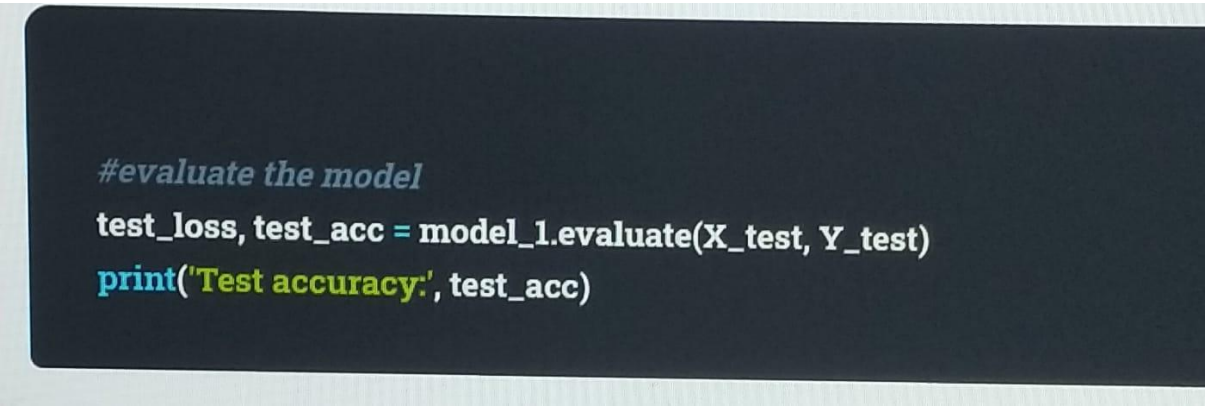
What is observing the metrics?

A metric is any value you can measure over time. It can be blocks used on a filesystem, the number of nodes in a cluster, or a temperature reading. Observe reports Metrics in the form of a time series: a set of values in time order.

Evaluate the Handwritten Digit Recognition Model on Test Data



```
#evaluate the model
test_loss, test_acc = model_1.evaluate(X_test, Y_test)
print('Test accuracy:', test_acc)
```

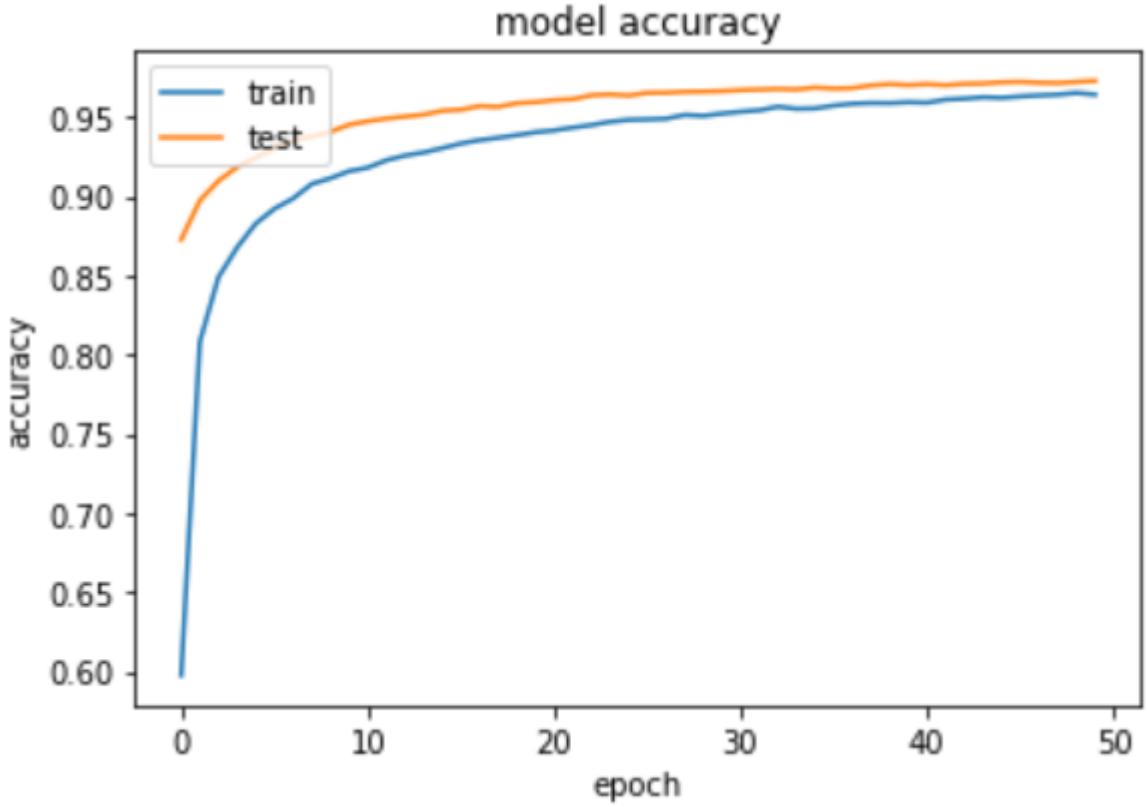
Output:

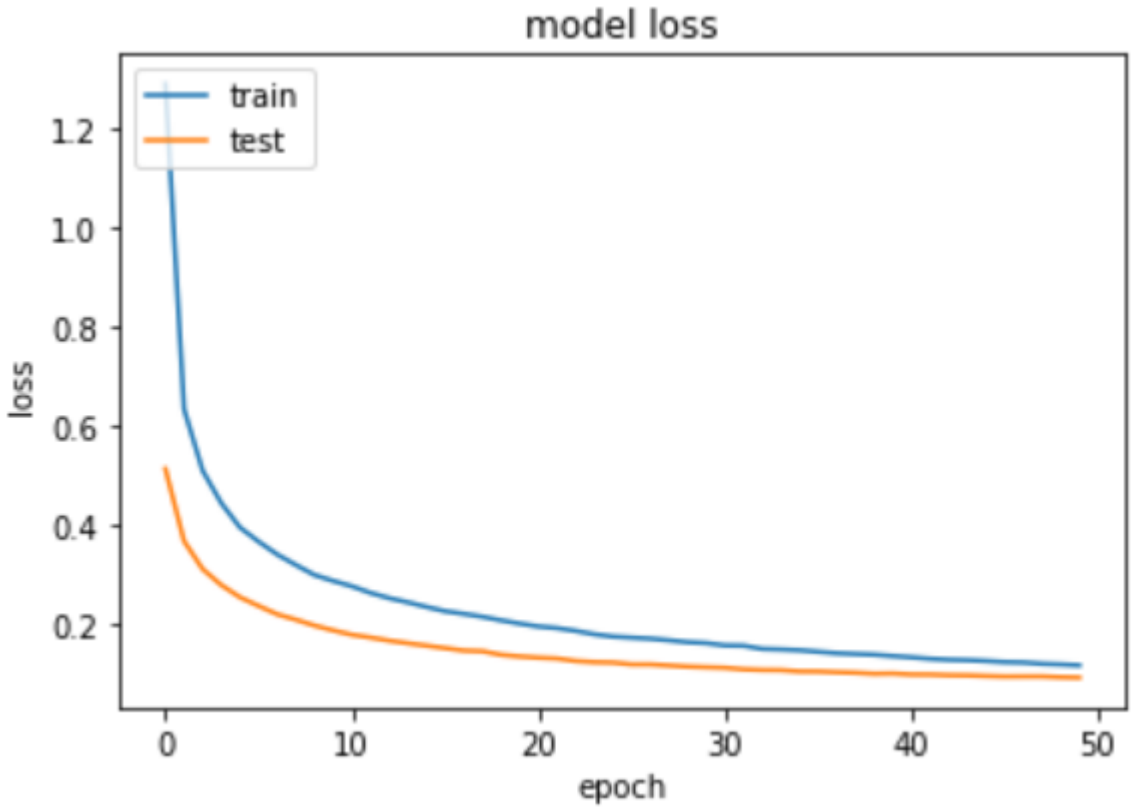
313/313 [==============================] – 0s 1ms/stop – loss: 0.0935 – Accuracy: 0.9737 Test Accuracy: 0.9736999869346619The second model is giving an output of 97%. Further, we can improve the model by adding a dropout to avoid overfitting.

Plot the change in metrics per epoch:

Model will be now trained on the on the training data. For this we will be defining the epochs, batchsize, and validation size

- **epoch:** Number of times that the model will run through the training dataset
- **batch_size:** Number of training instances to be shown to the model before a weight is updated
- **validation_split:** Defines the fraction of data to be used for validation purpose


model accuracy
train
test
accuracy
epoch
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0
10
20
30
40
50


model loss
train
test
loss
epoch
1.2
1.0
0.8
0.6
0.4
0.2
0
10
20
30
40
50

```
import matplotlib.pyplot as plt
%matplotlib inline
# list all data in training
print(training.history.keys())
# summarize training for accuracy
plt.plot(training.history['accuracy'])
plt.plot(training.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
# summarize traning for loss
plt.plot(training.history['loss'])
plt.plot(training.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```