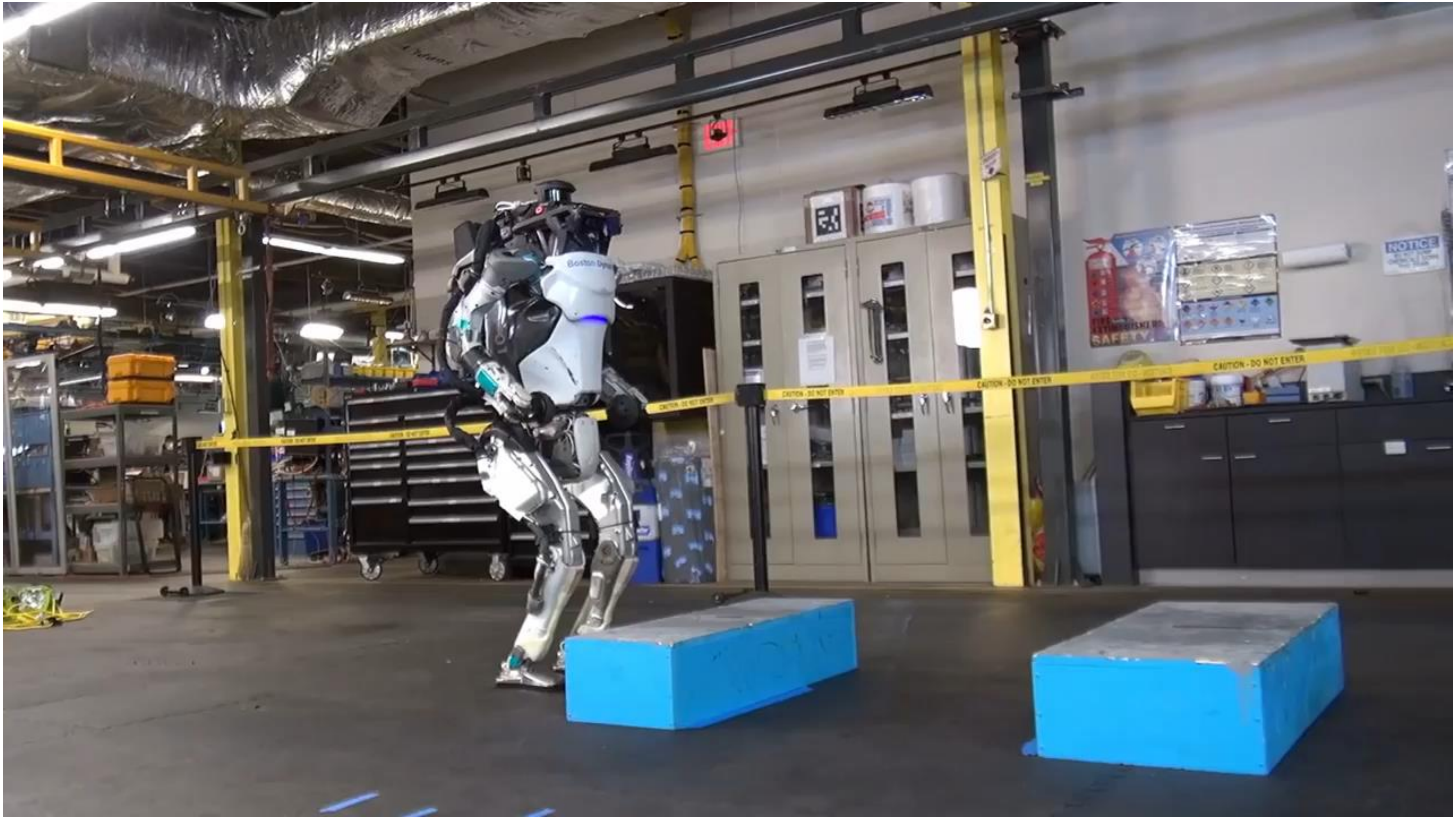
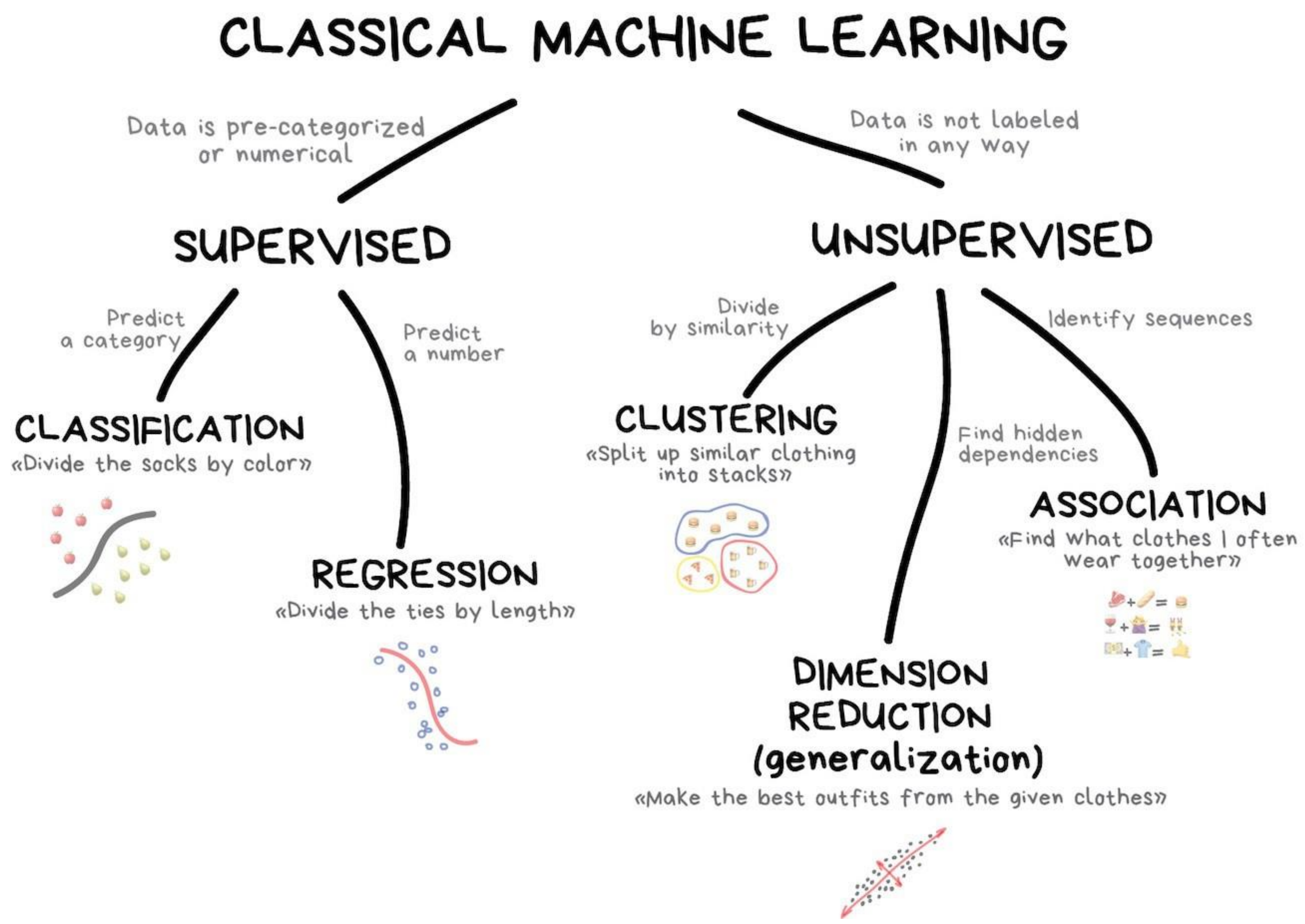
What is Machine Learning?

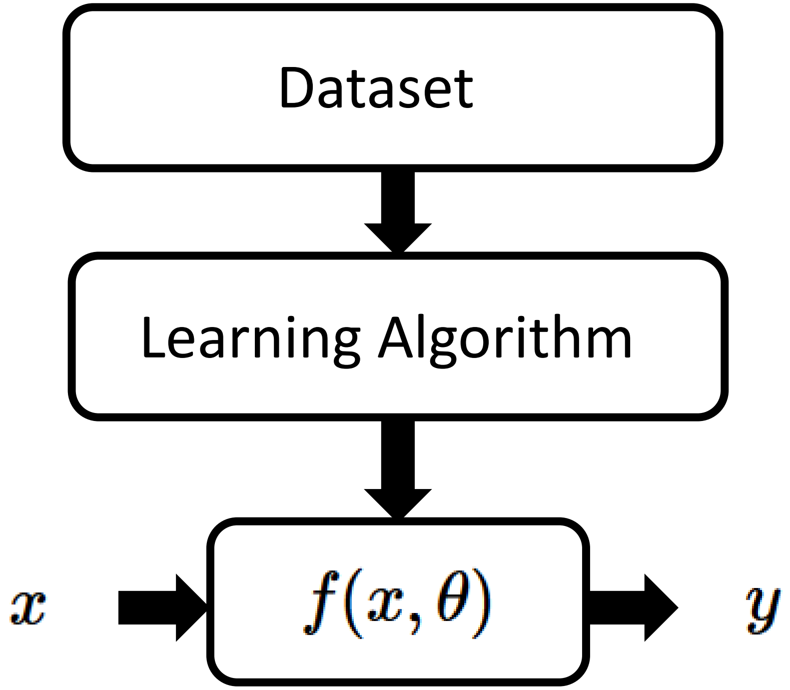

CAUTION - DO NOT ENTER
NOTICE


Boston Dynamics


GSMArena.com



https://vas3k.com/blog/machine_learning


Dataset
Learning Algorithm
x
$f(x, \theta)$
y

Images blocked | Show images | Don't block this sender | Turn off image blocking

This is incredibly important - PLEASE READ

aaabeline@aol.com to you show details May 2

IPad_Image.jpg (36 KB)

Open this now. Hi, This is incredibly important, so PLEASE READ THIS EMAIL IN IT'S ENTIRETY

Check it here now
 A stay at home wor dpress programmer discoved a major Google loophole using a little know technique and built a plugin to automate the method. </p> This technique replaces the ori ginal search box with a Google "Adsense for Search" Custom Search box quickly and easily using a simple plugin he developed. As with most Wordpress themes, you'd have to change this manually, which would require some php programming... But now you can do it with the simple "push of a button" by adding the Google Adsense Accelerat or plugin to your wordpress site. NO PROGRAMMING...

