مشروع نهاية الفصل 8

مقرر Machine Learning — Dimensionality Reduction — مشروع تقليل الأبعاد

الهدف: تشوف هل تقليل الأبعاد بالـ PCA بيفيد فعلاً، وتختار عدد الأبعاد بالـ cross-validation.

الداتا: sklearn.datasets.load_digits (١٧٩٧ صورة، ٦٤ بكسل). مصدرها UCI. مدمجة في scikit-learn.

الأسئلة العملية

1 جهّز (Prepare)

- حمّل digits وقسّمها ٨٠/٢٠ بـ stratify و random_state=0.

2 بدون PCA (No PCA)

- درّب SVC واطبع دقة الـ test.

3 كام بُعد؟ (How many dimensions?)

- اعمل PCA على الـ training واطبع عدد الـ components اللازم لـ ٩٠٪ و ٩٥٪ و ٩٩٪ variance.

4 Pipeline (Pipeline)

- جرّب PCA(n_components=n) + SVC لـ n = 10, 20, 30 بـ cv=3 على الـ training.

5 النهائي (Final)

- اختار أحسن n وقيس الـ test.

بونس

- بونس: اضغط صورة وفكّها بـ inverse_transform واطبع الخطأ.
- بونس: جرّب t-SNE للرسم بس.

أسئلة تفكير (اكتب إجابة في سطرين)

1. ليه بنعمل PCA على الـ training بس؟
2. إيه الفرق بين PCA للضغط وt-SNE للرسم؟

ماذا تسلّم؟

- ملف الكود (أو notebook) بيشتغل من الأول للآخر.
- فقرة قصيرة (٥ أسطر) بتلخّص النتيجة وإيه اللي اتعلمته.

الحل النموذجي

جرّب بنفسك الأول، وبعدين قارن. الأرقام الحقيقية في النتيجة تحت.

الكود

```
# dataset ships inside scikit-learn (no download needed)
import numpy as np
from sklearn.datasets import load_digits
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.decomposition import PCA
from sklearn.pipeline import make_pipeline
from sklearn.svm import SVC

X, y = load_digits(return_X_y=True)
X_tr, X_te, y_tr, y_te = train_test_split(X, y, test_size=0.2, random_state=0,
stratify=y)
print('no PCA test accuracy:', round(SVC().fit(X_tr, y_tr).score(X_te, y_te), 3))
cum = np.cumsum(PCA().fit(X_tr).explained_variance_ratio_)
for v in (0.90, 0.95, 0.99):
    print(int(v * 100), '% variance needs', int(np.argmax(cum >= v)) + 1, 'components')
res = {n: cross_val_score(make_pipeline(PCA(n_components=n), SVC()), X_tr, y_tr,
cv=3).mean() for n in (10, 20, 30)}
for n, s in res.items():
    print('n =', n, 'cv accuracy', round(s, 3))
best = max(res, key=res.get)
final = make_pipeline(PCA(n_components=best), SVC()).fit(X_tr, y_tr)
print('best n:', best, '| test accuracy:', round(final.score(X_te, y_te), 3))
```

النتيجة

```
no PCA test accuracy: 0.983
90 % variance needs 21 components
95 % variance needs 28 components
99 % variance needs 41 components
n = 10 cv accuracy 0.966
n = 20 cv accuracy 0.989
n = 30 cv accuracy 0.988
best n: 20 | test accuracy: 0.989
```

ملاحظات على الإجابات

- الـ PCA بيتعمل على الـ training بس عشان ما نسرّبش معلومات من الـ test.

- الـ PCA بيضغط ويحافظ على الـ variance، والـ t-SNE بيعمل رسم يظهر المجموعات ومش مناسب كـ features.
- قارن دقة الـ pipeline بالدقة من غير PCA: لو قريبة، التقليل مفيد للسرعة.

Credits: topic follows the matching chapter of Aurélien Géron, Hands-On Machine Learning (3rd ed.). Questions, code and explanations are original.