Artificial Intelligence Nanodegree

Recurrent Neural Network Projects

Welcome to the Recurrent Neural Network Project in the Artificial Intelligence Nanodegree! In this notebook, some template code has already been provided for you, and you will need to implement additional functionality to successfully complete this project. You will not need to modify the included code beyond what is requested. Sections that begin with **'Implementation'** in the header indicate that the following block of code will require additional functionality which you must provide. Instructions will be provided for each section and the specifics of the implementation are marked in the code block with a 'TODO' statement. Please be sure to read the instructions carefully!

In addition to implementing code, there will be questions that you must answer which relate to the project and your implementation. Each section where you will answer a question is preceded by a **'Question X'** header. Carefully read each question and provide thorough answers in the following text boxes that begin with **'Answer:'**. Your project submission will be evaluated based on your answers to each of the questions and the implementation you provide.

> **Note:** Code and Markdown cells can be executed using the **Shift + Enter** keyboard shortcut. In addition, Markdown cells can be edited by typically double-clicking the cell to enter edit mode.

Implementation TODOs in this notebook

This notebook contains two problems, cut into a variety of TODOs. Make sure to complete each section containing a TODO marker throughout the notebook. For convinence we provide links to each of these sections below.

TODO #1: Implement a function to window time series

TODO #2: Create a simple RNN model using keras to perform regression

TODO #3: Finish cleaning a large text corpus

TODO #4: Implement a function to window a large text corpus

TODO #5: Create a simple RNN model using keras to perform multiclass classification

TODO #6: Generate text using a fully trained RNN model and a variety of input sequences

Problem 1: Perform time series prediction

In this project you will perform time series prediction using a Recurrent Neural Network regressor. In particular you will re-create the figure shown in the notes - where the stock price of Apple was forecasted (or predicted) 7 days in advance. In completing this exercise you will learn how to construct RNNs using Keras, which will also aid in completing the second project in this notebook.

The particular network architecture we will employ for our RNN is known as Long Term Short Memory (LTSM) (https://en.wikipedia.org/wiki/Long_short-term_memory), which helps significantly avoid technical problems with optimization of RNNs.

1.1 Getting started

First we must load in our time series - a history of around 140 days of Apple's stock price. Then we need to perform a number of pre-processing steps to prepare it for use with an RNN model. First off, it is good practice to normalize time series - by normalizing its range. This helps us avoid serious numerical issues associated how common activation functions (like tanh) transform very large (positive or negative) numbers, as well as helping us to avoid related issues when computing derivatives.

Here we normalize the series to lie in the range [0,1] using this scikit function (http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html), but it is also commonplace to normalize by a series standard deviation.

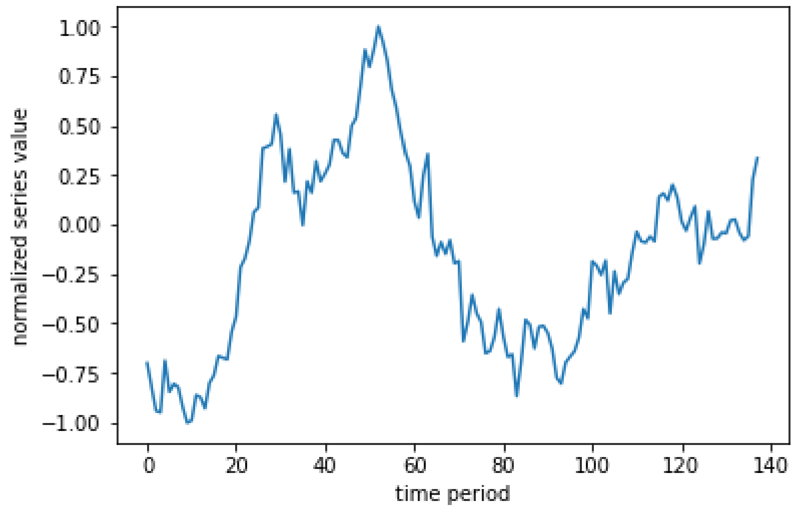
In [1]:
```
### Load in necessary libraries for data input and normalization
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt

### load in and normalize the dataset
dataset = np.loadtxt('datasets/normalized_apple_prices.csv')
```

Lets take a quick look at the (normalized) time series we'll be performing predictions on.

In [2]:
```
# lets take a look at our time series
plt.plot(dataset)
plt.xlabel('time period')
plt.ylabel('normalized series value')
```

Out[2]: <matplotlib.text.Text at 0x7fbc589cf8d0>

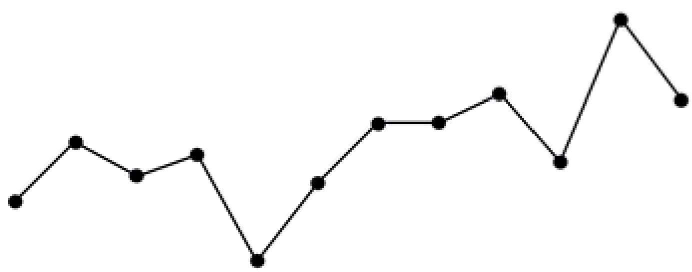


1.2 Cutting our time series into sequences

Remember, our time series is a sequence of numbers that we can represent in general mathematically as

$$s_0, s_1, s_2, \ldots, s_P$$

where s_p is the numerical value of the time series at time period p and where P is the total length of the series. In order to apply our RNN we treat the time series prediction problem as a regression problem, and so need to use a sliding window to construct a set of associated input/output pairs to regress on. This process is animated in the gif below.



For example - using a window of size T = 5 (as illustrated in the gif above) we produce a set of input/output pairs like the one shown in the table below

Input	Output
$\langle s_1, s_2, s_3, s_4, s_5 \rangle$	s_6
$\langle s_2, s_3, s_4, s_5, s_6 \rangle$	s_7
$\vdots$	$\vdots$
$\langle s_{P-5}, s_{P-4}, s_{P-3}, s_{P-2}, s_{P-1} \rangle$	s_P

Notice here that each input is a sequence (or vector) of length 4 (and in general has length equal to the window size T) while each corresponding output is a scalar value. Notice also how given a time series of length P and window size T = 5 as shown above, we created P - 5 input/output pairs. More generally, for a window size T we create P - T such pairs.

Now its time for you to window the input time series as described above!

TODO: Fill in the function below - called **window_transform_series** - that runs a sliding window along the input series and creates associated input/output pairs. Note that this function should input a) the series and b) the window length, and return the input/output subsequences. Make sure to format returned input/output as generally shown in table above (where window_size = 5), and make sure your returned input is a numpy array.

In [3]:
```
### Fill out the function below that transforms the input series and win
dow-size into a set of input/output pairs for use with our RNN model
def window_transform_series(series,window_size):
    # containers for input/output pairs
    X = []
    y = []

    # Slice the series into windows
    for i in range(window_size, len(series)):
        X.append(series[i-window_size:i])
        y.append(series[i])

    # reshape each
    X = np.asarray(X)
    X.shape = (np.shape(X)[0:2])
    y = np.asarray(y)
    y.shape = (len(y),1)

    return X,y
```

You can test your function on the list of odd numbers given below

In [4]:
```
odd_nums = np.array([1,3,5,7,9,11,13])
```

To window this sequence with a window_size = 2 using the **window_transform_series** you should get the following input/output pairs

In [5]:
```
# run a window of size 2 over the odd number sequence and display the re
sults
window_size = 2
X,y = window_transform_series(odd_nums,window_size)

# print out input/output pairs --> here input = X, corresponding output
 = y
print ('--- the input X will look like ----')
print (X)

print ('--- the associated output y will look like ----')
print (y)

print ('the shape of X is ' + str(np.shape(X)))
print ('the shape of y is ' + str(np.shape(y)))
print('the type of X is ' + str(type(X)))
print('the type of y is ' + str(type(y)))
```

```
--- the input X will look like ----
[[ 1  3]
 [ 3  5]
 [ 5  7]
 [ 7  9]
 [ 9 11]]
--- the associated output y will look like ----
[[ 5]
 [ 7]
 [ 9]
 [11]
 [13]]
the shape of X is (5, 2)
the shape of y is (5, 1)
the type of X is <class 'numpy.ndarray'>
the type of y is <class 'numpy.ndarray'>
```

Again - you can check that your completed **window_transform_series** function works correctly by trying it on the odd_nums sequence - you should get the above output.

(remember to copy your completed function into the script *my_answers.py* function titled *window_transform_series* before submitting your project)

With this function in place apply it to the series in the Python cell below. We use a window_size = 7 for these experiments.

In [6]:
```
# window the data using your windowing function
window_size = 7
X,y = window_transform_series(series = dataset,window_size =
window_size)
```

1.3 Splitting into training and testing sets

In order to perform proper testing on our dataset we will lop off the last 1/3 of it for validation (or testing). This is that once we train our model we have something to test it on (like any regression problem!). This splitting into training/testing sets is done in the cell below.

Note how here we are **not** splitting the dataset *randomly* as one typically would do when validating a regression model. This is because our input/output pairs *are related temporally*. We don't want to validate our model by training on a random subset of the series and then testing on another random subset, as this simulates the scenario that we receive new points *within the timeframe of our training set*.

We want to train on one solid chunk of the series (in our case, the first full 2/3 of it), and validate on a later chunk (the last 1/3) as this simulates how we would predict *future* values of a time series.

In [7]:

```
# split our dataset into training / testing sets
train_test_split = int(np.ceil(2*len(y)/float(3)))   # set the split poi
nt

# partition the training set
X_train = X[:train_test_split,:]
y_train = y[:train_test_split]

# keep the last chunk for testing
X_test = X[train_test_split:,:]
y_test = y[train_test_split:]

# NOTE: to use keras's RNN LSTM module our input must be reshaped to [sa
mples, window size, stepsize]
X_train = np.asarray(np.reshape(X_train, (X_train.shape[0], window_size,
 1)))
X_test = np.asarray(np.reshape(X_test, (X_test.shape[0], window_size,
1)))
```

1.4 Build and run an RNN regression model

Having created input/output pairs out of our time series and cut this into training/testing sets, we can now begin setting up our RNN. We use Keras to quickly build a two hidden layer RNN of the following specifications

- layer 1 uses an LSTM module with 5 hidden units (note here the input_shape = (window_size,1))
- layer 2 uses a fully connected module with one unit
- the 'mean_squared_error' loss should be used (remember: we are performing regression here)

This can be constructed using just a few lines - see e.g., the general Keras documentation (https://keras.io/getting-started/sequential-model-guide/) and the LTSM documentation in particular (https://keras.io/layers/recurrent/) for examples of how to quickly use Keras to build neural network models. Make sure you are initializing your optimizer given the keras-recommended approach for RNNs (https://keras.io/optimizers/)

(given in the cell below). (remember to copy your completed function into the script *my_answers.py* function titled *build_part1_RNN* before submitting your project)

In [8]:
```
### create required RNN model
# import keras network libraries
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
import keras

# given - fix random seed - so we can all reproduce the same results on
 our default time series
np.random.seed(0)


# build an RNN to perform regression on our time series input/output dat
a
model = Sequential()
model.add(LSTM(5, input_shape=(window_size,1)))
model.add(Dense(1))


# build model using keras documentation recommended optimizer initializa
tion
optimizer = keras.optimizers.RMSprop(lr=0.001, rho=0.9, epsilon=1e-08, d
ecay=0.0)

# compile the model
model.compile(loss='mean_squared_error', optimizer=optimizer)
```

```
Using TensorFlow backend.
```

With your model built you can now fit the model by activating the cell below! Note: the number of epochs (np_epochs) and batch_size are preset (so we can all produce the same results). You can choose to toggle the verbose parameter - which gives you regular updates on the progress of the algorithm - on and off by setting it to 1 or 0 respectively.

