8. IMPLEMENTATION OF DIFFERENT CLASSIFICATION TECHNIQUES

Contents

Introduction:

Certainly! Implementing different classification techniques involves using various algorithms to train models and make predictions on datasets. Below are examples of implementing three popular classification techniques using Python and scikit-learn:

1. Decision Tree Classifier:

~~~
```python
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report

# Load the Iris dataset
iris = load_iris()
X_train, X_test, y_train, y_test = train_test_split(iris.data, iris.target, test_size=0.2, random_state=42)

# Create a Decision Tree Classifier
clf = DecisionTreeClassifier()

# Train the classifier
clf.fit(X_train, y_train)

# Make predictions on the test set
y_pred = clf.predict(X_test)

# Evaluate the performance
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.2f}")

# Display classification report
print("Classification Report:\n", classification_report(y_test, y_pred))
~~~

~~~
```
~~~

2. Support Vector Machine (SVM) Classifier:

```python
from sklearn.svm import SVC
from sklearn.preprocessing import StandardScaler

# Standardize the feature matrix
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Create an SVM Classifier
svm_clf = SVC(kernel='linear')

# Train the classifier
svm_clf.fit(X_train_scaled, y_train)

# Make predictions on the scaled test set
y_pred_svm = svm_clf.predict(X_test_scaled)

# Evaluate the performance
accuracy_svm = accuracy_score(y_test, y_pred_svm)
print(f"SVM Accuracy: {accuracy_svm:.2f}")

# Display classification report
print("SVM Classification Report:\n", classification_report(y_test, y_pred_svm))
```

3. Random Forest Classifier:

```python
from sklearn.ensemble import RandomForestClassifier


# Create a Random Forest Classifier
rf_clf = RandomForestClassifier(n_estimators=100, random_state=42)


# Train the classifier
rf_clf.fit(X_train, y_train)


# Make predictions on the test set
y_pred_rf = rf_clf.predict(X_test)


# Evaluate the performance
accuracy_rf = accuracy_score(y_test, y_pred_rf)
print(f"Random Forest Accuracy: {accuracy_rf:.2f}")


# Display classification report
print("Random Forest Classification Report:\n", classification_report(y_test, y_pred_rf))
```

These examples cover three different classification algorithms: Decision Tree, Support Vector Machine (SVM), and Random Forest. Depending on your dataset and problem, you may choose the most suitable algorithm and fine-tune its parameters for better performance. It's important to note that these are basic examples, and in real-world scenarios, you might need to perform more extensive preprocessing, hyperparameter tuning, and handle class imbalances or other challenges specific to your data.