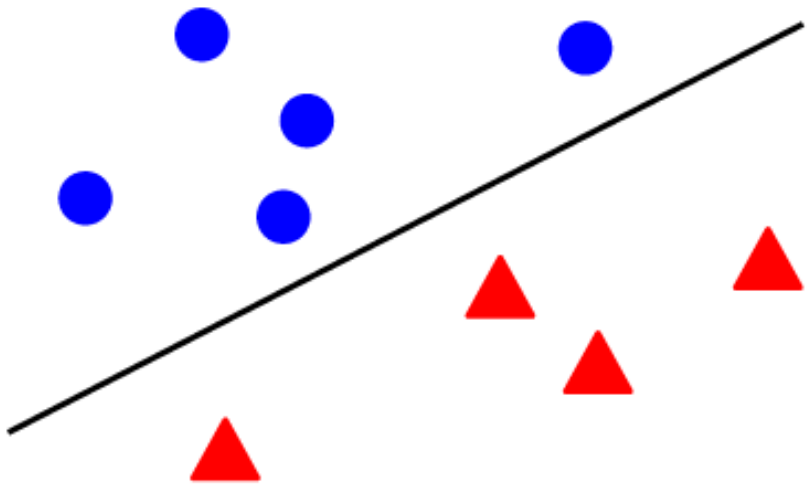
Prototype Support Vector Machines: Supervised Classification in Complex Datasets

April Shen

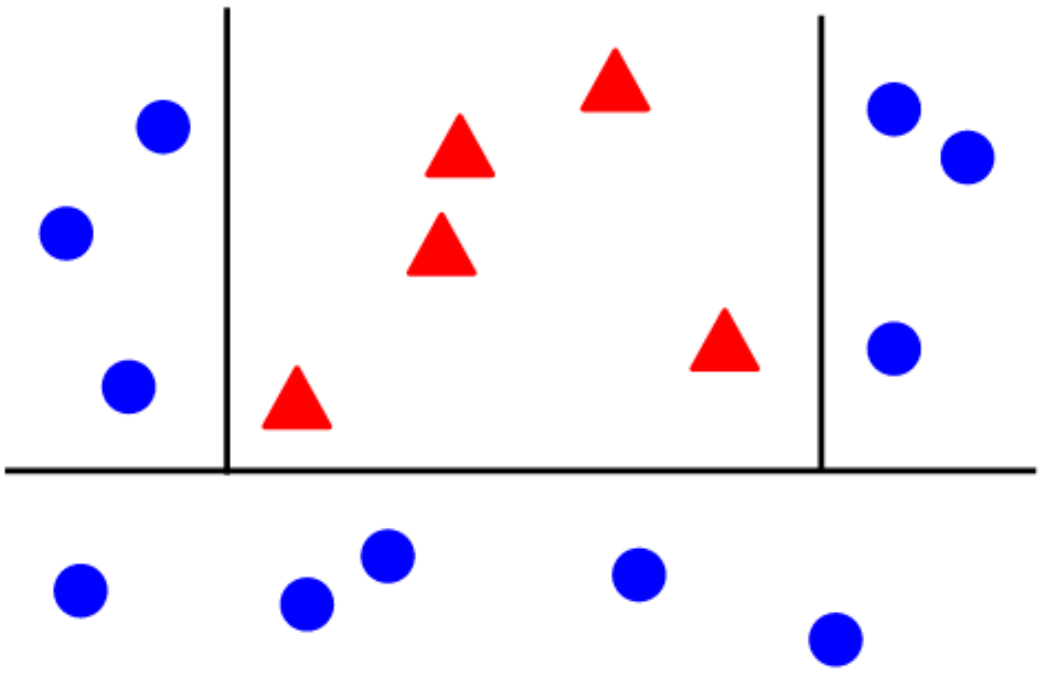
Williams College
Computer Science Department
Advisor: Andrea Danyluk

May 13, 2013

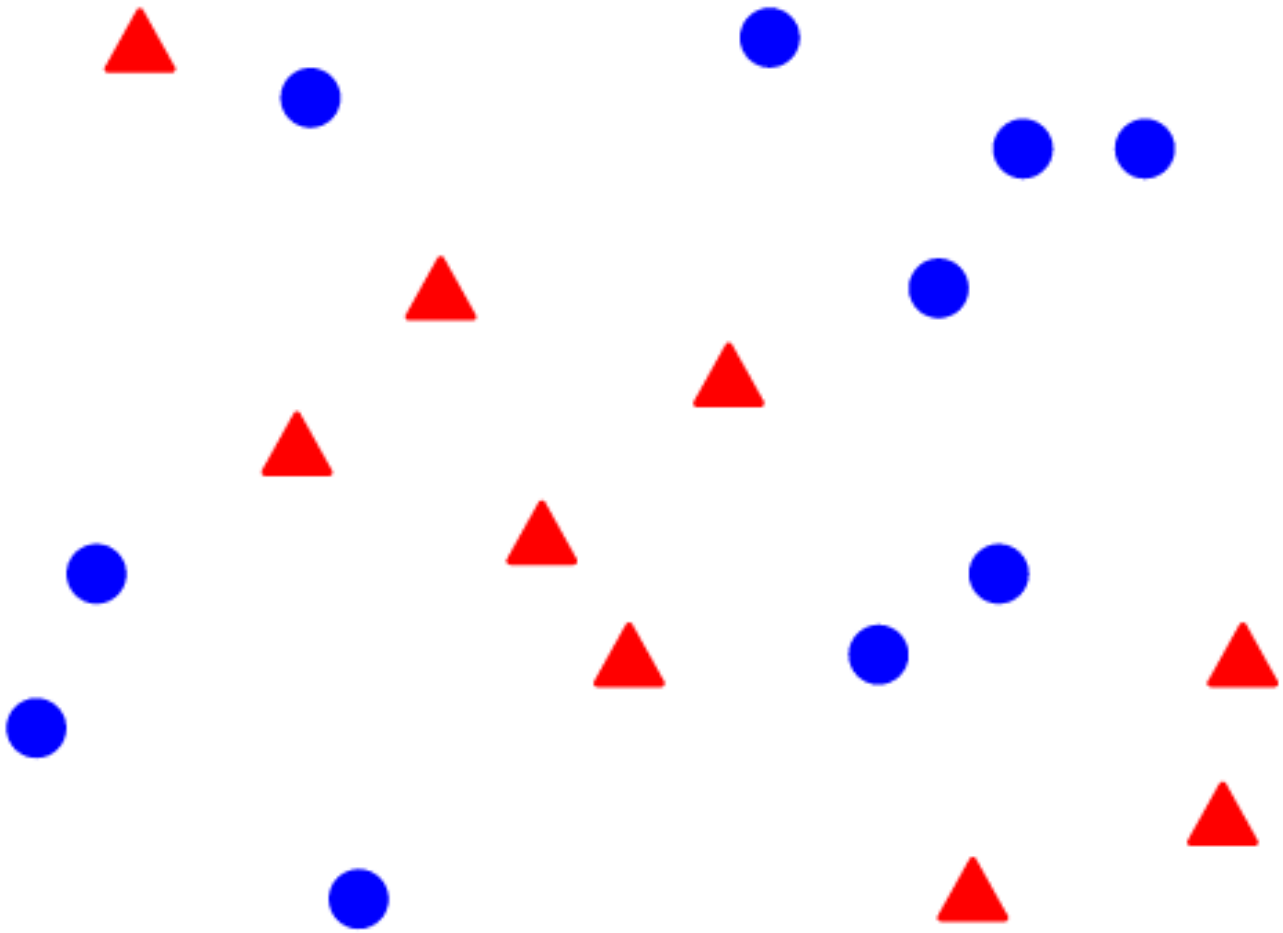
Classification in Complex Datasets



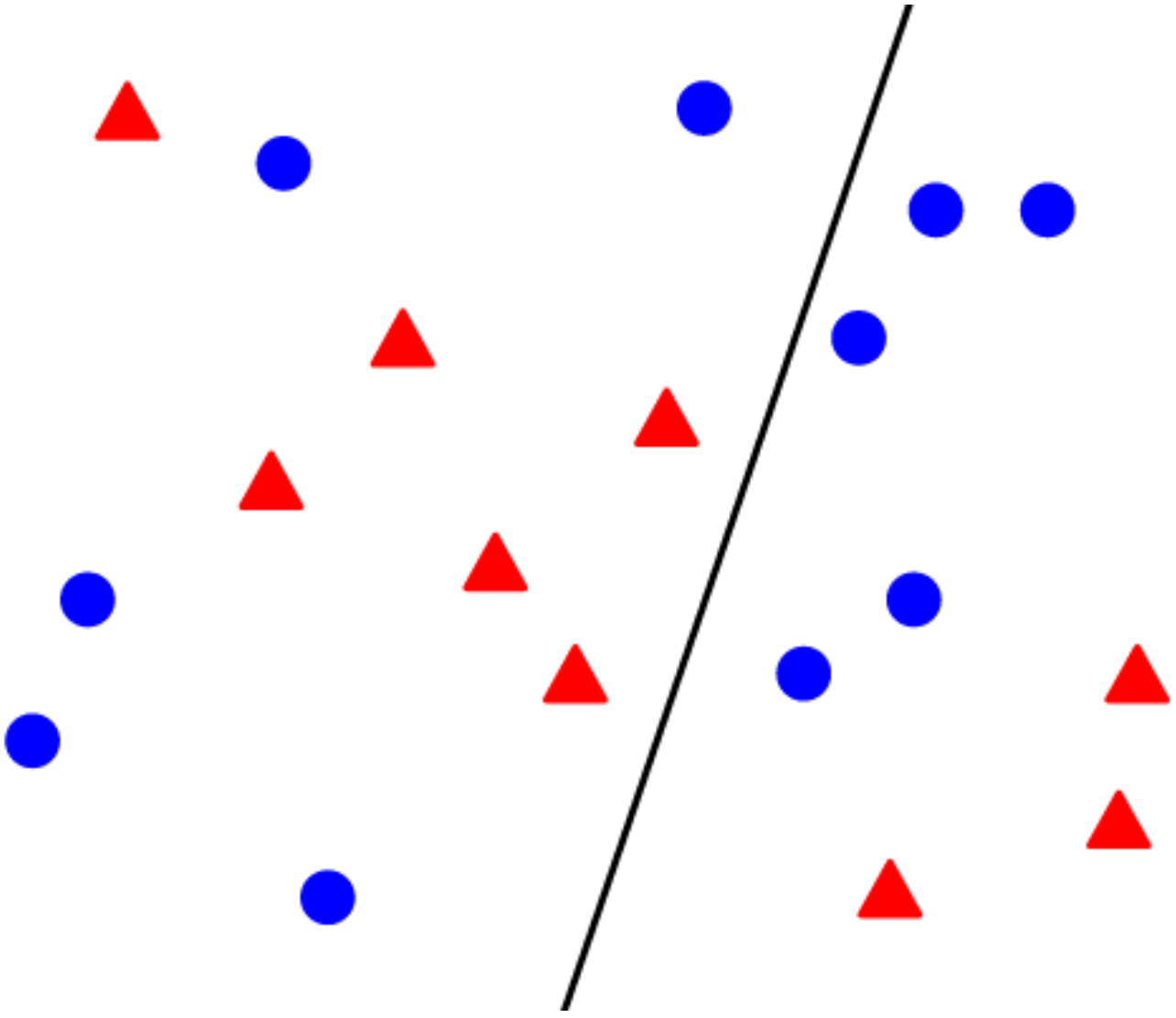
Classification in Complex Datasets



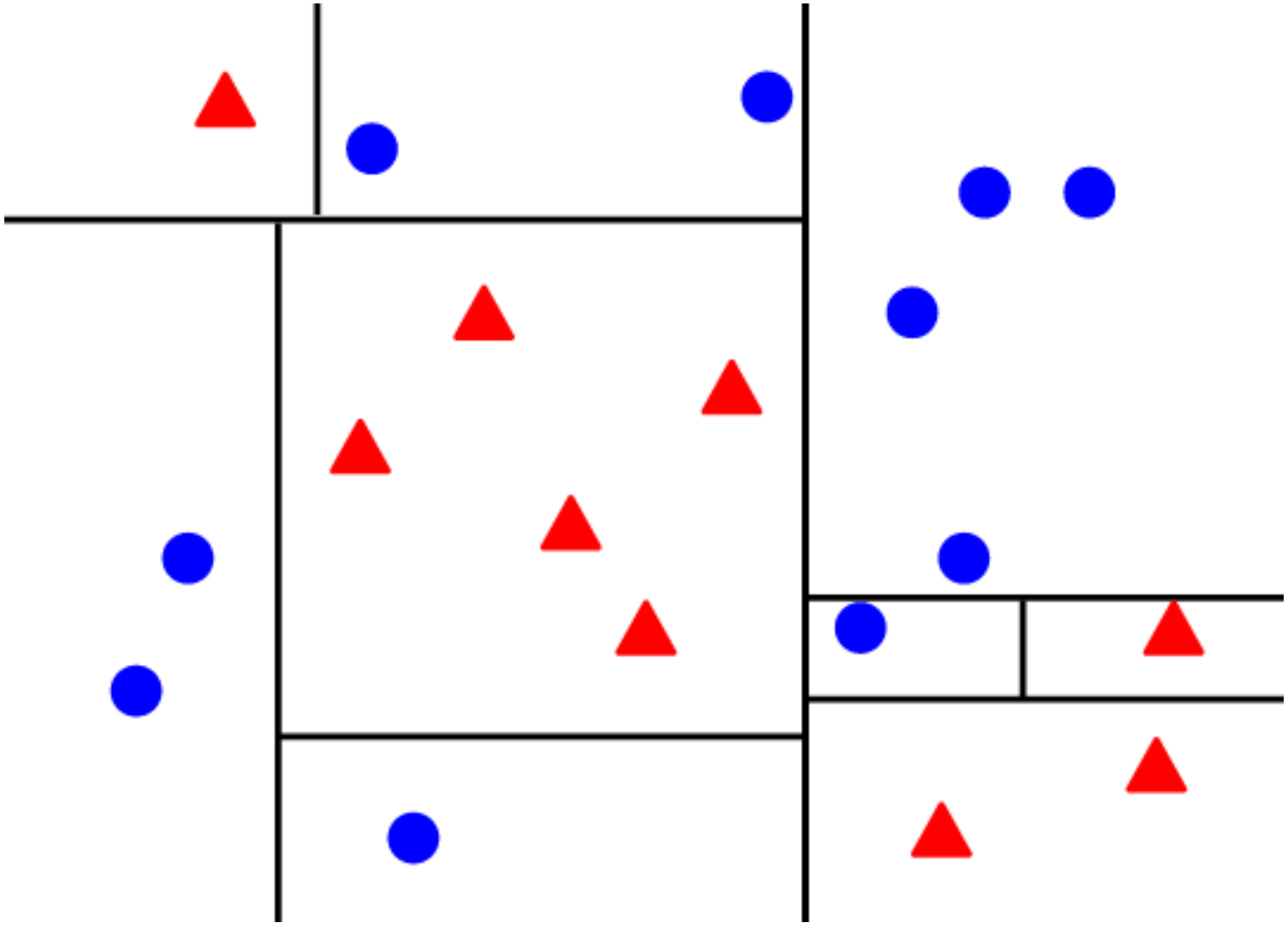
Classification in Complex Datasets



Classification in Complex Datasets



Classification in Complex Datasets



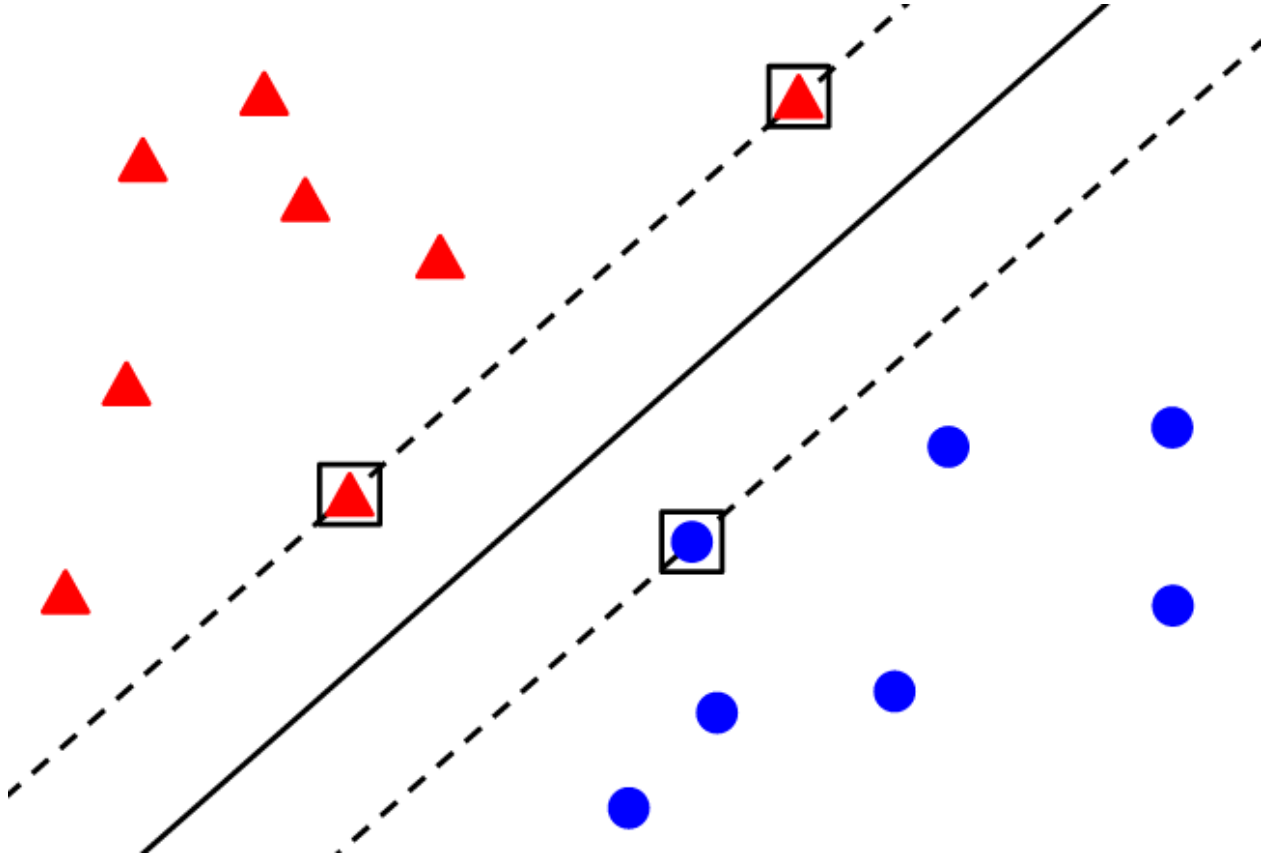
Model Selection

- Finding the right model without knowing anything about your data is tricky
- Requires making assumptions about data distributions

Ensemble Methods: AdaBoost

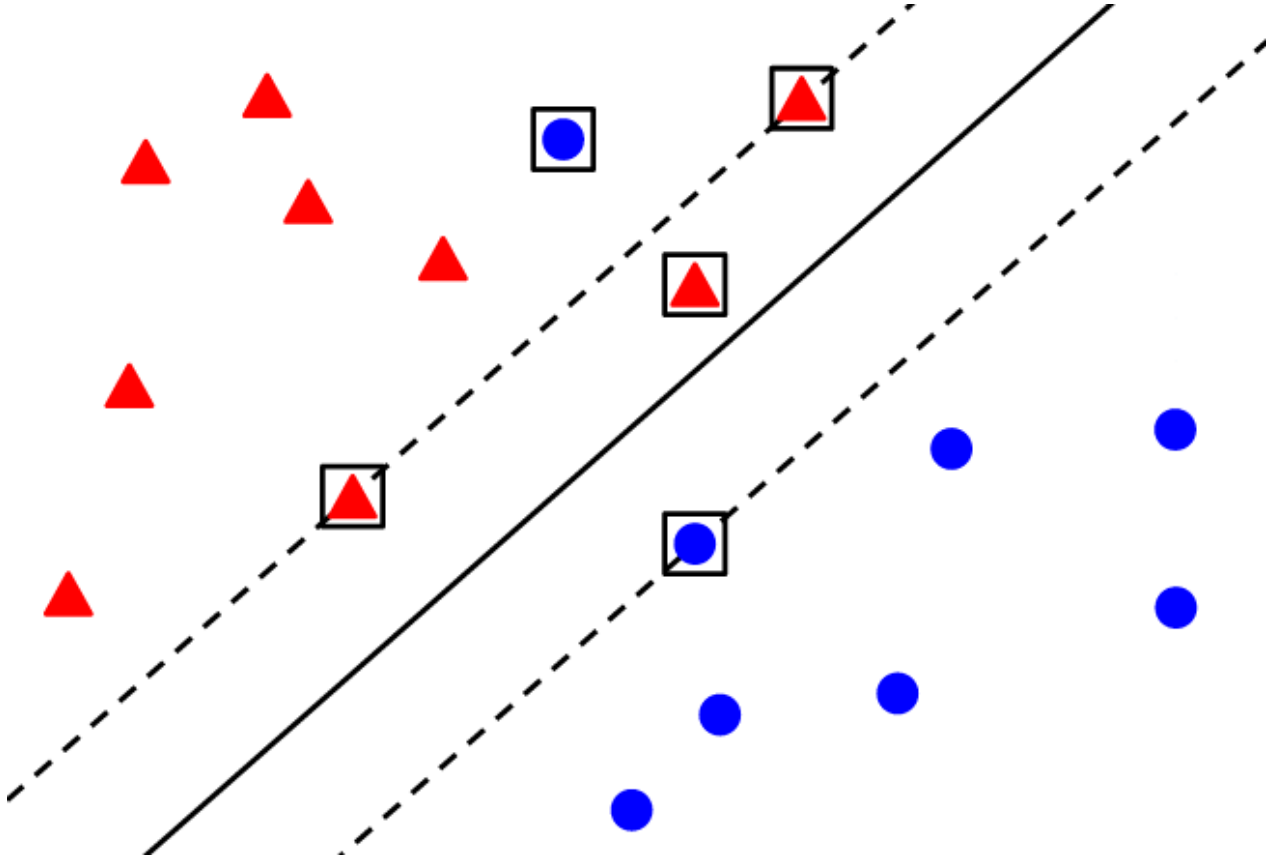
- Freund and Schapire, 1995 [3]
- Iteratively train classifiers on different samples of training data
- Focus on hard-to-capture data
- Requires choice of base classifier

Support Vector Machines (SVMs)



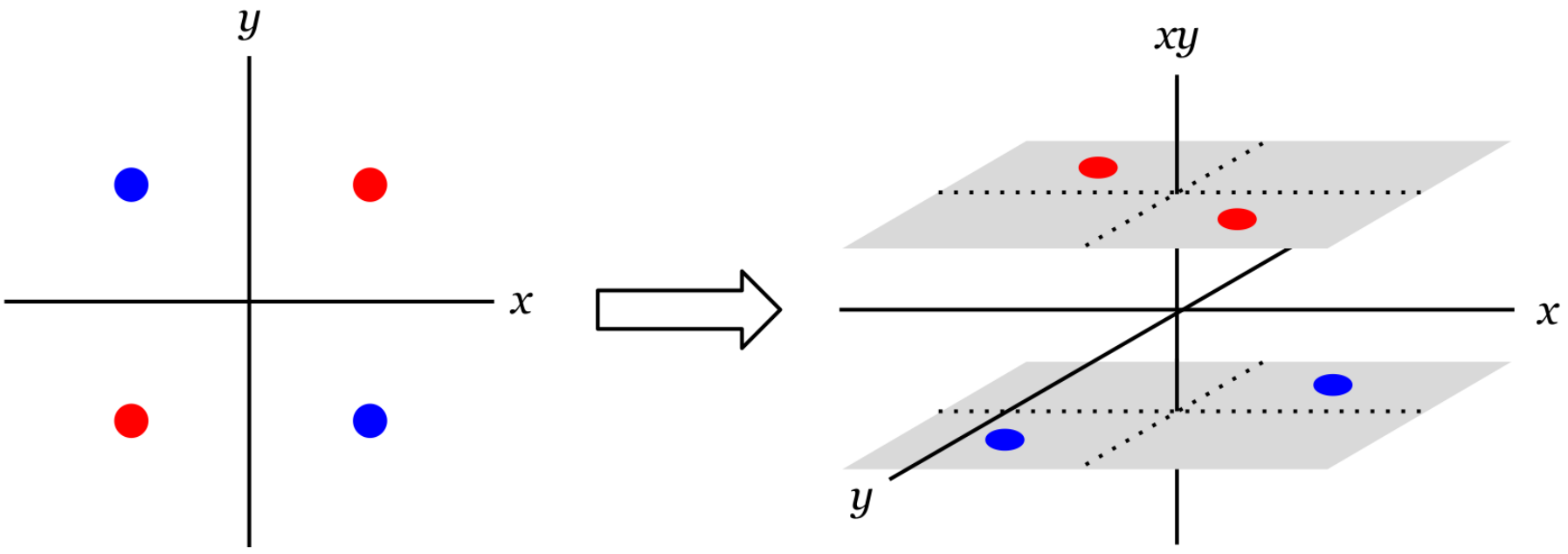
- Cortes and Vapnik, 1995 [1]
- Simplest case: separating hyperplane with minimum error and maximum margin

