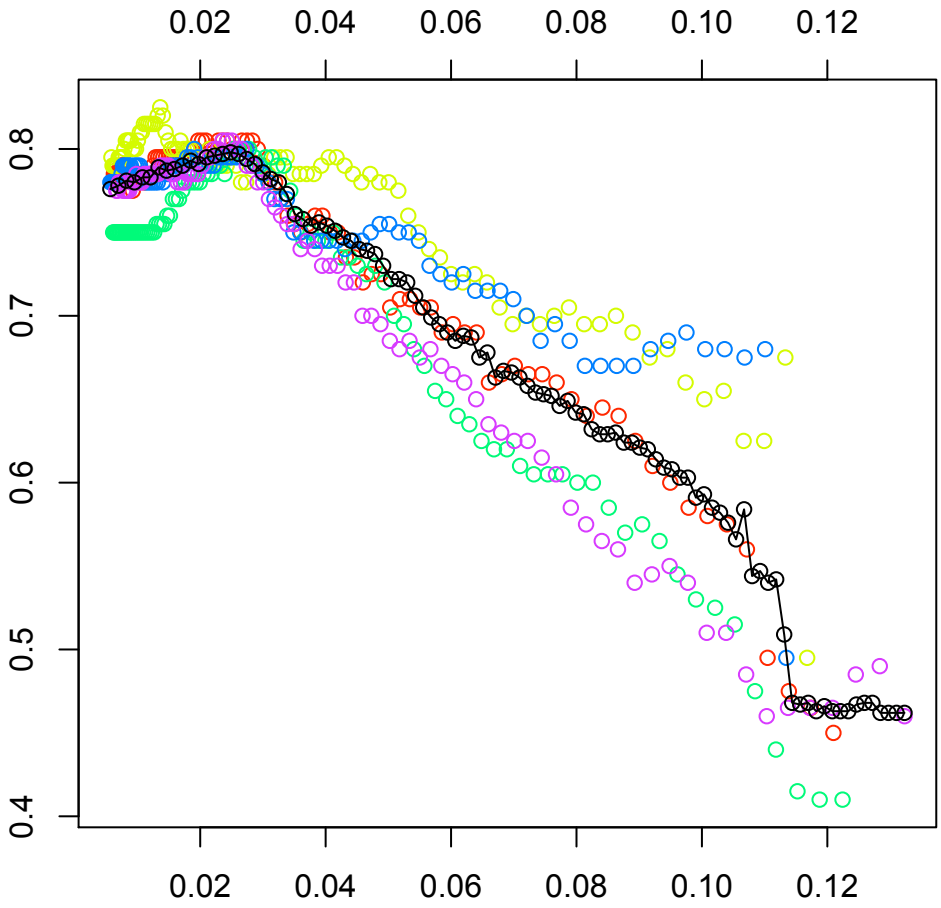
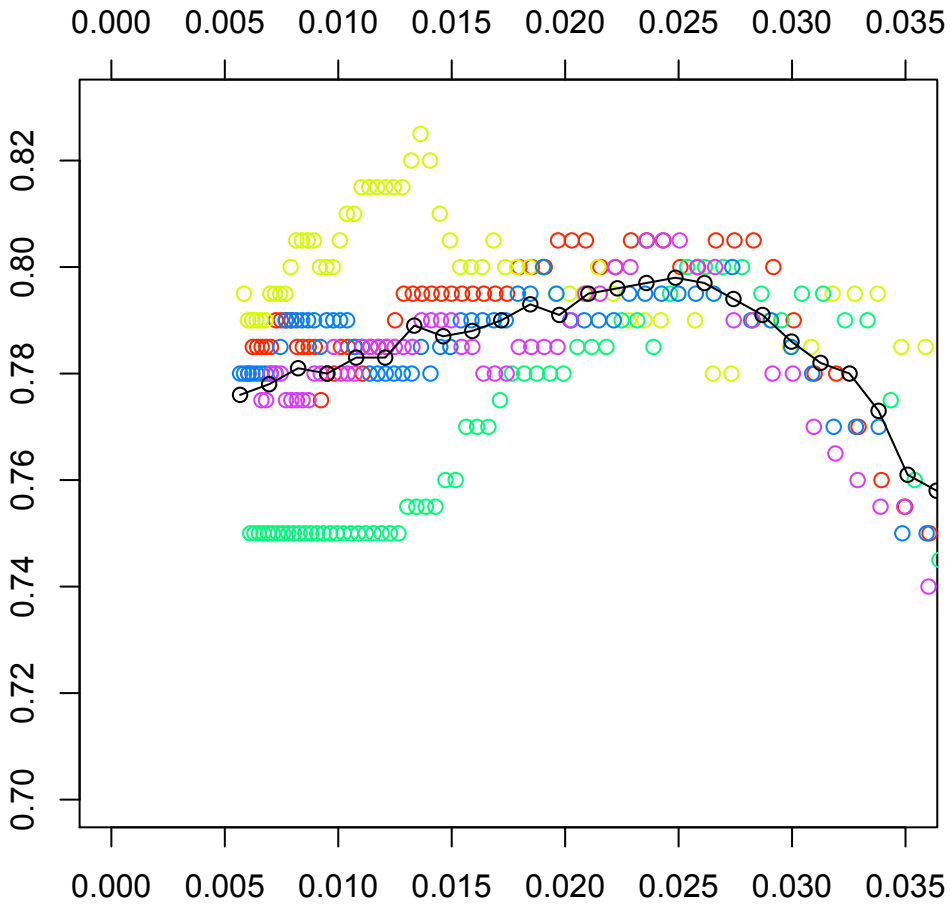
Jan 22 2010 experiments (Brendan)
On the pang&lee 2002, 2004 dataset (version 2.0), using their unigram-ization
On my half-size, 1000 (500 vs 500) dev set
5-fold, 5-way-predicted reconstructed cross-validation

