```
import cv2
import numpy as np
import os
from tensorflow.keras.applications.vgg16 import VGG16, preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array


base_model = VGG16(weights='imagenet', include_top=False, input_shape=(100, 100, 3))


def load_and_preprocess_images(folder):
    images = []
    for filename in os.listdir(folder):
        img_path = os.path.join(folder, filename)
        img = cv2.imread(img_path)
        if img is not None:
            img = cv2.resize(img, (100, 100))
            img = img_to_array(img)
            img = np.expand_dims(img, axis=0)
            img = preprocess_input(img)
            features = base_model.predict(img)
            images.append(features.flatten())
    return np.array(images)


cat_features = load_and_preprocess_images(r"H:\Data\Cats.Dogs\TV_cats")
dog_features = load_and_preprocess_images(r"H:\Data\Cats.Dogs\TV_dogs")
```

```
1/1 [==============================] - 0s 234ms/step
1/1 [==============================] - 0s 56ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 76ms/step
1/1 [==============================] - 0s 70ms/step
1/1 [==============================] - 0s 83ms/step
1/1 [==============================] - 0s 75ms/step
1/1 [==============================] - 0s 78ms/step
1/1 [==============================] - 0s 74ms/step
1/1 [==============================] - 0s 71ms/step
1/1 [==============================] - 0s 84ms/step
1/1 [==============================] - 0s 94ms/step
1/1 [==============================] - 0s 82ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 80ms/step
1/1 [==============================] - 0s 84ms/step
1/1 [==============================] - 0s 79ms/step
1/1 [==============================] - 0s 69ms/step
1/1 [==============================] - 0s 83ms/step
1/1 [==============================] - 0s 75ms/step
1/1 [==============================] - 0s 68ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 64ms/step
1/1 [==============================] - 0s 54ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 56ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 56ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 55ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 51ms/step
1/1 [==============================] - 0s 57ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 54ms/step
```

```
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 54ms/step
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 50ms/step
1/1 [==============================] - 0s 66ms/step
1/1 [==============================] - 0s 61ms/step
1/1 [==============================] - 0s 51ms/step
1/1 [==============================] - 0s 59ms/step
1/1 [==============================] - 0s 62ms/step
1/1 [==============================] - 0s 65ms/step
1/1 [==============================] - 0s 54ms/step
1/1 [==============================] - 0s 58ms/step
1/1 [==============================] - 0s 51ms/step
1/1 [==============================] - 0s 59ms/step
```

```
cat_labels = np.zeros(len(cat_features))  # label 0 for cats
dog_labels = np.ones(len(dog_features))   # label 1 for dogs


x = np.concatenate((cat_features, dog_features), axis=0)
y = np.concatenate((cat_labels, dog_labels), axis=0)


from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report


x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)


svm_model = SVC(kernel='linear')


svm_model.fit(x_train, y_train)
```

```
SVC(kernel='linear')
```

```
y_pred = svm_model.predict(x_test)


accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")
```

```
Accuracy: 0.88625
```

```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

         0.0       0.90      0.88      0.89       422
         1.0       0.87      0.89      0.88       378

    accuracy                           0.89       800
   macro avg       0.89      0.89      0.89       800
weighted avg       0.89      0.89      0.89       800
```

```
sample= load_and_preprocess_images(r"H:\Data\Cats.Dogs\Sample_Dog")
sample_predict= svm_model.predict(sample)
```

```
1/1 [==============================] - 1s 569ms/step
```

```
if sample_predict[0] == 0:
    print("Predicted: Cat")
else:
    print("Predicted: Dog")
```

```
Predicted: Dog
```