~~>>http://www.jvzoo.com/c/113414/ 3678

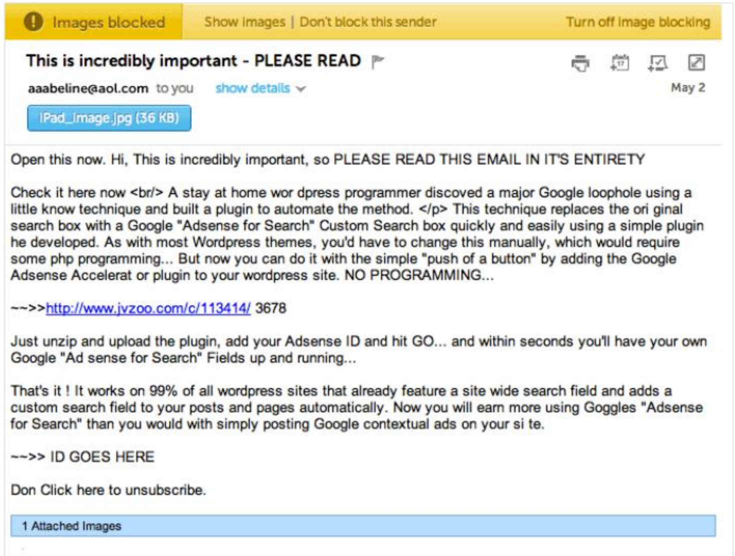
Just unzip and upload the plugin, add your Adsense ID and hit GO... and within seconds you'll have your own Google "Ad sense for Search" Fields up and running...

That's it ! It works on 99% of all wordpress sites that already feature a site wide search field and adds a custom search field to your posts and pages automatically. Now you will earn more using Goggles "Adsense for Search" than you would with simply posting Google contextual ads on your si te.

~~>> ID GOES HERE

Don Click here to unsubscribe.

1 Attached Images


Images blocked | Show images | Don't block this sender | Turn off image blocking

This is incredibly important - PLEASE READ

aaabeline@aol.com to you show details May 2

iPad_image.jpg (36 KB)

Open this now. Hi, This is incredibly important, so PLEASE READ THIS EMAIL IN IT'S ENTIRETY

Check it here now
 A stay at home wor dpress programmer discoved a major Google loophole using a little know technique and built a plugin to automate the method. </p> This technique replaces the ori ginal search box with a Google "Adsense for Search" Custom Search box quickly and easily using a simple plugin he developed. As with most Wordpress themes, you'd have to change this manually, which would require some php programming... But now you can do it with the simple "push of a button" by adding the Google Adsense Accelerat or plugin to your wordpress site. NO PROGRAMMING...

~~>>http://www.jvzoo.com/c/113414/ 3678

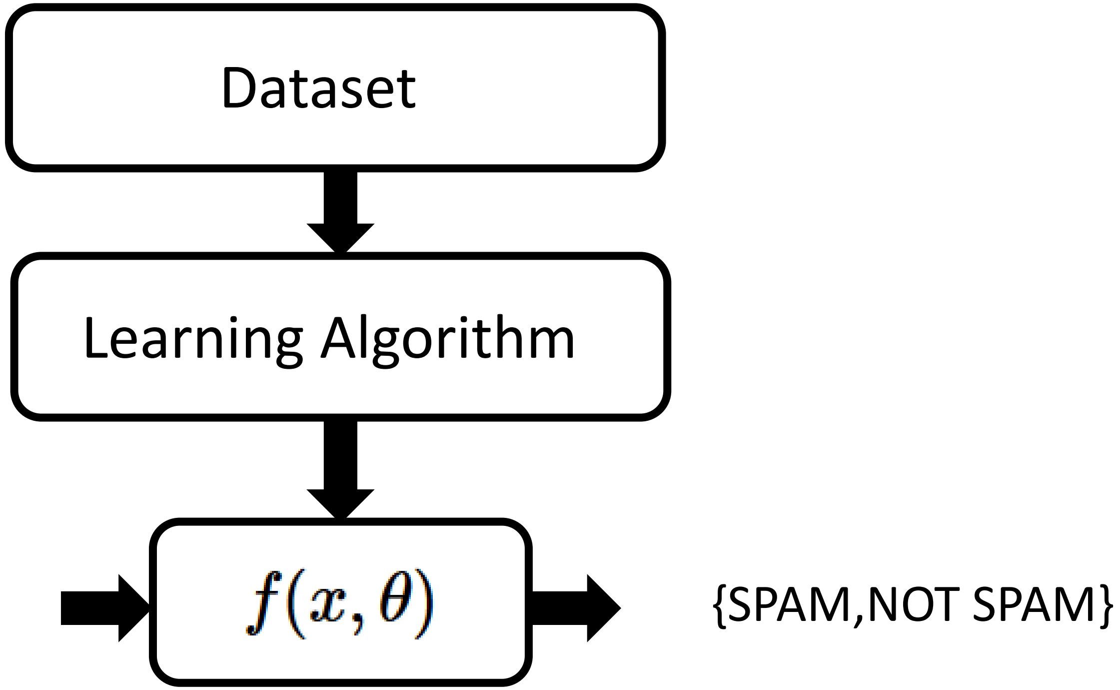
Just unzip and upload the plugin, add your Adsense ID and hit GO... and within seconds you'll have your own Google "Ad sense for Search" Fields up and running...

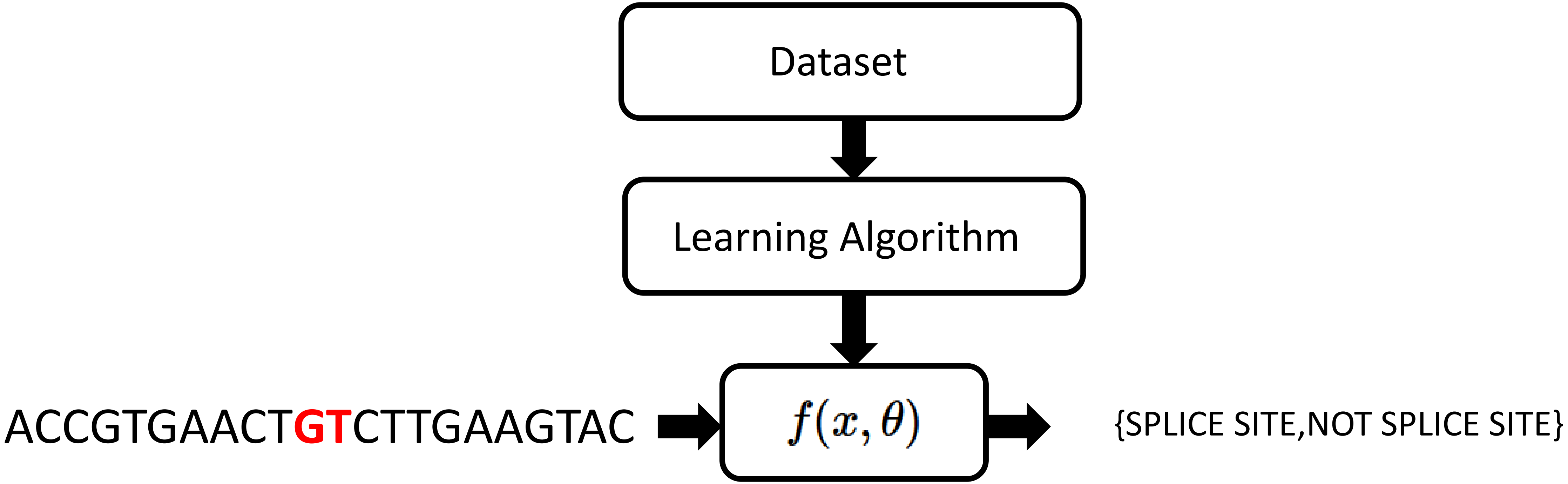
That's it ! It works on 99% of all wordpress sites that already feature a site wide search field and adds a custom search field to your posts and pages automatically. Now you will earn more using Goggles "Adsense for Search" than you would with simply posting Google contextual ads on your si te.

