**تقرير مشروع BIA601: نظام ويب لاختيار الخصائص باستخدام الخوارزمية الجينية**

**رابط الحل :**

**الملخص التنفيذي**

هذا المشروع يقدّم تطبيق ويب (Django) موجه بالعربية يقوم بتحميل بيانات CSV، وتنفيذ عملية اختيار الخصائص (Feature Selection) باستخدام الخوارزمية الجينية (Genetic Algorithm - GA)، ومقارنة النتائج مع طرق إحصائية/كلاسيكية بديلة.  
يركّز النظام على **الصحة، الوضوح، وتجربة المستخدم**، ويقوم بحفظ النتائج في ملفات JSON منظمة لتسهيل التتبع والتحليل لاحقًا.

**أبرز النتائج:**

* محرك GA قابل لإعادة الاستخدام لتنفيذ اختيار الخصائص (engine/gafs.py).
* خط تقييم متكامل باستخدام Cross-validation مع اختيار تلقائي للمقياس المناسب حسب نوع المهمة.
* طرق بديلة (Baselines) لتنفيذ مقارنة عادلة في baselines/selectors.py.
* واجهة عربية كاملة باستخدام Django (رفع البيانات – تشغيل الخوارزمية – عرض النتائج).
* اختبارات مبدئية (tests/test\_ga.py) للتأكد من عمل GA في كل من التصنيف والانحدار.

**الأهداف والمتطلبات**

* تمكين المستخدم من رفع بيانات CSV، اختيار العمود الهدف، وتحديد نوع المهمة (تصنيف/انحدار).
* تشغيل خوارزمية GA مع معاملات (Hyperparameters) قابلة للتخصيص.
* مقارنة نتائج GA مع الطرق الإحصائية التقليدية.
* واجهة عربية بسيطة وسهلة الاستخدام.
* حفظ النتائج في ملفات قابلة للنقل والتوثيق.

**نظرة عامة على النظام**

يتكون النظام من ثلاث طبقات رئيسية:

1. **الطبقة الخوارزمية**
   * : engine/gafs.pyمحرك GA (GeneticFeatureSelector) ونتائج (GAResult).
   * baselines/selectors.py:

طرق بديلة (SelectKBest, Chi2, RFE, SelectFromModel)

* + utils/preprocessing.pyتحميل CSV، معالجة القيم المفقودة، استخراج الأعمدة الرقمية، واستنتاج نوع المهمة.

1. **طبقة الويب (Django)**
   * selector/forms.pyنموذج رفع البيانات وضبط إعدادات GA.
   * selector/views.pyدوال العرض (Upload – Run – Results).
   * urls.pyضبط المسارات.
   * settings.py دعم اللغة العربية والإعدادات الأساسية.
2. **التخزين**
   * الملفات المرفوعة: MEDIA\_ROOT/uploads/.
   * النتائج: MEDIA\_ROOT/results/.

**تصميم الخوارزمية الجينية (GA)**

* **تمثيل الكروموسوم**: قناع ثنائي (1 = خاصية مختارة، 0 = غير مختارة).
* **التقييم (Fitness)**:
  + التصنيف الثنائي: roc\_auc.
  + التصنيف متعدد الفئات: roc\_auc\_ovr.
  + الانحدار: r2.
  + مع عقوبة لحجم المجموعة المختارة (feature\_cost).
  + التخزين المؤقت للنتائج (Memoization) لتجنب إعادة الحساب.
* **العمليات الجينية**:
  + الاختيار: Tournament Selection.
  + التزاوج: Uniform Crossover.
  + الطفرة: Bit-flip mutation.
  + الإصلاح: ضمان اختيار خاصية واحدة على الأقل.
  + النخبة: نقل أفضل الأفراد مباشرة.
  + الإيقاف المبكر: التوقف عند عدم تحسن الأداء.
* **النماذج الافتراضية**:
  + التصنيف: Pipeline(StandardScaler -> LogisticRegression).
  + الانحدار: Pipeline(StandardScaler -> Ridge).

**المعالجة المسبقة للبيانات**

* تحميل CSV والتأكد من وجود العمود الهدف.
* الاقتصار على الأعمدة الرقمية للنسخة الحالية.
* إسقاط الصفوف التي تحتوي على قيم هدف ناقصة.
* تعويض القيم المفقودة في الميزات الرقمية بالوسيط (Median).
* استنتاج نوع المهمة تلقائيًا (تصنيف/انحدار).
* ترميز الهدف الفئوي إذا لزم.

**التدفق في تطبيق الويب**

1. المستخدم يرفع ملف CSV ويحدد العمود الهدف والمعاملات.
2. يتم تحميل البيانات ومعالجتها (load\_csv).
3. تشغيل GA (GeneticFeatureSelector.fit).
4. تشغيل طرق بديلة للمقارنة (run\_baselines).
5. كتابة النتائج في JSON.
6. عرض صفحة النتائج للمستخدم.

**طرق المقارنة (Baselines)**

* SelectKBest باستخدام f\_classif أو f\_regression.
* Chi2 (للتصنيف فقط) بعد MinMaxScaler.
* RFE مع LogisticRegression أو Ridge.
* SelectFromModel مع LogisticRegression/Lasso أو RandomForest.

**بروتوكول التقييم**

* Cross-validation بعدد طيات cv قابل للتخصيص
* مقياس الأداء يعتمد على نوع المهمة.
* مقارنة عادلة: baselines تُقيّم بعدد خصائص مساوٍ لعدد خصائص GA.

**النتائج والتفسير**

* **GA** يعطي نتيجة مزدوجة:
  + الدرجة مع العقوبة (لاختيار الخصائص).
  + الدرجة الخام بدون عقوبة (للشفافية).
* **Baselines** تُستخدم كمرجع لأداء الطرق التقليدية.
* النتائج محفوظة في JSON لضمان التتبع وإعادة الإنتاج.

**الاختبارات وضمان الجودة**

* tests/test\_ga.py يختبر:
  + عمل GA على بيانات صناعية صغيرة.
  + صحة الأبعاد وعدد الخصائص المختارة.
  + تغطية لكل من التصنيف والانحدار.

**كيفية التشغيل محليًا**

1. إنشاء بيئة افتراضية وتثبيت المكتبات (pip install -r requirements.txt).
2. تنفيذ أوامر Django:
3. python manage.py migrate
4. python manage.py runserver
5. فتح المتصفح على http://127.0.0.1:8000/.
6. رفع البيانات ومشاهدة النتائج.
7. لتشغيل الاختبارات: pytest -q.

**القيود والأعمال المستقبلية**

* التركيز الحالي على الخصائص الرقمية فقط.
* دعم الخصائص الفئوية مستقبلاً (One-hot encoding).
* تنفيذ نظام مهام خلفي (Celery/RQ) لعرض شريط تقدم.
* إضافة رسومات لعملية GA وتقرير PDF.
* النشر على السحابة (مثل Azure/Render).

**الخاتمة**

النظام يحقق أهداف مشروع **BIA601** من خلال:

* دمج خوارزمية جينية قوية لاختيار الخصائص.
* واجهة عربية سهلة الاستخدام.
* مقارنة عادلة مع الطرق التقليدية.

هذا يضع أساسًا متينًا لتطويرات مستقبلية، وللاستخدام في أبحاث أو تطبيقات عملية.