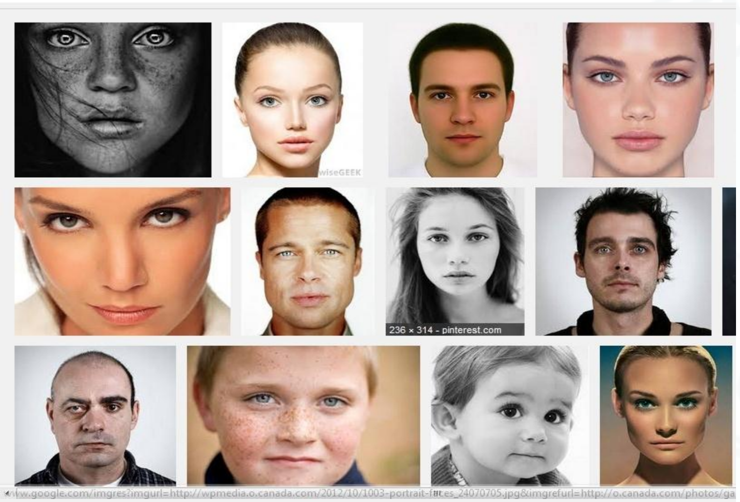
Multiple Instance Learning

Dr. Fayyaz ul Amir Afsar Minhas
PIEAS Biomedical Informatics Research Lab
Department of Computer and Information Sciences
Pakistan Institute of Engineering & Applied Sciences
PO Nilore, Islamabad, Pakistan
http://faculty.pieas.edu.pk/fayyaz/

Binary Classification



Faces (+1)

+1

-1

Non Faces (-1)

-1

+1

Labeling Ambiguities

- What if the labels are **ambiguous**?
  - Part-Whole Ambiguity
  - Polymorphism Ambiguity

Part-Whole Ambiguity

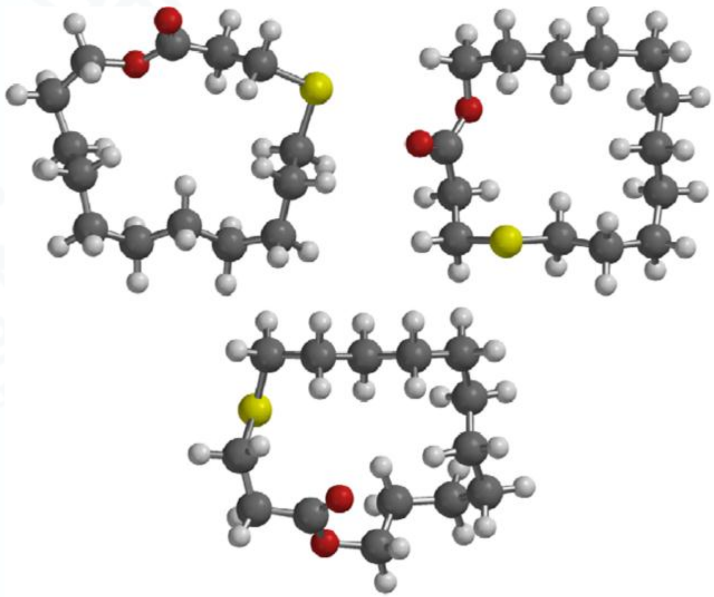
- Label of the object depends on one or more **<u>PARTS</u>** instead of **<u>WHOLE</u>** of the object
- Face Detection Problem

+1

Polymorphism Ambiguity

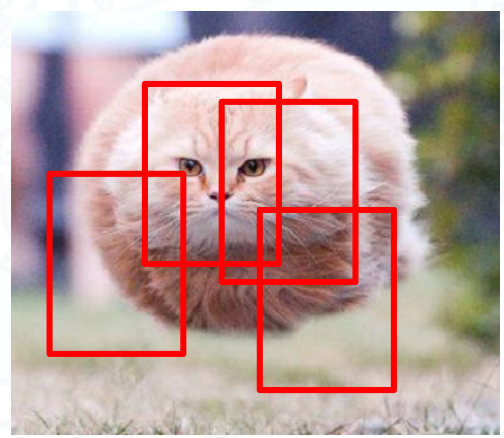
- Label of the object depends on one or more **PARTICULAR FORMS** of all

- Drug Activity Prediction



Multiple Instance Learning

- Allows to learn a classifier when such ambiguities are present
- Use group (bag) level labels instead of instances



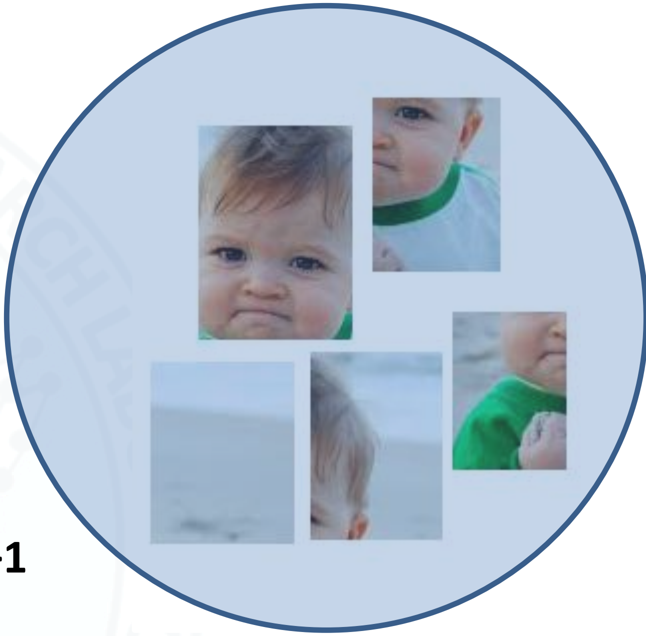
-1

Problem Formulation of MIL- Bags

- A bag is a set of input patterns
- With each bag B_l is associated a label Y_l
  - If $Y_l = -1$, then $y_i = -1$ for all $i \in I$
  - If $Y_l = +1$, then at least one pattern $x \in B_l$ is a positive example

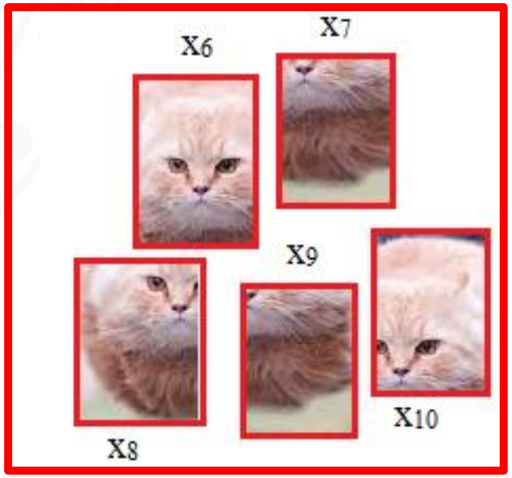
B_1

Y_1=+1



B_2

Y_1=-1



Problem Formulation of MIL

Find a discriminant function f *parameterized by* w *which can generate output values of any given training bag* B_I

$$Y_I = f(\boldsymbol{B_I}; \boldsymbol{w}) \quad \forall I$$

such that for classification the following constraints hold,

$$\sum_{i \in I} \frac{y_i + 1}{2} \geq 1, \qquad \forall I \; s.t \;\; Y_I = 1, \qquad and \qquad (1)$$

$$y_i = -1 \qquad \forall I \; s.t. \; Y_I = -1 \qquad (2)$$

where, y_i *is the label for instance* $\boldsymbol{x}_i$ *and* $\boldsymbol{x}_i \in \boldsymbol{B_I}$.

Multiple Instance Learning – The Goal

- To classify the unseen
  - bags
  - instances

Applications

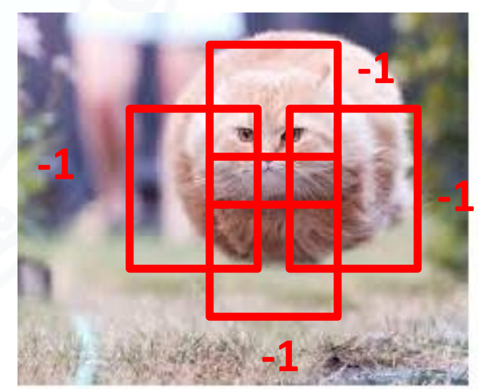
- Drug Activity Prediction
- Protein Interactions
- Computer Aided Diagnostics
- Visual Tracking
- Content-based image retrieval and classification
- Text categorization
- Natural Scene Classification

Naïve Solution to MIL Problem

- Assign the label of the "whole" to the "parts"
- Assign all the appearances the same label
- Use any classification technique
- Classification Accuracy Compromised!

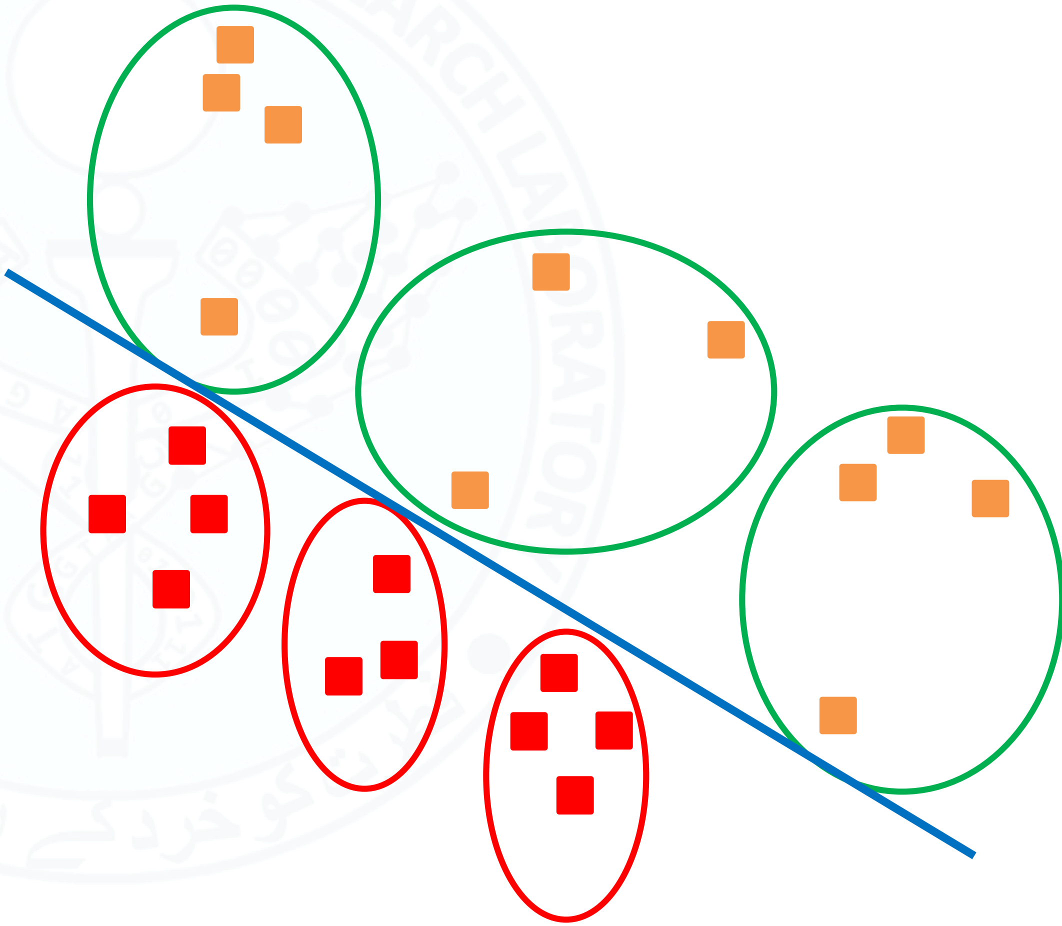
+1

-1



Properties of the Naïve Solution

- Labeling noise introduced in the data
- low margin and poor generalization
- Does not model the relationship between examples in a bag
  - Assumes that examples are independently labeled



