NOTE: the code in this notebook is hidden for better readability. To toggle on/off, click here.

Unsupervised Anomaly Detection in High Dimensions: SOD vs One-Class SVM

Spencer Stirling (https://github.com/sstirlin)

December 22, 2015

Introduction

In this article we test two algorithms that detect anomalies in high-dimensional data. For our purposes, "high-dimensional" means tens to hundreds of dimensions. Our results might generalize to "very high-dimensional" (100,000 dimensions is commonly seen in NLP and image processing), but we have not carefully experimented there.

Since we rarely have training data (nor even know what we are looking for), we are only interested in *unsupervised* algorithms.

The first algorithm, **Subspace Outlier Degree (SOD)** kriegel2009, is an unsupervised *local* anomaly detector. "Local" means that points are compared against their nearest neighbors (*not* against the entire dataset). The appeal of SOD is that it overcomes the curse of dimensionality that plagues distance and density-based algorithms such as Local Outlier Factor (http://www.dbs.ifi.lmu.de/Publikationen/Papers/LOF.pdf). For our experiments we use the implementation found in the `ELKI` (http://elki.dbs.ifi.lmu.de/) framework.

The second algorithm, **One-Class Support Vector Machine** scholkopf2001, is a semi-supervised global anomaly detector (i.e. we need a training set that contains only the "normal" class). However, since SVM decision boundaries are soft, it can be used unsupervised as well.

We experiment with this, as well as two variants ("eta" and "robust") that were recently proposed amer2013 explicitly for unsupervised applications. We use the implementation in `RapidMiner` (https://github.com/Markus-Go/rapidminer-anomalydetection) (which leverages a modified version of `libsvm` (https://www.csie.ntu.edu.tw/~cjlin/libsvm/) underneath).

All scripts and datasets can be obtained from the github repo (https://github.com/ActivisionGameScience/unsupervised_anomaly_detection_sod_vs_one_class_svm.git).

TL;DR

Surprisingly, we found that the ordinary one-class SVM produced the best results out of all of the algorithms tested, *even unsupervised*. It was also the fastest. The performance of the other algorithms was as follows:

- robust SVM was unreliable
- eta SVM quality was competitive, but it was slightly slower and is not widely implemented
- SOD quality was competitive *as long as the parameter* snn *was tuned optimally*. However, it was *much* slower

For SOD we observed a curious phenomenon: the optimal parameter tuning seems to be

$$\mathrm{snn} \gtrapprox \text{no. outliers in the dataset}$$

In real datasets we usually do not know how many outliers to expect, hence it is difficult to tune SOD in practice.

For SVM we only studied the RBF kernel. Two settings for the parameter γ were investigated:

1. "automated gamma tuning" as was proposed in evangelista2007
2. the simple heuristic $\gamma = \frac{1}{\text{no. of datapoints}}$

Again, our results were surprising. We found that, although automated tuning helped for some datasets, it was disastrous in other cases. The simple heuristic is faster and more reliable.

sklearn Users Beware

Users of `sklearn` (http://scikit-learn.org/stable/modules/generated/sklearn.svm.OneClassSVM.html#sklearn.svm.OneClassSVM) should be aware that leaving the default argument $\gamma = 0$ *enables* automated gamma tuning (despite what the documentation says). Since we found that automated tuning was unreliable, we recommend explicitly setting the value $\gamma = \frac{1}{\text{no. of datapoints}}$.

Also, although `RapidMiner` computes an "outlier score" that can be readily used, `sklearn` only returns the decision function. To compute the outlier score, first compute the maximum value of the decision function:

$$Q = \max_{x'}\ \mathrm{decision\_function}(x')$$

Then

$$\mathrm{outlier\_score}(x) = Q - \mathrm{decision\_function}(x)$$

Data preparation

We prepare labelled datasets similar to those found in amer2013, and we add several more. Our procedure is to pick an "anomaly class" in each dataset and cull it to a small size compared to the other class(es).

name
ionosphere (https://github.com/ActivisionGameScience/unsupervised_anomaly_detection_sod_vs_one_class_svm/ionosphere/ionc
shuttle (https://github.com/ActivisionGameScience/unsupervised_anomaly_detection_sod_vs_one_class_svm/shuttle/shuttle_for_e
breast_cancer (Wisc. diag.) (https://github.com/ActivisionGameScience/unsupervised_anomaly_detection_sod_vs_one_class_svm/breast_cancer_wisconsin_c
satellite (https://github.com/ActivisionGameScience/unsupervised_anomaly_detection_sod_vs_one_class_svm/satellite/sat_for_elk
mouse (https://github.com/ActivisionGameScience/unsupervised_anomaly_detection_sod_vs_one_class_svm/mouse/mouse_for_e
kddcup99_5000 (https://github.com/ActivisionGameScience/unsupervised_anomaly_detection_sod_vs_one_class_svm/kddcup99/
kddcup99_10000 (https://github.com/ActivisionGameScience/unsupervised_anomaly_detection_sod_vs_one_class_svm/kddcup99/kddcup_for_elki_
kddcup99_20000 (https://github.com/ActivisionGameScience/unsupervised_anomaly_detection_sod_vs_one_class_svm/kddcup99/kddcup_for_elki_
kddcup99_100000 (https://github.com/ActivisionGameScience/unsupervised_anomaly_detection_sod_vs_one_class_svm/kddcup99/kddcup_for_elki_

Each categorical variable was converted to dummy variables (using `get_dummies` (http://pandas.pydata.org/pandas-docs/stable/generated/pandas.get_dummies.html) in python). This is included in the dimensionalities listed above.

All variables (including dummy variables) were then standardized to have mean=0, stddev=1.

Use Precision-Recall instead of ROC

Although the ROC curve is the most popular measure of classifier quality, it is much better to use the Precision-Recall curve when the dataset is highly imbalanced davis2006. The explanation is simple. Recall that ROC measures the true positive rate (sensitivity)

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

versus the false positive rate

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}$$

When the negative class is much larger than the positive class (as happens in anomaly detection) then, even if the false positive *rate* is small, the actual number of false positives might be large compared to the number of true positives. The ROC curve would appear favorable, but the algorithm would be poor.

From now on we restrict our attention to the PR curve (and the area under the curve PR-AUC).

SOD quality depends on SNN

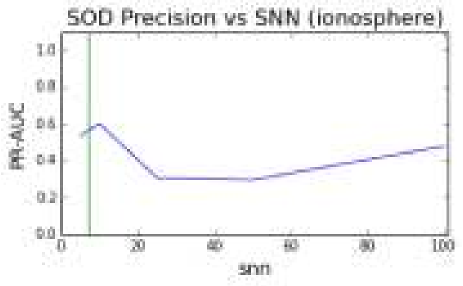
SOD depends on three parameters:

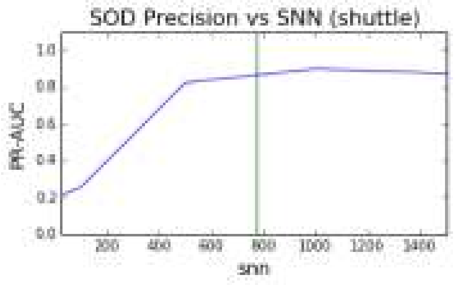
- number of **shared nearest neighbors** snn
- number of **ordinary nearest neighbors** knn (not important, but must be larger than snn. We use $2 * \mathrm{snn}$)
- a threshold parameter α (we use the default value 0.8)

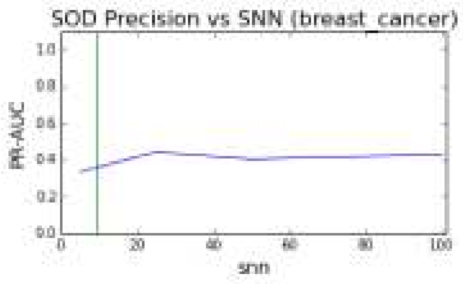
It is claimed in kriegel2009 that SOD is *not* sensitive to snn, provided snn is sufficiently large (so there are enough nearest neighbors to compare against).

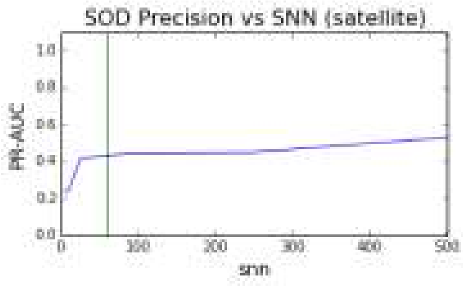
In our experiments, however, we *do* see sensitivity to snn. In the following graphs we plot the average quality (PR-AUC) versus snn for each dataset. To provide context, the vertical line depicts the number of anomalies in each dataset.

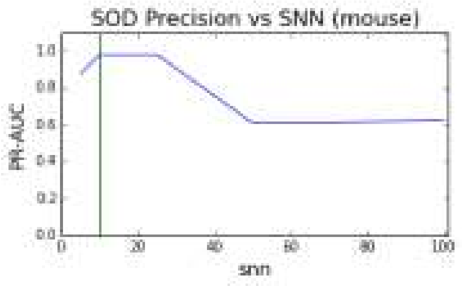
Because of performance limitations we could not make a full parameter sweep on kddcup99_20000, and were unable to run SOD on kddcup99_100000 at all.

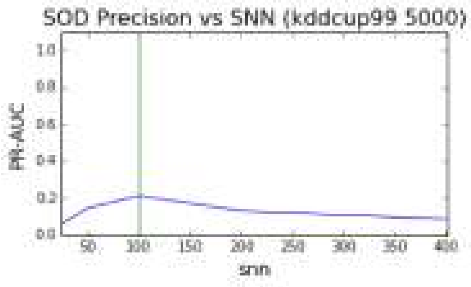

SOD Precision vs SNN (ionosphere)
PR-AUC
snn

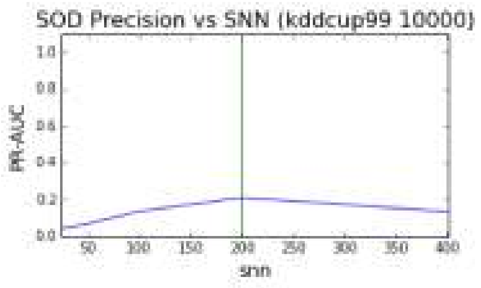

SOD Precision vs SNN (shuttle)
PR-AUC
snn

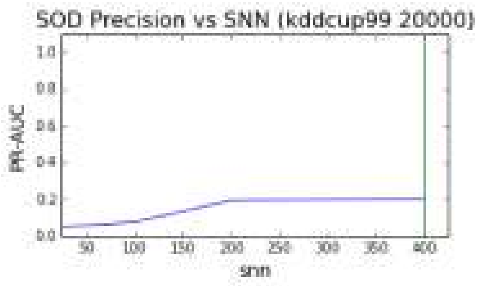

SOD Precision vs SNN (breast_cancer)
PR-AUC
snn


SOD Precision vs SNN (satellite)
PR-AUC
snn


SOD Precision vs SNN (mouse)
PR-AUC
snn


SOD Precision vs SNN (kddcup99 5000)
PR-AUC
snn





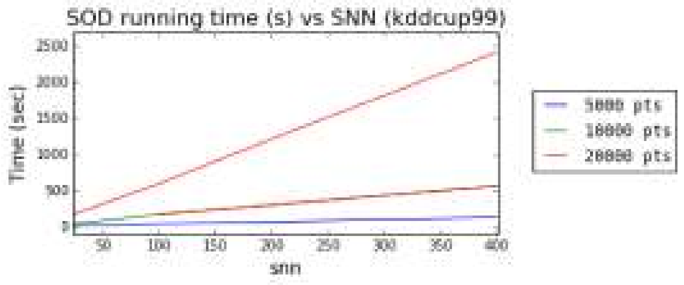
Discussion

From the above graphs it is clear that the quality of SOD depends on the choice of snn. Interestingly, the optimal setting for snn appears to be approximately the same as the number of anomalies in the dataset (or slightly larger).

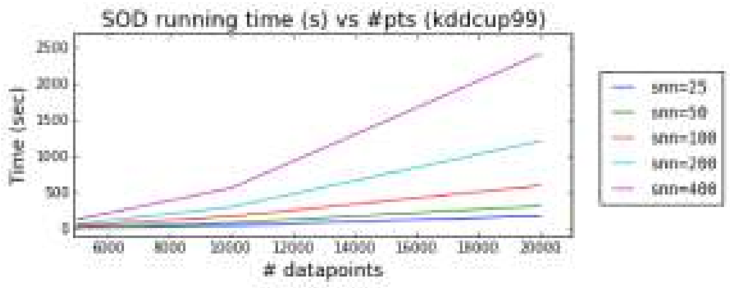
Since in practice we won't know the number of anomalies to expect, we won't be able to tune snn effectively. It is important to notice that the quality drops especially quickly if snn is chosen too small.

SOD running time

Since quality depends on snn, we need to know how running time depends on snn. The plot below shows this relationship for three separate datasets: kddcup99 5000, 10000, and 20000. Running time appears to grow linearly $\mathrm{O}(\mathrm{snn})$ in each case.



Likewise, we need to know how running time is affected by increasing the size of the dataset. Below we plot several curves with fixed snn and varying data size. It turns out that the running time grows quadratically $\mathrm{O}(n^2)$ in each case.



Discussion

Putting these together we see that SOD running time grows like $\mathrm{O}(n^2 \cdot \mathrm{snn})$.

However, we already saw that we should scale $\mathrm{snn} \gtrapprox (\text{no. of anomalies})$ to optimize quality. Since $(\text{no. of anomalies}) \propto (\text{size of dataset } n)$, we conclude that we should scale $\mathrm{snn} \propto n$.

So optimal SOD has time complexity $\mathrm{O}(n^3)$. Below we will see that SOD is far more expensive than SVM when we compare them head-to-head.

One-Class SVM quality

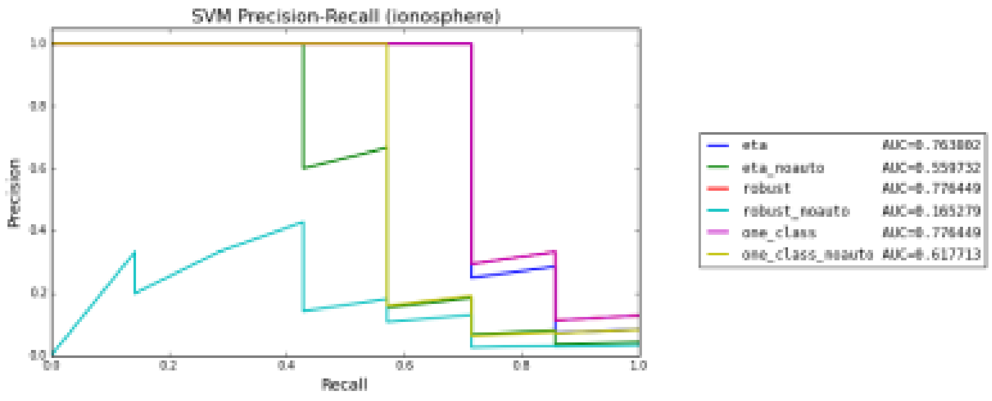
We already mentioned that we are comparing three variants of one-class SVM: ordinary one-class SVM, eta SVM, and robust SVM. Although we could study other kernels, in our experiments we limited ourselves to RBF. Two settings for the parameter γ were investigated:

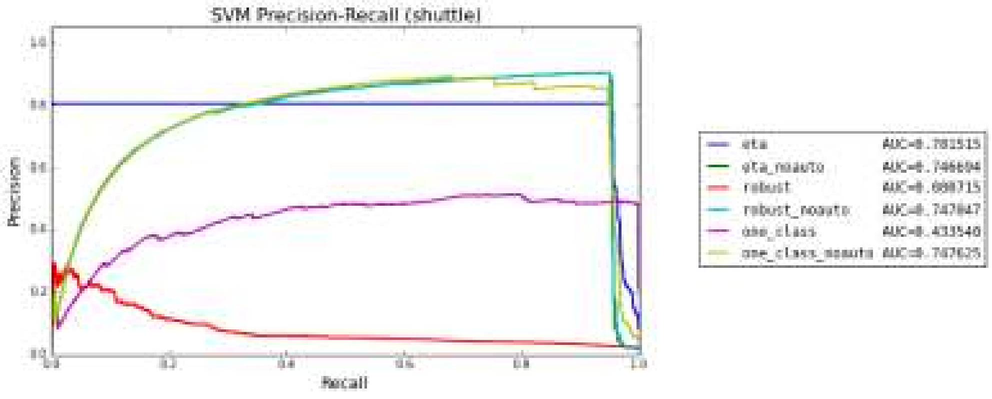
1. "automated gamma tuning" as was proposed in evangelista2007
2. the simple heuristic $\gamma = \frac{1}{\text{no. of datapoints}}$

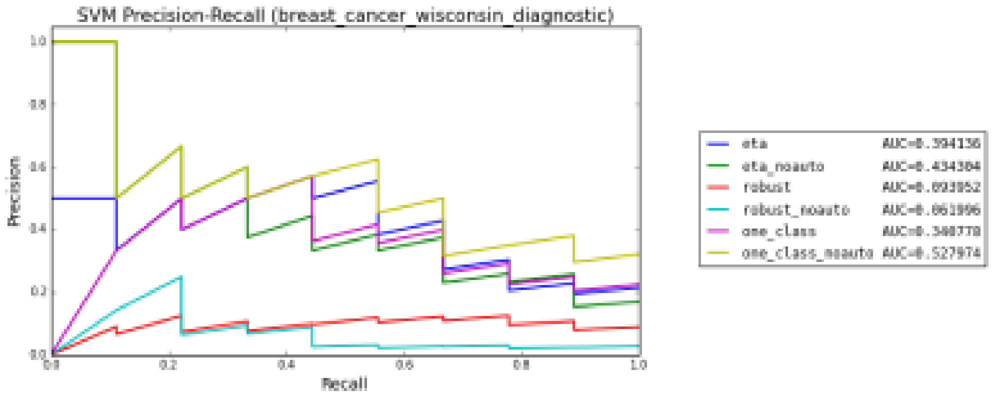
The regularization is bundled into a separate parameter ν. The default value $\nu = 0.5$ seemed to work well, but we did not systematically study this aspect.

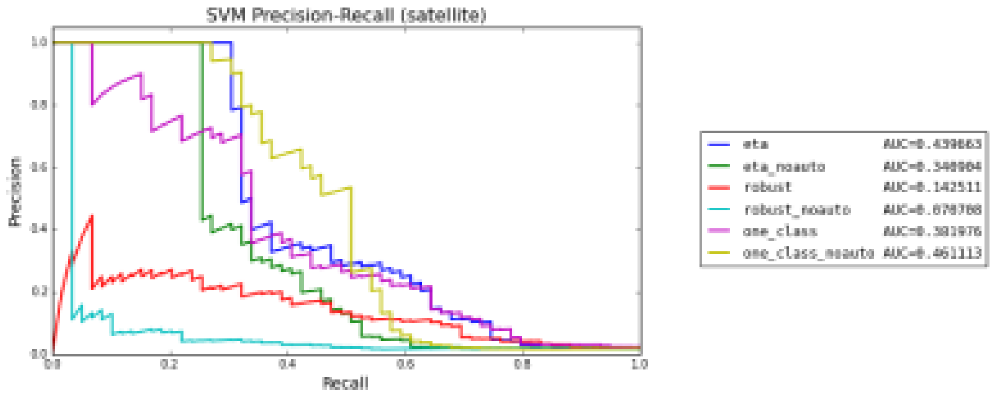
In the graphs below we compare all six Precision-Recall curves for each dataset:

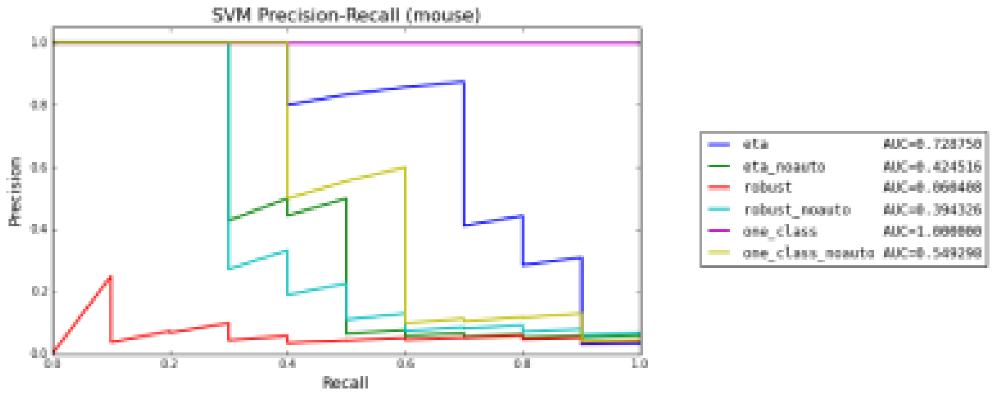
1. eta SVM *with* automated gamma tuning
2. eta SVM *without* automated gamma tuning
3. robust SVM *with* automated gamma tuning
4. robust SVM *without* automated gamma tuning
5. ordinary one-class SVM *with* automated gamma tuning
6. ordinary one-class SVM *without* automated gamma tuning

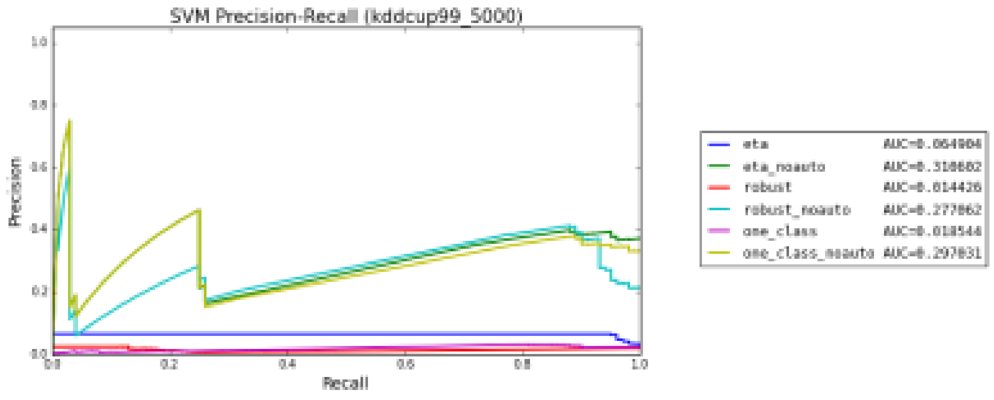

SVM Precision-Recall (ionosphere)
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0
eta AUC=0.763802
eta_noauto AUC=0.559732
robust AUC=0.776449
robust_noauto AUC=0.165279
one_class AUC=0.776449
one_class_noauto AUC=0.617713

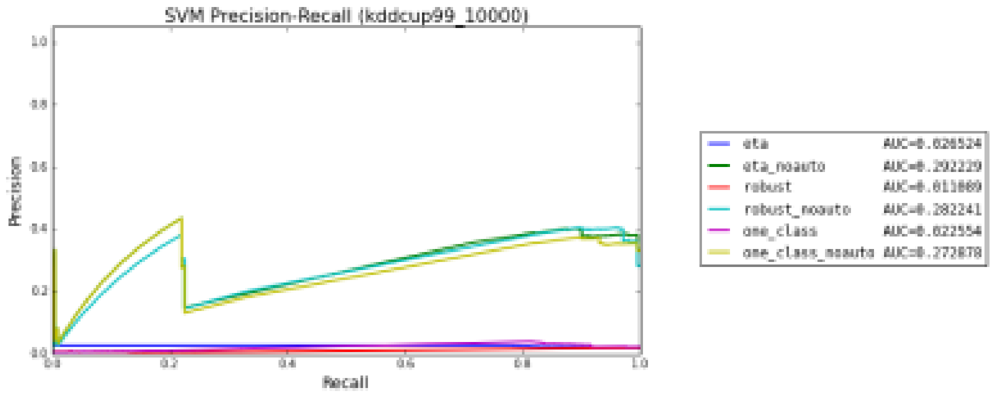

SVM Precision-Recall (shuttle)
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0
eta AUC=0.781515
eta_noauto AUC=0.746694
robust AUC=0.080715
robust_noauto AUC=0.747047
one_class AUC=0.433540
one_class_noauto AUC=0.747625


SVM Precision-Recall (breast_cancer_wisconsin_diagnostic)
Precision
Recall
0.0
0.2
0.4
0.6
0.8
1.0
eta AUC=0.394136
eta_noauto AUC=0.434304
robust AUC=0.093952
robust_noauto AUC=0.061996
one_class AUC=0.340778
one_class_noauto AUC=0.527974









Discussion

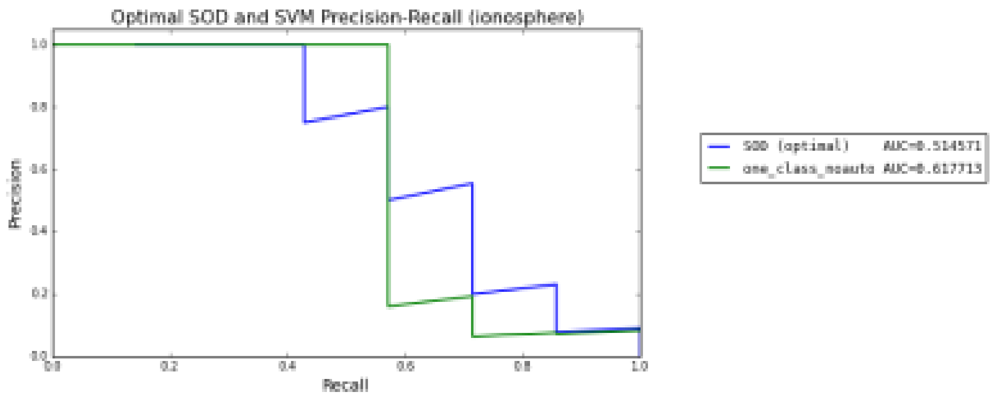
From the graphs it is clear that robust SVM (with or without automated gamma tuning) performs much worse on most datasets. Since it seems to be unreliable, we drop it from further discussion.

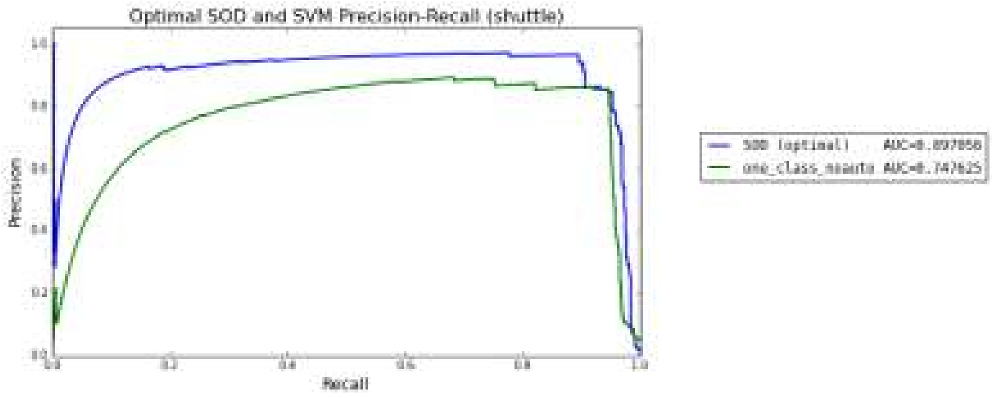
Now consider the effect of automated gamma tuning. Although it helps for the "ionosphere" and "mouse" datasets, results are mixed for the "shuttle" and "satellite" datasets. It actually *hurts* the "breast_cancer" results, and it *really hurts* the "kddcup99" results. Since it seems to be unreliable, we cannot recommend it.

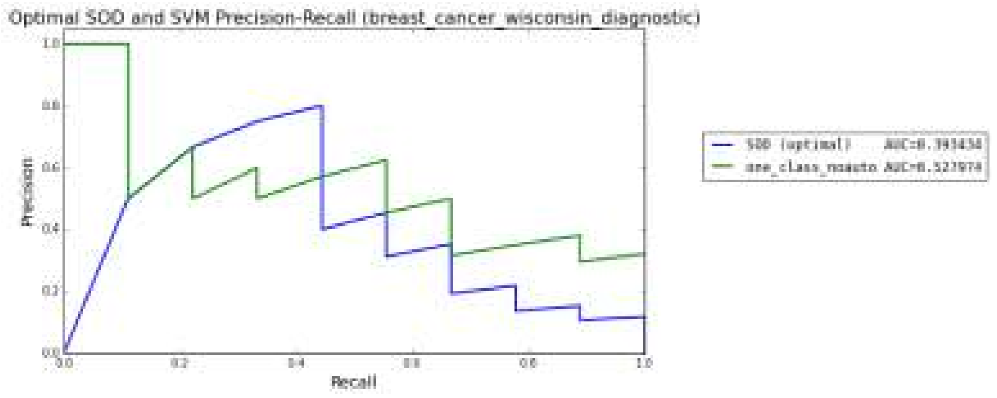
So we are left comparing eta SVM versus ordinary one-class SVM (both with automated gamma tuning *off*). We see that the results are mostly comparable. Since one-class SVM is standard in `libsvm`, we see no reason to use the eta variant.

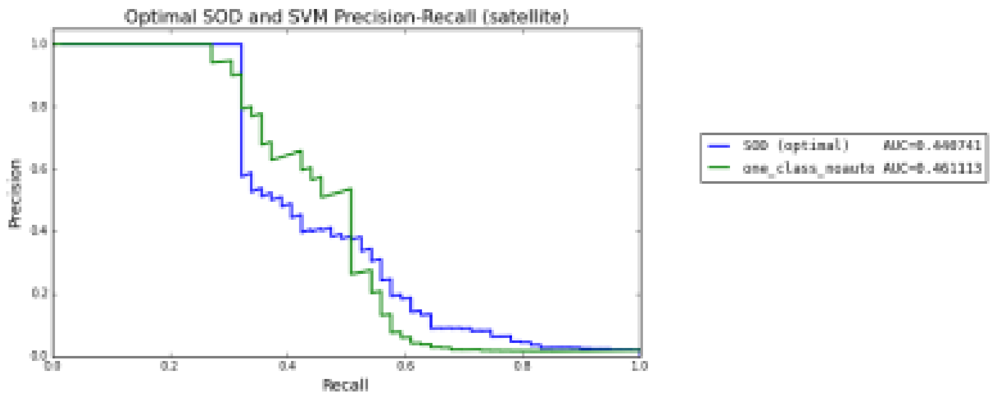
Head-to-head quality comparison: Optimal SOD vs One-Class SVM

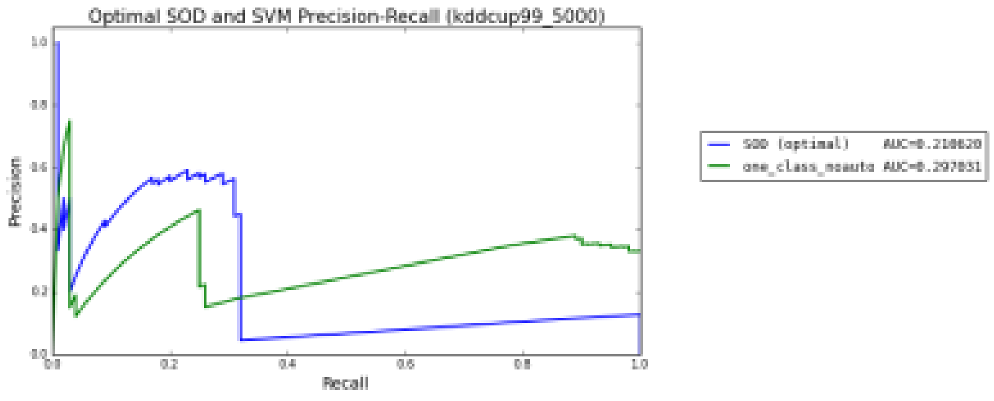
In the graphs below we compare ordinary one-class SVM (automated gamma tuning *off*) against SOD (where snn was tuned optimally for each dataset). We see that both algorithms are competitive, however one-class SVM seems to produce slightly better quality overall.

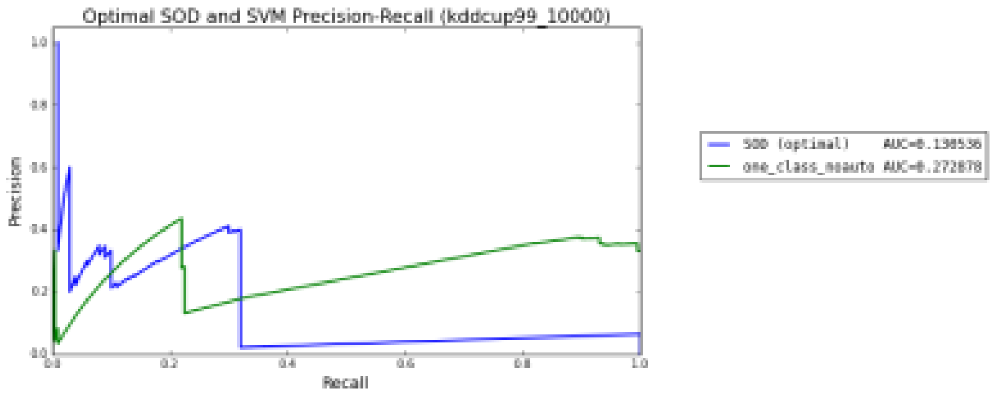

Optimal SOD and SVM Precision-Recall (ionosphere)
Precision
Recall
SOD (optimal) AUC=0.514571
one_class_noauto AUC=0.617713


Optimal SOD and SVM Precision-Recall (shuttle)
Precision
Recall
SOD (optimal) AUC=0.897856
one_class_noauto AUC=0.747625


Optimal SOD and SVM Precision-Recall (breast_cancer_wisconsin_diagnostic)
Precision
Recall
SOD (optimal) AUC=0.393434
one_class_noauto AUC=0.527974


Optimal SOD and SVM Precision-Recall (satellite)
Precision
Recall
SOD (optimal) AUC=0.440741
one_class_noauto AUC=0.461113



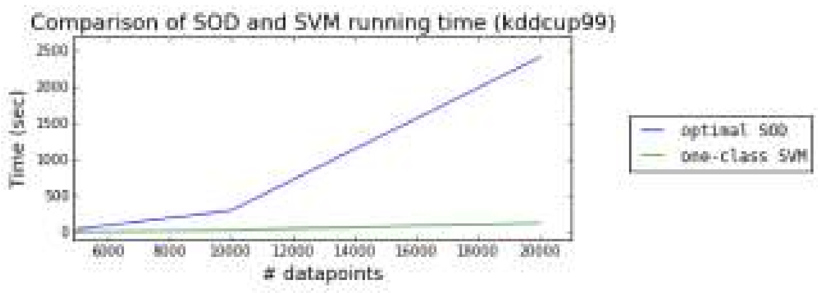


Head-to-head time comparison: Optimal SOD vs One-Class SVM

Finally, we compare the running time of one-class SVM (automated gamma tuning *off*) versus SOD (where snn was tuned optimally for each dataset).

We already argued above that Optimal SOD running time grows as $\mathrm{O}(n^3)$. Our experiments verify this. On the other hand, our experiments show that one-class SVM only grows as $\mathrm{O}(n^2)$.

Clearly, Optimal SOD is *much* more expensive than one-class SVM.



Conclusion

See TL;DR in the Introduction.

Bibliography

kriegel2009

Kriegel, Kröger, Schubert, Zimek (http://www.dbs.ifi.lmu.de/Publikationen/Papers/pakdd09_SOD.pdf), *Outlier Detection in Axis-Parallel Subspaces of High Dimensional Data*, Lec. Notes in Comp. Sci. Vol. 5476, 2009, pp 831-38

scholkopf2001

Schölkopf, Platt, Shawe-Taylor, Smola, Williamson (http://research.microsoft.com/pubs/69731/tr-99-87.pdf), *Estimating the Support of a High-Dimensional Distribution*, Neural Computation 13(7), 2001, pp 1443-71

amer2013

Amer, Goldstein, Abdennadher (http://outlier-analytics.org/odd13kdd/papers/amer,goldstein,abdennadher.pdf), *Enhancing One-class Support Vector Machines for Unsupervised Anomaly Detection*, ODD '13, 2013, pp 8-15

davis2006

Davis, Goadrich (http://www.autonlab.org/icml_documents/camera-ready/030_The_Relationship_Bet.pdf), *The Relationship Between Precision-Recall and ROC Curves*, Intl. Proc. on Mach. Learn., 2006

evangelista2007

Evangelista, Embrechts, Szymanski (http://www.cs.rpi.edu/~szymansk/papers/icann07.pdf), *Some properties of the gaussian kernel for one class learning*, Proc. of the 17th Int. Conf. on Artificial neural networks, ICANN'07, pages 269–278. Springer, 2007