In [9]:
```
# run your model!
model.fit(X_train, y_train, epochs=1000, batch_size=50, verbose=1)
```

```
Epoch 1/1000
88/88 [==============================] - 0s - loss: 0.1422
Epoch 2/1000
88/88 [==============================] - 0s - loss: 0.1343
Epoch 3/1000
88/88 [==============================] - 0s - loss: 0.1290
Epoch 4/1000
88/88 [==============================] - 0s - loss: 0.1245
Epoch 5/1000
88/88 [==============================] - 0s - loss: 0.1206
Epoch 6/1000
88/88 [==============================] - 0s - loss: 0.1170
Epoch 7/1000
88/88 [==============================] - 0s - loss: 0.1139
Epoch 8/1000
88/88 [==============================] - 0s - loss: 0.1106
Epoch 9/1000
88/88 [==============================] - 0s - loss: 0.1079
Epoch 10/1000
88/88 [==============================] - 0s - loss: 0.1051
Epoch 11/1000
88/88 [==============================] - 0s - loss: 0.1021
Epoch 12/1000
88/88 [==============================] - 0s - loss: 0.0994
Epoch 13/1000
88/88 [==============================] - 0s - loss: 0.0968
Epoch 14/1000
88/88 [==============================] - 0s - loss: 0.0944
Epoch 15/1000
88/88 [==============================] - 0s - loss: 0.0917
Epoch 16/1000
88/88 [==============================] - 0s - loss: 0.0893
Epoch 17/1000
88/88 [==============================] - 0s - loss: 0.0869
Epoch 18/1000
88/88 [==============================] - 0s - loss: 0.0846
Epoch 19/1000
88/88 [==============================] - 0s - loss: 0.0823
Epoch 20/1000
88/88 [==============================] - 0s - loss: 0.0803
Epoch 21/1000
88/88 [==============================] - 0s - loss: 0.0781
Epoch 22/1000
88/88 [==============================] - 0s - loss: 0.0763
Epoch 23/1000
88/88 [==============================] - 0s - loss: 0.0740
Epoch 24/1000
88/88 [==============================] - 0s - loss: 0.0721
Epoch 25/1000
88/88 [==============================] - 0s - loss: 0.0702
Epoch 26/1000
88/88 [==============================] - 0s - loss: 0.0685
Epoch 27/1000
88/88 [==============================] - 0s - loss: 0.0667
Epoch 28/1000
88/88 [==============================] - ETA: 0s - loss: 0.054 - 0s - l
oss: 0.0651
```

```
Epoch 29/1000
88/88 [==============================] - 0s - loss: 0.0635
Epoch 30/1000
88/88 [==============================] - 0s - loss: 0.0621
Epoch 31/1000
88/88 [==============================] - 0s - loss: 0.0608
Epoch 32/1000
88/88 [==============================] - 0s - loss: 0.0592
Epoch 33/1000
88/88 [==============================] - 0s - loss: 0.0580
Epoch 34/1000
88/88 [==============================] - 0s - loss: 0.0568
Epoch 35/1000
88/88 [==============================] - 0s - loss: 0.0557
Epoch 36/1000
88/88 [==============================] - 0s - loss: 0.0546
Epoch 37/1000
88/88 [==============================] - 0s - loss: 0.0536
Epoch 38/1000
88/88 [==============================] - 0s - loss: 0.0528
Epoch 39/1000
88/88 [==============================] - 0s - loss: 0.0519
Epoch 40/1000
88/88 [==============================] - 0s - loss: 0.0511
Epoch 41/1000
88/88 [==============================] - 0s - loss: 0.0504
Epoch 42/1000
88/88 [==============================] - 0s - loss: 0.0498
Epoch 43/1000
88/88 [==============================] - 0s - loss: 0.0491
Epoch 44/1000
88/88 [==============================] - 0s - loss: 0.0486
Epoch 45/1000
88/88 [==============================] - 0s - loss: 0.0481
Epoch 46/1000
88/88 [==============================] - 0s - loss: 0.0477
Epoch 47/1000
88/88 [==============================] - 0s - loss: 0.0473
Epoch 48/1000
88/88 [==============================] - 0s - loss: 0.0469
Epoch 49/1000
88/88 [==============================] - 0s - loss: 0.0466
Epoch 50/1000
88/88 [==============================] - 0s - loss: 0.0465
Epoch 51/1000
88/88 [==============================] - 0s - loss: 0.0462
Epoch 52/1000
88/88 [==============================] - 0s - loss: 0.0457
Epoch 53/1000
88/88 [==============================] - 0s - loss: 0.0456
Epoch 54/1000
88/88 [==============================] - 0s - loss: 0.0452
Epoch 55/1000
88/88 [==============================] - 0s - loss: 0.0451
Epoch 56/1000
88/88 [==============================] - 0s - loss: 0.0448
Epoch 57/1000
```

```
88/88 [==============================] - 0s - loss: 0.0447
Epoch 58/1000
88/88 [==============================] - 0s - loss: 0.0443
Epoch 59/1000
88/88 [==============================] - 0s - loss: 0.0443
Epoch 60/1000
88/88 [==============================] - 0s - loss: 0.0441
Epoch 61/1000
88/88 [==============================] - 0s - loss: 0.0438
Epoch 62/1000
88/88 [==============================] - 0s - loss: 0.0436
Epoch 63/1000
88/88 [==============================] - 0s - loss: 0.0434
Epoch 64/1000
88/88 [==============================] - 0s - loss: 0.0432
Epoch 65/1000
88/88 [==============================] - 0s - loss: 0.0429
Epoch 66/1000
88/88 [==============================] - 0s - loss: 0.0427
Epoch 67/1000
88/88 [==============================] - 0s - loss: 0.0426
Epoch 68/1000
88/88 [==============================] - 0s - loss: 0.0423
Epoch 69/1000
88/88 [==============================] - 0s - loss: 0.0421
Epoch 70/1000
88/88 [==============================] - 0s - loss: 0.0419
Epoch 71/1000
88/88 [==============================] - 0s - loss: 0.0417
Epoch 72/1000
88/88 [==============================] - 0s - loss: 0.0414
Epoch 73/1000
88/88 [==============================] - ETA: 0s - loss: 0.039 - 0s - l
oss: 0.0413
Epoch 74/1000
88/88 [==============================] - ETA: 0s - loss: 0.038 - 0s - l
oss: 0.0411
Epoch 75/1000
88/88 [==============================] - 0s - loss: 0.0408
Epoch 76/1000
88/88 [==============================] - 0s - loss: 0.0406
Epoch 77/1000
88/88 [==============================] - 0s - loss: 0.0404
Epoch 78/1000
88/88 [==============================] - 0s - loss: 0.0403
Epoch 79/1000
88/88 [==============================] - 0s - loss: 0.0400
Epoch 80/1000
88/88 [==============================] - 0s - loss: 0.0399
Epoch 81/1000
88/88 [==============================] - 0s - loss: 0.0396
Epoch 82/1000
88/88 [==============================] - 0s - loss: 0.0393
Epoch 83/1000
88/88 [==============================] - 0s - loss: 0.0392
Epoch 84/1000
88/88 [==============================] - 0s - loss: 0.0391
```

```
Epoch 85/1000
88/88 [==============================] - 0s - loss: 0.0389
Epoch 86/1000
88/88 [==============================] - 0s - loss: 0.0387
Epoch 87/1000
88/88 [==============================] - 0s - loss: 0.0384
Epoch 88/1000
88/88 [==============================] - 0s - loss: 0.0383
Epoch 89/1000
88/88 [==============================] - 0s - loss: 0.0380
Epoch 90/1000
88/88 [==============================] - 0s - loss: 0.0382
Epoch 91/1000
88/88 [==============================] - 0s - loss: 0.0376
Epoch 92/1000
88/88 [==============================] - ETA: 0s - loss: 0.027 - 0s - l
oss: 0.0376
Epoch 93/1000
88/88 [==============================] - 0s - loss: 0.0375
Epoch 94/1000
88/88 [==============================] - 0s - loss: 0.0372
Epoch 95/1000
88/88 [==============================] - 0s - loss: 0.0374
Epoch 96/1000
88/88 [==============================] - 0s - loss: 0.0370
Epoch 97/1000
88/88 [==============================] - 0s - loss: 0.0370
Epoch 98/1000
88/88 [==============================] - 0s - loss: 0.0367
Epoch 99/1000
88/88 [==============================] - 0s - loss: 0.0367
Epoch 100/1000
88/88 [==============================] - 0s - loss: 0.0364
Epoch 101/1000
88/88 [==============================] - 0s - loss: 0.0363
Epoch 102/1000
88/88 [==============================] - 0s - loss: 0.0361
Epoch 103/1000
88/88 [==============================] - 0s - loss: 0.0360
Epoch 104/1000
88/88 [==============================] - 0s - loss: 0.0358
Epoch 105/1000
88/88 [==============================] - 0s - loss: 0.0357
Epoch 106/1000
88/88 [==============================] - 0s - loss: 0.0355
Epoch 107/1000
88/88 [==============================] - 0s - loss: 0.0354
Epoch 108/1000
88/88 [==============================] - 0s - loss: 0.0352
Epoch 109/1000
88/88 [==============================] - 0s - loss: 0.0350
Epoch 110/1000
88/88 [==============================] - 0s - loss: 0.0350
Epoch 111/1000
88/88 [==============================] - 0s - loss: 0.0346
Epoch 112/1000
88/88 [==============================] - 0s - loss: 0.0345
```

```
Epoch 113/1000
88/88 [==============================] - 0s - loss: 0.0343
Epoch 114/1000
88/88 [==============================] - 0s - loss: 0.0341
Epoch 115/1000
88/88 [==============================] - 0s - loss: 0.0342
Epoch 116/1000
88/88 [==============================] - 0s - loss: 0.0339
Epoch 117/1000
88/88 [==============================] - 0s - loss: 0.0338
Epoch 118/1000
88/88 [==============================] - 0s - loss: 0.0336
Epoch 119/1000
88/88 [==============================] - 0s - loss: 0.0336
Epoch 120/1000
88/88 [==============================] - 0s - loss: 0.0332
Epoch 121/1000
88/88 [==============================] - 0s - loss: 0.0332
Epoch 122/1000
88/88 [==============================] - 0s - loss: 0.0329
Epoch 123/1000
88/88 [==============================] - 0s - loss: 0.0328
Epoch 124/1000
88/88 [==============================] - 0s - loss: 0.0326
Epoch 125/1000
88/88 [==============================] - 0s - loss: 0.0325
Epoch 126/1000
88/88 [==============================] - 0s - loss: 0.0323
Epoch 127/1000
88/88 [==============================] - 0s - loss: 0.0322
Epoch 128/1000
88/88 [==============================] - 0s - loss: 0.0320
Epoch 129/1000
88/88 [==============================] - 0s - loss: 0.0320
Epoch 130/1000
88/88 [==============================] - 0s - loss: 0.0318
Epoch 131/1000
88/88 [==============================] - 0s - loss: 0.0315
Epoch 132/1000
88/88 [==============================] - 0s - loss: 0.0313
Epoch 133/1000
88/88 [==============================] - 0s - loss: 0.0314
Epoch 134/1000
88/88 [==============================] - 0s - loss: 0.0315
Epoch 135/1000
88/88 [==============================] - 0s - loss: 0.0309
Epoch 136/1000
88/88 [==============================] - 0s - loss: 0.0308
Epoch 137/1000
88/88 [==============================] - 0s - loss: 0.0308
Epoch 138/1000
88/88 [==============================] - 0s - loss: 0.0306
Epoch 139/1000
88/88 [==============================] - 0s - loss: 0.0304
Epoch 140/1000
88/88 [==============================] - 0s - loss: 0.0303
Epoch 141/1000
```

```
88/88 [==============================] - 0s - loss: 0.0301
Epoch 142/1000
88/88 [==============================] - 0s - loss: 0.0301
Epoch 143/1000
88/88 [==============================] - ETA: 0s - loss: 0.031 - 0s - l
oss: 0.0299
Epoch 144/1000
88/88 [==============================] - 0s - loss: 0.0297
Epoch 145/1000
88/88 [==============================] - 0s - loss: 0.0295
Epoch 146/1000
88/88 [==============================] - 0s - loss: 0.0294
Epoch 147/1000
88/88 [==============================] - 0s - loss: 0.0295
Epoch 148/1000
88/88 [==============================] - 0s - loss: 0.0291
Epoch 149/1000
88/88 [==============================] - 0s - loss: 0.0290
Epoch 150/1000
88/88 [==============================] - 0s - loss: 0.0289
Epoch 151/1000
88/88 [==============================] - 0s - loss: 0.0287
Epoch 152/1000
88/88 [==============================] - 0s - loss: 0.0286
Epoch 153/1000
88/88 [==============================] - 0s - loss: 0.0287
Epoch 154/1000
88/88 [==============================] - 0s - loss: 0.0284
Epoch 155/1000
88/88 [==============================] - 0s - loss: 0.0282
Epoch 156/1000
88/88 [==============================] - 0s - loss: 0.0280
Epoch 157/1000
88/88 [==============================] - 0s - loss: 0.0280
Epoch 158/1000
88/88 [==============================] - 0s - loss: 0.0278
Epoch 159/1000
88/88 [==============================] - 0s - loss: 0.0276
Epoch 160/1000
88/88 [==============================] - 0s - loss: 0.0277
Epoch 161/1000
88/88 [==============================] - 0s - loss: 0.0275
Epoch 162/1000
88/88 [==============================] - 0s - loss: 0.0275
Epoch 163/1000
88/88 [==============================] - 0s - loss: 0.0272
Epoch 164/1000
88/88 [==============================] - 0s - loss: 0.0270
Epoch 165/1000
88/88 [==============================] - 0s - loss: 0.0270
Epoch 166/1000
88/88 [==============================] - 0s - loss: 0.0268
Epoch 167/1000
88/88 [==============================] - 0s - loss: 0.0268
Epoch 168/1000
88/88 [==============================] - 0s - loss: 0.0266
Epoch 169/1000
```

```
88/88 [==============================] - 0s - loss: 0.0265
Epoch 170/1000
88/88 [==============================] - 0s - loss: 0.0265
Epoch 171/1000
88/88 [==============================] - 0s - loss: 0.0263
Epoch 172/1000
88/88 [==============================] - 0s - loss: 0.0264
Epoch 173/1000
88/88 [==============================] - 0s - loss: 0.0262
Epoch 174/1000
88/88 [==============================] - 0s - loss: 0.0259
Epoch 175/1000
88/88 [==============================] - 0s - loss: 0.0259
Epoch 176/1000
88/88 [==============================] - 0s - loss: 0.0258
Epoch 177/1000
88/88 [==============================] - 0s - loss: 0.0256
Epoch 178/1000
88/88 [==============================] - 0s - loss: 0.0256
Epoch 179/1000
88/88 [==============================] - 0s - loss: 0.0254
Epoch 180/1000
88/88 [==============================] - 0s - loss: 0.0254
Epoch 181/1000
88/88 [==============================] - 0s - loss: 0.0253
Epoch 182/1000
88/88 [==============================] - 0s - loss: 0.0251
Epoch 183/1000
88/88 [==============================] - 0s - loss: 0.0252
Epoch 184/1000
88/88 [==============================] - 0s - loss: 0.0249
Epoch 185/1000
88/88 [==============================] - 0s - loss: 0.0248
Epoch 186/1000
88/88 [==============================] - 0s - loss: 0.0248
Epoch 187/1000
88/88 [==============================] - 0s - loss: 0.0245
Epoch 188/1000
88/88 [==============================] - 0s - loss: 0.0244
Epoch 189/1000
88/88 [==============================] - 0s - loss: 0.0244
Epoch 190/1000
88/88 [==============================] - 0s - loss: 0.0242
Epoch 191/1000
88/88 [==============================] - 0s - loss: 0.0241
Epoch 192/1000
88/88 [==============================] - 0s - loss: 0.0240
Epoch 193/1000
88/88 [==============================] - 0s - loss: 0.0239
Epoch 194/1000
88/88 [==============================] - 0s - loss: 0.0241
Epoch 195/1000
88/88 [==============================] - 0s - loss: 0.0238
Epoch 196/1000
88/88 [==============================] - 0s - loss: 0.0236
Epoch 197/1000
88/88 [==============================] - 0s - loss: 0.0235
```

```
Epoch 198/1000
88/88 [==============================] - 0s - loss: 0.0234
Epoch 199/1000
88/88 [==============================] - 0s - loss: 0.0233
Epoch 200/1000
88/88 [==============================] - 0s - loss: 0.0231
Epoch 201/1000
88/88 [==============================] - 0s - loss: 0.0231
Epoch 202/1000
88/88 [==============================] - 0s - loss: 0.0229
Epoch 203/1000
88/88 [==============================] - 0s - loss: 0.0229
Epoch 204/1000
88/88 [==============================] - 0s - loss: 0.0228
Epoch 205/1000
88/88 [==============================] - 0s - loss: 0.0228
Epoch 206/1000
88/88 [==============================] - 0s - loss: 0.0226
Epoch 207/1000
88/88 [==============================] - 0s - loss: 0.0224
Epoch 208/1000
88/88 [==============================] - 0s - loss: 0.0223
Epoch 209/1000
88/88 [==============================] - 0s - loss: 0.0222
Epoch 210/1000
88/88 [==============================] - 0s - loss: 0.0223
Epoch 211/1000
88/88 [==============================] - 0s - loss: 0.0221
Epoch 212/1000
88/88 [==============================] - 0s - loss: 0.0219
Epoch 213/1000
88/88 [==============================] - 0s - loss: 0.0220
Epoch 214/1000
88/88 [==============================] - 0s - loss: 0.0217
Epoch 215/1000
88/88 [==============================] - 0s - loss: 0.0219
Epoch 216/1000
88/88 [==============================] - 0s - loss: 0.0216
Epoch 217/1000
88/88 [==============================] - 0s - loss: 0.0216
Epoch 218/1000
88/88 [==============================] - 0s - loss: 0.0217
Epoch 219/1000
88/88 [==============================] - 0s - loss: 0.0215
Epoch 220/1000
88/88 [==============================] - 0s - loss: 0.0213
Epoch 221/1000
88/88 [==============================] - 0s - loss: 0.0213
Epoch 222/1000
88/88 [==============================] - 0s - loss: 0.0211
Epoch 223/1000
88/88 [==============================] - 0s - loss: 0.0211
Epoch 224/1000
88/88 [==============================] - 0s - loss: 0.0211
Epoch 225/1000
88/88 [==============================] - 0s - loss: 0.0209
Epoch 226/1000
```

```
88/88 [==============================] - ETA: 0s - loss: 0.020 - 0s - l
oss: 0.0212
Epoch 227/1000
88/88 [==============================] - 0s - loss: 0.0208
Epoch 228/1000
88/88 [==============================] - 0s - loss: 0.0207
Epoch 229/1000
88/88 [==============================] - 0s - loss: 0.0207
Epoch 230/1000
88/88 [==============================] - 0s - loss: 0.0206
Epoch 231/1000
88/88 [==============================] - 0s - loss: 0.0205
Epoch 232/1000
88/88 [==============================] - 0s - loss: 0.0205
Epoch 233/1000
88/88 [==============================] - 0s - loss: 0.0203
Epoch 234/1000
88/88 [==============================] - 0s - loss: 0.0206
Epoch 235/1000
88/88 [==============================] - 0s - loss: 0.0202
Epoch 236/1000
88/88 [==============================] - 0s - loss: 0.0201
Epoch 237/1000
88/88 [==============================] - 0s - loss: 0.0201
Epoch 238/1000
88/88 [==============================] - 0s - loss: 0.0200
Epoch 239/1000
88/88 [==============================] - 0s - loss: 0.0199
Epoch 240/1000
88/88 [==============================] - 0s - loss: 0.0202
Epoch 241/1000
88/88 [==============================] - 0s - loss: 0.0198
Epoch 242/1000
88/88 [==============================] - 0s - loss: 0.0197
Epoch 243/1000
88/88 [==============================] - 0s - loss: 0.0198
Epoch 244/1000
88/88 [==============================] - 0s - loss: 0.0196
Epoch 245/1000
88/88 [==============================] - 0s - loss: 0.0197
Epoch 246/1000
88/88 [==============================] - 0s - loss: 0.0196
Epoch 247/1000
88/88 [==============================] - 0s - loss: 0.0195
Epoch 248/1000
88/88 [==============================] - 0s - loss: 0.0196
Epoch 249/1000
88/88 [==============================] - 0s - loss: 0.0194
Epoch 250/1000
88/88 [==============================] - 0s - loss: 0.0196
Epoch 251/1000
88/88 [==============================] - 0s - loss: 0.0192
Epoch 252/1000
88/88 [==============================] - 0s - loss: 0.0192
Epoch 253/1000
88/88 [==============================] - 0s - loss: 0.0192
Epoch 254/1000
```

```
88/88 [==============================] - 0s - loss: 0.0191
Epoch 255/1000
88/88 [==============================] - 0s - loss: 0.0193
Epoch 256/1000
88/88 [==============================] - 0s - loss: 0.0190
Epoch 257/1000
88/88 [==============================] - 0s - loss: 0.0190
Epoch 258/1000
88/88 [==============================] - 0s - loss: 0.0190
Epoch 259/1000
88/88 [==============================] - 0s - loss: 0.0189
Epoch 260/1000
88/88 [==============================] - 0s - loss: 0.0188
Epoch 261/1000
88/88 [==============================] - 0s - loss: 0.0191
Epoch 262/1000
88/88 [==============================] - 0s - loss: 0.0190
Epoch 263/1000
88/88 [==============================] - 0s - loss: 0.0187
Epoch 264/1000
88/88 [==============================] - 0s - loss: 0.0187
Epoch 265/1000
88/88 [==============================] - 0s - loss: 0.0186
Epoch 266/1000
88/88 [==============================] - 0s - loss: 0.0188
Epoch 267/1000
88/88 [==============================] - 0s - loss: 0.0186
Epoch 268/1000
88/88 [==============================] - 0s - loss: 0.0188
Epoch 269/1000
88/88 [==============================] - 0s - loss: 0.0186
Epoch 270/1000
88/88 [==============================] - ETA: 0s - loss: 0.015 - 0s - l
oss: 0.0184
Epoch 271/1000
88/88 [==============================] - 0s - loss: 0.0187
Epoch 272/1000
88/88 [==============================] - 0s - loss: 0.0184
Epoch 273/1000
88/88 [==============================] - 0s - loss: 0.0184
Epoch 274/1000
88/88 [==============================] - 0s - loss: 0.0183
Epoch 275/1000
88/88 [==============================] - 0s - loss: 0.0183
Epoch 276/1000
88/88 [==============================] - 0s - loss: 0.0183
Epoch 277/1000
88/88 [==============================] - 0s - loss: 0.0186
Epoch 278/1000
88/88 [==============================] - 0s - loss: 0.0182
Epoch 279/1000
88/88 [==============================] - 0s - loss: 0.0183
Epoch 280/1000
88/88 [==============================] - 0s - loss: 0.0181
Epoch 281/1000
88/88 [==============================] - 0s - loss: 0.0181
Epoch 282/1000
```

```
88/88 [==============================] - 0s - loss: 0.0183
Epoch 283/1000
88/88 [==============================] - 0s - loss: 0.0183
Epoch 284/1000
88/88 [==============================] - 0s - loss: 0.0181
Epoch 285/1000
88/88 [==============================] - 0s - loss: 0.0180
Epoch 286/1000
88/88 [==============================] - 0s - loss: 0.0182
Epoch 287/1000
88/88 [==============================] - 0s - loss: 0.0180
Epoch 288/1000
88/88 [==============================] - 0s - loss: 0.0180
Epoch 289/1000
88/88 [==============================] - 0s - loss: 0.0180
Epoch 290/1000
88/88 [==============================] - 0s - loss: 0.0180
Epoch 291/1000
88/88 [==============================] - 0s - loss: 0.0180
Epoch 292/1000
88/88 [==============================] - 0s - loss: 0.0179
Epoch 293/1000
88/88 [==============================] - 0s - loss: 0.0179
Epoch 294/1000