Classifiers for MIL

- Learning Axis-Parallel Concepts
- Logistic Regression
- Diverse Density (DD) and its EM version
- Boosting
- Citation kNN
- Support Vector Machines

An SRM approach to MIL

- Y_I is the label for the bag $B_I, I = 1 \ldots N$
- The labels of the individual examples $\boldsymbol{x}$ in a bag are unknown
- We want labels of these examples such that
  – All positive bags are classified as positive
    - Equivalently: At least one example in all of the positive bags is labeled as positive
  – All negative bags are classified as negative
    - Equivalently: All negative examples in all negative bags are labeled negative
  – Margin is maximized

Bag Level Classification

- Let's assume a bag level classifier such that it produces a prediction score $F(B_I; \boldsymbol{w})$ given the bag B_I and the parameters $\boldsymbol{w}$
  - Thus, we want to satisfy the following constraints
    - All positive bags are classified positive
    - All negative bags are classified negative
  - This implies: $Y_I F(B_I; \boldsymbol{w}) > 0$ for all bags

Instance Level Classification

- Let's assume an instance level classifier such that it produces a prediction score $f(\boldsymbol{x}; \boldsymbol{w})$ given the instance $\boldsymbol{x}$ and the parameters $\boldsymbol{w}$
  - Thus, we want to satisfy the following constraints
    - At least one example in every bag is classified as positive
      - The score of the highest scoring example in a positive bag should be positive
      - This implies: $max_{\boldsymbol{x} \in B_I} f(\boldsymbol{x}; \boldsymbol{w}) > 0$ for all positive bags B_I
    - All negative examples in all negative bags are classified negative
      - The score of the highest scoring example in a negative bag should be negative
      - This implies: $max_{\boldsymbol{x} \in B_I} f(\boldsymbol{x}; \boldsymbol{w}) < 0$ for all negative bags B_I
    - OR: $Y_I max_{\boldsymbol{x} \in B_I} f(\boldsymbol{x}; \boldsymbol{w}) > 0$ for all bags

Relation between instance and bag level classification

- We can clearly see that

$$F(B_I; \boldsymbol{w}) = max_{\boldsymbol{x} \in B_I} f(\boldsymbol{x}; \boldsymbol{w})$$

- We can use a $f(\boldsymbol{x}; \boldsymbol{w}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle$
- To maximize the margin, our constraint becomes:

$$Y_I max_{\boldsymbol{x} \in B_I} f(\boldsymbol{x}; \boldsymbol{w}) \geq 1$$

- Making it soft by adding slacks, we get:

$$Y_I max_{\boldsymbol{x} \in B_I} f(\boldsymbol{x}; \boldsymbol{w}) \geq 1 - \xi_I, \xi_I \geq 0$$

Large Margin MIL

- The complete form of the SVM becomes

$$\max_{\boldsymbol{w},\xi} \frac{\lambda}{2} \boldsymbol{w}^{\boldsymbol{T}} \boldsymbol{w} + \sum_{I=1}^{N} \xi_I$$

Such that for all $I = 1 \ldots N$

$$Y_I max_{\boldsymbol{x} \in B_I} \langle \boldsymbol{w}, \boldsymbol{x} \rangle \geq 1 - \xi_I, \xi_I \geq 0$$

An alternate view

- A bag is misclassified if $Y_I max_{\boldsymbol{x} \in B_I} \langle \boldsymbol{w}, \boldsymbol{x} \rangle < 0$
- Thus the error function becomes

$$e(B_I; \boldsymbol{w}) = 1\left(- Y_I max_{\boldsymbol{x} \in B_I} \langle \boldsymbol{w}, \boldsymbol{x} \rangle\right)$$

- The convex over-approximation to this is the hinge loss function given by:

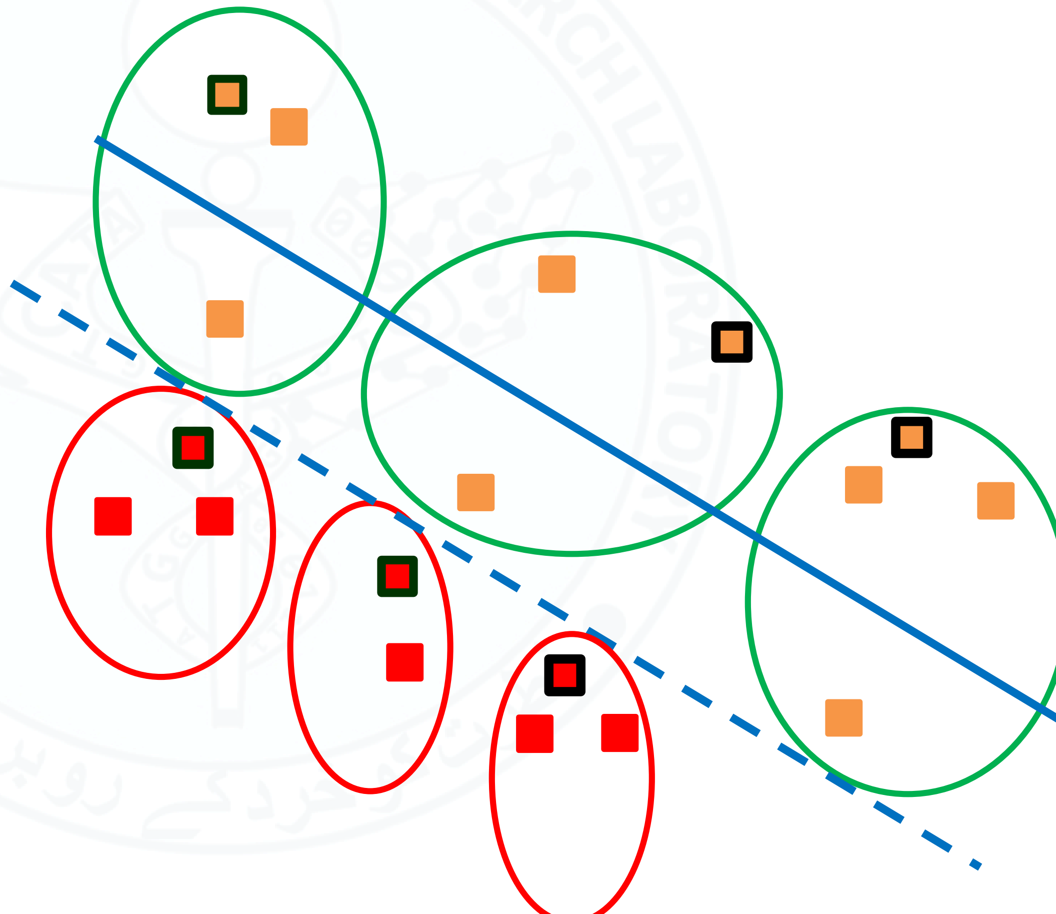
$$l(B_I; \boldsymbol{w}) = \max\{0,1 - Y_I\, max_{\boldsymbol{x} \in B_I} \langle \boldsymbol{w}, \boldsymbol{x} \rangle\}$$

- And the classifier can be written as:

$$\max_{\boldsymbol{w}} \frac{\lambda}{2} \boldsymbol{w}^{\boldsymbol{T}} \boldsymbol{w} + \sum_{I=1}^{N} \max\{0,1 - Y_I\, max_{\boldsymbol{x} \in B_I} \langle \boldsymbol{w}, \boldsymbol{x} \rangle\}$$

What would this do?

- The margin is determined by
  - Highest scoring examples in bags and not all examples



Issues

- The issues with this large margin MIL formulation is that it is non-linear due to the max operator in the hinge-loss function
- Solution:
  - Use Stochastic Subgradient Descent
    - Pick a bag B_I at random
    - Find its most positive example $max_{\boldsymbol{x} \in B_I} \langle \boldsymbol{w}, \boldsymbol{x} \rangle$
    - Calculate the sub-gradient of the objective function for the chosen example with respect to $\boldsymbol{w}$
    - Iterate!
  - Implemented in PyLemmings
    - "pyLemmings: A software suite for Multiple Instance Learning" by A. Asif, Mphil (CS) Thesis, Department of Computer and Information Sciences, Pakistan Institute of Engineering and Applied Sciences, Islamabad, Pakistan, 2015.

Generalized Multiple Instance Ranking

- In multiple instance ranking, we are given instances in bags
  - Ranks are for bags and not for individual instances
- Generalized MIL Instance Ranking
  - We are given pairs of bags $\left(B_i^1, B_i^2\right)$and the difference of their ranks R_i (B_i^1 is ranked higher than B_i^2) for $i = 1 \dots N$
  - Assume we have a bag-level ranking function $F(B; \boldsymbol{w})$, then the error function becomes
    - $e\left(\left(B_i^1, B_i^2\right); \boldsymbol{w}\right) = R_i 1\left(F\left(B_i^1; \boldsymbol{w}\right) < F\left(B_i^2; \boldsymbol{w}\right)\right)$
  - The convex over-approximation of this error function is:
    - $l\left(\left(B_i^1, B_i^2\right); \boldsymbol{w}\right) = \max\{0, R_i - \left(F\left(B_i^1; \boldsymbol{w}\right) - F\left(B_i^2; \boldsymbol{w}\right)\right)\}$
  - But what is $F(B; \boldsymbol{w})$?

Generalized Multiple Instance Ranking

- We assume that the rank of a bag is determined by the highest ranking example in that bag

$$F(B;\boldsymbol{w}) = max_{\boldsymbol{x}\in B} f(\boldsymbol{x};\boldsymbol{w})$$

- We can use a $f(\boldsymbol{x};\boldsymbol{w}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle$
- Thus, the learning problem now becomes:

$$\max_{\boldsymbol{w}} \frac{\lambda}{2}\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{w} + \sum_{I=1}^{N} \max\{0, R_i - \left(max_{\boldsymbol{x}\in B_i^1} f(\boldsymbol{x};\boldsymbol{w}) - max_{\boldsymbol{x}\in B_i^2} f(\boldsymbol{x};\boldsymbol{w})\right)\}$$

- This problem can be solved in a manner similar to PyLemmings Classification

Multiple Instance Regression

The target values are given at the bag level and not for individual instances

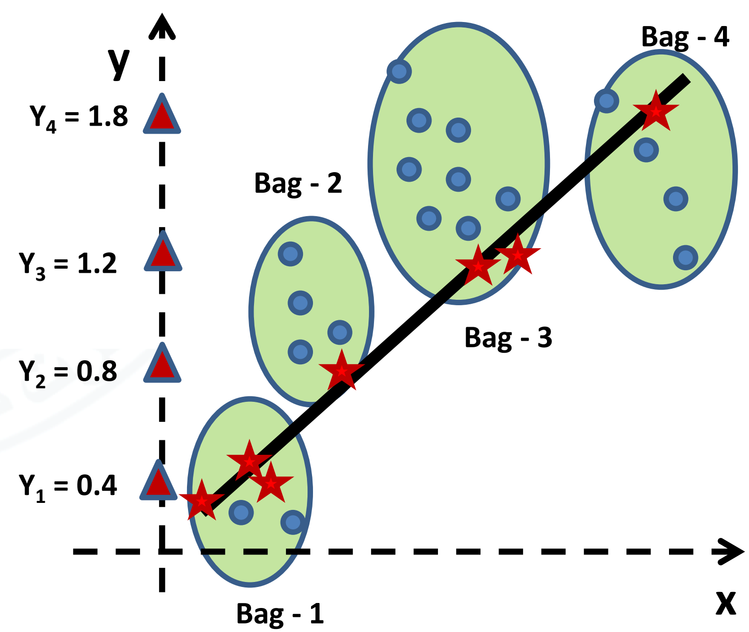
Find a function f parameterized by w which can generate output values of any given training bag B_I

$$Y_I = f(\boldsymbol{B}_I; \boldsymbol{w}) \quad \forall I$$

such that,

$$|Y_I - f(\boldsymbol{B}_I; \boldsymbol{w})| \leq \epsilon$$

Where, $\epsilon \geq 0$ is the maximum error allowed.



Multiple Instance Regression

- The ε-insensitive loss function is
  $l(B;\boldsymbol{w}) = \max\{0, \min_{\boldsymbol{x}\in B} | Y_I - f(\boldsymbol{x};\boldsymbol{w})| - \varepsilon\}$

It can only produce instance level outputs

Nonlinear MIL

- We have, up till now, used a linear discriminant function
- However, we can do nonlinear MIL by
  - Using kernel approximations
  - Locally linear approximations (implemented in PyLemmings!)

pyLEMMINGS

- **PY**thon based **L**arg**E** **M**argin **M**ultiple **I**nstance learni**NG** **S**ystem
  - Object Oriented Implementation for
    - Linear and Locally Linear Multiple Instance Classification, Ranking and Regression
  - Built-in module for
    - K-fold Cross Validation
    - Leave One Out Cross Validation
  - Support for sparse data format
  - Support for Parallelization for Cross Validation module
  - User-friendly implementation
  - Help for users and maintenance

Other Large Margin Solutions for MIL

- Proposed by Andrews et al. (2002)

- Two Formulations
  - Maximum Pattern Margin Formulation (mi-SVM)
    - Instance level Margin Maximization
  - Maximum Bag Margin Formulation (MI-SVM)
    - Bag level Margin Maximization

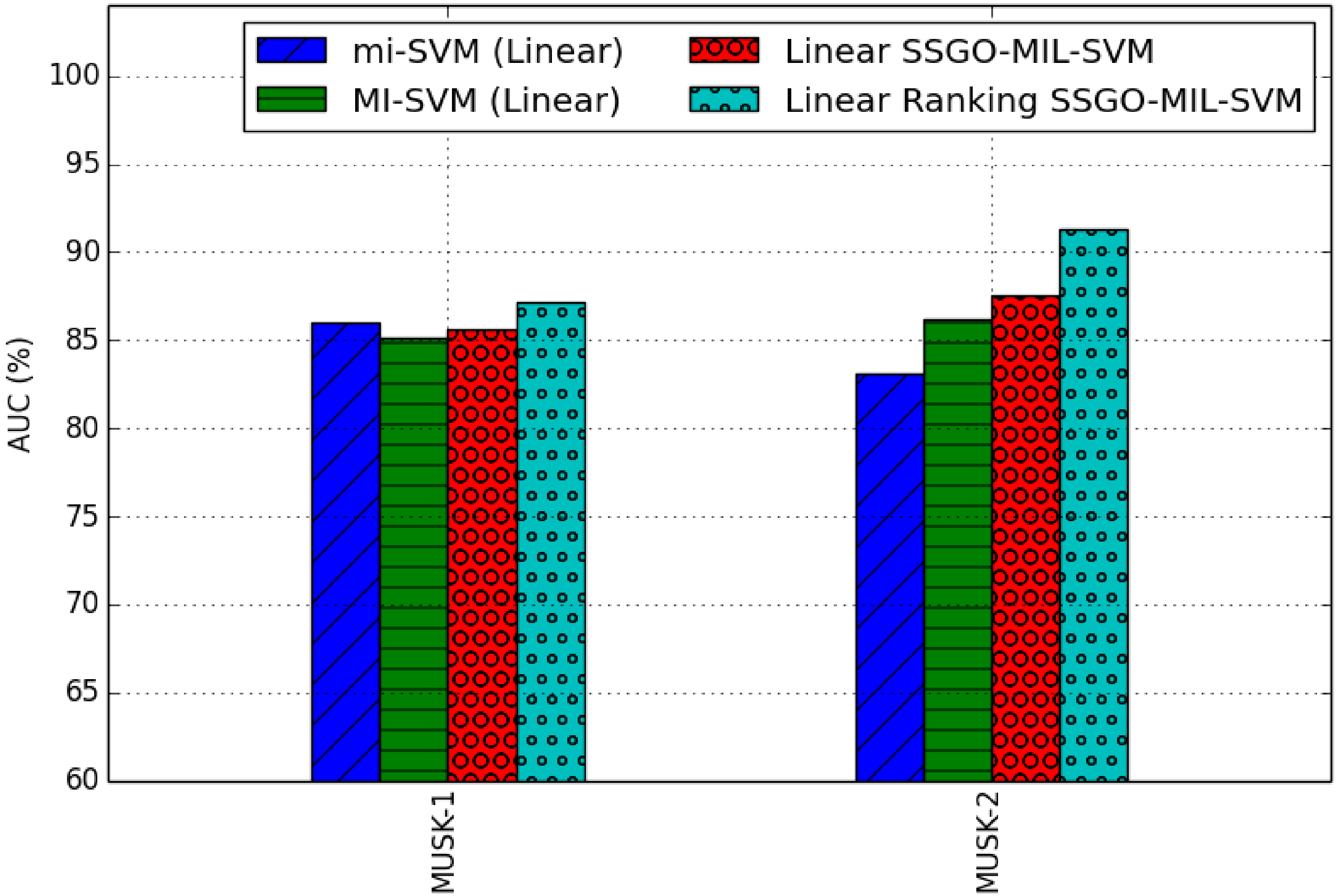
Benchmark Results

- Datasets

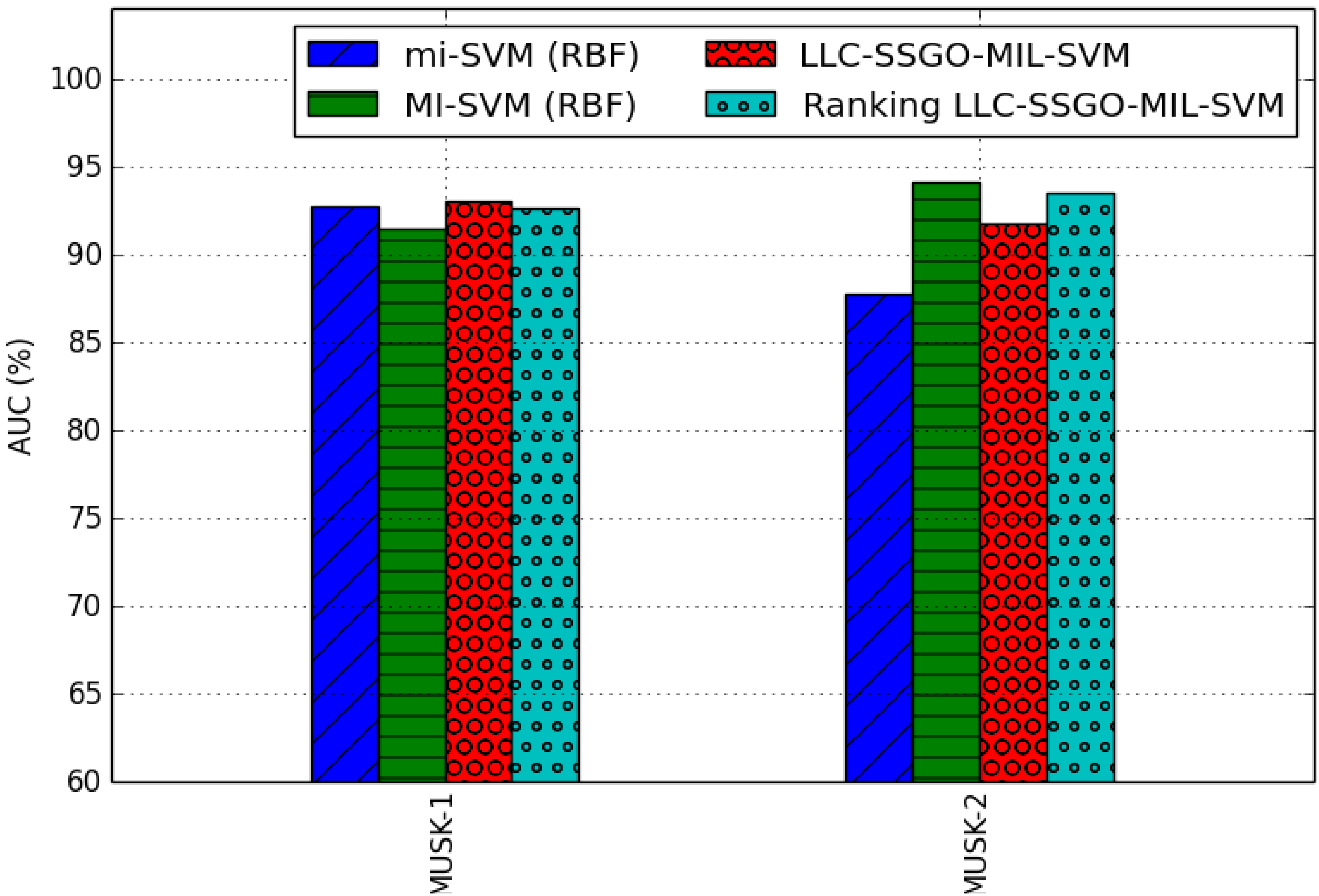
Dataset	Instances	Bags	Positive Bags	Negative Bags	Dimensions
MUSK-1	476	92	47	45	166
MUSK-2	6598	102	39	63	166

- Evaluation Model
  - Ten Fold Cross Validation
- Performance Metrics
  - AUC
  - Running Time

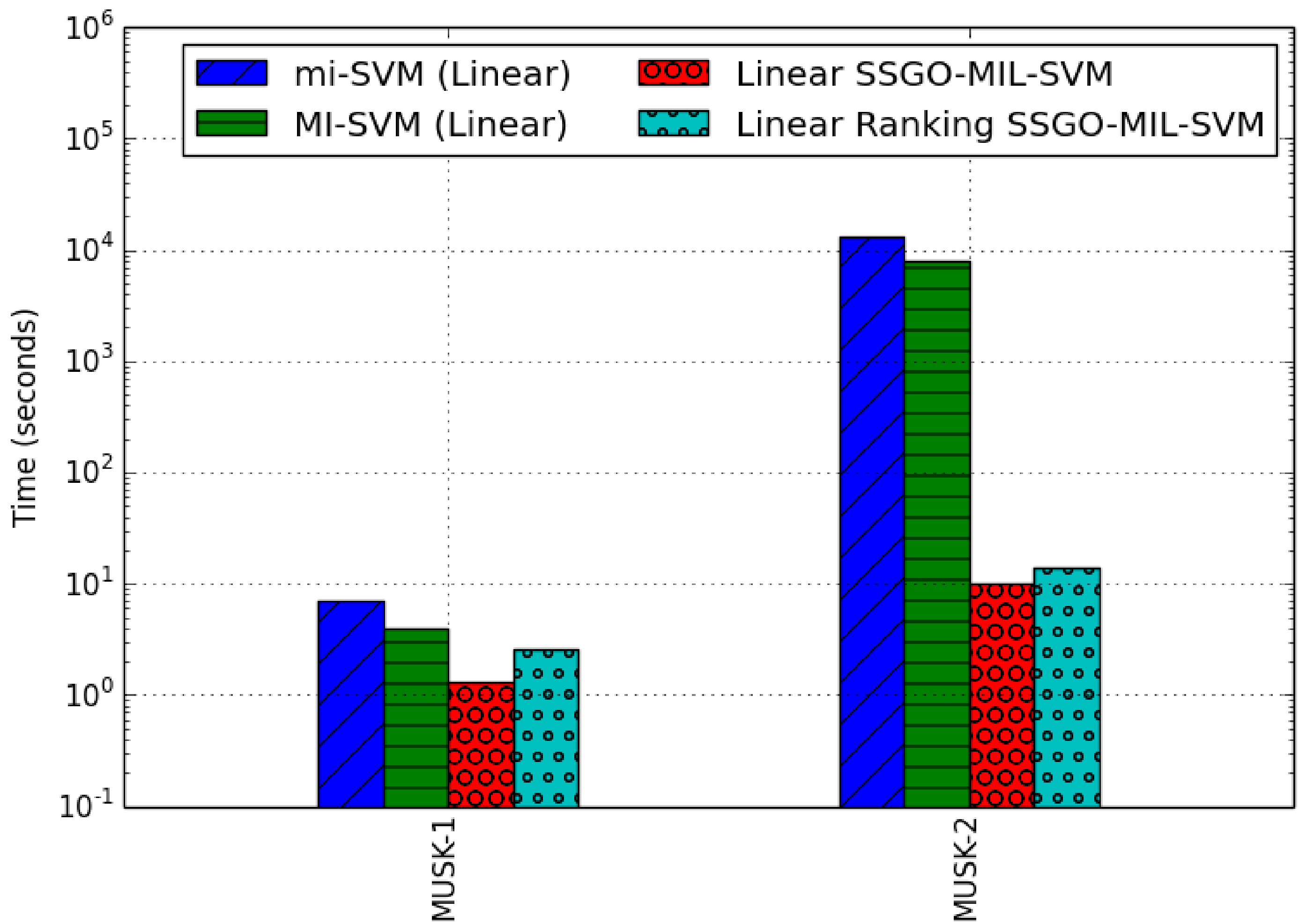
Benchmark Results- AUC



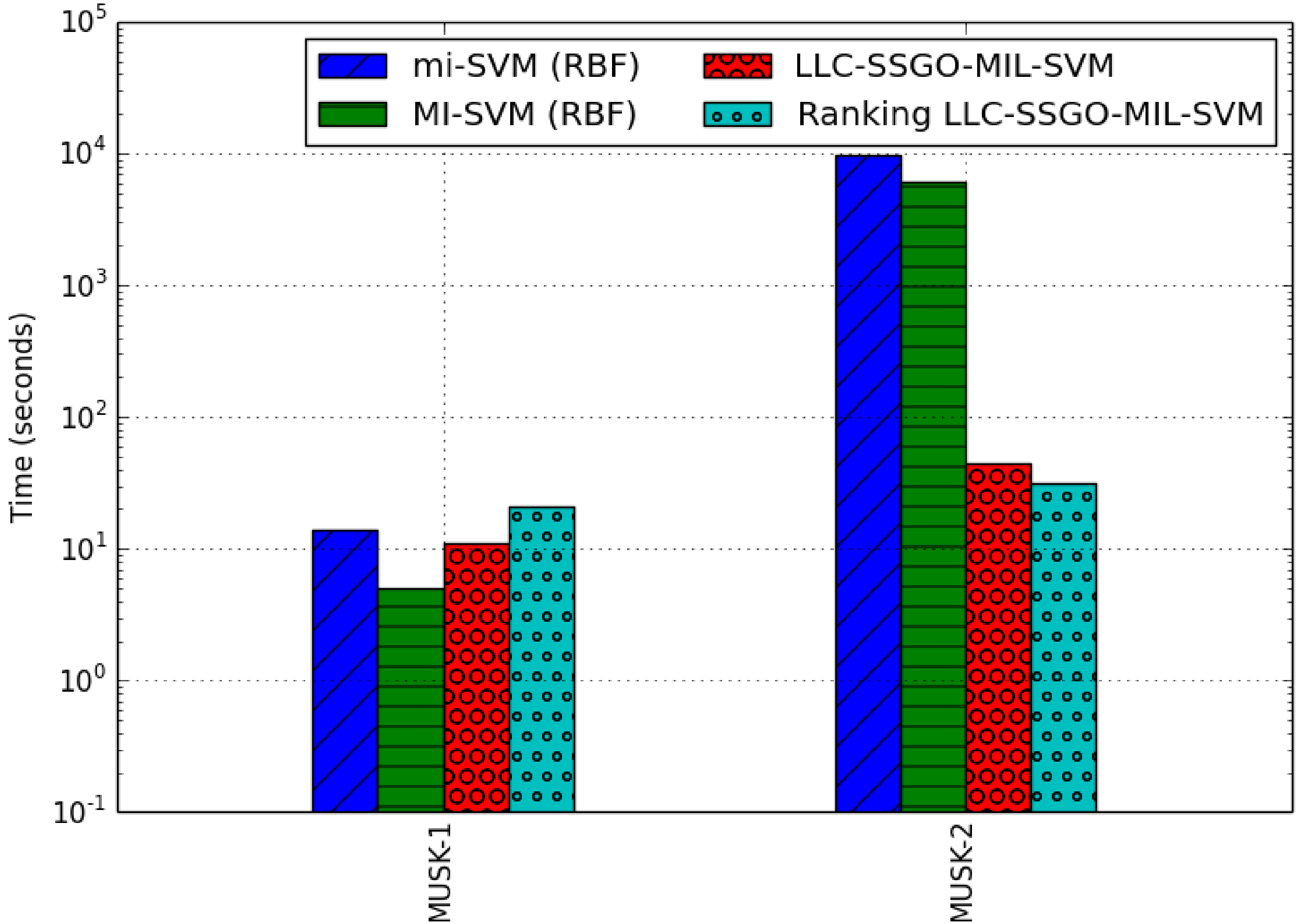
Benchmark Results- AUC



Benchmark Results- Running Time



Benchmark Results- Running Time

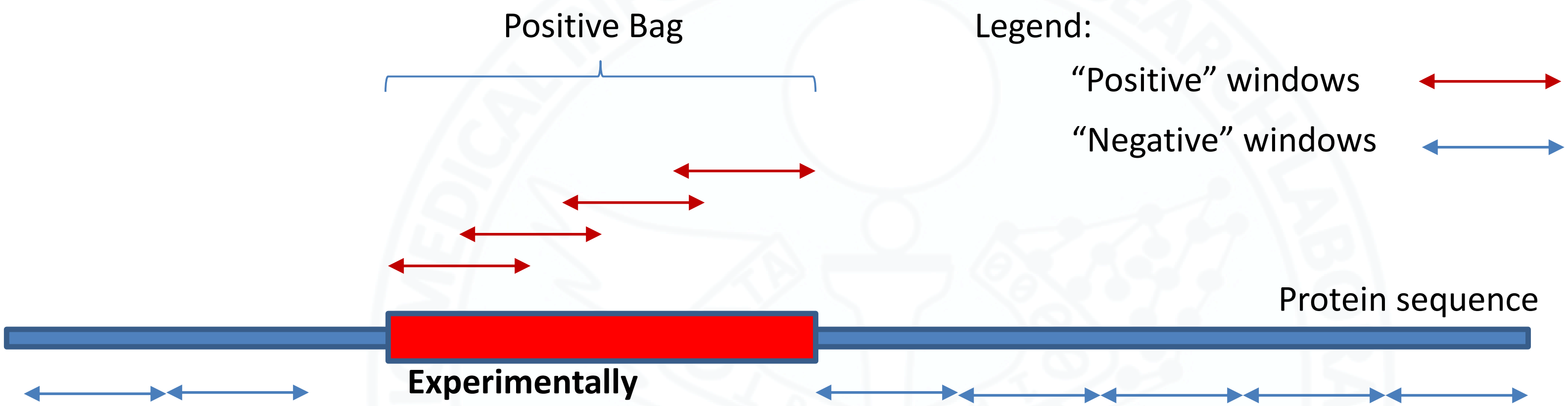


Applications

- Prediction of binding sites in CaM binding proteins
- Prediction of Prion forming domains
- Prediction of Amyloid cores

- In all of these problems, a protein has a certain property and a small part of it is responsible for that property. However that region has not been precisely determined due to time and cost constraints in biological experiments. We want to find those regions in novel proteins.

Concept Diagram



Experimentally determined site of Interest
(CaM binding, amyloid core or prion forming domain). Not all residues in this site are responsible for the protein's function and the site covers an area larger than the minimal set of residues due to experimental limitations. Sequence windows in this region form a positive bag.

All negative instances are grouped into a single negative bag

Evaluation Metrics

- Leave-One-Out Cross Validation
- AUC-ROC: Overall classification accuracy
- AUC-ROC 0.1: AUC-ROC up to the first 10% False positives. How does the classifier perform for high sensitivity and high precision.
- AUC-PR: Imbalanced data
- THR: Percentage of proteins in which the top scoring prediction was correct
- FHR: Percentage of negative windows in proteins in the validation set that scored higher than the top scoring positive window

CaM binding site prediction

- MI-1 and CaMELS
  - Wajid Arshad Abbasi, Amina Asif Shah, Saiqa Andleeb, and Fayyaz ul Amir Afsar Minhas and Asa Ben-Hur

Method	Features	AUC-ROC	AUC-ROC$_{0.1}$	AUC-PR	THR	FHR
CaMELS	PD-Blosum	99.2	79.0	87.0	77	1.0
	AAC+PD-1	98.9	77.6	85.6	75	1.0
	PD-GT	99.04	78.0	85.6	74	1.0
	PD-1	98.4	76.2	84.1	72	2.0
	propy	98.0	74.7	81.2	68	2.0
	AAC	97.9	72.3	80.7	68	2.0
MI-1	AAC+PD-1	96.9	59.0	--	75	1.2
mi-SVM	AAC+PD-1	96.2	55.6	--	68	1.9
SVM	AAC+PD-1	95.9	55.1	--	65	2.1

CaMELS

- Manuscript under preparation

(http://faculty.pieas.edu.pk/fayyaz/bmi.html)

CaMELS: *In silico* Prediction of Calmodulin Interactions

What is CaMELS?

We present a set of novel algorithms, called CaMELS (CalModulin intEraction Learning System) for predicting both the binding sites and the possibility of interactions of proteins with Calmodulin (CaM) using sequence information alone. The proposed algorithms give state of the art classification results and are available as a cloud based webserver. You can use it to make predictions for proteins of your interest.

Citation details:

If you use CaMELS please cite:Wajid Arshad Abbasi (http://faculty.pieas.edu.pk/fayyaz/wajid/index.html) and Fayyaz-ul-Amir Afsar Minhas (http://faculty.pieas.edu.pk/fayyaz/index.html) (2016), "CaMELS: *In silico* Prediction of Calmodulin Binding Proteins and their Binding Sites", DCIS, PIEAS, (preprint release).

Select fasta file or paste the required sequence

(Sequence length>21 required)

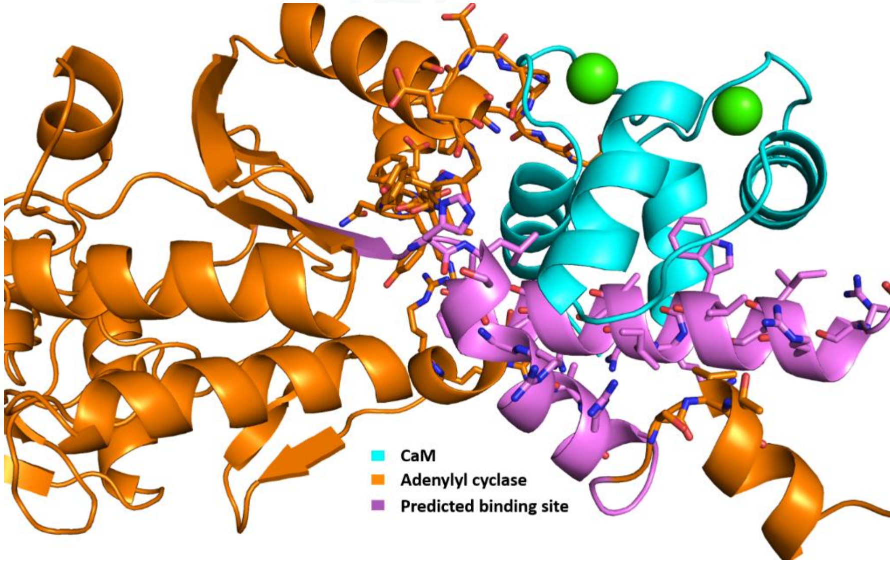
Prediction type: Interaction Prediction ▾

Input file in fasta format (containing only one protein): Choose File No file chosen

(or) Paste your protein sequence in plain text:

SUBMIT

http://faculty.pieas.edu.pk/fayyaz/software.html#camels



Prion Domain Localization

- pRANK using PyLemmings-style MIL
  - 100% correct domain localization
  - Can predict prion domains and prion forming proteins using their amino acid composition alone
  - More accurate than other existing methods

Classifier	AUC-ROC	AUC-PR
pRANK	**96.8**	**96.8**
miSVM	92.2	90.4
SVM	87.4	87.8
Random Forests	88.0	90.6
PAPA	95.1	96.8
PrionW	86.7	89.8
PLAAC	68.7	74.7

https://doi.org/10.1371/journal.pcbi.1005465.t002

Minhas FuA, Ross ED, Ben-Hur A (2017) Amino acid composition predicts prion activity. PLoS Comput Biol 13(4): e1005465. https://doi.org/10.1371/journal.pcbi.1005465

MILIAMP: Amyloid Detection

- Prediction of amyloid forming proteins
  - Whether a protein will form amyloids or not
- Prediction of hotspots in amyloid proteins
  - What part of the protein is the primary contributor to amyloid formation
- Prediction of effects of mutations on amyloid formation propensity
  - How will mutations affect amyloid formation
- Uses amino acid composition alone

SVM for MIL

- Proposed by Andrews et al. (2002)

- Two Formulations
  - Maximum Pattern Margin Formulation (mi-SVM)
    - Instance level Margin Maximization
  - Maximum Bag Margin Formulation (MI-SVM)
    - Bag level Margin Maximization

mi-SVM

- Find the optimal labels of the examples in the positive training bags

$$\min_{\{y_i\}} \min_{\boldsymbol{w},\xi} \frac{1}{2} \|\boldsymbol{w}\|^2 + C \sum_i \xi_i$$

$s.t. \quad \forall i:$

$$y_i(\langle \boldsymbol{w}, \boldsymbol{x_i} \rangle + b) \geq 1 - \xi_i,$$

$$\xi_i \geq 0,$$

$$y_i \in \{-1, 1\} \quad and \quad (1,2) \; hold$$

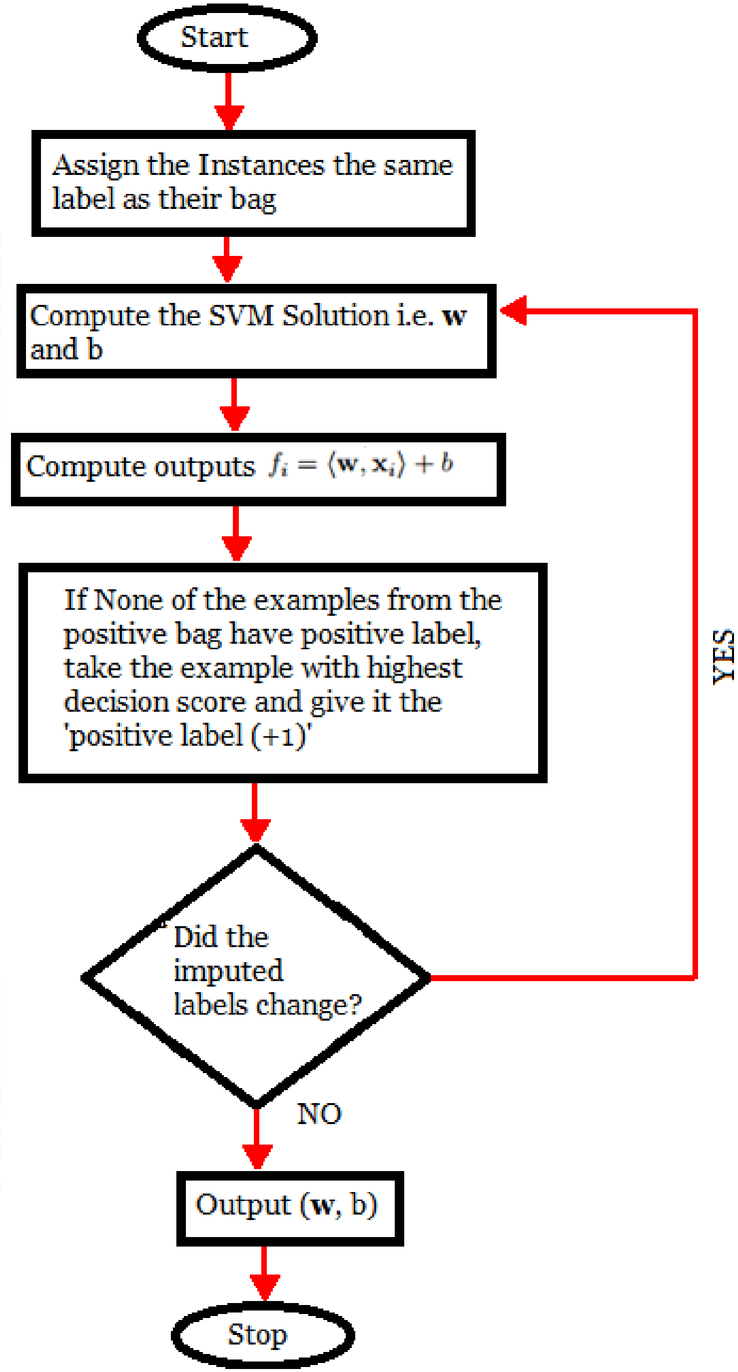
$$\sum_{i \in I} \frac{y_i + 1}{2} \geq 1, \qquad \forall\, I \; s.t \;\; Y_I = 1, \qquad and \qquad (1)$$

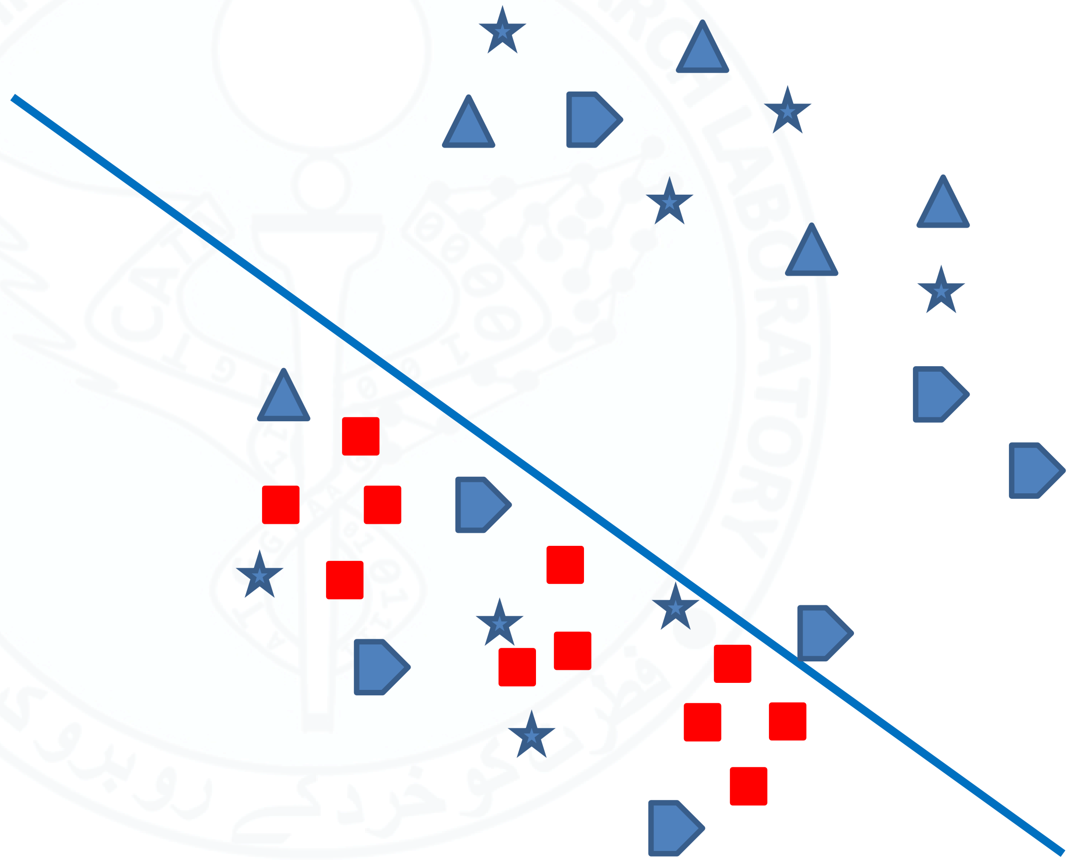
$$y_i = -1 \qquad \forall\, I \; s.t. \; Y_I = -1 \qquad (2)$$

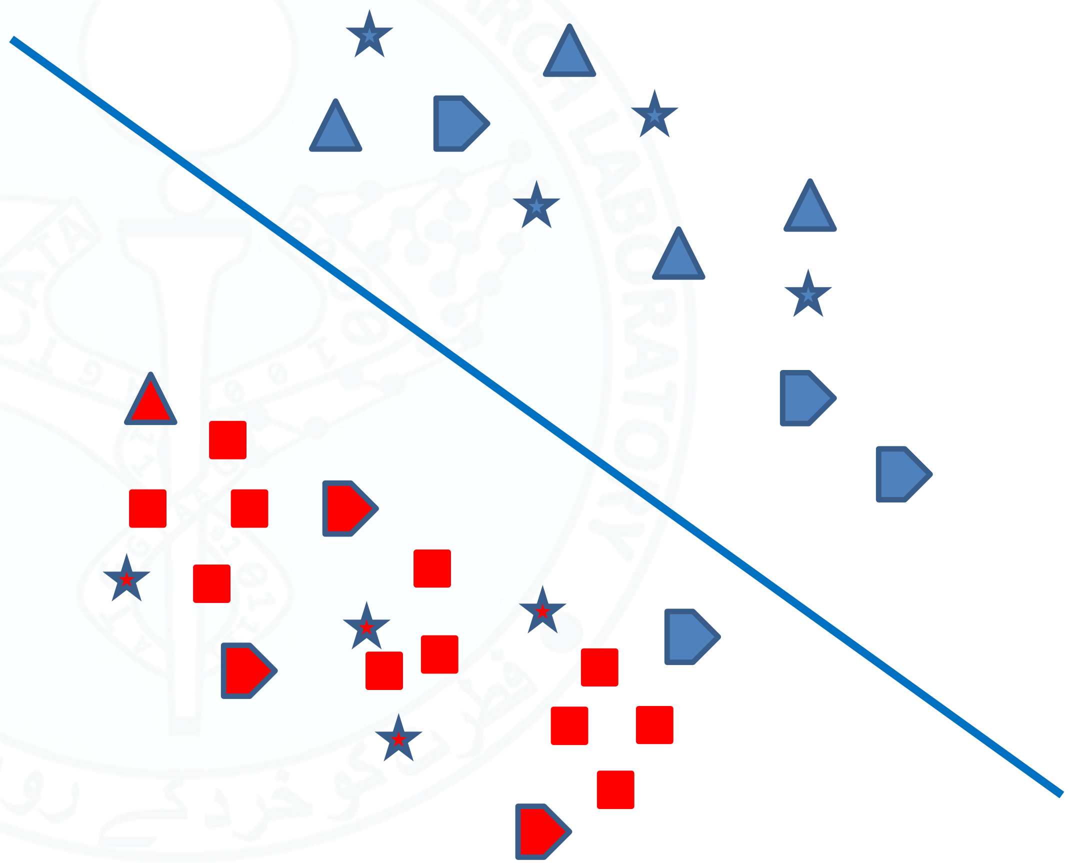
where, y_i *is the label for instance* $\boldsymbol{x}_i$ *and* $\boldsymbol{x}_i \in \boldsymbol{B_I}$.

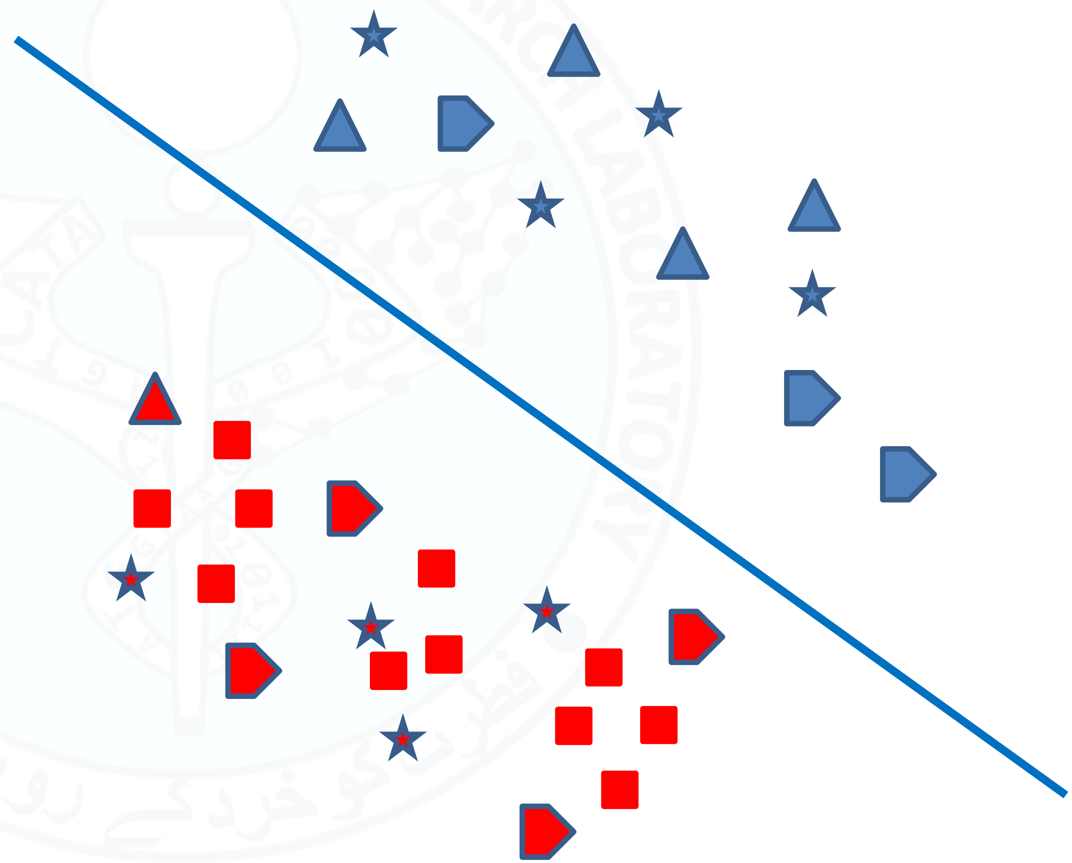
mi-SVM

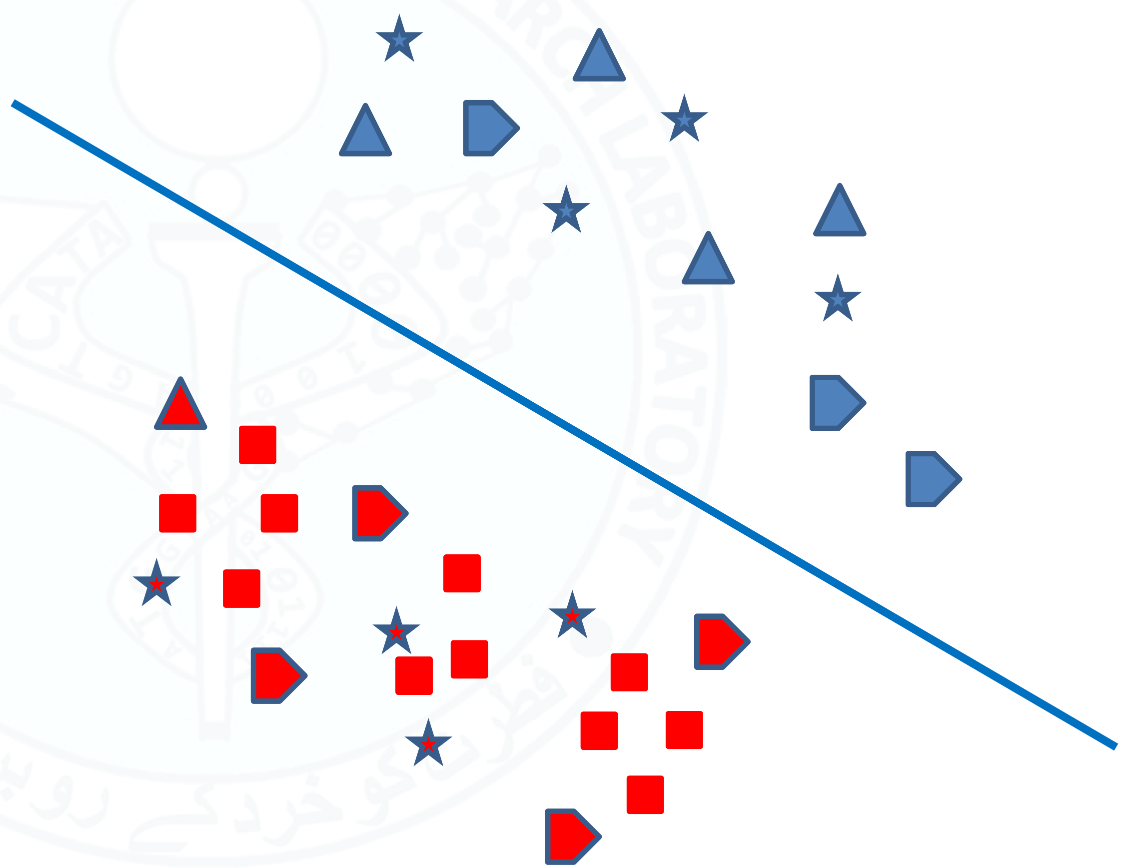
initialize $y_i = Y_I$ for $i \in I$
REPEAT
  compute SVM solution $\mathbf{w}, b$ for data set with imputed labels
  compute outputs $f_i = \langle \mathbf{w}, \mathbf{x}_i \rangle + b$ for all $\mathbf{x}_i$ in positive bags
  set $y_i = \text{sgn}(f_i)$ for every $i \in I$, $Y_I = 1$
  FOR (every positive bag B_I)
    IF $(\sum_{i \in I} (1 + y_i)/2 == 0)$
      compute $i^* = \arg\max_{i \in I} f_i$
      set $y_{i^*} = 1$
    END
  END
WHILE (imputed labels have changed)
OUTPUT $(\mathbf{w}, b)$









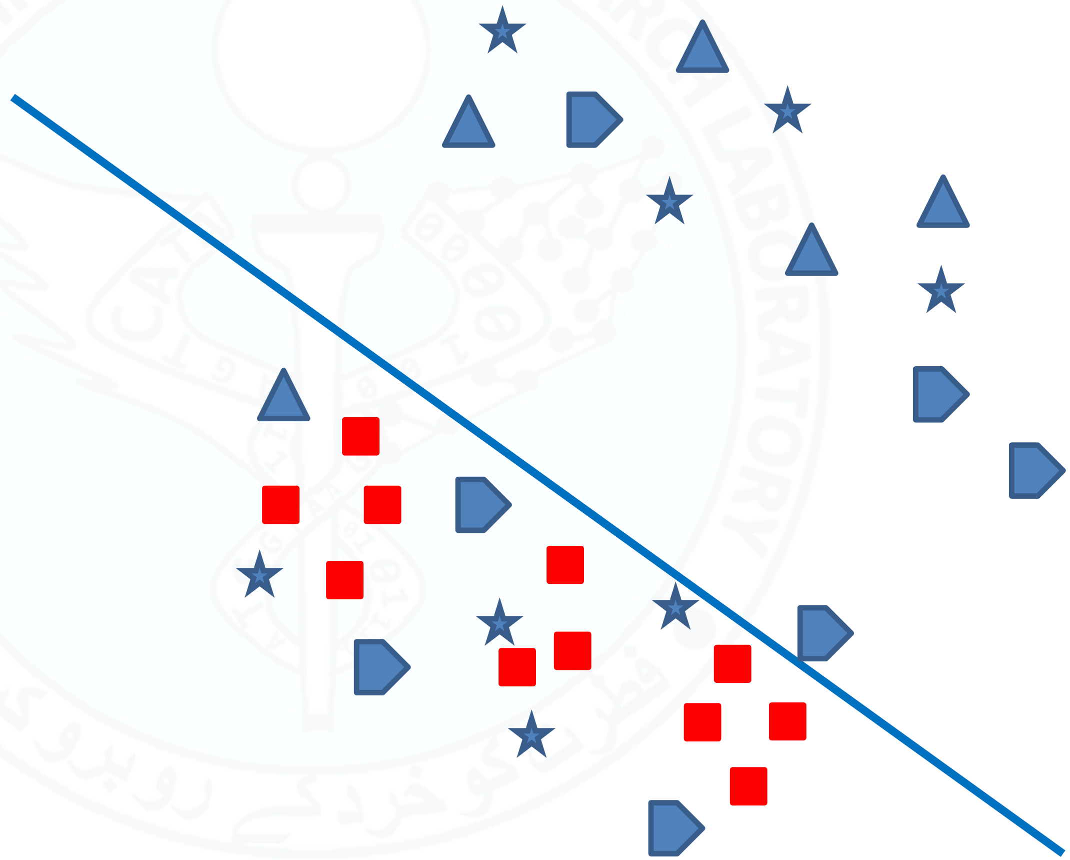


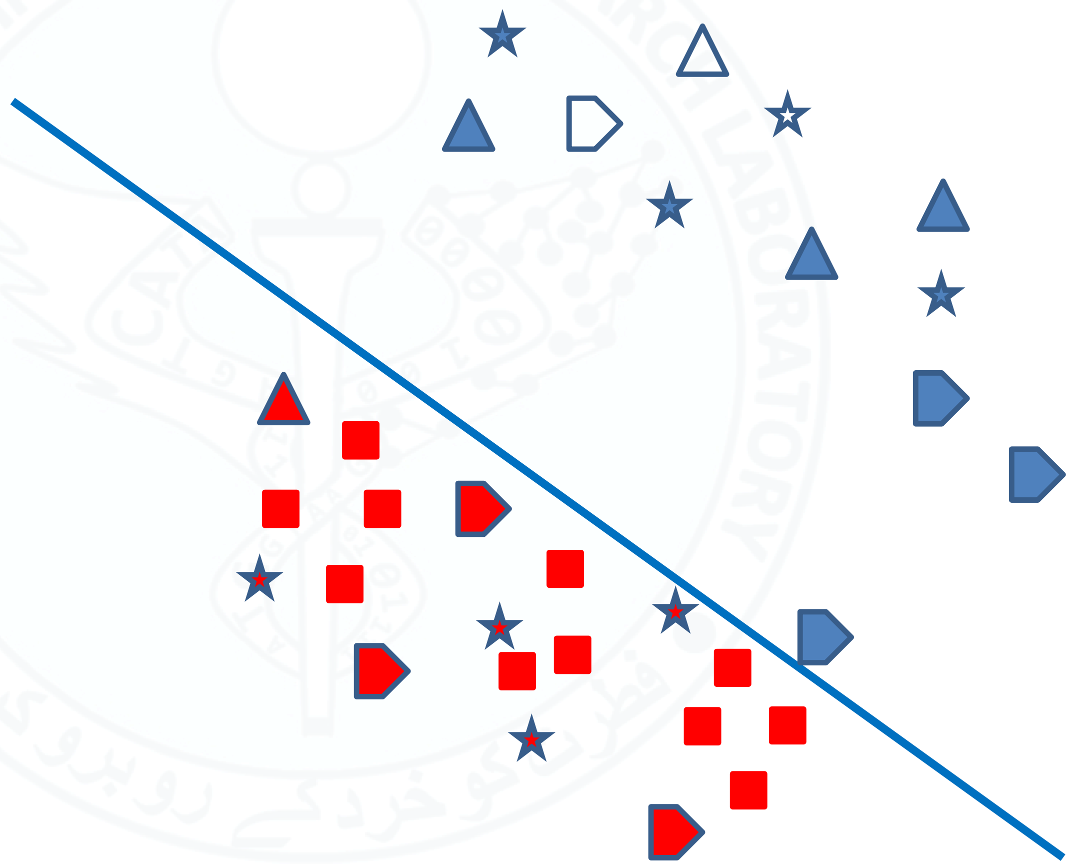
MI-SVM

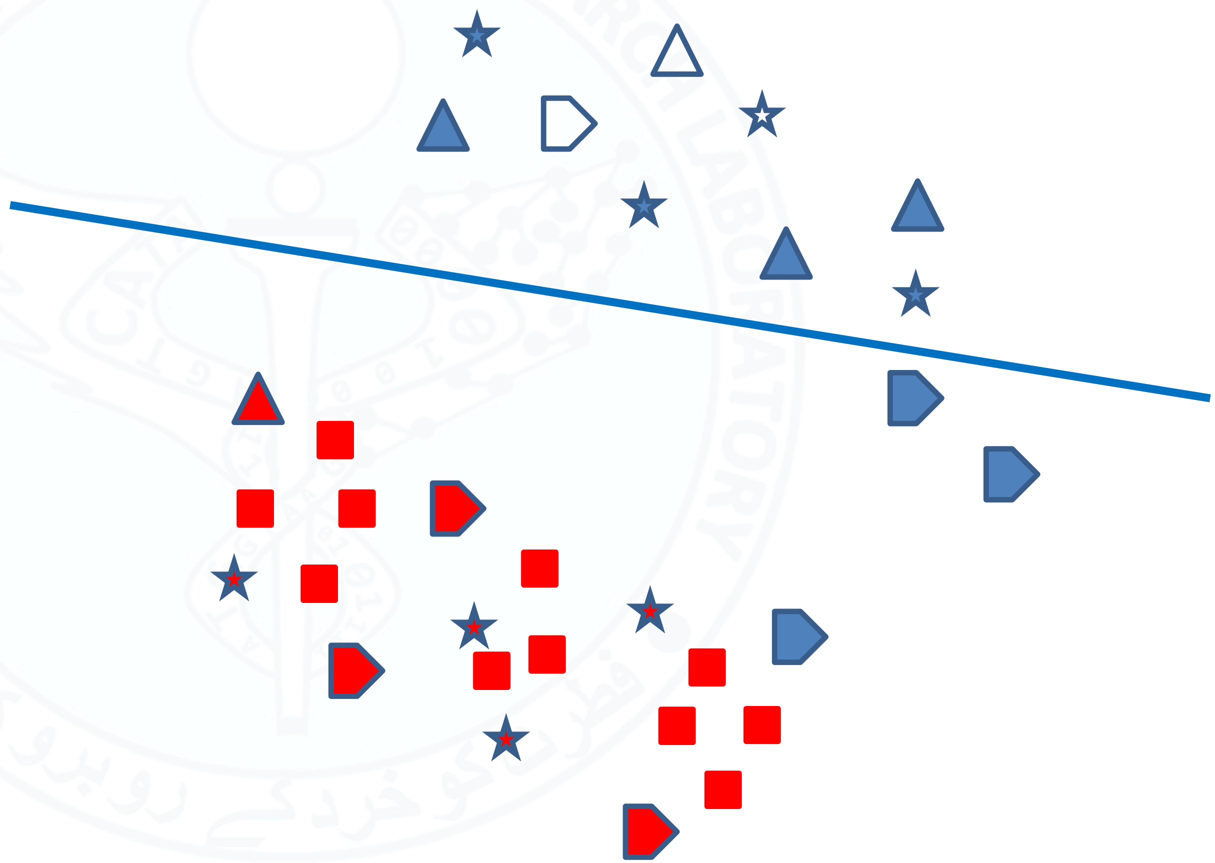
$$\textit{MI-SVM} \qquad \min_{\mathbf{w},b,\xi} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_I \xi_I$$

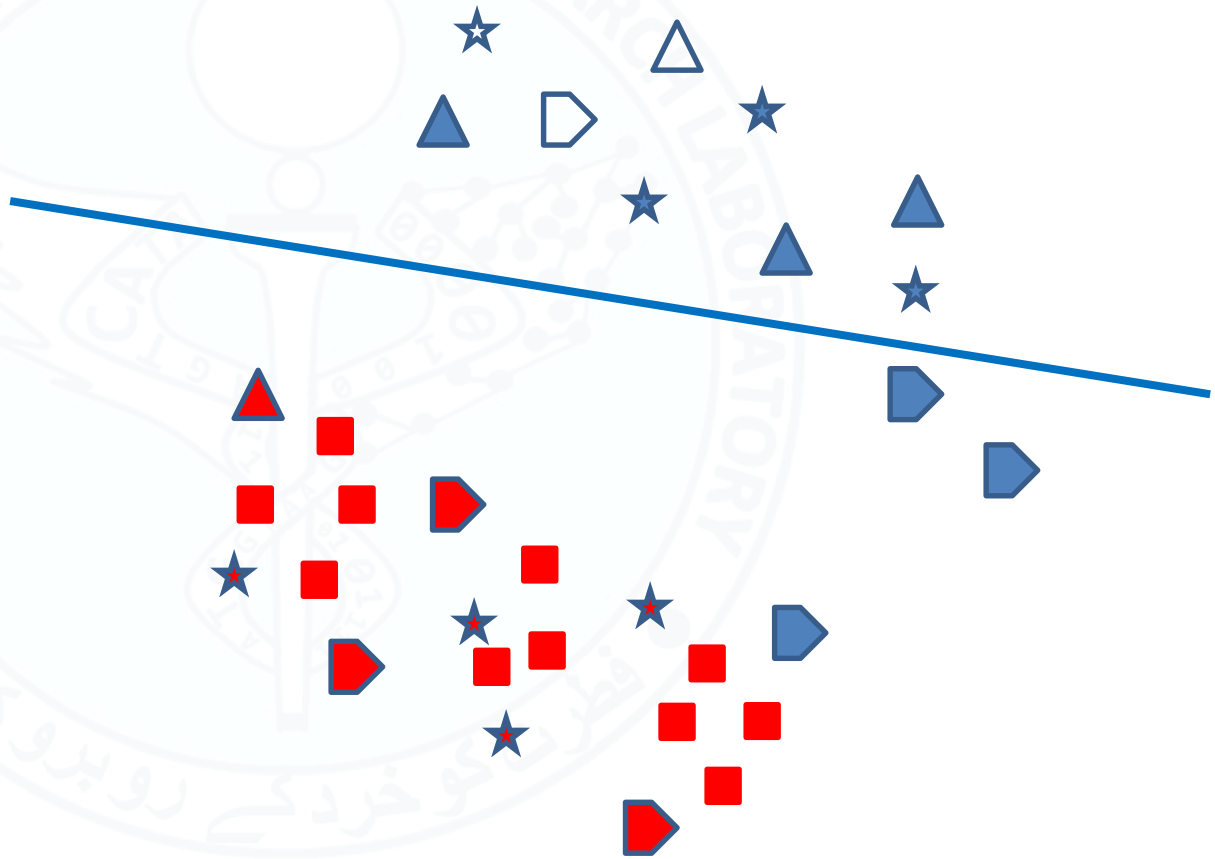
$$\text{s.t.} \quad \forall I: \; Y_I \max_{i\in I}(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) \geq 1 - \xi_I, \; \xi_I \geq 0.$$

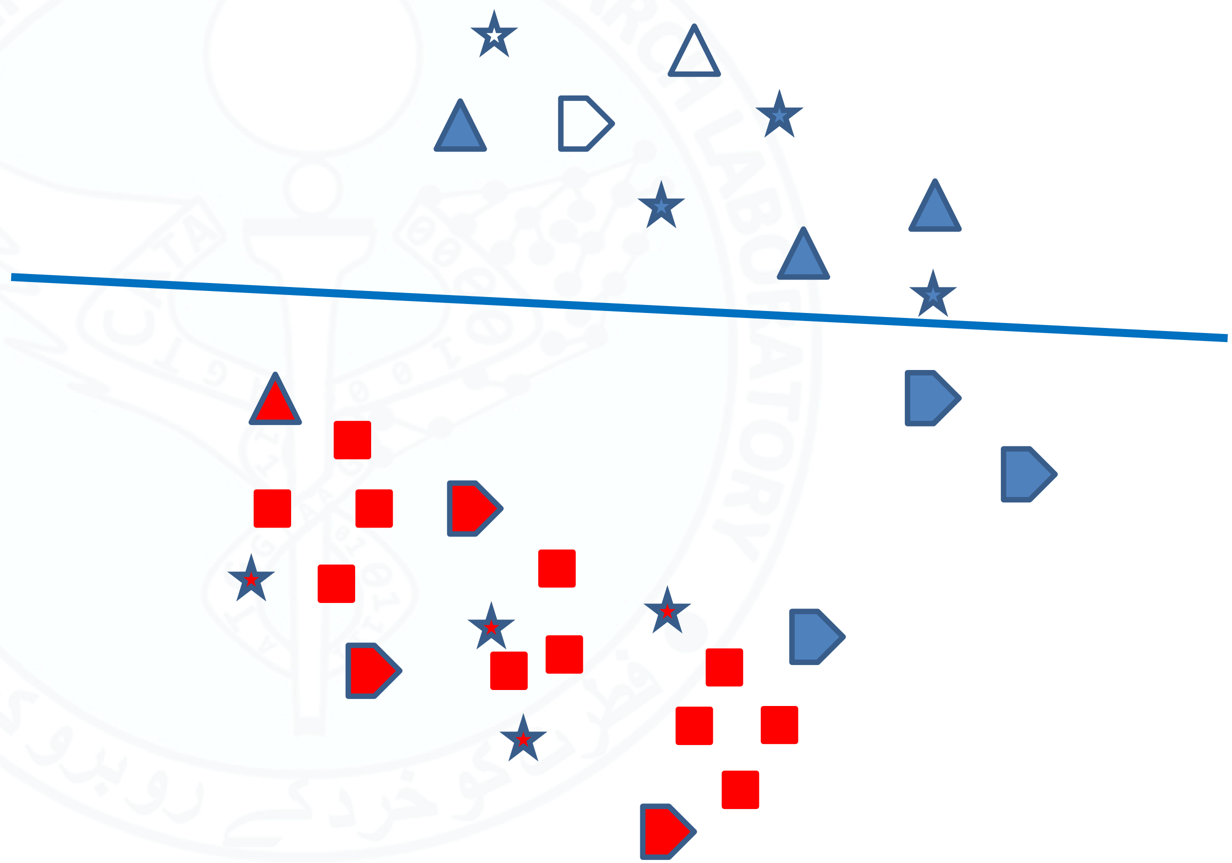
initialize $\mathbf{x}_I = \sum_{i\in I} \mathbf{x}_i / |I|$ for every positive bag B_I
REPEAT
    compute QP solution $\mathbf{w}, b$ for data set with
        positive examples $\{\mathbf{x}_I : Y_I = 1\}$
    compute outputs $f_i = \langle \mathbf{w}, \mathbf{x}_i \rangle + b$ for all $\mathbf{x}_i$ in positive bags
    set $\mathbf{x}_I = \mathbf{x}_{s(I)}$, $s(I) = \arg\max_{i\in I} f_i$ for every I, $Y_I = 1$
WHILE (selector variables $s(I)$ have changed)
OUTPUT $(\mathbf{w}, b)$







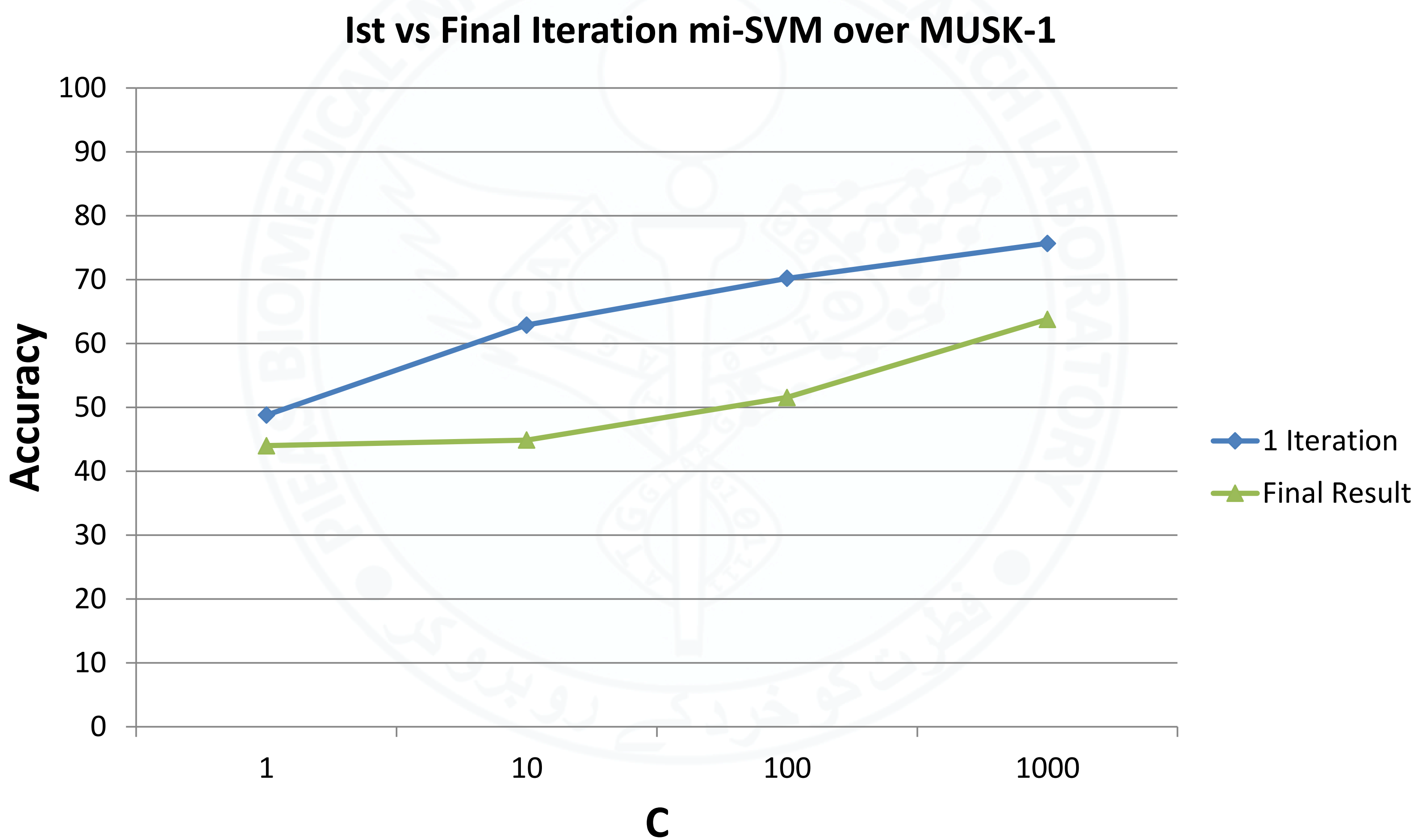




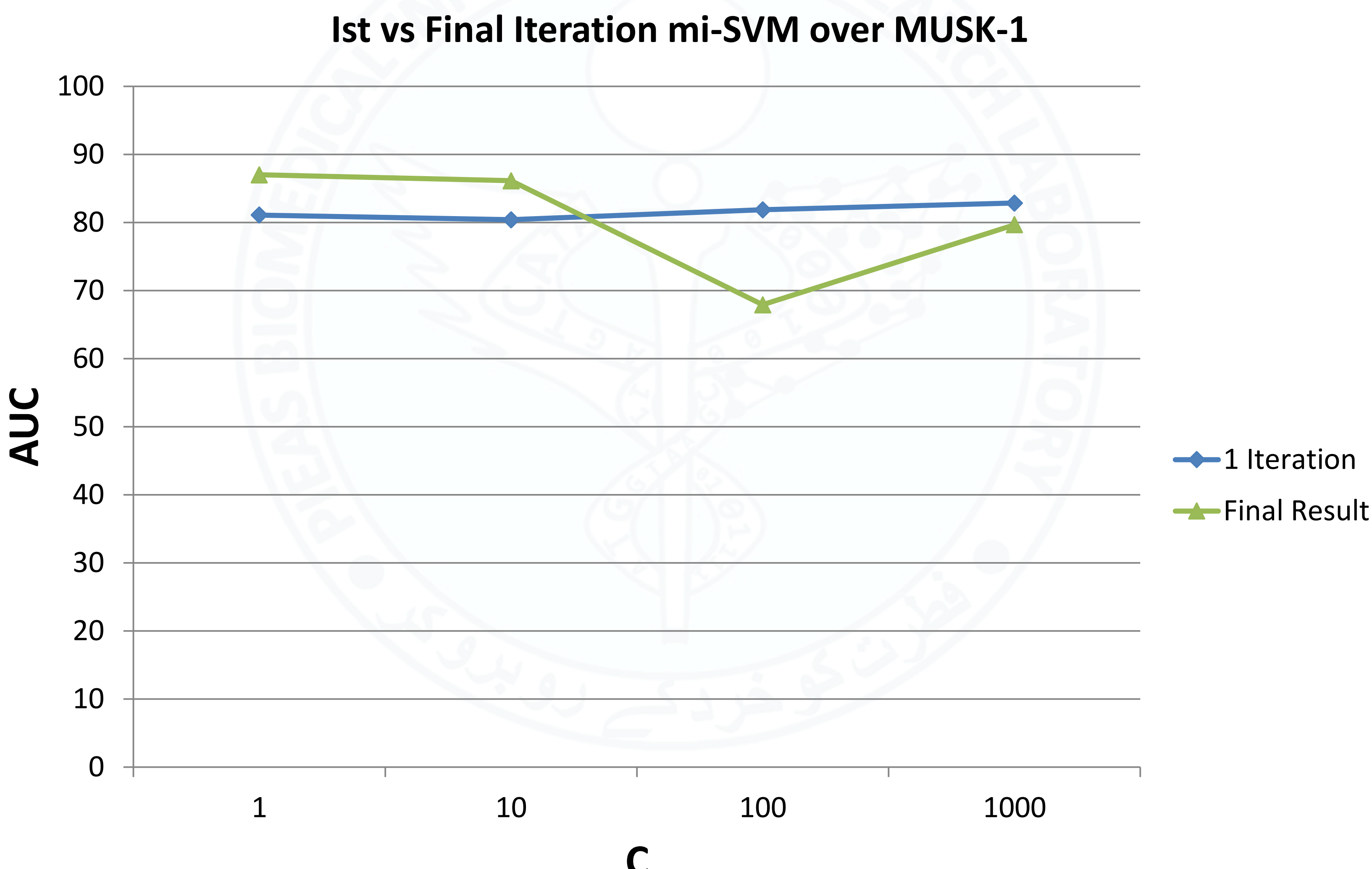
Problems

- mi-SVM and MI-SVM both use local optimization heuristics
  - Can produce sub-optimal solutions
- Require iterative training of the classifier
  - Computationally expensive
- Cannot work for big data

No Guaranteed Optimal Solution



No Guaranteed Optimal Solution



Multiple Instance Ranking

Find a ranking function f parameterized by w which can generate output values for any given training bag B_I

$$Y_I = f(\boldsymbol{B_I}; \boldsymbol{w}) \quad \forall I$$

such that,

$$f(\boldsymbol{B_I}; \boldsymbol{w}) > f(\boldsymbol{B_J}; \boldsymbol{w}) \quad if \;\; Y_I > Y_J \qquad (3)$$

Multiple Instance Regression

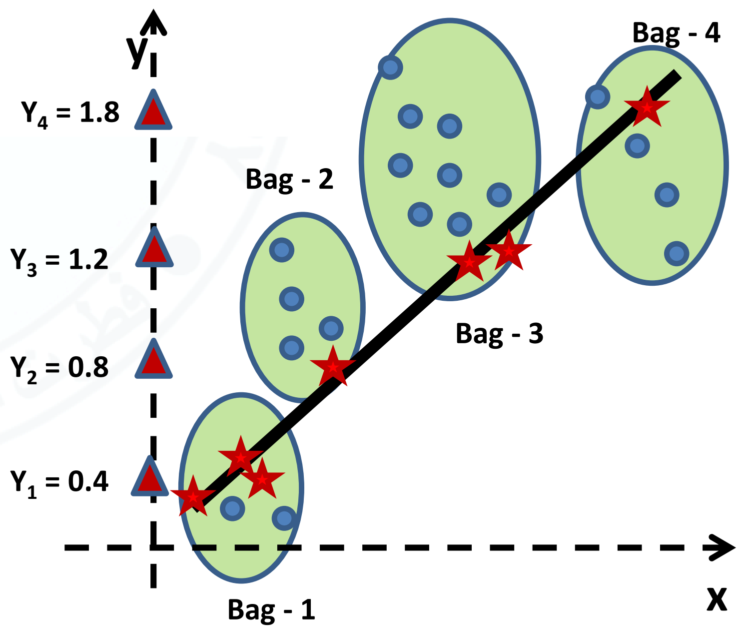
Find a function f parameterized by w which can generate output values of any given training bag B_I

$$Y_I = f(\boldsymbol{B_I};\boldsymbol{w}) \quad \forall I$$

such that,

$$|Y_I - f(\boldsymbol{B_I};\boldsymbol{w})| \leq \epsilon \qquad (4)$$

Where, $\epsilon \geq 0$ is the maximum error allowed.



Applications

Method	Features	AUC	$AUC_{0.1}$	TH %	FH %
Vanilla SVM	1-Spec	95.5	53.9	**66**	2.6
	PD-1	95.6	54.5	64	2.5
	Comb.	95.9	55.1	65	2.1
	Max. Std.	0.16	0.59	2.2	0.15
mi-SVM	1-Spec	95.5	**54.4**	64	2.6
	PD-1	96.0	55.8	69	2.1
	Comb.	96.2	55.6	68	1.9
MI-1 SVM	1-Spec	**96.0**	54.3	62	**2.1**
	PD-1	**96.8**	**58.5**	**72**	**1.3**
	Comb.	**96.9**	**59.0**	**75**	**1.2**
	Max Std.	0.14	0.80	3.4	0.11
	Gappy	96.5	58.5	68	1.6

- Andrews, Stuart, Ioannis Tsochantaridis, and Thomas Hofmann. 2003. “Support Vector Machines for Multiple-Instance Learning.” In *Advances in Neural Information Processing Systems 15*, 561–68. MIT Press.
- Leistner, Christian, Amir Saffari, and Horst Bischof. 2010. “MIForests: Multiple-Instance Learning with Randomized Trees.” In *Computer Vision – ECCV 2010*, edited by Kostas Daniilidis, Petros Maragos, and Nikos Paragios, 29–42. Lecture Notes in Computer Science 6316. Springer Berlin Heidelberg. http://link.springer.com/chapter/10.1007/978-3-642-15567-3_3.

- https://en.wikipedia.org/wiki/Multiple-instance_learning

We want to make a machine that will be proud of us.

- Danny Hillis