Support Vector Machines (SVMs)



- Cortes and Vapnik, 1995 [1]
- Simplest case: separating hyperplane with minimum error and maximum margin
- Unseparable case

Support Vector Machines (SVMs)



- Can nonlinearly map to high-dimensional spaces using kernels
- Requires choice of kernel function

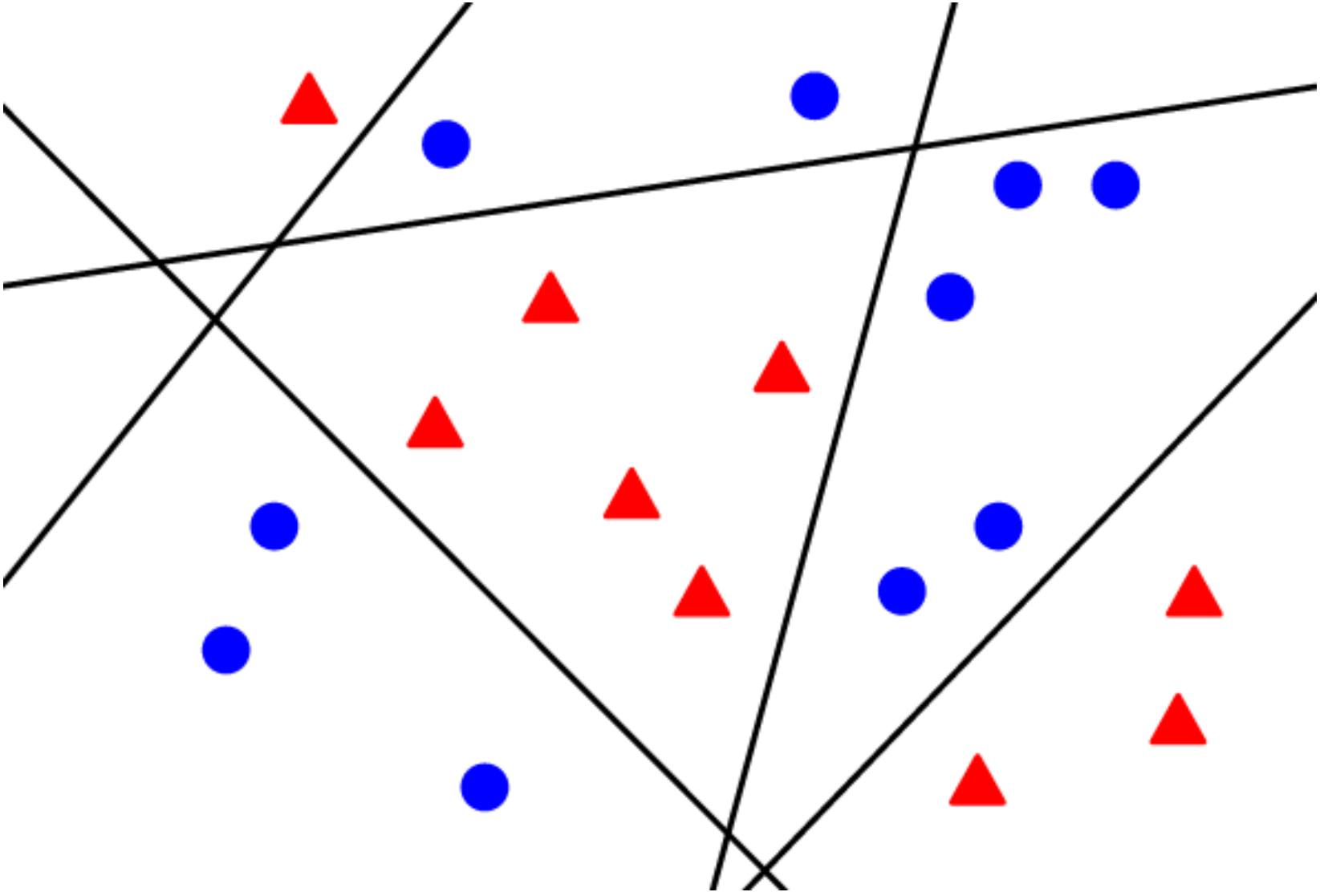
Exemplar SVMs (ESVMs)

- Malisiewicz et al., 2011 [2]
- Train classifier with single positive instance and many negative instances
- Ensemble of these can "vote" on new instances

Strengths and Weaknesses of ESVMs

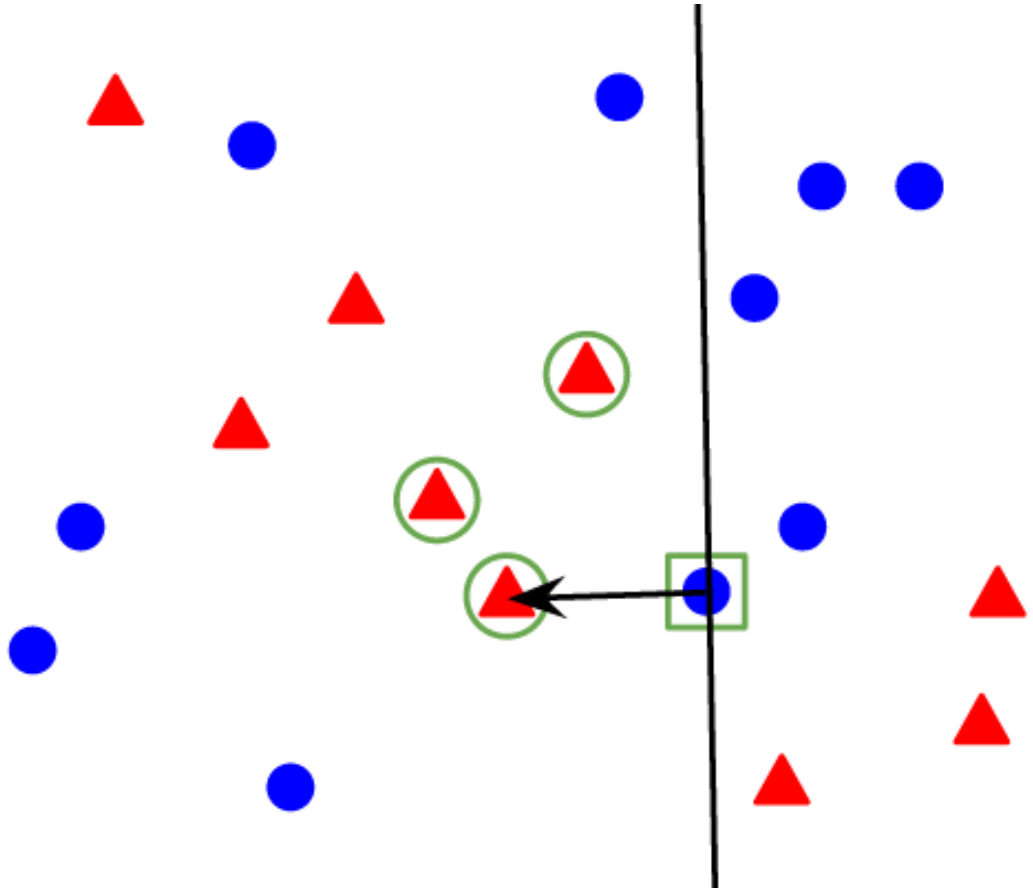
- Strengths
  - Ensemble of classifiers allows high accuracy
  - Placement of models is specific to data distribution
- Weaknesses
  - Fine-tuned to object recognition domain
  - Data does sometime occur in clusters

