epoch 358/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1586
Epoch 359/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1579
Epoch 360/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1571
Epoch 361/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1563
Epoch 362/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1555
Epoch 363/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1547
Epoch 364/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1540
Epoch 365/1000
1/1 [==============================] - 0s 23ms/step - loss: 0.1532
Epoch 366/1000
1/1 [==============================] - 0s 6ms/step - loss: 0.1524
Epoch 367/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1517
Epoch 368/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1509
Epoch 369/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1501
Epoch 370/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1494
```

```
Epoch 371/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1486
Epoch 372/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1479
Epoch 373/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1471
Epoch 374/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1463
Epoch 375/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.1456
Epoch 376/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1448
Epoch 377/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1441
Epoch 378/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1433
Epoch 379/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1426
Epoch 380/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1418
Epoch 381/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1411
Epoch 382/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1404
Epoch 383/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1396
Epoch 384/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1389
Epoch 385/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1381
Epoch 386/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1374
Epoch 387/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1367
Epoch 388/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1359
Epoch 389/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1352
Epoch 390/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1345
Epoch 391/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1338
Epoch 392/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1330
Epoch 393/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1323
Epoch 394/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1316
```

```
Epoch 395/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1309
Epoch 396/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1302
Epoch 397/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1295
Epoch 398/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1288
Epoch 399/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1281
Epoch 400/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1274
Epoch 401/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1267
Epoch 402/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1260
Epoch 403/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1253
Epoch 404/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1246
Epoch 405/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1239
Epoch 406/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1232
Epoch 407/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1225
Epoch 408/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1218
Epoch 409/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1212
Epoch 410/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1205
Epoch 411/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1198
Epoch 412/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1191
Epoch 413/1000
1/1 [==============================] - 0s 22ms/step - loss: 0.1185
Epoch 414/1000
1/1 [==============================] - 0s 5ms/step - loss: 0.1178
Epoch 415/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1172
Epoch 416/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1165
Epoch 417/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1158
Epoch 418/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1152
```

```
Epoch 419/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1146
Epoch 420/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1139
Epoch 421/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1133
Epoch 422/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1126
Epoch 423/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1120
Epoch 424/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1114
Epoch 425/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1107
Epoch 426/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1101
Epoch 427/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1095
Epoch 428/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1089
Epoch 429/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1082
Epoch 430/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1076
Epoch 431/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1070
Epoch 432/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1064
Epoch 433/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1058
Epoch 434/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1052
Epoch 435/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1046
Epoch 436/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1040
Epoch 437/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1034
Epoch 438/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1028
Epoch 439/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1022
Epoch 440/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.1016
Epoch 441/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.1010
Epoch 442/1000
1/1 [==============================] - 0s 4ms/step - loss: 0.1004
```

```
Epoch 443/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0999
Epoch 444/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0993
Epoch 445/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0987
Epoch 446/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0981
Epoch 447/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0976
Epoch 448/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0970
Epoch 449/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0964
Epoch 450/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0958
Epoch 451/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0953
Epoch 452/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0947
Epoch 453/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0942
Epoch 454/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0936
Epoch 455/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0930
Epoch 456/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0925
Epoch 457/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0919
Epoch 458/1000
1/1 [==============================] - 0s 23ms/step - loss: 0.0914
Epoch 459/1000
1/1 [==============================] - 0s 5ms/step - loss: 0.0909
Epoch 460/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0903
Epoch 461/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0898
Epoch 462/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0893
Epoch 463/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0887
Epoch 464/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0882
Epoch 465/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0877
Epoch 466/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0872
```

```
Epoch 467/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0866
Epoch 468/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0861
Epoch 469/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0856
Epoch 470/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0851
Epoch 471/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0846
Epoch 472/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0841
Epoch 473/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0836
Epoch 474/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0831
Epoch 475/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0826
Epoch 476/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0821
Epoch 477/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0816
Epoch 478/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0811
Epoch 479/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0806
Epoch 480/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0801
Epoch 481/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0797
Epoch 482/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0792
Epoch 483/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0787
Epoch 484/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0782
Epoch 485/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0778
Epoch 486/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0773
Epoch 487/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0768
Epoch 488/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0764
Epoch 489/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0759
Epoch 490/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0754
```

```
Epoch 491/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0750
Epoch 492/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0745
Epoch 493/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0741
Epoch 494/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0736
Epoch 495/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0732
Epoch 496/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0727
Epoch 497/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0723
Epoch 498/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0718
Epoch 499/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0714
Epoch 500/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0710
Epoch 501/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0705
Epoch 502/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0701
Epoch 503/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0697
Epoch 504/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0693
Epoch 505/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0688
Epoch 506/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0684
Epoch 507/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0680
Epoch 508/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0676
Epoch 509/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0672
Epoch 510/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0668
Epoch 511/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0663
Epoch 512/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0659
Epoch 513/1000
1/1 [==============================] - 0s 30ms/step - loss: 0.0655
Epoch 514/1000
1/1 [==============================] - 0s 9ms/step - loss: 0.0651
```

```
Epoch 515/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0647
Epoch 516/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0643
Epoch 517/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0639
Epoch 518/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0635
Epoch 519/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0632
Epoch 520/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0628
Epoch 521/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0624
Epoch 522/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0620
Epoch 523/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0616
Epoch 524/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0613
Epoch 525/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0609
Epoch 526/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0605
Epoch 527/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0602
Epoch 528/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0598
Epoch 529/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0594
Epoch 530/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0591
Epoch 531/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0587
Epoch 532/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0584
Epoch 533/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0580
Epoch 534/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0577
Epoch 535/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0573
Epoch 536/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0570
Epoch 537/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0566
Epoch 538/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0563
```

```
Epoch 539/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0559
Epoch 540/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0556
Epoch 541/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0553
Epoch 542/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0549
Epoch 543/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0546
Epoch 544/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0543
Epoch 545/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0540
Epoch 546/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0536
Epoch 547/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0533
Epoch 548/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0530
Epoch 549/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0527
Epoch 550/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0523
Epoch 551/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0520
Epoch 552/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0517
Epoch 553/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0514
Epoch 554/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0511
Epoch 555/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0508
Epoch 556/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0505
Epoch 557/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0502
Epoch 558/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0499
Epoch 559/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0496
Epoch 560/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0493
Epoch 561/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0490
Epoch 562/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0487
```

```
Epoch 563/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0484
Epoch 564/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0481
Epoch 565/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0478
Epoch 566/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0475
Epoch 567/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0472
Epoch 568/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0470
Epoch 569/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0467
Epoch 570/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0464
Epoch 571/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0461
Epoch 572/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0458
Epoch 573/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0456
Epoch 574/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0453
Epoch 575/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0450
Epoch 576/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0447
Epoch 577/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0445
Epoch 578/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0442
Epoch 579/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0439
Epoch 580/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0437
Epoch 581/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0434
Epoch 582/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0432
Epoch 583/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0429
Epoch 584/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0426
Epoch 585/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0424
Epoch 586/1000
1/1 [==============================] - 0s 21ms/step - loss: 0.0421
```

