به نام خدا

پرستو غلامی
۹۹۳۶۱۳۰۴۸

پروژه دوم ماشین لرنینگ
الگوریتم خوشه بندی و کلاس بندی روی leaves.csv

این گزارش به تحلیل و اجرای کد ارائه شده برای پردازش داده‌ها، حذف داده‌های پرت، نرمال‌سازی، خوشه‌بندی، تقسیم داده‌های فایل leaves.csv به مجموعه‌های آموزشی و آزمایشی، آموزش چندین مدل یادگیری ماشین و ارزیابی دقت آن‌ها می‌پردازد.

- خواندن داده‌ها

داده‌ها از فایل CSV با مسیر `C://Users//ASUS//Desktop//leaves.csv` خوانده شده و به یک DataFrame تبدیل شده‌اند. سپس نام ستون‌ها به ترتیب `Column1` تا `Column16` تنظیم شده‌اند.

```
df = pd.read_csv("C://Users//ASUS//Desktop//leaves.csv", header=None)
column_names = ['Column1', 'Column2', ... , 'Column16']
df.columns = column_names
```

- حذف داده‌های پرت

با استفاده از z-score داده‌های پرت شناسایی و حذف شده‌اند. مقدار آستانه برای تشخیص پرت‌ها برابر با ۳ در نظر گرفته شده است.

```
z_scores = np.abs((df - df.mean()) / df.std())
threshold = 3
outliers = (z_scores > threshold).any(axis=1)
df = df[~outliers]
```

- نرمال‌سازی

تمام ستون‌ها به جز ستون اول (که لیبل‌ها هستند) با استفاده از حداکثر مقدار هر ستون نرمال‌سازی شده‌اند.

```
for col in df.columns[1:]:
   df[col] = df[col] / df[col].max()
```

- استانداردسازی

داده‌ها با استفاده از StandardScaler استانداردسازی شده‌اند.

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(df.drop(['Column1'], axis=1))
```

- لیبل

لیبل برای خود یک دیتافریم داشته و از ویژگی‌ها حذف شده است.

```
labels = df['Column1']
df = df.drop(['Column1'], axis=1)
```

در ادامه به خوشه بندی و کلاس بندی آنها پرداخته‌ایم:

- خوشه بندی

با استفاده از الگوریتمKMeans ، داده‌ها به ۳۶ خوشه تقسیم شده(این تعداد براساس تعداد لیبل هاست) و تعداد اعضای هر خوشه چاپ شده است.

```
kmeans = KMeans(n_clusters=labels.max(), random_state=42)
kmeans.fit(X_scaled)
cluster_labels = kmeans.labels_
```

خوشه ۱۵ با تعداد اعضای ۱۸تا بیشترین عضو و خوشه‌های ۲۶ و ۲۹ با تعداد اعضای ۱ کمترین عضو را دارد.

```
cluster 35: 13
cluster 8: 17
cluster 11: 14
cluster 1: 12
cluster 24: 17
cluster 15: 18
cluster 16: 8
cluster 27: 5
cluster 21: 12
cluster 33: 12
cluster 2: 11
cluster 14: 8
cluster 18: 10
cluster 23: 5
cluster 22: 3
cluster 4: 2
cluster 5: 8
cluster 3: 14
cluster 20: 11
cluster 30: 6
cluster 17: 7
cluster 25: 8
cluster 36: 2
cluster 19: 2
```

cluster 31: 10
cluster 9: 12
cluster 10: 2
cluster 13: 8
cluster 26: 1
cluster 12: 7
cluster 34: 7
cluster 7: 6
cluster 29: 1
cluster 28: 2
cluster 6: 2
cluster 32: 2

- کلاس بندی

سپس کلاس بندی انجام داده‌ایم که ابتدا داده‌ها را به مجموعه‌های آموزشی و آزمایشی به نسبت ۸۰ به ۲۰ تقسیم شده‌اند.

```
X_train, X_test, y_train, y_test = train_test_split(X_scaled, labels, test_size=0.2,
random_state=32)
```

برای آموزش مدل‌ها و ارزیابی دقت از چندین مدل یادگیری ماشین شاملSVM ، Random Forest، KNN، Naive Bayesو Decision Tree آموزش داده شده و دقت هر کدام محاسبه و مقایسه شده است.

```
models = {
   'SVM': SVC(kernel='linear', random_state=32),
   ...
   'Decision Tree': DecisionTreeClassifier(random_state=42)
}

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}%')
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}%')
```

بهترین مدل بر اساس دقت انتخاب شده و دقت آن چاپ شده است.

```
SVM Accuracy: 59.65%
SVM2 Accuracy: 45.61%
SVM3 Accuracy: 28.07%
Random Forest Accuracy: 56.14%
```

KNN Accuracy: 50.88%
KNN2 Accuracy: 43.86%
Naive Bayes Accuracy: 61.40%
Decision Tree Accuracy: 40.35%

Best Model: Naive Bayes with Accuracy: 61.40%

با توجه به پایین بودن دقت این مدل‌ها ۳ مدل با درصدهای بالاتر را برگزیدیم تا الگوریتم رای‌گیری برای آن‌ها اجرا شود این رای‌گیری براساس اکثریت بین مدل‌هایSVM ، Random Forestو Naive Bayes انجام شده است.

