Classification (Linear)

- Figure out, autonomously, which category (or class) an "unknown item" should be categorized into.

- Number of Categories/Classes:
    * Binary: 2 different categories
    * Mulitclass: More than 2 categories

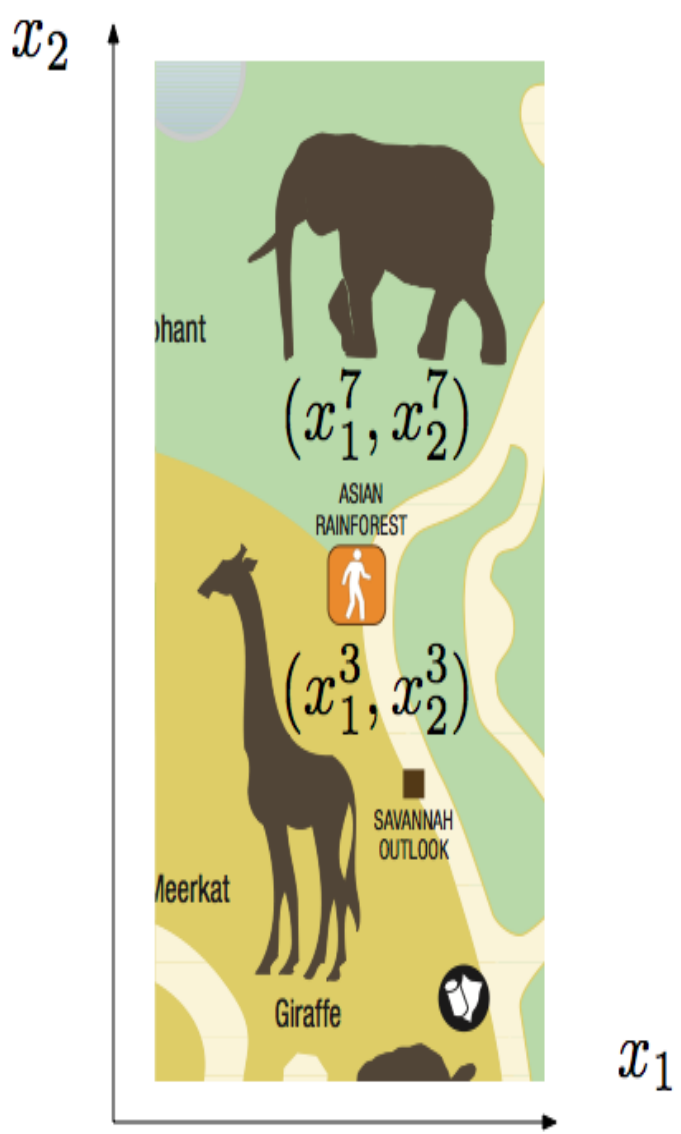
- Features: the measurable parts that make up the "unknown item" (or the information you have available to catagorize)

Illustrative Example: The Zoo

- Binary Classification:
    * Elephants vs Giraffes

- Features:
    * the coordinate of the "unknown" animal i in the zoo: (x_1^i, x_2^i) or

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$



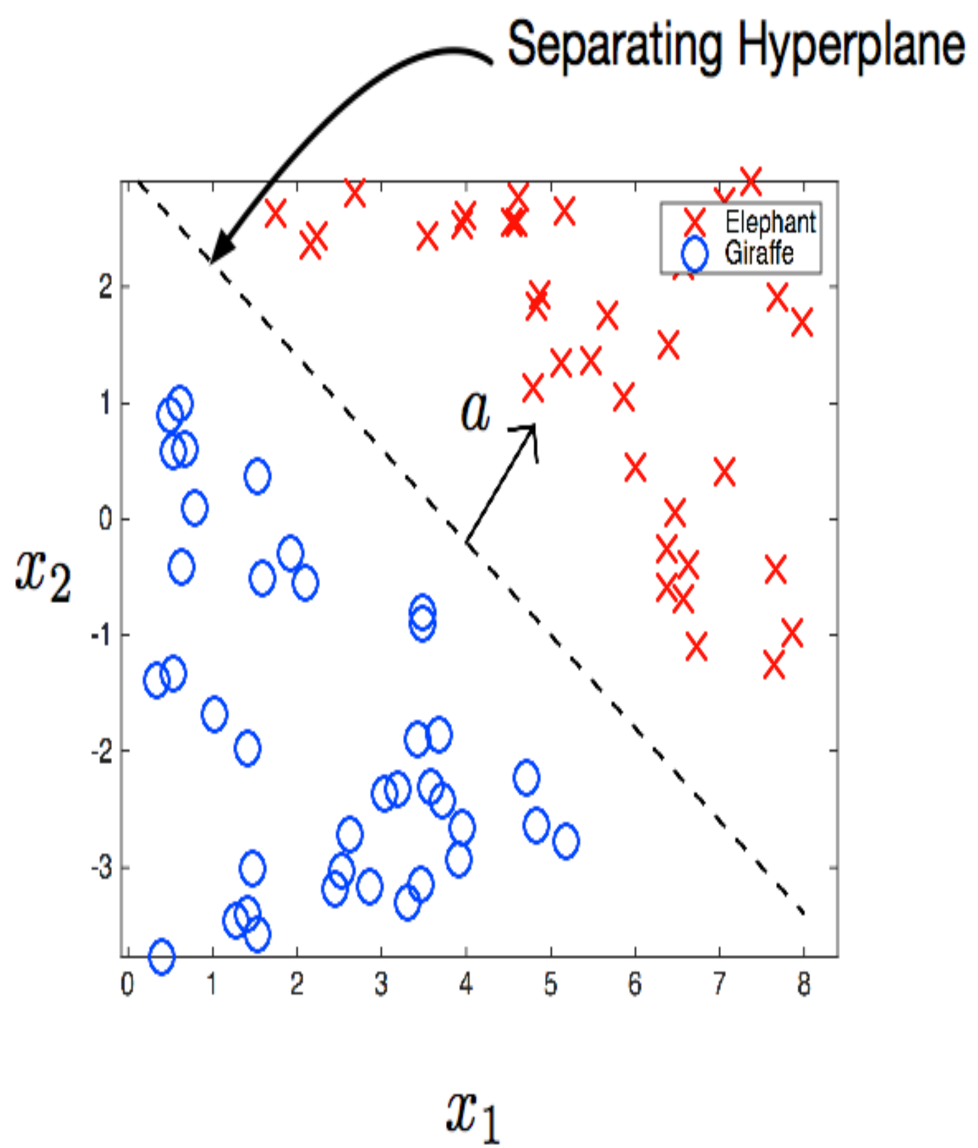
The Zoo: cont...