88/88 [==============================] - 0s - loss: 0.0179
Epoch 295/1000
88/88 [==============================] - 0s - loss: 0.0179
Epoch 296/1000
88/88 [==============================] - 0s - loss: 0.0180
Epoch 297/1000
88/88 [==============================] - 0s - loss: 0.0179
Epoch 298/1000
88/88 [==============================] - 0s - loss: 0.0179
Epoch 299/1000
88/88 [==============================] - 0s - loss: 0.0177
Epoch 300/1000
88/88 [==============================] - 0s - loss: 0.0177
Epoch 301/1000
88/88 [==============================] - 0s - loss: 0.0178
Epoch 302/1000
88/88 [==============================] - 0s - loss: 0.0177
Epoch 303/1000
88/88 [==============================] - 0s - loss: 0.0179
Epoch 304/1000
88/88 [==============================] - 0s - loss: 0.0176
Epoch 305/1000
88/88 [==============================] - 0s - loss: 0.0177
Epoch 306/1000
88/88 [==============================] - 0s - loss: 0.0178
Epoch 307/1000
88/88 [==============================] - 0s - loss: 0.0178
Epoch 308/1000
88/88 [==============================] - 0s - loss: 0.0176
Epoch 309/1000
88/88 [==============================] - 0s - loss: 0.0176
Epoch 310/1000
88/88 [==============================] - 0s - loss: 0.0178
```

```
Epoch 311/1000
88/88 [==============================] - ETA: 0s - loss: 0.015 - 0s - l
oss: 0.0175
Epoch 312/1000
88/88 [==============================] - ETA: 0s - loss: 0.014 - 0s - l
oss: 0.0176
Epoch 313/1000
88/88 [==============================] - 0s - loss: 0.0176
Epoch 314/1000
88/88 [==============================] - 0s - loss: 0.0175
Epoch 315/1000
88/88 [==============================] - ETA: 0s - loss: 0.013 - 0s - l
oss: 0.0176
Epoch 316/1000
88/88 [==============================] - 0s - loss: 0.0175
Epoch 317/1000
88/88 [==============================] - 0s - loss: 0.0178
Epoch 318/1000
88/88 [==============================] - 0s - loss: 0.0175
Epoch 319/1000
88/88 [==============================] - 0s - loss: 0.0175
Epoch 320/1000
88/88 [==============================] - 0s - loss: 0.0177
Epoch 321/1000
88/88 [==============================] - 0s - loss: 0.0177
Epoch 322/1000
88/88 [==============================] - 0s - loss: 0.0175
Epoch 323/1000
88/88 [==============================] - 0s - loss: 0.0175
Epoch 324/1000
88/88 [==============================] - 0s - loss: 0.0174
Epoch 325/1000
88/88 [==============================] - ETA: 0s - loss: 0.015 - 0s - l
oss: 0.0179
Epoch 326/1000
88/88 [==============================] - 0s - loss: 0.0174
Epoch 327/1000
88/88 [==============================] - 0s - loss: 0.0174
Epoch 328/1000
88/88 [==============================] - 0s - loss: 0.0175
Epoch 329/1000
88/88 [==============================] - 0s - loss: 0.0175
Epoch 330/1000
88/88 [==============================] - 0s - loss: 0.0176
Epoch 331/1000
88/88 [==============================] - 0s - loss: 0.0174
Epoch 332/1000
88/88 [==============================] - 0s - loss: 0.0174
Epoch 333/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 334/1000
88/88 [==============================] - 0s - loss: 0.0174
Epoch 335/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 336/1000
88/88 [==============================] - 0s - loss: 0.0175
Epoch 337/1000
```

```
88/88 [==============================] - 0s - loss: 0.0176
Epoch 338/1000
88/88 [==============================] - 0s - loss: 0.0174
Epoch 339/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 340/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 341/1000
88/88 [==============================] - 0s - loss: 0.0175
Epoch 342/1000
88/88 [==============================] - 0s - loss: 0.0174
Epoch 343/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 344/1000
88/88 [==============================] - ETA: 0s - loss: 0.014 - 0s - l
oss: 0.0172
Epoch 345/1000
88/88 [==============================] - 0s - loss: 0.0176
Epoch 346/1000
88/88 [==============================] - 0s - loss: 0.0174
Epoch 347/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 348/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 349/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 350/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 351/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 352/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 353/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 354/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 355/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 356/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 357/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 358/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 359/1000
88/88 [==============================] - 0s - loss: 0.0175
Epoch 360/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 361/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 362/1000
88/88 [==============================] - 0s - loss: 0.0174
Epoch 363/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 364/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 365/1000
```

```
88/88 [==============================] - 0s - loss: 0.0172
Epoch 366/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 367/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 368/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 369/1000
88/88 [==============================] - ETA: 0s - loss: 0.019 - 0s - l
oss: 0.0171
Epoch 370/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 371/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 372/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 373/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 374/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 375/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 376/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 377/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 378/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 379/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 380/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 381/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 382/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 383/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 384/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 385/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 386/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 387/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 388/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 389/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 390/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 391/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 392/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 393/1000
```

```
88/88 [==============================] - 0s - loss: 0.0170
Epoch 394/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 395/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 396/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 397/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 398/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 399/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 400/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 401/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 402/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 403/1000
88/88 [==============================] - 0s - loss: 0.0175
Epoch 404/1000
88/88 [==============================] - ETA: 0s - loss: 0.020 - 0s - l
oss: 0.0170
Epoch 405/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 406/1000
88/88 [==============================] - ETA: 0s - loss: 0.017 - 0s - l
oss: 0.0169
Epoch 407/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 408/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 409/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 410/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 411/1000
88/88 [==============================] - 0s - loss: 0.0175
Epoch 412/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 413/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 414/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 415/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 416/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 417/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 418/1000
88/88 [==============================] - 0s - loss: 0.0173
Epoch 419/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 420/1000
88/88 [==============================] - 0s - loss: 0.0169
```

```
Epoch 421/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 422/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 423/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 424/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 425/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 426/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 427/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 428/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 429/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 430/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 431/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 432/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 433/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 434/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 435/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 436/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 437/1000
88/88 [==============================] - 0s - loss: 0.0176
Epoch 438/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 439/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 440/1000
88/88 [==============================] - ETA: 0s - loss: 0.012 - 0s - l
oss: 0.0171
Epoch 441/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 442/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 443/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 444/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 445/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 446/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 447/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 448/1000
88/88 [==============================] - 0s - loss: 0.0169
```

```
Epoch 449/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 450/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 451/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 452/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 453/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 454/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 455/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 456/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 457/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 458/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 459/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 460/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 461/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 462/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 463/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 464/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 465/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 466/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 467/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 468/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 469/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 470/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 471/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 472/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 473/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 474/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 475/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 476/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 477/1000
```

```
88/88 [==============================] - 0s - loss: 0.0167
Epoch 478/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 479/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 480/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 481/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 482/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 483/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 484/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 485/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 486/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 487/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 488/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 489/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 490/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 491/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 492/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 493/1000
88/88 [==============================] - 0s - loss: 0.0172
Epoch 494/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 495/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 496/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 497/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 498/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 499/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 500/1000
88/88 [==============================] - ETA: 0s - loss: 0.019 - 0s - l
oss: 0.0167
Epoch 501/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 502/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 503/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 504/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 505/1000
```

```
88/88 [==============================] - 0s - loss: 0.0168
Epoch 506/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 507/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 508/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 509/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 510/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 511/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 512/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 513/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 514/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 515/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 516/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 517/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 518/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 519/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 520/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 521/1000
88/88 [==============================] - 0s - loss: 0.0170
Epoch 522/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 523/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 524/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 525/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 526/1000
88/88 [==============================] - ETA: 0s - loss: 0.012 - 0s - l
oss: 0.0167
Epoch 527/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 528/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 529/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 530/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 531/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 532/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 533/1000
```

```
88/88 [==============================] - 0s - loss: 0.0170
Epoch 534/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 535/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 536/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 537/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 538/1000
88/88 [==============================] - 0s - loss: 0.0171
Epoch 539/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 540/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 541/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 542/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 543/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 544/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 545/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 546/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 547/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 548/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 549/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 550/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 551/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 552/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 553/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 554/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 555/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 556/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 557/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 558/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 559/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 560/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 561/1000
88/88 [==============================] - 0s - loss: 0.0166
```

```
Epoch 562/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 563/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 564/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 565/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 566/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 567/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 568/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 569/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 570/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 571/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 572/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 573/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 574/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 575/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 576/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 577/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 578/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 579/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 580/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 581/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 582/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 583/1000
88/88 [==============================] - ETA: 0s - loss: 0.019 - 0s - l
oss: 0.0166
Epoch 584/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 585/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 586/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 587/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 588/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 589/1000
88/88 [==============================] - 0s - loss: 0.0165
```

```
Epoch 590/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 591/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 592/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 593/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 594/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 595/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 596/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 597/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 598/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 599/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 600/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 601/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 602/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 603/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 604/1000
88/88 [==============================] - ETA: 0s - loss: 0.018 - 0s - l
oss: 0.0166
Epoch 605/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 606/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 607/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 608/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 609/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 610/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 611/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 612/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 613/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 614/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 615/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 616/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 617/1000
88/88 [==============================] - 0s - loss: 0.0167
```

```
Epoch 618/1000
88/88 [==============================] - ETA: 0s - loss: 0.013 - 0s - l
oss: 0.0164
Epoch 619/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 620/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 621/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 622/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 623/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 624/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 625/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 626/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 627/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 628/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 629/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 630/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 631/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 632/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 633/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 634/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 635/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 636/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 637/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 638/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 639/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 640/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 641/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 642/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 643/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 644/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 645/1000
