sl-support-vector-mechanism-1

August 26, 2023

###Name : J.Keerthana ###Roll No : 21X05A6721 ###Branch : Data Science ###College : Narasimha Reddy Enginnering College

##Project Title: ###Using the Support Vector Mechanism Algorithm of supervised machine learning predict iris.csv datasets to find out species will be same

0.1 Problem Statement:

A American waist botnical garden grow iris flower in their labs but using bio technology in a single tree different types of variety flower is grow. As a Data Science Engineer find out how much accuracy is their all categories contains same species.

###Task 1: ### Preprocess the data in skit.learn library. ###Task 2: ### Load the data using sklearn model selection defaukt argument. ###Task 3: ### On the basis of your dataset train,test and split SVM model. ###Task 4: ### Implement the Support Vector Mechanism Classifier using svm_classifier. the svm must be "Linear". ###Task 5: ### Train the classifier on the training data. ###Task 6: ### Find out the prediction value on the test data ###Task 7: ### Test the model with the help of accuracy ,accuracy should be lie in the range 0-1

0.2 Conclusion:

0.2.1 According to my support vector mechanism model the species or linear. With the accuracy of 1.00.

0.2.2 Hence proved model was successfully implement

[3]:

```
# Load the Iris dataset
iris = load_iris()
X = iris.data
y = iris.target
```

[2]:

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
```

[4]:

```
# Consider only two classes for simplicity
X = X[y != 2]
y = y[y != 2]
```

[5]:
```
# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[6]:
```
# Create an SVM classifier
svm_classifier = SVC(kernel='linear')
```

[7]:
```
# Train the classifier on the training data
svm_classifier.fit(X_train, y_train)
```

[7]:
```
SVC(kernel='linear')
```

[8]:
```
# Make predictions on the test data
y_pred = svm_classifier.predict(X_test)
```

[9]:
```
# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.2f}")
```

```
Accuracy: 1.00
```

[9]:
```

```