```
Epoch 587/1000
1/1 [==============================] - 0s 7ms/step - loss: 0.0419
Epoch 588/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0416
Epoch 589/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0414
Epoch 590/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0411
Epoch 591/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0409
Epoch 592/1000
1/1 [==============================] - 0s 4ms/step - loss: 0.0407
Epoch 593/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0404
Epoch 594/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0402
Epoch 595/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0399
Epoch 596/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0397
Epoch 597/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0395
Epoch 598/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0392
Epoch 599/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0390
Epoch 600/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0388
Epoch 601/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0385
Epoch 602/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0383
Epoch 603/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0381
Epoch 604/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0379
Epoch 605/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0376
Epoch 606/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0374
Epoch 607/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0372
Epoch 608/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0370
Epoch 609/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0368
Epoch 610/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0365
```

```
Epoch 611/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0363
Epoch 612/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0361
Epoch 613/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0359
Epoch 614/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0357
Epoch 615/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0355
Epoch 616/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0353
Epoch 617/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0351
Epoch 618/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0349
Epoch 619/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0347
Epoch 620/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0345
Epoch 621/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0343
Epoch 622/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0341
Epoch 623/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0339
Epoch 624/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0337
Epoch 625/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0335
Epoch 626/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0333
Epoch 627/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0331
Epoch 628/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0329
Epoch 629/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0327
Epoch 630/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0325
Epoch 631/1000
1/1 [==============================] - 0s 4ms/step - loss: 0.0323
Epoch 632/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0321
Epoch 633/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0319
Epoch 634/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0318
```

```
Epoch 635/1000
1/1 [==============================] - 0s 29ms/step - loss: 0.0316
Epoch 636/1000
1/1 [==============================] - 0s 6ms/step - loss: 0.0314
Epoch 637/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0312
Epoch 638/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0310
Epoch 639/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0309
Epoch 640/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0307
Epoch 641/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0305
Epoch 642/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0303
Epoch 643/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0302
Epoch 644/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0300
Epoch 645/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0298
Epoch 646/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0296
Epoch 647/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0295
Epoch 648/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0293
Epoch 649/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0291
Epoch 650/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0290
Epoch 651/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0288
Epoch 652/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0287
Epoch 653/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0285
Epoch 654/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0283
Epoch 655/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0282
Epoch 656/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0280
Epoch 657/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0279
Epoch 658/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0277
```

```
Epoch 659/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0275
Epoch 660/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0274
Epoch 661/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0272
Epoch 662/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0271
Epoch 663/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0269
Epoch 664/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0268
Epoch 665/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0266
Epoch 666/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0265
Epoch 667/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0263
Epoch 668/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0262
Epoch 669/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0260
Epoch 670/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0259
Epoch 671/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0258
Epoch 672/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0256
Epoch 673/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0255
Epoch 674/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0253
Epoch 675/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0252
Epoch 676/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0251
Epoch 677/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0249
Epoch 678/1000
1/1 [==============================] - 0s 18ms/step - loss: 0.0248
Epoch 679/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0246
Epoch 680/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0245
Epoch 681/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0244
Epoch 682/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0242
```

```
Epoch 683/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0241
Epoch 684/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0240
Epoch 685/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0239
Epoch 686/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0237
Epoch 687/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0236
Epoch 688/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0235
Epoch 689/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0234
Epoch 690/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0232
Epoch 691/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0231
Epoch 692/1000
1/1 [==============================] - 0s 4ms/step - loss: 0.0230
Epoch 693/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0229
Epoch 694/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0228
Epoch 695/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0226
Epoch 696/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0225
Epoch 697/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0224
Epoch 698/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0223
Epoch 699/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0222
Epoch 700/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0221
Epoch 701/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0219
Epoch 702/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0218
Epoch 703/1000
1/1 [==============================] - 0s 986us/step - loss: 0.0217
Epoch 704/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0216
Epoch 705/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0215
Epoch 706/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0214
```

```
Epoch 707/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0213
Epoch 708/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0212
Epoch 709/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0211
Epoch 710/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0209
Epoch 711/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0208
Epoch 712/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0207
Epoch 713/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0206
Epoch 714/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0205
Epoch 715/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0204
Epoch 716/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0203
Epoch 717/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0202
Epoch 718/1000
1/1 [==============================] - 0s 28ms/step - loss: 0.0201
Epoch 719/1000
1/1 [==============================] - 0s 5ms/step - loss: 0.0200
Epoch 720/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0199
Epoch 721/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0198
Epoch 722/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0197
Epoch 723/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0196
Epoch 724/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0195
Epoch 725/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0194
Epoch 726/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0193
Epoch 727/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0192
Epoch 728/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0191
Epoch 729/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0190
Epoch 730/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0189
```

```
Epoch 731/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0188
Epoch 732/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0187
Epoch 733/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0186
Epoch 734/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0185
Epoch 735/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0184
Epoch 736/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0184
Epoch 737/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0183
Epoch 738/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0182
Epoch 739/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0181
Epoch 740/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0180
Epoch 741/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0179
Epoch 742/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0178
Epoch 743/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0177
Epoch 744/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0176
Epoch 745/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0176
Epoch 746/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0175
Epoch 747/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0174
Epoch 748/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0173
Epoch 749/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0172
Epoch 750/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0171
Epoch 751/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0170
Epoch 752/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0170
Epoch 753/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0169
Epoch 754/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0168
```

```
Epoch 755/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0167
Epoch 756/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0166
Epoch 757/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0165
Epoch 758/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0165
Epoch 759/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0164
Epoch 760/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0163
Epoch 761/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0162
Epoch 762/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0161
Epoch 763/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0161
Epoch 764/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0160
Epoch 765/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0159
Epoch 766/1000
1/1 [==============================] - 0s 24ms/step - loss: 0.0158
Epoch 767/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0157
Epoch 768/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0157
Epoch 769/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0156
Epoch 770/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0155
Epoch 771/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0154
Epoch 772/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0154
Epoch 773/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0153
Epoch 774/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0152
Epoch 775/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0151
Epoch 776/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0151
Epoch 777/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0150
Epoch 778/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0149
```

```
Epoch 779/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0148
Epoch 780/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0148
Epoch 781/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0147
Epoch 782/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0146
Epoch 783/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0146
Epoch 784/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0145
Epoch 785/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0144
Epoch 786/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0143
Epoch 787/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0143
Epoch 788/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0142
Epoch 789/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0141
Epoch 790/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0141
Epoch 791/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0140
Epoch 792/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0139
Epoch 793/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0139
Epoch 794/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0138
Epoch 795/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0137
Epoch 796/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0137
Epoch 797/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0136
Epoch 798/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0135
Epoch 799/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0135
Epoch 800/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0134
Epoch 801/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0133
Epoch 802/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0133
```

