0.1 IrisFlowersClassificationMachineLearningProject

0.2 ProjectOverview

This project is designed to classify Iris flowers into three species: Setosa, Versicolor, and Virginica. The dataset used is the Iris dataset, which is well-known in the machine learning community for classification problems.

0.3 Dataset

The Iris dataset can be fetched from the UCI Machine Learning Repository. It contains measurements of flower petals and sepals, which are used to classify the flowers.

0.3.1 1. ImportNecessaryLibraries

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report,␣
↪confusion_matrix
```

0.3.2 2. Load the Iris Dataset

[2]:
```
from ucimlrepo import fetch_ucirepo

# Fetch dataset
iris = fetch_ucirepo(id=53)
# Data (as pandas dataframes)
X = iris.data.features
y = iris.data.targets
# Metadata
print(iris.metadata)
```

```
# Variable information
print(iris.variables)
```

{'uci_id': 53, 'name': 'Iris', 'repository_url': 'https://archive.ics.uci.edu/dataset/53/iris', 'data_url': 'https://archive.ics.uci.edu/static/public/53/data.csv', 'abstract': 'A small classic dataset from Fisher, 1936. One of the earliest known datasets used for evaluating classification methods.\n', 'area': 'Biology', 'tasks': ['Classification'], 'characteristics': ['Tabular'], 'num_instances': 150, 'num_features': 4, 'feature_types': ['Real'], 'demographics': [], 'target_col': ['class'], 'index_col': None, 'has_missing_values': 'no', 'missing_values_symbol': None, 'year_of_dataset_creation': 1936, 'last_updated': 'Tue Sep 12 2023', 'dataset_doi': '10.24432/C56C76', 'creators': ['R. A. Fisher'], 'intro_paper': {'title': 'The Iris data set: In search of the