**11.Create a sample dataset, having linearly separable data, from sklearn.dataset.sample\_generator for classification using SVM**

**SOURCE CODE:**

**# Import the necessary libraries**

from sklearn.datasets import make\_classification

from sklearn.model\_selection import train\_test\_split

from sklearn.svm import SVC

from sklearn.metrics import accuracy\_score

# Create a sample dataset with 500 samples and two features

X, y = make\_classification(n\_samples=500, n\_features=2, n\_informative=2, n\_redundant=0, n\_clusters\_per\_class=1, random\_state=0)

# Split the dataset into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=0)

# Create the SVM and train the model

svm = SVC(kernel='linear')

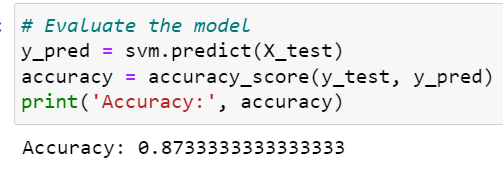
svm.fit(X\_train, y\_train)

# Evaluate the model

y\_pred = svm.predict(X\_test)

accuracy = accuracy\_score(y\_test, y\_pred)

print('Accuracy:', accuracy)SVC(kernel='linear')



**OBSERVATION:**

* we use the make\_classification function from the scikit-learn library to create a sample dataset with 500 samples and two features. We set the n\_informative parameter to 2 to ensure that the data is linearly separable. We also set n\_redundant to 0 to avoid creating redundant features.
* We then split the dataset into training and testing sets using the train\_test\_split function. In this example, we use 30% of the data for testing.
* We create an SVM model with a linear kernel and train the model using the fit() function and passing the training data and labels.
* Finally, we evaluate the performance of the SVM model on the testing set using the predict() function and calculating the accuracy using the accuracy\_score function from scikit-learn.
* **Accuracy: 0.8733333333333333=87.33%**