IRIS FLOWER CLASSIFICATION

P SAIKARTHIK

VU21CSEN0100398

1. Introduction

The Iris dataset is a well-known dataset in machine learning, primarily used for classification tasks. It consists of 150 samples of iris flowers, with each sample containing four features: sepal length, sepal width, petal length, and petal width. The dataset also includes a target variable, which is the species of the iris plant.

2. Data Preparation

Loading Data: The dataset is loaded using Pandas from a CSV file named 'IRIS.csv'.

Feature and Target Variable: The features (X) are the sepal length, sepal width, petal length, and petal width. The target variable (y) is the species of the iris.

Encoding Labels: The target variable 'species' is encoded into numerical values using LabelEncoder.

	A	B	C	D	E
1	sepal_leng	sepal_widt	petal_leng	petal_widt	species
2	5.1	3.5	1.4	0.2	Iris-setosa
3	4.9	3	1.4	0.2	Iris-setosa
4	4.7	3.2	1.3	0.2	Iris-setosa
5	4.6	3.1	1.5	0.2	Iris-setosa
6	5	3.6	1.4	0.2	Iris-setosa
7	5.4	3.9	1.7	0.4	Iris-setosa
8	4.6	3.4	1.4	0.3	Iris-setosa
9	5	3.4	1.5	0.2	Iris-setosa
10	4.4	2.9	1.4	0.2	Iris-setosa
11	4.9	3.1	1.5	0.1	Iris-setosa
12	5.4	3.7	1.5	0.2	Iris-setosa
13	4.8	3.4	1.6	0.2	Iris-setosa
14	4.8	3	1.4	0.1	Iris-setosa
15	4.3	3	1.1	0.1	Iris-setosa
16	5.8	4	1.2	0.2	Iris-setosa
17	5.7	4.4	1.5	0.4	Iris-setosa
18	5.4	3.9	1.3	0.4	Iris-setosa
19	5.1	3.5	1.4	0.3	Iris-setosa
20	5.7	3.8	1.7	0.3	Iris-setosa
21	5.1	3.8	1.5	0.3	Iris-setosa
22	5.4	3.4	1.7	0.2	Iris-setosa
23	5.1	3.7	1.5	0.4	Iris-setosa
24	4.6	3.6	1	0.2	Iris-setosa
25	5.1	3.3	1.7	0.5	Iris-setosa
26	4.8	3.4	1.9	0.2	Iris-setosa
27	5	3	1.6	0.2	Iris-setosa
28	5	3.4	1.6	0.4	Iris-setosa
29	5.2	3.5	1.5	0.2	Iris-setosa
30	5.2	3.4	1.4	0.2	Iris-setosa

3. Data Splitting

Training and Testing Split: The dataset is split into training and testing sets using an 80-20 ratio (train_test_split with test_size=0.2).

Feature Scaling: The features are scaled using StandardScaler to standardize the data, improving the performance of the Random Forest algorithm.

	A	B	C	D	E	F	G
1		sepal_leng	sepal_widt	petal_leng	petal_widt	species	
2	0	4.6	3.1	1.5	0.2	Iris-setosa	
3	1	4.6	3.4	1.4	0.3	Iris-setosa	
4	2	4.9	3.1	1.5	0.1	Iris-setosa	
5	3	4.8	3	1.4	0.1	Iris-setosa	
6	4	5.7	4.4	1.5	0.4	Iris-setosa	
7	5	5.7	3.8	1.7	0.3	Iris-setosa	
8	6	5.1	3.7	1.5	0.4	Iris-setosa	
9	7	4.8	3.4	1.9	0.2	Iris-setosa	
10	8	5.2	3.5	1.5	0.2	Iris-setosa	
11	9	4.8	3.1	1.6	0.2	Iris-setosa	
12	10	5.5	4.2	1.4	0.2	Iris-setosa	
13	11	5.5	3.5	1.3	0.2	Iris-setosa	
14	12	5.1	3.4	1.5	0.2	Iris-setosa	
15	13	4.4	3.2	1.3	0.2	Iris-setosa	
16	14	4.8	3	1.4	0.3	Iris-setosa	
17	15	5.3	3.7	1.5	0.2	Iris-setosa	
18	16	6.4	3.2	4.5	1.5	Iris-versicolor	
19	17	6.5	2.8	4.6	1.5	Iris-versicolor	
20	18	4.9	2.4	3.3	1	Iris-versicolor	
21	19	5	2	3.5	1	Iris-versicolor	
22	20	6.1	2.9	4.7	1.4	Iris-versicolor	
23	21	5.6	3	4.5	1.5	Iris-versicolor	
24	22	5.6	2.5	3.9	1.1	Iris-versicolor	
25	23	6.3	2.5	4.9	1.5	Iris-versicolor	
26	24	6.6	3	4.4	1.4	Iris-versicolor	
27	25	6	2.9	4.5	1.5	Iris-versicolor	
28	26	5.5	2.4	3.7	1	Iris-versicolor	
29	27	5.4	3	4.5	1.5	Iris-versicolor	
30	28	6.3	2.3	4.4	1.3	Iris-versicolor	

test_data

4. Model Training and Evaluation

Random Forest Classifier: A RandomForestClassifier with 100 trees is trained on the scaled training data.