~~>> ID GOES HERE

Don Click here to unsubscribe.

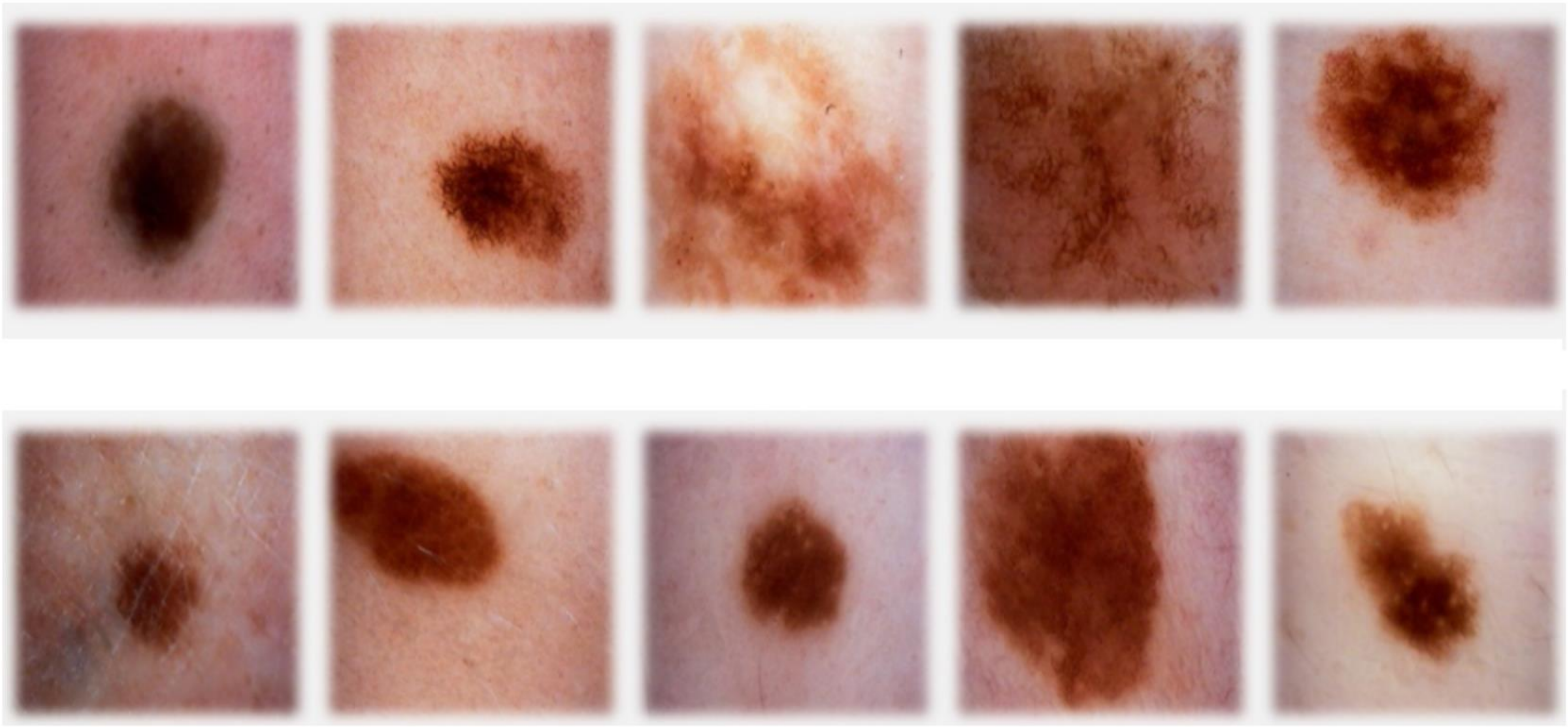
1 Attached Images



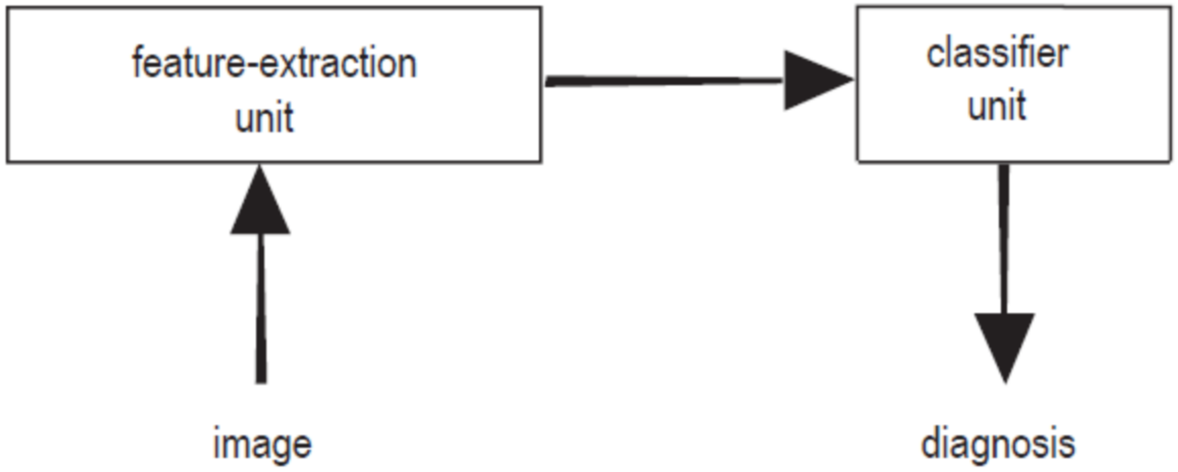


Dataset
Learning Algorithm
ACCGTGAACTGTCTTGAAGTAC
$f(x, \theta)$
{SPLICE SITE,NOT SPLICE SITE}

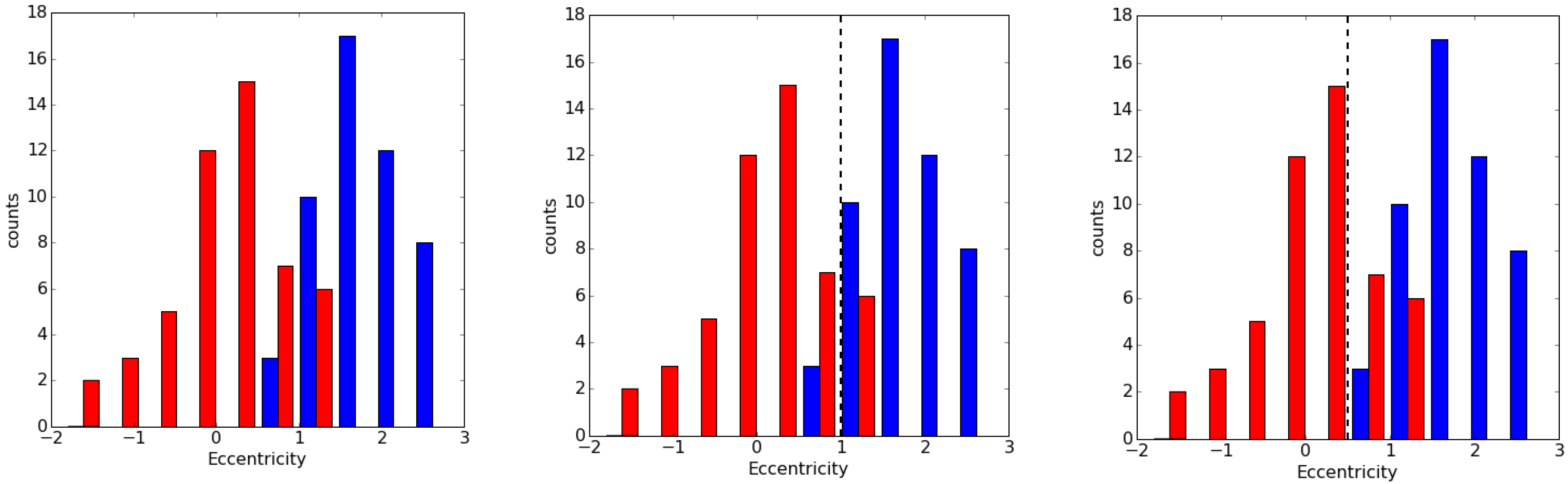
concepts



- sign of cancer
- top row malignant
- bottom row benign

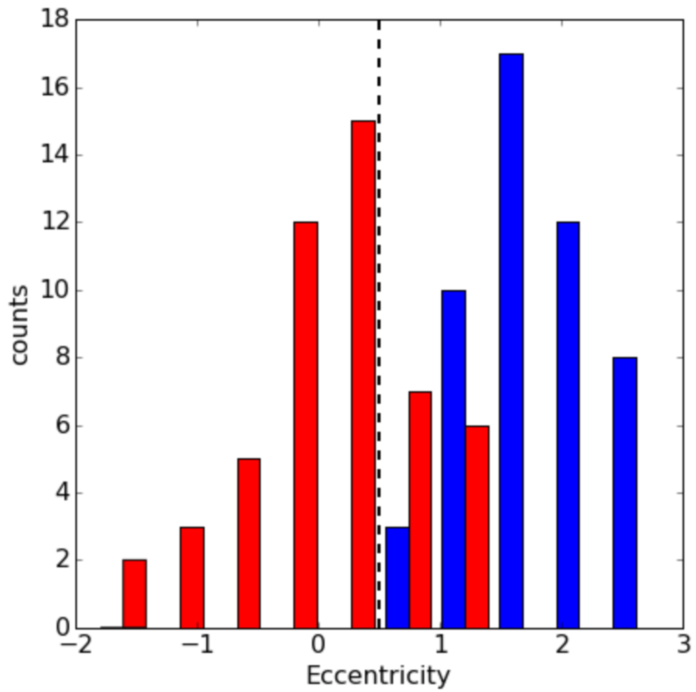


- feature extraction: features (a.k.a. properties or attributes)
- data set, sample (a.k.a. example, instance or data point), label (a.k.a. target)