```
Epoch 803/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0132
Epoch 804/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0131
Epoch 805/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0131
Epoch 806/1000
1/1 [==============================] - 0s 26ms/step - loss: 0.0130
Epoch 807/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0130
Epoch 808/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0129
Epoch 809/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0128
Epoch 810/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0128
Epoch 811/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0127
Epoch 812/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0126
Epoch 813/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0126
Epoch 814/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0125
Epoch 815/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0125
Epoch 816/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0124
Epoch 817/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0123
Epoch 818/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0123
Epoch 819/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0122
Epoch 820/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0122
Epoch 821/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0121
Epoch 822/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0121
Epoch 823/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0120
Epoch 824/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0119
Epoch 825/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0119
Epoch 826/1000
1/1 [==============================] - 0s 4ms/step - loss: 0.0118
```

```
Epoch 827/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0118
Epoch 828/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0117
Epoch 829/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0117
Epoch 830/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0116
Epoch 831/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0115
Epoch 832/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0115
Epoch 833/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0114
Epoch 834/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0114
Epoch 835/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0113
Epoch 836/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0113
Epoch 837/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0112
Epoch 838/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0112
Epoch 839/1000
1/1 [==============================] - 0s 5ms/step - loss: 0.0111
Epoch 840/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0110
Epoch 841/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0110
Epoch 842/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0109
Epoch 843/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0109
Epoch 844/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0108
Epoch 845/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0108
Epoch 846/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0107
Epoch 847/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0107
Epoch 848/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0106
Epoch 849/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0106
Epoch 850/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0105
```

```
Epoch 851/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0105
Epoch 852/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0104
Epoch 853/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0104
Epoch 854/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0103
Epoch 855/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0103
Epoch 856/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0102
Epoch 857/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0102
Epoch 858/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0101
Epoch 859/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0101
Epoch 860/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0100
Epoch 861/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0100
Epoch 862/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0099
Epoch 863/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0099
Epoch 864/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0099
Epoch 865/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0098
Epoch 866/1000
1/1 [==============================] - 0s 925us/step - loss: 0.0098
Epoch 867/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0097
Epoch 868/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0097
Epoch 869/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0096
Epoch 870/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0096
Epoch 871/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0095
Epoch 872/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0095
Epoch 873/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0094
Epoch 874/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0094
```

```
Epoch 875/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0094
Epoch 876/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0093
Epoch 877/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0093
Epoch 878/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0092
Epoch 879/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0092
Epoch 880/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0091
Epoch 881/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0091
Epoch 882/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0090
Epoch 883/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0090
Epoch 884/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0090
Epoch 885/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0089
Epoch 886/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0089
Epoch 887/1000
1/1 [==============================] - 0s 27ms/step - loss: 0.0088
Epoch 888/1000
1/1 [==============================] - 0s 5ms/step - loss: 0.0088
Epoch 889/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0087
Epoch 890/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0087
Epoch 891/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0087
Epoch 892/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0086
Epoch 893/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0086
Epoch 894/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0085
Epoch 895/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0085
Epoch 896/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0085
Epoch 897/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0084
Epoch 898/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0084
```

```
Epoch 899/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0083
Epoch 900/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0083
Epoch 901/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0083
Epoch 902/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0082
Epoch 903/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0082
Epoch 904/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0081
Epoch 905/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0081
Epoch 906/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0081
Epoch 907/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0080
Epoch 908/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0080
Epoch 909/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0079
Epoch 910/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0079
Epoch 911/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0079
Epoch 912/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0078
Epoch 913/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0078
Epoch 914/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0078
Epoch 915/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0077
Epoch 916/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0077
Epoch 917/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0076
Epoch 918/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0076
Epoch 919/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0076
Epoch 920/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0075
Epoch 921/1000
1/1 [==============================] - 0s 8ms/step - loss: 0.0075
Epoch 922/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0075
```

```
Epoch 923/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0074
Epoch 924/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0074
Epoch 925/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0074
Epoch 926/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0073
Epoch 927/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0073
Epoch 928/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0073
Epoch 929/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0072
Epoch 930/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0072
Epoch 931/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0072
Epoch 932/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0071
Epoch 933/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0071
Epoch 934/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0071
Epoch 935/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0070
Epoch 936/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0070
Epoch 937/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0070
Epoch 938/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0069
Epoch 939/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0069
Epoch 940/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0069
Epoch 941/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0068
Epoch 942/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0068
Epoch 943/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0068
Epoch 944/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0067
Epoch 945/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0067
Epoch 946/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0067
```

```
Epoch 947/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0067
Epoch 948/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0066
Epoch 949/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0066
Epoch 950/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0066
Epoch 951/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0065
Epoch 952/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0065
Epoch 953/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0065
Epoch 954/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0064
Epoch 955/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0064
Epoch 956/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0064
Epoch 957/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0064
Epoch 958/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0063
Epoch 959/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0063
Epoch 960/1000
1/1 [==============================] - 0s 989us/step - loss: 0.0063
Epoch 961/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0062
Epoch 962/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0062
Epoch 963/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0062
Epoch 964/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0062
Epoch 965/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0061
Epoch 966/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0061
Epoch 967/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0061
Epoch 968/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0061
Epoch 969/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0060
Epoch 970/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0060
```

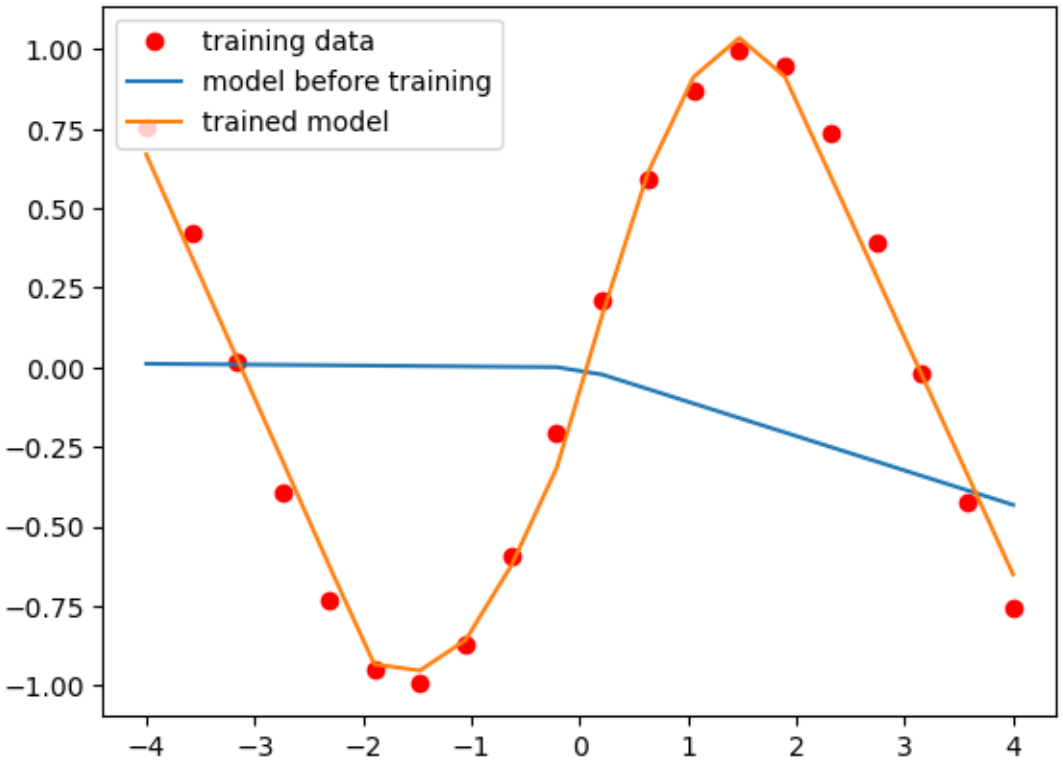
```
Epoch 971/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0060
Epoch 972/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0059
Epoch 973/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0059
Epoch 974/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0059
Epoch 975/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0059
Epoch 976/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0058
Epoch 977/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0058
Epoch 978/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0058
Epoch 979/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0058
Epoch 980/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0057
Epoch 981/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0057
Epoch 982/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0057
Epoch 983/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0057
Epoch 984/1000
1/1 [==============================] - 0s 2ms/step - loss: 0.0056
Epoch 985/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0056
Epoch 986/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0056
Epoch 987/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0056
Epoch 988/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0056
Epoch 989/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0055
Epoch 990/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0055
Epoch 991/1000
1/1 [==============================] - 0s 16ms/step - loss: 0.0055
Epoch 992/1000
1/1 [==============================] - 0s 10ms/step - loss: 0.0055
Epoch 993/1000
1/1 [==============================] - 0s 3ms/step - loss: 0.0054
Epoch 994/1000
1/1 [==============================] - 0s 4ms/step - loss: 0.0054
```

```
Epoch 995/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0054
Epoch 996/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0054
Epoch 997/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0053
Epoch 998/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0053
Epoch 999/1000
1/1 [==============================] - 0s 968us/step - loss: 0.0053
Epoch 1000/1000
1/1 [==============================] - 0s 1ms/step - loss: 0.0053
```

[28]:
```
<keras.src.callbacks.History at 0x2838a5eb0>
```

[29]:
```
# Let's take a look at how the model looks like after training

y_pred = model(x)
plt.figure()
plt.plot(x,y,'ro',label = "training data")
plt.plot(x,y_pred_before_training,label = 'model before training')

plt.plot(x,y_pred,label = 'trained model')
plt.legend()
```

[29]:
```
<matplotlib.legend.Legend at 0x28390af40>
```



3 3 NSL-KDD Example

3.1 3.1. Data Ingestion and Preprocess with Spark

We are going to use the NSLKDD dataset, so we start with rerunning our old code to load and preprocess the data.

[30]:
```
import pyspark
from pyspark.sql import SparkSession, SQLContext
from pyspark.ml import Pipeline,Transformer
from pyspark.ml.feature import␣
 ↪Imputer,StandardScaler,StringIndexer,OneHotEncoder, VectorAssembler

from pyspark.sql.functions import *
from pyspark.sql.types import *
import numpy as np

col_names = ["duration","protocol_type","service","flag","src_bytes",
"dst_bytes","land","wrong_fragment","urgent","hot","num_failed_logins",
"logged_in","num_compromised","root_shell","su_attempted","num_root",
"num_file_creations","num_shells","num_access_files","num_outbound_cmds",
"is_host_login","is_guest_login","count","srv_count","serror_rate",
```

```
"srv_serror_rate","rerror_rate","srv_rerror_rate","same_srv_rate",
"diff_srv_rate","srv_diff_host_rate","dst_host_count","dst_host_srv_count",
"dst_host_same_srv_rate","dst_host_diff_srv_rate","dst_host_same_src_port_rate",
"dst_host_srv_diff_host_rate","dst_host_serror_rate","dst_host_srv_serror_rate",
"dst_host_rerror_rate","dst_host_srv_rerror_rate","class","difficulty"]

nominal_cols = ['protocol_type','service','flag']
binary_cols = ['land', 'logged_in', 'root_shell', 'su_attempted',␣
 ↪'is_host_login',
'is_guest_login']
continuous_cols = ['duration' ,'src_bytes', 'dst_bytes', 'wrong_fragment'␣
 ↪,'urgent', 'hot',
'num_failed_logins', 'num_compromised', 'num_root' ,'num_file_creations',
'num_shells', 'num_access_files', 'num_outbound_cmds', 'count' ,'srv_count',
'serror_rate', 'srv_serror_rate' ,'rerror_rate' ,'srv_rerror_rate',
'same_srv_rate', 'diff_srv_rate', 'srv_diff_host_rate' ,'dst_host_count',
'dst_host_srv_count' ,'dst_host_same_srv_rate' ,'dst_host_diff_srv_rate',
'dst_host_same_src_port_rate' ,'dst_host_srv_diff_host_rate',
'dst_host_serror_rate' ,'dst_host_srv_serror_rate', 'dst_host_rerror_rate',
'dst_host_srv_rerror_rate']

class OutcomeCreater(Transformer): # this defines a transformer that creates␣
 ↪the outcome column

    def __init__(self):
        super().__init__()

    def _transform(self, dataset):
        label_to_binary = udf(lambda name: 0.0 if name == 'normal' else 1.0)
        output_df = dataset.withColumn('outcome',␣
 ↪label_to_binary(col('class'))).drop("class")
        output_df = output_df.withColumn('outcome', col('outcome').
 ↪cast(DoubleType()))
        output_df = output_df.drop('difficulty')
        return output_df

class FeatureTypeCaster(Transformer): # this transformer will cast the columns␣
 ↪as appropriate types
    def __init__(self):
        super().__init__()

    def _transform(self, dataset):
        output_df = dataset
        for col_name in binary_cols + continuous_cols:
            output_df = output_df.withColumn(col_name,col(col_name).
 ↪cast(DoubleType()))
```

```
        return output_df
class ColumnDropper(Transformer): # this transformer drops unnecessary columns
    def __init__(self, columns_to_drop = None):
        super().__init__()
        self.columns_to_drop=columns_to_drop
    def _transform(self, dataset):
        output_df = dataset
        for col_name in self.columns_to_drop:
            output_df = output_df.drop(col_name)
        return output_df

def get_preprocess_pipeline():
    # Stage where columns are casted as appropriate types
    stage_typecaster = FeatureTypeCaster()