- Is it possible to distinguish between an elephant and a giraffe by it's coordinates on a map of the zoo?

- We need to FIND a separating hyperplane (or a line in 2D):

$$\begin{bmatrix} a_1 & a_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - b = 0$$

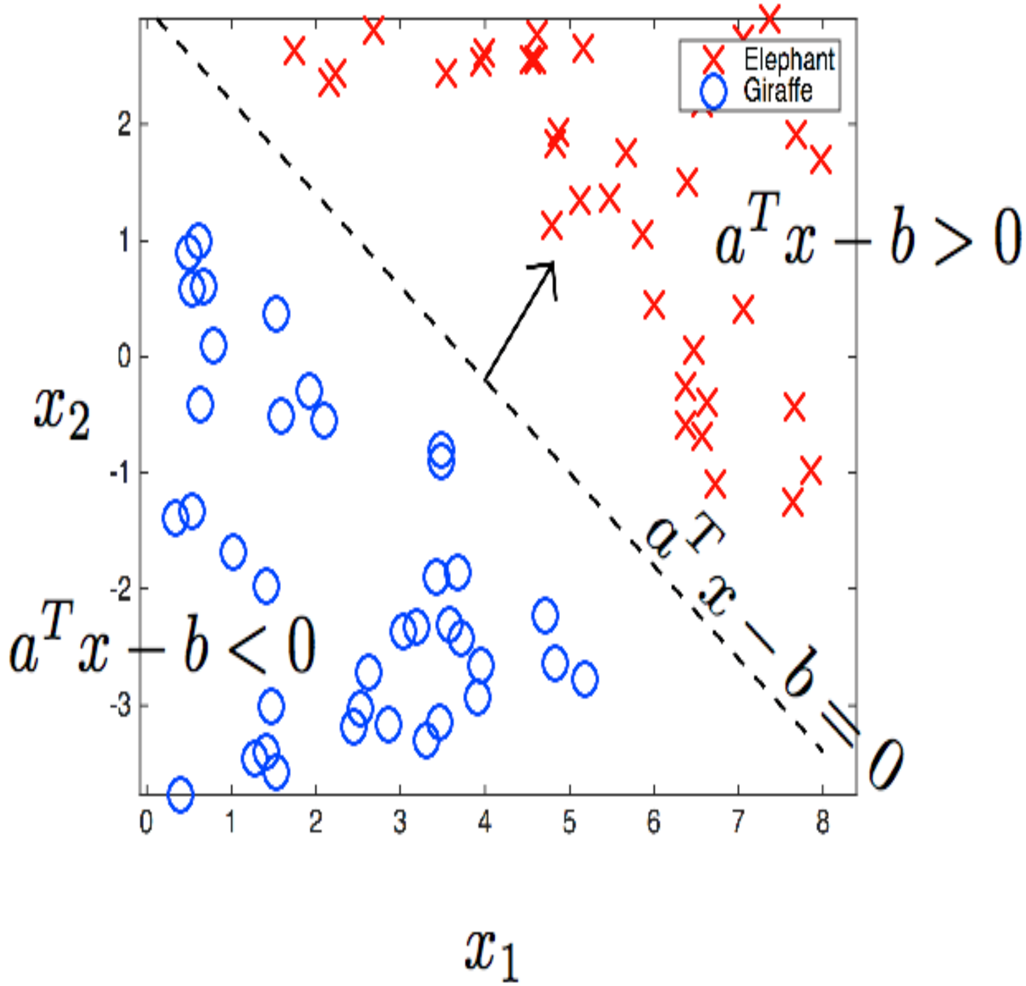
or $\quad a^T x - b = 0$



The Zoo: cont...

- Given:
  * Hyperplane defined by a and b
  * an animals coordinates (or features) x

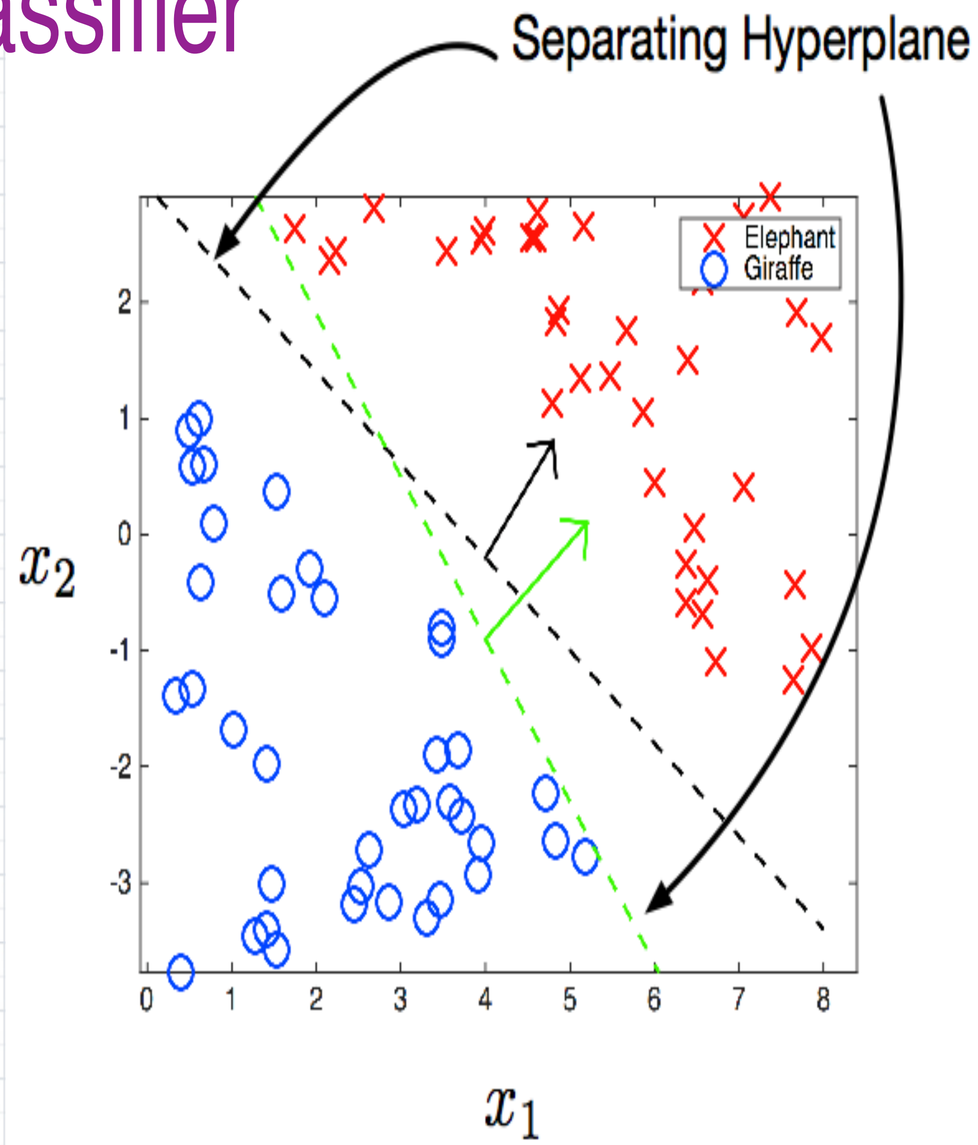
- Decision Making:
  * if $a^T x - b > 0$ than it's an elephant
  * if $a^T x - b < 0$ than it's a Giraffe



