# Image CLASSIFICATION USING CNN

In [33]:

```
from keras.preprocessing.image import ImageDataGenerator, load_img
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D
from keras.layers import Activation, Dropout, Flatten, Dense
from keras import backend as K


from keras.models import load_model
from keras.preprocessing import image
import numpy as np
from os import listdir
from os.path import isfile, join
```

## Set hyper parameters

In [1]:

```
#ReShaping the images to same size
img_width = 150
img_height = 150

train_data_dir = 'D:/data/train'
validation_data_dir = 'D:/data/test'
train_samples = 20
validation_samples = 10
epochs = 100
batch_size = 2
fc_size=32 #size of the output of final FC layer
input_shape = (img_width, img_height, 3)
```

## The model defined with the following layers

- input later
- conv layer 1 with 32 filters of kernel size[5,5],
- pooling layer 1 with pool size[2,2] and stride 2
- conv layer 2 with 64 filters of kernel size[5,5],
- pooling layer 2 with pool size[2,2] and stride 2
- dense layer whose output size is fixed in the hyper parameter: fc_size=32
- drop out layer with droput probability 0.4
- predict the class by doing a softmax on the output of the dropout layers

Training

- For training fefine the loss function and minimize it
- For evaluation calculate the accuracy

In [62]:

```
import tensorflow as tf

model = Sequential()
model.add(tf.keras.layers.BatchNormalization(input_shape=input_shape))

model.add(tf.keras.layers.Conv2D(32, kernel_size=(5,5), strides=2,activation=tf.keras.l
ayers.LeakyReLU(alpha=0.05)))
model.add(tf.keras.layers.MaxPool2D(pool_size=(2,2)))
model.add(tf.keras.layers.Conv2D(64, kernel_size=(5,5), strides=2,activation=tf.keras.l
ayers.LeakyReLU(alpha=0.05)))
model.add(tf.keras.layers.MaxPool2D(pool_size=(2,2)))
model.add(tf.keras.layers.Flatten())
model.add(tf.keras.layers.Dense(units = fc_size, activation=tf.keras.layers.LeakyReLU(a
lpha=0.03)))
model.add(tf.keras.layers.Dropout(0.4))
model.add(tf.keras.layers.Dense(64, activation=tf.keras.layers.LeakyReLU(alpha=0.03)))
#Output layer
model.add(tf.keras.layers.Dense(1, activation='sigmoid'))
```

In [63]:

```
import keras
from keras import optimizers
model.compile(loss='binary_crossentropy',
              optimizer=keras.optimizers.Adam(lr=.0001),
              metrics=['accuracy'])
```

ImageDatagenerator will run through your image data and apply random transformations to each individual image as it is passed to the model so that it never sees the exact same image twice during training.

These transformations are parameters on the generator that can be set when instantiated and can include rotations, shears, flips, and zooms.

The benefit here is that the model will become more robust as it trains on images that are slightly distorted, and it helps to prevent the model from learning noise in your data such as where features are located in the image.

In [52]:

```
train_datagen = ImageDataGenerator(
    rescale=1. / 255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True)

# for testing:
# only rescaling, avoiding having same training and validation data.
test_datagen = ImageDataGenerator(rescale=1. / 255)
```

```
In [53]:
```

```python
train_generator = train_datagen.flow_from_directory(
    train_data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='binary')
```

```
Found 40 images belonging to 2 classes.
```

```
In [54]:
```

```python
print(train_generator.class_indices)
```

```
{'cats': 0, 'dogs': 1}
```

```
In [55]:
```

```python
validation_generator = test_datagen.flow_from_directory(
    validation_data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='binary')
```

```
Found 20 images belonging to 2 classes.
```

In [64]:

```
history = model.fit(
    train_generator,
    steps_per_epoch=train_samples // batch_size,
    epochs=epochs,
    validation_data=validation_generator,
    validation_steps=validation_samples // batch_size)
```

```
Epoch 1/100
10/10 [==============================] - 1s 48ms/step - loss: 0.7209 - acc
uracy: 0.3120 - val_loss: 0.6853 - val_accuracy: 0.6000
Epoch 2/100
10/10 [==============================] - 0s 32ms/step - loss: 0.6493 - acc
uracy: 0.7674 - val_loss: 0.7035 - val_accuracy: 0.4000
Epoch 3/100
10/10 [==============================] - 0s 32ms/step - loss: 0.6993 - acc
uracy: 0.5286 - val_loss: 0.6917 - val_accuracy: 0.6000
Epoch 4/100
10/10 [==============================] - 0s 31ms/step - loss: 0.6938 - acc
uracy: 0.5894 - val_loss: 0.6921 - val_accuracy: 0.6000
Epoch 5/100
10/10 [==============================] - 0s 30ms/step - loss: 0.7196 - acc
uracy: 0.3684 - val_loss: 0.7021 - val_accuracy: 0.4000
Epoch 6/100
10/10 [==============================] - 0s 32ms/step - loss: 0.6933 - acc
uracy: 0.4149 - val_loss: 0.6911 - val_accuracy: 0.5000
Epoch 7/100
10/10 [==============================] - 0s 37ms/step - loss: 0.6506 - acc
uracy: 0.6822 - val_loss: 0.6887 - val_accuracy: 0.5000
Epoch 8/100
10/10 [==============================] - 0s 32ms/step - loss: 0.6930 - acc
uracy: 0.5480 - val_loss: 0.6674 - val_accuracy: 0.7000
Epoch 9/100
10/10 [==============================] - 0s 31ms/step - loss: 0.7327 - acc
uracy: 0.3175 - val_loss: 0.6948 - val_accuracy: 0.6000
Epoch 10/100
10/10 [==============================] - 0s 30ms/step - loss: 0.8077 - acc
uracy: 0.4145 - val_loss: 0.6977 - val_accuracy: 0.5000
Epoch 11/100
10/10 [==============================] - 0s 32ms/step - loss: 0.7056 - acc
uracy: 0.4562 - val_loss: 0.6974 - val_accuracy: 0.6000
Epoch 12/100
10/10 [==============================] - 0s 30ms/step - loss: 0.6906 - acc
uracy: 0.5383 - val_loss: 0.7024 - val_accuracy: 0.5000
Epoch 13/100
10/10 [==============================] - 0s 31ms/step - loss: 0.6208 - acc
uracy: 0.7400 - val_loss: 0.6814 - val_accuracy: 0.5000
Epoch 14/100
10/10 [==============================] - 0s 31ms/step - loss: 0.6823 - acc
uracy: 0.5714 - val_loss: 0.6683 - val_accuracy: 0.6000
Epoch 15/100
10/10 [==============================] - 0s 30ms/step - loss: 0.6625 - acc
uracy: 0.6100 - val_loss: 0.6856 - val_accuracy: 0.4000
Epoch 16/100
10/10 [==============================] - 0s 31ms/step - loss: 0.7456 - acc
uracy: 0.4449 - val_loss: 0.7081 - val_accuracy: 0.3000
Epoch 17/100
10/10 [==============================] - 0s 30ms/step - loss: 0.6667 - acc
uracy: 0.6287 - val_loss: 0.6897 - val_accuracy: 0.5000
Epoch 18/100
10/10 [==============================] - 0s 30ms/step - loss: 0.6453 - acc
uracy: 0.6575 - val_loss: 0.7020 - val_accuracy: 0.5000
Epoch 19/100
10/10 [==============================] - 0s 32ms/step - loss: 0.6144 - acc
uracy: 0.7232 - val_loss: 0.7018 - val_accuracy: 0.4000
Epoch 20/100
10/10 [==============================] - 0s 32ms/step - loss: 0.6470 - acc
uracy: 0.6212 - val_loss: 0.6685 - val_accuracy: 0.6000
Epoch 21/100
```

```
10/10 [==============================] - 0s 31ms/step - loss: 0.5974 - acc
uracy: 0.6494 - val_loss: 0.7260 - val_accuracy: 0.3000
Epoch 22/100
10/10 [==============================] - 0s 31ms/step - loss: 0.6501 - acc
uracy: 0.6874 - val_loss: 0.6849 - val_accuracy: 0.6000
Epoch 23/100
10/10 [==============================] - 0s 30ms/step - loss: 0.6581 - acc
uracy: 0.6034 - val_loss: 0.7032 - val_accuracy: 0.7000
Epoch 24/100
10/10 [==============================] - 0s 31ms/step - loss: 0.6765 - acc
uracy: 0.5388 - val_loss: 0.7216 - val_accuracy: 0.5000
Epoch 25/100
10/10 [==============================] - 0s 31ms/step - loss: 0.7692 - acc
uracy: 0.6369 - val_loss: 0.7044 - val_accuracy: 0.3000
Epoch 26/100
10/10 [==============================] - 0s 32ms/step - loss: 0.6392 - acc
uracy: 0.6471 - val_loss: 0.7135 - val_accuracy: 0.5000
Epoch 27/100
10/10 [==============================] - 0s 31ms/step - loss: 0.6604 - acc
uracy: 0.5266 - val_loss: 0.6913 - val_accuracy: 0.7000
Epoch 28/100
10/10 [==============================] - 0s 33ms/step - loss: 0.5878 - acc
uracy: 0.7344 - val_loss: 0.7407 - val_accuracy: 0.2000
Epoch 29/100
10/10 [==============================] - 0s 31ms/step - loss: 0.6267 - acc
uracy: 0.7321 - val_loss: 0.7142 - val_accuracy: 0.4000
Epoch 30/100
10/10 [==============================] - 0s 31ms/step - loss: 0.4568 - acc
uracy: 0.9646 - val_loss: 0.7090 - val_accuracy: 0.4000
Epoch 31/100
10/10 [==============================] - 0s 31ms/step - loss: 0.6213 - acc
uracy: 0.6174 - val_loss: 0.6808 - val_accuracy: 0.5000
Epoch 32/100
10/10 [==============================] - 0s 30ms/step - loss: 0.6521 - acc
uracy: 0.6708 - val_loss: 0.7380 - val_accuracy: 0.3000
Epoch 33/100
10/10 [==============================] - 0s 30ms/step - loss: 0.7666 - acc
uracy: 0.4672 - val_loss: 0.7081 - val_accuracy: 0.6000
Epoch 34/100
10/10 [==============================] - 0s 30ms/step - loss: 0.6179 - acc
uracy: 0.7725 - val_loss: 0.6951 - val_accuracy: 0.5000
Epoch 35/100
10/10 [==============================] - 0s 31ms/step - loss: 0.6378 - acc
uracy: 0.7252 - val_loss: 0.7451 - val_accuracy: 0.5000
Epoch 36/100
10/10 [==============================] - 0s 32ms/step - loss: 0.6169 - acc
uracy: 0.7527 - val_loss: 0.7324 - val_accuracy: 0.4000
Epoch 37/100
10/10 [==============================] - 0s 31ms/step - loss: 0.5664 - acc
uracy: 0.6832 - val_loss: 0.7702 - val_accuracy: 0.3000
Epoch 38/100
10/10 [==============================] - 0s 31ms/step - loss: 0.5181 - acc
uracy: 0.7609 - val_loss: 0.6929 - val_accuracy: 0.5000
Epoch 39/100
10/10 [==============================] - 0s 30ms/step - loss: 0.7247 - acc
uracy: 0.5651 - val_loss: 0.6795 - val_accuracy: 0.7000
Epoch 40/100
10/10 [==============================] - 0s 31ms/step - loss: 0.7117 - acc
uracy: 0.5331 - val_loss: 0.6834 - val_accuracy: 0.6000
Epoch 41/100
10/10 [==============================] - 0s 31ms/step - loss: 0.7018 - acc
```

```
uracy: 0.4217 - val_loss: 0.7556 - val_accuracy: 0.4000
Epoch 42/100
10/10 [==============================] - 0s 31ms/step - loss: 0.6021 - acc
uracy: 0.7955 - val_loss: 0.6733 - val_accuracy: 0.6000
Epoch 43/100
10/10 [==============================] - 0s 31ms/step - loss: 0.5429 - acc
uracy: 0.8143 - val_loss: 0.8085 - val_accuracy: 0.5000
Epoch 44/100
10/10 [==============================] - 0s 33ms/step - loss: 0.6877 - acc
uracy: 0.7498 - val_loss: 0.7101 - val_accuracy: 0.4000
Epoch 45/100
10/10 [==============================] - 0s 32ms/step - loss: 0.6099 - acc
uracy: 0.7639 - val_loss: 0.8263 - val_accuracy: 0.3000
Epoch 46/100
10/10 [==============================] - 0s 31ms/step - loss: 0.5822 - acc
uracy: 0.7823 - val_loss: 0.7252 - val_accuracy: 0.6000
Epoch 47/100
10/10 [==============================] - 0s 32ms/step - loss: 0.5033 - acc
uracy: 0.7887 - val_loss: 0.6965 - val_accuracy: 0.7000
Epoch 48/100
10/10 [==============================] - 0s 32ms/step - loss: 0.5337 - acc
uracy: 0.7647 - val_loss: 0.7832 - val_accuracy: 0.4000
Epoch 49/100
10/10 [==============================] - 0s 31ms/step - loss: 0.5935 - acc
uracy: 0.6548 - val_loss: 0.7895 - val_accuracy: 0.5000
Epoch 50/100
10/10 [==============================] - 0s 32ms/step - loss: 0.5253 - acc
uracy: 0.7932 - val_loss: 0.6693 - val_accuracy: 0.6000
Epoch 51/100
10/10 [==============================] - 0s 31ms/step - loss: 0.4541 - acc
uracy: 0.8569 - val_loss: 0.7919 - val_accuracy: 0.5000
Epoch 52/100
10/10 [==============================] - 0s 31ms/step - loss: 0.5892 - acc
uracy: 0.6861 - val_loss: 0.6739 - val_accuracy: 0.6000
Epoch 53/100
10/10 [==============================] - 0s 30ms/step - loss: 0.6551 - acc
uracy: 0.6753 - val_loss: 0.6214 - val_accuracy: 0.6000
Epoch 54/100
10/10 [==============================] - 0s 32ms/step - loss: 0.5178 - acc
uracy: 0.8151 - val_loss: 0.6731 - val_accuracy: 0.5000
Epoch 55/100
10/10 [==============================] - 0s 30ms/step - loss: 0.5707 - acc
uracy: 0.7007 - val_loss: 0.6682 - val_accuracy: 0.6000
Epoch 56/100
10/10 [==============================] - 0s 32ms/step - loss: 0.3723 - acc
uracy: 0.9909 - val_loss: 0.8442 - val_accuracy: 0.5000
Epoch 57/100
10/10 [==============================] - 0s 31ms/step - loss: 0.5013 - acc
uracy: 0.7467 - val_loss: 0.8086 - val_accuracy: 0.5000
Epoch 58/100
10/10 [==============================] - 0s 30ms/step - loss: 0.5091 - acc
uracy: 0.8201 - val_loss: 0.7967 - val_accuracy: 0.6000
Epoch 59/100
10/10 [==============================] - 0s 30ms/step - loss: 0.5165 - acc
uracy: 0.7302 - val_loss: 0.6744 - val_accuracy: 0.7000
Epoch 60/100
10/10 [==============================] - 0s 30ms/step - loss: 0.5075 - acc
uracy: 0.7523 - val_loss: 0.6966 - val_accuracy: 0.6000
Epoch 61/100
10/10 [==============================] - 0s 29ms/step - loss: 0.4747 - acc
uracy: 0.8251 - val_loss: 0.5527 - val_accuracy: 0.9000
```

```
Epoch 62/100
10/10 [==============================] - 0s 31ms/step - loss: 0.4569 - acc
uracy: 0.7994 - val_loss: 0.7049 - val_accuracy: 0.8000
Epoch 63/100
10/10 [==============================] - 0s 31ms/step - loss: 0.3881 - acc
uracy: 0.9305 - val_loss: 0.6982 - val_accuracy: 0.6000
Epoch 64/100
10/10 [==============================] - 0s 30ms/step - loss: 0.4576 - acc
uracy: 0.8249 - val_loss: 0.7020 - val_accuracy: 0.6000
Epoch 65/100
10/10 [==============================] - 0s 30ms/step - loss: 0.4155 - acc
uracy: 0.8096 - val_loss: 0.9892 - val_accuracy: 0.4000
Epoch 66/100
10/10 [==============================] - 0s 31ms/step - loss: 0.4072 - acc
uracy: 0.8597 - val_loss: 0.7440 - val_accuracy: 0.5000
Epoch 67/100
10/10 [==============================] - 0s 29ms/step - loss: 0.3571 - acc
uracy: 0.8825 - val_loss: 0.7725 - val_accuracy: 0.5000
Epoch 68/100
10/10 [==============================] - 0s 31ms/step - loss: 0.4399 - acc
uracy: 0.9018 - val_loss: 0.7282 - val_accuracy: 0.7000
Epoch 69/100
10/10 [==============================] - 0s 29ms/step - loss: 0.5528 - acc
uracy: 0.7690 - val_loss: 0.8272 - val_accuracy: 0.5000
Epoch 70/100
10/10 [==============================] - 0s 30ms/step - loss: 0.4618 - acc
uracy: 0.7567 - val_loss: 0.6252 - val_accuracy: 0.7000
Epoch 71/100
10/10 [==============================] - 0s 30ms/step - loss: 0.3362 - acc
uracy: 1.0000 - val_loss: 0.7720 - val_accuracy: 0.7000
Epoch 72/100
10/10 [==============================] - 0s 31ms/step - loss: 0.4229 - acc
uracy: 0.7830 - val_loss: 0.9068 - val_accuracy: 0.5000
Epoch 73/100
10/10 [==============================] - 0s 29ms/step - loss: 0.3016 - acc
uracy: 0.9107 - val_loss: 1.0271 - val_accuracy: 0.5000
Epoch 74/100
10/10 [==============================] - 0s 30ms/step - loss: 0.4503 - acc
uracy: 0.8212 - val_loss: 0.7683 - val_accuracy: 0.3000
Epoch 75/100
10/10 [==============================] - 0s 31ms/step - loss: 0.3467 - acc
uracy: 0.9859 - val_loss: 0.8589 - val_accuracy: 0.5000
Epoch 76/100
10/10 [==============================] - 0s 30ms/step - loss: 0.4299 - acc
uracy: 0.6212 - val_loss: 1.0642 - val_accuracy: 0.4000
Epoch 77/100
10/10 [==============================] - 0s 30ms/step - loss: 0.4095 - acc
uracy: 0.8284 - val_loss: 0.6949 - val_accuracy: 0.6000
Epoch 78/100
10/10 [==============================] - 0s 30ms/step - loss: 0.3644 - acc
uracy: 0.8336 - val_loss: 0.7135 - val_accuracy: 0.6000
Epoch 79/100
10/10 [==============================] - 0s 30ms/step - loss: 0.2955 - acc
uracy: 0.9802 - val_loss: 0.8143 - val_accuracy: 0.7000
Epoch 80/100
10/10 [==============================] - 0s 30ms/step - loss: 0.3470 - acc
uracy: 0.9167 - val_loss: 0.9783 - val_accuracy: 0.3000
Epoch 81/100
10/10 [==============================] - 0s 30ms/step - loss: 0.3472 - acc
uracy: 0.8557 - val_loss: 0.8012 - val_accuracy: 0.4000
Epoch 82/100
```

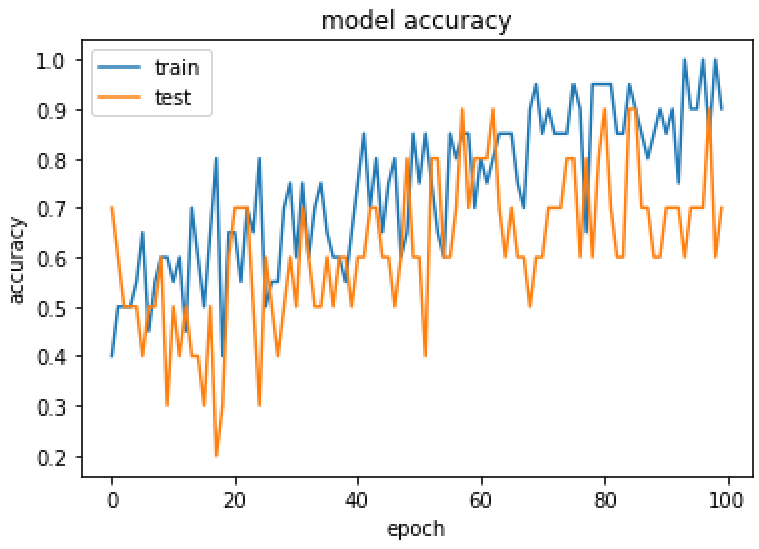
```
10/10 [==============================] - 0s 30ms/step - loss: 0.3996 - acc
uracy: 0.7559 - val_loss: 1.2102 - val_accuracy: 0.4000
Epoch 83/100
10/10 [==============================] - 0s 31ms/step - loss: 0.2525 - acc
uracy: 0.8688 - val_loss: 1.2254 - val_accuracy: 0.5000
Epoch 84/100
10/10 [==============================] - 0s 35ms/step - loss: 0.1942 - acc
uracy: 0.9398 - val_loss: 1.0969 - val_accuracy: 0.5000
Epoch 85/100
10/10 [==============================] - 0s 37ms/step - loss: 0.4225 - acc
uracy: 0.7675 - val_loss: 0.5903 - val_accuracy: 0.7000
Epoch 86/100
10/10 [==============================] - 0s 38ms/step - loss: 0.2444 - acc
uracy: 0.9570 - val_loss: 1.1636 - val_accuracy: 0.5000
Epoch 87/100
10/10 [==============================] - 0s 38ms/step - loss: 0.2108 - acc
uracy: 0.9737 - val_loss: 0.7693 - val_accuracy: 0.7000
Epoch 88/100
10/10 [==============================] - 0s 37ms/step - loss: 0.3327 - acc
uracy: 0.7180 - val_loss: 0.6167 - val_accuracy: 0.8000
Epoch 89/100
10/10 [==============================] - 0s 39ms/step - loss: 0.2180 - acc
uracy: 0.9174 - val_loss: 0.8669 - val_accuracy: 0.7000
Epoch 90/100
10/10 [==============================] - 0s 37ms/step - loss: 0.3229 - acc
uracy: 0.9042 - val_loss: 1.1183 - val_accuracy: 0.5000
Epoch 91/100
10/10 [==============================] - 0s 38ms/step - loss: 0.2858 - acc
uracy: 0.9307 - val_loss: 1.3569 - val_accuracy: 0.5000
Epoch 92/100
10/10 [==============================] - 0s 38ms/step - loss: 0.2018 - acc
uracy: 0.9397 - val_loss: 1.1958 - val_accuracy: 0.5000
Epoch 93/100
10/10 [==============================] - 0s 37ms/step - loss: 0.3673 - acc
uracy: 0.8193 - val_loss: 1.1985 - val_accuracy: 0.5000
Epoch 94/100
10/10 [==============================] - 0s 37ms/step - loss: 0.2378 - acc
uracy: 0.8120 - val_loss: 0.8720 - val_accuracy: 0.7000
Epoch 95/100
10/10 [==============================] - 0s 37ms/step - loss: 0.3522 - acc
uracy: 0.8623 - val_loss: 0.9412 - val_accuracy: 0.7000
Epoch 96/100
10/10 [==============================] - 0s 38ms/step - loss: 0.2254 - acc
uracy: 0.9570 - val_loss: 1.0121 - val_accuracy: 0.5000
Epoch 97/100
10/10 [==============================] - 0s 37ms/step - loss: 0.2530 - acc
uracy: 0.9818 - val_loss: 0.8761 - val_accuracy: 0.5000
Epoch 98/100
10/10 [==============================] - 0s 37ms/step - loss: 0.2570 - acc
uracy: 0.8423 - val_loss: 0.8176 - val_accuracy: 0.8000
Epoch 99/100
10/10 [==============================] - 0s 37ms/step - loss: 0.1767 - acc
uracy: 0.9859 - val_loss: 1.0067 - val_accuracy: 0.6000
Epoch 100/100
10/10 [==============================] - 0s 38ms/step - loss: 0.2740 - acc
uracy: 0.8620 - val_loss: 1.3121 - val_accuracy: 0.5000
```

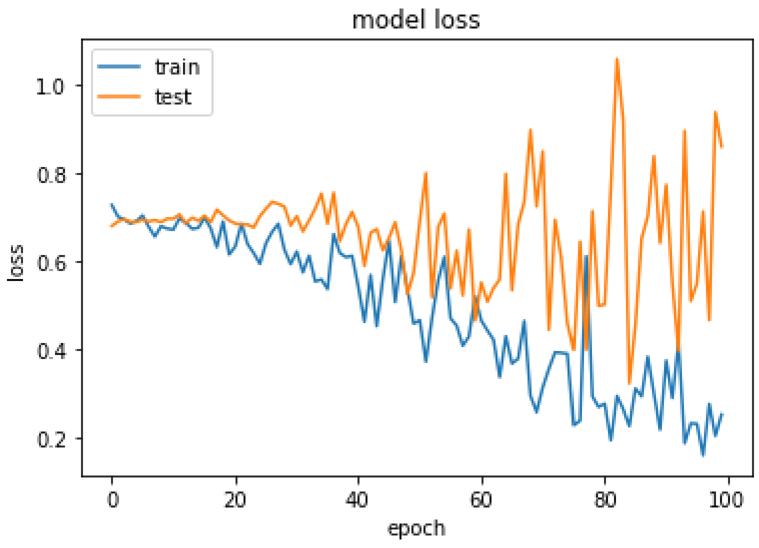
In [57]:

```
import matplotlib.pyplot as plt
%matplotlib inline

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

```
dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```


model accuracy
train
test
accuracy
epoch


model loss
train
test
loss
epoch