88/88 [==============================] - 0s - loss: 0.0164
```

```
Epoch 646/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 647/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 648/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 649/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 650/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 651/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 652/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 653/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 654/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 655/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 656/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 657/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 658/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 659/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 660/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 661/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 662/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 663/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 664/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 665/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 666/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 667/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 668/1000
88/88 [==============================] - ETA: 0s - loss: 0.017 - 0s - l
oss: 0.0165
Epoch 669/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 670/1000
88/88 [==============================] - ETA: 0s - loss: 0.016 - 0s - l
oss: 0.0165
Epoch 671/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 672/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 673/1000
```

```
88/88 [==============================] - 0s - loss: 0.0164
Epoch 674/1000
88/88 [==============================] - ETA: 0s - loss: 0.019 - 0s - l
oss: 0.0164
Epoch 675/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 676/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 677/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 678/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 679/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 680/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 681/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 682/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 683/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 684/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 685/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 686/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 687/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 688/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 689/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 690/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 691/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 692/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 693/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 694/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 695/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 696/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 697/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 698/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 699/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 700/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 701/1000
```

```
88/88 [==============================] - 0s - loss: 0.0167
Epoch 702/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 703/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 704/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 705/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 706/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 707/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 708/1000
88/88 [==============================] - ETA: 0s - loss: 0.018 - 0s - l
oss: 0.0163
Epoch 709/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 710/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 711/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 712/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 713/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 714/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 715/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 716/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 717/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 718/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 719/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 720/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 721/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 722/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 723/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 724/1000
88/88 [==============================] - 0s - loss: 0.0169
Epoch 725/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 726/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 727/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 728/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 729/1000
```

```
88/88 [==============================] - 0s - loss: 0.0164
Epoch 730/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 731/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 732/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 733/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 734/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 735/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 736/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 737/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 738/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 739/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 740/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 741/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 742/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 743/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 744/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 745/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 746/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 747/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 748/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 749/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 750/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 751/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 752/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 753/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 754/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 755/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 756/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 757/1000
88/88 [==============================] - 0s - loss: 0.0163
```

```
Epoch 758/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 759/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 760/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 761/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 762/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 763/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 764/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 765/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 766/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 767/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 768/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 769/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 770/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 771/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 772/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 773/1000
88/88 [==============================] - ETA: 0s - loss: 0.017 - 0s - l
oss: 0.0167
Epoch 774/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 775/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 776/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 777/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 778/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 779/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 780/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 781/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 782/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 783/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 784/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 785/1000
88/88 [==============================] - 0s - loss: 0.0166
```

```
Epoch 786/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 787/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 788/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 789/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 790/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 791/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 792/1000
88/88 [==============================] - ETA: 0s - loss: 0.019 - 0s - l
oss: 0.0164
Epoch 793/1000
88/88 [==============================] - ETA: 0s - loss: 0.015 - 0s - l
oss: 0.0164
Epoch 794/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 795/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 796/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 797/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 798/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 799/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 800/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 801/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 802/1000
88/88 [==============================] - ETA: 0s - loss: 0.014 - 0s - l
oss: 0.0162
Epoch 803/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 804/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 805/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 806/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 807/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 808/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 809/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 810/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 811/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 812/1000
88/88 [==============================] - 0s - loss: 0.0162
```

```
Epoch 813/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 814/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 815/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 816/1000
88/88 [==============================] - 0s - loss: 0.0167
Epoch 817/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 818/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 819/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 820/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 821/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 822/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 823/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 824/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 825/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 826/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 827/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 828/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 829/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 830/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 831/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 832/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 833/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 834/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 835/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 836/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 837/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 838/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 839/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 840/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 841/1000
```

```
88/88 [==============================] - 0s - loss: 0.0162
Epoch 842/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 843/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 844/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 845/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 846/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 847/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 848/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 849/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 850/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 851/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 852/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 853/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 854/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 855/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 856/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 857/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 858/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 859/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 860/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 861/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 862/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 863/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 864/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 865/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 866/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 867/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 868/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 869/1000
88/88 [==============================] - 0s - loss: 0.0163
```

```
Epoch 870/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 871/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 872/1000
88/88 [==============================] - ETA: 0s - loss: 0.019 - 0s - l
oss: 0.0162
Epoch 873/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 874/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 875/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 876/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 877/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 878/1000
88/88 [==============================] - 0s - loss: 0.0168
Epoch 879/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 880/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 881/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 882/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 883/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 884/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 885/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 886/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 887/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 888/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 889/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 890/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 891/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 892/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 893/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 894/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 895/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 896/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 897/1000
88/88 [==============================] - 0s - loss: 0.0163
```

```
Epoch 898/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 899/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 900/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 901/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 902/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 903/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 904/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 905/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 906/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 907/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 908/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 909/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 910/1000
88/88 [==============================] - ETA: 0s - loss: 0.016 - 0s - l
oss: 0.0162
Epoch 911/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 912/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 913/1000
88/88 [==============================] - ETA: 0s - loss: 0.020 - 0s - l
oss: 0.0161
Epoch 914/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 915/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 916/1000
88/88 [==============================] - ETA: 0s - loss: 0.020 - 0s - l
oss: 0.0162
Epoch 917/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 918/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 919/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 920/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 921/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 922/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 923/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 924/1000
88/88 [==============================] - 0s - loss: 0.0161
```

```
Epoch 925/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 926/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 927/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 928/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 929/1000
88/88 [==============================] - 0s - loss: 0.0165
Epoch 930/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 931/1000
88/88 [==============================] - 0s - loss: 0.0160
Epoch 932/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 933/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 934/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 935/1000
88/88 [==============================] - 0s - loss: 0.0160
Epoch 936/1000
88/88 [==============================] - 0s - loss: 0.0164
Epoch 937/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 938/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 939/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 940/1000
88/88 [==============================] - 0s - loss: 0.0166
Epoch 941/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 942/1000
88/88 [==============================] - 0s - loss: 0.0160
Epoch 943/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 944/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 945/1000
88/88 [==============================] - 0s - loss: 0.0160
Epoch 946/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 947/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 948/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 949/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 950/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 951/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 952/1000
88/88 [==============================] - 0s - loss: 0.0160
Epoch 953/1000
```

```
88/88 [==============================] - 0s - loss: 0.0164
Epoch 954/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 955/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 956/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 957/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 958/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 959/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 960/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 961/1000
88/88 [==============================] - ETA: 0s - loss: 0.017 - 0s - l
oss: 0.0160
Epoch 962/1000
88/88 [==============================] - 0s - loss: 0.0160
Epoch 963/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 964/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 965/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 966/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 967/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 968/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 969/1000
88/88 [==============================] - 0s - loss: 0.0160
Epoch 970/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 971/1000
88/88 [==============================] - 0s - loss: 0.0160
Epoch 972/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 973/1000
88/88 [==============================] - ETA: 0s - loss: 0.014 - 0s - l
oss: 0.0160
Epoch 974/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 975/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 976/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 977/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 978/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 979/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 980/1000
88/88 [==============================] - 0s - loss: 0.0166
```

```
Epoch 981/1000
88/88 [==============================] - 0s - loss: 0.0163
Epoch 982/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 983/1000
88/88 [==============================] - 0s - loss: 0.0160
Epoch 984/1000
88/88 [==============================] - 0s - loss: 0.0160
Epoch 985/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 986/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 987/1000
88/88 [==============================] - 0s - loss: 0.0160
Epoch 988/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 989/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 990/1000
88/88 [==============================] - 0s - loss: 0.0160
Epoch 991/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 992/1000
88/88 [==============================] - 0s - loss: 0.0160
Epoch 993/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 994/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 995/1000
88/88 [==============================] - 0s - loss: 0.0160
Epoch 996/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 997/1000
88/88 [==============================] - 0s - loss: 0.0160
Epoch 998/1000
88/88 [==============================] - 0s - loss: 0.0161
Epoch 999/1000
88/88 [==============================] - 0s - loss: 0.0162
Epoch 1000/1000
88/88 [==============================] - 0s - loss: 0.0162
```

