این کد شامل مراحل پردازش تصویر، استخراج ویژگی‌ها و آموزش مدل‌های یادگیری ماشین برای دسته‌بندی تصاویر برگ‌ها است. مراحل اصلی شامل خواندن تصاویر، پردازش آن‌ها، استخراج ویژگی‌ها و آموزش مدل‌های مختلف یادگیری ماشین می‌باشد. هدف نهایی، مقایسه دقت مدل‌ها و انتخاب بهترین مدل است.

مراحل پردازش تصویر

وارد کردن کتابخانه‌ها
کتابخانه‌های مورد نیاز شامل OpenCV برای پردازش تصویر، Numpyبرای محاسبات عددی، و کتابخانه‌های مختلف sklearnبرای یادگیری ماشین و اعتبارسنجی هستند.

```
import os
import cv2
import numpy as np
import imghdr
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from skimage.feature import hog
from skimage.segmentation import find_boundaries
from sklearn.cluster import KMeans
```

تعریف توابع پردازش تصویر
چندین تابع برای پردازش تصویر و استخراج ویژگی‌ها تعریف شده‌اند:
:`nmd` -خوشه‌بندی رنگ‌ها و یافتن مرزهای شی.
:`lineArt` -ایجاد طرح‌های خطی از تصویر.
:`canny_img` -استخراج لبه‌ها با استفاده از الگوریتم.Canny
:`hog_img` -استخراج ویژگی‌های.HOG
:`laplace_img` -فیلتر لاپلاس برای تشخیص لبه‌ها.
:`momentom` -محاسبه مومنت‌های تصویری.
:`hist_img` -محاسبه هیستوگرام رنگ
در این کد، ویژگی‌های مختلفی از تصاویر استخراج می‌شوند که هر کدام مزایای خاص خود را در تحلیل و دسته‌بندی تصاویر دارند. در زیر به توضیح مزایای هر یک از این ویژگی‌ها پرداخته می‌شود:

- (HOG)
استخراج لبه‌ها و ویژگی‌های محلی HOG :بر اساس گرادیان‌های شدت تصویر عمل می‌کند و الگوهای لبه و گوشه‌ها را تشخیص می‌دهد.
تطبیق پذیری با تغییرات روشنایی: با توجه به استفاده از گرادیان‌ها، HOGنسبت به تغییرات روشنایی و کنتراست تصویر مقاوم است.
کارایی در تشخیص اشیا: به ویژه در تشخیص اشیا و اشکال با لبه‌های واضح بسیار موثر است.

- Line Art

برجسته کردن خطوط اصلی تصویر: این ویژگی خطوط اصلی و مرزهای اشیا را برجسته می‌کند.
سادگی محاسباتی: پردازش تصویر به خطوط و مرزها ساده و سریع است.
استفاده در تشخیص اشیا: می‌تواند به عنوان یک ویژگی تکمیلی در کنار سایر ویژگی‌ها برای تشخیص اشیا مورد استفاده قرار گیرد.

- Canny

تشخیص دقیق لبه‌ها: الگوریتم Canny به دلیل استفاده از فیلترهای گوسی و محاسبات دقیق، لبه‌ها را به خوبی تشخیص می‌دهد.
کاهش نویز: فیلترهای گوسی در این الگوریتم به کاهش نویز و استخراج لبه‌های واقعی کمک می‌کنند.
استفاده در تشخیص شکل و لبه: مناسب برای کاربردهایی که نیاز به تشخیص دقیق شکل‌ها و لبه‌ها دارند.

- Hu Moments

تشخیص شکل‌ها Hu Moments :برای تشخیص اشکال و الگوهای کلی در تصاویر استفاده می‌شود.
مقاومت در برابر تغییرات چرخشی و مقیاسی:** این ویژگی‌ها نسبت به چرخش، مقیاس و انتقال تصویر مقاوم هستند.
کاربرد در تطبیق اشکال: می‌تواند برای تطبیق و تشخیص اشکال مشابه در تصاویر مختلف مورد استفاده قرار گیرد.

