In [1]:

```
import os, shutil
```

In [2]:

```
original_dataset_dir = 'C:\\Users\\theoj\\Downloads\\Week_6\\train\\train'
```

In [3]:

```
base_dir = 'C:\\Users\\theoj\\Downloads\\Week_6\\cats_and_dogs_small'
os.mkdir(base_dir)
```

In [4]:

```
train_dir = os.path.join(base_dir, 'train')
os.mkdir(train_dir)
validation_dir = os.path.join(base_dir, 'validation')
os.mkdir(validation_dir)
test_dir = os.path.join(base_dir, 'test')
os.mkdir(test_dir)
```

In [5]:

```
train_cats_dir = os.path.join(train_dir, 'cats')
os.mkdir(train_cats_dir)
```

In [6]:

```
train_dogs_dir = os.path.join(train_dir, 'dogs')
os.mkdir(train_dogs_dir)
```

In [7]:

```
validation_cats_dir = os.path.join(validation_dir, 'cats')
os.mkdir(validation_cats_dir)
```

In [8]:

```
validation_dogs_dir = os.path.join(validation_dir, 'dogs')
os.mkdir(validation_dogs_dir)
```

In [9]:

```
test_cats_dir = os.path.join(test_dir, 'cats')
os.mkdir(test_cats_dir)
```

In [10]:

```
test_dogs_dir = os.path.join(test_dir, 'dogs')
os.mkdir(test_dogs_dir)
```

In [11]:

```
fnames = ['cat.{}.jpg'.format(i) for i in range(1000)]
for fname in fnames:
    src = os.path.join(original_dataset_dir, fname)
    dst = os.path.join(train_cats_dir, fname)
    shutil.copyfile(src, dst)
```

In [12]:

```
fnames = ['cat.{}.jpg'.format(i) for i in range(1000, 1500)]
for fname in fnames:
    src = os.path.join(original_dataset_dir, fname)
    dst = os.path.join(validation_cats_dir, fname)
    shutil.copyfile(src, dst)
```

In [13]:

```
fnames = ['cat.{}.jpg'.format(i) for i in range(1500, 2000)]
for fname in fnames:
    src = os.path.join(original_dataset_dir, fname)
    dst = os.path.join(test_cats_dir, fname)
    shutil.copyfile(src, dst)
```

In [14]:

```
fnames = ['dog.{}.jpg'.format(i) for i in range(1000)]
for fname in fnames:
    src = os.path.join(original_dataset_dir, fname)
    dst = os.path.join(train_dogs_dir, fname)
    shutil.copyfile(src, dst)
```

In [15]:

```
fnames = ['dog.{}.jpg'.format(i) for i in range(1000, 1500)]
for fname in fnames:
    src = os.path.join(original_dataset_dir, fname)
    dst = os.path.join(validation_dogs_dir, fname)
    shutil.copyfile(src, dst)
```

In [16]:

```
fnames = ['dog.{}.jpg'.format(i) for i in range(1500, 2000)]
for fname in fnames:
    src = os.path.join(original_dataset_dir, fname)
    dst = os.path.join(test_dogs_dir, fname)
    shutil.copyfile(src, dst)
```

In [17]:

```
print('total training cat images:', len(os.listdir(train_cats_dir)))
```

```
total training cat images: 1000
```

In [18]:

```
print('total training dog images:', len(os.listdir(train_dogs_dir)))
```

```
total training dog images: 1000
```

In [19]:

```
 print('total validation cat images:', len(os.listdir(validation_cats_dir)))
```

```
total validation cat images: 500
```

In [20]:

```
print('total validation dog images:', len(os.listdir(validation_dogs_dir)))
```

```
total validation dog images: 500
```

In [21]:

```
print('total test cat images:', len(os.listdir(test_cats_dir)))
```

```
total test cat images: 500
```

In [22]:

```
print('total test dog images:', len(os.listdir(test_dogs_dir)))
```

```
total test dog images: 500
```

In [23]:

```
from keras import layers
from keras import models
```

In [24]:

```
model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu',
input_shape=(150, 150, 3)))
```

In [25]:

```
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(128, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
model.add(layers.Dense(512, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

