Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №8
з дисципліни
Аналіз даних з використанням мови Python

Виконав:

студент групи ІА-24:
Криворучек В.С.

Перевірила:

ст. викладач
Тимофєєва Ю.С.

Київ 2025

Тема: Класифікація в scikit-learn

Мета роботи: Ознайомитись з побудовою моделей для вирішення задачі класифікації в scikit-learn, оцінкою та способами покращення результатів.

Хід роботи

Завдання:

penguins.csv.

Написати програму, яка навчає та тестує не менше трьох моделей, що виконують задачу бінарної класифікації відповідно до варіанту, оцінити моделі за допомогою відповідних метрик та спробувати покращити результати за допомогою:

а) підбору гіперпараметрів моделі;

б) вибору ознак або створення нових;

в) проєктування ознак на нижчу розмірність.

Код програми:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.decomposition import PCA
from sklearn.metrics import classification_report
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
import warnings

warnings.filterwarnings('ignore')

# === 1. Завантаження даних ===
data = pd.read_csv("penguins.csv")

# === 2. Попередня обробка ===
data.dropna(inplace=True)  # Видаляємо пропущені значення
data['target'] = (data['species'] == 'Adelie').astype(int)  # Бінарна цільова змінна

# One-hot encoding для категоріальних змінних
data = pd.get_dummies(data, columns=['island', 'sex'], drop_first=True)

# Розділення на ознаки та ціль
```

```
X = data.drop(columns=['species', 'target'])
y = data['target']

# === 3. Розділення на тренувальну та тестову вибірки ===
X_train, X_test, y_train, y_test = train_test_split(
    X, y, stratify=y, test_size=0.2, random_state=42
)

# === 4. Побудова та оцінка трьох моделей ===
models = {
    "Logistic Regression": LogisticRegression(max_iter=1000),
    "Random Forest": RandomForestClassifier(),
    "SVM": SVC()
}

print("\n=== Базові моделі ===")
for name, model in models.items():
    pipeline = Pipeline([
        ('scaler', StandardScaler()),
        ('classifier', model)
    ])
    pipeline.fit(X_train, y_train)
    y_pred = pipeline.predict(X_test)
    print(f"\n{name}:\n")
    print(classification_report(y_test, y_pred))

# === 5. Покращення: підбір гіперпараметрів Random Forest ===
print("\n=== Підбір гіперпараметрів для Random Forest ===")
param_grid = {
    'classifier__n_estimators': [50, 100, 200],
    'classifier__max_depth': [None, 5, 10]
}
rf_pipeline = Pipeline([
    ('scaler', StandardScaler()),
    ('classifier', RandomForestClassifier())
])
grid_search = GridSearchCV(rf_pipeline, param_grid, cv=5, scoring='f1')
grid_search.fit(X_train, y_train)
best_rf = grid_search.best_estimator_
y_pred_rf = best_rf.predict(X_test)

print("Найкращі параметри:", grid_search.best_params_)
print("\nОцінка кращої моделі Random Forest:\n")
print(classification_report(y_test, y_pred_rf))

# === 6. Зменшення розмірності за допомогою PCA ===
print("\n=== Logistic Regression з PCA ===")
pca_pipeline = Pipeline([
    ('scaler', StandardScaler()),
    ('pca', PCA(n_components=2)),
    ('classifier', LogisticRegression())
])
pca_pipeline.fit(X_train, y_train)
y_pred_pca = pca_pipeline.predict(X_test)
```

```
print(classification_report(y_test, y_pred_pca))
```

Результати виконання коду:

```
=== Базові моделі ===

Logistic Regression:

              precision    recall  f1-score   support

           0       1.00      0.97      0.99        39
           1       0.97      1.00      0.98        30

    accuracy                           0.99        69
   macro avg       0.98      0.99      0.99        69
weighted avg       0.99      0.99      0.99        69


Random Forest:

              precision    recall  f1-score   support

           0       0.97      0.97      0.97        39
           1       0.97      0.97      0.97        30

    accuracy                           0.97        69
   macro avg       0.97      0.97      0.97        69
weighted avg       0.97      0.97      0.97        69
```

```
SVM:

              precision    recall  f1-score   support

           0       1.00      1.00      1.00        39
           1       1.00      1.00      1.00        30

    accuracy                           1.00        69
   macro avg       1.00      1.00      1.00        69
weighted avg       1.00      1.00      1.00        69


=== Підбір гіперпараметрів для Random Forest ===
Найкращі параметри: {'classifier__max_depth': 10, 'classifier__n_estimators': 50}

Оцінка кращої моделі Random Forest:

              precision    recall  f1-score   support

           0       1.00      0.97      0.99        39
           1       0.97      1.00      0.98        30

    accuracy                           0.99        69
   macro avg       0.98      0.99      0.99        69
weighted avg       0.99      0.99      0.99        69
```

```
=== Logistic Regression з PCA ===
              precision    recall  f1-score   support

           0       0.88      0.90      0.89        39
           1       0.86      0.83      0.85        30

    accuracy                           0.87        69
   macro avg       0.87      0.87      0.87        69
weighted avg       0.87      0.87      0.87        69


Process finished with exit code 0
```

Висновок: У ході виконання даної лабораторної роботи я ознайомився з побудовою моделей для вирішення задачі класифікації в scikit-learn, оцінкою та способами покращення результатів.