Tagging Raw Job Descriptions

April 7, 2017

1 Training Dataset Quality

I appreciated the dataset provide by Indeed and HackerRank. It is great opportunity for me to learn data science.

As I commented in the forum: I guess that the tags in training sets are generated by some machine learning algorithms in Indeed. It looks that the best precision and recall we have achieved here are about 0.7, which leads to a F1 score of 0.7. I doubt that it is hard to get higher F1 score because the mis-classifications in training sets. It is also possible that there are similar mis-classifications in test sets.

2 Data Pre-processing

The task is considered as binary classification for each tag. Snow Ball Stemmer from NLTK library was used for text stemming. CountVectorizer from Scikit-Learning is used to convert a collection of text documents to a matrix of token counts. TfidfTransformer is used to transform a count matrix to a normalized tf or tf-idf representation.

Use features from 1-gram, 2-grams and 3-grams

3 Model

3.1 Rule-based

Unsupervised learning with keyword filter with: [["part time", "part-time"], ["full time", "full-time"], ["hourly", "hour"], ["salary"], ["associate "], ["bachelor", " bs ", "/bs " " bs/"], ["master", "phd"], ["licence", "license", "certif", "eligible"], ["1 year", "one year"], ["2 years" , "3 years" ,"4 years", "two years" , "three years" ,"four years", "2+ years", "3+ years", "4+ years"], ["5 years", "5+ years", "five years", "0 years"], ["supervi"]]

Best result (F-1 score): 0.559

3.2 Linear SVM Classifier

LinearSVC from Scikit-learn library.

Best result: 0.690943485457555

3.3 Xgboost

Xgboost library.

Best result: 0.7041314971123944

For the sake of over-fitting, I used the best model from cross-validation for submission.