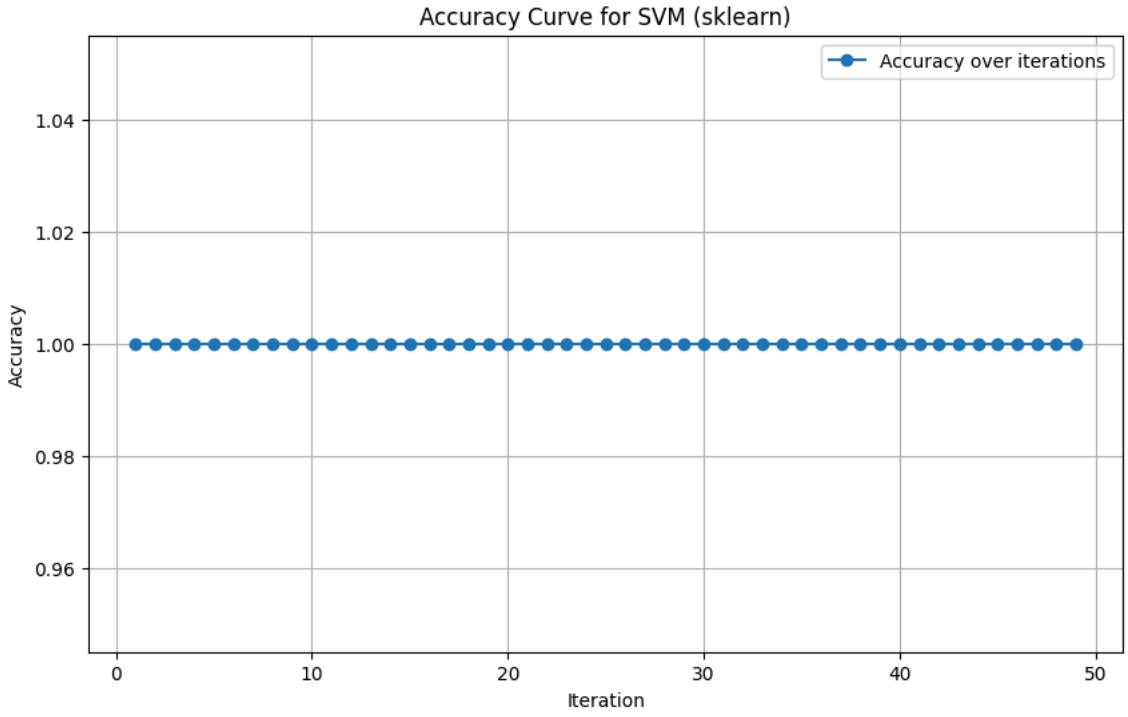
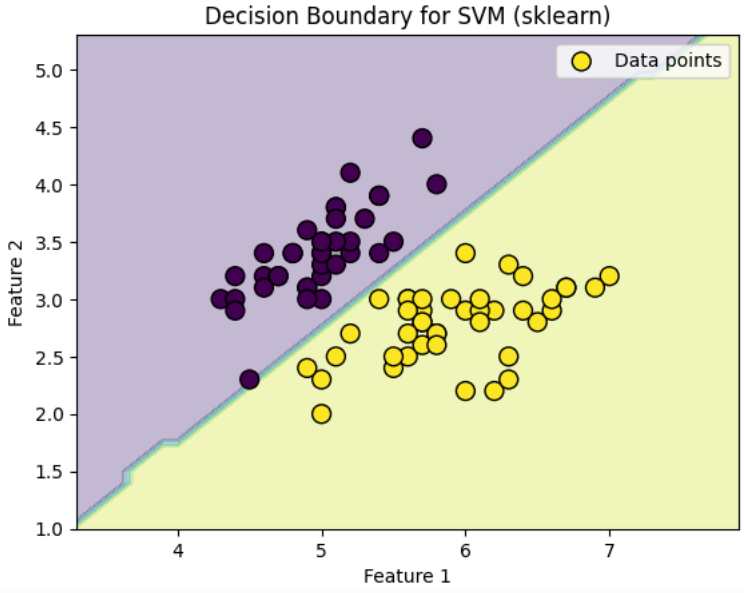
```
#svm considering the data with linear classification
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
iris = datasets.load_iris()
X = iris.data[:100, :2]
y = iris.target[:100]



X_train, X_test,y_train,y_test = train_test_split(X,y,test_size = 0.2,random_state=42)

svm_classifier = SVC()
accuracies = []
for i in range(1, 50):
    X_train_split, _, y_train_split, _ = train_test_split(X_train, y_train, test_size=0.2, random_state=i)
    svm_classifier.fit(X_train_split, y_train_split)
    y_pred = svm_classifier.predict(X_test)
    acc = accuracy_score(y_test, y_pred)
    accuracies.append(acc)

plt.figure(figsize=(10, 6))
plt.plot(range(1, 50), accuracies, marker='o', label="Accuracy over iterations")
plt.title("Accuracy Curve for SVM (sklearn)")
plt.xlabel("Iteration")
plt.ylabel("Accuracy")
plt.grid()
plt.legend()
plt.show()

def plot_decision_boundary(X,y,model):
  X_min,X_max = X[:,0].min()-1,X[:,0].max()+1
  Y_min,Y_max = X[:,1].min()-1,X[:,1].max()+1
  xx, yy = np.meshgrid(np.arange(X_min,X_max,0.1),np.arange(Y_min,Y_max,0.1))
  Z = model.predict(np.c_[xx.ravel(),yy.ravel()])
  Z = Z.reshape(xx.shape)

  plt.contourf(xx, yy, Z, alpha=0.3)
  plt.scatter(X[:, 0], X[:, 1], c=y, marker='o', s=100, edgecolor='k', label='Data points')
  plt.title("Decision Boundary for SVM (sklearn)")
  plt.xlabel("Feature 1")
  plt.ylabel("Feature 2")
  plt.legend()
  plt.show()

plot_decision_boundary(X_train, y_train, svm_classifier)
```




```
X_train, X_test,y_train,y_test = train_test_split(X,y,test_size = 0.2,random_state=42)

svm_classifier = SVC(kernel='rbf')
accuracies = []
for i in range(1, 50):
    X_train_split, _, y_train_split, _ = train_test_split(X_train, y_train, test_size=0.2, random_state=i
    svm_classifier.fit(X_train_split, y_train_split)
    y_pred = svm_classifier.predict(X_test)
    acc = accuracy_score(y_test, y_pred)
    accuracies.append(acc)

plt.figure(figsize=(10, 6))
plt.plot(range(1, 50), accuracies, marker='o', label="Accuracy over iterations")
plt.title("Accuracy Curve for SVM (sklearn)")
plt.xlabel("Iteration")
plt.ylabel("Accuracy")
plt.grid()
plt.legend()
plt.show()

def plot_decision_boundary(X,y,model):
  X_min,X_max = X[:,0].min()-1,X[:,0].max()+1
  Y_min,Y_max = X[:,1].min()-1,X[:,1].max()+1
  xx, yy = np.meshgrid(np.arange(X_min,X_max,0.1),np.arange(Y_min,Y_max,0.1))
  Z = model.predict(np.c_[xx.ravel(),yy.ravel()])
  Z = Z.reshape(xx.shape)
```

```
plt.contourf(xx, yy, Z, alpha=0.3)
plt.scatter(X[:, 0], X[:, 1], c=y, marker='o', s=100, edgecolor='k', label='Data points')
plt.title("Decision Boundary for SVM (sklearn)")
plt.xlabel("Feature 1")
plt.ylabel("Feature 2")
plt.legend()
plt.show()
```