source of virginica', 'authors': 'A. Unwin, K. Kleinman', 'published_in': 'Significance, 2021', 'year': 2021, 'url': 'https://www.semanticscholar.org/paper/4599862ea877863669a6a8e63a3c707a787d5d7e', 'doi': '1740-9713.01589'}, 'additional_info': {'summary': 'This is one of the earliest datasets used in the literature on classification methods and widely used in statistics and machine learning. The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are not linearly separable from each other.\n\nPredicted attribute: class of iris plant.\n\nThis is an exceedingly simple domain.\n\nThis data differs from the data presented in Fishers article (identified by Steve Chadwick, spchadwick@espeedaz.net) The 35th sample should be: 4.9,3.1,1.5,0.2,"Iris-setosa" where the error is in the fourth feature. The 38th sample: 4.9,3.6,1.4,0.1,"Iris-setosa" where the errors are in the second and third features. ', 'purpose': 'N/A', 'funded_by': None, 'instances_represent': 'Each instance is a plant', 'recommended_data_splits': None, 'sensitive_data': None, 'preprocessing_description': None, 'variable_info': None, 'citation': None}}

	name	role	type	demographic	description	units	missing_values
0	sepallength	Feature	Continuous	None	None	cm	no
1	sepalwidth	Feature	Continuous	None	None	cm	no
2	petallength	Feature	Continuous	None	None	cm	no
3	petalwidth	Feature	Continuous	None	None	cm	no
4	class	Target	Categorical	None	classof iris plant:Iris Setosa, Iris Versico...	None	no

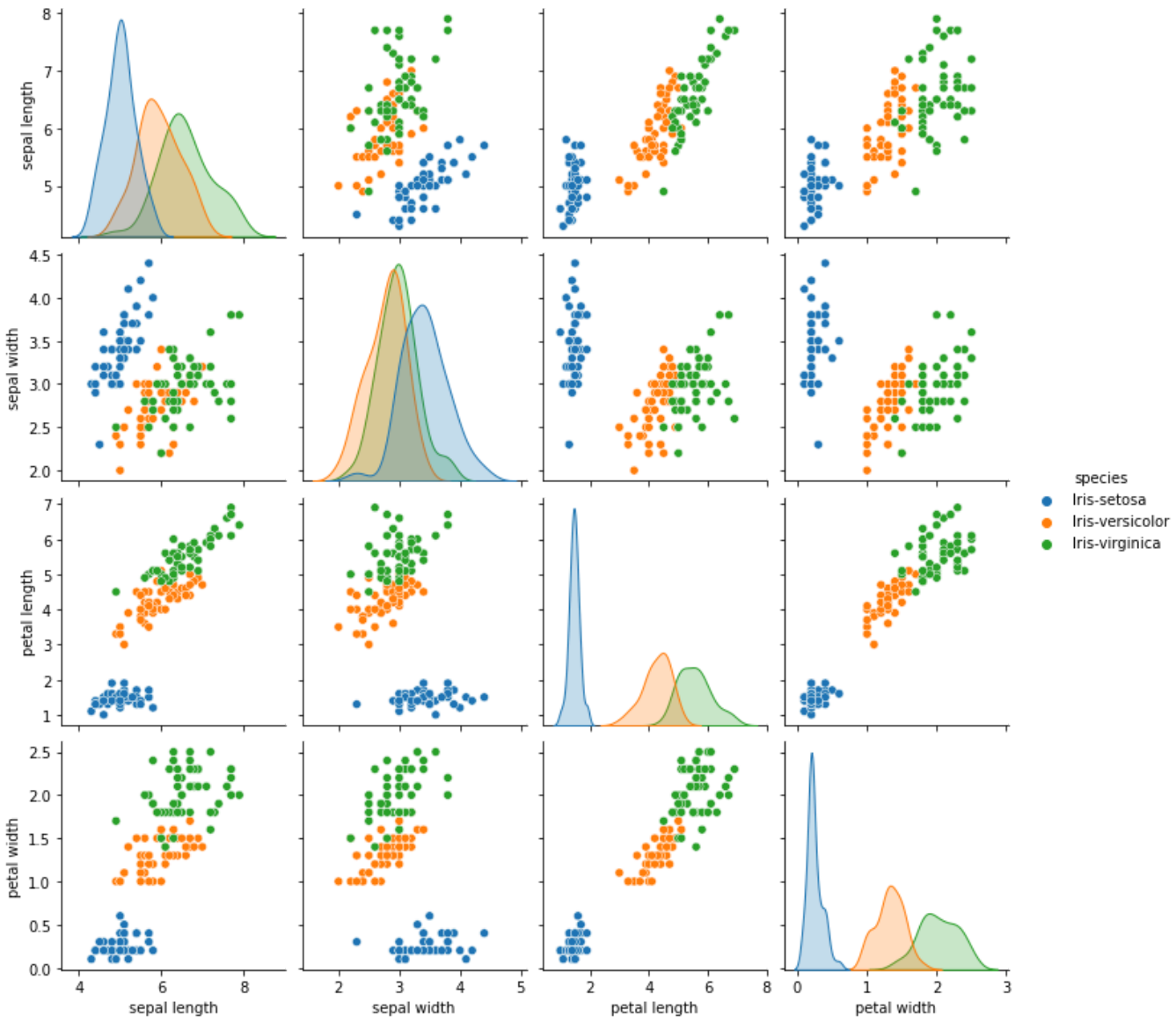
0.3.3 3. Exploratory Data Analysis (EDA)

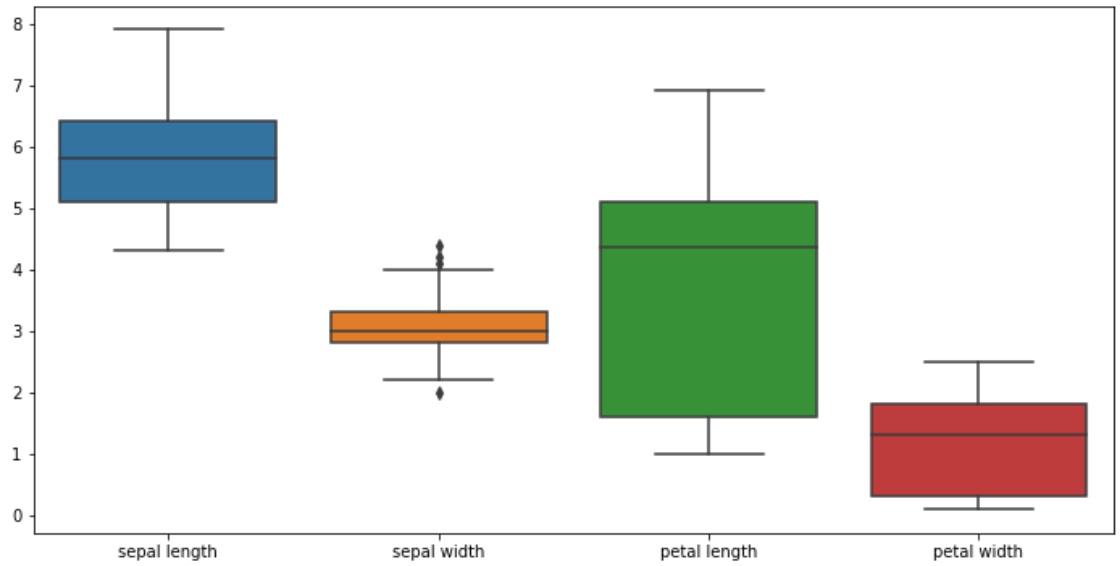
[3]:
```
print(X.head())
 print(y.head())


print(X.describe())