- eccentricity of lesion (how nearly circular the lesion is)
- threshold ***t***
- consequence of the predictions

- malignant positive class, benign negative
- consequence of the predictions

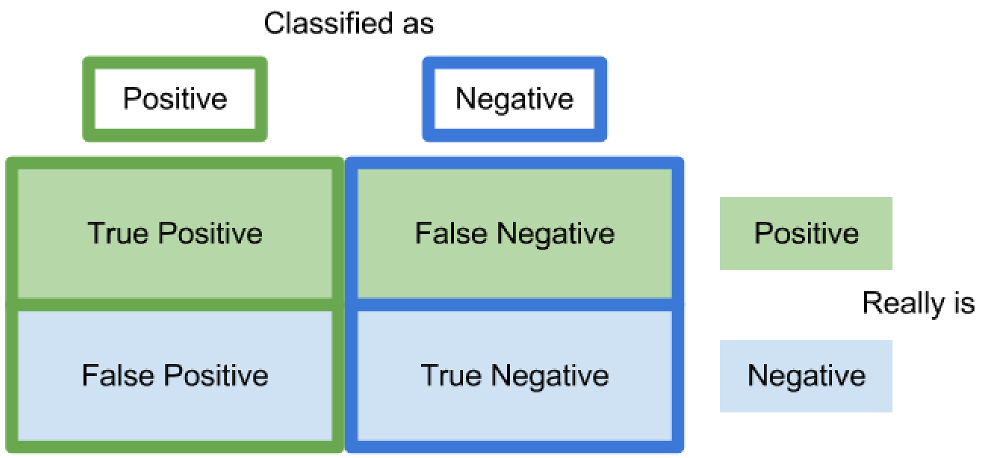


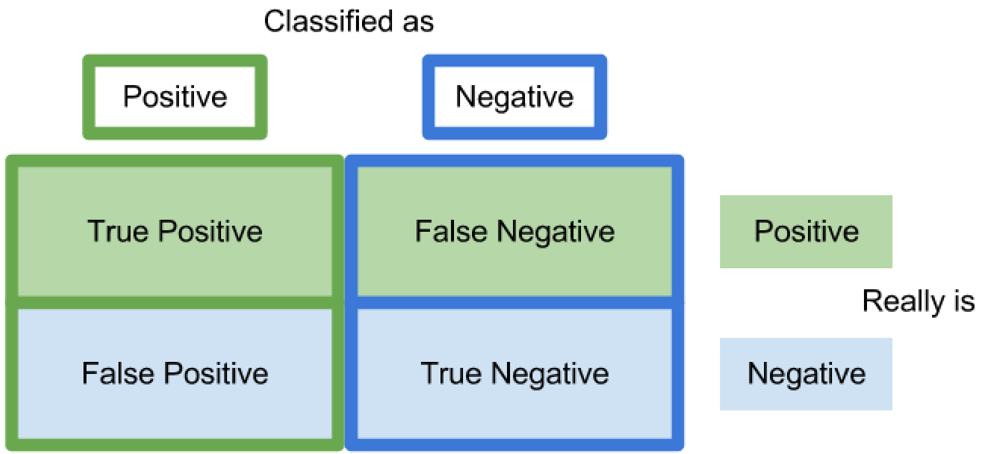
count the number of malignant images with eccentricity value $\geq t$: **true positive** predictions (TP)

count the number of malignant images with eccentricity value $< t$: **false negative** predictions (FN)

count the number of benign images with eccentricity value $\geq t$: **false positive** predictions (FP)

count the number of benign images with eccentricity value $< t$: **true negative** predictions (TN)



