مشروع نهاية الفصل 2

مقرر End-to-End Machine Learning Project — Machine Learning — مشروع Machine Learning من الأول للآخر

الهدف: تمشي كل خطوات المشروع على داتا انحدار (regression): تحديد المشكلة، تقسيم الداتا، baseline، تجهيز، مقارنة نماذج بالـ cross-validation، واختبار نهائي واحد.

الداتا: sklearn.datasets.load_diabetes (٤٤٢ مريض، ١٠ خصائص، الهدف: تطوّر المرض بعد سنة). Efron وآخرون ٢٠٠٤. مدمجة في scikit-learn.

الأسئلة العملية

1 حدّد المشكلة (Frame the problem)

- إيه نوع المشكلة؟ وإيه مقياس الأداء المناسب (RMSE ولا MAE)؟ ليه؟
- اكتب جملة: مين ممكن يستخدم التوقّع ده؟

2 الداتا (Look at the data)

- حمّل الداتا واطبع شكلها، وأقل وأكبر قيمة للـ target.

3 قسّم الداتا (Split)

- قسّم ٨٠/٢٠ بـ random_state=42 وخلّي الـ test على جنب.

4 الـ Baseline (Baseline)

- احسب RMSE لنموذج بيتوقّع متوسط الـ training دايماً.

5 قارن نموذجين (Compare models)

- ابني Pipeline (StandardScaler + Ridge) و Pipeline (StandardScaler + RandomForestRegressor).
- قارنهم بـ 5-fold cross-validation على الـ training بمقياس RMSE.

6 الاختبار النهائي (Final test)

- اختار الأحسن، درّبه على كل الـ training، وقيسه على الـ test مرة واحدة.

بونس

- بونس: جرّب GridSearchCV على alpha للـ Ridge (0.1, 1, 10, 100) وشوف إذا كان بيحسّن.
- بونس: اطبع أهم ٣ features في الـ RandomForest.

أسئلة تفكير (اكتب إجابة في سطرين)

1. ليه بنبدأ بـ baseline بسيط؟
2. ليه الـ test set بيتستخدم مرة واحدة بس؟

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_diabetes
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import Ridge
from sklearn.ensemble import RandomForestRegressor

X, y = load_diabetes(return_X_y=True)
print('rows, features:', X.shape, '| target range:', y.min(), y.max())
X_tr, X_te, y_tr, y_te = train_test_split(X, y, test_size=0.2, random_state=42)
rmse = lambda t, p: float(np.sqrt(np.mean((t - p) ** 2)))
print('baseline RMSE:', round(rmse(y_te, np.full(len(y_te), y_tr.mean())), 1))
models = {'ridge': make_pipeline(StandardScaler(), Ridge()), 'forest':
make_pipeline(StandardScaler(), RandomForestRegressor(n_estimators=100,
random_state=0))}
cv = {n: -cross_val_score(m, X_tr, y_tr, cv=5,
scoring='neg_root_mean_squared_error').mean() for n, m in models.items()}
for n, v in cv.items():
    print(n, 'cv RMSE', round(v, 1))
best = min(cv, key=cv.get)
print('chosen:', best)
print('test RMSE:', round(rmse(y_te, models[best].fit(X_tr, y_tr).predict(X_te)), 1))
```

النتيجة

```
rows, features: (442, 10) | target range: 25.0 346.0
baseline RMSE: 73.2
ridge cv RMSE 55.9
forest cv RMSE 59.0
chosen: ridge
test RMSE: 53.8
```

ملاحظات على الإجابات

- نوع المشكلة: Supervised regression لأننا بنتوقّع رقم. الـ RMSE بيعاقب الأخطاء الكبيرة أكتر، والـ MAE أقل حساسية للقيم الشاذة.

- الـ baseline بيدّينا رقم لازم نتفوّق عليه، ولو النموذج مش أحسن منه يبقى مفيش داعي له.
- قارن الناتج: لو الـ Ridge والـ forest قريبين، فضّل الأبسط والأسهل في الشرح.
- الـ test بيتقاس مرة واحدة عشان ما نختارش النموذج بناءً عليه، وإلا النتيجة تبقى متفائلة.

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