    # Stage where nominal columns are transformed to index columns using␣
 ↪StringIndexer
    nominal_id_cols = [x+"_index" for x in nominal_cols]
    nominal_onehot_cols = [x+"_encoded" for x in nominal_cols]
    stage_nominal_indexer = StringIndexer(inputCols = nominal_cols, outputCols␣
 ↪= nominal_id_cols )

    # Stage where the index columns are further transformed using OneHotEncoder
    stage_nominal_onehot_encoder = OneHotEncoder(inputCols=nominal_id_cols,␣
 ↪outputCols=nominal_onehot_cols)

    # Stage where all relevant features are assembled into a vector (and␣
 ↪dropping a few)
    feature_cols = continuous_cols+binary_cols+nominal_onehot_cols
    corelated_cols_to_remove =␣
 ↪["dst_host_serror_rate","srv_serror_rate","dst_host_srv_serror_rate",
                    ␣
 ↪"srv_rerror_rate","dst_host_rerror_rate","dst_host_srv_rerror_rate"]
    for col_name in corelated_cols_to_remove:
        feature_cols.remove(col_name)
    stage_vector_assembler = VectorAssembler(inputCols=feature_cols,␣
 ↪outputCol="vectorized_features")

    # Stage where we scale the columns
    stage_scaler = StandardScaler(inputCol= 'vectorized_features', outputCol=␣
 ↪'features')


    # Stage for creating the outcome column representing whether there is␣
 ↪attack
```

```
    stage_outcome = OutcomeCreater()

    # Removing all unnecessary columbs, only keeping the 'features' and␣
 ↪'outcome' columns
    stage_column_dropper = ColumnDropper(columns_to_drop =␣
 ↪nominal_cols+nominal_id_cols+
        nominal_onehot_cols+ binary_cols + continuous_cols +␣
 ↪['vectorized_features'])
    # Connect the columns into a pipeline
    pipeline =␣
 ↪Pipeline(stages=[stage_typecaster,stage_nominal_indexer,stage_nominal_onehot_encoder,
        stage_vector_assembler,stage_scaler,stage_outcome,stage_column_dropper])
    return pipeline
```

[31]:
```
# if you installed Spark on windows,
# you may need findspark and need to initialize it prior to being able to use␣
 ↪pyspark
# Also, you may need to initialize SparkContext yourself.
# Uncomment the following lines if you are using Windows!
#import findspark
#findspark.init()
#findspark.find()

spark = SparkSession.builder \
    .master("local[*]") \
    .appName("GenericAppName") \
    .getOrCreate()

nslkdd_raw = spark.read.csv('./NSL-KDD/KDDTrain+.txt',header=False).
 ↪toDF(*col_names)
nslkdd_test_raw = spark.read.csv('./NSL-KDD/KDDTest+.txt',header=False).
 ↪toDF(*col_names)

preprocess_pipeline = get_preprocess_pipeline()
preprocess_pipeline_model = preprocess_pipeline.fit(nslkdd_raw)

nslkdd_df = preprocess_pipeline_model.transform(nslkdd_raw)
nslkdd_df_test = preprocess_pipeline_model.transform(nslkdd_test_raw)
```

```
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use
setLogLevel(newLevel).
24/09/15 19:52:52 WARN NativeCodeLoader: Unable to load native-hadoop library
for your platform… using builtin-java classes where applicable
24/09/15 19:53:05 WARN SparkStringUtils: Truncated the string representation of
a plan since it was too large. This behavior can be adjusted by setting
'spark.sql.debug.maxToStringFields'.
```

In addition to the train and test dataset, let's also create a validation set.

Normally, the validation set should be a subset of the training set, but our training set is not challenging enough which makes the later tuning part not interesting. Therefore, for today's purpose, we are going to use 50% of the test set as the validation set, and the remaining 50% is left as the test set.

After splitting, we convert the three dataframes to pandas dataframes.

[32]:
```
to_array = udf(lambda v: v.toArray().tolist(), ArrayType(FloatType()))

nslkdd_df_train = nslkdd_df
nslkdd_df_validate,nslkdd_df_test = nslkdd_df_test.randomSplit([0.5,0.5])

nslkdd_df_train_pandas = nslkdd_df_train.withColumn('features',␣
 ↪to_array('features')).toPandas()
nslkdd_df_validate_pandas = nslkdd_df_validate.withColumn('features',␣
 ↪to_array('features')).toPandas()
nslkdd_df_test_pandas = nslkdd_df_test.withColumn('features',␣
 ↪to_array('features')).toPandas()
```

We further convert the pandas dataframes to nparrays, and then tensors. We will use these tensors for today's lecture.

[33]:
```
import tensorflow as tf
from tensorflow import keras

# Converting the pandas DataFrame to tensors
# Note we are using 3 data sets train, validate, test

x_train = tf.constant(np.array(nslkdd_df_train_pandas['features'].values.
 ↪tolist()))
y_train = tf.constant(np.array(nslkdd_df_train_pandas['outcome'].values.
 ↪tolist()))

x_validate = tf.constant(np.array(nslkdd_df_validate_pandas['features'].values.
 ↪tolist()))
y_validate = tf.constant(np.array(nslkdd_df_validate_pandas['outcome'].values.
 ↪tolist()))


x_test = tf.constant(np.array(nslkdd_df_test_pandas['features'].values.
 ↪tolist()))
y_test = tf.constant(np.array(nslkdd_df_test_pandas['outcome'].values.tolist()))
```

3.2 3.2 Building Neural Networks for NSL-KDD with tf.keras

Let's create a simple neural network.

[34]:
```
model = keras.Sequential( [keras.layers.Dense(10,activation='relu'),
                           keras.layers.Dense(10,activation='relu'),
                           keras.layers.Dense(10,activation='relu'),
                           keras.layers.Dense(10,activation='relu') ,
                           keras.layers.Dense(2)] )


y_pred = model(x_train)
model.summary()
```

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_13 (Dense)            (125973, 10)              1140

 dense_14 (Dense)            (125973, 10)              110

 dense_15 (Dense)            (125973, 10)              110

 dense_16 (Dense)            (125973, 10)              110

 dense_17 (Dense)            (125973, 2)               22

=================================================================
Total params: 1492 (5.83 KB)
Trainable params: 1492 (5.83 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Let's now compile the model. You need to use the `SparseCategoricalCrossentropy` as the loss.

[35]:
```
# Compile the model
model.compile(optimizer = 'sgd',
    loss=keras.losses.SparseCategoricalCrossentropy(from_logits=True))
```

Note that it is important to set `from_logits = True` in `keras.losses.SparseCategoricalCrossentropy`. This is because the output of the neural network is the score (in other words, "logits"), not the probability.

If you want to use `from_logits = False` (which is the default value), you need to convert the score to the probability via the softmax function. See experiment below.

[36]:
```
loss_func_from_logit_true = keras.losses.
↪SparseCategoricalCrossentropy(from_logits=True)
```