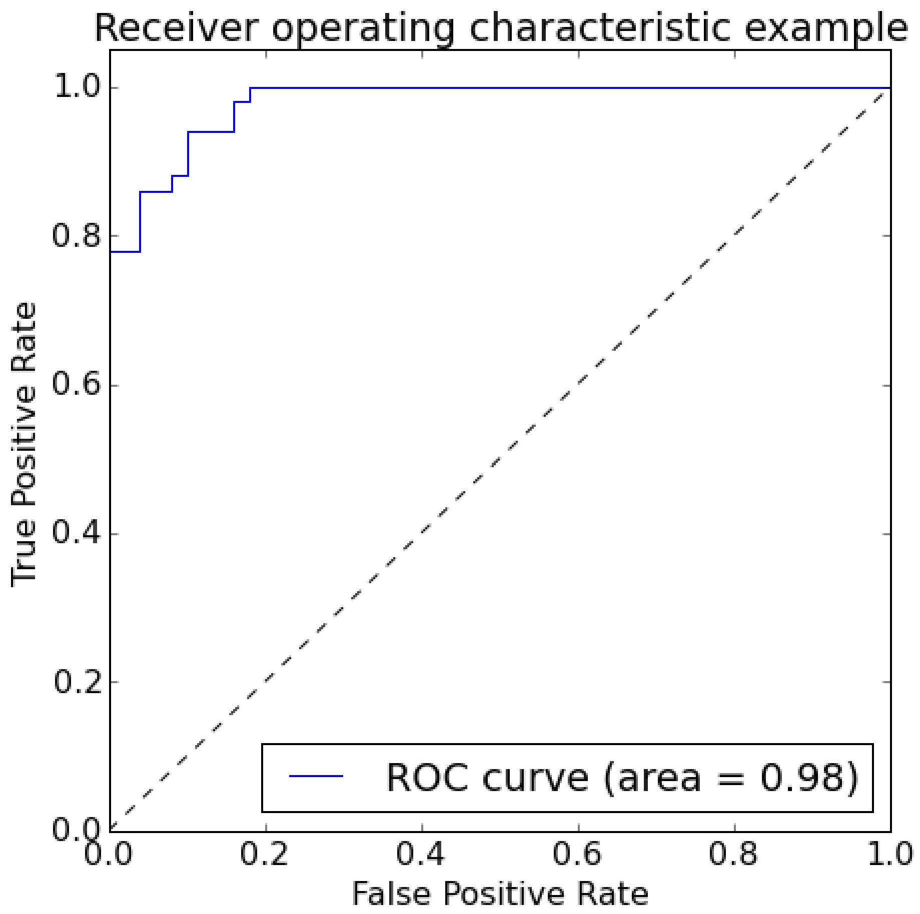


$$\text{accuracy} = \frac{TP + TN}{TP + FP + TN + FN}$$

$$\text{TPR} = \frac{TP}{TP + FN}$$

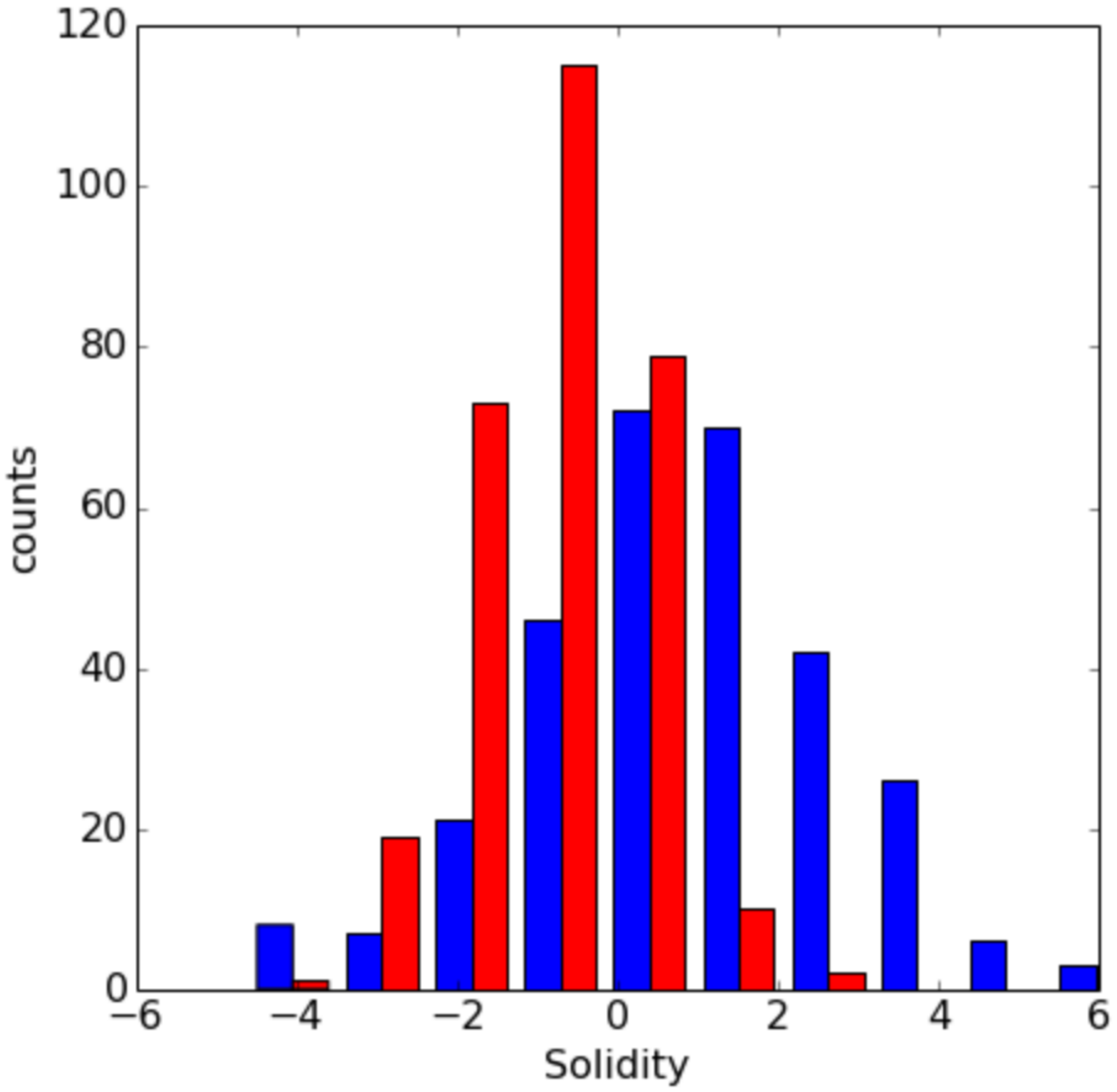
$$\text{FPR} = \frac{FP}{FP + TN}$$

$$\mathrm{TPR} = \frac{TP}{TP + FN}$$

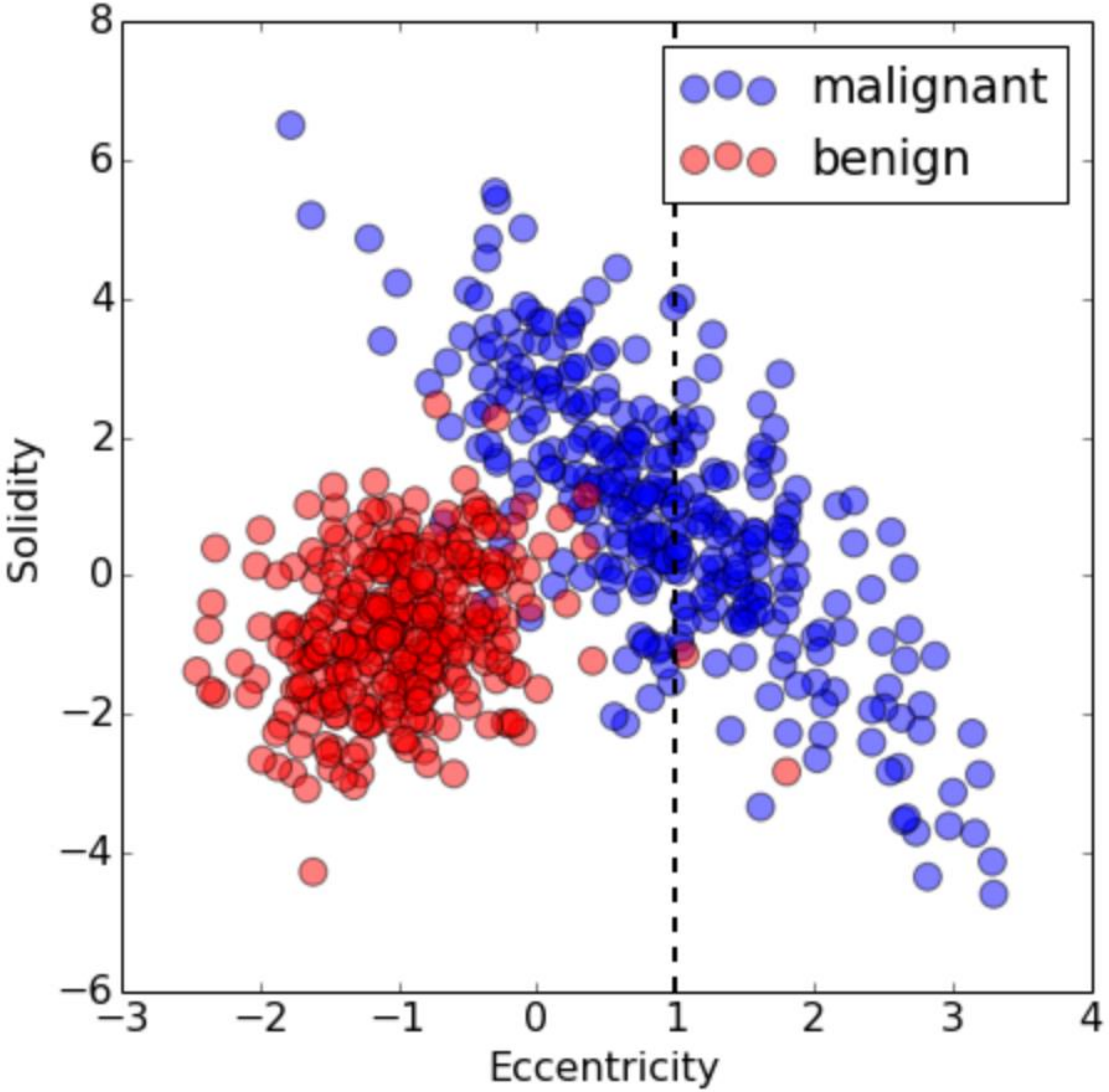


$$\mathrm{FPR} = \frac{FP}{FP + TN}$$

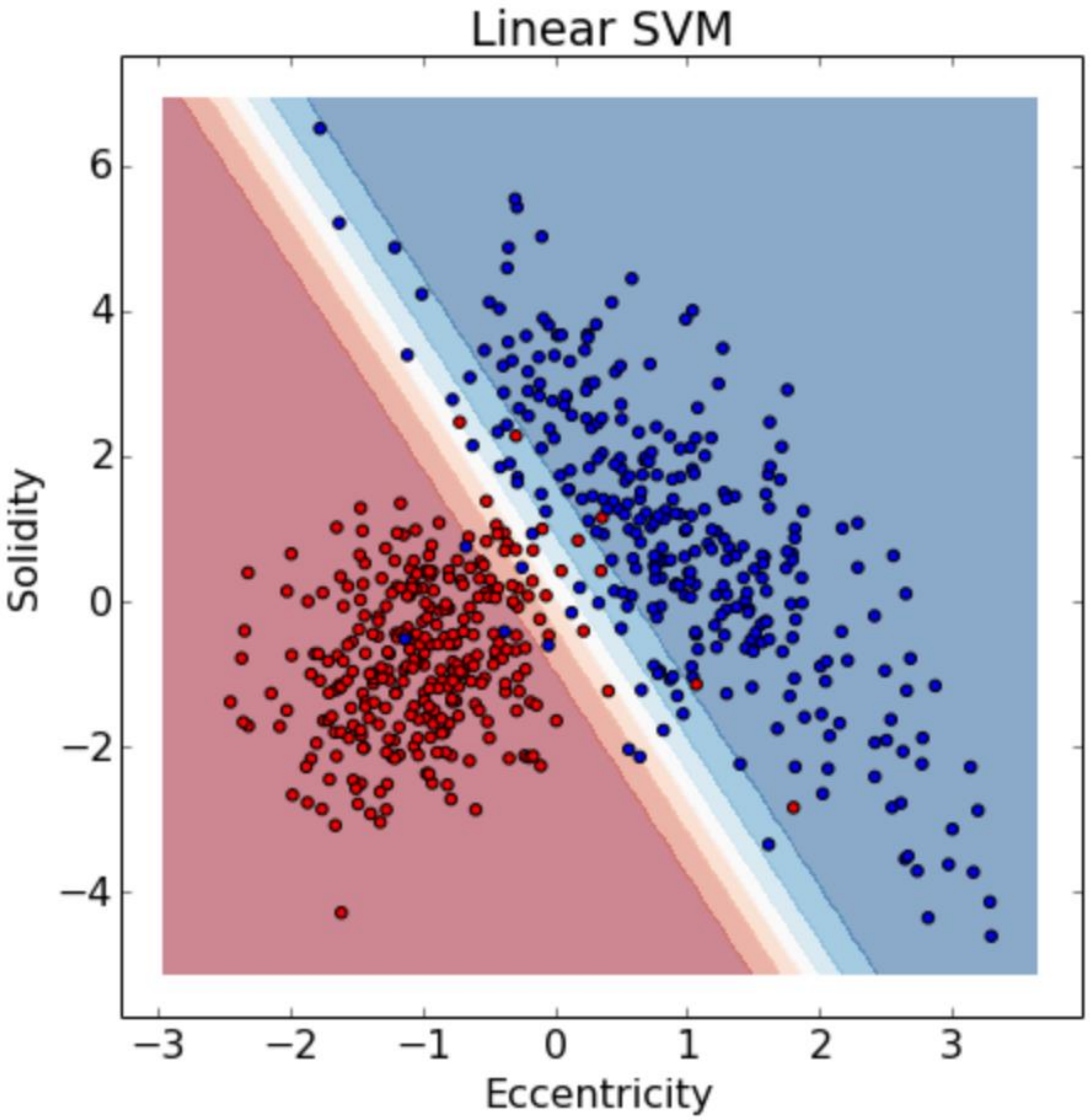
- model that classifies all images as malignant: TPR=1 and FPR=1
- model that classifies all images a benign: TPR=0 and FPR=0
- vary threshold *t*
- *AUC*



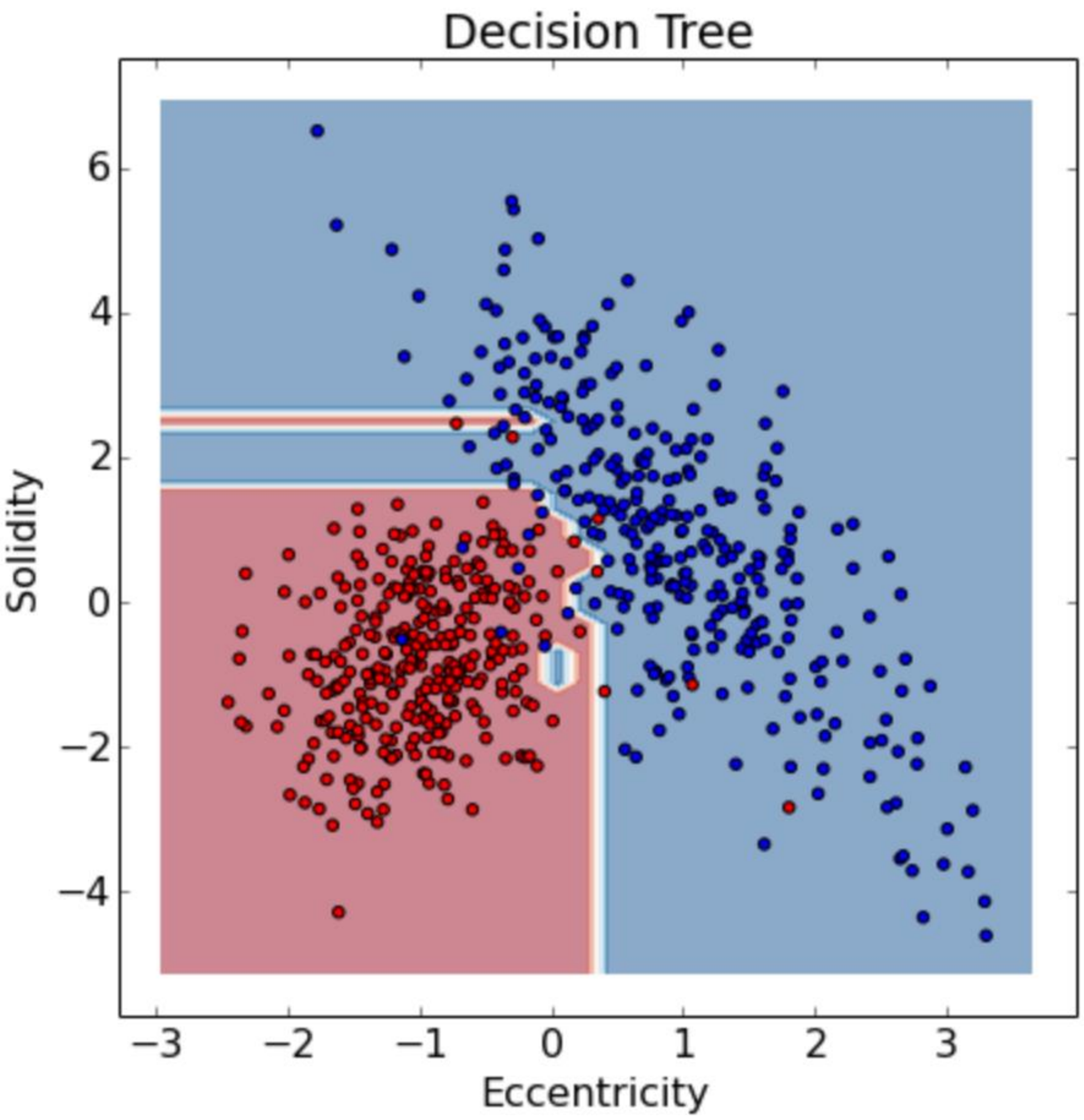
- add another feature?
- feature vector X
- Euclidean vector space



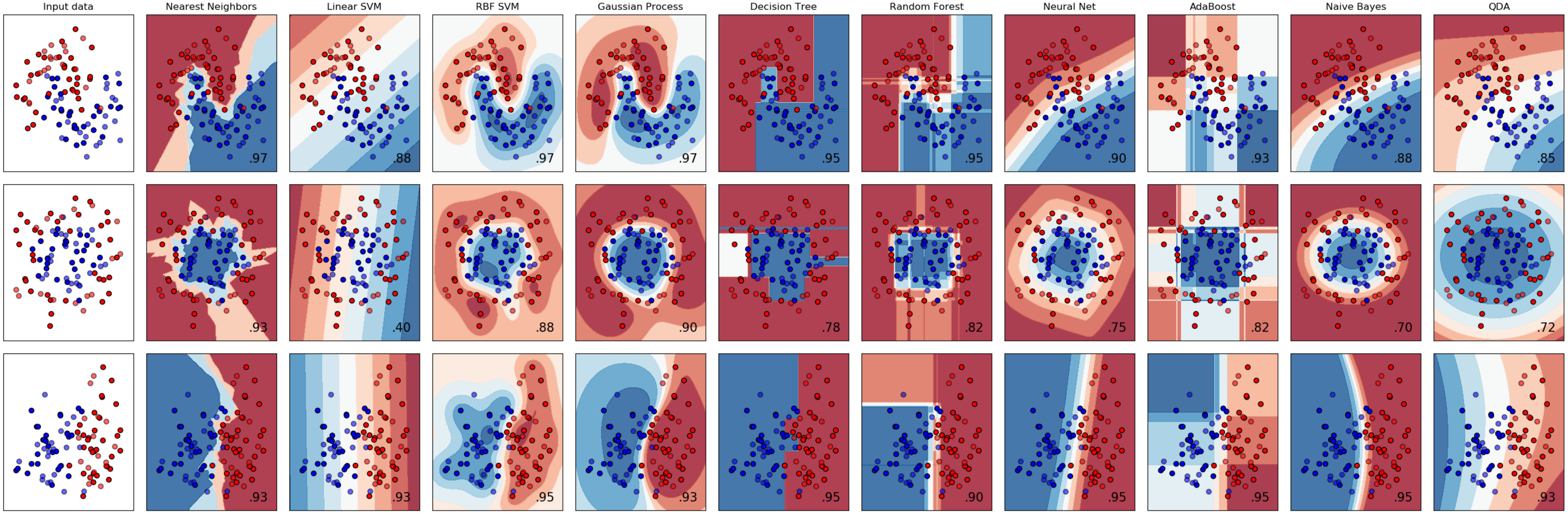
- feature vector X
- Euclidean vector space



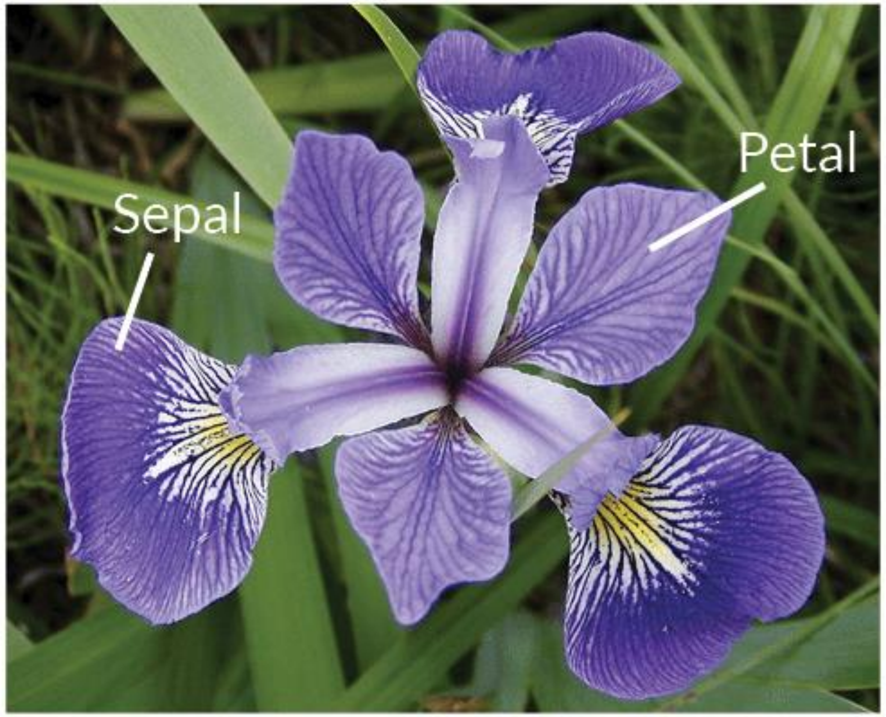
- linear decision boundary
- blue region malignant, red region benign
- yet more features
- can’t look at the decision boundary
- more complex



