Міністерство освіти і науки України  
НТУУ «КПІ ім. Ігоря Сікорського»

Навчально-науковий інститут атомної та теплової енергетики

Кафедра цифрових технологій в енергетиці

Лабораторна робота №4

з дисципліни «Вступ до інтелектуального аналізу даних»

Тема ««Машинне навчання в Python»

Варіант №16

Виконав:

студент 3-го курсу, НН ІАТЕ гр. ТР-23

Ровний Григорій

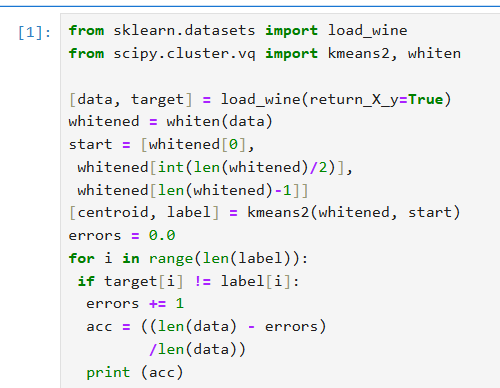
Перевірив: проф. Путренко В.В.

КИЇВ 2025

**Мета:** Опрацювати приклад роботи з машинним навчанням, використовуючи Jupyter Notebook.

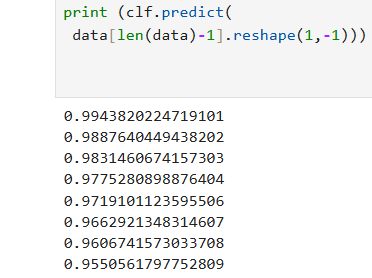
**Хід роботи**

**Метод K-Means.**

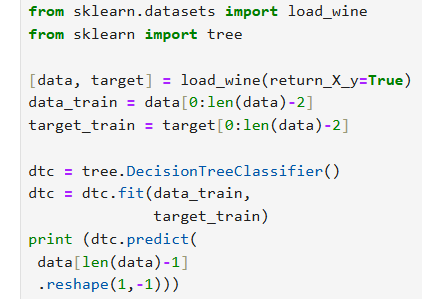


Даний програмний код використовує алгоритм K-means для кластеризації даних, що стосуються вин. Для реалізації цього використовуються бібліотеки scikit-learn та scipy. В першу чергу дані про вина завантажуються з набору даних за допомогою load\_wine з scikit-learn, а саме load\_wine(return\_X\_y=True) повертає дані та мітки класів. Далі, за допомогою функції whiten з бібліотеки scipy, дані "відбілюються", тобто стандартизуються для полегшення процесу кластеризації. Початкові центри кластерів обираються з відбіленого набору даних, і алгоритм K-means використовується для кластеризації. Результати цього процесу - центроїди кластерів та мітки для кожного зразка - зберігаються у відповідних змінних. Після кластеризації виконується порівняння справжніх міток класів і отриманих міток від K-means. Якщо вони не співпадають, збільшується лічильник помилок. Далі, розраховується точність кластеризації (acc), яка представляє собою відношення кількості правильно класифікованих зразків до загальної кількості зразків. Отримані точності виводяться на екрані.

Результат:



**Побудова моделі класифікації за допомогою дерева рішень.**

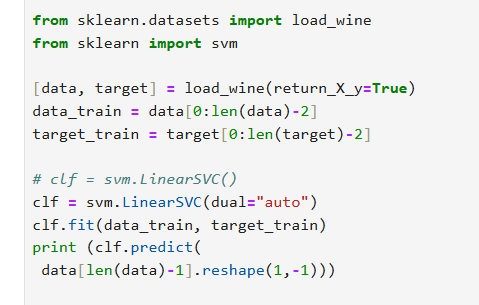


Цей код демонструє простий приклад використання дерева рішень для класифікації даних про вина. Тут використовуємо бібліотеки scikit-learn для завантаження даних про вина (wine dataset) та побудови моделі класифікації з використанням дерева рішень. Спочатку, дані та їх мітки завантажуються з набору даних про вина за допомогою load\_wine(return\_X\_y=True). Потім дані розділяються на тренувальний набір (data\_train, target\_train) та тестовий зразок. У цьому випадку, останні два зразки відокремлюються для тестування. Після цього створюється модель класифікатора за допомогою дерева рішень (DecisionTreeClassifier()), яка потім навчається на тренувальних даних (fit(data\_train, target\_train)). Остання стрічка коду використовує навчену модель для передбачення класу останнього зразка у тестовому наборі. Результат передбачення виводиться на екран за допомогою print(dtc.predict(data[len(data)-1].reshape(1, -1))).

Результат:



**Побудова моделі класифікації з використанням методу опорних векторів (SVM).**



Дані та їх мітки завантажуємо з набору даних про вина за допомогою load\_wine(return\_X\_y=True). Далі дані розділяються на тренувальний набір (data\_train, target\_train). У цьому випадку, останні два зразки відокремлюються для тестування. Модель класифікатора створюється за допомогою методу опорних векторів (SVM) з використанням класу LinearSVC. Опція dual="auto" вказує системі самостійно вибрати оптимальний метод для розв'язку двоїстої задачі оптимізації. Навчання моделі виконується за допомогою методу fit(data\_train, target\_train), де дані навчаються на тренувальному наборі. Остання стрічка коду використовує навчену модель для передбачення класу останнього зразка у тестовому наборі. Результат передбачення виводиться на екран за допомогою print(clf.predict(data[len(data)-1].reshape(1, -1))). Метод опорних векторів (SVM) використовує геометричний підхід для класифікації даних. У випадку лінійного ядра, як в даному випадку з параметром LinearSVC, модель намагається розділити класи гіперплощиною так, щоб максимізувати відстань між класами. Результат передбачення вказує на клас, до якого належить останній зразок у тестовому наборі.

Результат:



**Висновок:**

Під час виконання лабораторної роботи було набуто практичні навички роботи з методами, які є складовими сімейства алгоритмів машинного навчання та використовуються для різних завдань, таких як кластеризація, класифікація та регресія, а саме: K-means, SVM (Support Vector Machine) та дерево рішень.

**Програмний код:**

from sklearn.datasets import load\_wine  
from scipy.cluster.vq import kmeans2, whiten  
  
[data, target] = load\_wine(return\_X\_y=True)  
whitened = whiten(data)  
start = [whitened[0],  
 whitened[int(len(whitened)/2)],  
 whitened[len(whitened)-1]]  
[centroid, label] = kmeans2(whitened, start)  
errors = 0.0  
for i in range(len(label)):  
 if target[i] != label[i]:  
 errors += 1  
 acc = ((len(data) - errors)  
 /len(data))  
 print (acc)  
from sklearn.datasets import load\_wine  
from sklearn import tree  
  
[data, target] = load\_wine(return\_X\_y=True)  
data\_train = data[0:len(data)-2]  
target\_train = target[0:len(data)-2]  
  
dtc = tree.DecisionTreeClassifier()  
dtc = dtc.fit(data\_train,  
 target\_train)  
print (dtc.predict(  
 data[len(data)-1]  
 .reshape(1,-1)))  
  
from sklearn.datasets import load\_wine  
from sklearn import svm  
  
[data, target] = load\_wine(return\_X\_y=True)  
data\_train = data[0:len(data)-2]  
target\_train = target[0:len(target)-2]  
  
# clf = svm.LinearSVC()  
clf = svm.LinearSVC(dual="auto")  
clf.fit(data\_train, target\_train)  
print (clf.predict(  
 data[len(data)-1].reshape(1,-1)))