```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
import os



dataset_path = '/content/drive/MyDrive/Dataset_ML'



# List folders (classes) in the dataset
print("Classes in the dataset:", os.listdir(dataset_path))
```

```
Classes in the dataset: ['0. Cut Shot', '1. Cover Drive', '2. Straight Drive', '3. Pull Shot', '4. Leg Glance Shot', '5.
```

```
# List class folders
classes = os.listdir(dataset_path)
print("Classes in the dataset:", classes)
```

```
Classes in the dataset: ['0. Cut Shot', '1. Cover Drive', '2. Straight Drive', '3. Pull Shot', '4. Leg Glance Shot', '5.
```

```
# Count images in each folder
for class_name in classes:
    class_folder = os.path.join(dataset_path, class_name)
    image_count = len(os.listdir(class_folder))
    print(f"Class '{class_name}' contains {image_count} images.")
```

```
Class '0. Cut Shot' contains 641 images.
Class '1. Cover Drive' contains 600 images.
Class '2. Straight Drive' contains 600 images.
Class '3. Pull Shot' contains 600 images.
Class '4. Leg Glance Shot' contains 600 images.
Class '5. Scoop Shot' contains 600 images.
```

```
import cv2
import numpy as np
from tensorflow.keras.utils import to_categorical



IMG_SIZE = 224  # Resize all images to 224x224
classes = ['0. Cut Shot', '1. Cover Drive', '2. Straight Drive', '3. Pull Shot', '4. Leg Glance
class_to_label = {name: idx for idx, name in enumerate(classes)}  # Map class names to numeric l



data = []
labels = []



# Load images from each class folder
for class_name in classes:
    class_folder = os.path.join(dataset_path, class_name)
    label = class_to_label[class_name]  # Get numeric label

    for file_name in os.listdir(class_folder):
        file_path = os.path.join(class_folder, file_name)

        # Read the image
        img = cv2.imread(file_path, cv2.IMREAD_GRAYSCALE)  # Load in grayscale
        if img is None:
```

```
            print(f"Failed to load {file_path}. Skipping...")
            continue

        # Resize image
        img = cv2.resize(img, (IMG_SIZE, IMG_SIZE))

        # Normalize image
        img = img / 255.0  # Scale pixel values to [0, 1]

        # Flatten the image to 1D (SVM requires 1D feature vectors)
        img = img.flatten()

        # Append to data and labels
        data.append(img)
        labels.append(label)


# Convert lists to NumPy arrays
data = np.array(data, dtype="float32")
labels = np.array(labels)
```

```
import cv2
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import classification_report, accuracy_score
from tensorflow.keras.utils import to_categorical

# Split the data into training and testing sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(data, labels, test_size=0.2, random_state=42


# Train an SVM classifier
svm_classifier = SVC(kernel='linear')  # You can try other kernels like 'rbf' or 'poly'
svm_classifier.fit(X_train, y_train)
```

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

```
# Evaluate the model
y_pred = svm_classifier.predict(X_test)


# Print classification report and accuracy
print("Classification Report:\n", classification_report(y_test, y_pred, target_names=classes))
print("Accuracy Score:", accuracy_score(y_test, y_pred))
```

```
Classification Report:
                     precision    recall  f1-score   support

          0. Cut Shot       0.71      0.63      0.66       150
        1. Cover Drive       0.62      0.63      0.62       123
     2. Straight Drive       0.67      0.72      0.69       127
         3. Pull Shot       0.65      0.62      0.64       109
    4. Leg Glance Shot       0.69      0.71      0.70       114
         5. Scoop Shot       0.75      0.79      0.77       106

             accuracy                           0.68       729
            macro avg       0.68      0.68      0.68       729
         weighted avg       0.68      0.68      0.68       729

    Accuracy Score: 0.6803840877914952
```

```
Start coding or generate with AI.
```