Out[9]:
```
<keras.callbacks.History at 0x7fbc3cd1cb38>
```

1.5 Checking model performance

With your model fit we can now make predictions on both our training and testing sets.

In [10]:
```python
# generate predictions for training
train_predict = model.predict(X_train)
test_predict = model.predict(X_test)
```

In the next cell we compute training and testing errors using our trained model - you should be able to achieve at least

$training\_error < 0.02$

and

$testing\_error < 0.02$

with your fully trained model.

If either or both of your accuracies are larger than 0.02 re-train your model - increasing the number of epochs you take (a maximum of around 1,000 should do the job) and/or adjusting your batch_size.

In [11]:
```
# print out training and testing errors
training_error = model.evaluate(X_train, y_train, verbose=0)
print('training error = ' + str(training_error))

testing_error = model.evaluate(X_test, y_test, verbose=0)
print('testing error = ' + str(testing_error))
```

```
training error = 0.0159912935712
testing error = 0.0139957363826
```

Activating the next cell plots the original data, as well as both predictions on the training and testing sets.

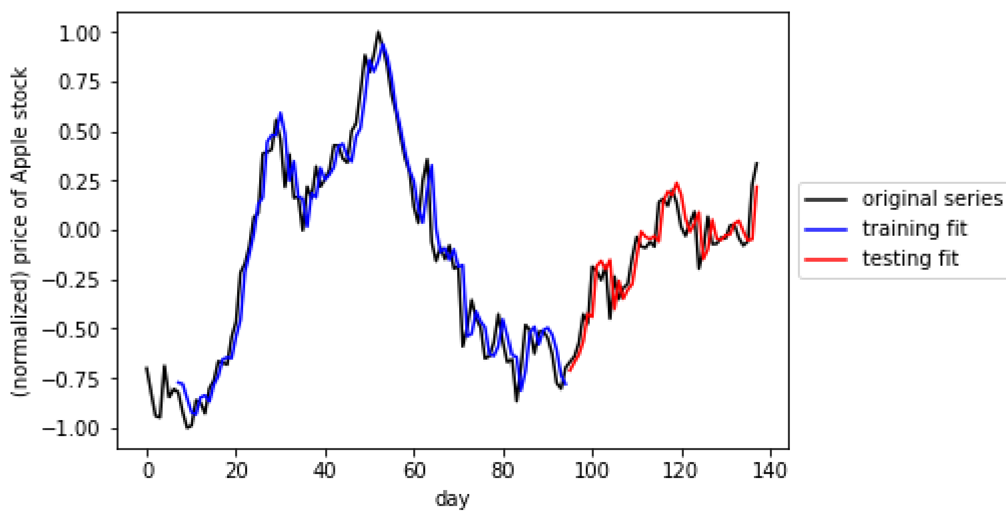
In [12]:
```
### Plot everything - the original series as well as predictions on trai
ning and testing sets
import matplotlib.pyplot as plt
%matplotlib inline

# plot original series
plt.plot(dataset,color = 'k')

# plot training set prediction
split_pt = train_test_split + window_size
plt.plot(np.arange(window_size,split_pt,1),train_predict,color = 'b')

# plot testing set prediction
plt.plot(np.arange(split_pt,split_pt +
len(test_predict),1),test_predict,color = 'r')

# pretty up graph
plt.xlabel('day')
plt.ylabel('(normalized) price of Apple stock')
plt.legend(['original series','training fit','testing fit'],loc='center
 left', bbox_to_anchor=(1, 0.5))
plt.show()
```



Note: you can try out any time series for this exercise! If you would like to try another see e.g., this site containing thousands of time series (https://datamarket.com/data/list/?q=provider%3Atsdl) and pick another one!

Problem 2: Create a sequence generator

2.1 Getting started

In this project you will implement a popular Recurrent Neural Network (RNN) architecture to create an English language sequence generator capable of building semi-coherent English sentences from scratch by building them up character-by-character. This will require a substantial amount amount of parameter tuning on a large training corpus (at least 100,000 characters long). In particular for this project we will be using a complete version of Sir Arthur Conan Doyle's classic book The Adventures of Sherlock Holmes.

How can we train a machine learning model to generate text automatically, character-by-character? *By showing the model many training examples so it can learn a pattern between input and output.* With this type of text generation each input is a string of valid characters like this one

dogs are grea

whlie the corresponding output is the next character in the sentence - which here is 't' (since the complete sentence is 'dogs are great'). We need to show a model many such examples in order for it to make reasonable predictions.

Fun note: For those interested in how text generation is being used check out some of the following fun resources:

- Generate wacky sentences (http://www.cs.toronto.edu/~ilya/rnn.html) with this academic RNN text generator
- Various twitter bots that tweet automatically generated text likethis one (http://tweet-generator-alex.herokuapp.com/).
- the NanoGenMo (https://github.com/NaNoGenMo/2016) annual contest to automatically produce a 50,000+ novel automatically
- Robot Shakespeare (https://github.com/genekogan/RobotShakespeare) a text generator that automatically produces Shakespear-esk sentences

2.2 Preprocessing a text dataset

Our first task is to get a large text corpus for use in training, and on it we perform a several light pre-processing tasks. The default corpus we will use is the classic book Sherlock Holmes, but you can use a variety of others as well - so long as they are fairly large (around 100,000 characters or more).

In [1]:

```
# read in the text, transforming everything to lower case
text = open('datasets/holmes.txt').read().lower()
print('our original text has ' + str(len(text)) + ' characters')
```

```
our original text has 581864 characters
```

Next, lets examine a bit of the raw text. Because we are interested in creating sentences of English words automatically by building up each word character-by-character, we only want to train on valid English words. In other words - we need to remove all of the other junk characters that aren't words!

In [2]:
```
### print out the first 1000 characters of the raw text to get a sense o
f what we need to throw out
text[:2000]
```

Out[2]:
```
"\ufeffproject gutenberg's the adventures of sherlock holmes, by arthur
conan doyle\n\nthis ebook is for the use of anyone anywhere at no cost
 and with\nalmost no restrictions whatsoever.  you may copy it, give it
away or\nre-use it under the terms of the project gutenberg license inc
luded\nwith this ebook or online at www.gutenberg.net\n\n\ntitle: the a
dventures of sherlock holmes\n\nauthor: arthur conan doyle\n\nposting d
ate: april 18, 2011 [ebook #1661]\nfirst posted: november 29, 2002\n\nl
anguage: english\n\n\n*** start of this project gutenberg ebook the adv
entures of sherlock holmes ***\n\n\n\n\nproduced by an anonymous projec
t gutenberg volunteer and jose menendez\n\n\n\n\n\n\n\n\n\nthe adventur
es of sherlock holmes\n\nby\n\nsir arthur conan doyle\n\n\n\n\n   i. a sc
andal in bohemia\n  ii. the red-headed league\n iii. a case of identity
\n  iv. the boscombe valley mystery\n   v. the five orange pips\n  vi.
 the man with the twisted lip\n vii. the adventure of the blue carbuncl
e\nviii. the adventure of the speckled band\n  ix. the adventure of the
engineer's thumb\n   x. the adventure of the noble bachelor\n  xi. the
 adventure of the beryl coronet\n xii. the adventure of the copper beec
hes\n\n\n\n\nadventure i. a scandal in bohemia\n\ni.\n\nto sherlock hol
mes she is always the woman. i have seldom heard\nhim mention her under
any other name. in his eyes she eclipses\nand predominates the whole of
her sex. it was not that he felt\nany emotion akin to love for irene ad
ler. all emotions, and that\none particularly, were abhorrent to his co
ld, precise but\nadmirably balanced mind. he was, i take it, the most p
erfect\nreasoning and observing machine that the world has seen, but as
a\nlover he would have placed himself in a false position. he never\nsp
oke of the softer passions, save with a gibe and a sneer. they\nwere ad
mirable things for the observer--excellent for drawing the\nveil from m
en's motives and actions. but for the trained reasoner\nto admit such i
ntrusions into his own delicate and finely\nadjusted temperament was to
introduce a dist"
```

Wow - there's a lot of junk here (i.e., weird uncommon character combinations - as this first character chunk contains the title and author page, as well as table of contents)! e.g., all the carriage return and newline sequences '\n' and '\r' sequences. We want to train our RNN on a large chunk of real english sentences - we don't want it to start thinking non-english words or strange characters are valid! - so lets clean up the data a bit.

First, since the dataset is so large and the first few hundred characters contain a lot of junk, lets cut it out. Lets also find-and-replace those newline tags with empty spaces.

In [3]:
```
### find and replace '\n' and '\r' symbols - replacing them
text = text[1302:]
text = text.replace('\n',' ')    # replacing '\n' with '' simply removes
 the sequence
text = text.replace('\r',' ')
```

Lets see how the first 1000 characters of our text looks now!

In [4]:
```
### print out the first 1000 characters of the raw text to get a sense o
f what we need to throw out
text[:1000]
```

Out[4]:
```
"is eyes she eclipses and predominates the whole of her sex. it was not
that he felt any emotion akin to love for irene adler. all emotions, an
d that one particularly, were abhorrent to his cold, precise but admira
bly balanced mind. he was, i take it, the most perfect reasoning and ob
serving machine that the world has seen, but as a lover he would have p
laced himself in a false position. he never spoke of the softer passion
s, save with a gibe and a sneer. they were admirable things for the obs
erver--excellent for drawing the veil from men's motives and actions. b
ut for the trained reasoner to admit such intrusions into his own delic
ate and finely adjusted temperament was to introduce a distracting fact
or which might throw a doubt upon all his mental results. grit in a sen
sitive instrument, or a crack in one of his own high-power lenses, woul
d not be more disturbing than a strong emotion in a nature such as his.
and yet there was but one woman to him, and that woman was the late ire
ne ad"
```

TODO: finish cleaning the text

Lets make sure we haven't left any other non-English/proper punctuation (commas, periods, etc., are ok) characters lurking around in the depths of the text. You can do this by ennumerating all the text's unique characters, examining them, and then replacing any unwanted (non-english) characters with empty spaces! Once we find all of the text's unique characters, we can remove all of the non-English/proper punctuation ones in the next cell. Note: don't remove necessary punctuation marks! (given in the cell below).

(remember to copy your completed function into the script *my_answers.py* function titled *clean_text* before submitting your project)

In [5]:
```
### ist all unique characters in the text and remove any non-english one
s
# find all unique characters in the text
uniques = set(text)
#print(uniques)

# remove as many non-english characters and character sequences as you c
an

# get all valid characters (alphabets, digits, punctuation)
import string
valid_characters = list(string.ascii_letters)
valid_characters = valid_characters + list(string.digits)
valid_characters = valid_characters + list(string.punctuation)
valid_characters = valid_characters + list(string.whitespace)
#print(valid_characters)

# determine characters to be removed
invalid_characters = uniques.difference(valid_characters)
#print(invalid_characters)
for c in invalid_characters:
    text = text.replace(c, '')

# shorten any extra dead space created above
text = text.replace('  ',' ')
```

With your chosen characters removed print out the first few hundred lines again just to double check that everything looks good.

In [6]:
```
### print out the first 2000 characters of the raw text to get a sense o
f what we need to throw out
text[:2000]
```

Out[6]:
```
"is eyes she eclipses and predominates the whole of her sex. it was not
that he felt any emotion akin to love for irene adler. all emotions, an
d that one particularly, were abhorrent to his cold, precise but admira
bly balanced mind. he was, i take it, the most perfect reasoning and ob
serving machine that the world has seen, but as a lover he would have p
laced himself in a false position. he never spoke of the softer passion
s, save with a gibe and a sneer. they were admirable things for the obs
erver--excellent for drawing the veil from men's motives and actions. b
ut for the trained reasoner to admit such intrusions into his own delic
ate and finely adjusted temperament was to introduce a distracting fact
or which might throw a doubt upon all his mental results. grit in a sen
sitive instrument, or a crack in one of his own high-power lenses, woul
d not be more disturbing than a strong emotion in a nature such as his.
and yet there was but one woman to him, and that woman was the late ire
ne adler, of dubious and questionable memory. i had seen little of holm
es lately. my marriage had drifted us away from each other. my own comp
lete happiness, and the home-centred interests which rise up around the
man who first finds himself master of his own establishment, were suffi
cient to absorb all my attention, while holmes, who loathed every form
 of society with his whole bohemian soul, remained in our lodgings in b
aker street, buried among his old books, and alternating from week to w
eek between cocaine and ambition, the drowsiness of the drug, and the f
ierce energy of his own keen nature. he was still, as ever, deeply attr
acted by the study of crime, and occupied his immense faculties and ext
raordinary powers of observation in following out those clues, and clea
ring up those mysteries which had been abandoned as hopeless by the off
icial police. from time to time i heard some vague account of his doing
s: of his summons to odessa in the case of the trepoff murder, of his c
learing up "
```

Now that we have thrown out a good number of non-English characters/character sequences lets print out some statistics about the dataset - including number of total characters and number of unique characters.

In [7]:
```
# count the number of unique characters in the text
chars = sorted(list(set(text)))

# print some of the text, as well as statistics
print ("this corpus has " +  str(len(text)) + " total number of characte
rs")
print ("this corpus has " +  str(len(chars)) + " unique characters")
```

```
this corpus has 577662 total number of characters
this corpus has 54 unique characters
```

2.3 Cutting data into input/output pairs

Now that we have our text all cleaned up, how can we use it to train a model to generate sentences automatically? First we need to train a machine learning model - and in order to do that we need a set of input/output pairs for a model to train on. How can we create a set of input/output pairs from our text to train on?

Remember in part 1 of this notebook how we used a sliding window to extract input/output pairs from a time series? We do the same thing here! We slide a window of length T along our giant text corpus - everything in the window becomes one input while the character following becomes its corresponding output. This process of extracting input/output pairs is illustrated in the gif below on a small example text using a window size of T = 5.

`d o g s   a r e   g r e a t`

Notice one aspect of the sliding window in this gif that does not mirror the analaogous gif for time series shown in part 1 of the notebook - we do not need to slide the window along one character at a time but can move by a fixed step size M greater than 1 (in the gif indeed $M = 1$). This is done with large input texts (like ours which has over 500,000 characters!) when sliding the window along one character at a time we would create far too many input/output pairs to be able to reasonably compute with.

More formally lets denote our text corpus - which is one long string of characters - as follows

$$s_0, s_1, s_2, \ldots, s_P$$

where P is the length of the text (again for our text $P \approx 500,000!$). Sliding a window of size T = 5 with a step length of M = 1 (these are the parameters shown in the gif above) over this sequence produces the following list of input/output pairs

Input	Output
$\langle s_1, s_2, s_3, s_4, s_5 \rangle$	s_6
$\langle s_2, s_3, s_4, s_5, s_6 \rangle$	s_7
$\vdots$	$\vdots$
$\langle s_{P-5}, s_{P-4}, s_{P-3}, s_{P-2}, s_{P-1} \rangle$	s_P

Notice here that each input is a sequence (or vector) of 4 characters (and in general has length equal to the window size T) while each corresponding output is a single character. We created around P total number of input/output pairs (for general step size M we create around ceil(P/M) pairs).

Now its time for you to window the input time series as described above!

TODO: Create a function that runs a sliding window along the input text and creates associated input/output pairs. A skeleton function has been provided for you. Note that this function should input a) the text b) the window size and c) the step size, and return the input/output sequences. Note: the return items should be *lists* - not numpy arrays.

(remember to copy your completed function into the script *my_answers.py* function titled *window_transform_text* before submitting your project)

In [8]:
```
### fill out the function below that transforms the input text and windo
w-size into a set of input/output pairs for use with our RNN model
import numpy as np
def window_transform_text(text,window_size,step_size):
    # containers for input/output pairs
    inputs = []
    outputs = []

    # Slice the series into windows
    for i in range(window_size, len(text), step_size):
        inputs.append(text[i-window_size:i])
        outputs.append(text[i])

    return inputs,outputs
```

With our function complete we can now use it to produce input/output pairs! We employ the function in the next cell, where the window_size = 50 and step_size = 5.

