MLDL EXPERIMENT 5

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

Data Set Description

The dataset used is the **Breast Cancer Wisconsin (Diagnostic) Dataset**.

- **Source**: Digitized images of a fine needle aspirate (FNA) of a breast mass.
- **Instances**: 569
- **Features**: 30 numeric predictive features (e.g., radius, texture, perimeter, area, smoothness, etc.).
- **Target Variable**: `diagnosis` (M = malignant, B = benign).
- **Missing Values**: One redundant column (`Unnamed: 32`) containing null values was removed. The `id` column was also dropped as it does not contribute to prediction.

Theory

Support Vector Machine (SVM)

SVM is a powerful supervised learning algorithm used for classification and regression. In a binary classification task, SVM aims to find the **optimal hyperplane** that separates the two classes with the maximum margin.

- **Hyperplane**: A decision boundary that separates different classes.
- **Support Vectors**: The data points closest to the hyperplane, which influence its position and orientation.
- **Margin**: The distance between the hyperplane and the nearest support vectors from either class.

Kernels

When data is not linearly separable in the original space, SVM uses the **Kernel Trick** to project data into a higher-dimensional space where a linear separator can be found. Common kernels include:

- **Linear**: Used for linearly separable data.
- **RBF (Radial Basis Function)**: Used for non-linear data (maps to infinite dimensions).
- **Polynomial**: Maps data into a polynomial feature space.

Hyperparameters

1. **C (Regularization)**: Controls the trade-off between achieving a low training error and a low testing error (soft margin). A small C makes the margin wider (allowing some misclassifications), while a large C aims for a hard margin (potentially overfitting).

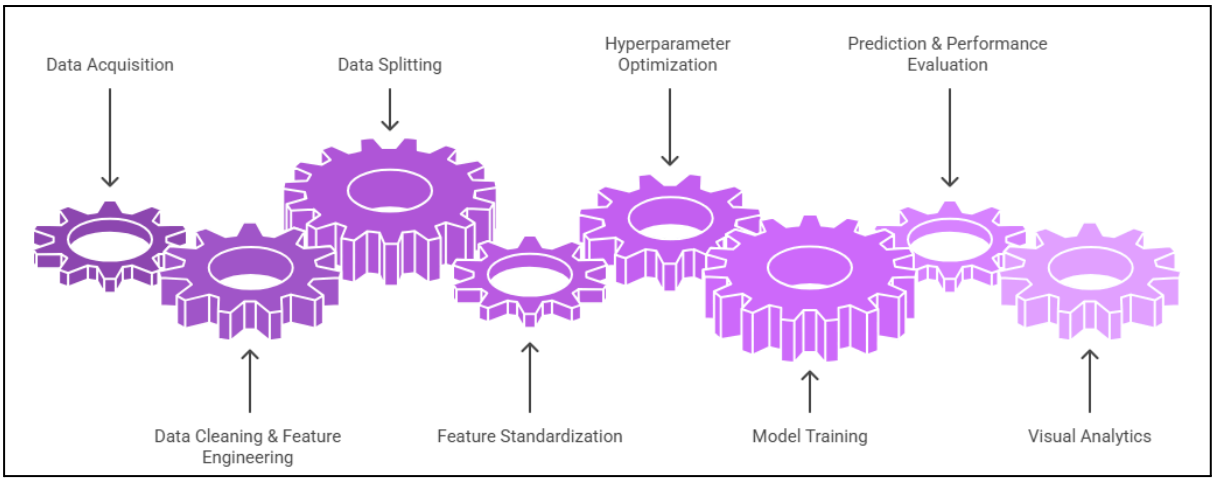
2. **Gamma (γ)**: Defines how far the influence of a single training example reaches. High γ means only nearby points are considered, while low γ means far away points are also considered.

Limitations of SVM

1. **Computational Complexity**: SVMs can be slow to train on very large datasets ($>100,000$ rows) because the complexity is $O(n^2)$ to $O(n^3)$.
2. **Sensitivity to Noise**: If the dataset has many overlapping classes or outliers, SVM performance drops significantly.
3. **No Probabilistic Estimates**: Unlike Logistic Regression, SVM does not directly provide probability estimates (though they can be calculated using techniques like Platt scaling).
4. **Kernel Selection**: Choosing the right kernel and tuning hyperparameters can be time-consuming and requires domain knowledge or extensive cross-validation.