print(X.isnull().sum())
```

```
   sepallength  sepal width  petal length  petalwidth
0          5.1          3.5           1.4         0.2
1          4.9          3.0           1.4         0.2
2          4.7          3.2           1.3         0.2
3          4.6          3.1           1.5         0.2
4          5.0          3.6           1.4         0.2
0
1
2
3
4
Name: class, dtype: object
       sepal length  sepal width  petal length  petal width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.054000      3.758667     1.198667
std        0.828066     0.433594      1.764420     0.763161
min        4.300000     2.000000      1.000000     0.100000
25%        5.100000     2.800000      1.600000     0.300000
50%        5.800000     3.000000      4.350000     1.300000
75%        6.400000     3.300000      5.100000     1.800000
max        7.900000     4.400000      6.900000     2.500000
sepal length    0
sepal width     0
petal length    0
petal width     0
dtype: int64
```

[4]:
```
import seaborn as sns
import matplotlib.pyplot as plt

iris_data = X.copy()
iris_data['species'] = y
s
sns.pairplot(iris_data, hue='species')
plt.show()
```

```
plt.figure(figsize=(12, 6))
sns.boxplot(data=iris_data.drop('species', axis=1))
plt.show()
```





0.3.4 4. Data Preprocessing

[14]:
```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
X = iris.data.features
y = iris.data.targets
y = y.squeeze()
y = y.astype('category').cat.codes
# Ensure y is a 1D array
y = y.ravel()
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,␣
↪random_state=42)
```

0.3.5 5. Feature Scaling

[15]:
```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

0.3.6 6. Train the Model

[16]:
```
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(X_train_scaled, y_train)
```

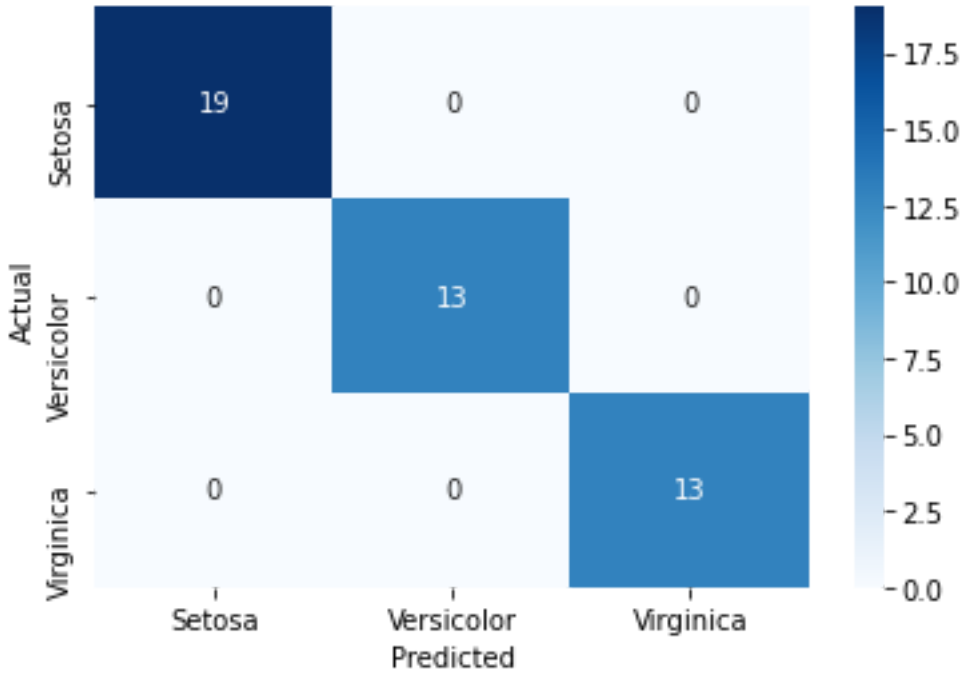
[16]:
```
KNeighborsClassifier()
```

0.3.7 7. Evaluate the Model

[11]:
```
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix


y_pred = knn.predict(X_test_scaled)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy:.2f}')

# Classification report
print(classification_report(y_test, y_pred))
# Confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)
sns.heatmap(conf_matrix, annot=True, cmap='Blues', fmt='d', xticklabels=['Setosa', 'Versicolor', 'Virginica'], yticklabels=['Setosa', 'Versicolor','Virginica'])plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```

Accuracy: 1.00

	precision	recall	f1-score	support
Iris-setosa	1.00	1.00	1.00	19
Iris-versicolor	1.00	1.00	1.00	13
Iris-virginica	1.00	1.00	1.00	13
accuracy			1.00	45
macroavg	1.00	1.00	1.00	45
weightedavg	1.00	1.00	1.00	45



Observation: Machine learning model achieved perfect classification on the Iris dataset. With precision, recall, and F1-scores all at 1.00, it flawlessly predicted the three classes: Iris-setosa, Iris-versicolor, and Iris-virginica. The support values—19 for Iris-setosa and 13 for each of the other two classes—highlight the robustness of the mo del.