- unseen external images
- generalization
- overfitting



Input data
Nearest Neighbors
Linear SVM
RBF SVM
Gaussian Process
Decision Tree
Random Forest
Neural Net
AdaBoost
Naive Bayes
QDA
.97
.88
.97
.97
.95
.95
.90
.93
.88
.85
.93
.40
.88
.90
.78
.82
.75
.82
.70
.72
.93
.93
.95
.93
.95
.90
.95
.95
.95
.93



Iris Versicolor

Iris Setosa

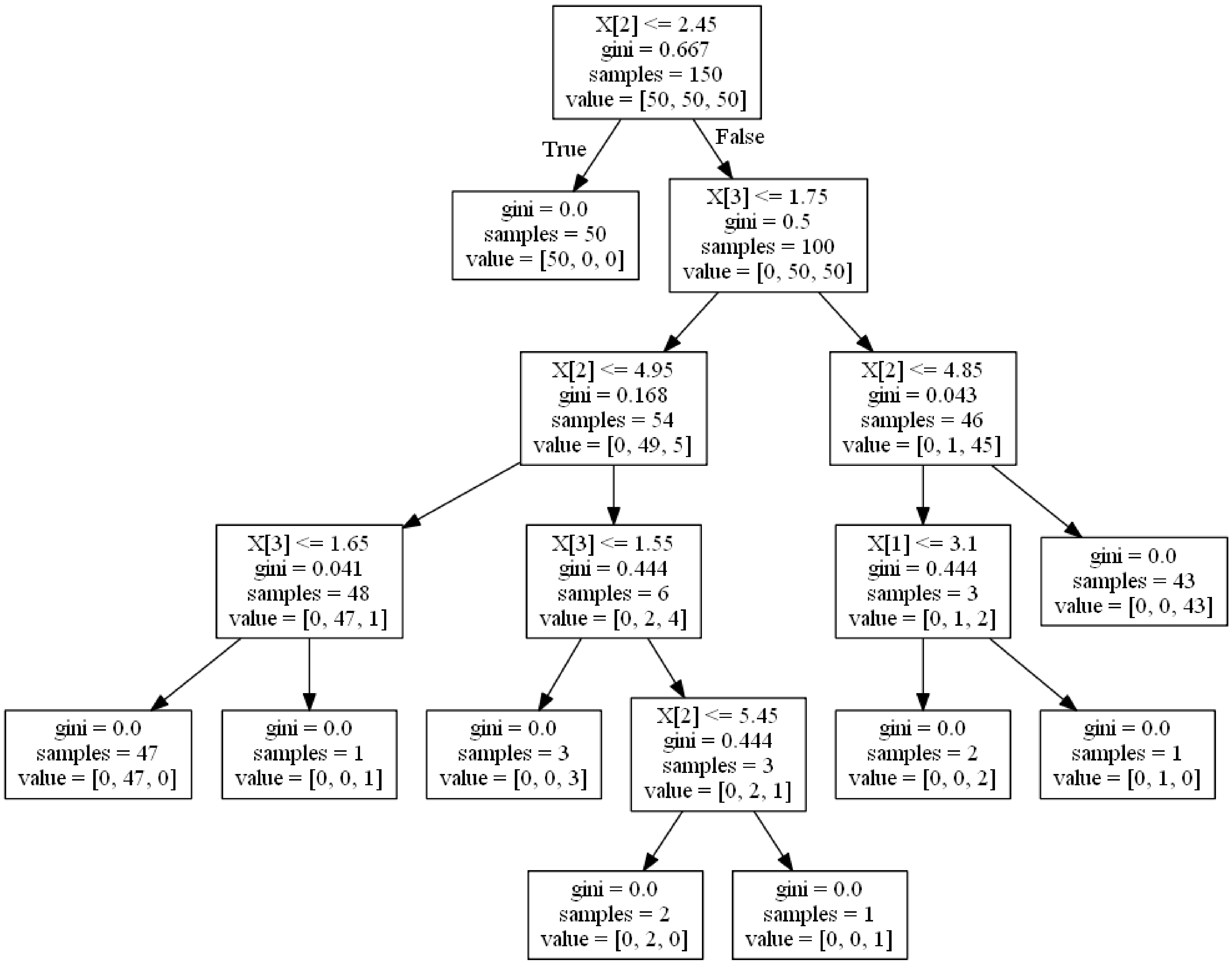
Iris Virginica

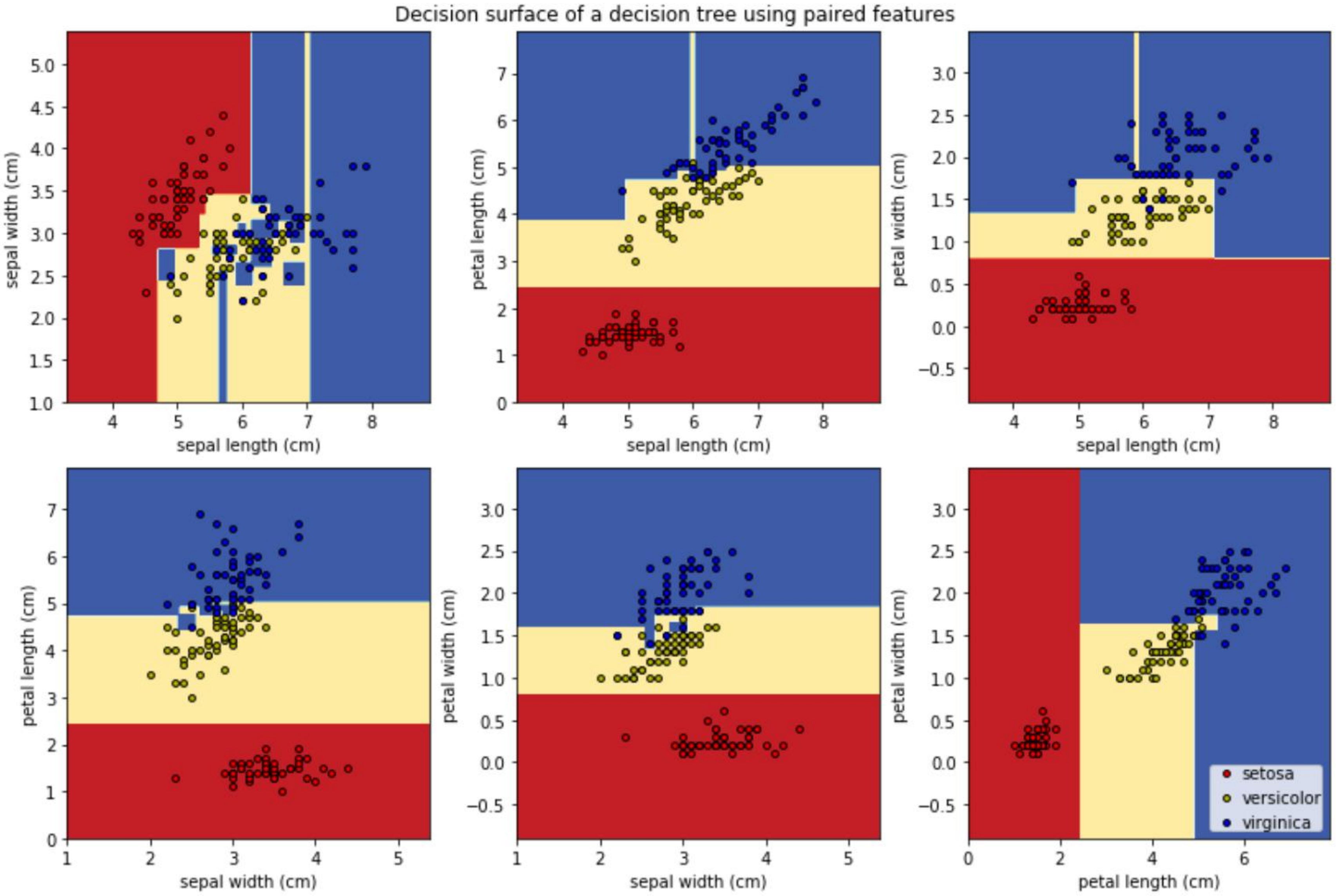
Table I

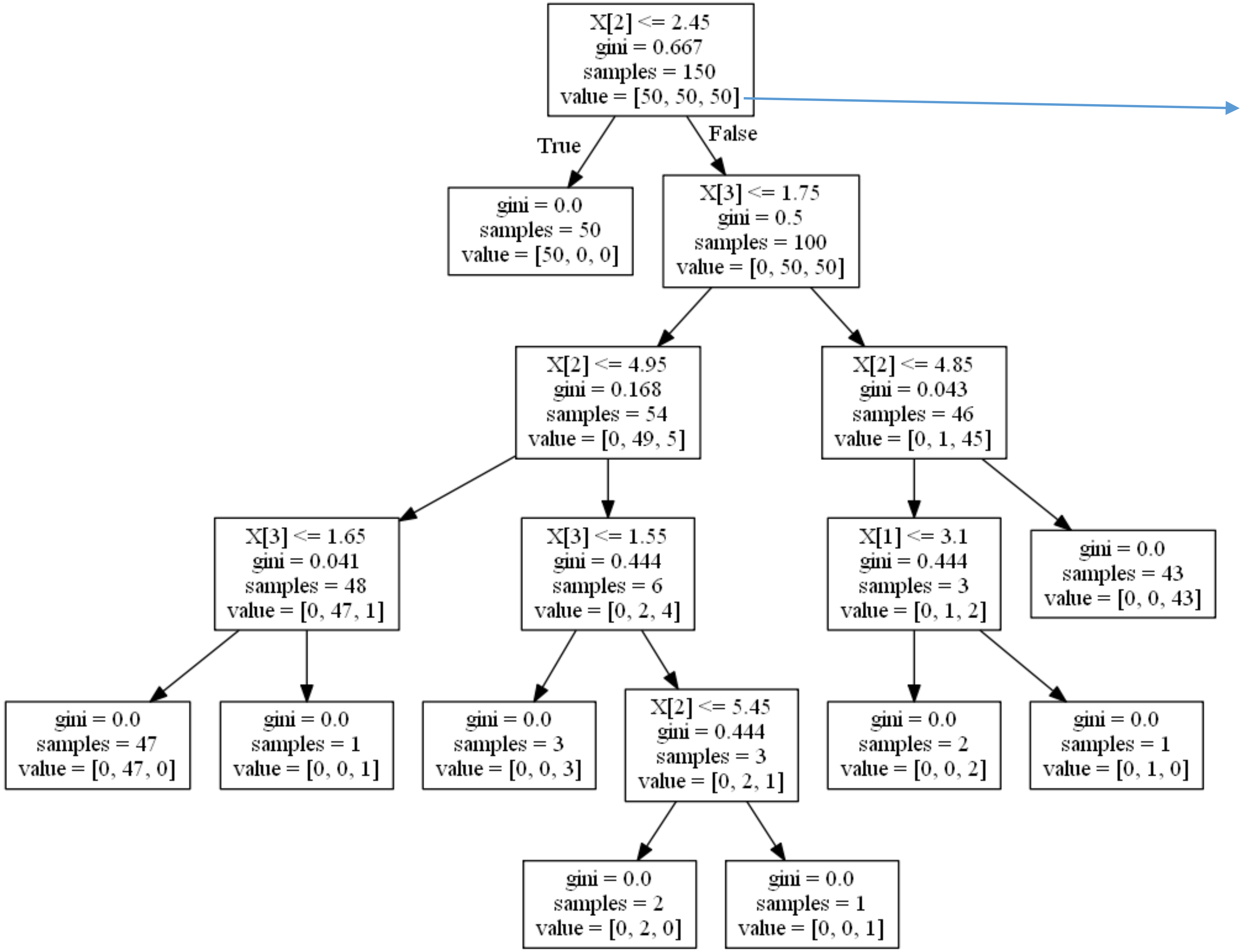
Iris setosa				Iris versicolor				Iris virginica			
Sepal length	Sepal width	Petal length	Petal width	Sepal length	Sepal width	Petal length	Petal width	Sepal length	Sepal width	Petal length	Petal width
5·1	3·5	1·4	0·2	7·0	3·2	4·7	1·4	6·3	3·3	6·0	2·5
4·9	3·0	1·4	0·2	6·4	3·2	4·5	1·5	5·8	2·7	5·1	1·9
4·7	3·2	1·3	0·2	6·9	3·1	4·9	1·5	7·1	3·0	5·9	2·1
4·6	3·1	1·5	0·2	5·5	2·3	4·0	1·3	6·3	2·9	5·6	1·8
5·0	3·6	1·4	0·2	6·5	2·8	4·6	1·5	6·5	3·0	5·8	2·2
5·4	3·9	1·7	0·4	5·7	2·8	4·5	1·3	7·6	3·0	6·6	2·1
4·6	3·4	1·4	0·3	6·3	3·3	4·7	1·6	4·9	2·5	4·5	1·7
5·0	3·4	1·5	0·2	4·9	2·4	3·3	1·0	7·3	2·9	6·3	1·8
4·4	2·9	1·4	0·2	6·6	2·9	4·6	1·3	6·7	2·5	5·8	1·8
4·9	3·1	1·5	0·1	5·2	2·7	3·9	1·4	7·2	3·6	6·1	2·5
5·4	3·7	1·5	0·2	5·0	2·0	3·5	1·0	6·5	3·2	5·1	2·0
4·8	3·4	1·6	0·2	5·9	3·0	4·2	1·5	6·4	2·7	5·3	1·9
4·8	3·0	1·4	0·1	6·0	2·2	4·0	1·0	6·8	3·0	5·5	2·1