```
loss_func_from_logit_false = keras.losses.
 ↪SparseCategoricalCrossentropy(from_logits=False)

loss_1 = loss_func_from_logit_true(y_train[:5],y_pred[:5])                    ␣
 ↪            # This is correct as y_pred is score and from_logits = True in␣
 ↪loss_func_1
loss_2 = loss_func_from_logit_false(y_train[:5],y_pred[:5])                   ␣
 ↪            # This is wrong as y_pred is score but from_logits = False in␣
 ↪loss_func_2
loss_3 = loss_func_from_logit_false(y_train[:5],
                    tf.keras.activations.softmax(y_pred[:5]))  # This is␣
 ↪correct as now we have converted the score to probability using the softmax

print("loss_1 = ", loss_1 )
print("loss_2 = ", loss_2 )
print("loss_3 = ", loss_3)
```

```
loss_1 =  tf.Tensor(0.6546897, shape=(), dtype=float32)
loss_2 =  tf.Tensor(0.6992447, shape=(), dtype=float32)
loss_3 =  tf.Tensor(0.6546897, shape=(), dtype=float32)
```

Because of the equivalence between `loss_1` and `loss_3` above, another way to create the model is to add a softmax activation at the output layer, in which case you need to set `from_logits = False` when you compile.

[37]:
```
model2 = keras.Sequential( [keras.layers.Dense(10,activation='relu'),
                            keras.layers.Dense(10,activation='relu'),
                            keras.layers.Dense(10,activation='relu'),
                            keras.layers.Dense(10,activation='relu') ,
                            keras.layers.Dense(2,activation='softmax')] )

model2.compile(optimizer = 'sgd',loss=keras.losses.
 ↪SparseCategoricalCrossentropy(from_logits=False))
```

Let's now fit the model. In addition to specify the training data `x_train,y_train`, here we can also specify `validation_data` as our validation data set. We can also provide a list of metrics, and we are using Accuracy.

After every epoch during the fitting process, the model will be evaluated using the validation dataset, and the metrics will be computed.

One thing to note is here we set `verbose = 2` to reduce the printing of the `fit` function to only print per epoch information.

[38]:
```
model = keras.Sequential( [keras.layers.Dense(10,activation='relu'),
                           keras.layers.Dense(10,activation='relu'),
                           keras.layers.Dense(10,activation='relu'),
                           keras.layers.Dense(10,activation='relu') ,
                           keras.layers.Dense(2)] )
```

```
model.compile(optimizer = keras.optimizers.SGD(learning_rate=0.02),
    loss=keras.losses.SparseCategoricalCrossentropy(from_logits=True),
    metrics=[keras.metrics.SparseCategoricalAccuracy()])

model.fit(x_train,y_train, epochs = 5,batch_size = 64,␣
 ↪validation_data=(x_validate,y_validate),verbose = 2)
```

```
WARNING:absl:At this time, the v2.11+ optimizer `tf.keras.optimizers.SGD` runs
slowly on M1/M2 Macs, please use the legacy Keras optimizer instead, located at
`tf.keras.optimizers.legacy.SGD`.
WARNING:absl:There is a known slowdown when using v2.11+ Keras optimizers on
M1/M2 Macs. Falling back to the legacy Keras optimizer, i.e.,
`tf.keras.optimizers.legacy.SGD`.

Epoch 1/5
1969/1969 - 1s - loss: 0.1190 - sparse_categorical_accuracy: 0.9635 - val_loss:
0.8395 - val_sparse_categorical_accuracy: 0.7533 - 810ms/epoch - 411us/step
Epoch 2/5
1969/1969 - 1s - loss: 0.0563 - sparse_categorical_accuracy: 0.9799 - val_loss:
0.8437 - val_sparse_categorical_accuracy: 0.7711 - 534ms/epoch - 271us/step
Epoch 3/5
1969/1969 - 1s - loss: 0.0360 - sparse_categorical_accuracy: 0.9879 - val_loss:
0.9250 - val_sparse_categorical_accuracy: 0.8025 - 542ms/epoch - 275us/step
Epoch 4/5
1969/1969 - 1s - loss: 0.0299 - sparse_categorical_accuracy: 0.9904 - val_loss:
1.0321 - val_sparse_categorical_accuracy: 0.8032 - 549ms/epoch - 279us/step
Epoch 5/5
1969/1969 - 1s - loss: 0.0276 - sparse_categorical_accuracy: 0.9913 - val_loss:
1.2070 - val_sparse_categorical_accuracy: 0.7859 - 548ms/epoch - 278us/step
```

[38]:
```
<keras.src.callbacks.History at 0x2d55cb940>
```

Finally, let's evaluate on Test Data

[39]:
```
model.evaluate(x_test,y_test, verbose = 2)
```

```
350/350 - 0s - loss: 1.2451 - sparse_categorical_accuracy: 0.7797 - 89ms/epoch -
255us/step
```

[39]:
```
[1.245111346244812, 0.7797065377235413]
```

4 4. TensorBoard

4.1 4.1 Displaying the results in TensorBoard

TensorBoard is a very powerful visualization tool. One simple usage of TensorBoard is to visualize the fitting process.

To that end, we need to create a `tf.keras.callbacks.TensorBoard` callback object that points to a log directory - you can pick any directory you want. Then, when calling `Model.fit()`, you need to pass the callback object to `Model.fit()`.

[40]:
```
import datetime

model = keras.Sequential( [keras.layers.Dense(10,activation='relu'),
    keras.layers.Dense(10,activation='relu'),
    keras.layers.Dense(10,activation='relu'),
    keras.layers.Dense(10,activation='relu') ,
    keras.layers.Dense(2)] )
model.compile(loss=keras.losses.SparseCategoricalCrossentropy(from_logits=True),
        metrics=[keras.metrics.SparseCategoricalAccuracy(name='Accuracy')])
log_dir = "logs14763/myfirstlog/" + datetime.datetime.now().
 ↪strftime("%Y%m%d-%H%M%S")
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir,␣
 ↪histogram_freq=1)
model.fit(x=x_train, y=y_train,
          epochs=20, verbose = 2,
          validation_data=(x_validate, y_validate),
          callbacks=[tensorboard_callback])
```

