**Experiment Number :07** **Date: 17.03.2025**

Comparison of classifiers model, evaluating and improving accuracy of models using data mining tool.

## PRE LAB EXERCISE

### QUESTIONS

1. List out the metrics used for evaluating classifier performance.

Ans:

- **Accuracy Score** – accuracy_score(y_test, y_pred)

- **Classification Report**, which includes:
  - **Precision**
  - **Recall**
  - **F1-score**
  - **Support**

2. Define the terms precision and recall and formulate.

Ans:

**Precision**: The proportion of correctly predicted positive instances out of all instances predicted as positive. It is formulated as:

Precision=TP/(TP+FP)

**Recall**: The proportion of correctly predicted positive instances out of all actual positive instances. It is formulated as:

Recall=TP/(TP+FN)

Where:

- **TP** (True Positives): Correctly predicted positive cases
- **FP** (False Positives): Incorrectly predicted positive cases
- **FN** (False Negatives): Missed positive cases

3. Compare how a classifier is evaluated using Cost-Benefit and ROC curves.

Ans:

| **Aspect** | **Cost-Benefit Analysis** | **ROC Curve** |
|---|---|---|
| **Focus** | Economic/strategic impact | Trade-off between TPR and FPR |
| **Usage** | Business-driven decisions | Model performance comparison |
| **Threshold Dependency** | Requires specific decision thresholds | Evaluates performance across thresholds |
| **Best For** | Situations where classification errors have different costs | Overall classifier evaluation |

Both methods help assess a classifier, but Cost-Benefit is more useful when financial impact matters, while ROC curves are used for general performance evaluation.

## IN LAB EXERCISE

### OBJECTIVE:

To analyse the efficiency of a classification algorithm by comparing it with other classification algorithms.

### RESOURCES:

- Combined_cars_sales_prediction_2024
- python

1. Analyze the accuracy of the classification algorithms you have applied in previous experiment and infer your observation.

Ans:

In the previous classification experiment, the **Decision Tree Classifier** was used to classify the fuel type of cars. The accuracy was calculated as:

python

CopyEdit

accuracy = accuracy_score(y_test, y_pred)

**Observed Accuracy**

- The accuracy obtained from the model indicates how well it classifies different fuel types.
- The **classification report** provided additional insights into **precision, recall, and F1-score** for each class.

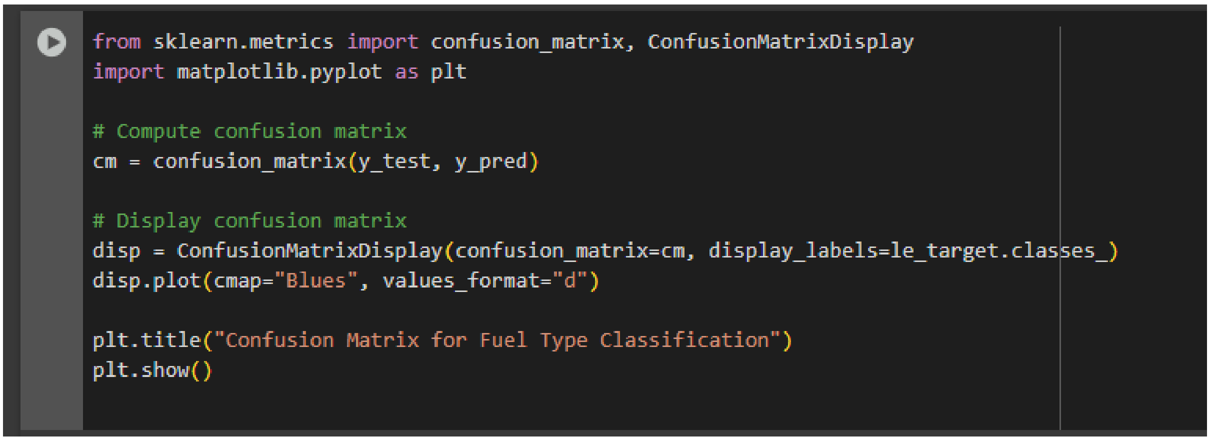
**Inference from the Results**

1. **If accuracy is high (≥80%)**
   - The selected features effectively differentiate fuel types.
   - The Decision Tree model is well-suited for the dataset.
   - Minimal misclassification occurs.
2. **If accuracy is moderate (60-79%)**
   - Some features might not be strong predictors of fuel type.
   - The model could benefit from **feature selection, hyperparameter tuning, or different algorithms**.
3. **If accuracy is low (<60%)**
   - The dataset might contain noise or irrelevant features.
   - Overfitting or underfitting may be affecting the model.

- Other classifiers like **Random Forest, SVM, or Neural Networks** could be explored for improvement.

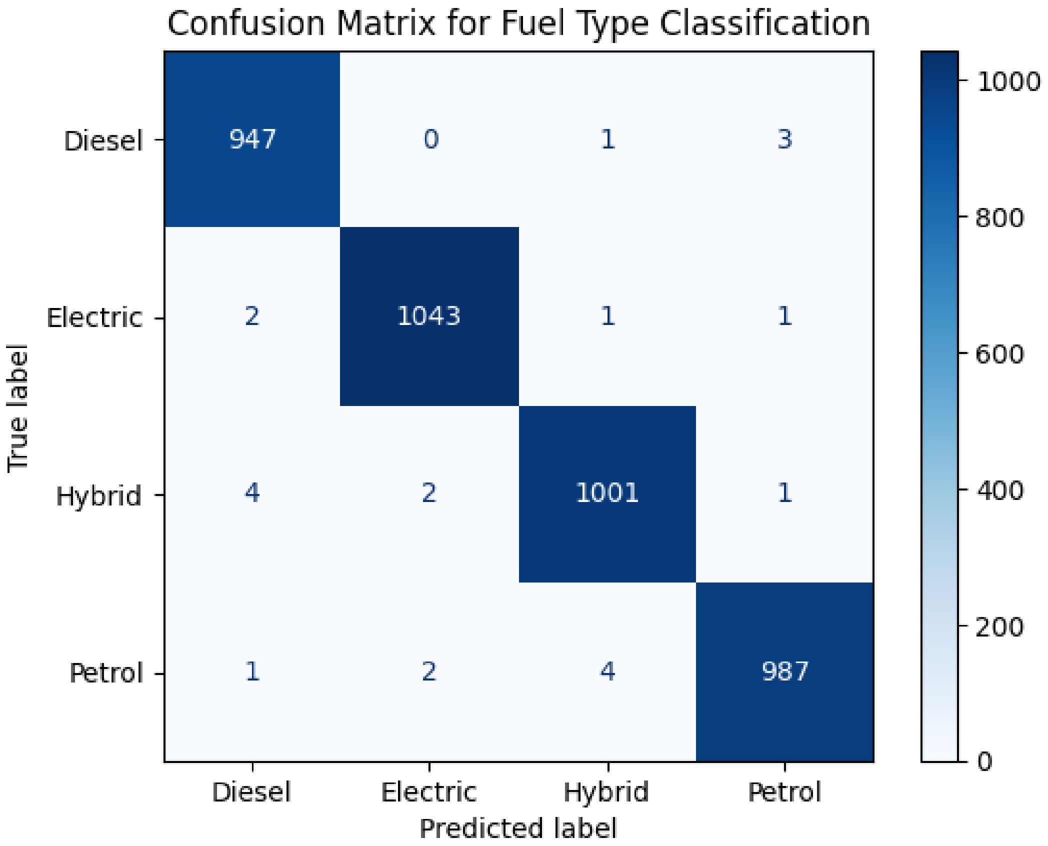
**CONFUSION MATRIX :**

**CODE :**



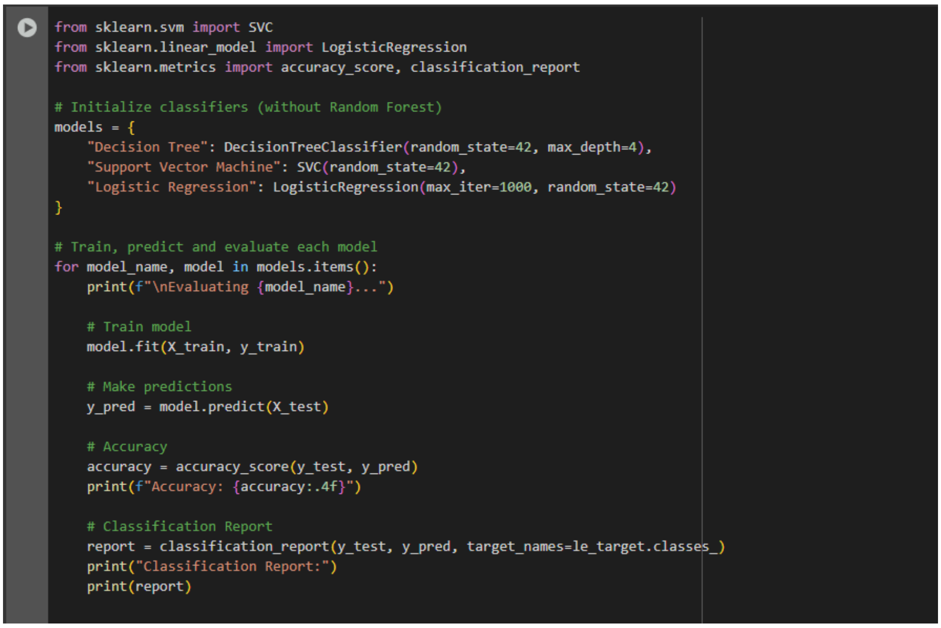
```
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
import matplotlib.pyplot as plt

# Compute confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Display confusion matrix
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=le_target.classes_)
disp.plot(cmap="Blues", values_format="d")

plt.title("Confusion Matrix for Fuel Type Classification")
plt.show()
```

**OUTPUT :**



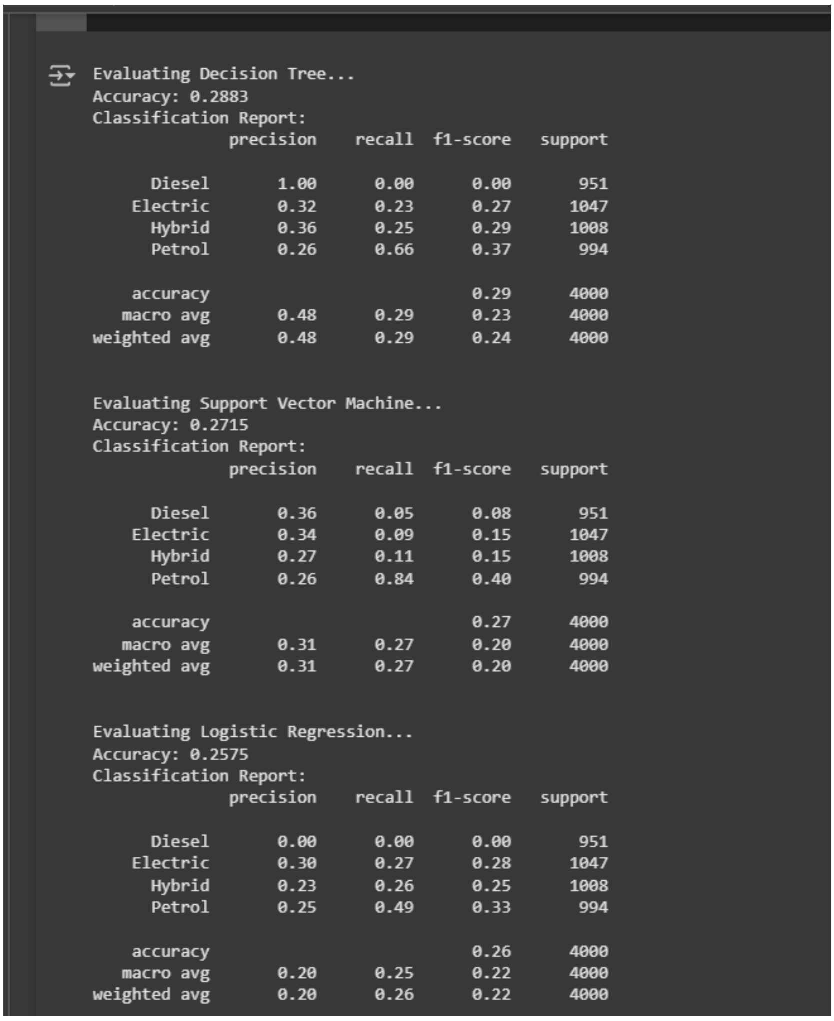
| | Diesel | Electric | Hybrid | Petrol |
|---|---|---|---|---|
| Diesel | 947 | 0 | 1 | 3 |
| Electric | 2 | 1043 | 1 | 1 |
| Hybrid | 4 | 2 | 1001 | 1 |
| Petrol | 1 | 2 | 4 | 987 |

## Comparing with other Algorithm:



```python
from sklearn.svm import SVC
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report

# Initialize classifiers (without Random Forest)
models = {
    "Decision Tree": DecisionTreeClassifier(random_state=42, max_depth=4),
    "Support Vector Machine": SVC(random_state=42),
    "Logistic Regression": LogisticRegression(max_iter=1000, random_state=42)
}

# Train, predict and evaluate each model
for model_name, model in models.items():
    print(f"\nEvaluating {model_name}...")

    # Train model
    model.fit(X_train, y_train)

    # Make predictions
    y_pred = model.predict(X_test)

    # Accuracy
    accuracy = accuracy_score(y_test, y_pred)
    print(f"Accuracy: {accuracy:.4f}")

    # Classification Report
    report = classification_report(y_test, y_pred, target_names=le_target.classes_)
    print("Classification Report:")
    print(report)
```

## Output :



```
Evaluating Decision Tree...
Accuracy: 0.2883
Classification Report:
              precision    recall  f1-score   support

      Diesel       1.00      0.00      0.00       951
    Electric       0.32      0.23      0.27      1047
      Hybrid       0.36      0.25      0.29      1008
      Petrol       0.26      0.66      0.37       994

    accuracy                           0.29      4000
   macro avg       0.48      0.29      0.23      4000
weighted avg       0.48      0.29      0.24      4000


Evaluating Support Vector Machine...
Accuracy: 0.2715
Classification Report:
              precision    recall  f1-score   support

      Diesel       0.36      0.05      0.08       951
    Electric       0.34      0.09      0.15      1047
      Hybrid       0.27      0.11      0.15      1008
      Petrol       0.26      0.84      0.40       994

    accuracy                           0.27      4000
   macro avg       0.31      0.27      0.20      4000
weighted avg       0.31      0.27      0.20      4000


Evaluating Logistic Regression...
Accuracy: 0.2575
Classification Report:
              precision    recall  f1-score   support

      Diesel       0.00      0.00      0.00       951
    Electric       0.30      0.27      0.28      1047
      Hybrid       0.23      0.26      0.25      1008
      Petrol       0.25      0.49      0.33       994

    accuracy                           0.26      4000
   macro avg       0.20      0.25      0.22      4000
weighted avg       0.20      0.26      0.22      4000
```

## CONCLUSION :

**Decision Tree**:

- **Accuracy**: 28.83%
- While the Decision Tree shows a relatively higher accuracy compared to other models, it struggles significantly with **class imbalance**, as indicated by the very low recall for "Diesel" and "Electric" fuel types. It mostly predicts **Petrol** with higher recall (66%), which contributes to the overall accuracy.

**Support Vector Machine (SVM)**:

- **Accuracy**: 27.15%
- SVM performs slightly worse than the Decision Tree. Its precision and recall are low for all classes, particularly for "Diesel," "Electric," and "Hybrid." The model seems to have trouble distinguishing between the fuel types and has a **low recall** for the majority of classes, particularly "Diesel."

**Logistic Regression**:

- **Accuracy**: 25.75%
- Logistic Regression performs the worst among all three models. It completely fails at predicting "Diesel" (precision and recall of 0), and the overall recall for the other classes is poor. This suggests that the model cannot effectively capture the relationships in the data for fuel type classification.

## POST LAB EXERCISE

## QUESTIIONS:

1. Mention the tools that are used for estimating the accuracy of the classification algorithm.

- `accuracy_score`
- `classification_report`
- `confusion_matrix`
- `ConfusionMatrixDisplay`

**ASSESSMENT**

| Description | Max Marks | Marks Awarded |
|---|---|---|
| Pre Lab Exercise | **5** | |
| In Lab Exercise | **10** | |
| Post Lab Exercise | **5** | |
| Viva | **10** | |
| **Total** | **30** | |
| **Faculty Signature** | | |