```
In [26]: model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d (Conv2D)             (None, 148, 148, 32)      896       
                                                                 
 max_pooling2d (MaxPooling2D  (None, 74, 74, 32)       0         
 )                                                               
                                                                 
 conv2d_1 (Conv2D)           (None, 72, 72, 64)        18496     
                                                                 
 max_pooling2d_1 (MaxPooling  (None, 36, 36, 64)       0         
 2D)                                                             
                                                                 
 conv2d_2 (Conv2D)           (None, 34, 34, 128)       73856     
                                                                 
 max_pooling2d_2 (MaxPooling  (None, 17, 17, 128)      0         
 2D)                                                             
                                                                 
 conv2d_3 (Conv2D)           (None, 15, 15, 128)       147584    
                                                                 
 max_pooling2d_3 (MaxPooling  (None, 7, 7, 128)        0         
 2D)                                                             
                                                                 
 flatten (Flatten)           (None, 6272)              0         
                                                                 
 dense (Dense)               (None, 512)               3211776   
                                                                 
 dense_1 (Dense)             (None, 1)                 513       
                                                                 
=================================================================
Total params: 3,453,121
Trainable params: 3,453,121
Non-trainable params: 0
_________________________________________________________________
```

In [27]:

```
from keras import optimizers
model.compile(loss='binary_crossentropy',
optimizer=optimizers.RMSprop(lr=1e-4),
metrics=['acc'])
```

```
C:\Users\theoj\AppData\Roaming\Python\Python39\site-packages\keras\optimizers\optimizer_v2\rmsprop.py:140: Use
rWarning: The `lr` argument is deprecated, use `learning_rate` instead.
  super().__init__(name, **kwargs)
```

In [28]:

```
from keras.preprocessing.image import ImageDataGenerator
```

In [29]:

```
train_datagen = ImageDataGenerator(rescale=1./255)
test_datagen = ImageDataGenerator(rescale=1./255)
```

In [30]:

```
train_generator = train_datagen.flow_from_directory(train_dir,
target_size=(150, 150),batch_size=20,class_mode='binary')
```

```
Found 2000 images belonging to 2 classes.
```

In [33]:

```
validation_generator = test_datagen.flow_from_directory(
validation_dir,
    target_size=(150, 150),
batch_size=20,
class_mode='binary')
```

```
Found 1000 images belonging to 2 classes.
```

In [32]:

```
for data_batch, labels_batch in train_generator:
    print('data batch shape:', data_batch.shape)
    print('labels batch shape:', labels_batch.shape)
break
```

```
data batch shape: (20, 150, 150, 3)
labels batch shape: (20,)
data batch shape: (20, 150, 150, 3)
labels batch shape: (20,)
data batch shape: (20, 150, 150, 3)
labels batch shape: (20,)
data batch shape: (20, 150, 150, 3)
labels batch shape: (20,)
data batch shape: (20, 150, 150, 3)
labels batch shape: (20,)
data batch shape: (20, 150, 150, 3)
labels batch shape: (20,)
data batch shape: (20, 150, 150, 3)
labels batch shape: (20,)
data batch shape: (20, 150, 150, 3)
labels batch shape: (20,)
data batch shape: (20, 150, 150, 3)
labels batch shape: (20,)
data batch shape: (20, 150, 150, 3)
labels batch shape: (20,)
```

In [34]:

```
history = model.fit_generator(
train_generator,
steps_per_epoch=100,
epochs=30,
validation_data=validation_generator,
validation_steps=50)
```

```
Epoch 1/30

C:\Users\theoj\AppData\Local\Temp\ipykernel_27392\2724264653.py:1: UserWarning: `Model.fit_generator` is depre
cated and will be removed in a future version. Please use `Model.fit`, which supports generators.
  history = model.fit_generator(

100/100 [==============================] - 17s 84ms/step - loss: 0.6888 - acc: 0.5395 - val_loss: 0.6743 - val
_acc: 0.5860
Epoch 2/30
100/100 [==============================] - 6s 64ms/step - loss: 0.6572 - acc: 0.6120 - val_loss: 0.6588 - val_
acc: 0.5680
Epoch 3/30
100/100 [==============================] - 6s 63ms/step - loss: 0.6062 - acc: 0.6710 - val_loss: 0.5988 - val_
acc: 0.6670
Epoch 4/30
100/100 [==============================] - 6s 64ms/step - loss: 0.5619 - acc: 0.7105 - val_loss: 0.7107 - val_
acc: 0.5770
Epoch 5/30
100/100 [==============================] - 6s 64ms/step - loss: 0.5397 - acc: 0.7350 - val_loss: 0.6151 - val_
acc: 0.6590
Epoch 6/30
100/100 [==============================] - 6s 64ms/step - loss: 0.5080 - acc: 0.7430 - val_loss: 0.5762 - val_
acc: 0.6890
Epoch 7/30
100/100 [==============================] - 6s 64ms/step - loss: 0.4800 - acc: 0.7610 - val_loss: 0.6345 - val_
acc: 0.6820
Epoch 8/30
100/100 [==============================] - 6s 64ms/step - loss: 0.4592 - acc: 0.7725 - val_loss: 0.5599 - val_
acc: 0.7100
Epoch 9/30
100/100 [==============================] - 6s 64ms/step - loss: 0.4231 - acc: 0.8030 - val_loss: 0.6570 - val_
acc: 0.6780
Epoch 10/30
100/100 [==============================] - 6s 64ms/step - loss: 0.4106 - acc: 0.8065 - val_loss: 0.5437 - val_
acc: 0.7170
Epoch 11/30
```