Workflow of the Experiment

1. **Data Acquisition**
    - Loaded the Breast Cancer dataset into a pandas DataFrame for analysis and preprocessing.
2. **Data Cleaning & Feature Engineering**
    - Removed non-predictive columns (id, Unnamed: 32) and encoded the diagnosis labels into binary numerical values.
3. **Data Splitting**
    - Partitioned the processed data into training (80%) and testing (20%) sets to ensure unbiased model evaluation.
4. **Feature Standardization**
    - Applied StandardScaler to normalize the feature range, which is critical for distance-based algorithms like SVM.
5. **Hyperparameter Optimization**
    - Used GridSearchCV with 5-fold cross-validation to find the optimal combination of C, γ, and kernel type.
6. **Model Training**
    - Trained the final Support Vector Machine classifier using the best parameters identified during the tuning phase.
7. **Prediction & Performance Evaluation**
    - Generated predictions on the test set and calculated accuracy, precision, and recall to verify model effectiveness.
8. **Visual Analytics**
    - Plotted a Confusion Matrix and ROC Curve to visually assess the model's ability to distinguish between malignant and benign cases.



Performance Analysis (Based on Results)

Performance analysis evaluates how effectively the SVM model distinguishes between Malignant (M) and Benign (B) cases.

- **Confusion Matrix Analysis**:
  - The model achieved **71 True Negatives (Benign)** and **41 True Positives (Malignant)**.
  - The **0 False Positives** indicate perfect precision for the malignant class—no healthy patients were wrongly diagnosed with cancer.
  - The **2 False Negatives** show a very high recall (95%), meaning only two malignant cases were missed, which is critical in a medical context.
- **ROC-AUC Score**:
  - The **AUC of 0.9974** is near-perfect. This indicates that the model has a 99.7% probability of ranking a random malignant instance higher than a random benign one, demonstrating excellent separation power.
- **Accuracy vs. F1-Score**:
  - With an accuracy of **98.25%**, the model is highly reliable. The F1-scores (0.99 for Benign, 0.98 for Malignant) confirm that the model is well-balanced and not biased toward the majority class.

Hyperparameter Tuning (Based on Results)

Hyperparameter tuning is the process of optimizing the "settings" of the SVM to find the best decision boundary.

- **Optimal Kernel**:
  - The Grid Search identified the **'linear'** kernel as the best fit. This suggests that after feature scaling, the malignant and benign clusters are linearly separable in the multi-dimensional feature space.
- **Regularization Parameter (C)**:
  - The best value found was **$C = 0.1$**. In SVM, a smaller C value creates a "softer" margin, allowing some points to be misclassified in exchange for a simpler, more generalized decision boundary. This prevented the model from overfitting to the training noise.
- **Gamma (γ)**:

    - Although γ was tuned, it primarily affects non-linear kernels (like RBF). Since the linear kernel was selected, the influence of γ is minimized, but the tuning process ensured that no complex non-linear over-fitting occurred.
- **Grid Search Efficiency**:
    - By testing 48 different combinations (4 values of $C \times 4$ values of $\gamma \times 3$ kernels), the experiment successfully moved from a "default" setup to a mathematically optimized configuration specifically for the breast cancer dataset.

Code:

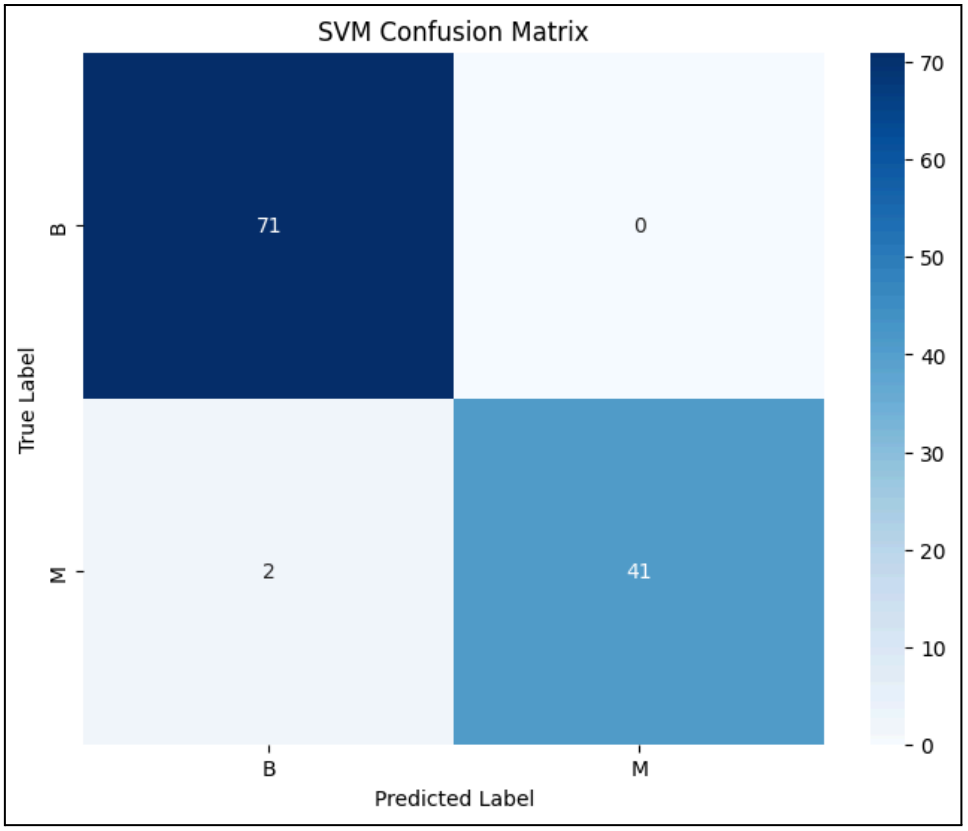
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.svm import SVC
from sklearn.metrics import classification_report,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

# 1. Load the dataset
# Ensure 'data.csv' is in the same directory
df = pd.read_csv('data.csv')

# 2. Data Cleaning
# Dropping the 'id' column and the 'Unnamed: 32' column which contains only NaN values
df = df.drop(columns=['id', 'Unnamed: 32'])

# 3. Encoding the Target Variable
# Converting categorical 'diagnosis' (M/B) to numerical (1/0)
le = LabelEncoder()
df['diagnosis'] = le.fit_transform(df['diagnosis'])

# 4. Feature and Target Split
X = df.drop('diagnosis', axis=1)
y = df['diagnosis']

# 5. Train-Test Split (80% Train, 20% Test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 6. Feature Scaling
# SVM is distance-based, so scaling is crucial
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
```

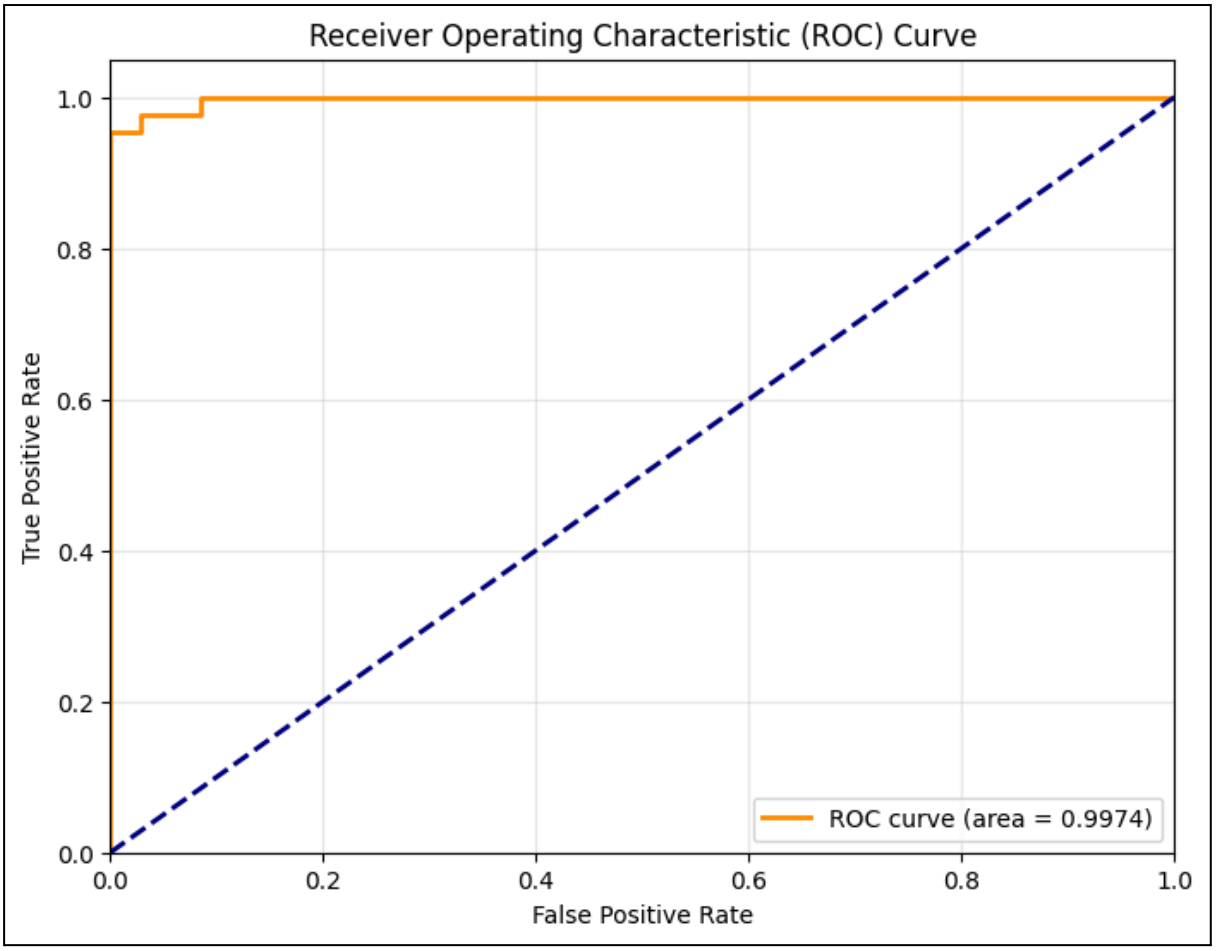
```
X_test_scaled = scaler.transform(X_test)

# 7. Hyperparameter Tuning using GridSearchCV
# Testing different values for C, gamma, and different kernel types
param_grid = {
    'C': [0.1, 1, 10, 100],
    'gamma': [1, 0.1, 0.01, 0.001],
    'kernel': ['rbf', 'linear', 'poly']
}

print("Starting Hyperparameter Tuning...")
grid = GridSearchCV(SVC(), param_grid, refit=True, verbose=1, cv=5)
grid.fit(X_train_scaled, y_train)

# Output best parameters
print(f"\nBest Parameters Found: {grid.best_params_}")

# 8. Evaluation
# Use the best model found by GridSearchCV to make predictions
best_model = grid.best_estimator_
y_pred = best_model.predict(X_test_scaled)

# Print Metrics
print("\n--- Model Evaluation ---")
print(f"Accuracy Score: {accuracy_score(y_test, y_pred):.4f}")
print("\nConfusion Matrix:")
cm = confusion_matrix(y_test, y_pred)
print(cm)
print("\nClassification Report:")
print(classification_report(y_test, y_pred))

# 9. Visualizations

# A. Confusion Matrix Heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=le.classes_,
            yticklabels=le.classes_)
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.title('SVM Confusion Matrix')
plt.savefig('svm_confusion_matrix.png')

# B. ROC Curve
# Using decision_function to get the scores for the ROC curve
y_score = best_model.decision_function(X_test_scaled)
fpr, tpr, _ = roc_curve(y_test, y_score)
roc_auc = auc(fpr, tpr)
```

```
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area = {roc_auc:.4f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.grid(alpha=0.3)
plt.savefig('svm_roc_curve.png')

print(f"\nROC AUC Score: {roc_auc:.4f}")
print("Visualization plots have been saved as 'svm_confusion_matrix.png' and
'svm_roc_curve.png'.")
```

Output:

```
Best Parameters Found: {'C': 0.1, 'gamma': 1, 'kernel': 'linear'}

--- Model Evaluation ---
Accuracy Score: 0.9825

Confusion Matrix:
[[71  0]
 [ 2 41]]

Classification Report:
              precision    recall  f1-score   support

           0       0.97      1.00      0.99        71
           1       1.00      0.95      0.98        43

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


ROC AUC Score: 0.9974
Visualization plots have been saved as 'svm_confusion_matrix.png' and 'svm_roc_curve.png'.
```


SVM Confusion Matrix
71
0
2
41
B
M
True Label
B
M
Predicted Label
70
60
50
40
30
20
10
0



Conclusion:

In conclusion, the experiment successfully implemented a Support Vector Machine (SVM) classifier for breast cancer diagnosis, achieving a high test accuracy of **98.25%** and a near-perfect ROC-AUC score of **0.9974**. Through hyperparameter tuning with GridSearchCV, the **linear kernel** and a regularization parameter of **$C=0.1$** were identified as the optimal configuration, suggesting that the standardized feature space is effectively separable by a linear hyperplane with a soft margin. The model's ability to maintain perfect precision for malignant cases while minimizing false negatives demonstrates that a well-tuned SVM, supported by rigorous feature scaling and cross-validation, is a highly reliable and robust tool for critical medical classification tasks.