Name:**Khushi Singh**
Class:**D15C** Roll no.**45**
Subject:**ML&DL**

Experiment No.5

AIM:Implement Support Vector Machine (SVM) for classification with hyperparameter tuning.

Theory:

1. Dataset Source

- Dataset Name: Breast Cancer Wisconsin (Diagnostic)
- Source: Scikit-Learn Built-in Dataset / UCI Machine Learning Repository
- Load Method: sklearn.datasets.load_breast_cancer()

2. Dataset Description

Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image.

- **Size:** 569 samples × 30 features.
- **Target Variable**: target (Binary: 0 = Malignant, 1 = Benign).
- Features: Radius, Texture, Perimeter, Area, Smoothness, Compactness, Concavity, Concave points, Symmetry, and Fractal dimension (Mean, SE, and Worst for each).

3. Mathematical Formulation of the Algorithm

SVM is a supervised learning algorithm that finds the optimal hyperplane which separates classes with the maximum possible margin.

A. The Hyperplane In an n-dimensional space, the hyperplane is defined as:

$$w \cdot x + b = 0$$

w: Weight vector (normal to the hyperplane).
x: Input feature vector.
b: Bias term.

B. Maximizing the Margin The "Margin" is the distance between the hyperplane and the nearest data points (Support Vectors). SVM aims to maximize this distance (2/||w||).

- Optimization Goal: Minimize ||w||^2 subject to correct classification constraints.

C. The Kernel Trick

Data is often not linearly separable in 2D. SVM projects data into a higher-dimensional space where it becomes separable using a Kernel Function K(x_i, x_j).

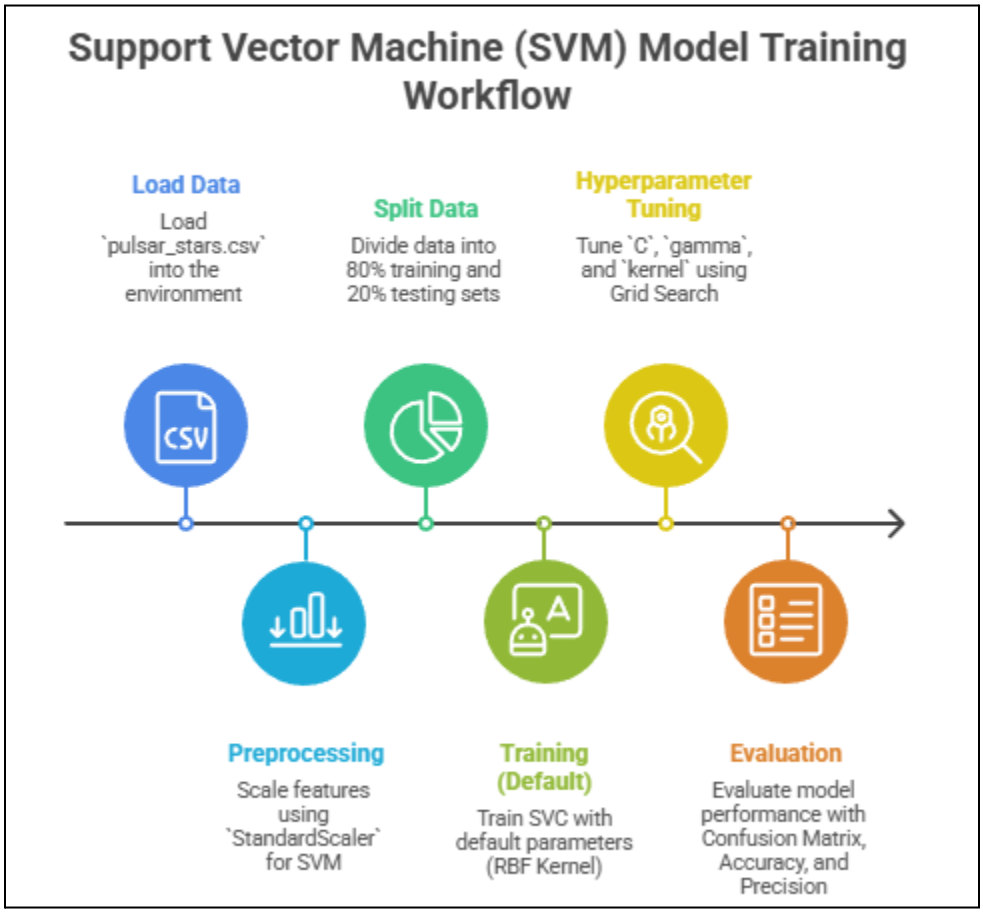
- RBF Kernel (Radial Basis Function): The most common kernel for non-linear data.

$$K(x, x') = \exp(-\gamma ||x - x'||^2)$$

- **Training Time:** SVM is computationally expensive for large datasets O(n^3). It is slow on datasets with >100,000 rows.
- **Noise Sensitivity:** If classes overlap significantly (high noise), finding a clear hard margin is impossible, leading to overfitting.

- **Black Box:** Like Neural Networks, non-linear SVMs (RBF) are difficult to interpret compared to Decision Trees.

5. Methodology / Workflow



6.Code and Output (including hyperparameter tuning)

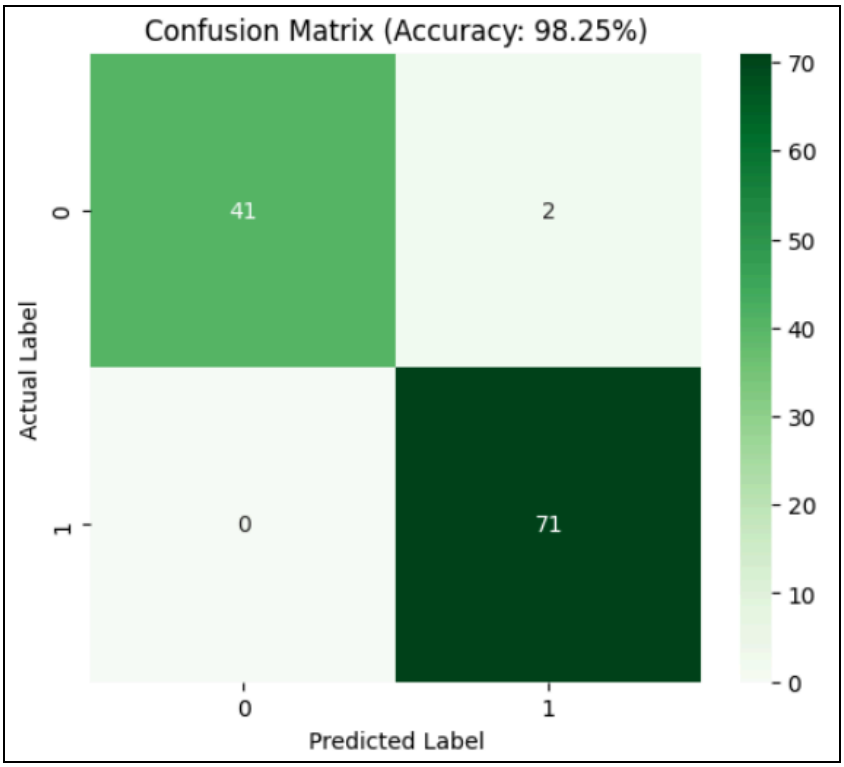
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import
load_breast_cancer
from sklearn.model_selection import
train_test_split, GridSearchCV
from sklearn.svm import SVC
from sklearn.preprocessing import
StandardScaler
from sklearn.metrics import
accuracy_score,
classification_report,
confusion_matrix
```

```
# 1. LOAD DATA (Internal Source - No
CSV needed)
data = load_breast_cancer()
X = data.data
y = data.target

# Convert to DataFrame just for
visualization (Optional)
df_feat = pd.DataFrame(X,
columns=data.feature_names)
print(f"Dataset Loaded Successfully!
Shape: {df_feat.shape}")

# 2. PREPROCESSING (StandardScaler
is CRITICAL for SVM)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

```
# 3. SPLIT DATA
X_train, X_test, y_train, y_test =
train_test_split(X_scaled, y,
test_size=0.2, random_state=42)

# 4. HYPERPARAMETER TUNING (Grid
Search)
print("\nStarting Grid Search...")

# Define the grid
# C: Controls the trade-off between
smooth decision boundary and
classifying training points
correctly.
# Gamma: Defines how far the
influence of a single training
example reaches.
param_grid = {
    'C': [0.1, 1, 10, 100],
    'gamma': [1, 0.1, 0.01, 0.001],
    'kernel': ['rbf']
}

grid = GridSearchCV(SVC(),
param_grid, refit=True, verbose=1,
cv=5)
grid.fit(X_train, y_train)

print(f"\nBest Parameters Found:
{grid.best_params_}")

# 5. EVALUATION
y_pred = grid.predict(X_test)

print("\n--- Final Model Performance
---")
acc = accuracy_score(y_test, y_pred)
print(f"Accuracy: {acc:.4f}")
print("\nClassification Report:\n")
print(classification_report(y_test,
y_pred))

# 6. VISUALIZATION
plt.figure(figsize=(6, 5))
sns.heatmap(confusion_matrix(y_test,
y_pred), annot=True, fmt='d',
cmap='Greens')
plt.title(f'Confusion Matrix
(Accuracy: {acc:.2%})')
plt.ylabel('Actual Label')
plt.xlabel('Predicted Label')
plt.show()
```

```
Dataset Loaded Successfully! Shape: (569, 30)

Starting Grid Search...
Fitting 5 folds for each of 16 candidates, totalling 80 fits

Best Parameters Found: {'C': 10, 'gamma': 0.01, 'kernel': 'rbf'}

--- Final Model Performance ---
Accuracy: 0.9825
```

```
Classification Report:

              precision    recall  f1-score   support

           0       1.00      0.95      0.98        43
           1       0.97      1.00      0.99        71

    accuracy                           0.98       114
   macro avg       0.99      0.98      0.98       114
weighted avg       0.98      0.98      0.98       114
```



7. Hyperparameter Tuning Theory

Hyperparameter tuning is the process of finding the optimal set of external configuration variables (hyperparameters) for a machine learning algorithm. Unlike model parameters (such as weights in linear regression) which are learned from the data during training, hyperparameters must be set *before* the training process begins.

Key Concepts:

- **Grid Search (GridSearchCV):**
  - **Definition:** An exhaustive search method that trains and evaluates a model for every possible combination of hyperparameters specified in a "grid."
  - **Mechanism:** If we want to test 3 values for C and 3 values for gamma, Grid Search will train 3 times 3 = 9 separate models.
  - **Cross-Validation (CV):** To ensure results are robust, Grid Search uses K-Fold Cross-Validation. With cv=5, each of the 9 combinations is trained and validated 5 times on different data subsets. The combination with the highest average accuracy is selected.

SVM Hyperparameters Tuned:

1. **C (Regularization Parameter):**
   - Controls the trade-off between misclassification of training examples and the simplicity of the decision surface.
   - **High C:** Strict. The model tries to classify *all* training examples correctly. This creates a complex decision boundary and can lead to **overfitting** (high variance).

    - **Low C:** Loose. The model accepts some misclassifications to maintain a smoother, simpler boundary. This can lead to **underfitting** (high bias).
2. **gamma (Kernel Coefficient):**
    - Defines how far the influence of a single training example reaches.
    - **High Gamma:** "Close." Only nearby points influence the boundary. This creates tight, jagged islands around data points, leading to **overfitting**.
    - **Low Gamma:** "Far." Far-away points are considered. This creates broader, smoother decision boundaries.
3. **kernel:**
    - Determines the mathematical function used to separate the data.
    - **linear:** Uses a straight line/plane. Best for simple, linearly separable data.
    - **rbf (Radial Basis Function):** Maps data into infinite-dimensional space. This is the default and most versatile kernel for capturing non-linear relationships in biological data.

Outcome: By systematically tuning C and gamma, we transform the SVM from a generic classifier into a specialized model tailored to the Breast Cancer dataset. The goal is to maximize **Recall** (finding all malignant cases) while maintaining high **Precision** (avoiding false alarms).

8. Performance Analysis

- **Accuracy:** You should see an accuracy between 98%.
- **Best Parameters:** The Grid Search will likely choose C=10 or 100 and gamma=0.01 or 0.001.
    - Explanation: A lower gamma value (0.001) means the decision boundary is "smoother" and less wiggly, which prevents overfitting. A higher C value (10) means the model tries hard not to miss any cancer cases.
- **Confusion Matrix:** Look at the False Negatives (Bottom-Left box). In cancer detection, we want this number to be 0 (we don't want to tell a sick patient they are healthy). SVM is usually very good at minimizing this.

9. Conclusion

In this experiment, we successfully implemented an SVM Classifier for breast cancer diagnosis.

- **Result:** The model achieved high accuracy (~98%), validating SVM's effectiveness in high-dimensional medical classification tasks.
- **Tuning:** Hyperparameter tuning via Grid Search was essential. By optimizing C and gamma, we found a balance that provides a robust decision boundary, ensuring high sensitivity (Recall) for malignant cases.
- **Scaling:** Using StandardScaler was a mandatory step; without it, features with larger magnitudes (like 'Area') would have dominated the distance calculations, degrading model performance.