```
100/100 [==============================] - 6s 64ms/step - loss: 0.3788 - acc: 0.8300 - val_loss: 0.6037 - val_
acc: 0.7000
Epoch 12/30
100/100 [==============================] - 6s 64ms/step - loss: 0.3546 - acc: 0.8445 - val_loss: 0.5605 - val_
acc: 0.7210
Epoch 13/30
100/100 [==============================] - 6s 64ms/step - loss: 0.3336 - acc: 0.8560 - val_loss: 0.5511 - val_
acc: 0.7310
Epoch 14/30
100/100 [==============================] - 6s 63ms/step - loss: 0.3143 - acc: 0.8700 - val_loss: 0.6061 - val_
acc: 0.7280
Epoch 15/30
100/100 [==============================] - 6s 63ms/step - loss: 0.2887 - acc: 0.8780 - val_loss: 0.6655 - val_
acc: 0.7170
Epoch 16/30
100/100 [==============================] - 6s 63ms/step - loss: 0.2754 - acc: 0.8990 - val_loss: 0.5885 - val_
acc: 0.7300
Epoch 17/30
100/100 [==============================] - 6s 64ms/step - loss: 0.2470 - acc: 0.9085 - val_loss: 0.6594 - val_
acc: 0.7220
Epoch 18/30
100/100 [==============================] - 6s 64ms/step - loss: 0.2272 - acc: 0.9070 - val_loss: 0.6858 - val_
acc: 0.7170
Epoch 19/30
100/100 [==============================] - 6s 63ms/step - loss: 0.2107 - acc: 0.9250 - val_loss: 0.6215 - val_
acc: 0.7320
Epoch 20/30
100/100 [==============================] - 6s 64ms/step - loss: 0.1887 - acc: 0.9320 - val_loss: 0.7340 - val_
acc: 0.7170
Epoch 21/30
100/100 [==============================] - 6s 63ms/step - loss: 0.1672 - acc: 0.9415 - val_loss: 0.6930 - val_
acc: 0.7280
Epoch 22/30
100/100 [==============================] - 7s 65ms/step - loss: 0.1528 - acc: 0.9535 - val_loss: 0.6847 - val_
acc: 0.7370
Epoch 23/30
100/100 [==============================] - 6s 64ms/step - loss: 0.1327 - acc: 0.9565 - val_loss: 0.7152 - val_
acc: 0.7380
Epoch 24/30
100/100 [==============================] - 6s 63ms/step - loss: 0.1151 - acc: 0.9640 - val_loss: 0.8074 - val_
acc: 0.7220
Epoch 25/30
100/100 [==============================] - 6s 64ms/step - loss: 0.1045 - acc: 0.9685 - val_loss: 0.7750 - val_
```

```
acc: 0.7160
Epoch 26/30
100/100 [==============================] - 6s 64ms/step - loss: 0.0880 - acc: 0.9730 - val_loss: 1.0080 - val_
acc: 0.7060
Epoch 27/30
100/100 [==============================] - 6s 63ms/step - loss: 0.0750 - acc: 0.9790 - val_loss: 0.9859 - val_
acc: 0.7120
Epoch 28/30
100/100 [==============================] - 6s 65ms/step - loss: 0.0778 - acc: 0.9740 - val_loss: 0.8656 - val_
acc: 0.7280
Epoch 29/30
100/100 [==============================] - 6s 64ms/step - loss: 0.0644 - acc: 0.9800 - val_loss: 0.9399 - val_
acc: 0.7200
Epoch 30/30
100/100 [==============================] - 6s 64ms/step - loss: 0.0467 - acc: 0.9885 - val_loss: 0.9238 - val_
acc: 0.7350
```

In [35]:
```
model.save('cats_and_dogs_small_1.h5')
```

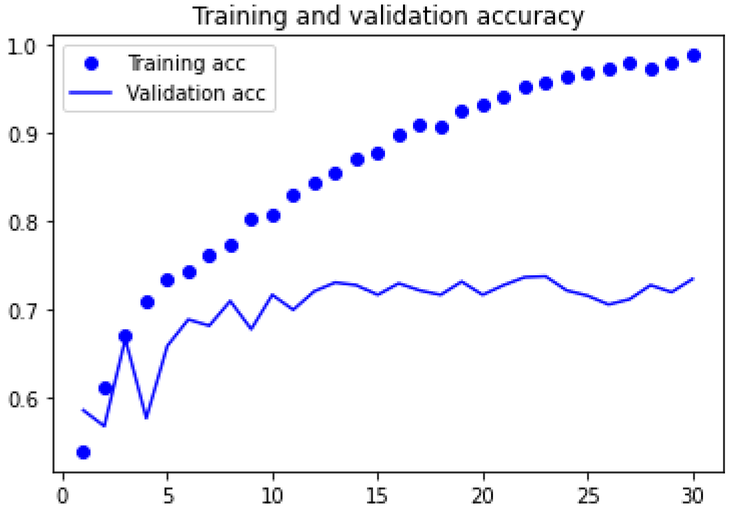
In [36]:
```
import matplotlib.pyplot as plt
```

In [37]:
```
acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']
```

In [38]:
```
epochs = range(1, len(acc) + 1)
```

```
In [39]: plt.plot(epochs, acc, 'bo', label='Training acc')
         plt.plot(epochs, val_acc, 'b', label='Validation acc')
         plt.title('Training and validation accuracy')
         plt.legend()
```

```
Out[39]: <matplotlib.legend.Legend at 0x1e823008d60>
```



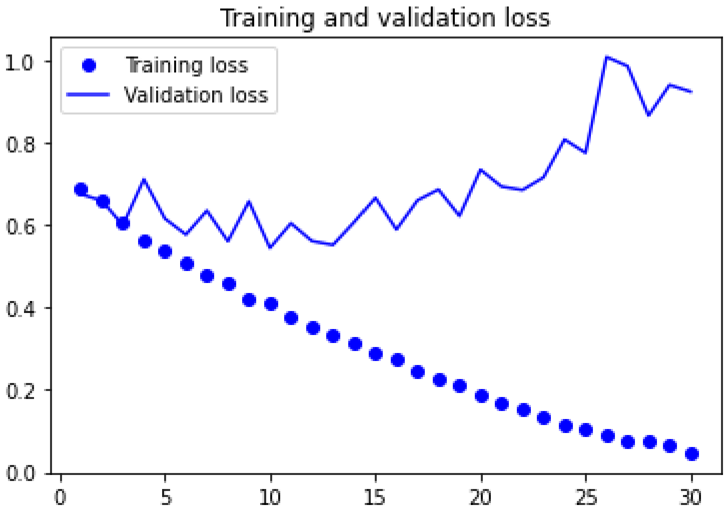
```
In [40]: plt.figure()
```

```
Out[40]: <Figure size 432x288 with 0 Axes>

         <Figure size 432x288 with 0 Axes>
```

In [41]:
```
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()
```

Out[41]:
```
<matplotlib.legend.Legend at 0x1e822fb0d30>
```



In [42]:
```
plt.show()
```

In [43]:
```
datagen = ImageDataGenerator(
rotation_range=40,
width_shift_range=0.2,
height_shift_range=0.2,
shear_range=0.2,
zoom_range=0.2,
horizontal_flip=True,
fill_mode='nearest')
```

In [44]:
```
from tensorflow.keras.preprocessing import image
```

In [45]:

```
fnames = [os.path.join(train_cats_dir, fname) for
    fname in os.listdir(train_cats_dir)]
```

In [46]:

```
img_path = fnames[3]
```

In [47]:

```
img = image.load_img(img_path, target_size=(150, 150))
```

In [48]:

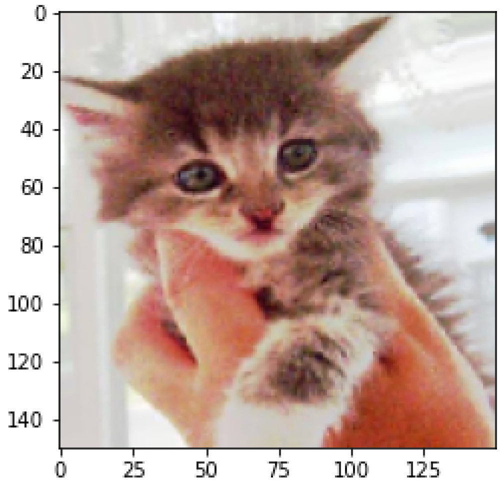
```
x = image.img_to_array(img)
```

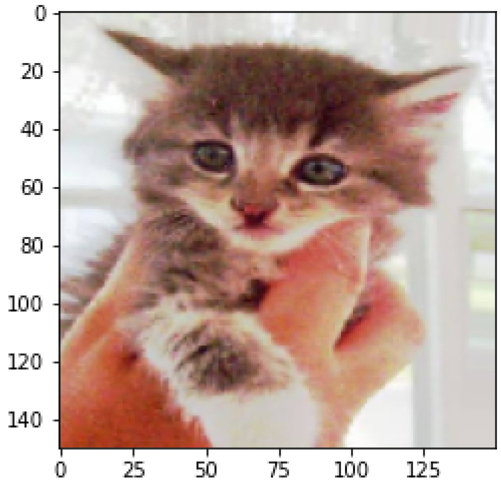
In [49]:

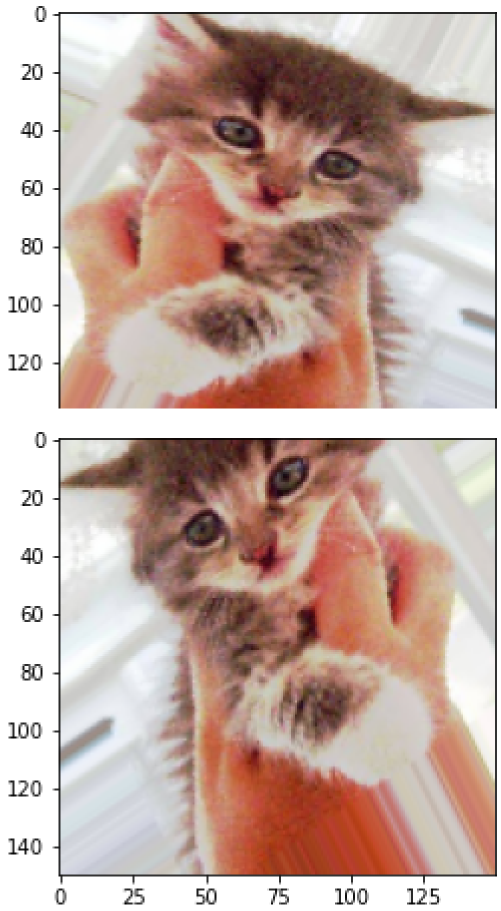
```
x = x.reshape((1,) + x.shape)
```

In [50]:

```
i=0
for batch in datagen.flow(x, batch_size=1):
    plt.figure(i)
    imgplot = plt.imshow(image.array_to_img(batch[0]))
    i += 1
    if i % 4 == 0:
        break
plt.show()
```







In [51]:

```
train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,)
test_datagen = ImageDataGenerator(rescale=1./255)
```

In [52]:

```
train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=(150, 150),
    batch_size=32,
    class_mode='binary')
```

```
Found 2000 images belonging to 2 classes.
```

In [53]:

```
validation_generator = test_datagen.flow_from_directory(
    validation_dir,
    target_size=(150, 150),
    batch_size=32,
    class_mode='binary')
```

```
Found 1000 images belonging to 2 classes.
```

In [54]:

```
history = model.fit_generator(
    train_generator,
    steps_per_epoch=100,
    epochs=100,
    validation_data=validation_generator,
    validation_steps=50)
```

```
C:\Users\theoj\AppData\Local\Temp\ipykernel_27392\121998857.py:1: UserWarning: `Model.fit_generator` is deprec
ated and will be removed in a future version. Please use `Model.fit`, which supports generators.
  history = model.fit_generator(

Epoch 1/100
 63/100 [=================>...........] - ETA: 7s - loss: 0.7114 - acc: 0.6960WARNING:tensorflow:Your input r
an out of data; interrupting training. Make sure that your dataset or generator can generate at least `steps_p
er_epoch * epochs` batches (in this case, 10000 batches). You may need to use the repeat() function when build
ing your dataset.
WARNING:tensorflow:Your input ran out of data; interrupting training. Make sure that your dataset or generator
can generate at least `steps_per_epoch * epochs` batches (in this case, 50 batches). You may need to use the r
epeat() function when building your dataset.
100/100 [==============================] - 16s 156ms/step - loss: 0.7114 - acc: 0.6960 - val_loss: 0.5703 - va
l_acc: 0.7150
```

In [55]:

```
model.save('cats_and_dogs_small_2.h5')
```

In [ ]: