**Entity set expansion on KDD2001, binding to thrombin (learning to identify activator compounds):**

N.B: All of these performance estimates are somewhat virtual - and potentially dealing with an unnecessarily constrained problem - the number of positives we're hiding in N is negligible - and the fact that the rest of the new unlabeled set is surely not in P does not help either.

There are 42 positives and 1867 negatives. We can hide a part of these positives in the negative set - therefore forming the unlabeled set U. Then, we can try to see how these fair with our algorithms:

1. Bayesian sets, tracking mean average precision:

a) Revealing 10 of the positives:

map@10 = 0.5

map@42 = 0.2619 - meaning it only identified one more than it was given.

b) Revealing 20 of the positives yields:

map@10 = 0.6

map@20 = 0.35

map@42 = 0.2381 - fewer than if only 10 were used!

c) Using all 42 of the positives:

map@10 = 0.8

map@20 = 0.6

map@42 = 0.3571

Given that for data augmentation we need to be able to extract hidden positives with high certainty, and Bayesian sets perform badly at extracting even those that they're given as training material, I suspect they will not be very applicable with this dataset. I believe this is mostly to do with the relatively low number of positives provided, as well as because of how those positives distribution relates to the actual distribution of the data... If this is the case, none of my methods will work well.

2. Spy-EM algorithm, observing the precision, recall and f-score of the expanded set:

a) Revealing 10 of the positives:

If we reveal the first 10, the algorithm extracts 92 elements, with the following quality parameters:

precision, recall, f-score = 0.3261, 0.7143, 0.4478.

This means that the newly extracted set found 30 positives, i.e. 20 new ones, along with 62 negatives wrongly classified as positive.

b) Revealing 20 of the positives: We obtain an entity set of 96 - 33 positives inside.

precision, recall, f-score = 0.3438 0.7857 0.4783

c) Revealing all 42 positives:

The algorithm extracts 120 positives, i.e. adds 78 invalid ones to the set:

precision, recall, f-score = 0.3500 1.0000 0.5185

3. RocSVM: Not yet fast enough to handle this.

Furthermore, SVM has problems processing all of this data - I just don't have enough RAM memory to use for SVM training.

**Entity set expansion in the Reuters 21578 data set**

Treating it as binary classification, where label 1 is the first class, 2-58 are the second class. The train/test data have the same proportion (~40% ) of positives. Redux train sets consist of random samples of test - 10%, 33%, 80%. More importantly, We measure the change in SVM classification that data augmentation of the test set yields - and how this performance relates to the "perfect" case, i.e. using the whole training set. Additionally, we measure the success of entity set expansion as a PU problem alike to those cited in relevant work.

Training the SVM on the full train set, and then applying it to the test set yields a classification with the following parameters: [pr, rc, fsc] = [0.9263, 0.9673, 0.9464].

With SVM, we're using kktviolationlevel = 0, the default. SVM achieves perfect precision/recall on training set - might be overfitting? No, having tested this, we can observe that using a higher threshold results in an SVM not adapted to train, with far worse subsequent results on train.

Reduced set 1 (10%):

Training the SVM with reduxSet1, yields a classification of the following quality (applied on testSet).

[p,r,f] = 0.8212 0.9538 0.8826

Applying Bayesian sets, we obtain the augmented\_redux1 = redux1 + newPositives. This set is obtained from the set of P+U, disregarding N**(could the knowledge of N be applied to improve classification?)** The datasets to apply our SSL methods to, for each of the redux sets, have been saved as well.Training the SVM with this set yields the following parameters:

sEM algorithm:

Bayesian sets:

The quality of the new set of positives is:

The quality of subsequent classification using SVM is: 0.7790 0.7490 0.7637

Reuters is not binarized!!

Ok, redo whole method - given train & test, generate all three reduxes and appropriate unlabeled sets randomly, train SVMs on each, test, then augment each redux, and retrain SVMs on this. Output all three results. It must take the proportion of positives and the proportion of negatives as input.