From Exemplars to Prototypes

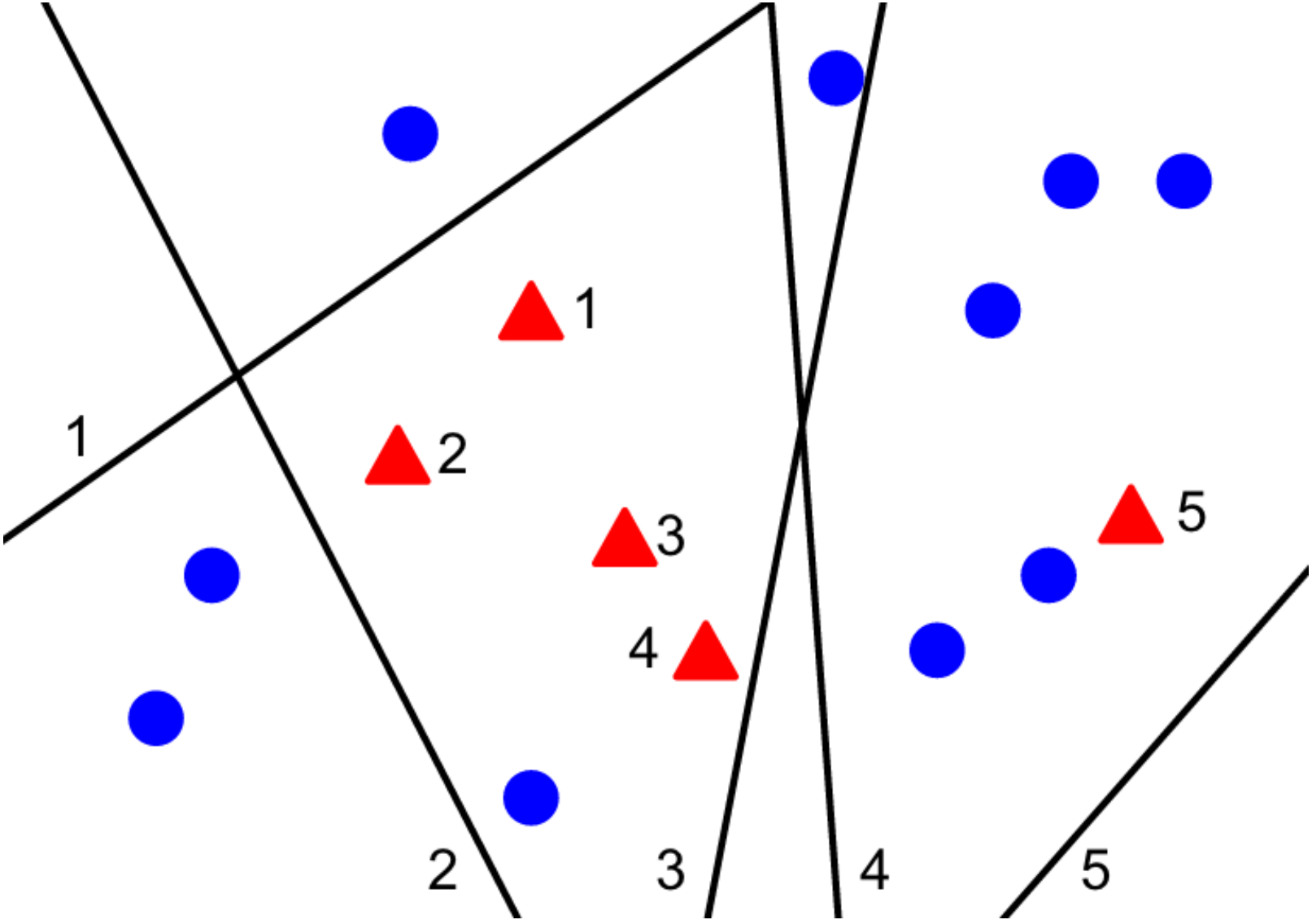


Initialization

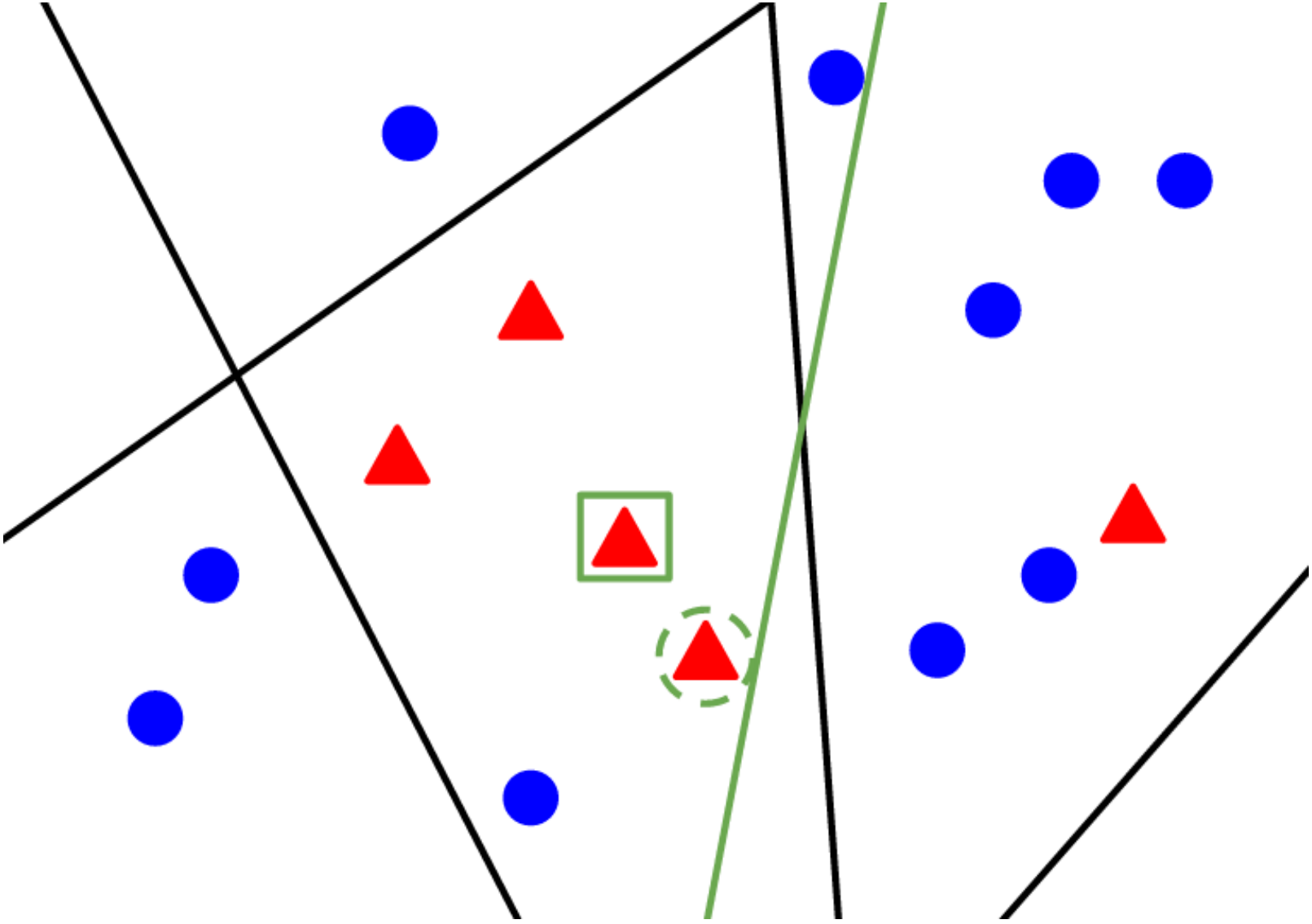
- Seeding the positive sets
- Seeding the negative sets



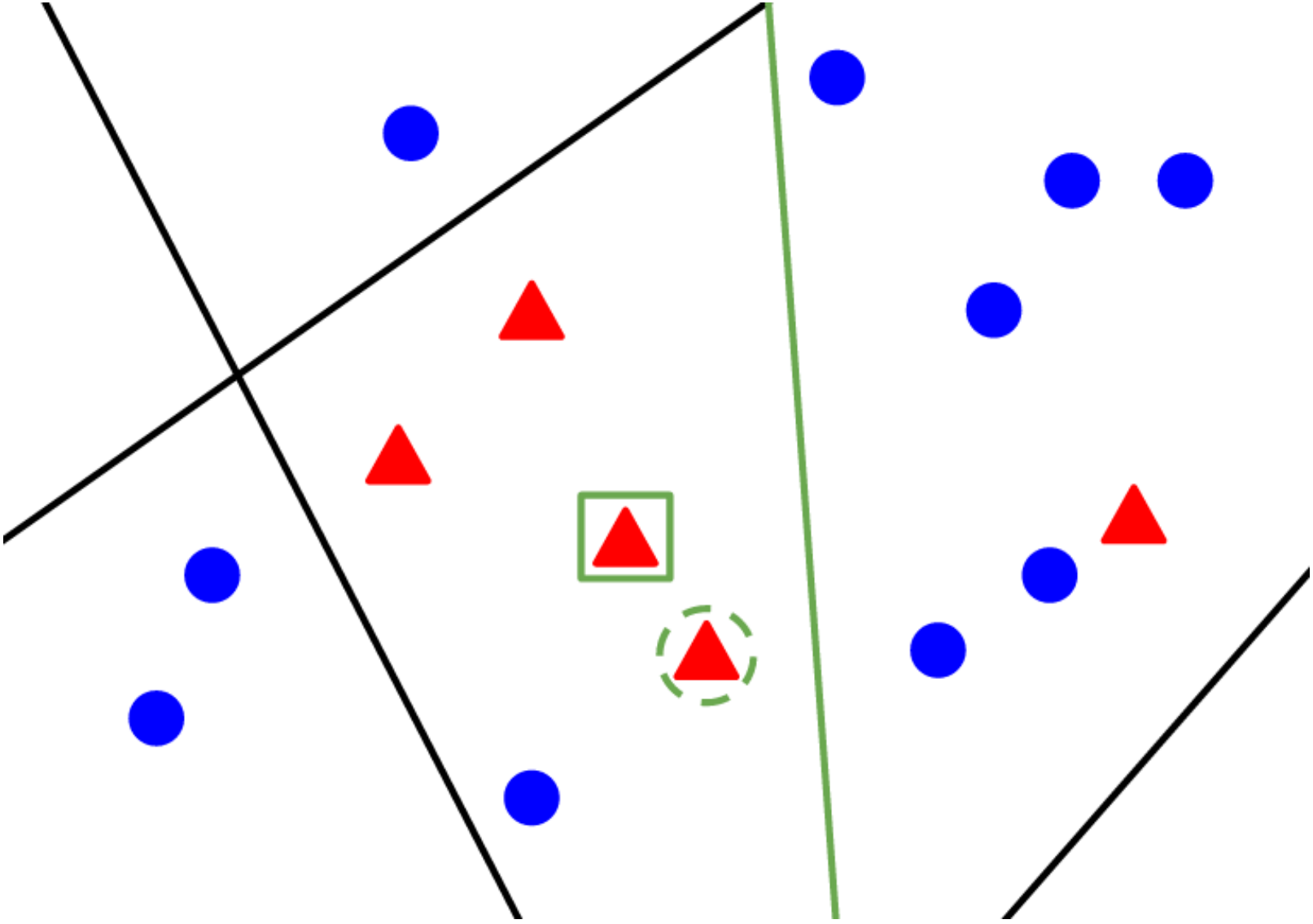
Shifting



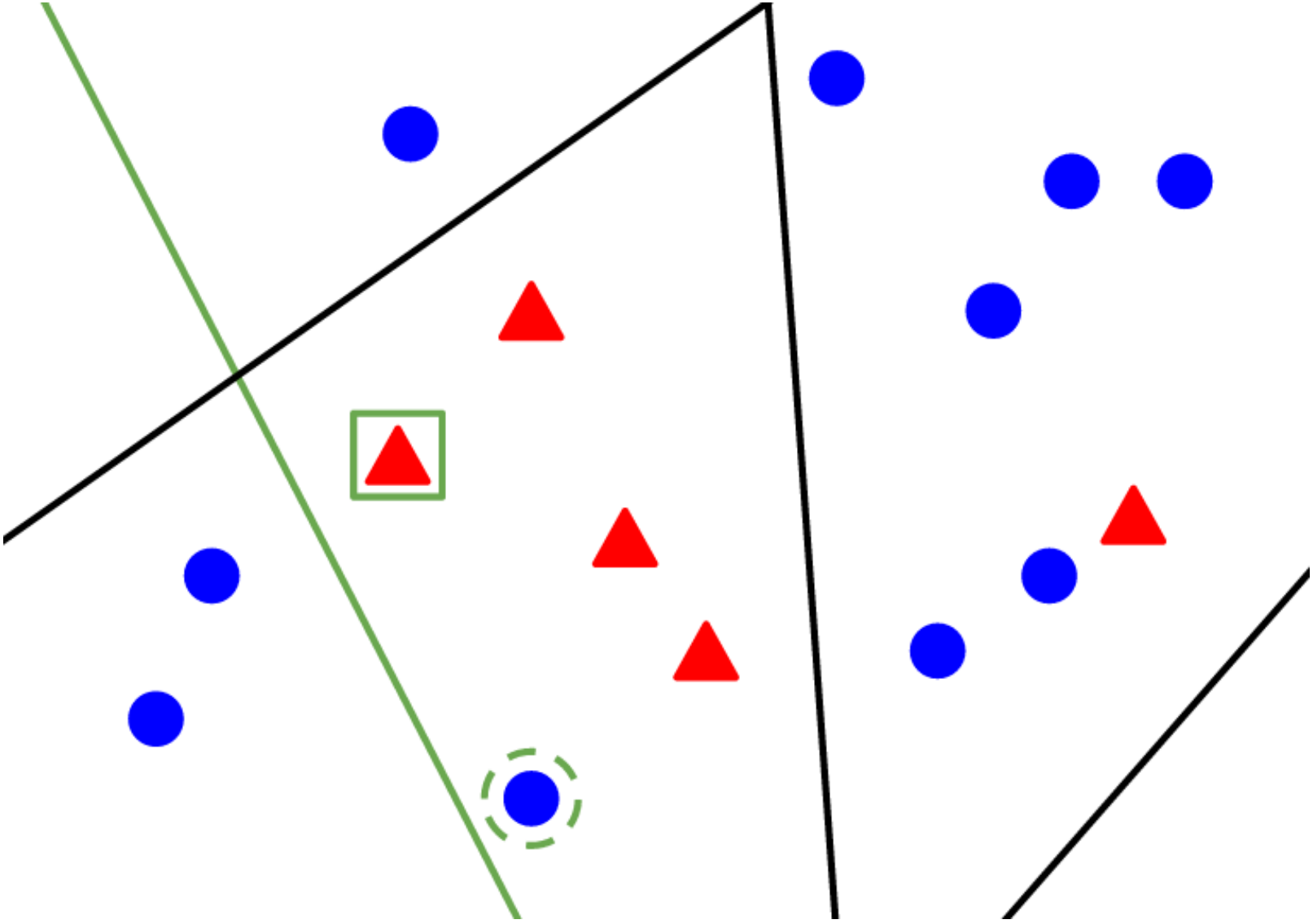
Shifting



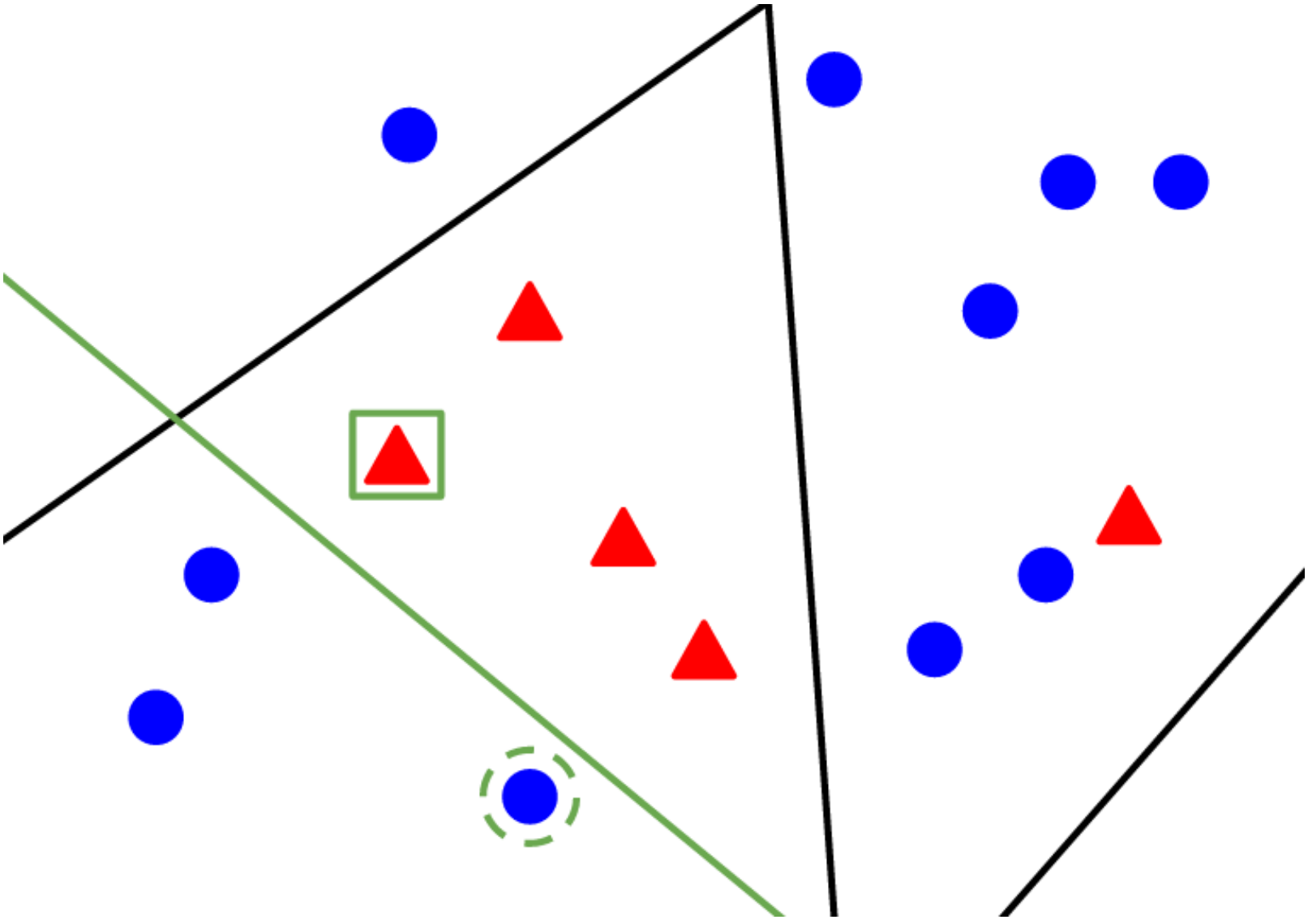
Shifting



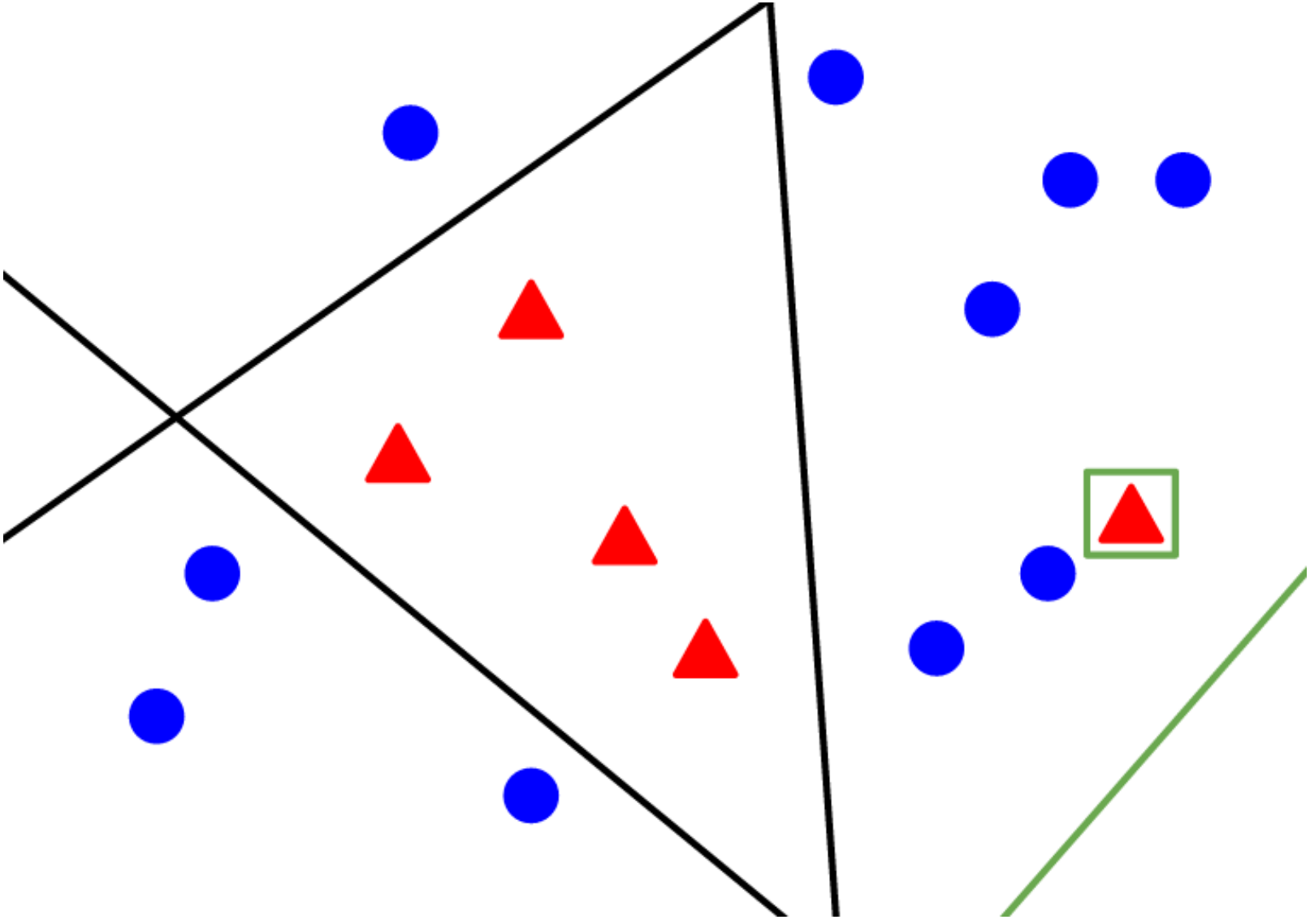
Shifting



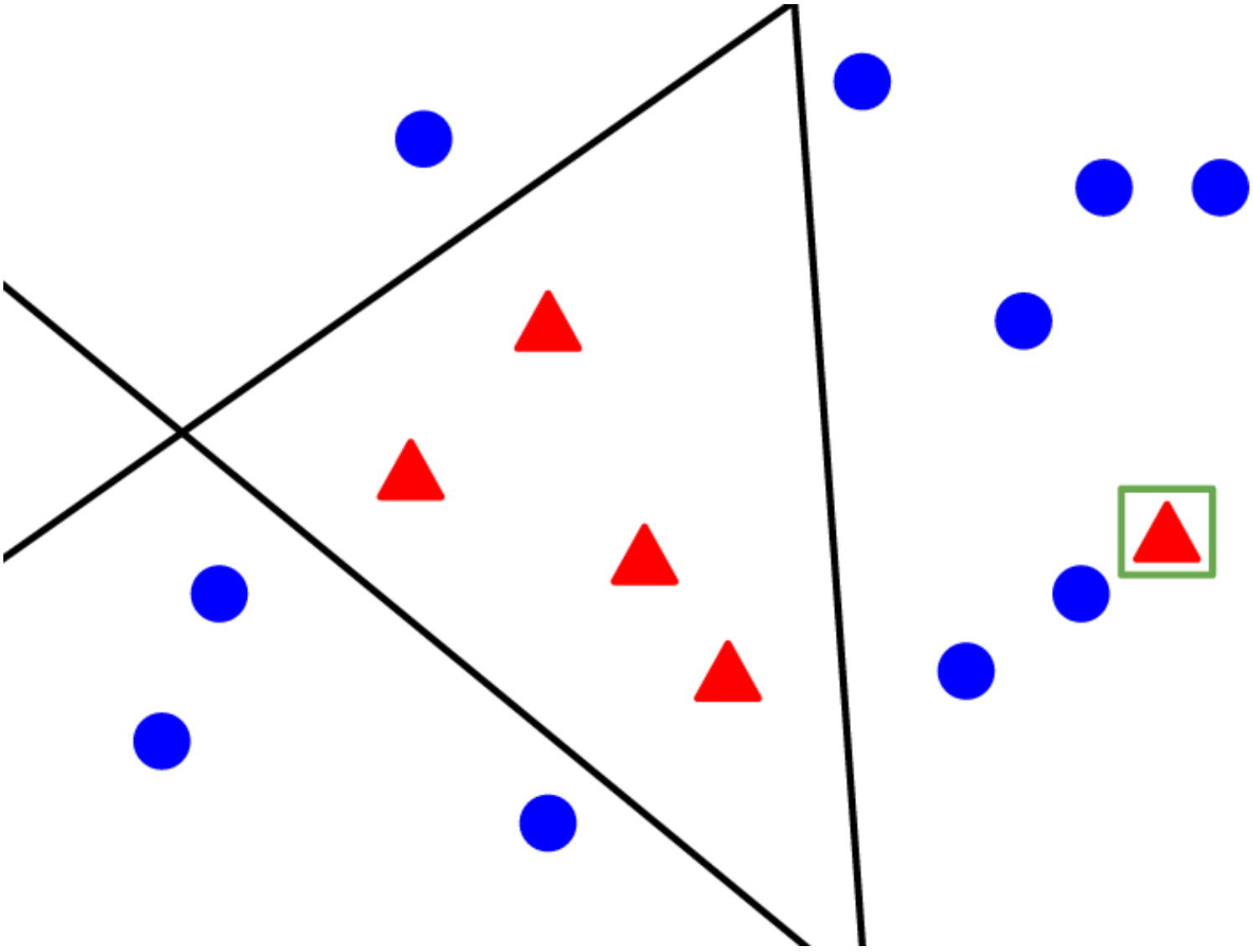
Shifting



Shifting



Shifting



Prediction

- Choose best iteration's ensemble based on accuracy on validation set
- If a model classifies a new instance positively, makes a weighted vote for its class
- Class with maximum number of votes is the predicted class

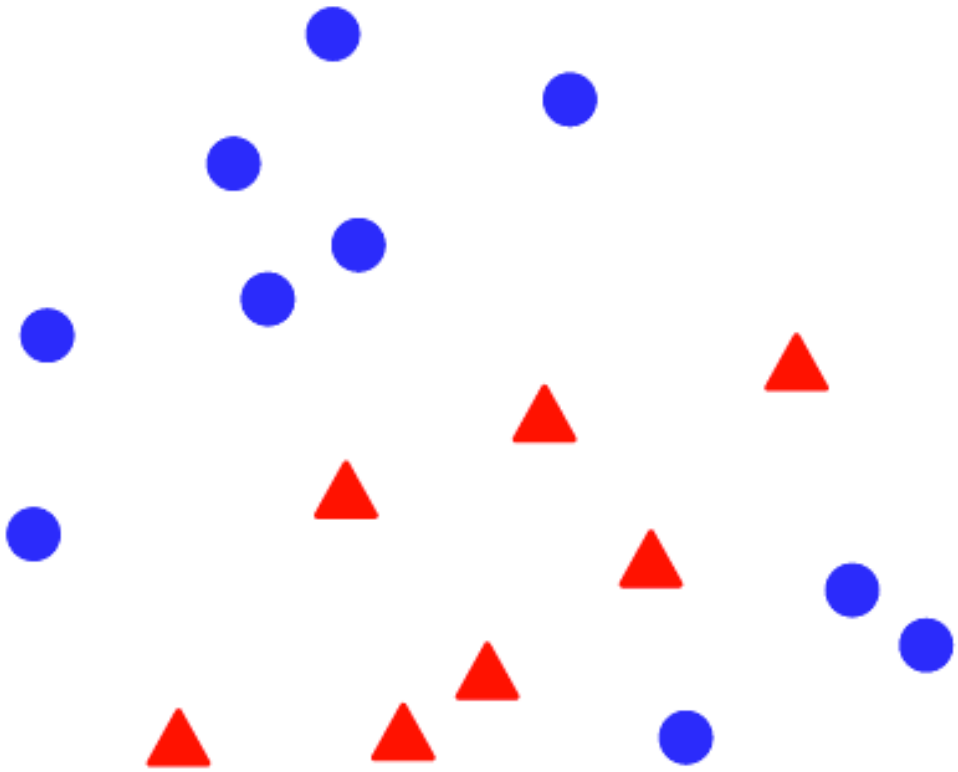
Strengths and Weaknesses

- Strengths
  - Inherits benefits of ESVMs
  - Balances flexibility with performance
  - No need for model selection
- Weaknesses
  - Potentially large ensemble sizes
  - Slower than other algorithms
  - Too many models for simple distributions
  - Too few models for complex distributions

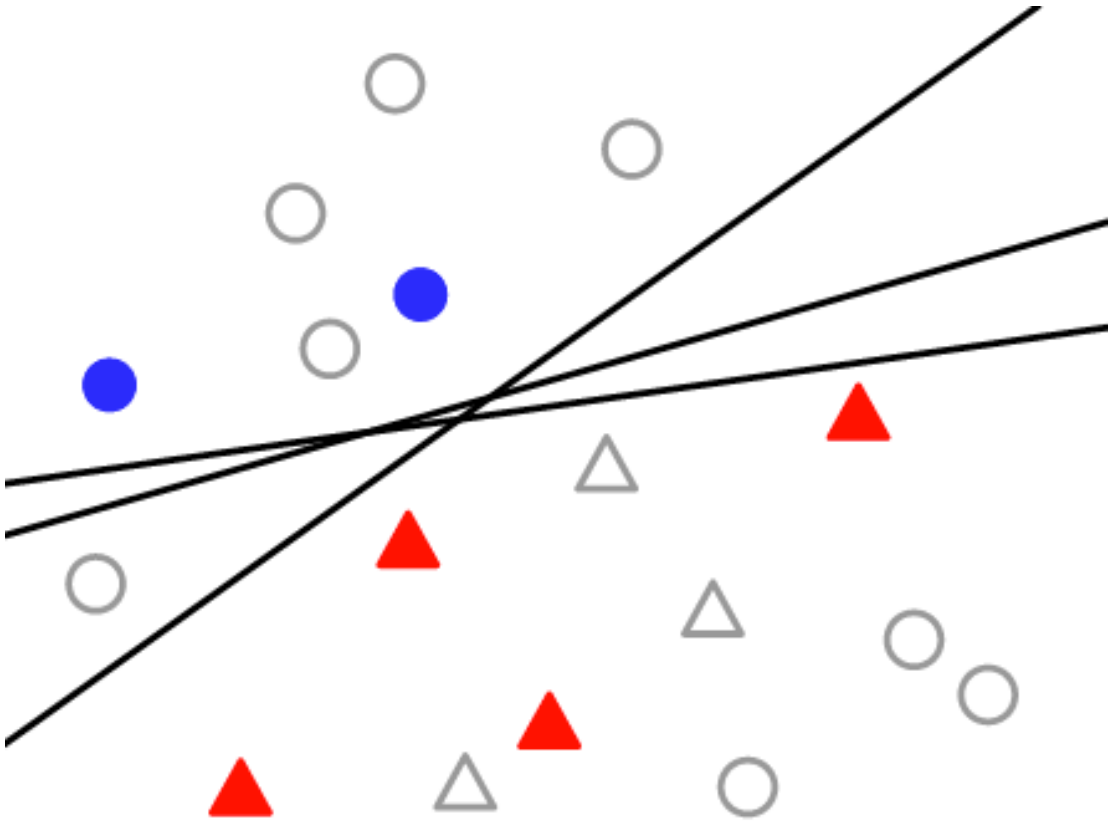
PSVMs with Weighted Sampling

- Sample data, so don't overdo simple regions
- Resample using AdaBoost-style weighting, so focus on difficult regions

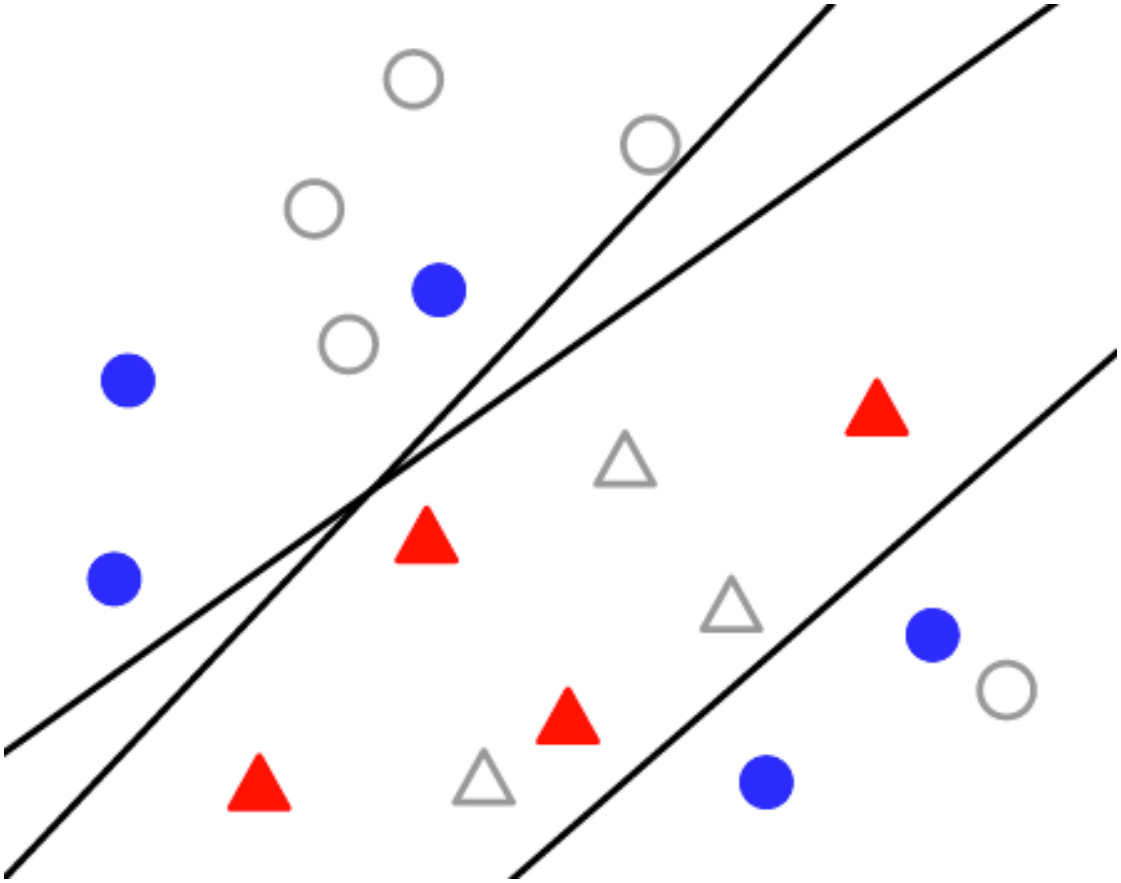
Weighted Sampling



Weighted Sampling



Weighted Sampling



Strengths and Weaknesses

- Strengths
  - Greater sensitivity to complex structure
  - Smaller ensemble sizes
- Weaknesses
  - Underperforms in simpler/smaller datasets
  - More parameters must be optimized

Experiments

Remember: there is no free lunch!

Experiments

Algorithms

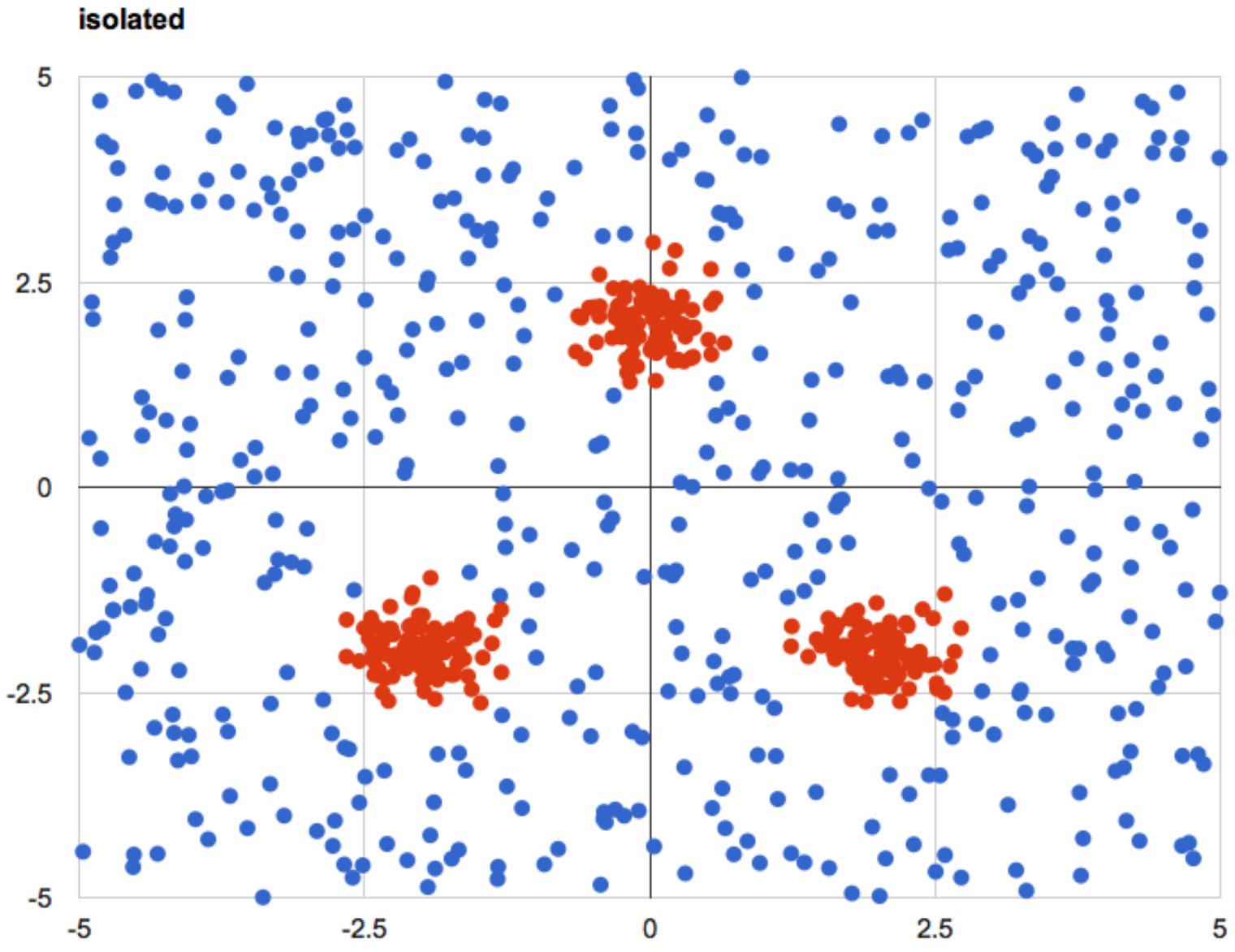
- C4.5 (decision tree)
- AdaBoosted C4.5
- Linear SVM (LibSVM)
- AdaBoosted Linear SVM
- Polynomial SVM
- Multilayer Perceptron

Datasets

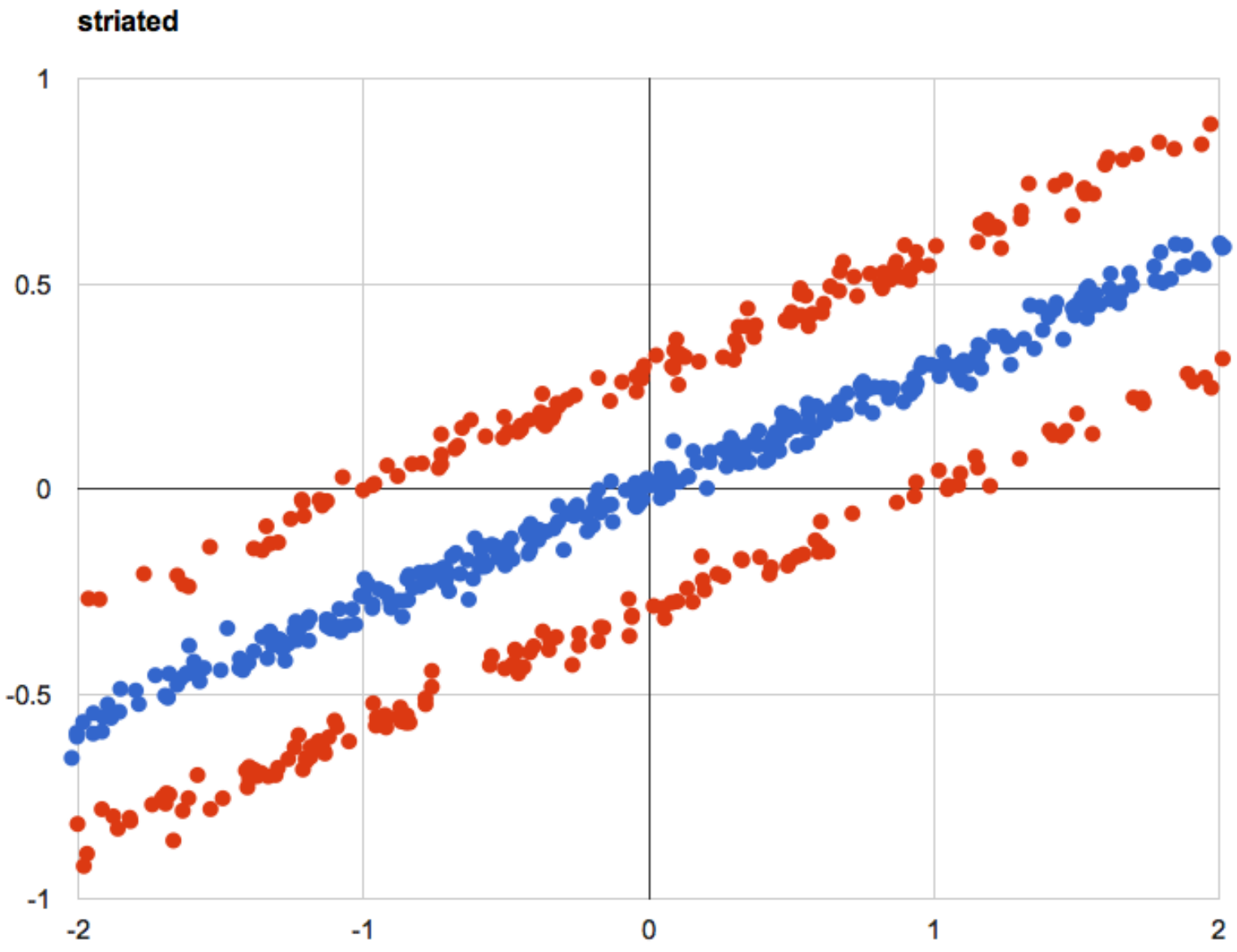
- Synthetic
- UCI benchmarks
- Twitter

(Results averaged over 10-fold cross validation, using Weka's implementations and default parameters)

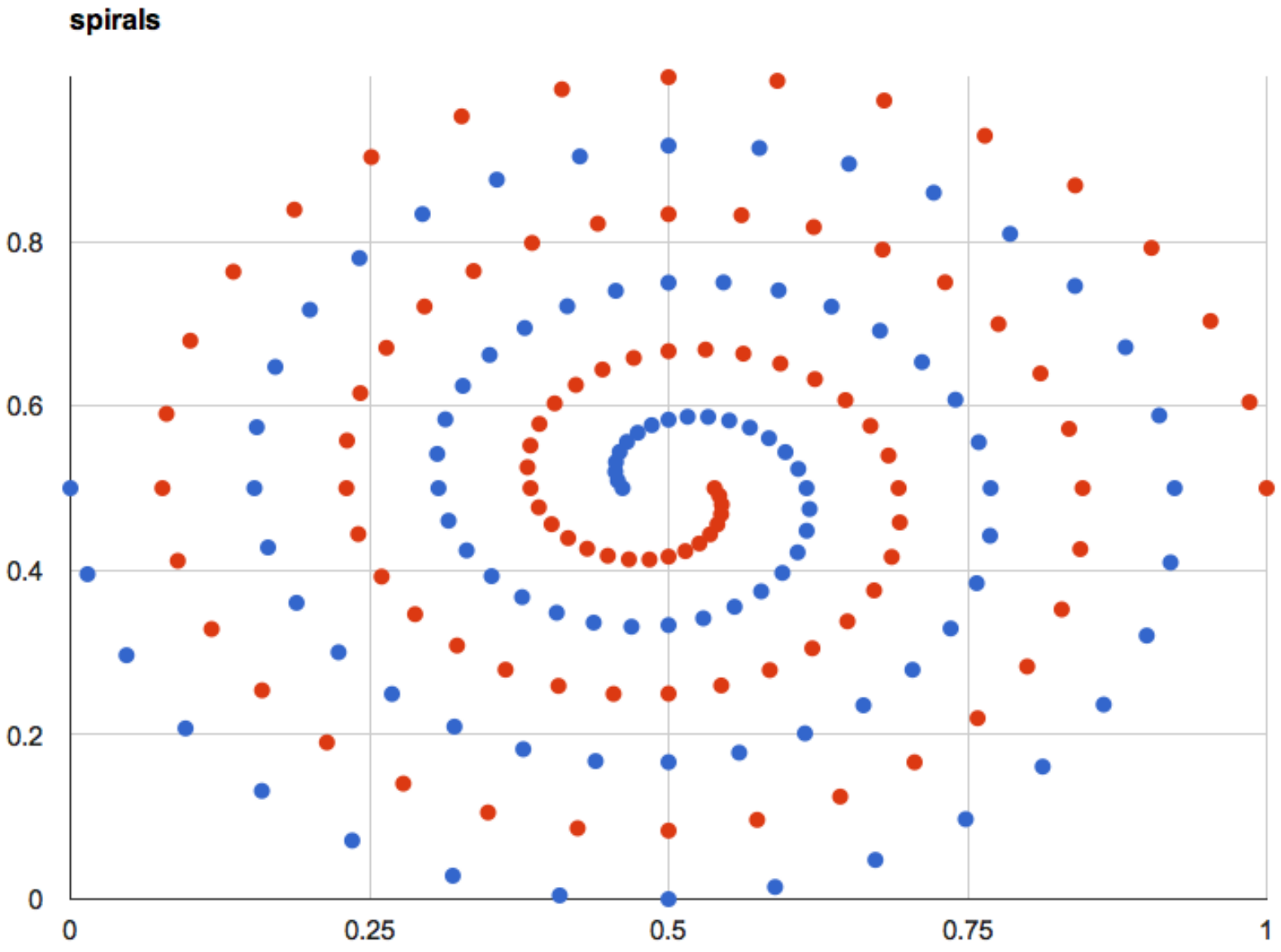
Synthetic Datasets



Synthetic Datasets



Synthetic Datasets



Results

Dataset	PSVM	C4.5	Boosted C4.5	Linear SVM
isolated	90.9 ± 4.07	98.1 ± 1.79 ●	98.5 ± 1.46 ●	62.5 ± 0.0 ○
striated	97.3 ± 1.66	69.4 ± 13.4 ○	90.1 ± 7.38 ○	48.5 ± 3.94 ○
spirals	21.8 ± 18.9	0.0 ± 0.0 ○	0.0 ± 0.0 ○	0.0 ± 0.0 ○
iris	96.0 ± 4.42	94.0 ± 6.29	94.0 ± 5.54	98.7 ± 2.67
glass	52.8 ± 8.72	69.1 ± 6.40 ●	72.9 ± 7.85 ●	63.5 ± 8.08 ●
vehicle	79.4 ± 4.49	73.8 ± 4.48 ○	75.7 ± 3.56	80.4 ± 4.50
segment	95.2 ± 1.18	97.1 ± 0.93 ●	98.1 ± 0.85 ●	96.3 ± 0.93 ●
twitter	55.8 ± 5.12	54.5 ± 2.89	60.5 ± 7.99	62.2 ± 3.66 ●

Results

Dataset	PSVM	Boosted Linear SVM	Polynomial SVM	Multilayer Perceptron
isolated	90.9 ± 4.07	62.5 ± 0.0 ∘	81.9 ± 4.38 ∘	80.6 ± 2.81 ∘
striated	97.3 ± 1.66	53.4 ± 16.3 ∘	72.8 ± 4.18 ∘	75.0 ± 25.0 ∘
spirals	21.8 ± 18.9	5.63 ± 8.97 ∘	0.0 ± 0.0 ∘	20.6 ± 28.9
iris	96.0 ± 4.42	97.3 ± 3.27	96.7 ± 4.47	97.3 ± 4.42
glass	52.8 ± 8.72	63.1 ± 7.73 •	69.7 ± 6.55 •	70.6 ± 8.82 •
vehicle	79.4 ± 4.49	80.4 ± 4.23	80.4 ± 4.53	79.7 ± 4.61
segment	95.2 ± 1.18	96.1 ± 0.89	95.8 ± 1.38	96.2 ± 1.30
twitter	55.8 ± 5.12	62.2 ± 4.15 •	52.3 ± 5.54	52.3 ± 5.54

Results

Dataset	Sampling	Regular
isolated	84.0 ± 7.0	90.9 ± 4.07 •
striated	96.9 ± 3.84	97.3 ± 1.66
spirals	60.7 ± 34.9	21.8 ± 18.9 ∘
iris	86.7 ± 13.0	96.0 ± 4.42
glass	47.6 ± 12.9	52.8 ± 8.72
vehicle	75.4 ± 3.83	79.4 ± 4.49
segment	94.4 ± 1.86	95.2 ± 1.18
twitter	54.0 ± 8.86	55.8 ± 5.12

Noise Experiments

- Synthetic and benchmarks are unrealistically clean
- Inject 10% noise in class labels

Results: Noisy Data

Dataset	PSVM	C4.5	Boosted C4.5	Linear SVM
isolated	71.9 ± 7.71	84.8 ± 4.36 •	83.6 ± 4.65 •	59.3 ± 3.67 ◦
striated	76.8 ± 7.42	60.3 ± 7.24 ◦	64.3 ± 6.55 ◦	52.0 ± 2.86 ◦
spirals	29.0 ± 15.4	9.79 ± 6.28 ◦	9.79 ± 6.28 ◦	11.9 ± 10.2 ◦
iris	88.7 ± 5.21	84.7 ± 7.33	83.3 ± 7.45	90.0 ± 6.83
glass	50.0 ± 5.50	62.2 ± 7.46 •	66.3 ± 8.87 •	53.7 ± 8.91
vehicle	68.1 ± 4.43	63.1 ± 1.89 ◦	67.6 ± 2.54	71.5 ± 3.57
segment	84.3 ± 2.18	85.5 ± 2.30	85.4 ± 2.23	84.1 ± 1.82

Results: Noisy Data

Dataset	PSVM	Boosted Linear SVM	Polynomial SVM	Multilayer Perceptron
isolated	71.9 ± 7.71	57.9 ± 3.71 ○	75.5 ± 4.91	73.9 ± 3.51
striated	76.8 ± 7.42	56.8 ± 7.36 ○	66.8 ± 3.12 ○	67.4 ± 18.5
spirals	29.0 ± 15.4	16.9 ± 17.1	10.3 ± 6.95 ○	17.1 ± 13.4
iris	88.7 ± 5.21	90.0 ± 6.15	90.0 ± 5.37	91.3 ± 6.70
glass	50.0 ± 5.50	57.5 ± 7.90 ●	61.6 ± 8.95 ●	61.7 ± 4.14 ●
vehicle	68.1 ± 4.43	69.1 ± 4.24	57.4 ± 5.12 ○	71.0 ± 2.61
segment	84.3 ± 2.18	84.0 ± 2.12	67.8 ± 3.95 ○	85.4 ± 2.51

Results: Noisy Data

Dataset	Sampling	Regular
isolated	68.8 ± 5.81	71.9 ± 7.71
striated	75.1 ± 10.1	76.8 ± 7.42
spirals	57.8 ± 28.6	29.0 ± 15.4 $\circ$
iris	76.7 ± 10.4	88.7 ± 5.21 $\bullet$
glass	43.4 ± 9.40	50.0 ± 5.50 $\bullet$
vehicle	59.1 ± 5.69	68.1 ± 4.43
segment	80.6 ± 3.15	84.3 ± 2.18 $\bullet$

Final Ensembles

		Noiseless		Noisy	
	ESVM	Regular	Sampling	Regular	Sampling
isolated	720	323	124	122	70
striated	720	149	88	170	93
spirals	175	47	29	56	20
iris	135	54	13	24	11
glass	193	35	15	37	20
vehicle	761	110	61	113	64
segment	2079	453	172	323	164
twitter	540	42	32	X	X

Summary of Results

- Win-Lose-Tie counts:

	Noiseless	Noisy
Regular	16 - 13 - 19	14 - 7 - 21
Sampling	14 - 23 - 11	13 - 23 - 6

- Performs very well in most difficult datasets
- Performs as well as other algorithms in all datasets
- Degrades gracefully with noise
- Standard PSVM outperforms version with sampling

Future Work

- Further empirical studies
  - More datasets, especially real-world data
  - More thorough comparison in noisy datasets
  - How dataset complexity affects ensemble size
- Algorithm improvements
  - Feature selection for each model (e.g. 1-norm SVMs [4])
  - Different metrics besides Euclidean distance
- Prove can approximate functions to within certain error
  - Given a minimum size of training set?
  - Within a certain number of shifting iterations?
- Explicitly incorporate clustering

References


Corinna Cortes and Vladimir Vapnik.
Support-vector networks.
Machine Learning, 20(3), 1995.

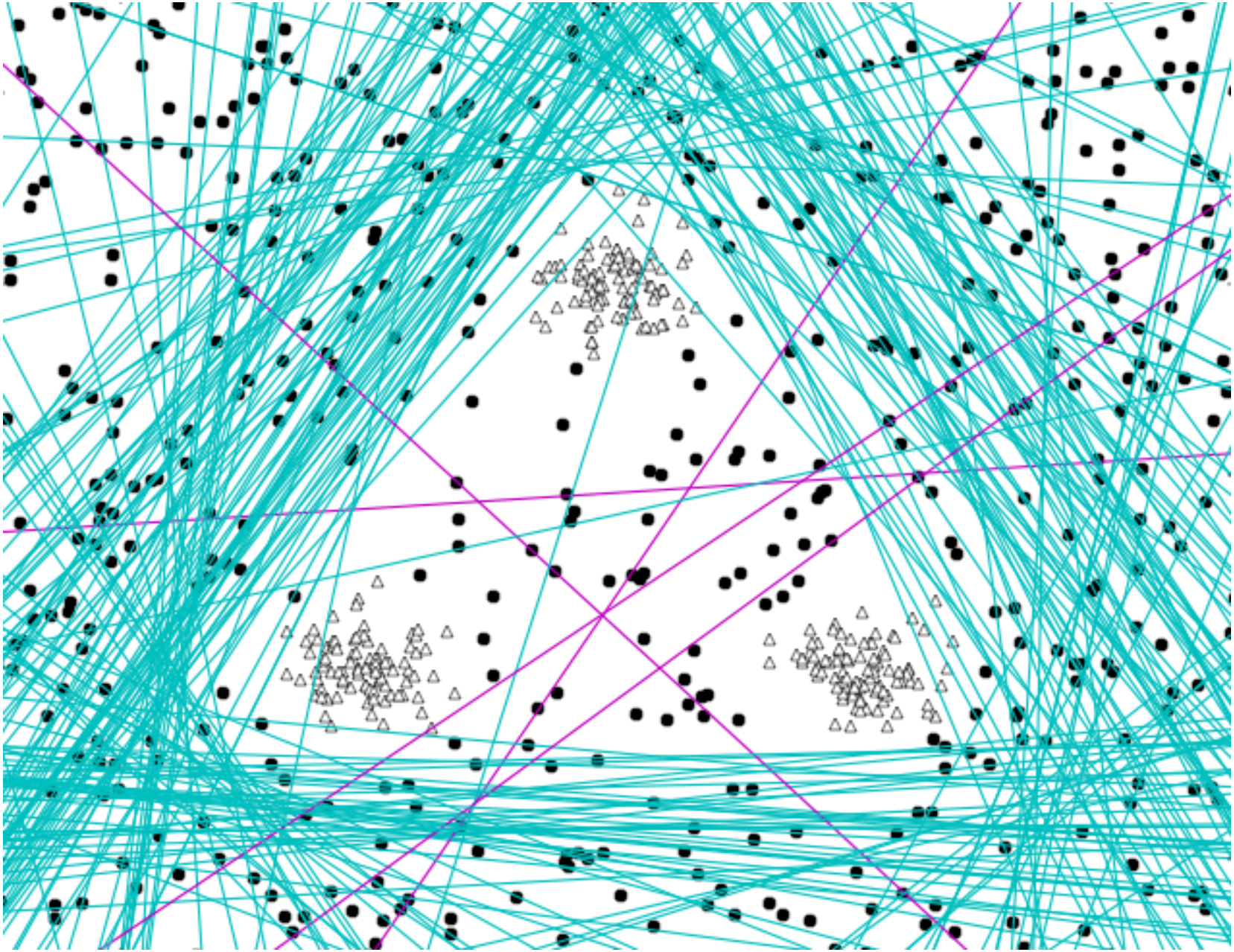

Tomasz Malisiewicz, Abhinav Gupta, and Alexei A. Efros.
Ensemble of exemplar-SVMs for object detection and beyond.
In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, 2011.


Robert E. Schapire.
A brief introduction to boosting.
In *Proceedings of the 1999 International Joint Conference on Artificial Intelligence*, IJCAI '99, 1999.

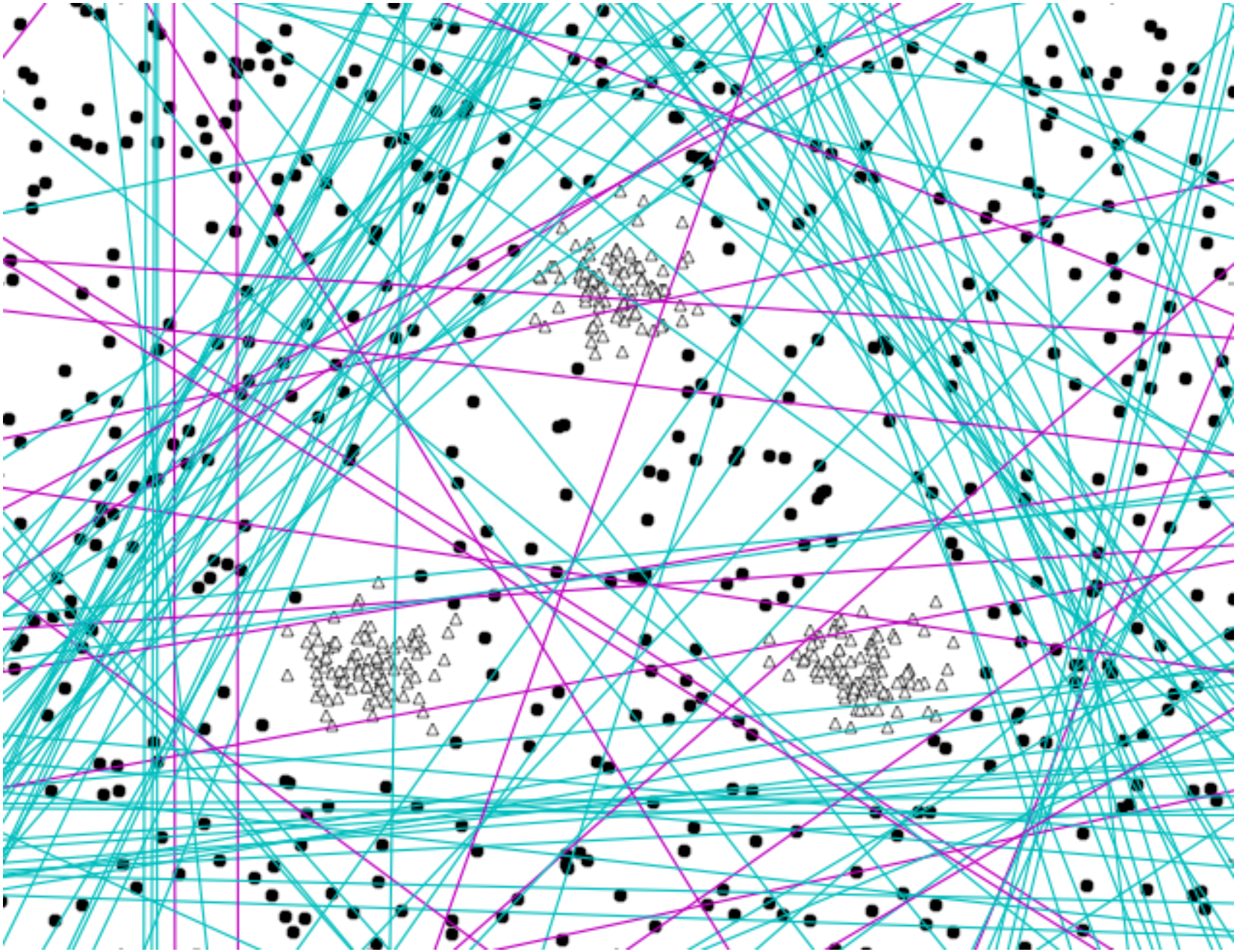

J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani.
1-norm support vector machines.
In *Advances in Neural Information Processing Systems 16*, 2004.

BONUS MATERIAL

Final Ensembles



Final Ensembles



Train

Input: set of data, T
Parameters: number of iterations, s

Split data into training and validation sets, D and V.
$P \leftarrow [[d_i] \text{ for } d_i \in D]$
$N \leftarrow [\textsc{ChooseNegatives}(D, d_i) \text{ for } d_i \in D]$
for $j = 0, \ldots, s$ **do**
    $E_j \leftarrow [\,]$
    for $P_i \in P$ and $N_i \in N$ **do**
        Train a linear SVM, using P_i and N_i.
        Add this SVM to E_j.
    $a_j \leftarrow \textsc{Test}(V, E_j)$
    $P, N \leftarrow \textsc{Shift}(D, E_j, P, N)$
return ensemble E_j with highest accuracy on V

ChooseNegatives

Input: set of training data, D, and training instance, d_i
Parameters: number of negatives to return, k

$N_i \leftarrow [\,]$
$D_i \leftarrow$ all instances in D of a different class label from d_i
Compute Euclidean distance from d_i to each element of D_i.
Sort D_i in ascending order by distance.
$x \leftarrow$ closest negative in D_i
$\vec{n} \leftarrow (x - d_i)/||x - d_i||$
for $d_j \in D_i$ **do**
    if $\vec{n} \cdot (d_j - d_i) > 0$ **then**
        Add d_j to N_i
        if $|N_i| = k$ **then**
            return N_i

Shift

Input: set of training data, D; ensemble of models, E;
positive and negative sets for each model, P and N
Parameters: probability to add to negative set, p

$C \leftarrow$ [[ ] for $d_j \in D$] (candidate models)
for $m_i \in E$ and $d_j \in D$ **do**
  if m_i classifies d_j positively **then**
    if d_j's class matches m_i's class **then**
      Compute the distance of d_j to m_i's exemplar.
      Add m_i and its distance to the list of candidates C_j.
    else
      Add d_j to m_i's negative set N_i with some probability p.
for $d_j \in D$ **do**
  Add d_j to the positive set P_k of closest candidate model m_k
for $m_i \in E$ **do**
  if m_i did not classify anything positively **then**
    Remove m_i from the ensemble
return P, N

Test

Input: set of testing data, D, and ensemble of models, E

1: **for** $m_i \in E$ and $d_j \in D$ **do**
2: **if** m_i classifies d_j positively **then**
3: Record weighted vote for m_i's class.
4: **for** $d_j \in D$ **do**
5: Output class with max votes as prediction for d_j.
6: $a \leftarrow$ classification accuracy over all data in D.
7: **return** a

TrainWithSampling

Input: set of training data, D
Parameters: number of iterations, r

1: Initialize all weights to $1/n$.
2: $S \leftarrow \textsc{Sample}(D)$
3: **for** r iterations **do**
4: $\quad E \leftarrow \textsc{Train}(S)$
5: $\quad a \leftarrow \textsc{Test}(D, E)$
6: $\quad \textsc{Reweight}(D, a)$
7: $\quad S \leftarrow S \cup \textsc{Sample}(D)$
8: **return** E

Reweight

Input: set of training data, D, and accuracy of the ensemble, a

1: $\varepsilon = 1 - a$
2: $\alpha = \frac{1}{2} \ln \left(\frac{1-\varepsilon}{\varepsilon} \right)$
3: **for** $d_i \in D$ **do**
4: **if** d_i was correctly classified **then**
5: $weight(d_i) = 0$
6: **else**
7: $weight(d_i) = \frac{weight(d_i) \times e^{\alpha}}{z}$ ($z =$ normalization factor)