L & T

PROJECT

OIL SPILL CLASSIFICATION

BATCH :61

SHAIK MOHAMMAD SUHAIL

S.AYMAN TAHA

P.BEERAIAH

Y.SREEKANTH

Github: https://github.com/ShaikMohammadSuhail7/SVCET-ECE-BATCH-61-4.git

Github: https://github.com/ShaikMohammadSuhail7/SVCET-ECE-BATCH-61-4.git

Oil Spill Classification Project

1. AIM OF THE PROJECT

The aim of this project is to predict and classify oil spills using machine learning algorithms.

2. UNIQUENESS/DIFFERENTIATOR TRIED OUT IN THE PROJECT

The project leverages a Random Forest model with advanced feature engineering techniques like Polynomial Features and Principal Component Analysis (PCA).

3. INFERENCE

The tried-out SVM model has an accuracy of 0.58, while the Random Forest classifier has a significantly higher accuracy of 0.99.

4. METRICS OF THE PROJECT

The following metrics were used to evaluate the models:

Model accuracy for Logistic Regression: 0.75

Model accuracy for Random Forest Classifier: 0.81

Model accuracy for KNeighbors Classifier: 0.77

Model accuracy for SVM: 0.74

Model accuracy for Linear Regression: 0.9976

5. METRICS

The calculated metrics include:

Precision-Recall AUC: 0.9935

Precision: 1

Accuracy: 0.8000

6. Live Prediction

To perform live predictions, use the following code snippet to set up and execute predictions with the Random Forest Classifier, which achieved the highest accuracy.

Github: https://github.com/ShaikMohammadSuhail7/SVCET-ECE-BATCH-61-4.git

```
from sklearn.model_selection import StratifiedKFold, cross_validate
from sklearn.metrics import average_precision_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler, PolynomialFeatures
from imblearn.over_sampling import SMOTE
from imblearn.pipeline import Pipeline as ImbPipeline
from sklearn.compose import ColumnTransformerf1
from sklearn.decomposition import PCA
import numpy as np

print("Import successful.")

# StratifiedKFold to ensure each fold is representative of the overall class distribution
cv = StratifiedKFold(n_splits=3, shuffle=True, random_state=42)
print("StratifiedKFold initialized.")

# Define the preprocessing steps
preprocessor = ColumnTransformer(
    transformers=[
        ('poly', PolynomialFeatures(degree=2), slice(0, X.shape[1])), # Apply Polynomial Features to
all columns
        ('pca', PCA(n_components=5), slice(0, X.shape[1]))          # Add PCA to reduce dimensionality
    ],
    remainder='passthrough'
)

# Define RandomForest model
model = RandomForestClassifier(n_estimators=100, max_depth=10, random_state=42)

# Define the pipeline with SMOTE, StandardScaler, and RandomForest
pipeline = ImbPipeline(steps=[
    ('smote', SMOTE(random_state=42)),  # Handle imbalanced data
    ('preprocessor', preprocessor),     # Advanced feature engineering with PCA and
PolynomialFeatures
    ('scaler', StandardScaler()),       # Scaling features
    ('model', model)                    # Use RandomForestClassifier
])

# Cross-validated predictions
results = cross_validate(pipeline, X, y, cv=cv, scoring='average_precision', return_estimator=True)

# Aggregate predictions
y_pred_prob = np.zeros(len(y))
for est in results['estimator']:
    y_pred_prob += est.predict_proba(X)[:, 1]
y_pred_prob /= len(results['estimator'])

# Calculate Precision-Recall AUC
pr_auc = average_precision_score(y, y_pred_prob)
print(f'Precision-Recall AUC: {pr_auc:.4f}')
```

Code Explanation :

1. **Importing Libraries:**
   - StratifiedKFold and cross_validate from sklearn.model_selection: Used for cross-validation.
   - average_precision_score from sklearn.metrics: Used to calculate the Precision-Recall AUC.
   - RandomForestClassifier from sklearn.ensemble: The machine learning model used for classification.
   - StandardScaler and PolynomialFeatures from sklearn.preprocessing: Used for scaling features and generating polynomial features.
   - SMOTE from imblearn.over_sampling: Used to handle imbalanced data by oversampling the minority class.
   - Pipeline from imblearn.pipeline: Used to create a pipeline for the preprocessing steps and the model.
   - ColumnTransformer from sklearn.compose: Used to apply different preprocessing steps to different columns.
   - PCA from sklearn.decomposition: Used for dimensionality reduction.
   - numpy: A library for numerical operations.
2. **StratifiedKFold Initialization:**
   - StratifiedKFold(n_splits=3, shuffle=True, random_state=42): Initializes StratifiedKFold to ensure each fold is representative of the overall class distribution.
3. **Preprocessing Steps:**
   - ColumnTransformer: Applies Polynomial Features and PCA to all columns.
     - PolynomialFeatures(degree=2): Generates polynomial features of degree 2.
     - PCA(n_components=5): Reduces the dimensionality to 5 principal components.
   - remainder='passthrough': Ensures that any columns not specified in the transformers are passed through without transformation.
4. **RandomForest Model:**
   - RandomForestClassifier(n_estimators=100, max_depth=10, random_state=42): Initializes the Random Forest classifier with 100 trees and a maximum depth of 10.

5. **Pipeline Definition:**
    - ImbPipeline: Creates a pipeline with the following steps:
        - SMOTE(random_state=42): Handles imbalanced data by oversampling the minority class.
        - preprocessor: Applies the preprocessing steps defined earlier.
        - StandardScaler(): Scales the features.
        - model: Uses the Random Forest classifier.
6. **Cross-Validated Predictions:**
    - cross_validate(pipeline, X, y, cv=cv, scoring='average_precision', return_estimator=True): Performs cross-validation and returns the estimators.
7. **Aggregating Predictions:**
    - y_pred_prob = np.zeros(len(y)): Initializes an array to store the aggregated predictions.
    - for est in results['estimator']: Iterates over the estimators from cross-validation.
        - y_pred_prob += est.predict_proba(X)[:, 1]: Adds the predicted probabilities for the positive class.
    - y_pred_prob /= len(results['estimator']): Averages the predicted probabilities.
8. **Calculating Precision-Recall AUC:**
    - average_precision_score(y, y_pred_prob): Calculates the Precision-Recall AUC.
    - print(f'Precision-Recall AUC: {pr_auc:.4f}'): Prints the Precision-Recall AUC.

This code demonstrates how to preprocess data, handle imbalanced data, apply advanced feature engineering techniques, train a Random Forest classifier, and evaluate its performance using cross-validation and Precision-Recall AUC. If you have any more questions or need further clarification, feel free to ask!

7. Conclusion :

This project successfully demonstrates the use of machine learning algorithms to classify oil spills. The Random Forest model achieved the highest accuracy, while the SVM model provided valuable insights despite its lower accuracy. The metrics used for evaluation include accuracy, precision, and Precision-Recall AUC, offering a comprehensive view of the model's performance.