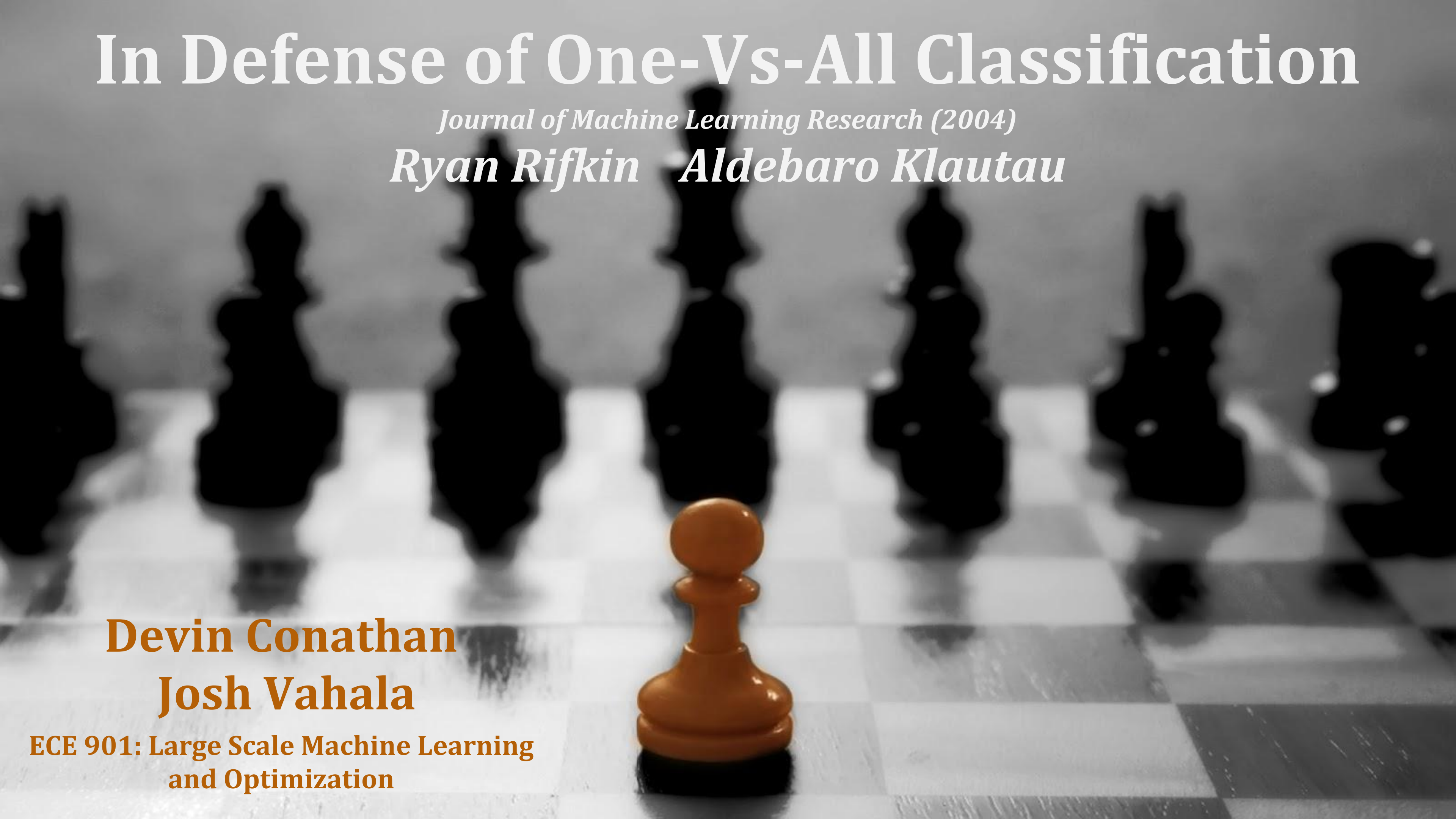


In Defense of One-Vs-All Classification

Journal of Machine Learning Research (2004)

Ryan Rifkin   Aldebaro Klautau

Devin Conathan
Josh Vahala

ECE 901: Large Scale Machine Learning and Optimization

Context: Multiclass Classification

Many machine learning classification problems are ***multiclass*** in nature

The loss function and optimization still fit our optimization framework

'plane'

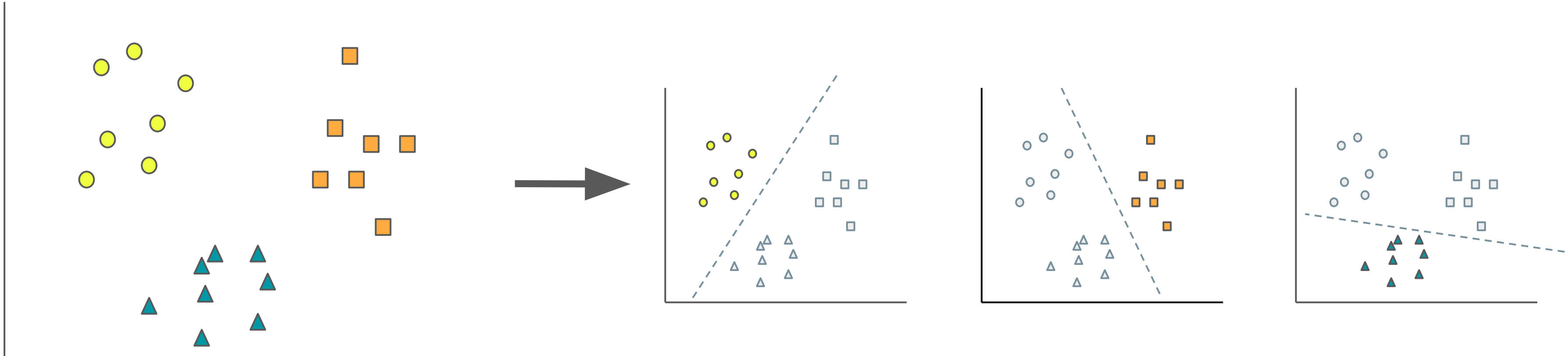
$$\min_f \sum_{i=1}^{n} L(f(x_i), y_i) + R(f)$$

'cat'

'dog'

One vs. All Classification (OVA)

- About the simplest thing you could do
- For a multiclass problem with *C* classes: train *C* classifiers, one for each class
  - A new sample gets assigned class with the greatest predicted probability
- Simple to implement and embarrassingly parallelizable
- Python's scikit-learn provides a `OneVsRestClassifier` wrapper
- Alternatively, consider All vs. All Classification (AVA), where you train *C(C-1)/2* classifiers: one for each possible pair of classes



y_i = [*a, b, c, a, a, c, b, ...*]

y_i = [1, 0, 0, 1, 1, 0, 0, ...]

y_i = [0, 1, 0, 0, 0, 0, 1, ...]

y_i = [0, 0, 1, 0, 0, 1, 0, ...]

But...

Some people decided that it's more fun to do things in way more complicated ways

This paper is essentially a literature review for these more complicated ways

The general theme is that the more complicated ways are:

- harder to implement
- slower to train and often computationally infeasible
- provide negligible (if any) performance boost

Asymptotic Single-Machine Approach

Theoretical motivation:

Derive a multiclass SVM that asymptotically behaves like the Bayes-optimal solution

If $p(\mathbf{x})$ is the probability that $\mathbf{x}$ is in some class, then:

the minimizer of $E[(1 - yf(\mathbf{x}))_+)]$ is $f(\mathbf{x}) = \text{sign}(p(\mathbf{x}) - \frac{1}{2})$

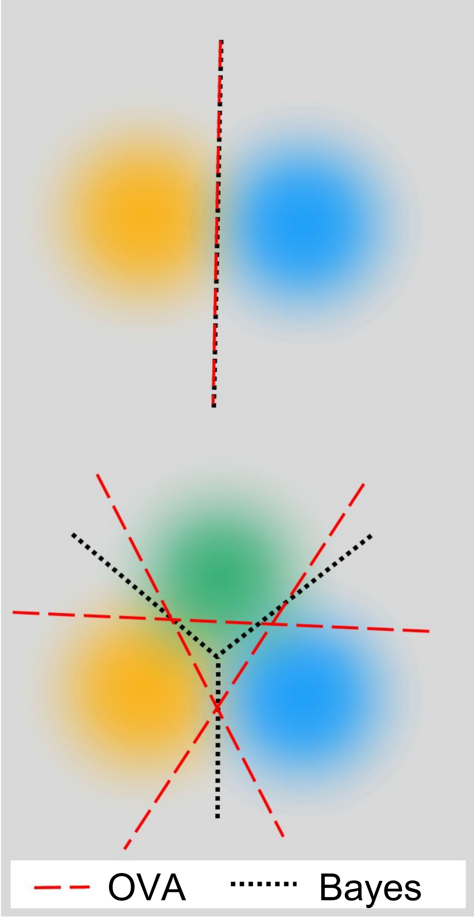
(the Bayes-optimal solution)

One-VS-All SVM does NOT have this property:

$f_i(\mathbf{x}) \to \text{sign}(p_i(\mathbf{x}) - \frac{1}{2})$ as $\ell \to \infty$

$\arg\max_i p_i(\mathbf{x}) \geq \frac{1}{2}$ $\quad f_i(\mathbf{x}) = 1$, and $f_j(\mathbf{x}) = -1$ for $j \neq i$

$\arg\max_i p_i(\mathbf{x}) < \frac{1}{2}$ $\quad f_i(\mathbf{x}) = -1 \ \forall i$



Method

Define a target vector v_i for $1 \leq i \leq N$

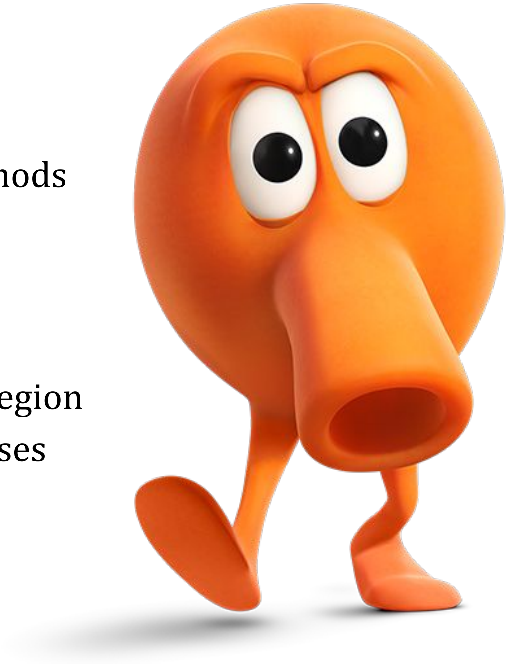
$$v_i = \begin{array}{cl} \begin{bmatrix} -\frac{1}{N-1} \\ \vdots \\ 1 \\ \vdots \\ -\frac{1}{N-1} \end{bmatrix} & \begin{array}{l} \\ \\ \longleftarrow i\text{th coordinate} \\ \\ \\ \end{array} \end{array}$$

$$\boxed{\begin{array}{ll} \min\limits_{f_1,\ldots,f_N \in \mathcal{H}_K} & \frac{1}{\ell}\sum_{i=1}^{\ell}\sum_{j=1,j\neq y_i}^{N}\left(f_j(\mathbf{x}_i) + \frac{1}{N-1}\right)_+ + \lambda\sum_{j=1}^{C}||f_j||_K^2 \\ \text{subject to :} & \sum_{j=1}^{C} f_j(\mathbf{x}) = 0, \qquad \forall \mathbf{x}. \end{array}}$$

Then, $f_i(\mathbf{x}) = 1$ if class i is the most likely and $f_i(\mathbf{x}) = -\frac{1}{N-1}$ otherwise.

Problems

1) Entirely asymptotic
   a) Equivalent to other asymptotically accurate density estimation methods
   b) With limited data, discriminative methods like SVM perform better
   c) Ignores regularization
2) Overlapping class densities create additional issues
   a) Classification accuracy is dependent on the most likely class in the region
   b) High dimensional data will require many points to differentiate classes



Multiclass Generalization of SVMs

Standard approach: find $f(x) = \sum_{j=1}^{\ell} c_j K(\mathbf{x}, \mathbf{x_j}) + b.$

Proposed approach: find $f_i(x) = \sum_{j=1} c_{ij} K(\mathbf{x}, \mathbf{x_j}) + b_i.$ ← find N functions $\{f_1, f_2, \dots, f_N\}$ at once

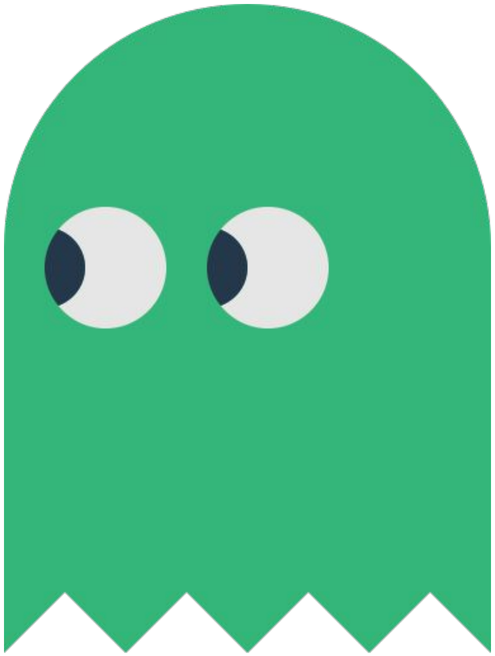
Instead of incurring cost for each machine (f), incur cost relative to values from other machines

Vapnik and Blantz (1998), *Weston and Watkins* (1998)	*Crammer and Singer* (2001)
$\min_{\mathbf{f_1},\dots,\mathbf{f_N} \in \mathcal{H}, \xi \in \mathbb{R}^{\ell(\mathbf{N-1})}} \sum_{i=1}^{N} \|\|f_i\|\|_K^2 + C \sum_{i=1}^{\ell} \sum_{j \neq y_i} \xi_{ij}$ $\text{subject to}: \; f_{y_i}(\mathbf{x_i}) + b_{y_i} \geq f_j(\mathbf{x_i}) + b_j + 2 - \xi_{ij},$ $\xi_{ij} \geq 0.$ $(N-1)\ell$ slack variables	$\min_{\mathbf{f_1},\dots,\mathbf{f_N} \in \mathcal{H}, \xi \in \mathbb{R}^{\ell}} \sum_{i=1}^{N} \|\|f_i\|\|_K^2 + C \sum_{i=1}^{\ell} \xi_i$ $\text{subject to}: \; f_{y_i}(\mathbf{x_i}) \geq f_j(\mathbf{x_i}) + 1 - \xi_i,$ $\xi_i \geq 0.$ ℓ slack variables

where $K(\mathbf{x_1}, \mathbf{x_2}) = \exp^{-\gamma\|\|\mathbf{x_1} - \mathbf{x_2}\|\|^2}$, and $\|\|f_i\|\|_K^2 = \mathbf{c_i}.^T K \mathbf{c_i}.,$

Multiclass Generalization of SVMs

1) Claimed performance boost over OVA
    a) Likely didn’t tune OVA parameters well
    b) Similar performance overall, but harder optimization problem



Error-Correcting Code Approaches

Define a matrix $M \in \{-1, 1\}^{N \times F}$

where N is the number of classes and F is the number of machines

jth machine solves $$\min \sum_{i=1}^{\ell} V(f_j(\mathbf{x_i}), M_{y_i j}) + \lambda ||f_j||_K^2$$

To classify a new $\boldsymbol{x}$, calculate $[f_1, f_2, \ldots, f_F]$ and choose the class that minimizes the **Hamming distance** from the corresponding row of M:

$$f(\mathbf{x}) = \arg \min_{r \in 1, \ldots, N} \sum_{i=1}^{F} \left(\frac{1 - \text{sign}(M_{ri} f_i(\mathbf{x}))}{2} \right)$$

machine-2 assigns class-1 to its positive metaclass

$$\text{class } (N{=}3) \quad \begin{array}{c} 1 \\ 2 \\ 3 \end{array} \begin{bmatrix} 1 & \boxed{1} & -1 & 1 \\ -1 & 1 & 1 & -1 \\ -1 & -1 & 1 & 1 \end{bmatrix}$$

$$\begin{array}{cccc} 1 & 2 & 3 & 4 \end{array}$$

machine $(F{=}4)$

Dietterich and Bakiri (1995)

Improved Error-Correcting Code Approach

Expand the previous concept to allow zeros in *M*:

$$M \in \{-1, 0, 1\}^{N \times F}$$

This framework now encapsulates *OVA, AVA,* and general *Error-Correcting Code* classifiers

Also, introduce a more general **loss-based decoding** scheme, which significantly improves performance over Hamming distance:

$$f(\mathbf{x}) = \arg \min_{r \in 1,\ldots,N} \sum_{i=1}^{F} L(M_{ri} f_i(\mathbf{x})).$$

$$\text{class } (N{=}4) \quad \begin{bmatrix} 1 & -1 & -1 & -1 \\ -1 & 1 & -1 & -1 \\ -1 & -1 & 1 & -1 \\ -1 & -1 & -1 & 1 \end{bmatrix}$$

One-vs-All

$$\text{class } (N{=}4) \quad \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 1 & 0 \\ 0 & -1 & 0 & -1 & 0 & 1 \\ 0 & 0 & -1 & 0 & -1 & -1 \end{bmatrix}$$

All-vs-All

Allwein, Shapire, and Singer (2000)

Error-Correcting Code Problems

1. Lots of tuning
   a. For general error-correcting, not clear what matrix is the best
   b. If you have many classes, large number of possibilities
2. Difference in performance is negligible
3. Results reported in *Allwein et al.* (2000) show better performance than OVA, but their OVA SVM classifiers were not tuned

Theoretical Results

If the following conditions hold:

- Underlying classifier is a regularized least squares classifier ($RLSC$)
- The coding scheme's classifiers are independent
- The coding matrix contains no zeros

Then the problem reduces to an OVA multiclass classifier

i.e. the predictions will be ***exactly the same***

In particular, the *complete coding scheme*, which is the matrix that contains all unique {+1, -1} codes, has these properties.

Experiments and Results

Name	OVA	AVA	COM	DEN	SPA
soybean-large	19	171	262143	43	64
letter	26	325	33554431	48	71
satimage	6	15	31	26	39
abalone	29	406	268435455	49	73
optdigits	10	45	511	34	50
glass	6	15	31	26	39
car	4	6	7	20	30
spectrometer	48	1128	1.407e+014	56	84
yeast	10	45	511	34	50
page-blocks	5	10	15	24	35

Table 5: Number of possible binary classifiers for each code matrix.

Experiments and Results

Data Set	AVA	OVA	DIFF	AGREE	BOOTSTRAP
soybean-large	6.38	5.85	0.530	0.971	[-0.008, 0.019]
letter	3.85	2.75	1.09	0.978	[0.008, 0.015]
satimage	8.15	7.80	0.350	0.984	[-5E-4, 0.008]
abalone	72.32	79.69	-7.37	0.347	[-0.102, -0.047]
optdigits	3.78	2.73	1.05	0.982	[0.006, 0.016]
glass	30.37	30.84	-.470	0.818	[-0.047, 0.037]
car	0.41	1.50	-1.09	0.987	[-0.016, -0.006]
spectrometer	42.75	53.67	-10.920	0.635	[-0.143, -0.075]
yeast	41.04	40.30	0.740	0.855	[-0.006, 0.021]
page-blocks	3.38	3.40	-.020	0.991	[-0.002, 0.002]

Table 6: SVM test error rate (%), OVA vs. AVA.

Data Set	DEN	OVA	DIFF	AGREE	BOOTSTRAP
soybean-large	5.58	5.85	-0.270	0.963	[-0.019, 0.013]
letter	2.95	2.75	0.200	0.994	[5E-4, 0.004]
satimage	7.65	7.80	-0.150	0.985	[-0.006, 0.003]
abalone	73.18	79.69	-6.51	0.393	[-0.092, -0.039]
optdigits	2.61	2.73	-0.12	0.993	[-0.004, 0.002]
glass	29.44	30.84	-1.40	0.911	[-0.042, 0.014]
car	-	1.50	-	-	-
spectrometer	54.43	53.67	-0.760	0.866	[-0.011, 0.026]
yeast	40.30	40.30	0.00	0.900	[-0.011, 0.011]
page-blocks	-	3.40	-	-	-

Table 7: SVM test error rate (%), OVA vs. DENSE.

Data Set	SPA	OVA	DIFF	AGREE	BOOTSTRAP
soybean-large	6.12	5.85	0.270	0.968	[-0.011, 0.016]
letter	3.55	2.75	0.800	0.980	[0.005, 0.011]
satimage	8.85	7.80	1.05	0.958	[0.003, 0.018]
abalone	75.67	79.69	-4.02	0.352	[-0.067, -0.014]
optdigits	3.01	2.73	0.280	0.984	[-0.002, 0.008]
glass	28.97	30.84	-1.87	0.738	[-0.070, 0.033]
car	0.81	1.50	-0.69	0.988	[-0.011, -0.003]
spectrometer	52.73	53.67	-0.940	0.744	[-0.038, 0.019]
yeast	40.16	40.30	-0.140	0.855	[-0.015, 0.013]
page-blocks	3.84	3.40	0.440	0.979	[0.001, 0.007]

Table 8: SVM test error rate (%), OVA vs. SPARSE.

Data Set	COM	OVA	DIFF	AGREE	BOOTSTRAP
soybean-large	-	5.85	-	-	-
letter	-	2.75	-	-	-
satimage	7.80	7.80	0.00	0.999	[-1E-3, 1E-3]
abalone	-	79.69	-	-	-
optdigits	2.67	2.73	-0.060	0.996	[-0.003, 0.002]
glass	29.44	30.84	-1.340	0.911	[-0.042, 0.014]
car	1.68	1.50	-0.180	0.998	[5.79E-4, 0.003]
spectrometer	-	53.67	-	-	-
yeast	38.61	40.30	-1.690	0.906	[-0.028, -0.005]
page-blocks	3.49	3.40	-0.090	0.983	[-0.002, 0.004]

Table 9: SVM test error rate (%), OVA vs. COMPLETE.

Takeaways

1) OVA and AVA are very simple to implement and perform well

2) OVA is not significantly outperformed by proposed multiclass methods
    a) This does not mean there isn't a method that will perform better!

3) AVA has a speed advantage to OVA because fewer examples per optimization, but requires more trained classifiers

Questions?