Prediction: The model predicts the species of the test data.

Performance Metrics: The model's accuracy and detailed classification report are printed:

Accuracy: The accuracy of the model on the test set is calculated using accuracy_score.

Classification Report: The report includes precision, recall, and F1-score for each class, providing a detailed assessment of the model's performance.

5. Additional Data Processing

Extracting Test Data: A subset of the data is extracted and saved into a new CSV file 'test_data.csv'. This subset consists of every third sample from the original dataset, which includes all features and the species label.

6. Conclusion

The Random Forest model effectively classifies iris species with high accuracy. The standardization of features and the use of a Random Forest algorithm contribute to the model's robust performance.

IMPLEMENTATION

[16]:
```
import pandas as pd
from sklearn.preprocessing import LabelEncoder,StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report , accuracy_score
```

[2]:
```
df = pd.read_csv('IRIS.csv')
```

[3]:
```
df.head(1)
```

[3]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa

[6]:
```
y = df['species']
labelencoder = LabelEncoder()
y = labelencoder.fit_transform(y)

x = df.drop('species',axis=1)
```

[11]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=42)
```

[13]:
```
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)
```

[15]:
```
clr = RandomForestClassifier(n_estimators=100,random_state=42)
clr.fit(x_train,y_train)
```

[15]:
```
RandomForestClassifier
```

[17]:
```python
y_pred = clr.predict(x_test)
print("accuracy : ",accuracy_score(y_test,y_pred))
print(classification_report(y_test,y_pred))
```

```
accuracy :  1.0
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        10
           1       1.00      1.00      1.00         9
           2       1.00      1.00      1.00        11

    accuracy                           1.00        30
   macro avg       1.00      1.00      1.00        30
weighted avg       1.00      1.00      1.00        30
```

[19]:
```python
df = pd.read_csv('IRIS.csv')
df.head(1)
```

[19]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa

[20]:
```python
l = []
for i in range(3,len(df),3):
    l.append([df['sepal_length'][i],df['sepal_width'][i],df['petal_length'][i],df['petal_width'][i],df['species'][i]])
dff = pd.DataFrame(l,columns=['sepal_length','sepal_width','petal_length','petal_width','species'])
```

[21]:
```python
dff.to_csv('test_data.csv')
```

[22]:
```python
df = pd.read_csv('test_data.csv')
```

[40]:
```python
y = df['species']
```


[19]:
```python
df = pd.read_csv('IRIS.csv')
df.head(1)
```

[19]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa

[20]:
```python
l = []
for i in range(3,len(df),3):
    l.append([df['sepal_length'][i],df['sepal_width'][i],df['petal_length'][i],df['petal_width'][i],df['species'][i]])
dff = pd.DataFrame(l,columns=['sepal_length','sepal_width','petal_length','petal_width','species'])
```

[21]:
```python
dff.to_csv('test_data.csv')
```

[22]:
```python
df = pd.read_csv('test_data.csv')
```


[40]:
```python
y = df['species']
x = df.drop('species',axis=1)
```

[43]:
```python
x= x.drop('Unnamed: 0',axis=1)
```

[45]:
```python
x = scaler.transform(x)
```

[48]:
```python
y_tes = clr.predict(x)
```

[51]:
```python
y = labelencoder.fit_transform(y)
print("accuracy :",accuracy_score(y,y_tes)*100)
```

```
accuracy : 100.0
```

7. Code Summary

python

Copy code

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, accuracy_score

# Load and prepare data
df = pd.read_csv('IRIS.csv')
y = df['species']
labelencoder = LabelEncoder()
y = labelencoder.fit_transform(y)
x = df.drop('species', axis=1)

# Split data
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)
scaler = StandardScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)

# Train model
clr = RandomForestClassifier(n_estimators=100, random_state=42)
clr.fit(x_train, y_train)
```

```
y_pred = clr.predict(x_test)


# Evaluate model
print("accuracy : ", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))


# Save a subset of data
l = []
for i in range(3, len(df), 3):
    l.append([df['sepal_length'][i], df['sepal_width'][i], df['petal_length'][i], df['petal_width'][i], df['species'][i]])
dff = pd.DataFrame(l, columns=['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'species'])
dff.to_csv('test_data.csv')
```

OUTPUT

```
[51]: y = labelencoder.fit_transform(y)
      print("accuracy :",accuracy_score(y,y_tes)*100)

      accuracy : 100.0
```

🌐 Sources

geeksforgeeks.org - Random Forest Classifier using Scikit-learn

kaggle.com - IRIS Classification with Machine Learning: Basics

datacamp.com - Random Forest Classification with Scikit-Learn

scikit-learn.org - RandomForestClassifier

scikit-learn.org - load_iris

embedded-robotics.com - Iris Dataset Classification Using 3 Machine Learning Algos