```
pred['vote'] = pred.apply(find_most_frequent, axis=1)
accuracy = accuracy_score(y_test, pred['vote'])
print(accuracy)
```

متاسفانه دقت به ۵۹/۶۴٪ کاهش یافت.
برای ارزیابی naïve bayes از روش cross validation استفاده کردیم. این الگوریتم با تعداد پنج قسمت (5-fold) انجام شده و میانگین دقت آن محاسبه شده است.

```
kfold = KFold(n_splits=5, shuffle=True, random_state=42)
cv_scores = []
for train_index, test_index in kfold.split(df):
print("Average Cross-Validation Accuracy:", np.mean(cv_scores))
```

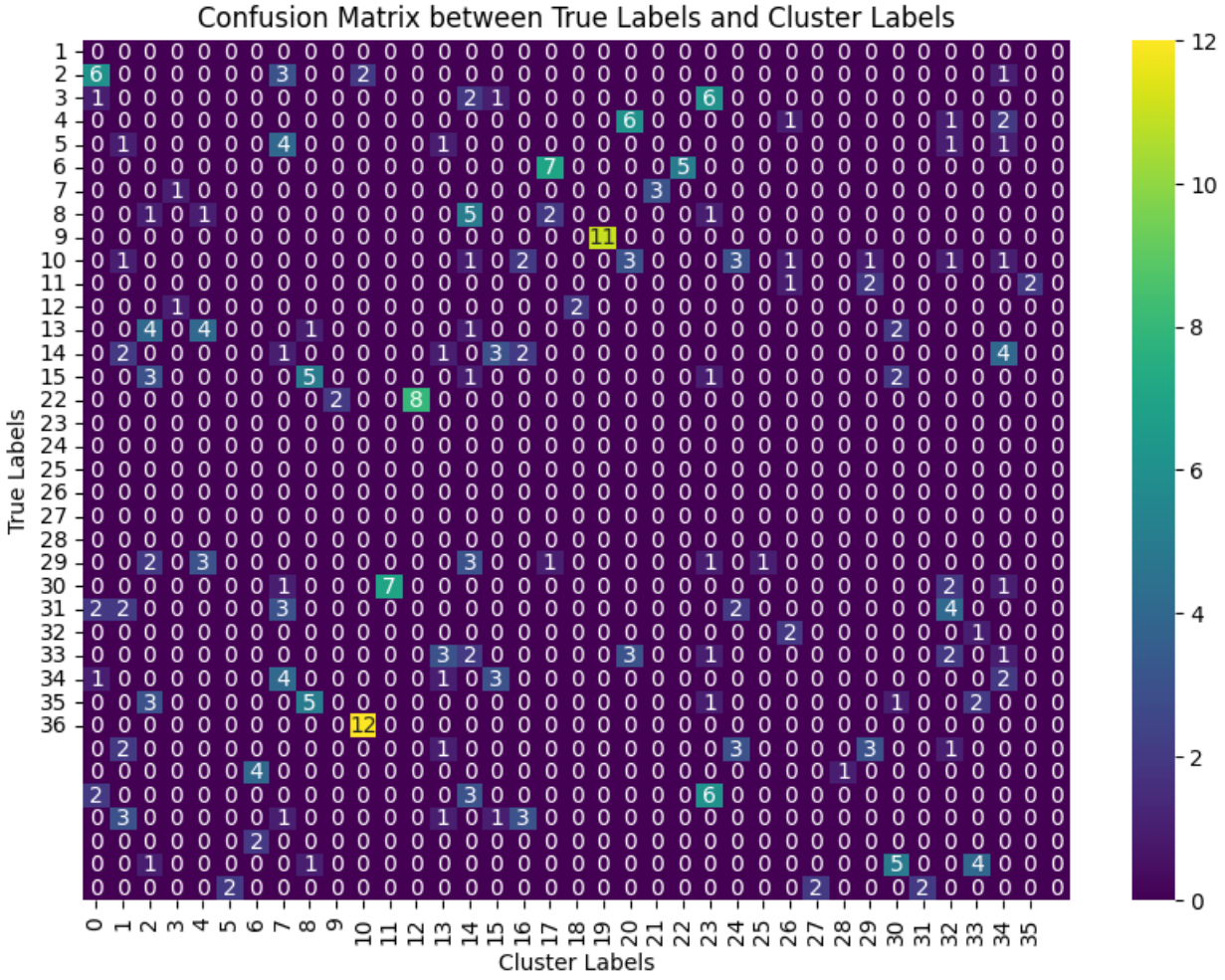
میانگین دقت ۶۴/۵۶٪ است.

برای یافتن ارتباط خوشه ها کد زیر را میزنیم

```
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

conf_matrix = confusion_matrix(labels, cluster_labels)

plt.figure(figsize=(10, 7))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='viridis',
xticklabels=np.unique(cluster_labels), yticklabels=np.unique(labels))
plt.xlabel('Cluster Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix between True Labels and Cluster Labels')
plt.show()
```


Confusion Matrix between True Labels and Cluster Labels
True Labels
Cluster Labels
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35
12
10
8
6
4
2
0