HW2-Writeup 2018-01-25 01:56:12.251059

Yezheng Li, Daizhen Li

Baselines

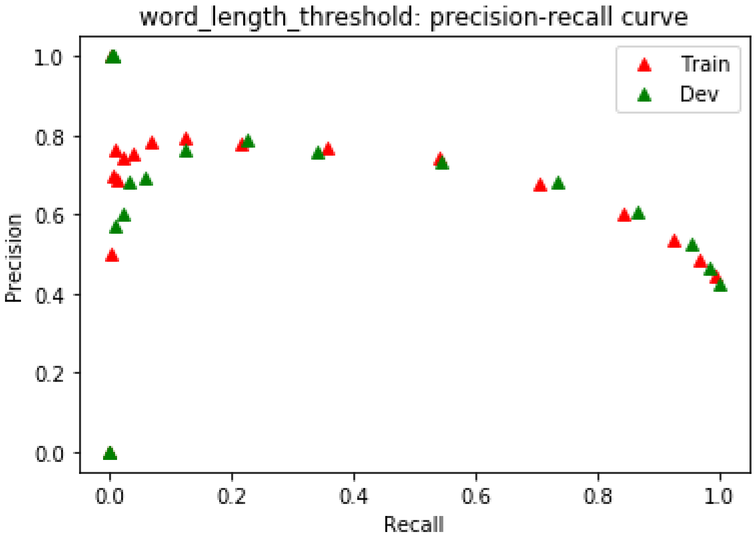
All-complex Baseline:

Train: precision 0.43275 recall 1.0 F-score 0.604083057058105

Dev: precision 0.418 recall 1.0 F-score 0.5895627644569816

Word-length Baseline:

Range of thresholds: 3 to 20 with optimal threshold: 7

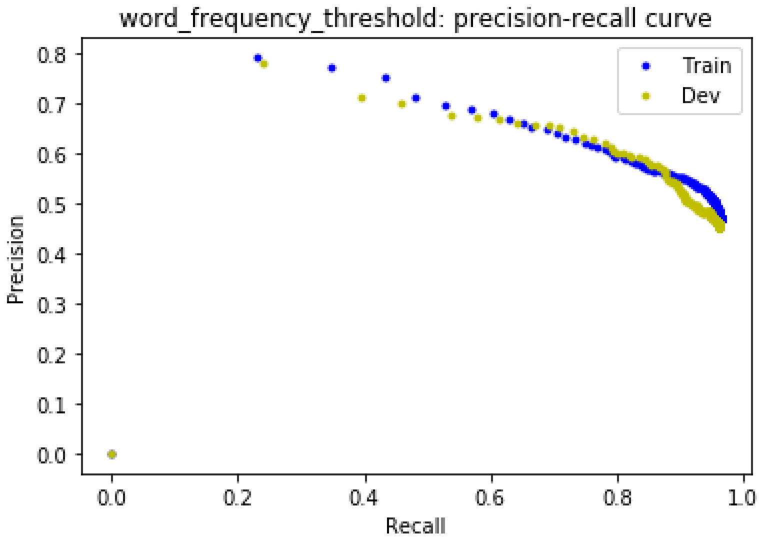


Train: precision 0.6007401315789473 recall 0.8440207972270364 F-score 0.7018976699495555

Dev: precision 0.6053511705685619 recall 0.8660287081339713 F-score 0.7125984251968505

Word-frequency Baseline:

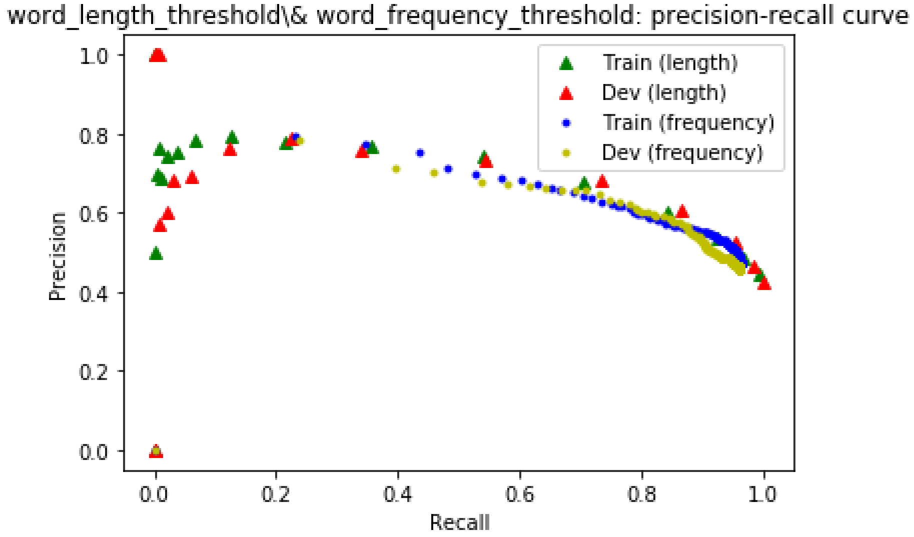
Range of thresholds: 137 to 1120679362 with optimal threshold: 19904037



Train: precision 0.6140191169776968 recall 0.7793183131138071 F-score 0.6868635437881874

Dev: precision 0.599644128113879 recall 0.80622009569378 F-score 0.6877551020408164

Plot Precison-Recall curve for various thresholds for both baselines together:



Which classifier looks better on average? I think word_length_threshold is better.

Naive Bayes:

Train: precision 0.867128827267 recall 0.5972940708316753 F-score 0.707351555137

Dev: precision 0.894736842105 recall 0.5917721518987342 F-score 0.712380952381

Logistic regression:

Train: precision 0.643558636626 recall 0.7159383033419023 F-score 0.677821721935

Dev: precision 0.684210526316 recall 0.7240506329113924 F-score 0.70356703567

Add a paragraph to your write up that discusses which model performed better on this task.

Although naive Bayes performslightly better in term of F-score, I think logistic regression have more balanced performance between precision and recall.

Build your own model

Please include a description of all features that you tried (not including length and frequency).

Besides length and frequency (with and without thresholds -- 4 features), I tried: [see preprocess_yezheng(...)]

--- count of character 'aeiou-' (I also tried string.ascii_lowercase,etc. but the latter has worse performance) -- 6 features;

--- count_syllables(...) -- 1 features

--- convolution of feature with itself: np.convolve(X,X) -- does not work

In all, there are 11 features

Please include a description of all models that you tried.

Besides improved Naive Bayes and improved logistic regression (with penality l1 or l2), I tried: random forest, decision tree, svm.SVC, svm.LinearSVC, LDA (linear discriminant analysis), QDA (Quadratic Discriminant Analysis), AdaBoost, Gradient Boost.

Perform a detailed error analysis of your models.

See table below for a summary. **Red** highlights are best performances (it varies depending on different experiments, I just highlight best performances in common (for various experiments.)).

Train: Dev

precision recall F-score precision recall F-score classifier

Baselines--

0.432750000 1.000000000 0.604083057|0.418000000 1.000000000 0.589562764 All-complex

0.600740132 0.844020797 0.701897670|0.605351171 0.866028708 0.712598425 Word-length

0.614019117 0.779318313 0.686863544|0.599644128 0.806220096 0.687755102 Word-frequency

----

0.867128827 0.597294071 0.707351555|0.894736842 0.591772152 0.712380952 Naive Bayes

0.643558637 0.715938303 0.677821722|0.684210526 0.724050633 0.703567036 Logistic regression

----

0.773541306 0.702518363 0.736321144|0.794258373 0.723311547 0.757126568 Naive Bayes (improved)

0.738301560 0.740440324 0.739369395|0.767942584 0.741339492 0.754406580 Logistic regression (improved)

0.686886193 0.709850746 0.698179683|0.712918660 0.712918660 0.712918660 Random forest

0.790872328 0.724722075 0.756353591|0.811004785 0.701863354 0.752497225 Decision tree

0.798382438 0.734715577 **0.765227021**|0.806220096 0.729437229 **0.765909091** SVM SVC

0.741767764 0.734553776 0.738143145|0.770334928 0.743648961 0.756756757 SVM linear SVC

0.916233391 0.637459807 0.751836928|0.904306220 0.618657938 0.734693878 QDA

0.734835355 0.734835355 0.734835355|0.767942584 0.741339492 0.754406580 LDA

0.754477181 0.766881973 0.760629004|0.799043062 0.734065934 0.765177549 AdaBoost

0.812247256 0.773377338 **0.792335869**|0.827751196 0.726890756 **0.774049217** Gradient Boost

----

Analyze your model

An important part of text classification tasks is to determine what your model is getting correct, and what your model is getting wrong. For this problem, you must train your best model on the training data, and report the precision, recall, and f-score on the development data.

As a result, I think our best model is **SVM SVC** as well as **Gradient Boost** (Although GradBoost generally outperforms SVM SVC in both Train and Dev with respect to F score, it is surprising that when it comes to Leaderboard, SVM SVC results in better ranking (typically with 2\% better Fscore).)

Give several examples of words on which your best model performs well. Also give examples of words which your best model performs poorly on, and identify at least TWO categories of words on which your model is making errors.

Gradient Boost: correct prediction:

Examples of true positive ['derailed', 'magma', 'emergency', 'aced', 'assistance', 'fatalities', 'fair-weather', 'complicated', 'krill', 'affirmed']

Examples of false negative ['string', 'shaping', 'worked', 'away', 'spray', 'wear', 'closely', 'code-named', 'blood', 'pass']

Incorrect prediction:

Examples of false positive (i.e. not complex, but are predicted to be) ['asylum-seekers', 'considers', 'airliners', 'jumping', 'makings', 'fishermen', 'worldwide', 'breathing', 'motorcycle', 'destroying']

Examples of true negative (i.e. complex, but are predicted not to be) ['required', 'canoes', 'potential', 'concerts', 'patch', 'nylon', 'assist', 'wartime', 'brisk', 'ironic']

SVM SVC: correct prediction:

Examples of true positive ['derailed', 'emergency', 'assistance', 'fatalities', 'complicated', 'affirmed', 'undermining', 'certificates']

Examples of false negative ['string', 'worked', 'away', 'spray', 'wear', 'closely', 'code-named', 'blood']

Incorrect prediction:

Examples of false positive (i.e. not complex, but are predicted to be) ['asylum-seekers', 'shaping', 'considers', 'airliners', 'jumping', 'choices', 'makings', 'jumped']

Examples of true negative (i.e. complex, but are predicted not to be) ['magma', 'aced', 'required', 'canoes', 'fair-weather', 'krill', 'potential', 'chug']

Time elapsed: 31.625770092010498