

NAME : S DIVYA

REG NO.: 22MID0355

Task 7:

CODE:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split, GridSearchCV, cross_val_score
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.metrics import classification_report, confusion_matrix

df = pd.read_csv('breast-cancer.csv')
print(df.head())

# Step 2: Preprocess Data
# Drop any unnamed columns if present
df = df.loc[:, ~df.columns.str.contains('^Unnamed')]

# Check target column - assume it's named "diagnosis" or similar
if 'diagnosis' in df.columns:
    df['diagnosis'] = df['diagnosis'].map({'M': 1, 'B': 0}) # Malignant = 1, Benign = 0

# Split features and target
X = df.drop('diagnosis', axis=1)
y = df['diagnosis']

# Normalize features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Train/test split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
```

```
# Step 3: Train SVM (Linear)
svm_linear = SVC(kernel='linear', C=1)
svm_linear.fit(X_train, y_train)
print("Linear SVM Accuracy:", svm_linear.score(X_test, y_test))

# Step 4: Train SVM (RBF Kernel)
svm_rbf = SVC(kernel='rbf', C=1, gamma='scale')
svm_rbf.fit(X_train, y_train)
print("RBF SVM Accuracy:", svm_rbf.score(X_test, y_test))

# Step 5: Visualization using PCA (2D projection)
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

# Retrain on PCA-reduced data for visualization
X_train_pca, X_test_pca, y_train_pca, y_test_pca = train_test_split(X_pca, y, test_size=0.2, random_state=42)
svm_vis = SVC(kernel='rbf', C=1, gamma='scale')
svm_vis.fit(X_train_pca, y_train_pca)

# Plot decision boundary
def plot_decision_boundary(X, y, model):
    h = .02
    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, h),
                        np.arange(y_min, y_max, h))
    Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)
```

```

plt.contourf(xx, yy, Z, cmap=plt.cm.coolwarm, alpha=0.8)
plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.coolwarm, edgecolors='k')
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.title('SVM Decision Boundary (RBF Kernel)')
plt.show()

plot_decision_boundary(X_test_pca, y_test_pca, svm_vis)

# Step 6: Hyperparameter Tuning with GridSearchCV
param_grid = {
    'C': [0.1, 1, 10],
    'gamma': ['scale', 0.1, 1, 10],
    'kernel': ['rbf']
}

grid = GridSearchCV(SVC(), param_grid, refit=True, verbose=2, cv=5)
grid.fit(X_train, y_train)

print("Best parameters:", grid.best_params_)
print("Best estimator accuracy:", grid.best_estimator_.score(X_test, y_test))

# Step 7: Cross-Validation
cv_scores = cross_val_score(grid.best_estimator_, X_scaled, y, cv=5)
print("Cross-validation scores:", cv_scores)
print("Mean CV accuracy:", cv_scores.mean())

```

OUTPUT:

```

⇒ 0      id diagnosis  radius_mean  texture_mean  perimeter_mean  area_mean  \
1      842302        M      17.99      10.38      122.80      1001.0
2      842517        M      20.57      17.77      132.90      1326.0
3      84300903       M      19.69      21.25      130.00      1203.0
4      84348301        M      11.42      20.38      77.58      386.1
5      84358402        M      20.29      14.34      135.10      1297.0

      smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
0      0.11840      0.27760      0.3001      0.14710
1      0.08474      0.07864      0.0869      0.07017
2      0.10960      0.15990      0.1974      0.12790
3      0.14250      0.28390      0.2414      0.10520
4      0.10030      0.13280      0.1980      0.10430

      ... radius_worst  texture_worst  perimeter_worst  area_worst  \
0      ...      25.38      17.33      184.60      2019.0
1      ...      24.99      23.41      158.80      1956.0
2      ...      23.57      25.53      152.50      1709.0
3      ...      14.91      26.50      98.87      567.7
4      ...      22.54      16.67      152.20      1575.0

      smoothness_worst  compactness_worst  concavity_worst  concave points_worst  \
0      0.1622      0.6656      0.7119      0.2654
1      0.1238      0.1866      0.2416      0.1860
2      0.1444      0.4245      0.4504      0.2430
3      0.2098      0.8663      0.6869      0.2575
4      0.1374      0.2050      0.4000      0.1625

      symmetry_worst  fractal_dimension_worst
0      0.4601      0.11890
1      0.2750      0.08902
2      0.3613      0.08758
3      0.6638      0.17300
4      0.2364      0.07678

[5 rows x 32 columns]
Linear SVM Accuracy: 0.956140350877193
RBF SVM Accuracy: 0.9824561403508771

```



```
[CV] END .....C=1, gamma=1, kernel=rbf; total time= 0.0s
[CV] END .....C=1, gamma=1, kernel=rbf; total time= 0.0s
[CV] END .....C=1, gamma=1, kernel=rbf; total time= 0.0s
[CV] END .....C=1, gamma=10, kernel=rbf; total time= 0.0s
[CV] END .....C=1, gamma=10, kernel=rbf; total time= 0.0s
[CV] END .....C=1, gamma=10, kernel=rbf; total time= 0.0s
[CV] END .....C=1, gamma=10, kernel=rbf; total time= 0.0s
[CV] END .....C=1, gamma=10, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=scale, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=scale, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=scale, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=scale, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=scale, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=0.1, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=0.1, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=0.1, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=0.1, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=1, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=1, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=1, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=1, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=10, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=10, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=10, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=10, kernel=rbf; total time= 0.0s
[CV] END .....C=10, gamma=10, kernel=rbf; total time= 0.0s
```

Best parameters: {'C': 1, 'gamma': 'scale', 'kernel': 'rbf'}

Best estimator accuracy: 0.9824561403508771

Cross-validation scores: [0.97368421 0.95614035 1. 0.96491228 0.97345133]

Mean CV accuracy: 0.9736376339077782