SupportVectorMachines

May 28, 2026

[4]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, confusion_matrix,␣
↪classification_report
```

[5]:
```
data = load_breast_cancer()
X = pd.DataFrame(data.data, columns=data.feature_names)
y = data.target

print("Shape:", X.shape)
print("Target classes:", np.unique(y))
```

```
Shape: (569, 30)
Target classes: [0 1]
```

[6]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42, stratify=y)

scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

[7]:
```
svm_model = SVC(kernel="rbf", C=2.0, gamma="scale", random_state=42)
svm_model.fit(X_train_scaled, y_train)
```

[7]:
```
SVC(C=2.0, random_state=42)
```

[8]:
```
y_pred = svm_model.predict(X_test_scaled)

acc = accuracy_score(y_test, y_pred)
cm = confusion_matrix(y_test, y_pred)
```

```
print("Accuracy:", round(acc, 4))
print("Confusion Matrix:\n", cm)
print("Classification Report:\n", classification_report(y_test, y_pred))
```

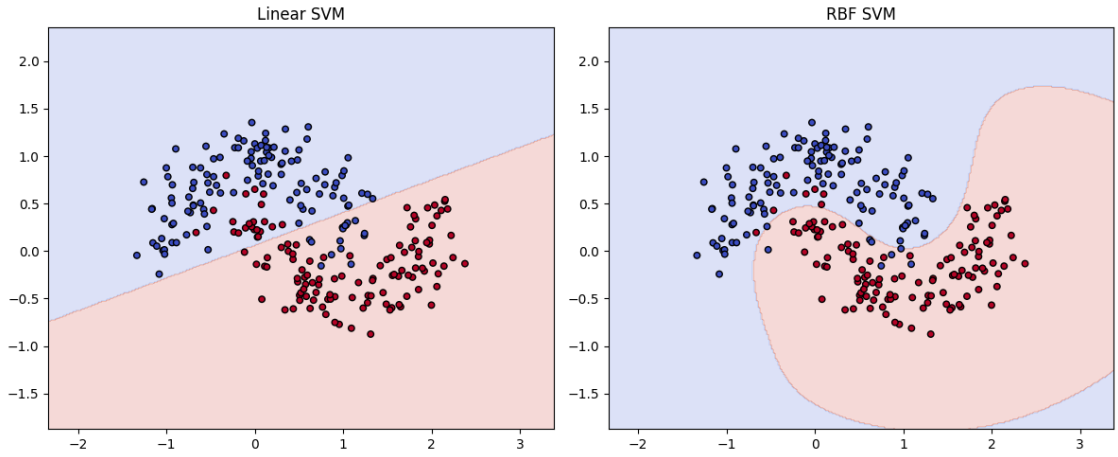
```
Accuracy: 0.9825
Confusion Matrix:
 [[41  1]
 [ 1 71]]
Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.98      0.98        42
           1       0.99      0.99      0.99        72

    accuracy                           0.98       114
   macro avg       0.98      0.98      0.98       114
weighted avg       0.98      0.98      0.98       114
```

[9]:
```
for c_val in [0.5, 1.0, 2.0, 5.0]:
    model = SVC(kernel="rbf", C=c_val, gamma="scale", random_state=42)
    model.fit(X_train_scaled, y_train)
    pred = model.predict(X_test_scaled)
    print(f"C={c_val}: accuracy={accuracy_score(y_test, pred):.4f}")
```

```
C=0.5: accuracy=0.9737
C=1.0: accuracy=0.9825
C=2.0: accuracy=0.9825
C=5.0: accuracy=0.9825
```

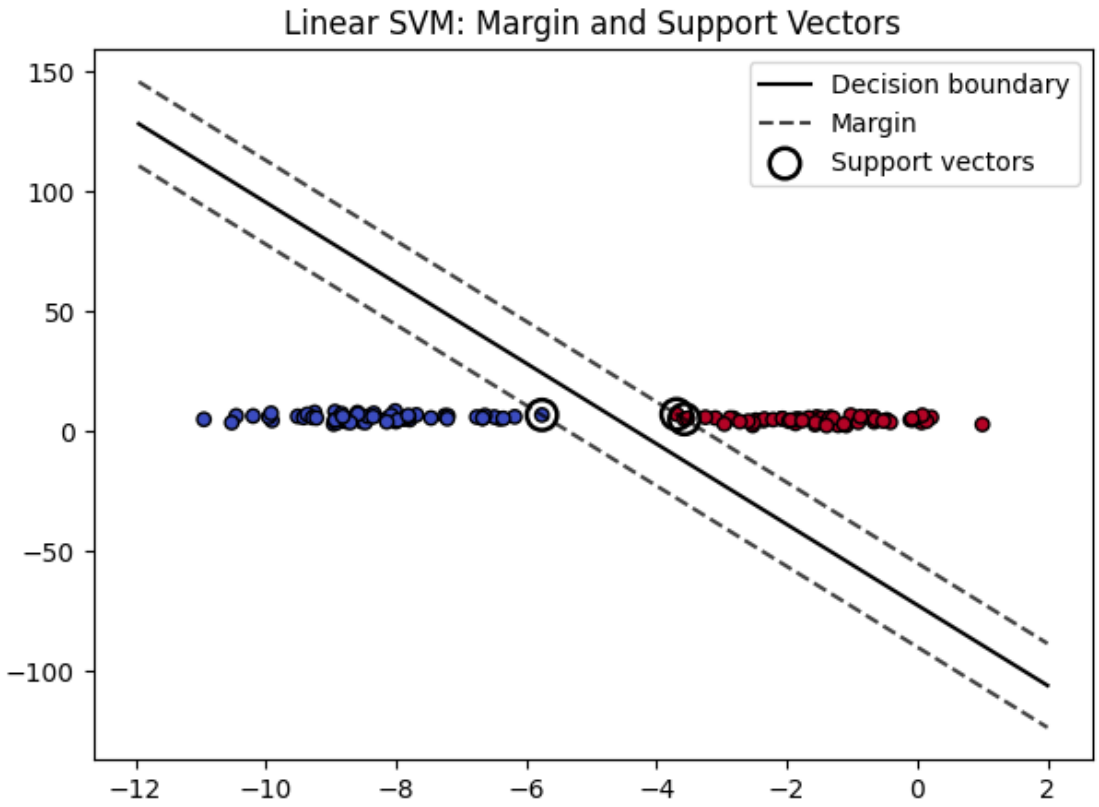
[10]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import make_moons
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import make_pipeline
from sklearn.svm import SVC

X2, y2 = make_moons(n_samples=250, noise=0.20, random_state=42)

def plot_boundary(ax, model, X, y, title):
    model.fit(X, y)
    xx, yy = np.meshgrid(
        np.linspace(X[:, 0].min() - 1, X[:, 0].max() + 1, 300),
        np.linspace(X[:, 1].min() - 1, X[:, 1].max() + 1, 300)
    )
    Z = model.predict(np.c_[xx.ravel(), yy.ravel()]).reshape(xx.shape)
    ax.contourf(xx, yy, Z, alpha=0.20, cmap="coolwarm")
    ax.scatter(X[:, 0], X[:, 1], c=y, cmap="coolwarm", s=22, edgecolor="k")
```

```
    ax.set_title(title)

fig, axes = plt.subplots(1, 2, figsize=(12, 5))
lin = make_pipeline(StandardScaler(), SVC(kernel="linear", C=1.0))
rbf = make_pipeline(StandardScaler(), SVC(kernel="rbf", C=1.0, gamma="scale"))

plot_boundary(axes[0], lin, X2, y2, "Linear SVM")
plot_boundary(axes[1], rbf, X2, y2, "RBF SVM")
plt.tight_layout()
plt.show()
```



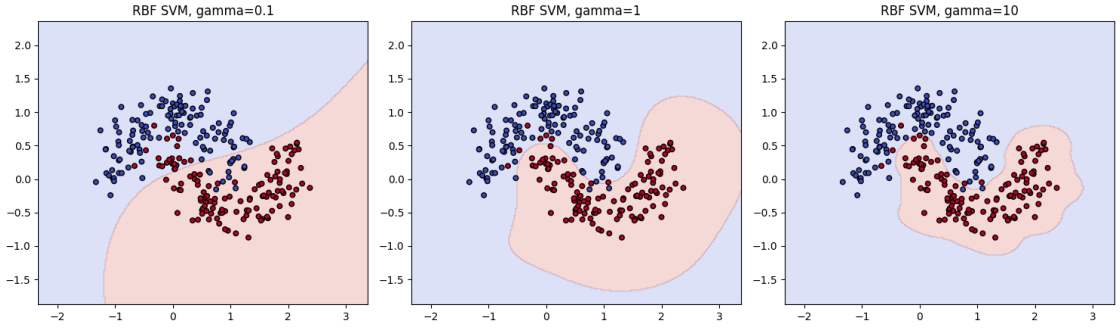
[11]:
```
from sklearn.datasets import make_blobs
from sklearn.svm import SVC

Xb, yb = make_blobs(n_samples=120, centers=2, cluster_std=1.2, random_state=7)
svc_lin = SVC(kernel="linear", C=1.0)
svc_lin.fit(Xb, yb)

w = svc_lin.coef_[0]
b = svc_lin.intercept_[0]
xx = np.linspace(Xb[:, 0].min() - 1, Xb[:, 0].max() + 1, 200)
yy = -(w[0] * xx + b) / w[1]
yy_down = -(w[0] * xx + b - 1) / w[1]
yy_up = -(w[0] * xx + b + 1) / w[1]

plt.figure(figsize=(7, 5))
plt.scatter(Xb[:, 0], Xb[:, 1], c=yb, cmap="coolwarm", s=28, edgecolor="k")
plt.plot(xx, yy, "k-", label="Decision boundary")
plt.plot(xx, yy_down, "k--", alpha=0.75, label="Margin")
plt.plot(xx, yy_up, "k--", alpha=0.75)
plt.scatter(
```

```
    svc_lin.support_vectors_[:, 0],
    svc_lin.support_vectors_[:, 1],
    s=140, facecolors="none", edgecolors="black", linewidth=1.8, label="Support␣
 ↪vectors"
)
plt.title("Linear SVM: Margin and Support Vectors")
plt.legend()
plt.show()
```



[12]:
```
gamma_values = [0.1, 1, 10]
fig, axes = plt.subplots(1, 3, figsize=(15, 4.5))

for i, gval in enumerate(gamma_values):
    model = make_pipeline(StandardScaler(), SVC(kernel="rbf", C=1.0,␣
 ↪gamma=gval))
    model.fit(X2, y2)

    xx, yy = np.meshgrid(
        np.linspace(X2[:, 0].min() - 1, X2[:, 0].max() + 1, 250),
        np.linspace(X2[:, 1].min() - 1, X2[:, 1].max() + 1, 250)
```

```
    )
    Z = model.predict(np.c_[xx.ravel(), yy.ravel()]).reshape(xx.shape)

    axes[i].contourf(xx, yy, Z, alpha=0.20, cmap="coolwarm")
    axes[i].scatter(X2[:, 0], X2[:, 1], c=y2, cmap="coolwarm", s=22,␣
 ↪edgecolor="k")
    axes[i].set_title(f"RBF SVM, gamma={gval}")

plt.tight_layout()
plt.show()
```



[ ]: