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

# 1. Load external breast cancer dataset
df = pd.read_csv('/content/task4-data.csv')

# Drop ID column and unnamed column if present
df = df.drop(columns=['id'], errors='ignore')
df = df.loc[:, ~df.columns.str.contains('^Unnamed')]

# Encode target variable
le = LabelEncoder()
df['diagnosis'] = le.fit_transform(df['diagnosis'])  # M=1, B=0

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

# 2. Train/test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=42)

# 3. Feature scaling
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# 4. Train SVM with linear and RBF kernels
svm_linear = SVC(kernel='linear', C=1)
svm_rbf = SVC(kernel='rbf', C=1, gamma='scale')

svm_linear.fit(X_train_scaled, y_train)
svm_rbf.fit(X_train_scaled, y_train)

# Evaluation function
def evaluate(model, name):
    y_pred = model.predict(X_test_scaled)
    print(f"\n{name} Evaluation:")
    print("Accuracy:", accuracy_score(y_test, y_pred))
    print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
    print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
evaluate(svm_linear, "Linear SVM")
evaluate(svm_rbf, "RBF SVM")

# 5. Hyperparameter tuning using GridSearchCV
param_grid = {
    'C': [0.1, 1, 10, 100],
    'gamma': ['scale', 0.01, 0.1, 1, 10]
}
grid = GridSearchCV(SVC(kernel='rbf'), param_grid, cv=5)
grid.fit(X_train_scaled, y_train)

print("\nBest Parameters from GridSearchCV:", grid.best_params_)

# 6. Cross-validation scores
best_model = grid.best_estimator_
cv_scores = cross_val_score(best_model, X_train_scaled, y_train, cv=5)
print("Cross-validation scores:", cv_scores)
print("Mean CV accuracy:", np.mean(cv_scores))

# 7. Visualization using PCA (2D projection)
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_train_scaled)
best_model.fit(X_pca, y_train)

def plot_decision_boundary(model, X, y, title):
    h = 0.02
    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=0.3)
    plt.scatter(X[:, 0], X[:, 1], c=y, cmap=plt.cm.coolwarm, edgecolors='k')
    plt.title("Breast cancer classification with SVM")
    plt.xlabel("PCA 1")
    plt.ylabel("PCA 2")
    plt.show()

plot_decision_boundary(best_model, X_pca, y_train, "SVM Decision Boundary
(PCA-reduced)")
```