CSBB 311 : MACHINE LEARNING

LAB ASSIGNMENT 8 : Feature Selection Using Backward Selection

Submitted By:

Name: KARTIK MITTAL

Roll No: 221210056

Branch: CSE

Semester: 5th Sem

Group: 2

Submitted To: Dr. Preeti Verma

Department of Computer Science and Engineering

NATIONAL INSTITUTE OF TECHNOLOGY DELHI

2024

Code -

```python
import pandas as pd
import numpy as np
from itertools import combinations
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
import time

# Load the Iris dataset
df = pd.read_csv("iris.csv")
X = df.drop(columns=['species'])  # Remove target column
y = df['species']
feature_names = X.columns  # Store feature names for readability

# Standardize the feature data
scaler = StandardScaler()
X = scaler.fit_transform(X)
X = pd.DataFrame(X, columns=feature_names)  # Keep column names for readability

# Initialize lists to store results
combinations_list = []
accuracies = []
best_accuracy = 0
best_combination = ()

# Start timing
start_time = time.time()
```

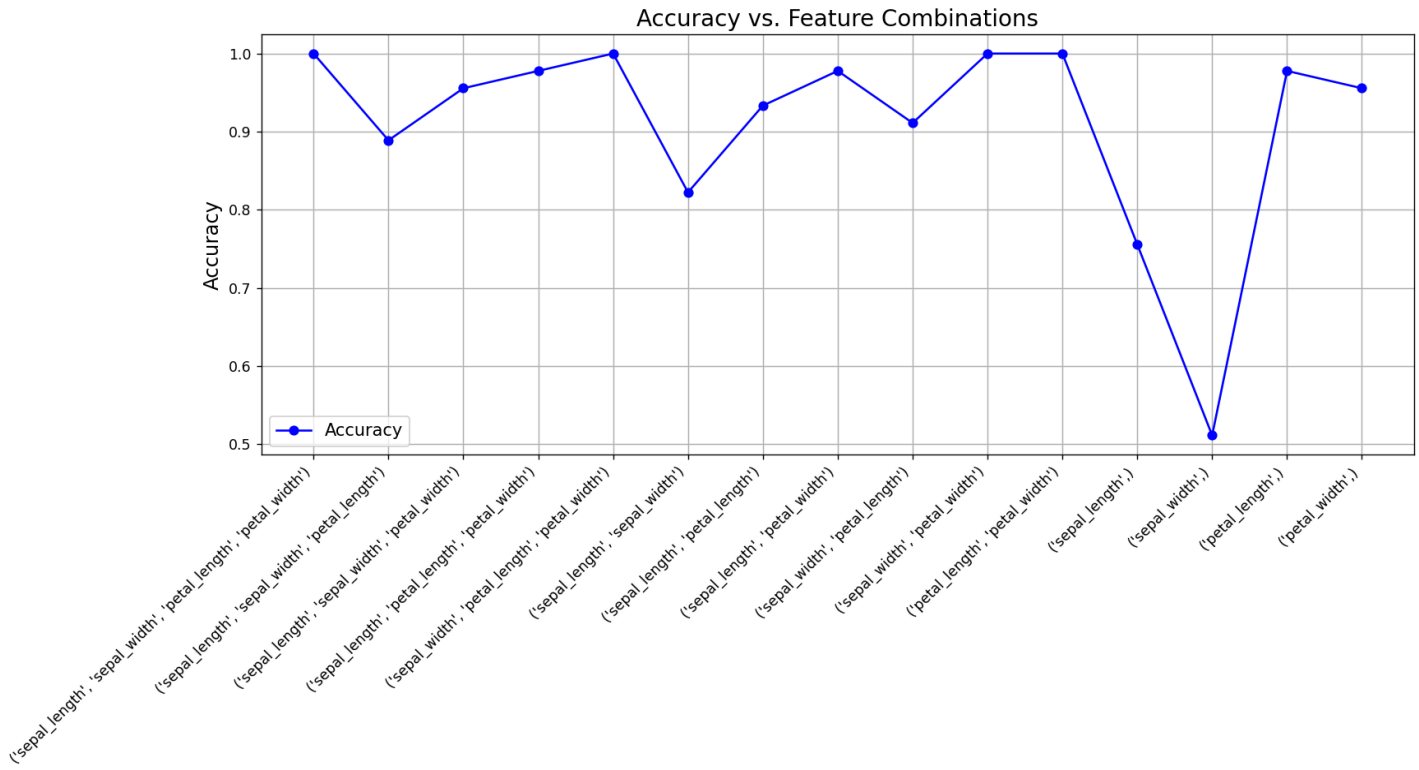
```python
# Iterate over all possible feature combinations in reverse order
for n_features in range(X.shape[1], 0, -1):  # Reverse range
    for combo in combinations(feature_names, n_features):
        # Select features based on the combination
        X_selected = X[list(combo)]
        X_train, X_test, y_train, y_test = train_test_split(X_selected, y, test_size=0.3, random_state=42)

        # Train and evaluate the KNN model
        knn = KNeighborsClassifier()
        knn.fit(X_train, y_train)
        y_pred = knn.predict(X_test)
        accuracy = accuracy_score(y_test, y_pred)

        # Store results
        combinations_list.append(combo)
        accuracies.append(accuracy)

        # Update the best combination:
        # 1. If the current accuracy is higher, update the best combination.
        # 2. If the current accuracy is equal to the best but uses fewer features, update it.
        if accuracy > best_accuracy or (accuracy == best_accuracy and len(combo) < len(best_combination)):
            best_accuracy = accuracy
            best_combination = combo

        print(f"Features: {combo}, Accuracy: {accuracy:.4f}")

# End timing
end_time = time.time()
total_time = end_time - start_time
```

```