Generate a (Linear) Classifier

- Automate the generation
  of a hyperplane.
  (or support vector)

- Training a classifier:
  * Given: a "training" set
    of labeled data:
    - Example: 100 animals
      (coordinates) are given to
      you, labeled "elephant" or "giraffe"



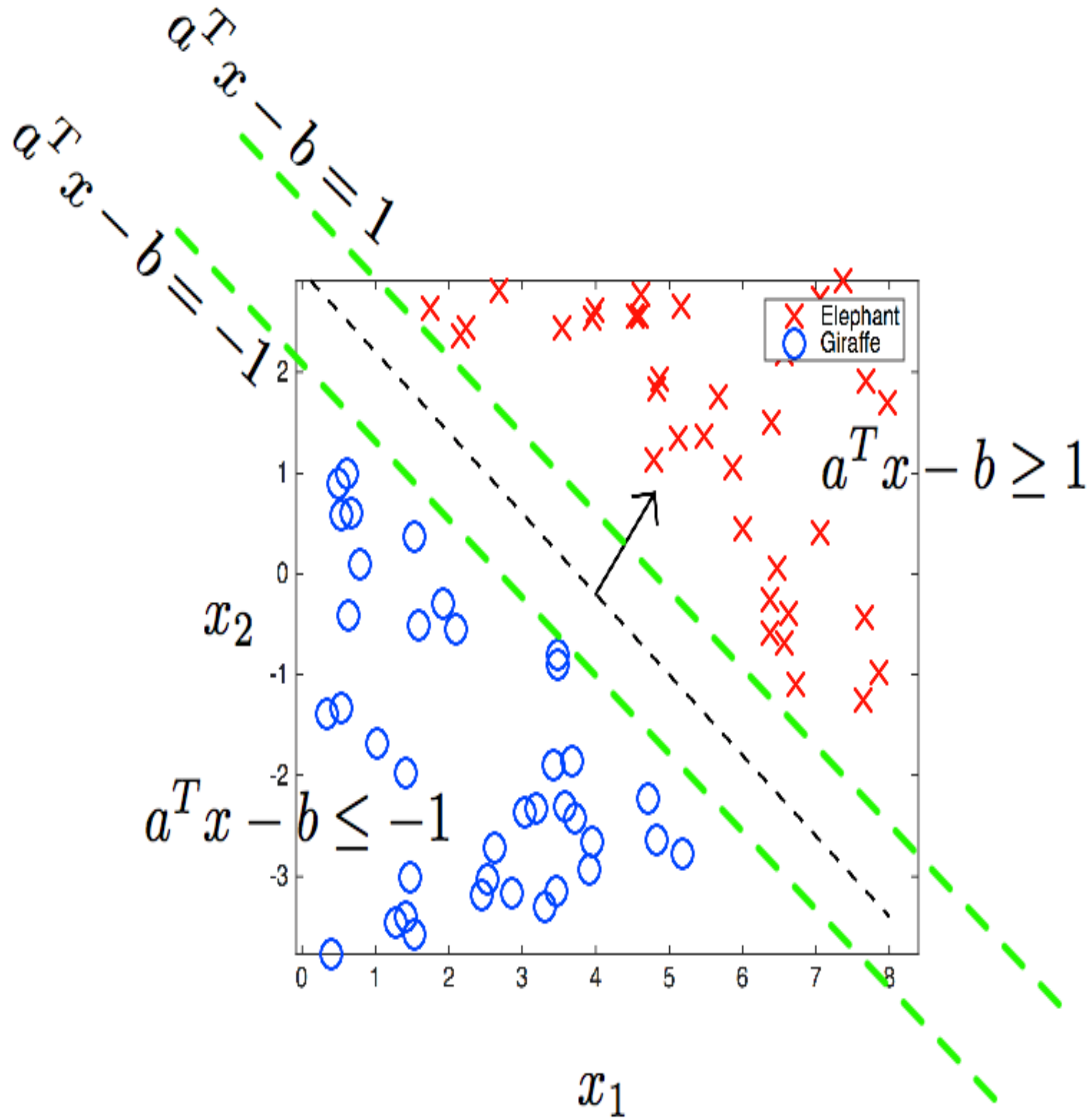
The Zoo: cont...

- Find a and b such that an elephant given $a^T x - b > 0$

or

- Find a and b such that an elephant given $a^T x - b \geq 1$

- Same problem if strictly seperable



LP Formulation 1

- assume n features (2 for the zoo problem)
- assume m data points $x_i = [x_1^i \; x_2^i]^T$ in training set (so 100 animals)
- assume q (of m) are elephants in the training set
- assume r (of m) are Giraffes
- a and b are the variables (unknown)

minimize Something

subject to

$$\text{elephant} \begin{cases} a^T x^1 - b \geq 1 \\ a^T x^2 - b \geq 1 \\ \vdots \\ a^T x^q - b \geq 1 \end{cases}$$

$$\text{giraffe} \begin{cases} a^T x^{q+1} - b \leq -1 \\ a^T x^{q+2} - b \leq -1 \\ \vdots \\ a^T x^{q+r} - b \leq -1 \end{cases}$$

LP Formulation 2

- assume n features
  (2 for the zoo problem)
- assume m data points in training set (so 100 animals)
- assume q (of m) are elephants in the training set
- assume r (of m) are Giraffes
- need slack variable u and v, where all the elements are positive (u_i, v_i >= 0)

minimize $\sum_{i=1}^{q} u_i + \sum_{i=1}^{r} v_i$

subject to

$$
\begin{aligned}
a^T x^1 - b &\geq 1 - u_1 \\
a^T x^2 - b &\geq 1 - u_2 \\
&\vdots \\
a^T x^q - b &\geq 1 - u_q \\
a^T x^{q+1} - b &\leq -(1 - v_1) \\
a^T x^{q+2} - b &\leq -(1 - v_2) \\
&\vdots \\
a^T x^{q+r} - b &\leq -(1 - v_{q+r})
\end{aligned}
$$

LP Formulation 3

- since $a^T x - b$ is a scalar, it can also be written as

$x^T a - b$

$$\begin{aligned} \text{minimize} \quad & \mathbf{1}^T u + \mathbf{1}^T v \\ \text{subject to} \quad & X_e a - b \geq \mathbf{1} - u \\ & X_g a - b \leq -(\mathbf{1} - v) \\ & u > \mathbf{0} \\ & v > \mathbf{0} \end{aligned}$$

- Where:

$$X_e = \begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_n^1 \\ x_1^2 & x_2^2 & \cdots & x_n^2 \\ \vdots & \vdots & \vdots & \vdots \\ x_1^q & x_2^q & \cdots & x_n^q \end{bmatrix} = \begin{bmatrix} (x^1)^T \\ (x^2)^T \\ \vdots \\ (x^q)^T \end{bmatrix}$$

and

$$X_g = \begin{bmatrix} (x^{q+1})^T \\ (x^{q+2})^T \\ \vdots \\ (x^{q+r})^T \end{bmatrix}$$

Code (using cvx)

$$\begin{aligned} \text{minimize} \quad & \mathbf{1}^T u + \mathbf{1}^T v \\ \text{subject to} \quad & X_e a - b \geq \mathbf{1} - u \\ & X_g a - b \leq -(\mathbf{1} - v) \\ & u > \mathbf{0} \\ & v > \mathbf{0} \end{aligned}$$

```
%%%%%%%% Pull out the Giraffes and
%%%%%%%% Elephants from the DATA
Giraffes = Animals(Flag2,:);
Elephants = Animals(Flag1,:);

R = length(Elephants(:,1));
Q = length(Giraffes(:,1));
n = 2;

%%%%%%%% Solve for the support vector
cvx_begin
    variables a(n) b(1) u(R) v(Q)
    minimize (ones(1,R)*u + ones(1,Q)*v)
    Elephants*a - b >= 1 - u;
    Giraffes*a - b <= -(1 - v);
    u >= 0;
    v >= 0;
cvx_end
```

```
>> Elephants

Elephants =

    2.6738    2.8013
    6.7301   -1.0902
    5.4748    1.3566
    4.5327    2.5654
    4.8288    1.8400
    5.8573    1.0614
```

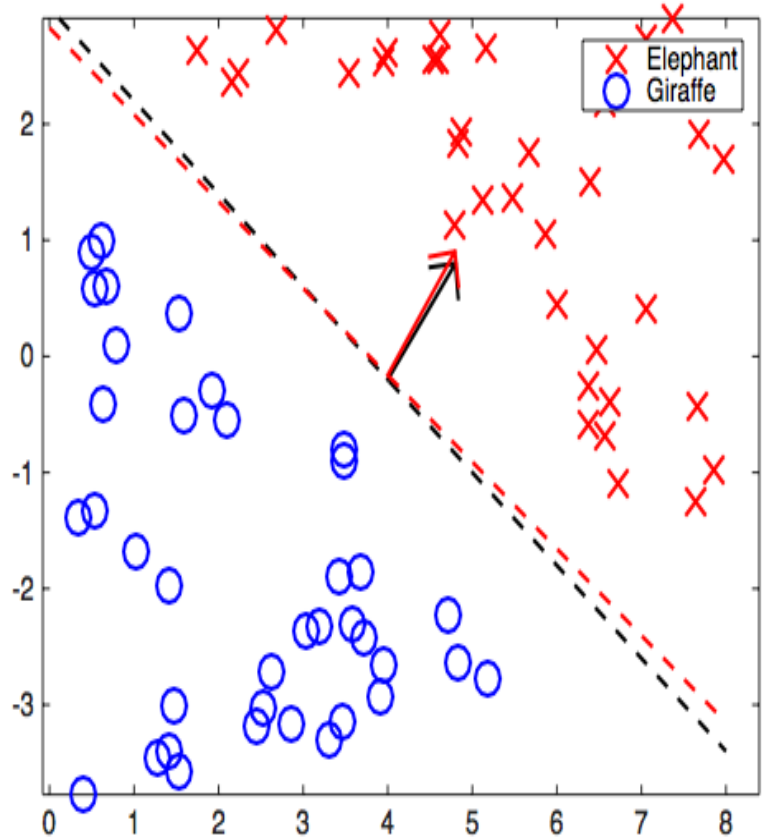
Solution:

```
>> a

a =

    0.5117
    0.6203

>> b

b =

    1.9312
```

Results 1

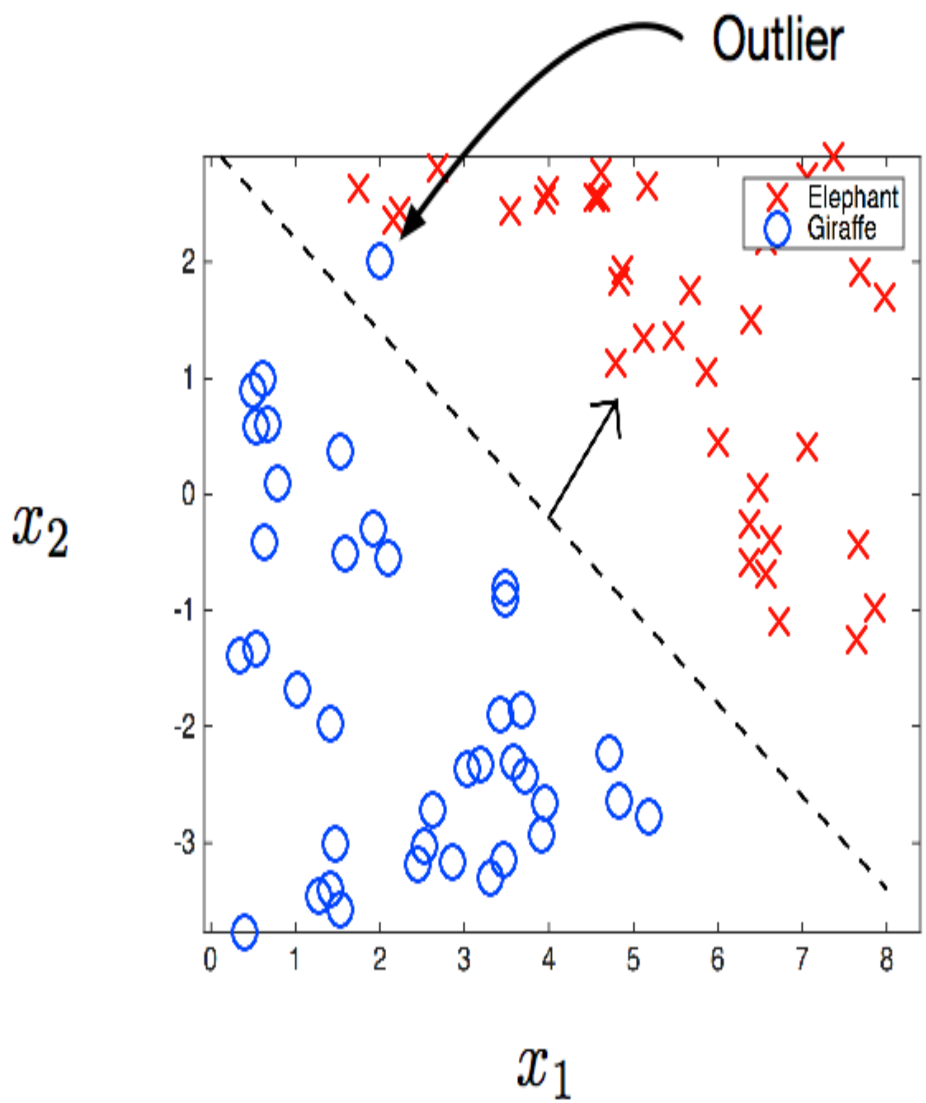
- Black Hyperplane: the actual hyperplane use to produce the x's and o's

- Red Hyperplane: the result of the optimization problem using the x's and o's as training data



Results 2: Outlier

- Note that in the "real-world", you may have noise, errors, or outliers that don't accurately represent the actual phenomena.

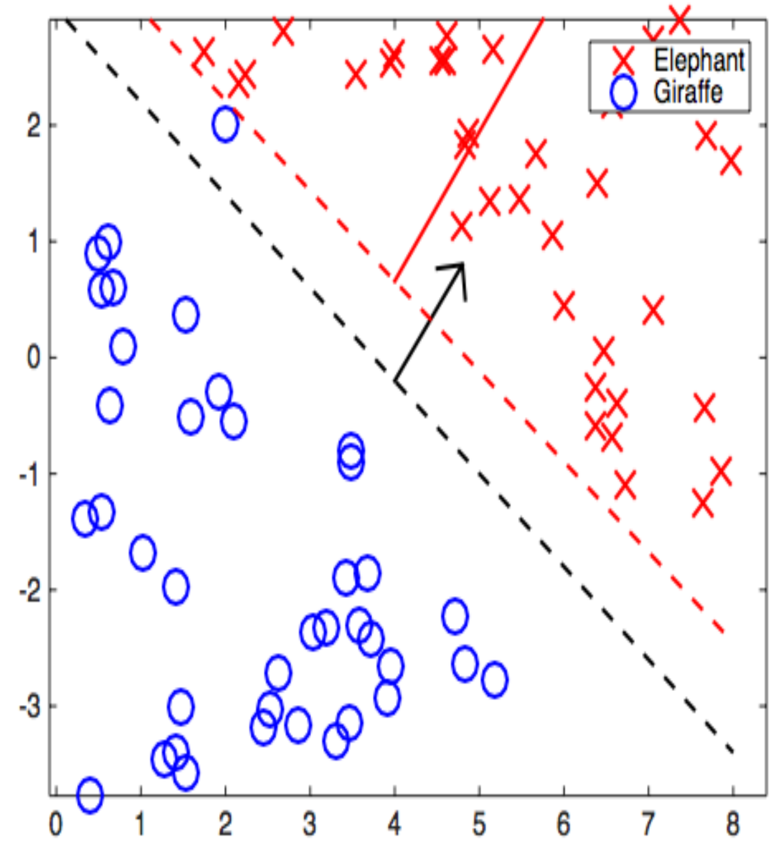


Results 3: Outlier

- Notice that the Red Hyperplane, is not as accurately represent the division due to the outlier

- Q: Can We do better when we have noisy data or outliers?

- A: Yes, but we need to look beyond LP.



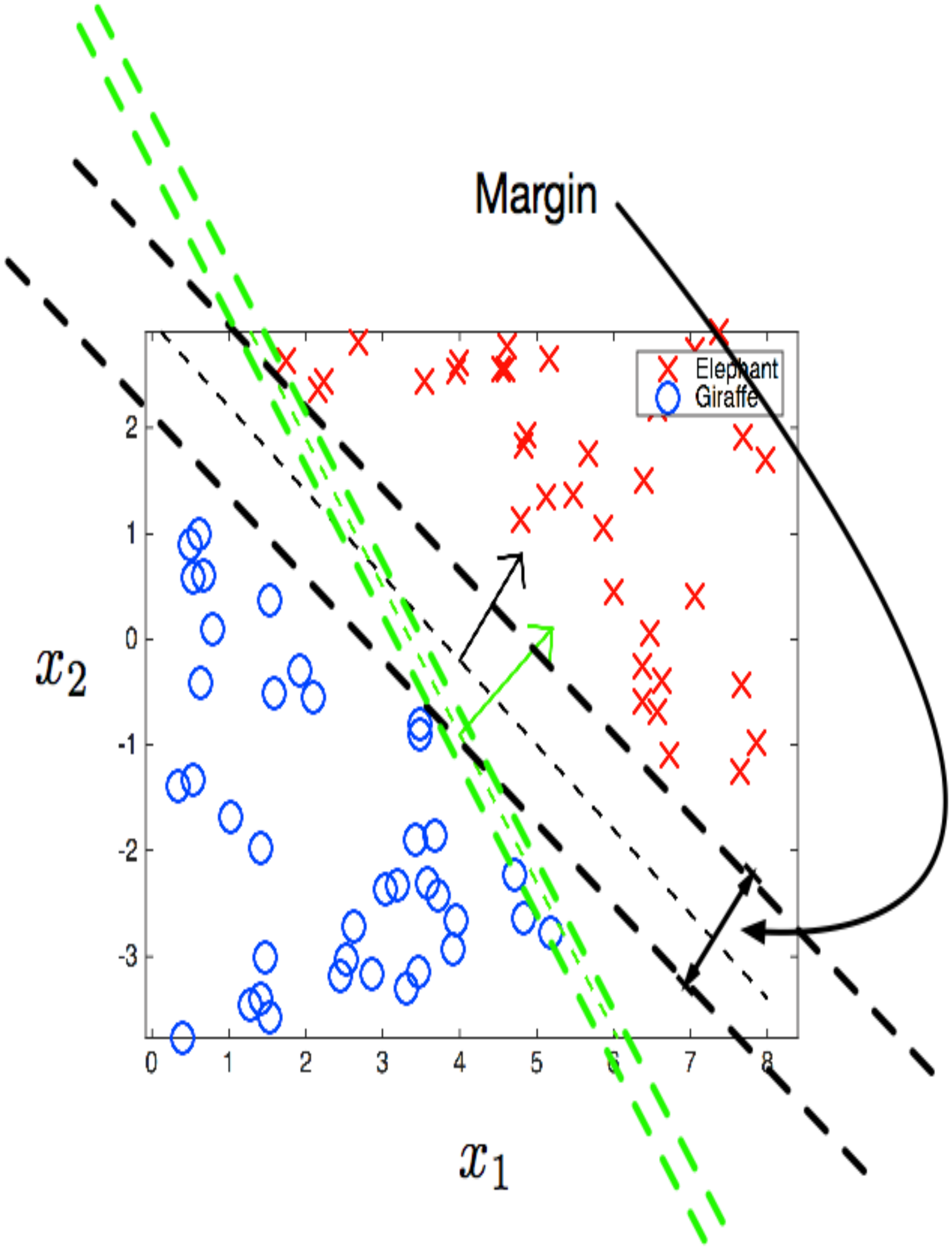
Maximize Margin

- Black's margin is bigger than green's

- Black is more accurate

- $$\text{Margin} = \frac{2}{||a||_2}$$

- minimize $||a||_2$ to maximize the margin



Code (using cvx)

```
%%%%%%%% Pull out the Giraffes and
%%%%%%%% Elephants from the DATA
Giraffes = Animals(Flag2,:);
Elephants = Animals(Flag1,:);

R = length(Elephants(:,1));
Q = length(Giraffes(:,1));
n = 2;
g = .1;

%%%%%%%% Solve for the support vector
cvx_begin
    variables a(n) b(1) u(R) v(Q)
    minimize (norm(a) + g*(ones(1,R)*u + ones(1,Q)*v))
    Elephants*a - b >= 1 - u;
    Giraffes*a - b <= -(1 - v);
    u >= 0;
    v >= 0;
cvx_end
```

$$\begin{aligned} \text{minimize} \quad & \|a\|_2 + \gamma(\mathbf{1}^T u + \mathbf{1}^T v) \\ \text{subject to} \quad & X_e a - b \geq \mathbf{1} - u \\ & X_g a - b \leq -(\mathbf{1} - v) \\ & u > 0 \\ & v > 0 \end{aligned}$$

- use γ as a weighting between the following 2 desires:

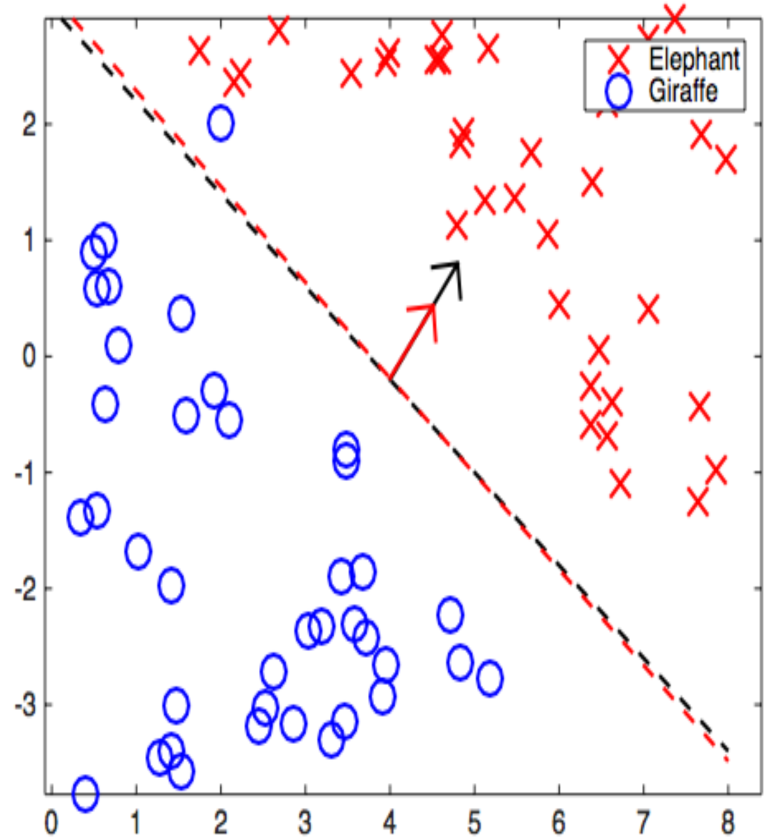
  * Bigger Margin given robustness to outliers
  * A hyperplane that has few (or no) errors

Results 4: Robust to Outliers

- Notice that the Red Hyperplane, is now more representative of the true division, but it will have an error due to the outlier.

- It is an iterative process to choose a γ that seems to give the "best results"

- Q: What do you mean by "best results"?



Steps to generating a Classifier

1.) Split your labelled data into:
   a.) Training Set
   b.) Test Set
2.) Train your Classifier (pick a γ) using your Training Set
3.) Test your Classifier with your Test Set which was not used in the training of the Classifier.
4.) Measure accuracy (total error for example) and go back to 2.)
5.) Possibly identify a minimal set of feature that has the same predictive power which reduces the model size
6.) Use the Classifier

APPLICATION:
Handwriting Character Recognition

- United States Postal Service (USPS) uses this to automate zip code reading.

- Multiclass Problem: need to differentiate between digits 0-9

- Q: what is the feature vector and how long is it?

Handwriting Images (Data)

- MNIST Database: yann.lecun.com/exdb/mnist/

- 60,000 character "training" data set and 10,000 character "testing" set

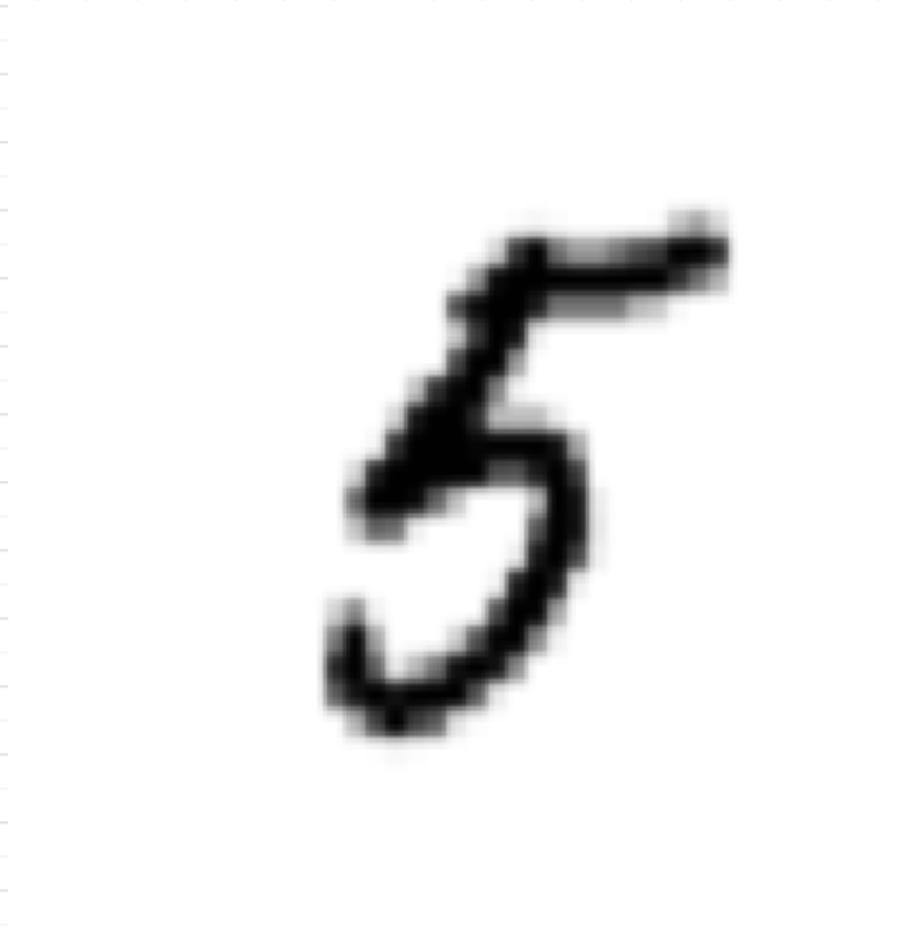
- Each image is a 28 X 28 pixel image where each image is a feature. This gives 784 features! (so n = 784, not 2 like the zoo)

Handwriting Images 2

- Database we will supply you gives you images as 784 long vectors. (ready for optimization)

- To view images, you need to reshape the vector into a 28 X 28 matrix of pixels:

```
ImageNumber = 42; % any number between 1-60000
IMAGE = reshape(images(ImageNumber,:),28,28)'
figure,imagesc(IMAGE)
colormap(flipud(gray(256)))
axis equal
set(gca, 'YTick', []);
set(gca, 'XTick', []);
axis off
```



Multiclass Classifier

- Option 1: ONE vs. ALL classifiers (10 total):
    * 0 vs All the rest
    * 1 vs All the rest
    * etc ...
    * 9 vs All the rest

- Option 2: Pairwise (many BINARY classifiers, 45 total):
    * 0 vs. 3
    * 1 vs. 3
    * 2 vs. 3
    * ...
    * 9 vs 3

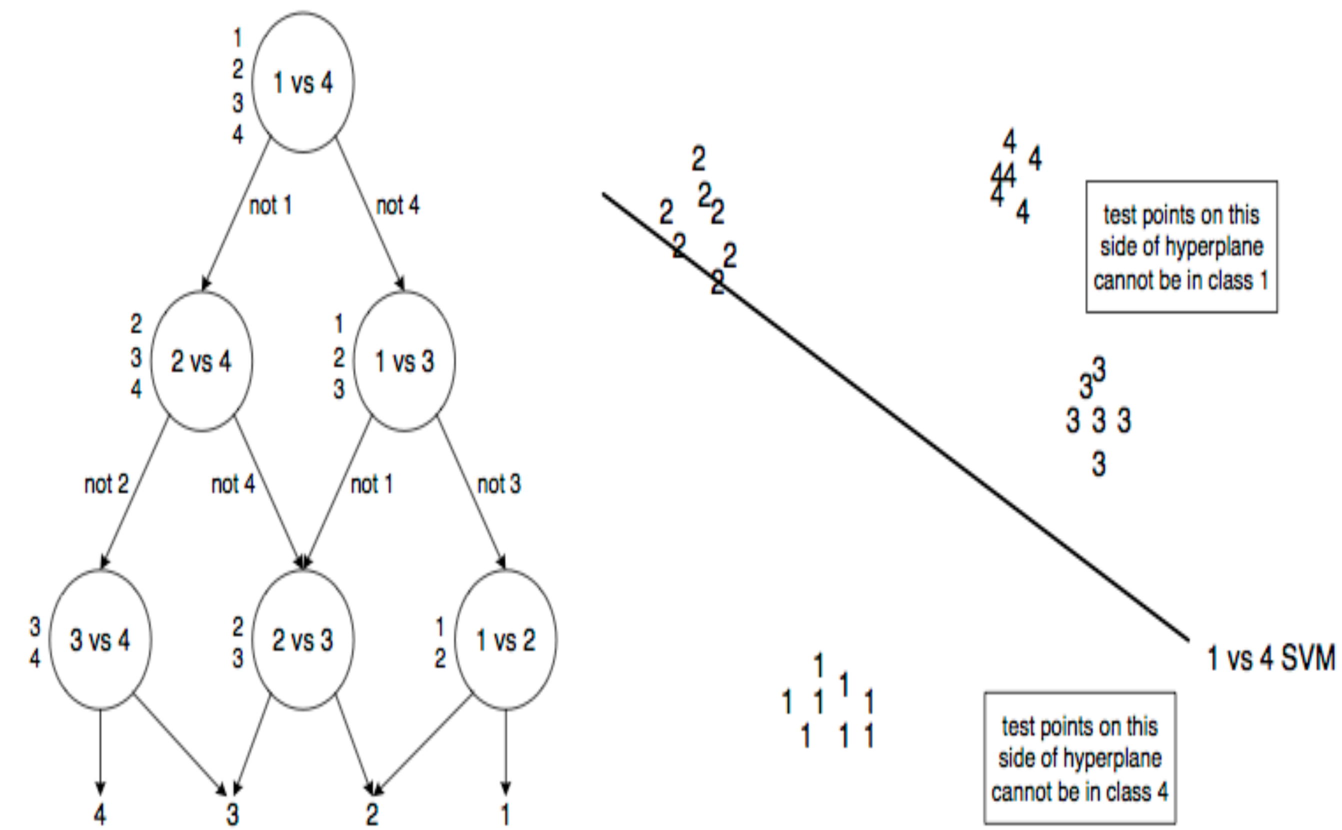
Directed - Acyclic - Graph (DAG)

Large Margin DAGs for Multiclass Classification

John C. Platt
Microsoft Research
1 Microsoft Way
Redmond, WA 98052
jplatt@microsoft.com

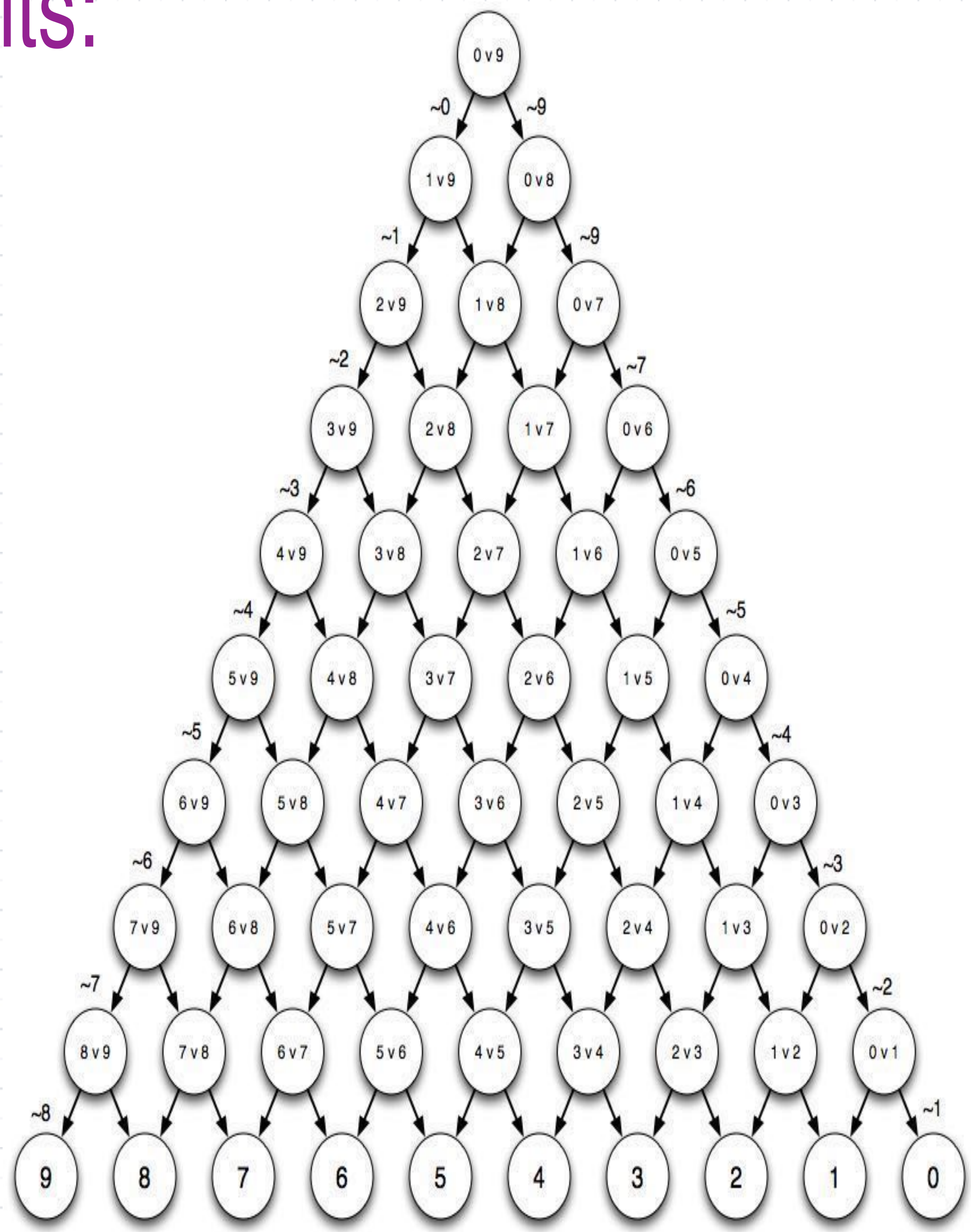
Nello Cristianini
Dept. of Engineering Mathematics
University of Bristol
Bristol, BS8 1TR - UK
nello.cristianini@bristol.ac.uk

John Shawe-Taylor
Department of Computer Science
Royal Holloway College - University of London
EGHAM, Surrey, TW20 0EX - UK
j.shawe-taylor@dcs.rhbnc.ac.uk



(a) (b)

Full DAG for Digits:



Steps to create Classifiers for Digit Recognition

1.) Train the 45 Binary Classifiers (so 45 different a's and b's)

* This will take the most time!
* Use the 60,000 training images images and their labels

3.) Test your Classifier via the DAG

* Use the 10,000 testing images

4.) Measure accuracy (total error for example) and go back to 2.)

```
%%%%%%% Pull out the zeros and ones from the DATA
Zeros = images(Zeros_Labels,:);
Nines = images(Nines_Labels,:);

R = length(Zeros(:,1));
Q = length(Nines(:,1));
g = .1;
n = 784;

cvx_solver sedumi
%%%%%%% Solve for the support vector
cvx_begin
    cvx_precision low % speed up solver
    variables a(n) b(1) u(R) v(Q)
    minimize (norm(a) + g*(ones(1,R)*u + ones(1,Q)*v))
    double(Zeros)*a - b >= 1 - u;
    double(Nines)*a - b <= -(1 - v);
    u >= 0;
    v >= 0;
cvx_end
toc
%%%%%%% First step of the DAG
Not9 = find(double(images_test)*a - b>0);
Not0 = find(double(images_test)*a - b<0);
```

Results:

```
Percent_Error_Total =

    0.9449

ans =

    0.9718
    0.9834
    0.9513
    0.9151
    0.9298
    0.8975
    0.9515
    0.9508
    0.9404
    0.9489
```

Mistaken as 0:

Mistaken as 9: