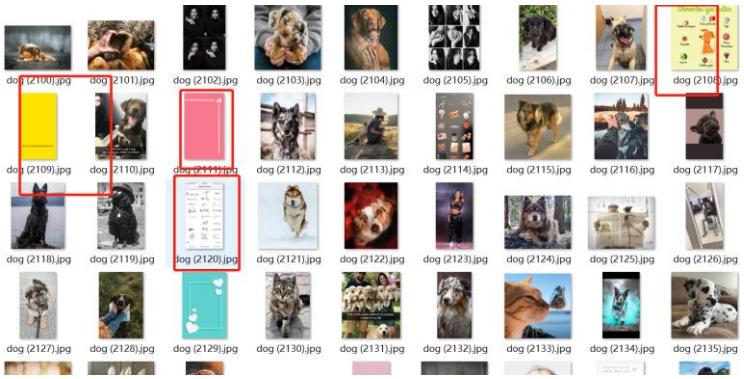
Use Google's ImageAssistant, search for cats and dogs in Pinterest, download, cats and dogs download ~1600 each. But in so many images there are many duplicate images mixed in, which will be misleading for the production of the dataset later, so manually remove the duplicate images.



If the total file is too large, select some images and divide them into trains and test folders to be used later.

I went to find pictures of dogs and cats because I saw that Kaggle had a cat vs dog proposition contest for the cat and dog training set a few years ago and that cats and dogs are the more classic cases in training.

Then the following is divided into three steps, dataset, train, and predict.

```
if isinstance(validation_size, float):
    validation_size = int(validation_size * images.shape[0])

validation_images = images[:validation_size]
validation_labels = labels[:validation_size]
validation_img_names = img_names[:validation_size]
validation_cls = cls[:validation_size]

train_images = images[validation_size:]
train_labels = labels[validation_size:]
train_img_names = img_names[validation_size:]
train_cls = cls[validation_size:]

data_sets.train = DataSet(train_images,train_labels,train_img_names,train_cls)
data_sets.valid = DataSet(validation_images,validation_labels,validation_img_names,validation_cls)

return data_sets
```

The following steps are divided into three, dataset, train, predict, train with tensorflowimport cv, os, glob, sklearn.utils, shuffle numpy as np.

```
image_size = 64
num_channels = 3
images = []

path = 'cat.1.jpg'
image = cv2.imread(path)
image = cv2.resize(image,(image_size,image_size),0,0,cv2.INTER_LINEAR)
images.append(image)
images = np.array(images,dtype=np.uint8)
images = images.astype('float32')
```