```
Epoch 1/20
3937/3937 - 2s - loss: 0.0713 - Accuracy: 0.9781 - val_loss: 1.0103 -
val_Accuracy: 0.7814 - 2s/epoch - 391us/step
Epoch 2/20
3937/3937 - 2s - loss: 0.0403 - Accuracy: 0.9899 - val_loss: 1.3445 -
val_Accuracy: 0.7787 - 2s/epoch - 548us/step
Epoch 3/20
3937/3937 - 1s - loss: 0.0367 - Accuracy: 0.9913 - val_loss: 1.3546 -
val_Accuracy: 0.7855 - 1s/epoch - 329us/step
Epoch 4/20
3937/3937 - 1s - loss: 0.0365 - Accuracy: 0.9918 - val_loss: 1.5001 -
val_Accuracy: 0.7838 - 1s/epoch - 314us/step
Epoch 5/20
3937/3937 - 1s - loss: 0.0359 - Accuracy: 0.9922 - val_loss: 2.1015 -
val_Accuracy: 0.7803 - 1s/epoch - 321us/step
Epoch 6/20
3937/3937 - 2s - loss: 0.0422 - Accuracy: 0.9922 - val_loss: 2.2532 -
val_Accuracy: 0.7783 - 2s/epoch - 384us/step
Epoch 7/20
3937/3937 - 1s - loss: 0.0415 - Accuracy: 0.9924 - val_loss: 2.0472 -
```

```
val_Accuracy: 0.7765 - 1s/epoch - 290us/step
Epoch 8/20
3937/3937 - 1s - loss: 0.0431 - Accuracy: 0.9923 - val_loss: 1.8469 -
val_Accuracy: 0.7898 - 1s/epoch - 294us/step
Epoch 9/20
3937/3937 - 1s - loss: 0.0410 - Accuracy: 0.9920 - val_loss: 1.9142 -
val_Accuracy: 0.7978 - 1s/epoch - 287us/step
Epoch 10/20
3937/3937 - 1s - loss: 0.0400 - Accuracy: 0.9918 - val_loss: 2.6968 -
val_Accuracy: 0.7828 - 1s/epoch - 294us/step
Epoch 11/20
3937/3937 - 1s - loss: 0.0393 - Accuracy: 0.9916 - val_loss: 3.0248 -
val_Accuracy: 0.7851 - 1s/epoch - 288us/step
Epoch 12/20
3937/3937 - 1s - loss: 0.0364 - Accuracy: 0.9915 - val_loss: 4.0466 -
val_Accuracy: 0.7870 - 1s/epoch - 285us/step
Epoch 13/20
3937/3937 - 1s - loss: 0.0385 - Accuracy: 0.9919 - val_loss: 4.3652 -
val_Accuracy: 0.7941 - 1s/epoch - 287us/step
Epoch 14/20
3937/3937 - 1s - loss: 0.0399 - Accuracy: 0.9918 - val_loss: 3.9634 -
val_Accuracy: 0.8111 - 1s/epoch - 285us/step
Epoch 15/20
3937/3937 - 1s - loss: 0.0376 - Accuracy: 0.9923 - val_loss: 4.3366 -
val_Accuracy: 0.8041 - 1s/epoch - 291us/step
Epoch 16/20
3937/3937 - 2s - loss: 0.0411 - Accuracy: 0.9918 - val_loss: 3.9977 -
val_Accuracy: 0.8330 - 2s/epoch - 457us/step
Epoch 17/20
3937/3937 - 1s - loss: 0.0493 - Accuracy: 0.9918 - val_loss: 5.2738 -
val_Accuracy: 0.8264 - 1s/epoch - 292us/step
Epoch 18/20
3937/3937 - 1s - loss: 0.0484 - Accuracy: 0.9918 - val_loss: 5.3899 -
val_Accuracy: 0.8402 - 1s/epoch - 299us/step
Epoch 19/20
3937/3937 - 1s - loss: 0.0516 - Accuracy: 0.9916 - val_loss: 4.8307 -
val_Accuracy: 0.8226 - 1s/epoch - 288us/step
Epoch 20/20
3937/3937 - 1s - loss: 0.0559 - Accuracy: 0.9917 - val_loss: 4.2636 -
val_Accuracy: 0.8103 - 1s/epoch - 289us/step
```

```
[40]: <keras.src.callbacks.History at 0x283b4bbb0>
```

To launch tensorboard in notebook, you can run the code below.

You can also run `tensorboard --logdir logs14763/myfirstlog/` in the terminal, which will show a URL, and you can open it in the browser.

No matter which way you use, it is important to specify the correct log directory - it should be the

same as the directory when you created the callback.

[ ]:
```
%load_ext tensorboard
%tensorboard --logdir logs14763/myfirstlog/
```

4.2 4.2 Hyper-Parameter Tuning with Tensor Board

TensorBoard also provides a powerful tool for visualizing hyper parameter tuning.

The first step is to specify what hyper parameters to tune and set up the configuration.

[42]:
```
from tensorboard.plugins.hparams import api as hp

HP_WIDTH = hp.HParam('NN_width', hp.Discrete([20,30]))
HP_DEPTH = hp.HParam('NN_depth', hp.Discrete([4,6]))


with tf.summary.create_file_writer('logs14763/hparam_tuning').as_default():
  hp.hparams_config(
    hparams=[HP_WIDTH, HP_DEPTH],
    metrics=[hp.Metric('Accuracy')],
  )
```

Next, we write a training function, which takes the hyper parameters as the input and train a model using the hyper parameters. This function will return the metrics corresponding to the hyper parameters.

[43]:
```
def train_test_model(hparams,logdir):
  model = keras.Sequential()
  for _ in range(hparams[HP_DEPTH]):
    model.add(keras.layers.Dense(hparams[HP_WIDTH],activation='relu'))
  model.add(keras.layers.Dense(2))
  model.compile(
      optimizer=keras.optimizers.SGD(),
      loss = keras.losses.SparseCategoricalCrossentropy(from_logits=True),
      metrics=[keras.metrics.SparseCategoricalAccuracy(name="Accuracy_epochs")])

  history = model.fit(x_train, y_train, epochs=5, verbose = 2,
  callbacks=[tf.keras.callbacks.TensorBoard(log_dir=logdir, histogram_freq=1)],
  validation_data = (x_validate, y_validate))

  accuracy = np.max(history.history["val_Accuracy_epochs"])
  return accuracy
```

Finally, we go through all combinations of hyper parameters and run the `train_test_model` above. After each run, we will record the hyper parameters we used in this run, and the corresponding metrics.

[44]:
```
for hp_width in HP_WIDTH.domain.values:
  for hp_depth in (HP_DEPTH.domain.values):
    hparams = {
        HP_WIDTH: hp_width,
        HP_DEPTH: hp_depth,
    }
    run_name = f"run-WIDTH{int(hparams[HP_WIDTH])}-DEPTH{hparams[HP_DEPTH]}"
    print('--- Starting trial: %s' % run_name)
    print({h.name: hparams[h] for h in hparams})

    run_dir = 'logs14763/hparam_tuning/' + run_name
    accuracy = train_test_model(hparams,run_dir)

    with tf.summary.create_file_writer(run_dir).as_default():
      hp.hparams(hparams)  # record the values used in this trial
      tf.summary.scalar("Accuracy", accuracy, step=1)
```

```
WARNING:absl:At this time, the v2.11+ optimizer `tf.keras.optimizers.SGD` runs
slowly on M1/M2 Macs, please use the legacy Keras optimizer instead, located at
`tf.keras.optimizers.legacy.SGD`.
WARNING:absl:There is a known slowdown when using v2.11+ Keras optimizers on
M1/M2 Macs. Falling back to the legacy Keras optimizer, i.e.,
`tf.keras.optimizers.legacy.SGD`.

--- Starting trial: run-WIDTH20-DEPTH4
{'NN_width': 20, 'NN_depth': 4}
Epoch 1/5
3937/3937 - 2s - loss: 0.0992 - Accuracy_epochs: 0.9702 - val_loss: 1.3471 -
val_Accuracy_epochs: 0.7597 - 2s/epoch - 396us/step
Epoch 2/5
3937/3937 - 2s - loss: 0.0397 - Accuracy_epochs: 0.9866 - val_loss: 1.3635 -
val_Accuracy_epochs: 0.7792 - 2s/epoch - 416us/step
Epoch 3/5
3937/3937 - 1s - loss: 0.0274 - Accuracy_epochs: 0.9912 - val_loss: 1.6254 -
val_Accuracy_epochs: 0.7856 - 1s/epoch - 258us/step
Epoch 4/5
3937/3937 - 1s - loss: 0.0234 - Accuracy_epochs: 0.9922 - val_loss: 1.5332 -
val_Accuracy_epochs: 0.7856 - 1s/epoch - 263us/step
Epoch 5/5
3937/3937 - 1s - loss: 0.0216 - Accuracy_epochs: 0.9928 - val_loss: 1.5466 -
val_Accuracy_epochs: 0.7853 - 1s/epoch - 259us/step

WARNING:absl:At this time, the v2.11+ optimizer `tf.keras.optimizers.SGD` runs
slowly on M1/M2 Macs, please use the legacy Keras optimizer instead, located at
`tf.keras.optimizers.legacy.SGD`.
WARNING:absl:There is a known slowdown when using v2.11+ Keras optimizers on
```

```
M1/M2 Macs. Falling back to the legacy Keras optimizer, i.e.,
`tf.keras.optimizers.legacy.SGD`.

--- Starting trial: run-WIDTH20-DEPTH6
{'NN_width': 20, 'NN_depth': 6}
Epoch 1/5
3937/3937 - 2s - loss: 0.1021 - Accuracy_epochs: 0.9714 - val_loss: 0.7667 -
val_Accuracy_epochs: 0.8026 - 2s/epoch - 401us/step
Epoch 2/5
3937/3937 - 1s - loss: 0.0325 - Accuracy_epochs: 0.9892 - val_loss: 0.8954 -
val_Accuracy_epochs: 0.7991 - 1s/epoch - 341us/step
Epoch 3/5
3937/3937 - 1s - loss: 0.0267 - Accuracy_epochs: 0.9905 - val_loss: 1.1704 -
val_Accuracy_epochs: 0.7897 - 1s/epoch - 273us/step
Epoch 4/5
3937/3937 - 1s - loss: 0.0240 - Accuracy_epochs: 0.9915 - val_loss: 1.2842 -
val_Accuracy_epochs: 0.7858 - 1s/epoch - 271us/step
Epoch 5/5
3937/3937 - 1s - loss: 0.0219 - Accuracy_epochs: 0.9928 - val_loss: 1.2027 -
val_Accuracy_epochs: 0.7931 - 1s/epoch - 288us/step

WARNING:absl:At this time, the v2.11+ optimizer `tf.keras.optimizers.SGD` runs
slowly on M1/M2 Macs, please use the legacy Keras optimizer instead, located at
`tf.keras.optimizers.legacy.SGD`.
WARNING:absl:There is a known slowdown when using v2.11+ Keras optimizers on
M1/M2 Macs. Falling back to the legacy Keras optimizer, i.e.,
`tf.keras.optimizers.legacy.SGD`.

--- Starting trial: run-WIDTH30-DEPTH4
{'NN_width': 30, 'NN_depth': 4}
Epoch 1/5
3937/3937 - 2s - loss: 0.0989 - Accuracy_epochs: 0.9696 - val_loss: 1.2051 -
val_Accuracy_epochs: 0.7461 - 2s/epoch - 424us/step
Epoch 2/5
3937/3937 - 1s - loss: 0.0429 - Accuracy_epochs: 0.9843 - val_loss: 1.2038 -
val_Accuracy_epochs: 0.7727 - 1s/epoch - 275us/step
Epoch 3/5
3937/3937 - 1s - loss: 0.0301 - Accuracy_epochs: 0.9902 - val_loss: 1.4188 -
val_Accuracy_epochs: 0.7753 - 1s/epoch - 267us/step
Epoch 4/5
3937/3937 - 1s - loss: 0.0252 - Accuracy_epochs: 0.9914 - val_loss: 1.3904 -
val_Accuracy_epochs: 0.7827 - 1s/epoch - 271us/step
Epoch 5/5
3937/3937 - 1s - loss: 0.0229 - Accuracy_epochs: 0.9918 - val_loss: 1.3934 -
val_Accuracy_epochs: 0.7810 - 1s/epoch - 265us/step

WARNING:absl:At this time, the v2.11+ optimizer `tf.keras.optimizers.SGD` runs
slowly on M1/M2 Macs, please use the legacy Keras optimizer instead, located at
`tf.keras.optimizers.legacy.SGD`.
WARNING:absl:There is a known slowdown when using v2.11+ Keras optimizers on
```

```
M1/M2 Macs. Falling back to the legacy Keras optimizer, i.e.,
`tf.keras.optimizers.legacy.SGD`.

--- Starting trial: run-WIDTH30-DEPTH6
{'NN_width': 30, 'NN_depth': 6}
Epoch 1/5
3937/3937 - 1s - loss: 0.0932 - Accuracy_epochs: 0.9735 - val_loss: 1.2155 -
val_Accuracy_epochs: 0.7480 - 1s/epoch - 347us/step
Epoch 2/5
3937/3937 - 2s - loss: 0.0323 - Accuracy_epochs: 0.9894 - val_loss: 1.1343 -
val_Accuracy_epochs: 0.7845 - 2s/epoch - 408us/step
Epoch 3/5
3937/3937 - 1s - loss: 0.0250 - Accuracy_epochs: 0.9914 - val_loss: 1.1070 -
val_Accuracy_epochs: 0.7817 - 1s/epoch - 350us/step
Epoch 4/5
3937/3937 - 1s - loss: 0.0220 - Accuracy_epochs: 0.9923 - val_loss: 1.2443 -
val_Accuracy_epochs: 0.7799 - 1s/epoch - 290us/step
Epoch 5/5
3937/3937 - 1s - loss: 0.0202 - Accuracy_epochs: 0.9930 - val_loss: 1.1642 -
val_Accuracy_epochs: 0.7814 - 1s/epoch - 288us/step
```