Graphs are lambda against accuracy

Using **raw counts**

```
> eval_multi_run(r5,data)
Multi best settings
     lambda   acc  df
1 0.02830493 0.805 162
2 0.01362418 0.825 384
3 0.02779230 0.800 188
4 0.02736469 0.800 171
5 0.02504831 0.805 210
Best lambda
[1] 0.0248596
Best acc
[1] 0.798
```

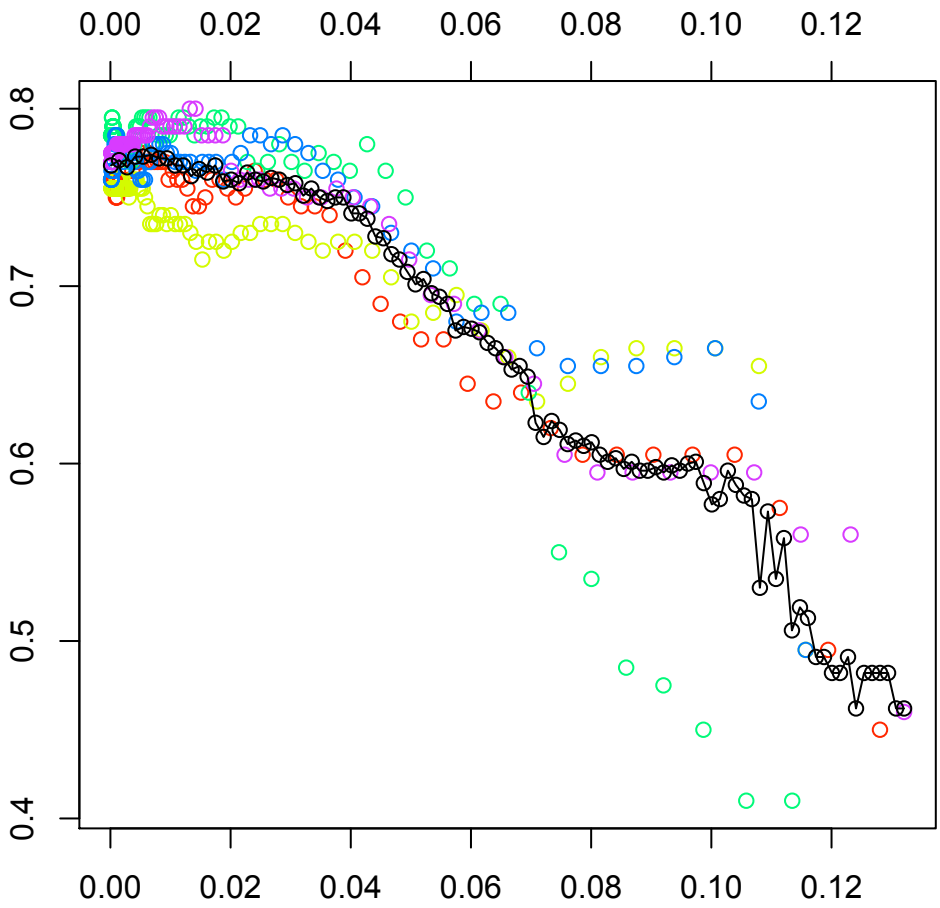

0.02
0.04
0.06
0.08
0.10
0.12
0.4
0.5
0.6
0.7
0.8



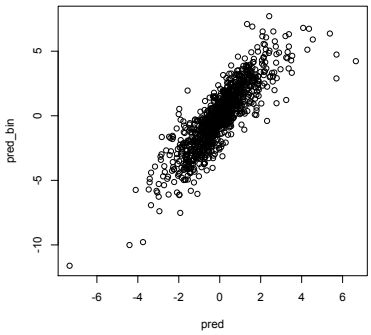
Using **word presence**. Less stable, wants more df. Worse accuracy, too.

```
> eval_multi_run(r5_bin,data)
Multi best settings
      lambda   acc  df
1 0.006373725 0.775 483
2 0.004356722 0.760 512
3 0.018490893 0.795 321
4 0.028657736 0.785 160
5 0.014161624 0.800 399
Best lambda
[1] 0.00677802
Best acc
```

[1] 0.774



And the predictions (links) are somewhat different but not significant by paired t-test (p=.9) … but the correctness vectors *do* have statistically significant change for the word counts-not-bools.



Finally, **log(1+count)** is nearly the same as count, at least acc-wise.

```
> eval_multi_run(r5_log,data)
Multi best settings
      lambda   acc  df
1 0.015277393 0.810 359
2 0.009648892 0.790 449
3 0.032351632 0.815 132
4 0.013321476 0.800 384
5 0.013015716 0.795 403
Best lambda
[1] 0.01282017
Best acc
[1] 0.795
```

Summary

Accuracy (5-fold)

Word counts: 0.798
Word log-counts: 0.795
Word presence: 0.774

Df and lambda (single model, trained on everything, with 5-fold-selected lambda)

```
> m
     Df   %Dev Lambda
[1,] 216 0.5175  0.024

> m_bin
     Df  %Dev Lambda
[1,] 558 0.867  0.007

> m_log
     Df   %Dev Lambda
[1,] 417 0.7475  0.013
```

It seems that the two dampening transformations make things more inefficient. Or something.

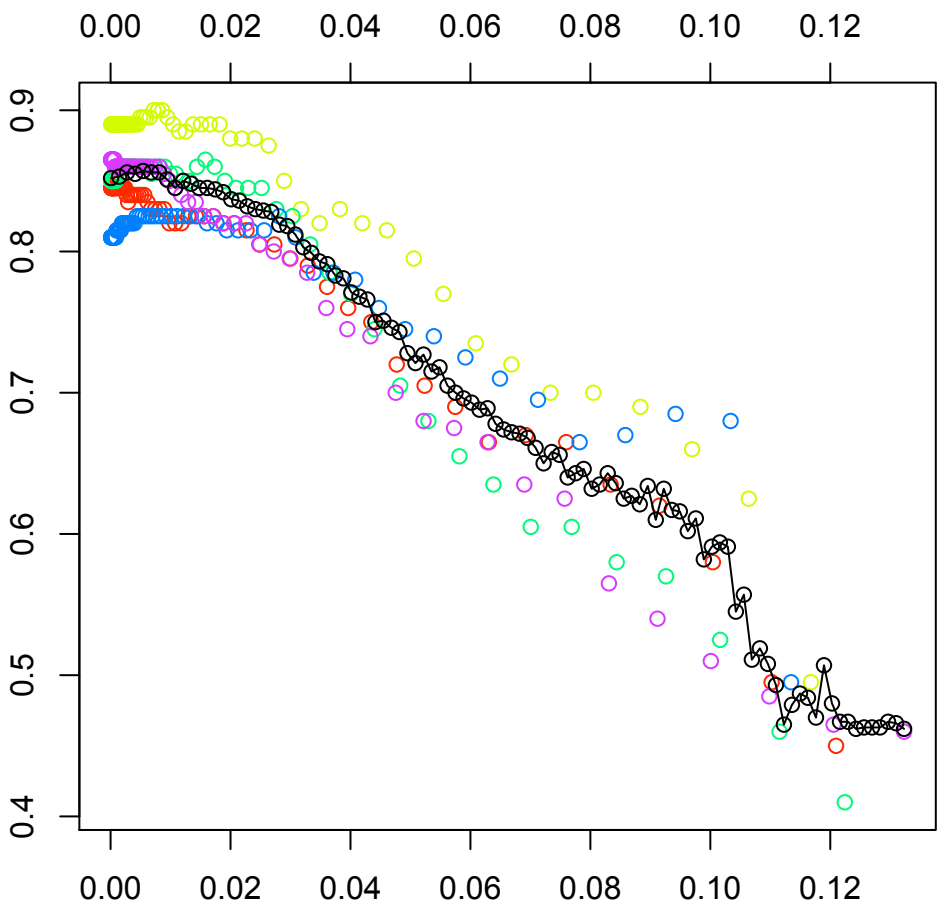
The selected active set changes a lot per fold.
Since training and test accuracies are different: typically 95% vs 80%, say. This implies that different active sets must get chosen.
Between the 5 active sets from each fold, the average pairwise jaccard is only 28%.

Is there a way to derive a proper, meaningful active set?

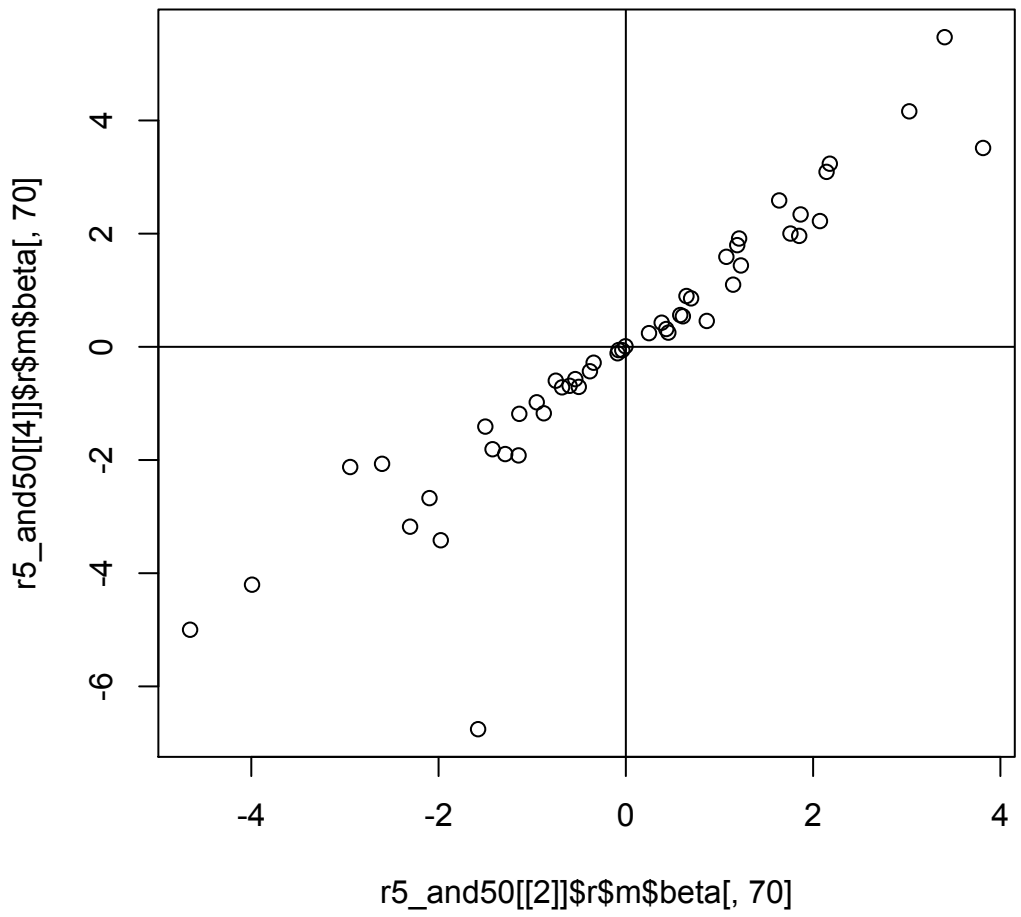
Let's try: take the *intersection* of all 5 model active sets. 50 features total.
Omg it was outperforms. Is this overfitting somehow?
Good: all 50 features are selected in all cases. (I wonder if could do zero regularization.)

```
> eval_multi_run(r5_and50,data2)
Multi best settings
       lambda   acc df
1 0.0016752460 0.850 50
2 0.0086309610 0.900 50
3 0.0158117415 0.865 50
```

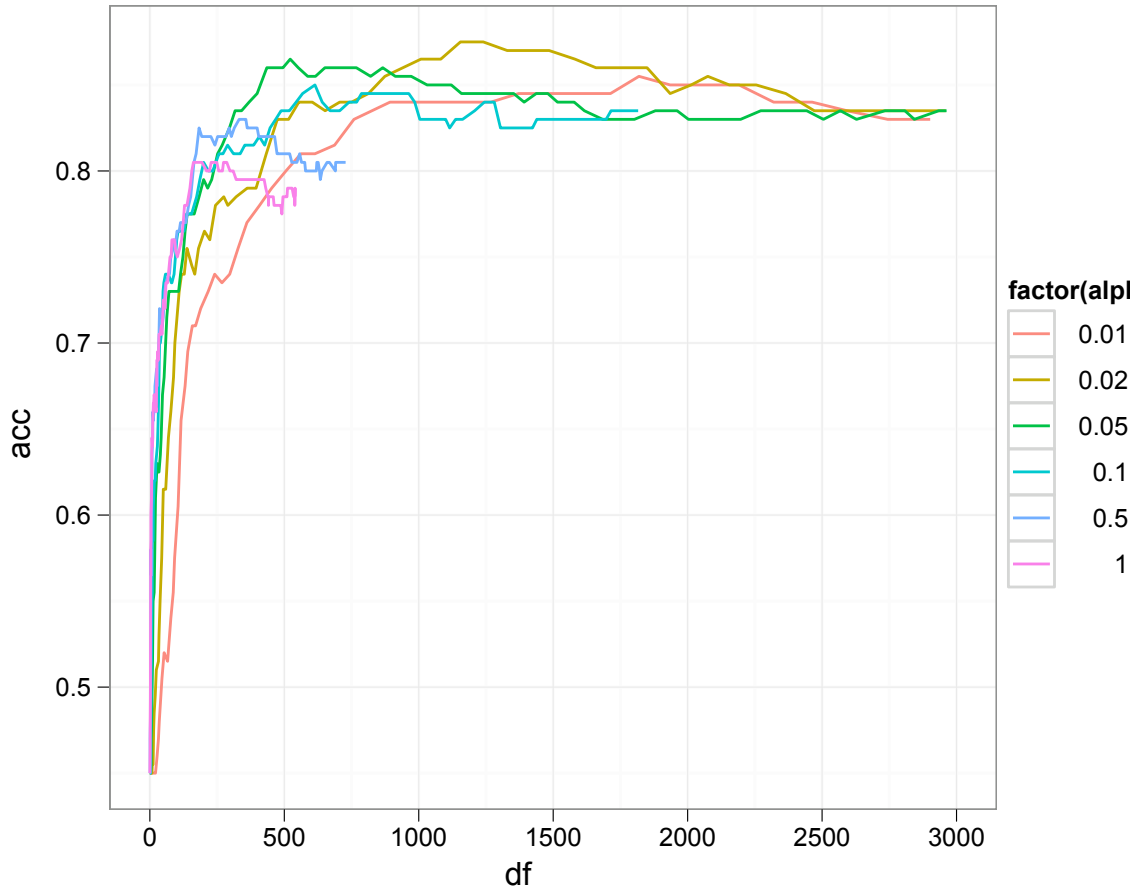
4 0.0281054575 0.825 50
5 0.0006584799 0.865 50
Best lambda
[1] 0.005427224
Best acc
[1] 0.857



Coefficients between different folds are mostly the same. I bet outliers are for words that don't appear in that fold's test set, so going funny with them doesn't matter.



Alpha sensitivity
Turns out want it pretty low. All the above results are lasso, but alpha=0.02 (that is, nearly pure L2) is way better.
Plot is for just the 1st fold.
So, more ridge-y causes it to peak later.


acc
0.8
0.7
0.6
0.5
0
500
1000
1500
2000
2500
3000
df
factor(alpl
0.01
0.02
0.05
0.1
0.5
1