# Display the best feature combination with the fewest features and timing information
print(f"\nTotal time taken: {total_time:.2f} seconds")
print(f"Best feature combination with the minimum features: {best_combination} with accuracy: {best_accuracy:.4f}")

# Plotting accuracy for each combination of features
plt.figure(figsize=(14, 8))
plt.plot(range(1, len(accuracies) + 1), accuracies, marker='o', linestyle='-', color='b')
plt.xticks(range(1, len(accuracies) + 1), [str(combo) for combo in combinations_list], rotation=45, ha='right')
plt.xlabel("Feature Combination", fontsize=14)
plt.ylabel("Accuracy", fontsize=14)
plt.title("Accuracy vs. Feature Combinations", fontsize=16)
plt.legend(['Accuracy'], fontsize=12)
plt.grid(True)
plt.tight_layout()
plt.show()

# Scatter plots for combinations of two features
for combo in combinations(feature_names, 2):
    plt.figure(figsize=(8, 6))
    plt.scatter(X[combo[0]], X[combo[1]], c=y.map({'setosa': 0, 'versicolor': 1, 'virginica': 2}),
                cmap='viridis', edgecolor='k')
    plt.xlabel(combo[0], fontsize=12)
    plt.ylabel(combo[1], fontsize=12)
    plt.title(f"Scatter Plot for Features: {combo}", fontsize=14)
    plt.colorbar(label='Species')
    plt.grid(True)
    plt.show()
```

Accuracy Prediction -

```
Features: ('sepal_length', 'sepal_width', 'petal_length', 'petal_width'), Accuracy: 1.0000
Features: ('sepal_length', 'sepal_width', 'petal_length'), Accuracy: 0.8889
Features: ('sepal_length', 'sepal_width', 'petal_width'), Accuracy: 0.9556
Features: ('sepal_length', 'petal_length', 'petal_width'), Accuracy: 0.9778
Features: ('sepal_width', 'petal_length', 'petal_width'), Accuracy: 1.0000
Features: ('sepal_length', 'sepal_width'), Accuracy: 0.8222
Features: ('sepal_length', 'petal_length'), Accuracy: 0.9333
Features: ('sepal_length', 'petal_width'), Accuracy: 0.9778
Features: ('sepal_width', 'petal_length'), Accuracy: 0.9111
Features: ('sepal_width', 'petal_width'), Accuracy: 1.0000
Features: ('petal_length', 'petal_width'), Accuracy: 1.0000
Features: ('sepal_length',), Accuracy: 0.7556
Features: ('sepal_width',), Accuracy: 0.5111
Features: ('petal_length',), Accuracy: 0.9778
Features: ('petal_width',), Accuracy: 0.9556

Total time taken: 0.11 seconds
Best feature combination with the minimum features: ('sepal_width', 'petal_width') with accuracy: 1.0000
```

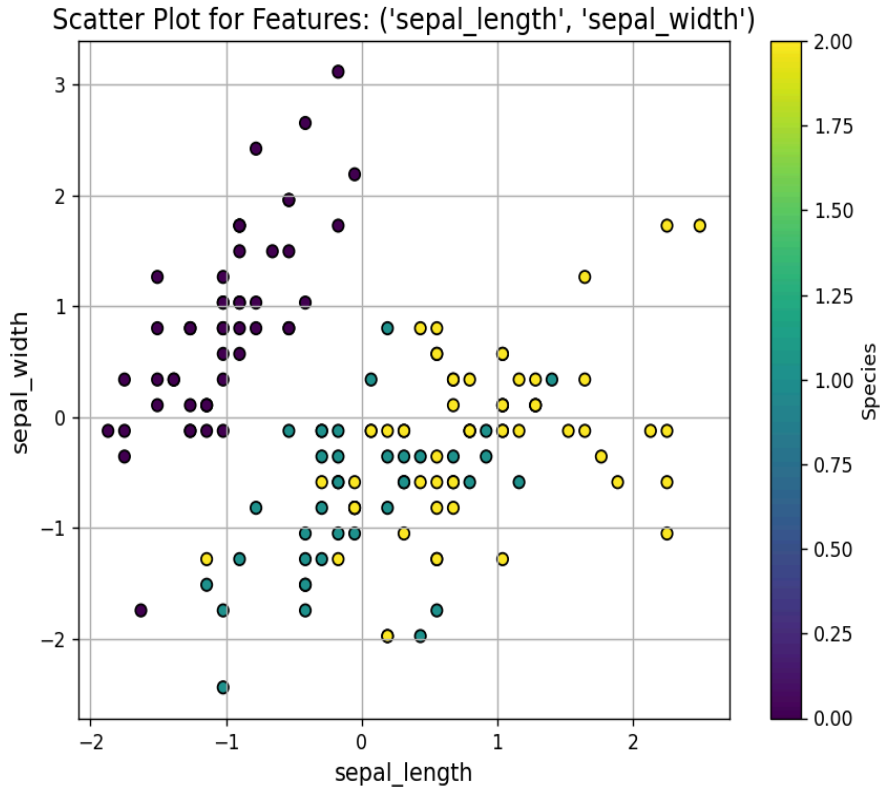
Output -

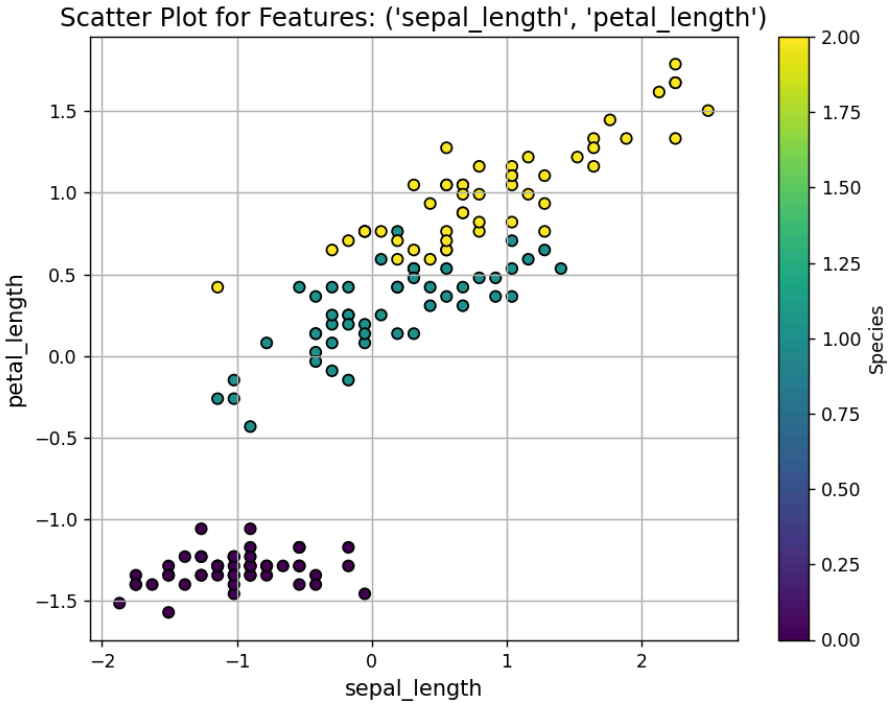


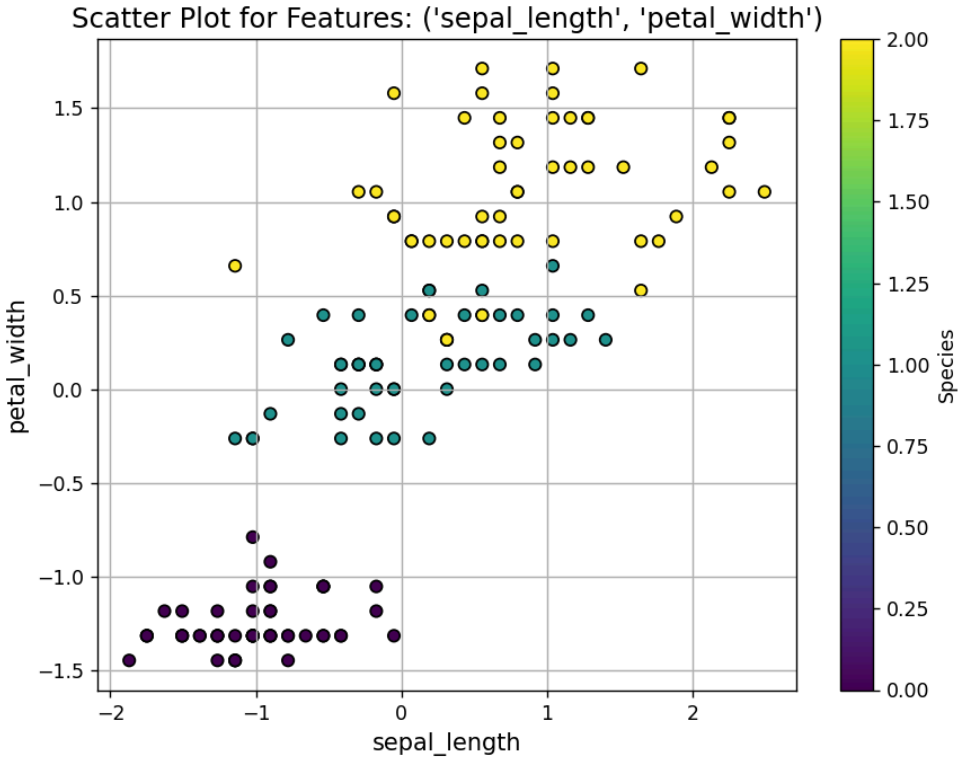
Accuracy Vs No. Of Features Graph

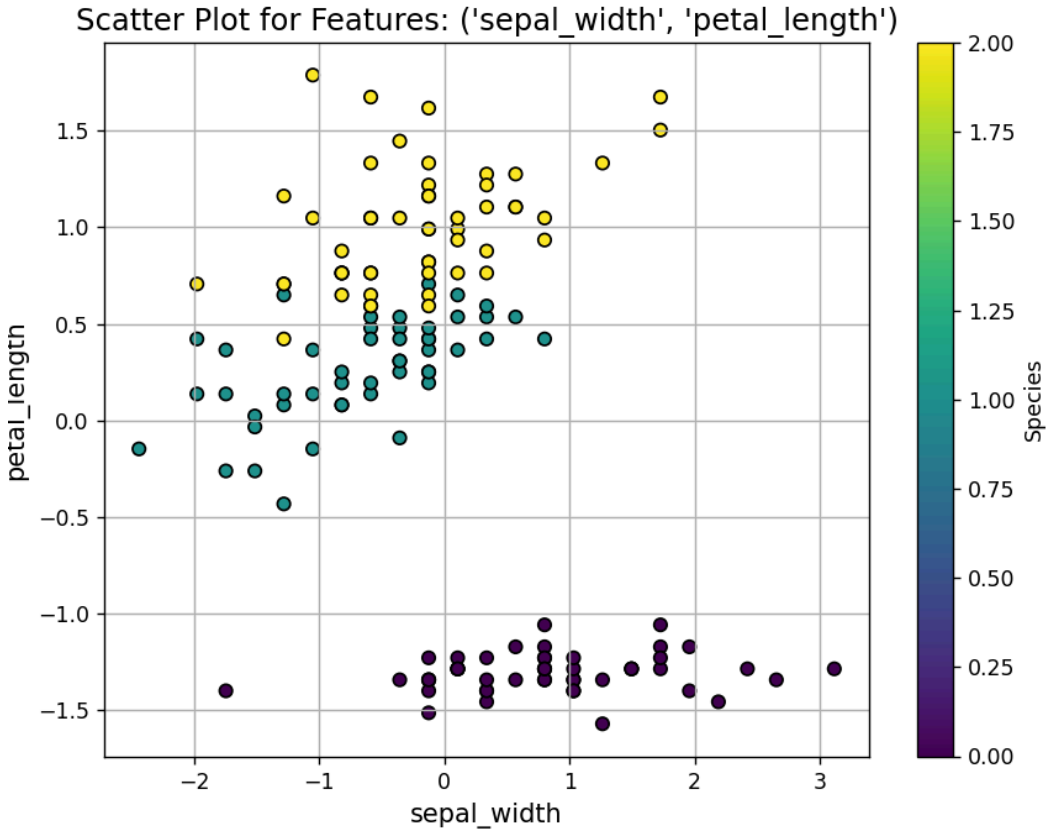
```
Total time taken: 0.11 seconds
Best feature combination with the minimum features: ('sepal_width', 'petal_width') with accuracy: 1.0000
```

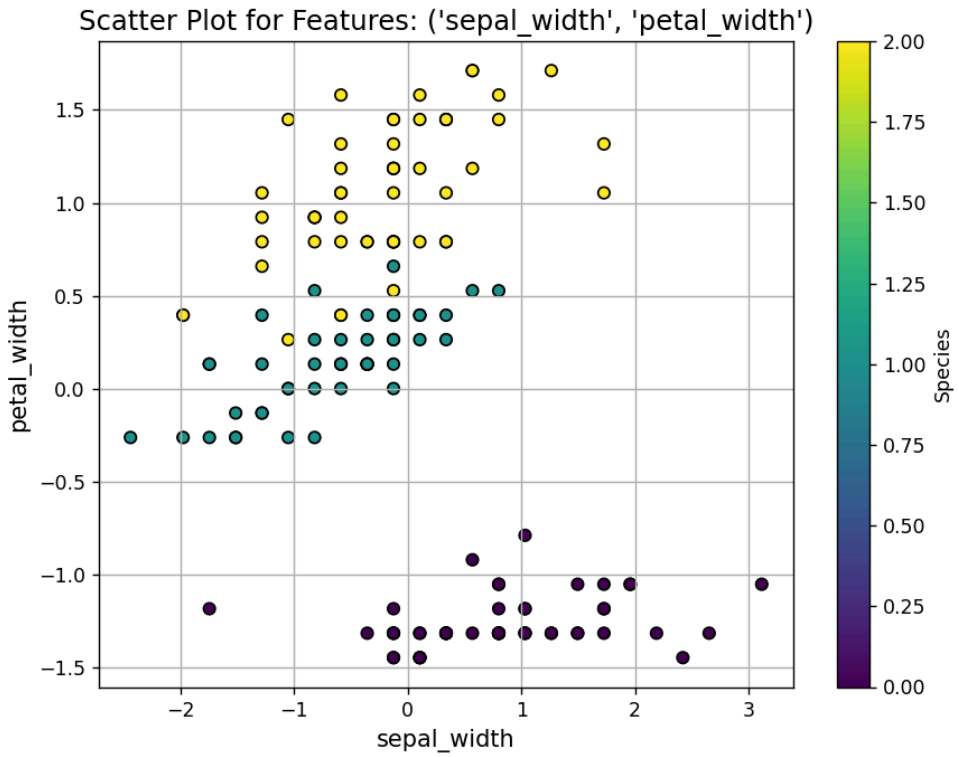
Scatter Plots

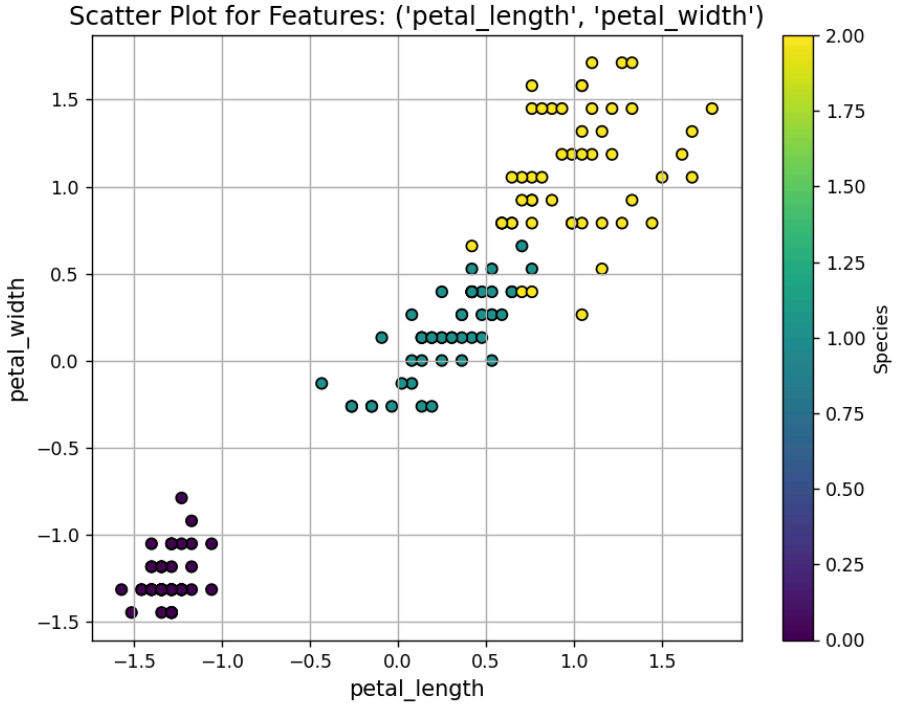





Scatter Plot for Features: ('sepal_length', 'petal_width')
petal_width
sepal_length
Species
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2
−1
0
1
2
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00


Scatter Plot for Features: ('sepal_width', 'petal_length')
petal_length
sepal_width
Species
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2
−1
0
1
2
3
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00


Scatter Plot for Features: ('sepal_width', 'petal_width')
petal_width
sepal_width
Species
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2
−1
0
1
2
3
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00


Scatter Plot for Features: ('petal_length', 'petal_width')
petal_width
petal_length
Species
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00