Name:Kaushik Kotian Roll no.:29 Div:D20B Batch:B

AIDS- 2 Experiment No.12

Aim : To evaluate the performance of different classification algorithms (Logistic Regression, Decision Tree, Random Forest, and Support Vector Machine).

Theory:

Classification is a supervised learning technique used to predict the categorical label of a given dataset. In fraud detection, we aim to classify whether a transaction is fraudulent or not based on historical data.

Various classification algorithms are used for this task, each having its strengths and weaknesses:

1. Logistic Regression: A linear model used for binary classification, useful for its simplicity and interpretability.
2. Decision Tree: A non-linear model that splits data into subsets based on the most informative features, making it easy to visualize and interpret.
3. Random Forest: An ensemble method that builds multiple decision trees and averages their predictions, which improves accuracy and reduces overfitting.
4. Support Vector Machine (SVM): A model that finds the optimal hyperplane to classify the data, effective in high-dimensional spaces.

These models are evaluated using the following metrics:

- Accuracy: Proportion of correct predictions.
- Precision: Proportion of correctly predicted positive cases out of all predicted positives.
- Recall: Proportion of actual positive cases correctly identified.
- F1-Score: A harmonic mean of precision and recall, providing a balanced measure when dealing with imbalanced data.

Steps to Implement:

1. **Import Required Libraries**: You'll need libraries for data manipulation, model building, and evaluation metrics.
2. **Load and Preprocess the Data**: This involves:
    - Removing irrelevant columns.
    - Converting categorical columns to numerical values using label encoding.
    - Splitting the dataset into training and test sets.
3. **Train and Evaluate Models**: Train various classification algorithms (Logistic Regression, Decision Tree, Random Forest, and SVM) and compute performance

metrics like accuracy, precision, recall, and F1 score.

4. **Compare Results**: Collect and compare the results of all the classifiers.

Code Implementation:

```
# Split the data into training and testing sets (70% train, 30% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Step 3: Initialize the classifiers
classifiers = {
    'Logistic Regression': LogisticRegression(max_iter=1000),
    'Decision Tree': DecisionTreeClassifier(),
    'Random Forest': RandomForestClassifier(),
    'Support Vector Machine': SVC()
}

# Dictionary to store evaluation results
evaluation_results = {}

# Step 4: Train, predict and evaluate each model
for name, clf in classifiers.items():
    # Train the model
    clf.fit(X_train, y_train)

    # Predict on the test set
    y_pred = clf.predict(X_test)

    # Calculate evaluation metrics
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred, zero_division=1)  # Handle any undefined precision issues
    recall = recall_score(y_test, y_pred)
    f1 = f1_score(y_test, y_pred)
```

```
    # Store the results
    evaluation_results[name] = {
        'Accuracy': accuracy,
        'Precision': precision,
        'Recall': recall,
        'F1 Score': f1
    }

# Step 5: Display the evaluation results
for model, metrics in evaluation_results.items():
    print(f"Model: {model}")
    for metric, value in metrics.items():
        print(f"{metric}: {value:.4f}")
    print("\n")
```

Output:

```
Model: Logistic Regression
Accuracy: 0.9384
Precision: 1.0000
Recall: 0.0000
F1 Score: 0.0000


Model: Decision Tree
Accuracy: 0.8947
Precision: 0.1921
Recall: 0.2211
F1 Score: 0.2055
```

```
Model: Random Forest
Accuracy: 0.9388
Precision: 1.0000
Recall: 0.0070
F1 Score: 0.0139


Model: Support Vector Machine
Accuracy: 0.9384
Precision: 1.0000
Recall: 0.0000
F1 Score: 0.0000
```

The performance of each classification model can be analyzed using the output metrics (accuracy, precision, recall, F1-score) to determine which model performs best for fraud detection in this specific dataset.

Based on the output, we can conclude:

- Logistic Regression might perform well when the relationship between features is linear.
- Decision Tree provides easily interpretable results but may overfit without tuning. ● Random Forest generally offers better accuracy due to its ensemble nature and can handle overfitting.
- SVM performs well in high-dimensional spaces but may require more computational resources and parameter tuning.

We can analyze these metrics to choose the best classifier based on the trade-off between precision and recall, especially if false positives (incorrectly labeling a legitimate case as fraud) or false negatives (missing actual fraud) have significant consequences.

Conclusion:

We performed different classification algorithms like Logistic Regression, Decision Tree, Random Forest, and Support Vector Machine and compared them.