- Color Histogram

توصیف کلی رنگ‌ها: هیستوگرام رنگ توزیع کلی رنگ‌ها در تصویر را نشان می‌دهد.
تطبیق رنگ‌ها: در کاربردهایی که رنگ‌ها مهم هستند (مانند تشخیص اشیاء بر اساس رنگ)، این ویژگی بسیار مفید است.
مقاومت در برابر تغییرات جزئی: نسبت به تغییرات جزئی در شکل و اندازه اشیاء مقاوم است.

- Laplacian Filter

تشخیص تغییرات سریع شدت: این فیلتر تغییرات سریع در شدت تصویر را تشخیص می‌دهد.
استخراج لبه‌ها و جزئیات: مناسب برای استخراج لبه‌ها و جزئیات دقیق در تصویر.

خواندن و پیش‌پردازش تصاویر
تصاویر از فولدر مشخص خوانده شده و پس از انجام پردازش‌های مختلف به سایز مشخص تغییر اندازه داده می‌شوند.
ویژگی‌های استخراج شده از تصاویر نیز نرمال‌سازی شده و به آرایه داده‌ها اضافه می‌شوند.

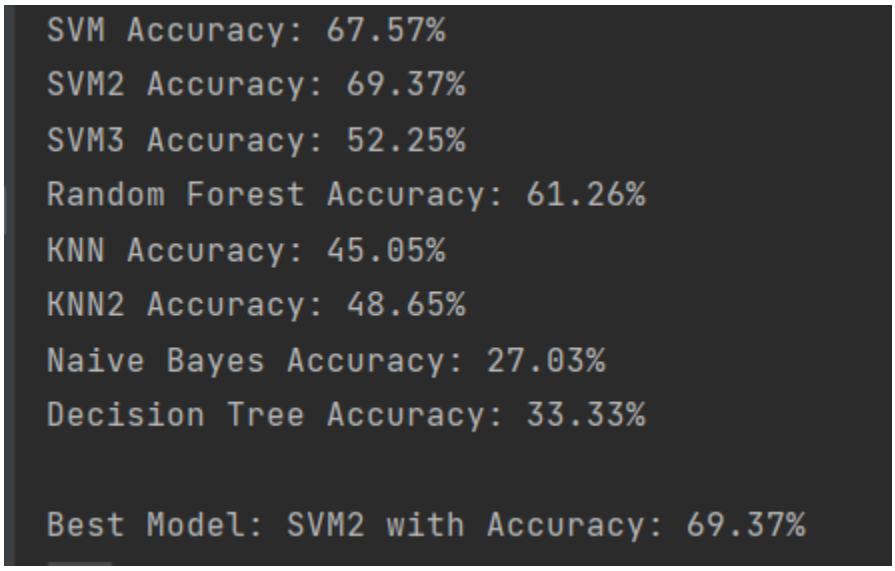
```
data_dir = 'C:\\Users\\ASUS\\Desktop\\leaves'
image_size = (128, 128)
data = []
labels = []
i = 0

for folder_name in os.listdir(data_dir):
    folder_path = os.path.join(data_dir, folder_name)
    if os.path.isdir(folder_path):
        for filename in os.listdir(folder_path):
            file_path = os.path.join(folder_path, filename)
            if imghdr.what(file_path):
```

```
image = cv2.imread(file_path, cv2.IMREAD_COLOR)
if image is not None:
    i += 1
    image2 = lineArt(image, i)
    image3 = canny_img(image, i)
    image4 = hog_img(image, i)
    image5 = momentom(image, i)
    image = cv2.resize(image, image_size)
    image2 = cv2.resize(image2, image_size)
    image3 = cv2.resize(image3, image_size)
    image4 = cv2.resize(image4, image_size)
    image5 = cv2.resize(image5, image_size)

    image = (image / 255.0).flatten()
    image2 = (image2 / 255.0).flatten()
    image3 = (image3 / 255.0).flatten()
    image4 = (image4 / 255.0).flatten()
    image5 = (image5 / 255.0).flatten()

    features = np.concatenate((image, image5))
    data.append(features)
    labels.append(folder_name)
```



Canny

```
SVM Accuracy: 53.15%
SVM2 Accuracy: 26.13%
SVM3 Accuracy: 7.21%
Random Forest Accuracy: 54.95%
KNN Accuracy: 41.44%
KNN2 Accuracy: 43.24%
Naive Bayes Accuracy: 7.21%
Decision Tree Accuracy: 18.02%

Best Model: Random Forest with Accuracy: 54.95%
```