In [9]:
```
# run your text window-ing function
window_size = 100
step_size = 5
inputs, outputs = window_transform_text(text,window_size,step_size)
```

Lets print out a few input/output pairs to verify that we have made the right sort of stuff!

In [10]:
```
# print out a few of the input/output pairs to verify that we've made th
e right kind of stuff to learn from
print('input = ' + inputs[2])
print('output = ' + outputs[2])
print('--------------')
print('input = ' + inputs[100])
print('output = ' + outputs[100])
```

```
input = e eclipses and predominates the whole of her sex. it was not th
at he felt any emotion akin to love f
output = o
--------------
input = er--excellent for drawing the veil from men's motives and actio
ns. but for the trained reasoner to a
output = d
```

Looks good!

2.4 Wait, what kind of problem is text generation again?

In part 1 of this notebook we used the same pre-processing technique - the sliding window - to produce a set of training input/output pairs to tackle the problem of time series prediction *by treating the problem as one of regression*. So what sort of problem do we have here now, with text generation? Well, the time series prediction was a regression problem because the output (one value of the time series) was a continuous value. Here - for character-by-character text generation - each output is a *single character*. This isn't a continuous value - but a distinct class - therefore **character-by-character text generation is a classification problem**.

How many classes are there in the data? Well, the number of classes is equal to the number of unique characters we have to predict! How many of those were there in our dataset again? Lets print out the value again.

In [11]:
```
# print out the number of unique characters in the dataset
chars = sorted(list(set(text)))
print ("this corpus has " +  str(len(chars)) + " unique characters")
print ('and these characters are ')
print (chars)
```

```
this corpus has 54 unique characters
and these characters are 
[' ', '!', '"', '$', '%', '&', "'", '(', ')', '*', ',', '-', '.', '/',
 '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', ':', ';', '?', '@',
 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n',
 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']
```

Rockin' - so we have a multi-class classification problem on our hands!

2.5 One-hot encoding characters

There's just one last issue we have to deal with before tackle: machine learning algorithm deal with numerical data and all of our input/output pairs are characters. So we just need to transform our characters into equivalent numerical values. The most common way of doing this is via a 'one-hot encoding' scheme. Here's how it works.

We transform each character in our inputs/outputs into a vector with length equal to the number of unique characters in our text. This vector is all zeros except one location where we place a 1 - and this location is unique to each character type. e.g., we transform 'a', 'b', and 'c' as follows

$$a \longleftarrow \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \quad b \longleftarrow \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \quad c \longleftarrow \begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \cdots$$

where each vector has 32 entries (or in general: number of entries = number of unique characters in text).

The first practical step towards doing this one-hot encoding is to form a dictionary mapping each unique character to a unique integer, and one dictionary to do the reverse mapping. We can then use these dictionaries to quickly make our one-hot encodings, as well as re-translate (from integers to characters) the results of our trained RNN classification model.

In [12]:
```
# this dictionary is a function mapping each unique character to a uniqu
e integer
chars_to_indices = dict((c, i) for i, c in enumerate(chars))  # map each
 unique character to unique integer

# this dictionary is a function mapping each unique integer back to a un
ique character
indices_to_chars = dict((i, c) for i, c in enumerate(chars))  # map each
 unique integer back to unique character
```

Now we can transform our input/output pairs - consisting of characters - to equivalent input/output pairs made up of one-hot encoded vectors. In the next cell we provide a function for doing just this: it takes in the raw character input/outputs and returns their numerical versions. In particular the numerical input is given as $\mathbf{X}$, and numerical output is given as the $\mathbf{y}$

In [13]:
```
# transform character-based input/output into equivalent numerical versi
ons
def encode_io_pairs(text,window_size,step_size):
    # number of unique chars
    chars = sorted(list(set(text)))
    num_chars = len(chars)

    # cut up text into character input/output pairs
    inputs, outputs = window_transform_text(text,window_size,step_size)

    # create empty vessels for one-hot encoded input/output
    X = np.zeros((len(inputs), window_size, num_chars), dtype=np.bool)
    y = np.zeros((len(inputs), num_chars), dtype=np.bool)

    # loop over inputs/outputs and tranform and store in X/y
    for i, sentence in enumerate(inputs):
        for t, char in enumerate(sentence):
            X[i, t, chars_to_indices[char]] = 1
        y[i, chars_to_indices[outputs[i]]] = 1

    return X,y
```

Now run the one-hot encoding function by activating the cell below and transform our input/output pairs!

In [14]:
```
# use your function
window_size = 100
step_size = 5
X,y = encode_io_pairs(text,window_size,step_size)
```

2.6 Setting up our RNN

With our dataset loaded and the input/output pairs extracted / transformed we can now begin setting up our RNN for training. Again we will use Keras to quickly build a single hidden layer RNN - where our hidden layer consists of LTSM modules.

Time to get to work: build a 3 layer RNN model of the following specification

- layer 1 should be an LSTM module with 200 hidden units --> note this should have input_shape = (window_size,len(chars)) where len(chars) = number of unique characters in your cleaned text
- layer 2 should be a linear module, fully connected, with len(chars) hidden units --> where len(chars) = number of unique characters in your cleaned text
- layer 3 should be a softmax activation (since we are solving a *multiclass classification*)
- Use the **categorical_crossentropy** loss

This network can be constructed using just a few lines - as with the RNN network you made in part 1 of this notebook. See e.g., the general Keras documentation (https://keras.io/getting-started/sequential-model-guide/) and the LTSM documentation in particular (https://keras.io/layers/recurrent/) for examples of how to quickly use Keras to build neural network models.

In [15]:
```
### necessary functions from the keras library
from keras.models import Sequential
from keras.layers import Dense, Activation, LSTM
from keras.optimizers import RMSprop
from keras.utils.data_utils import get_file
import keras
import random

# build the required RNN model: a single LSTM hidden layer with softmax
 activation, categorical_crossentropy loss
model = Sequential()
model.add(LSTM(220, input_shape=(window_size,len(chars))))
model.add(Dense(len(chars)))
model.add(Activation('softmax'))

# initialize optimizer
optimizer = keras.optimizers.RMSprop(lr=0.001, rho=0.9, epsilon=1e-08, d
ecay=0.0)

# compile model --> make sure initialized optimizer and callbacks - as d
efined above - are used
model.compile(loss='categorical_crossentropy', optimizer=optimizer)
```

```
Using TensorFlow backend.
```

2.7 Training our RNN model for text generation

With our RNN setup we can now train it! Lets begin by trying it out on a small subset of the larger version. In the next cell we take the first 10,000 input/output pairs from our training database to learn on.

In [16]:
```
# a small subset of our input/output pairs
Xsmall = X[:10000,:,:]
ysmall = y[:10000,:]
```

Now lets fit our model!

In [17]:

```
# train the model
model.fit(Xsmall, ysmall, batch_size=500, epochs=40,verbose = 1)

# save weights
model.save_weights('model_weights/best_RNN_small_textdata_weights.hdf5')
```

```
Epoch 1/40
10000/10000 [==============================] - 5s - loss: 3.1609
Epoch 2/40
10000/10000 [==============================] - 4s - loss: 2.9511
Epoch 3/40
10000/10000 [==============================] - 4s - loss: 2.9285
Epoch 4/40
10000/10000 [==============================] - 4s - loss: 2.9027
Epoch 5/40
10000/10000 [==============================] - 4s - loss: 2.8630
Epoch 6/40
10000/10000 [==============================] - 4s - loss: 2.8145
Epoch 7/40
10000/10000 [==============================] - 4s - loss: 2.7506
Epoch 8/40
10000/10000 [==============================] - 4s - loss: 2.6888
Epoch 9/40
10000/10000 [==============================] - 4s - loss: 2.6223
Epoch 10/40
10000/10000 [==============================] - 4s - loss: 2.5753
Epoch 11/40
10000/10000 [==============================] - 4s - loss: 2.5196
Epoch 12/40
10000/10000 [==============================] - 4s - loss: 2.4748
Epoch 13/40
10000/10000 [==============================] - 4s - loss: 2.4348
Epoch 14/40
10000/10000 [==============================] - 4s - loss: 2.4012
Epoch 15/40
10000/10000 [==============================] - 4s - loss: 2.3677
Epoch 16/40
10000/10000 [==============================] - 4s - loss: 2.3376
Epoch 17/40
10000/10000 [==============================] - 4s - loss: 2.3179
Epoch 18/40
10000/10000 [==============================] - 4s - loss: 2.2877
Epoch 19/40
10000/10000 [==============================] - 4s - loss: 2.2782
Epoch 20/40
10000/10000 [==============================] - 4s - loss: 2.2476
Epoch 21/40
10000/10000 [==============================] - 4s - loss: 2.2281
Epoch 22/40
10000/10000 [==============================] - 4s - loss: 2.2123
Epoch 23/40
10000/10000 [==============================] - 4s - loss: 2.1902
Epoch 24/40
10000/10000 [==============================] - 4s - loss: 2.1741
Epoch 25/40
10000/10000 [==============================] - 4s - loss: 2.1592
Epoch 26/40
10000/10000 [==============================] - 4s - loss: 2.1437
Epoch 27/40
10000/10000 [==============================] - 4s - loss: 2.1237
Epoch 28/40
10000/10000 [==============================] - 4s - loss: 2.1068
Epoch 29/40
```

```
10000/10000 [==============================] - 4s - loss: 2.0881
Epoch 30/40
10000/10000 [==============================] - 4s - loss: 2.0765
Epoch 31/40
10000/10000 [==============================] - 4s - loss: 2.0588
Epoch 32/40
10000/10000 [==============================] - 4s - loss: 2.0394
Epoch 33/40
10000/10000 [==============================] - 4s - loss: 2.0239
Epoch 34/40
10000/10000 [==============================] - 4s - loss: 2.0067
Epoch 35/40
10000/10000 [==============================] - 4s - loss: 1.9939
Epoch 36/40
10000/10000 [==============================] - 4s - loss: 1.9763
Epoch 37/40
10000/10000 [==============================] - 4s - loss: 1.9628
Epoch 38/40
10000/10000 [==============================] - 4s - loss: 1.9437
Epoch 39/40
10000/10000 [==============================] - 4s - loss: 1.9314
Epoch 40/40
10000/10000 [==============================] - 4s - loss: 1.9081
```

How do we make a given number of predictions (characters) based on this fitted model?

First we predict the next character after following any chunk of characters in the text of length equal to our chosen window size. Then we remove the first character in our input sequence and tack our prediction onto the end. This gives us a slightly changed sequence of inputs that still has length equal to the size of our window. We then feed in this updated input sequence into the model to predict the another character. Together then we have two predicted characters following our original input sequence. Repeating this process N times gives us N predicted characters.

In the next Python cell we provide you with a completed function that does just this - it makes predictions when given a) a trained RNN model, b) a subset of (window_size) characters from the text, and c) a number of characters to predict (to follow our input subset).

In [18]:
```
# function that uses trained model to predict a desired number of future
 characters
def predict_next_chars(model,input_chars,num_to_predict):
    # create output
    predicted_chars = ''
    for i in range(num_to_predict):
        # convert this round's predicted characters to numerical input

        x_test = np.zeros((1, window_size, len(chars)))
        for t, char in enumerate(input_chars):
            x_test[0, t, chars_to_indices[char]] = 1.

        # make this round's prediction
        test_predict = model.predict(x_test,verbose = 0)[0]

        # translate numerical prediction back to characters
        r = np.argmax(test_predict)                           # predict
 class of each test input
        d = indices_to_chars[r]

        # update predicted_chars and input
        predicted_chars+=d
        input_chars+=d
        input_chars = input_chars[1:]
    return predicted_chars
```

With your trained model try a few subsets of the complete text as input - note the length of each must be exactly equal to the window size. For each subset us the function above to predict the next 100 characters that follow each input.

In [19]:
```python
# choose an input sequence and use the prediction function in the previo
us Python cell to predict 100 characters following it
# get an appropriately sized chunk of characters from the text
start_inds = [2048]

# load in weights
model.load_weights('model_weights/best_RNN_small_textdata_weights.hdf5')
for s in start_inds:
    start_index = s
    input_chars = text[start_index: start_index + window_size]

    # use the prediction function
    predict_input = predict_next_chars(model,input_chars,num_to_predict
= 100)

    # print out input characters
    print('------------------')
    input_line = 'input chars = ' + '\n' +  input_chars + '"' + '\n'
    print(input_line)

    # print out predicted characters
    line = 'predicted chars = ' + '\n' +  predict_input + '"' + '\n'
    print(line)
```

```
------------------
input chars = 
 at trincomalee, and finally of the mission which he had accomplished s
o delicately and successfully"

predicted chars = 
 to the sour the sound the sout the sout the sout the sout the sound th
e sound he was the she sound "
```

This looks ok, but not great. Now lets try the same experiment with a larger chunk of the data - with the first 100,000 input/output pairs.

Tuning RNNs for a typical character dataset like the one we will use here is a computationally intensive endeavour and thus timely on a typical CPU. Using a reasonably sized cloud-based GPU can speed up training by a factor of 10. Also because of the long training time it is highly recommended that you carefully write the output of each step of your process to file. This is so that all of your results are saved even if you close the web browser you're working out of, as the processes will continue processing in the background but variables/output in the notebook system will not update when you open it again.

In the next cell we show you how to create a text file in Python and record data to it. This sort of setup can be used to record your final predictions.

In [20]:
```
### A simple way to write output to file
f = open('my_test_output.txt', 'w')              # create an output file
 to write too
f.write('this is only a test ' + '\n')          # print some output tex
t
x = 2
f.write('the value of x is ' + str(x) + '\n')    # record a variable val
ue
f.close()

# print out the contents of my_test_output.txt
f = open('my_test_output.txt', 'r')              # create an output file
 to write too
f.read()
```

Out[20]:
```
'this is only a test \nthe value of x is 2\n'
```

With this recording devices we can now more safely perform experiments on larger portions of the text. In the next cell we will use the first 100,000 input/output pairs to train our RNN model.

First we fit our model to the dataset, then generate text using the trained model in precisely the same generation method applied before on the small dataset.

Note: your generated words should be - by and large - more realistic than with the small dataset, but you won't be able to generate perfect English sentences even with this amount of data. A rule of thumb: your model is working well if you generate sentences that largely contain real English words.

In [23]:
```
# a small subset of our input/output pairs
Xlarge = X[:100000,:,:]
ylarge = y[:100000,:]

# fit to our larger dataset
model.fit(Xlarge, ylarge, batch_size=500, epochs=300,verbose = 1)

# save weights
model.save_weights('model_weights/best_RNN_large_textdata_weights.hdf5')
```

```
Epoch 1/300
100000/100000 [==============================] - 46s - loss: 1.9957
Epoch 2/300
100000/100000 [==============================] - 46s - loss: 1.9292
Epoch 3/300
100000/100000 [==============================] - 46s - loss: 1.8735
Epoch 4/300
100000/100000 [==============================] - 46s - loss: 1.8252
Epoch 5/300
100000/100000 [==============================] - 46s - loss: 1.7809
Epoch 6/300
100000/100000 [==============================] - 46s - loss: 1.7417
Epoch 7/300
100000/100000 [==============================] - 46s - loss: 1.7040
Epoch 8/300
100000/100000 [==============================] - 46s - loss: 1.6695
Epoch 9/300
100000/100000 [==============================] - 46s - loss: 1.6360
Epoch 10/300
100000/100000 [==============================] - 46s - loss: 1.6042
Epoch 11/300
100000/100000 [==============================] - 46s - loss: 1.5736
Epoch 12/300
100000/100000 [==============================] - 46s - loss: 1.5436
Epoch 13/300
100000/100000 [==============================] - 46s - loss: 1.5150
Epoch 14/300
100000/100000 [==============================] - 46s - loss: 1.4869
Epoch 15/300
100000/100000 [==============================] - 46s - loss: 1.4587
Epoch 16/300
100000/100000 [==============================] - 46s - loss: 1.4313
Epoch 17/300
100000/100000 [==============================] - 46s - loss: 1.4114
Epoch 18/300
100000/100000 [==============================] - 46s - loss: 1.3780
Epoch 19/300
100000/100000 [==============================] - 46s - loss: 1.3508
Epoch 20/300
100000/100000 [==============================] - 46s - loss: 1.3252
Epoch 21/300
100000/100000 [==============================] - 46s - loss: 1.2978
Epoch 22/300
100000/100000 [==============================] - 46s - loss: 1.2732
Epoch 23/300
100000/100000 [==============================] - 46s - loss: 1.2447
Epoch 24/300
100000/100000 [==============================] - 46s - loss: 1.2189
Epoch 25/300
100000/100000 [==============================] - 46s - loss: 1.1917
Epoch 26/300
100000/100000 [==============================] - 46s - loss: 1.1659
Epoch 27/300
100000/100000 [==============================] - 46s - loss: 1.1398
Epoch 28/300
100000/100000 [==============================] - 46s - loss: 1.1131
Epoch 29/300
```

```
100000/100000 [==============================] - 46s - loss: 1.0885
Epoch 30/300
100000/100000 [==============================] - 46s - loss: 1.0618
Epoch 31/300
100000/100000 [==============================] - 46s - loss: 1.0368
Epoch 32/300
100000/100000 [==============================] - 46s - loss: 1.0110
Epoch 33/300
100000/100000 [==============================] - 46s - loss: 0.9871
Epoch 34/300
100000/100000 [==============================] - 46s - loss: 0.9646
Epoch 35/300
100000/100000 [==============================] - 46s - loss: 0.9410
Epoch 36/300
100000/100000 [==============================] - 46s - loss: 0.9172
Epoch 37/300
100000/100000 [==============================] - 46s - loss: 0.8974
Epoch 38/300
100000/100000 [==============================] - 46s - loss: 0.8769
Epoch 39/300
100000/100000 [==============================] - 46s - loss: 0.8576
Epoch 40/300
100000/100000 [==============================] - 46s - loss: 0.8375
Epoch 41/300
100000/100000 [==============================] - 46s - loss: 0.8165
Epoch 42/300
100000/100000 [==============================] - 46s - loss: 0.7995
Epoch 43/300
100000/100000 [==============================] - 46s - loss: 0.7833
Epoch 44/300
100000/100000 [==============================] - 46s - loss: 0.7648
Epoch 45/300
100000/100000 [==============================] - 46s - loss: 0.7478
Epoch 46/300
100000/100000 [==============================] - 46s - loss: 0.7320
Epoch 47/300
100000/100000 [==============================] - 46s - loss: 0.7188
Epoch 48/300
100000/100000 [==============================] - 46s - loss: 0.7045
Epoch 49/300
100000/100000 [==============================] - 46s - loss: 0.6905
Epoch 50/300
100000/100000 [==============================] - 46s - loss: 0.6788
Epoch 51/300
100000/100000 [==============================] - 46s - loss: 0.6631
Epoch 52/300
100000/100000 [==============================] - 46s - loss: 0.6508
Epoch 53/300
100000/100000 [==============================] - 46s - loss: 0.6381
Epoch 54/300
100000/100000 [==============================] - 46s - loss: 0.6285
Epoch 55/300
100000/100000 [==============================] - 46s - loss: 0.6167
Epoch 56/300
100000/100000 [==============================] - 46s - loss: 0.6065
Epoch 57/300
100000/100000 [==============================] - 46s - loss: 0.5977
```

```
Epoch 58/300
100000/100000 [==============================] - 46s - loss: 0.5866
Epoch 59/300
100000/100000 [==============================] - 46s - loss: 0.5771
Epoch 60/300
100000/100000 [==============================] - 46s - loss: 0.5681
Epoch 61/300
100000/100000 [==============================] - 46s - loss: 0.5566
Epoch 62/300
100000/100000 [==============================] - 46s - loss: 0.5505
Epoch 63/300
100000/100000 [==============================] - 46s - loss: 0.5420
Epoch 64/300
100000/100000 [==============================] - 46s - loss: 0.5342
Epoch 65/300
100000/100000 [==============================] - 46s - loss: 0.5256
Epoch 66/300
100000/100000 [==============================] - 46s - loss: 0.5190
Epoch 67/300
100000/100000 [==============================] - 46s - loss: 0.5092
Epoch 68/300
100000/100000 [==============================] - 46s - loss: 0.5043
Epoch 69/300
100000/100000 [==============================] - 46s - loss: 0.4971
Epoch 70/300
100000/100000 [==============================] - 46s - loss: 0.4917
Epoch 71/300
100000/100000 [==============================] - 46s - loss: 0.4829
Epoch 72/300
100000/100000 [==============================] - 46s - loss: 0.4748
Epoch 73/300
100000/100000 [==============================] - 46s - loss: 0.4725
Epoch 74/300
100000/100000 [==============================] - 46s - loss: 0.4645
Epoch 75/300
100000/100000 [==============================] - 46s - loss: 0.4597
Epoch 76/300
100000/100000 [==============================] - 46s - loss: 0.4529
Epoch 77/300
100000/100000 [==============================] - 46s - loss: 0.4464
Epoch 78/300
100000/100000 [==============================] - 46s - loss: 0.4441
Epoch 79/300
100000/100000 [==============================] - 46s - loss: 0.4374
Epoch 80/300
100000/100000 [==============================] - 46s - loss: 0.4326
Epoch 81/300
100000/100000 [==============================] - 46s - loss: 0.4247
Epoch 82/300
100000/100000 [==============================] - 46s - loss: 0.4228
Epoch 83/300
100000/100000 [==============================] - 46s - loss: 0.4171
Epoch 84/300
100000/100000 [==============================] - 46s - loss: 0.4125
Epoch 85/300
100000/100000 [==============================] - 46s - loss: 0.4094
Epoch 86/300
```

```
100000/100000 [==============================] - 46s - loss: 0.4026
Epoch 87/300
100000/100000 [==============================] - 46s - loss: 0.3985
Epoch 88/300
100000/100000 [==============================] - 46s - loss: 0.3945
Epoch 89/300
100000/100000 [==============================] - 46s - loss: 0.3903
Epoch 90/300
100000/100000 [==============================] - 46s - loss: 0.3843
Epoch 91/300
100000/100000 [==============================] - 46s - loss: 0.3799
Epoch 92/300
100000/100000 [==============================] - 46s - loss: 0.3791
Epoch 93/300
100000/100000 [==============================] - 46s - loss: 0.3744
Epoch 94/300
100000/100000 [==============================] - 46s - loss: 0.3715
Epoch 95/300
100000/100000 [==============================] - 46s - loss: 0.3651
Epoch 96/300
100000/100000 [==============================] - 46s - loss: 0.3621
Epoch 97/300
100000/100000 [==============================] - 46s - loss: 0.3636
Epoch 98/300
100000/100000 [==============================] - 46s - loss: 0.3580
Epoch 99/300
100000/100000 [==============================] - 46s - loss: 0.3532
Epoch 100/300
100000/100000 [==============================] - 46s - loss: 0.3498
Epoch 101/300
100000/100000 [==============================] - 46s - loss: 0.3436
Epoch 102/300
100000/100000 [==============================] - 46s - loss: 0.3407
Epoch 103/300
100000/100000 [==============================] - 46s - loss: 0.3375
Epoch 104/300
100000/100000 [==============================] - 46s - loss: 0.3346
Epoch 105/300
100000/100000 [==============================] - 46s - loss: 0.3309
Epoch 106/300
100000/100000 [==============================] - 46s - loss: 0.3303
Epoch 107/300
100000/100000 [==============================] - 46s - loss: 0.3262
Epoch 108/300
100000/100000 [==============================] - 46s - loss: 0.3239
Epoch 109/300
100000/100000 [==============================] - 46s - loss: 0.3207
Epoch 110/300
100000/100000 [==============================] - 46s - loss: 0.3171
Epoch 111/300
100000/100000 [==============================] - 46s - loss: 0.3142
Epoch 112/300
100000/100000 [==============================] - 46s - loss: 0.3117
Epoch 113/300
100000/100000 [==============================] - 46s - loss: 0.3059
Epoch 114/300
100000/100000 [==============================] - 46s - loss: 0.3086
```

```
Epoch 115/300
100000/100000 [==============================] - 46s - loss: 0.3070
Epoch 116/300
100000/100000 [==============================] - 46s - loss: 0.3038
Epoch 117/300
100000/100000 [==============================] - 46s - loss: 0.2981
Epoch 118/300
100000/100000 [==============================] - 46s - loss: 0.2955
Epoch 119/300
100000/100000 [==============================] - 46s - loss: 0.2919
Epoch 120/300
100000/100000 [==============================] - 46s - loss: 0.2915
Epoch 121/300
100000/100000 [==============================] - 46s - loss: 0.2893
Epoch 122/300
100000/100000 [==============================] - 46s - loss: 0.2854
Epoch 123/300
100000/100000 [==============================] - 46s - loss: 0.2837
Epoch 124/300
100000/100000 [==============================] - 46s - loss: 0.2800
Epoch 125/300
100000/100000 [==============================] - 46s - loss: 0.2772
Epoch 126/300
100000/100000 [==============================] - 46s - loss: 0.2761
Epoch 127/300
100000/100000 [==============================] - 46s - loss: 0.2769
Epoch 128/300
100000/100000 [==============================] - 46s - loss: 0.2719
Epoch 129/300
100000/100000 [==============================] - 46s - loss: 0.2692
Epoch 130/300
100000/100000 [==============================] - 46s - loss: 0.2664
Epoch 131/300
100000/100000 [==============================] - 46s - loss: 0.2681
Epoch 132/300
100000/100000 [==============================] - 46s - loss: 0.2609
Epoch 133/300
100000/100000 [==============================] - 46s - loss: 0.2644
Epoch 134/300
100000/100000 [==============================] - 46s - loss: 0.2603
Epoch 135/300
100000/100000 [==============================] - 46s - loss: 0.2551
Epoch 136/300
100000/100000 [==============================] - 46s - loss: 0.2570
Epoch 137/300
100000/100000 [==============================] - 46s - loss: 0.2499
Epoch 138/300
100000/100000 [==============================] - 46s - loss: 0.2493
Epoch 139/300
100000/100000 [==============================] - 46s - loss: 0.2514
Epoch 140/300
100000/100000 [==============================] - 46s - loss: 0.2466
Epoch 141/300
100000/100000 [==============================] - 46s - loss: 0.2410
Epoch 142/300
100000/100000 [==============================] - 46s - loss: 0.2443
Epoch 143/300
```

```
100000/100000 [==============================] - 46s - loss: 0.2428
Epoch 144/300
100000/100000 [==============================] - 46s - loss: 0.2384
Epoch 145/300
100000/100000 [==============================] - 46s - loss: 0.2367
Epoch 146/300
100000/100000 [==============================] - 46s - loss: 0.2351
Epoch 147/300
100000/100000 [==============================] - 46s - loss: 0.2363
Epoch 148/300
100000/100000 [==============================] - 46s - loss: 0.2347
Epoch 149/300
100000/100000 [==============================] - 46s - loss: 0.2326
Epoch 150/300
100000/100000 [==============================] - 46s - loss: 0.2306
Epoch 151/300
100000/100000 [==============================] - 46s - loss: 0.2275
Epoch 152/300
100000/100000 [==============================] - 46s - loss: 0.2285
Epoch 153/300
100000/100000 [==============================] - 46s - loss: 0.2265
Epoch 154/300
100000/100000 [==============================] - 46s - loss: 0.2220
Epoch 155/300
100000/100000 [==============================] - 46s - loss: 0.2214
Epoch 156/300
100000/100000 [==============================] - 46s - loss: 0.2196
Epoch 157/300
100000/100000 [==============================] - 46s - loss: 0.2186
Epoch 158/300
100000/100000 [==============================] - 46s - loss: 0.2177
Epoch 159/300
100000/100000 [==============================] - 46s - loss: 0.2167
Epoch 160/300
100000/100000 [==============================] - 46s - loss: 0.2171
Epoch 161/300
100000/100000 [==============================] - 46s - loss: 0.2129
Epoch 162/300
100000/100000 [==============================] - 46s - loss: 0.2158
Epoch 163/300
100000/100000 [==============================] - 46s - loss: 0.2079
Epoch 164/300
100000/100000 [==============================] - 46s - loss: 0.2084
Epoch 165/300
100000/100000 [==============================] - 46s - loss: 0.2071
Epoch 166/300
100000/100000 [==============================] - 46s - loss: 0.2112
Epoch 167/300
100000/100000 [==============================] - 46s - loss: 0.2081
Epoch 168/300
100000/100000 [==============================] - 46s - loss: 0.2045
Epoch 169/300
100000/100000 [==============================] - 46s - loss: 0.2023
Epoch 170/300
100000/100000 [==============================] - 46s - loss: 0.2014
Epoch 171/300
100000/100000 [==============================] - 46s - loss: 0.2011
```

```
Epoch 172/300
100000/100000 [==============================] - 46s - loss: 0.2011
Epoch 173/300
100000/100000 [==============================] - 46s - loss: 0.1995
Epoch 174/300
100000/100000 [==============================] - 46s - loss: 0.2000
Epoch 175/300
100000/100000 [==============================] - 46s - loss: 0.1947
Epoch 176/300
100000/100000 [==============================] - 46s - loss: 0.1955
Epoch 177/300
100000/100000 [==============================] - 46s - loss: 0.1949
Epoch 178/300
100000/100000 [==============================] - 46s - loss: 0.1945
Epoch 179/300
100000/100000 [==============================] - 46s - loss: 0.1931
Epoch 180/300
100000/100000 [==============================] - 46s - loss: 0.1922
Epoch 181/300
100000/100000 [==============================] - 46s - loss: 0.1911
Epoch 182/300
100000/100000 [==============================] - 46s - loss: 0.1891
Epoch 183/300
100000/100000 [==============================] - 46s - loss: 0.1893
Epoch 184/300
100000/100000 [==============================] - 46s - loss: 0.1860
Epoch 185/300
100000/100000 [==============================] - 46s - loss: 0.1855
Epoch 186/300
100000/100000 [==============================] - 46s - loss: 0.1869
Epoch 187/300
100000/100000 [==============================] - 46s - loss: 0.1825
Epoch 188/300
100000/100000 [==============================] - 46s - loss: 0.1850
Epoch 189/300
100000/100000 [==============================] - 46s - loss: 0.1870
Epoch 190/300
100000/100000 [==============================] - 46s - loss: 0.1814
Epoch 191/300
100000/100000 [==============================] - 46s - loss: 0.1804
Epoch 192/300
100000/100000 [==============================] - 46s - loss: 0.1785
Epoch 193/300
100000/100000 [==============================] - 46s - loss: 0.1756
Epoch 194/300
100000/100000 [==============================] - 46s - loss: 0.1759
Epoch 195/300
100000/100000 [==============================] - 46s - loss: 0.1777
Epoch 196/300
100000/100000 [==============================] - 46s - loss: 0.1774
Epoch 197/300
100000/100000 [==============================] - 46s - loss: 0.1753
Epoch 198/300
100000/100000 [==============================] - 46s - loss: 0.1758
Epoch 199/300
100000/100000 [==============================] - 46s - loss: 0.1731
Epoch 200/300
```

```
100000/100000 [==============================] - 46s - loss: 0.1740
Epoch 201/300
100000/100000 [==============================] - 46s - loss: 0.1709
Epoch 202/300
100000/100000 [==============================] - 46s - loss: 0.1685
Epoch 203/300
100000/100000 [==============================] - 46s - loss: 0.1694
Epoch 204/300
100000/100000 [==============================] - 46s - loss: 0.1682
Epoch 205/300
100000/100000 [==============================] - 46s - loss: 0.1688
Epoch 206/300
100000/100000 [==============================] - 46s - loss: 0.1671
Epoch 207/300
100000/100000 [==============================] - 46s - loss: 0.1663
Epoch 208/300
100000/100000 [==============================] - 46s - loss: 0.1654
Epoch 209/300
100000/100000 [==============================] - 46s - loss: 0.1660
Epoch 210/300
100000/100000 [==============================] - 46s - loss: 0.1641
Epoch 211/300
100000/100000 [==============================] - 46s - loss: 0.1631
Epoch 212/300
100000/100000 [==============================] - 46s - loss: 0.1640
Epoch 213/300
100000/100000 [==============================] - 46s - loss: 0.1594
Epoch 214/300
100000/100000 [==============================] - 46s - loss: 0.1587
Epoch 215/300
100000/100000 [==============================] - 46s - loss: 0.1590
Epoch 216/300
100000/100000 [==============================] - 46s - loss: 0.1592
Epoch 217/300
100000/100000 [==============================] - 46s - loss: 0.1586
Epoch 218/300
100000/100000 [==============================] - 46s - loss: 0.1604
Epoch 219/300
100000/100000 [==============================] - 46s - loss: 0.1566
Epoch 220/300
100000/100000 [==============================] - 46s - loss: 0.1579
Epoch 221/300
100000/100000 [==============================] - 46s - loss: 0.1562
Epoch 222/300
100000/100000 [==============================] - 46s - loss: 0.1557
Epoch 223/300
100000/100000 [==============================] - 46s - loss: 0.1539
Epoch 224/300
100000/100000 [==============================] - 46s - loss: 0.1554
Epoch 225/300
100000/100000 [==============================] - 46s - loss: 0.1528
Epoch 226/300
100000/100000 [==============================] - 46s - loss: 0.1535
Epoch 227/300
100000/100000 [==============================] - 46s - loss: 0.1559
Epoch 228/300
100000/100000 [==============================] - 46s - loss: 0.1534
```

```
Epoch 229/300
100000/100000 [==============================] - 46s - loss: 0.1523
Epoch 230/300
100000/100000 [==============================] - 46s - loss: 0.1493
Epoch 231/300
100000/100000 [==============================] - 46s - loss: 0.1496
Epoch 232/300
100000/100000 [==============================] - 46s - loss: 0.1499
Epoch 233/300
100000/100000 [==============================] - 46s - loss: 0.1491
Epoch 234/300
100000/100000 [==============================] - 46s - loss: 0.1472
Epoch 235/300
100000/100000 [==============================] - 46s - loss: 0.1448
Epoch 236/300
100000/100000 [==============================] - 46s - loss: 0.1472
Epoch 237/300
100000/100000 [==============================] - 46s - loss: 0.1442
Epoch 238/300
100000/100000 [==============================] - 46s - loss: 0.1453
Epoch 239/300
100000/100000 [==============================] - 46s - loss: 0.1452
Epoch 240/300
100000/100000 [==============================] - 46s - loss: 0.1450
Epoch 241/300
100000/100000 [==============================] - 46s - loss: 0.1429
Epoch 242/300
100000/100000 [==============================] - 46s - loss: 0.1443
Epoch 243/300
100000/100000 [==============================] - 46s - loss: 0.1378
Epoch 244/300
100000/100000 [==============================] - 46s - loss: 0.1437
Epoch 245/300
100000/100000 [==============================] - 46s - loss: 0.1415
Epoch 246/300
100000/100000 [==============================] - 46s - loss: 0.1398
Epoch 247/300
100000/100000 [==============================] - 46s - loss: 0.1401
Epoch 248/300
100000/100000 [==============================] - 46s - loss: 0.1401
Epoch 249/300
100000/100000 [==============================] - 46s - loss: 0.1413
Epoch 250/300
100000/100000 [==============================] - 46s - loss: 0.1404
Epoch 251/300
100000/100000 [==============================] - 46s - loss: 0.1370
Epoch 252/300
100000/100000 [==============================] - 46s - loss: 0.1398
Epoch 253/300
100000/100000 [==============================] - 46s - loss: 0.1387
Epoch 254/300
100000/100000 [==============================] - 46s - loss: 0.1373
Epoch 255/300
100000/100000 [==============================] - 46s - loss: 0.1390
Epoch 256/300
100000/100000 [==============================] - 46s - loss: 0.1367
Epoch 257/300
```

```
100000/100000 [==============================] - 46s - loss: 0.1359
Epoch 258/300
100000/100000 [==============================] - 46s - loss: 0.1361
Epoch 259/300
100000/100000 [==============================] - 46s - loss: 0.1323
Epoch 260/300
100000/100000 [==============================] - 46s - loss: 0.1312
Epoch 261/300
100000/100000 [==============================] - 46s - loss: 0.1347
Epoch 262/300
100000/100000 [==============================] - 46s - loss: 0.1335
Epoch 263/300
100000/100000 [==============================] - 46s - loss: 0.1339
Epoch 264/300
100000/100000 [==============================] - 46s - loss: 0.1335
Epoch 265/300
100000/100000 [==============================] - 46s - loss: 0.1323
Epoch 266/300
100000/100000 [==============================] - 46s - loss: 0.1327
Epoch 267/300
100000/100000 [==============================] - 46s - loss: 0.1296
Epoch 268/300
100000/100000 [==============================] - 46s - loss: 0.1292
Epoch 269/300
100000/100000 [==============================] - 46s - loss: 0.1289
Epoch 270/300
100000/100000 [==============================] - 46s - loss: 0.1283
Epoch 271/300
100000/100000 [==============================] - 46s - loss: 0.1313
Epoch 272/300
100000/100000 [==============================] - 46s - loss: 0.1301
Epoch 273/300
100000/100000 [==============================] - 46s - loss: 0.1287
Epoch 274/300
100000/100000 [==============================] - 46s - loss: 0.1260
Epoch 275/300
100000/100000 [==============================] - 46s - loss: 0.1298
Epoch 276/300
100000/100000 [==============================] - 46s - loss: 0.1280
Epoch 277/300
100000/100000 [==============================] - 46s - loss: 0.1291
Epoch 278/300
100000/100000 [==============================] - 46s - loss: 0.1258
Epoch 279/300
100000/100000 [==============================] - 46s - loss: 0.1244
Epoch 280/300
100000/100000 [==============================] - 46s - loss: 0.1254
Epoch 281/300
100000/100000 [==============================] - 46s - loss: 0.1256
Epoch 282/300
100000/100000 [==============================] - 46s - loss: 0.1263
Epoch 283/300
100000/100000 [==============================] - 46s - loss: 0.1219
Epoch 284/300
100000/100000 [==============================] - 46s - loss: 0.1259
Epoch 285/300
100000/100000 [==============================] - 46s - loss: 0.1262
```

```
Epoch 286/300
100000/100000 [==============================] - 46s - loss: 0.1234
Epoch 287/300
100000/100000 [==============================] - 46s - loss: 0.1240
Epoch 288/300
100000/100000 [==============================] - 46s - loss: 0.1212
Epoch 289/300
100000/100000 [==============================] - 46s - loss: 0.1232
Epoch 290/300
100000/100000 [==============================] - 46s - loss: 0.1228
Epoch 291/300
100000/100000 [==============================] - 46s - loss: 0.1198
Epoch 292/300
100000/100000 [==============================] - 46s - loss: 0.1195
Epoch 293/300
100000/100000 [==============================] - 46s - loss: 0.1225
Epoch 294/300
100000/100000 [==============================] - 46s - loss: 0.1199
Epoch 295/300
100000/100000 [==============================] - 46s - loss: 0.1229
Epoch 296/300
100000/100000 [==============================] - 46s - loss: 0.1184
Epoch 297/300
100000/100000 [==============================] - 46s - loss: 0.1226
Epoch 298/300
100000/100000 [==============================] - 46s - loss: 0.1210
Epoch 299/300
100000/100000 [==============================] - 46s - loss: 0.1188
Epoch 300/300
100000/100000 [==============================] - 46s - loss: 0.1192
```

In [26]:

```
# choose an input sequence and use the prediction function in the previo
us Python cell to predict 100 characters following it
# get an appropriately sized chunk of characters from the text
start_inds = [1000, 2000, 3000]

# save output
f = open('text_gen_output/RNN_large_textdata_output.txt', 'w')  # create
 an output file to write too

# load weights
model.load_weights('model_weights/best_RNN_large_textdata_weights.hdf5')
for s in start_inds:
    start_index = s
    input_chars = text[start_index: start_index + window_size]

    # use the prediction function
    predict_input = predict_next_chars(model,input_chars,num_to_predict
= 100)

    # print out input characters
    line = '-------------------' + '\n'
    print(line)
    f.write(line)

    input_line = 'input chars = ' + '\n' +  input_chars + '"' + '\n'
    print(input_line)
    f.write(input_line)

    # print out predicted characters
    predict_line = 'predicted chars = ' + '\n' +  predict_input + '"' +
'\n'
    print(predict_line)
    f.write(predict_line)
f.close()
```

```
--------------------

input chars = 
ler, of dubious and questionable memory. i had seen little of holmes la
tely. my marriage had drifted"

predicted chars = 
 ither well ppening here of heards and frout opered in our coine of hai
n onficed the fathing and off"

--------------------

input chars = 
of the singular tragedy of the atkinson brothers at trincomalee, and fi
nally of the mission which he"

predicted chars = 
 had a gold more the able, for you and we knoor wouss. however, i shall
to breat gount, took they no"

--------------------

input chars = 
 his hands clasped behind him. to me, who knew his every mood and habi
t, his attitude and manner tol"

predicted chars = 
d at hurr. then was clamped his tigning of whick hards the fore of the
 wind. who to-she, but the let"
```

In [ ]: