Local Decomposition for Rare Class Analysis

J. Wu, H. Xiong, P. Wu, J. Chen

Nathan Deutscher

October 23, 2007

Outline

Why care about rare classes?

Why care about rare classes?

Often the rare classes are the ones we care about...

Why care about rare classes?

Often the rare classes are the ones we care about...

		Predicted		Class
		No	Yes	Error
Diabetes?	No	190	33	15%
	Yes	47	62	43%

Why care about rare classes?

Often the rare classes are the ones we care about...

		Predicted		Class
		No	Yes	Error
Diabetes?	No	190	33	15%
	Yes	47	62	43%

Can we do better?

Linear classifiers and rare classes

Linear classifiers:

Linear classifiers and rare classes

Linear classifiers:

- simple, efficient and easily understood;

Linear classifiers and rare classes

Linear classifiers:

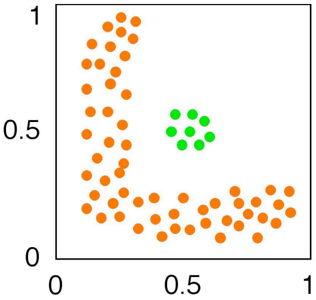
- simple, efficient and easily understood;
- less parameters $\Rightarrow$ more easily generalizable.

Linear classifiers and rare classes

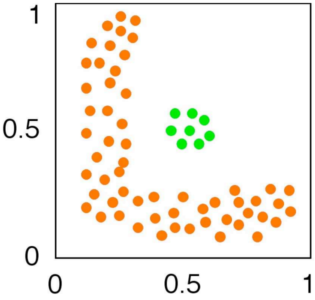
Linear classifiers:

- simple, efficient and easily understood;
- less parameters $\Rightarrow$ more easily generalizable.

Yet nonlinearly separable data exacerbates the rare class problem.



COG (Classification using lOcal clusterinG)



The Algorithm

Input:

# of clusters to find in common class

# of times to replicate rare class

Phase I: local clustering

Phase II: over-sampling

Phase III: training on altered class data

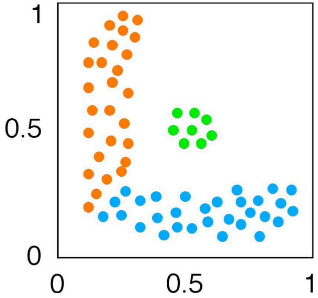
Phase IV: testing on original class data

Output:

Model on the training set.

Predictions for the test set.

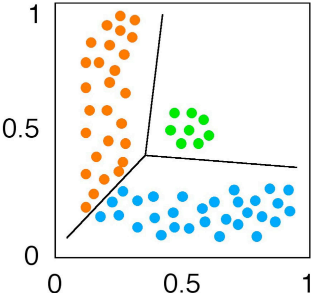
COG (Classification using lOcal clusterinG)



The Algorithm

Input:
# of clusters to find in common class
# of times to replicate rare class
Phase I: local clustering
Phase II: over-sampling
Phase III: training on altered class data
Phase IV: testing on original class data
Output:
Model on the training set.
Predictions for the test set.

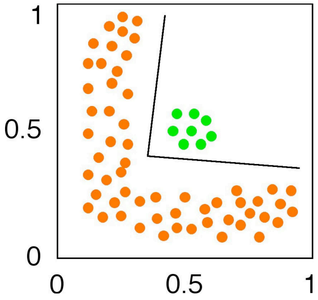
COG (Classification using lOcal clusterinG)



The Algorithm

Input:
# of clusters to find in common class
# of times to replicate rare class
Phase I: local clustering
Phase II: over-sampling
Phase III: training on altered class data
Phase IV: testing on original class data
Output:
Model on the training set.
Predictions for the test set.

COG (Classification using lOcal clusterinG)



The Algorithm

Input:

# of clusters to find in common class

# of times to replicate rare class

Phase I: local clustering

Phase II: over-sampling

Phase III: training on altered class data

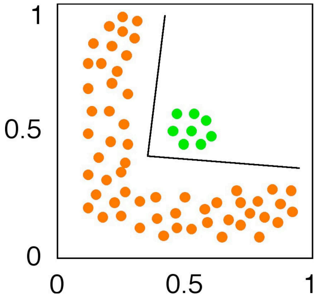
Phase IV: testing on original class data

Output:

Model on the training set.

Predictions for the test set.

COG (Classification using lOcal clusterinG)



The Algorithm

Input:

# of clusters to find in common class

# of times to replicate rare class

Phase I: local clustering

Phase II: over-sampling

Phase III: training on altered class data

Phase IV: testing on original class data

Output:

Model on the training set.

Predictions for the test set.

Initial Tests

Initial Tests

Split data into training (70%) and test (30%) sets.

Initial Tests

Split data into training (70%) and test (30%) sets.

- **Two-level**: Random sampling on small class to make it 'rare'.

Initial Tests

Split data into training (70%) and test (30%) sets.

- **Two-level**: Random sampling on small class to make it 'rare'.
- **Multi-level**: Random sampling not necessary...

Initial Tests

Split data into training (70%) and test (30%) sets.

- **Two-level**: Random sampling on small class to make it 'rare'.
- **Multi-level**: Random sampling not necessary...

How to measure success?

Initial Tests

Split data into training (70%) and test (30%) sets.

- **Two-level**: Random sampling on small class to make it 'rare'.
- **Multi-level**: Random sampling not necessary...

How to measure success?

The F measure

Recall:	Fraction of those *actually* in X that were placed there
Precision:	Fraction of those *placed* in X that were actually there

$$F = \frac{2 \cdot \mathrm{Recall} \cdot \mathrm{Precision}}{\mathrm{Recall} + \mathrm{Precision}}$$

Results

Pima (diabetes):

	SVM	COG(SVM)
rare	NA	0.373
common	0.949	0.940

This is typical for the two-level data... and for multi-level data?

Results

Pima (diabetes):

	SVM	COG(SVM)
rare	NA	0.373
common	0.949	0.940

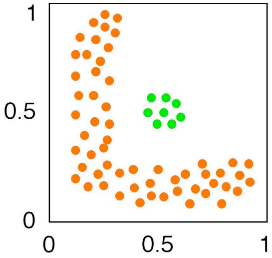
This is typical for the two-level data... and for multi-level data?

- The F-measure generally improves on small classes.

Comparisons

Two alternative approaches to rare classes:

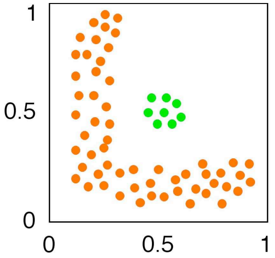
- over-sampling; and
- under-sampling.



Comparisons

Two alternative approaches to rare classes:

- over-sampling; and
- under-sampling.



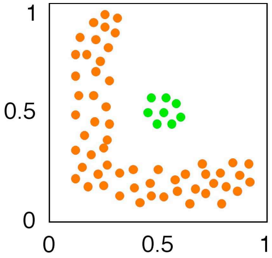
Apply these to the two-level data prior to SVM and...

- COG(SVM) is better on the rare *and* the common classes.

Comparisons

Two alternative approaches to rare classes:

- over-sampling; and
- under-sampling.



Apply these to the two-level data prior to SVM and...

- COG(SVM) is better on the rare *and* the common classes.

Further comparisons

A little modesty:

- COG does not improve nonlinear classifiers; and
- can impede performance.

Further comparisons

A little modesty:

- COG does not improve nonlinear classifiers; and
- can impede performance.

Final test: random partitioning is not as good as `kmeans` in the local clustering phase.

Local Clustering: `kmeans`

Some cleverness...

Local Clustering: `kmeans`

Some cleverness...also some confusion.

> “[`kmeans`] *tends to produce clusters with relatively uniform sizes*”

Local Clustering: `kmeans`

Some cleverness...also some confusion.

> "[`kmeans`] *tends to produce clusters with relatively uniform sizes*"

Better justification is efficiency and nonconvexity of clusters.

Local Clustering: `kmeans`

Some cleverness...also some confusion.

> "[`kmeans`] *tends to produce clusters with relatively uniform sizes*"

Better justification is efficiency and nonconvexity of clusters.

Other issues with `kmeans`:

- it is sensitive to noise; and
- requires highly nontrivial input – k!

Local Clustering: `kmeans`

Some cleverness...also some confusion.

> "[`kmeans`] *tends to produce clusters with relatively uniform sizes*"

Better justification is efficiency and nonconvexity of clusters.

Other issues with `kmeans`:

- it is sensitive to noise; and
- requires highly nontrivial input – k!

data set	#1	#2	#3	#4	#5
optimal k	8	8	6	6	2

Over-sampling

To over-sample they simply replicate. Now if rarity is:

- relative – then this is OK.
- absolute – then it is NOT!

Concluding Remarks

COG is a useful combination of:

- local clustering (unsupervised); and
- linear classification (supervised).

It generally improves performance.

Yet questions remain...