Laplacian

```
SVM Accuracy: 72.97%
SVM2 Accuracy: 72.07%
SVM3 Accuracy: 72.97%
Random Forest Accuracy: 67.57%
KNN Accuracy: 50.45%
KNN2 Accuracy: 52.25%
Naive Bayes Accuracy: 51.35%
Decision Tree Accuracy: 34.23%

Best Model: SVM with Accuracy: 72.97%
```

Artline

```
SVM Accuracy: 71.17%
SVM2 Accuracy: 58.56%
SVM3 Accuracy: 27.03%
Random Forest Accuracy: 62.16%
KNN Accuracy: 48.65%
KNN2 Accuracy: 53.15%
Naive Bayes Accuracy: 48.65%
Decision Tree Accuracy: 33.33%

Best Model: SVM with Accuracy: 71.17%
```

Canny Artline

```
SVM Accuracy: 72.97%
SVM2 Accuracy: 67.57%
SVM3 Accuracy: 31.53%
Random Forest Accuracy: 65.77%
KNN Accuracy: 55.86%
KNN2 Accuracy: 54.95%
Naive Bayes Accuracy: 3.60%
Decision Tree Accuracy: 28.83%

Best Model: SVM with Accuracy: 72.97%
```

Artline Canny

```
SVM Accuracy: 71.17%
SVM2 Accuracy: 70.27%
SVM3 Accuracy: 51.35%
Random Forest Accuracy: 65.77%
KNN Accuracy: 48.65%
KNN2 Accuracy: 53.15%
Naive Bayes Accuracy: 47.75%
Decision Tree Accuracy: 31.53%

Best Model: SVM with Accuracy: 71.17%
```

Momentum

```
SVM Accuracy: 72.97%
SVM2 Accuracy: 63.06%
SVM3 Accuracy: 26.13%
Random Forest Accuracy: 61.26%
KNN Accuracy: 54.05%
KNN2 Accuracy: 54.95%
Naive Bayes Accuracy: 3.60%
Decision Tree Accuracy: 40.54%

Best Model: SVM with Accuracy: 72.97%
```

Hog Artline hist_img

آموزش و ارزیابی مدل‌های یادگیری ماشین

داده‌ها به دو مجموعه آموزشی و آزمایشی با نسبت ۷۵ به ۲۵ تقسیم می‌شوند.

```
models = {
    'SVM': SVC(kernel='linear', random_state=32),
    'SVM2': SVC(kernel='poly', degree=3, random_state=42),
    'SVM3': SVC(kernel='poly', degree=2, random_state=42),
    'Random Forest': RandomForestClassifier(n_estimators=100, random_state=42),
    'KNN': KNeighborsClassifier(n_neighbors=3),
    'KNN2': KNeighborsClassifier(n_neighbors=4),
    'Naive Bayes': GaussianNB(),
```

```
    'Decision Tree': DecisionTreeClassifier(random_state=42)
}
```

3. ####آموزش و ارزیابی مدل‌ها
مدل‌ها بر روی داده‌های آموزشی آموزش داده شده و دقت آن‌ها بر روی داده‌های آزمایشی محاسبه می‌شود.

```
accuracies = {}
for model_name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    accuracies[model_name] = accuracy
    print(f'{model_name} Accuracy: {accuracy * 100:.2f}%')
```

بهترین مدل بر اساس دقت انتخاب و چاپ می‌شود.

```
`best_model_name = max(accuracies, key=accuracies.get)
best_accuracy = accuracies[best_model_name]
print(f'\nBest Model: {best_model_name} with Accuracy: {best_accuracy * 100:.2f}%')
```

نتیجه‌گیری
در این گزارش، فرآیند کامل پردازش تصویر، استخراج ویژگی‌ها و آموزش مدل‌های یادگیری ماشین برای دسته‌بندی تصاویر برگ‌ها شرح داده شد. مدل‌های مختلف یادگیری ماشین با استفاده از ویژگی‌های استخراج شده آموزش داده شده و دقت آن‌ها ارزیابی شد. بهترین مدل بر اساس دقت انتخاب و معرفی شد. در انتها الگوریتم svm خطی بهترین بود و ویژگی‌های منتخب Hog Artline hist_img بودند