

مشروع مقرر معالجة اللغات الطبيعية

استخراج المشاعر من النص

إشراف

د. ندى غنيم

م. زينة الدلال

إعداد الطلاب

حلا شبارة

عدي محمد

فادي الشعار

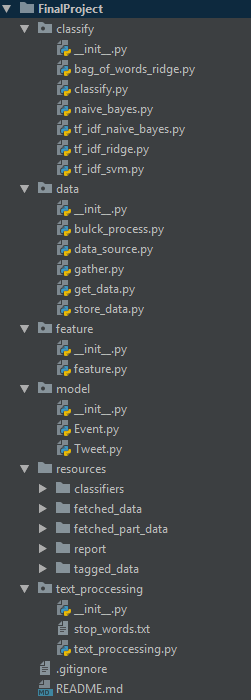
مقدمة:

يهدف هذا المشروع إلى تمكين الحاسب من استخراج المشاعر من النصوص، و يحتاج هذا النوع من المشاريع إلى حجم كبير نسبيا من العينات، و تم جمع العينات من موقع التواصل الإجتماعي Twitter، وهي عبارة عن عينات مكتوبة باللغة العربية العامية، حيث استهدفنا المملكة العربية السعودية و مصر لجمع العينات كونها من الدول العربية الأكثر استخداما لTwitter، ثم تم تطوير عدد من الclassifiers و تدريبها على قسم من العينات، بينما تم استخدام الجزء المتبقي في عملية الإختبار و تقييم النماذج.

الموارد المستخدمة:

استخدمت التقنيات و المكتبات التالية في إنجاز المشروع:

* لغة البرمجة python.
* مكتبة معالجة اللغات الطبيعية nltk.
* مكتبة Get Old Tweets لجمع العينات.
* مكتبة sklearn machine learning

بنية المشروع:

يتكون المشروع من مجموعة من الPackages:

* Classify

تتضمن الملفات المتعلقة بالتصنيف.

* Data

تتضمن الملفات المتعلقة بجمع العينات و برمترات

البحث عنها.

* Feature

تتضمن الملفات المتعلقة باستخراج الfeatures

من النص.

* Model:

تتضمن صفوف تمثل الmodels الخاصة بالمشروع.

* Text\_processing

تتضمن الملفات المتعلقة بمعالجة النص مثل عمليات

Normalization وtokenization.

* Resources

مجلد يتضمن الموارد الخاصة بالمشروع.

تم رفع المشروع إلى GitHub على الرابط:

<https://github.com/ODDdaaaaai/nlp>

مراحل العمل:

مر المشروع بمراحل العمل التالية:

1. جمع العينات.
2. تنظيم العينات و تنظيفها من النص غير المرغوب به.
3. تصنيف العينات يدويا عن طريق وضع tags لكل عينة.
4. استخراج الfeatures من العينات.
5. تدريب classifier يقوم بالتصنيف الآلي للعينات.
6. اختبار الmodels المقترحة و تقييم أدائها.

أولا: جمع العينات:

تم جمع العينات من موقع التواصل الاجتماعي Twitter باستخدام المكتبة Get Old Tweets و التي تعتمد على الSearch API الخاص بTwitter.

يحتاج استخراج المشاعر إلى مجموعة متنوعة من العينات للحصول على عينات تحتوي مشاعر متنوعة، لذلك كان من الأهمية تحديد الSearch criteria الخاصة بعملية البحث، مثلا:

للحصول على عينات تحوي مشاعر سعادة، بحثنا عن عينات نشرات خلال فترات الأعياد:

Event(['فطر', 'عيد', 'سعيد', 'كل عام و أنتم بخير'], '2018-6-14', '2018-6-16', ['مصر', 'السعودية'])

عينات غضب أو خوف أو حزن في فترات أحداث الثورة المصرية:

Event(['ميدان التحرير', 'جمعة الزحف', 'ثورة', 'تنحي'], '2011-2-11', '2011-2-12', ['مصر']),

و العديد من الsearch criteria الأخرى موجودة ضمن الملف data\_source.py.

نلاحظ أن المعلومات مخزة ضمن الصف Event الذي يحوي كلمات البحث و امتداد الفترة الزمنية التي يجب البحث ضمنها و موقع البحث.

يتم جمع العينات ضمن الملف get\_data.py باستخدام الsearch criteria السابقة، و يتم تخزين الداتا باستخدامpickle و بشكل تراكمي تفاديا لخسارة العينات في حال انقطاع الاتصال مثلا.

تم جمع 149,372 عينة فريدة بعد الانتهاء من جمع العينات.

ثانيا: Normalization:

يتم تنظيف العينات باستخدام التوابع الموجودة في الملف text\_proccess ووفق المراحل التالية:

1. الابقاء فقط على الأحرف العربية في النص أي الأحراف ضمن مجال Unicode: [0621-064A].
2. استبدال التاء المربوطة بالهاء.
3. استبدال الهمزات [ئ,ؤ,أ,إ,آ] بالهمزة "ء".
4. ازالة الأحرف الواردة بشكل متكرر مثل حالة: ههههههههههه.
5. ازلة الفراغات الزائدة.
6. ازالة الstop words.

تم تخزين العينات بعد تنظيفها باستخدام pickle و json.

ثالثا: تصنيف العينات يدويا:

و يعتبر الجزء الأكثر كلفة في المشروع، لذلك تم الاستعانة بالصيغة json لإنجاز هذه المهمة باستخدام json editor بدلا من استخدام الملفات النصية البسيطة.

تم تصنيف العينات باستخدام مجموعة من الtags حيث يعبر كل tag عن شعور معين:

* السعادة: H.
* الغضب: A.
* الحزن: S.
* الخوف: F.
* القلق: W.

لم نتمكن من تصنيف العينات بالكامل بسبب حجمها الكبير و تم تصنيف 25000 عينة يدويا.

رابعا: استخراج الfeatures:

الfeatures هي بجوهرها مجموعة من الكلمات التي تميز النص، و في حالتنا هي كلمة تعبر عن المشاعر الموجودة في النص، و تم استخدام عدة طرق لاستخراج الfeatures:

* Bag of Words
* TF\_IDF
* Custom method

جميع هذه الطرق موجودة على شكل توابع في الملف feature.py.

خامسا: تدريب المصنفات:

قمنا بتدريب المصنفات و بناء الmodels بعد استخراج الfeatures:

* Bag of Words + Ridge Classifier
* TF-IDF + Ridge Classifier
* Naive Bayes
* TF-IDF + Naïve Bayes
* TF-IDF + SVM

سادسا: تقييم المصنفات:

تم تقييم المصنفات لاختيار المصنف ذو الأداء الأفض و حصلنا على النتائج التالية:

|  |  |  |  |
| --- | --- | --- | --- |
| Recall | Precision | Accuracy | Classifier |
| 64% | 66% | 61.57% | TF\_IDF with Ridge Classifier |
| 62.61% | 68% | 64.67% | Bag of words with Ridge classifier |
| 90.36% | 30.81% | 64.8% | TF-IDF with Naive Bayes |
|  |  | 65.40% | Custom Features with Naive Bayes |
| 87.38% | 66.87% | 65.72% | TF-IDF with SVM |

نستنتج أن النموذج TF-IDF with SVM هو الأفضل، فعلا الرغم من أنه لا يمتلك الPrecision الأعلى لكن الRecall الخاص به يتفوق على باقي النماذج، و على الرغم من أن TF-IDF يمتلك recall أعلى إلا أن الPrecision الخاصة به منخفضة جدا مما يعطيfalse positives كثيرة.

لذلك يعد TF-IDF with SVM النموذج الأفضل، حيث:

Precision:

هو عدد التصنيفات الصحيحة للعينات الإيجابية على عدد التصنيفات الإيجابية للمصنف، أي تكون قيمة ال

Precision قليلة عندما يخطأ المصنف كثير اً بالتصنيفات الإيجابية أي FP و كبيرة عندما لا يخطأ فيها

كثيرا.

Recall*:*

*هو عدد التصنيفات الصحيحة الإيجابية على عدد العينات الإيجابية، أي تكون قيمة الrecall قليلة عندما يخطئ المصنف كثيراً بالتصنيفات السلبية أي FN و كبيرة عندما لا يخطئ بها كثيراً.*

مثال:

ليكن لدينا سيناريو:

**

Low precision, low recall:

أي إذا أردنا صيد السمك سنمسك عدد قليل من الأسماك بسبب قيمة ال recall المنخفضة و عدد كبير من

الطحالب فرضا بسبب قيمة ال precision المنخفضة

السيناريو الثاني:



Low precision, high recall

أي سنصيد عدد كبير من السمك بسبب ال recall العالي و كمية كبيرة من الطحالب بسبب ال precision القليل.

السيناريو الثالث:



High precision, high recall

أي سنصيد عدد قليل من السمك بسبب ال recall القليل وعدد قليل من الطحالب بسبب ال precision

العالي

السيناريو الرابع:



High precision, high recall

أي سنصيد كمية كبيرة من الأسماك بسبب ال recall العالي و كمية كبيرة من الطحالب بسبب

ال precision العالي.