4·3	3·0	1·1	0·1	6·1	2·9	4·7	1·4	5·7	2·5	5·0	2·0
5·8	4·0	1·2	0·2	5·6	2·9	3·6	1·3	5·8	2·8	5·1	2·4

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

Fisher R., *The use of multiple measurements in taxonomic problems* (1936)


X[2] <= 2.45
gini = 0.667
samples = 150
value = [50, 50, 50]
True
False
gini = 0.0
samples = 50
value = [50, 0, 0]
X[3] <= 1.75
gini = 0.5
samples = 100
value = [0, 50, 50]
X[2] <= 4.95
gini = 0.168
samples = 54
value = [0, 49, 5]
X[2] <= 4.85
gini = 0.043
samples = 46
value = [0, 1, 45]
X[3] <= 1.65
gini = 0.041
samples = 48
value = [0, 47, 1]
X[3] <= 1.55
gini = 0.444
samples = 6
value = [0, 2, 4]
X[1] <= 3.1
gini = 0.444
samples = 3
value = [0, 1, 2]
gini = 0.0
samples = 43
value = [0, 0, 43]
gini = 0.0
samples = 47
value = [0, 47, 0]
gini = 0.0
samples = 1
value = [0, 0, 1]
gini = 0.0
samples = 3
value = [0, 0, 3]
X[2] <= 5.45
gini = 0.444
samples = 3
value = [0, 2, 1]
gini = 0.0
samples = 2
value = [0, 0, 2]
gini = 0.0
samples = 1
value = [0, 1, 0]
gini = 0.0
samples = 2
value = [0, 2, 0]
gini = 0.0
samples = 1
value = [0, 0, 1]

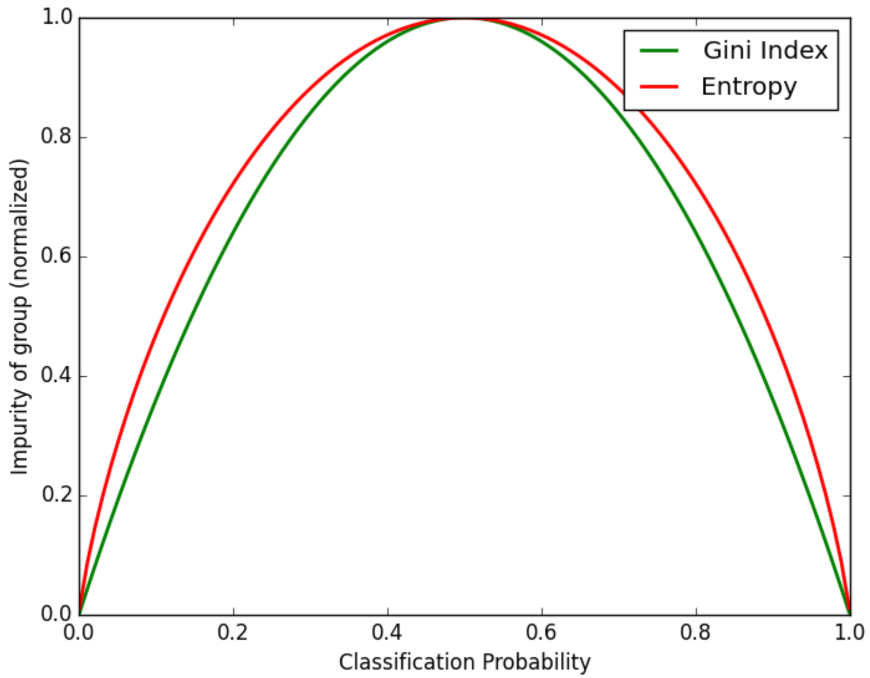

Decision surface of a decision tree using paired features
sepal length (cm)
sepal width (cm)
petal length (cm)
petal width (cm)
setosa
versicolor
virginica



$[\, p_1, p_2, p_3 \,]$

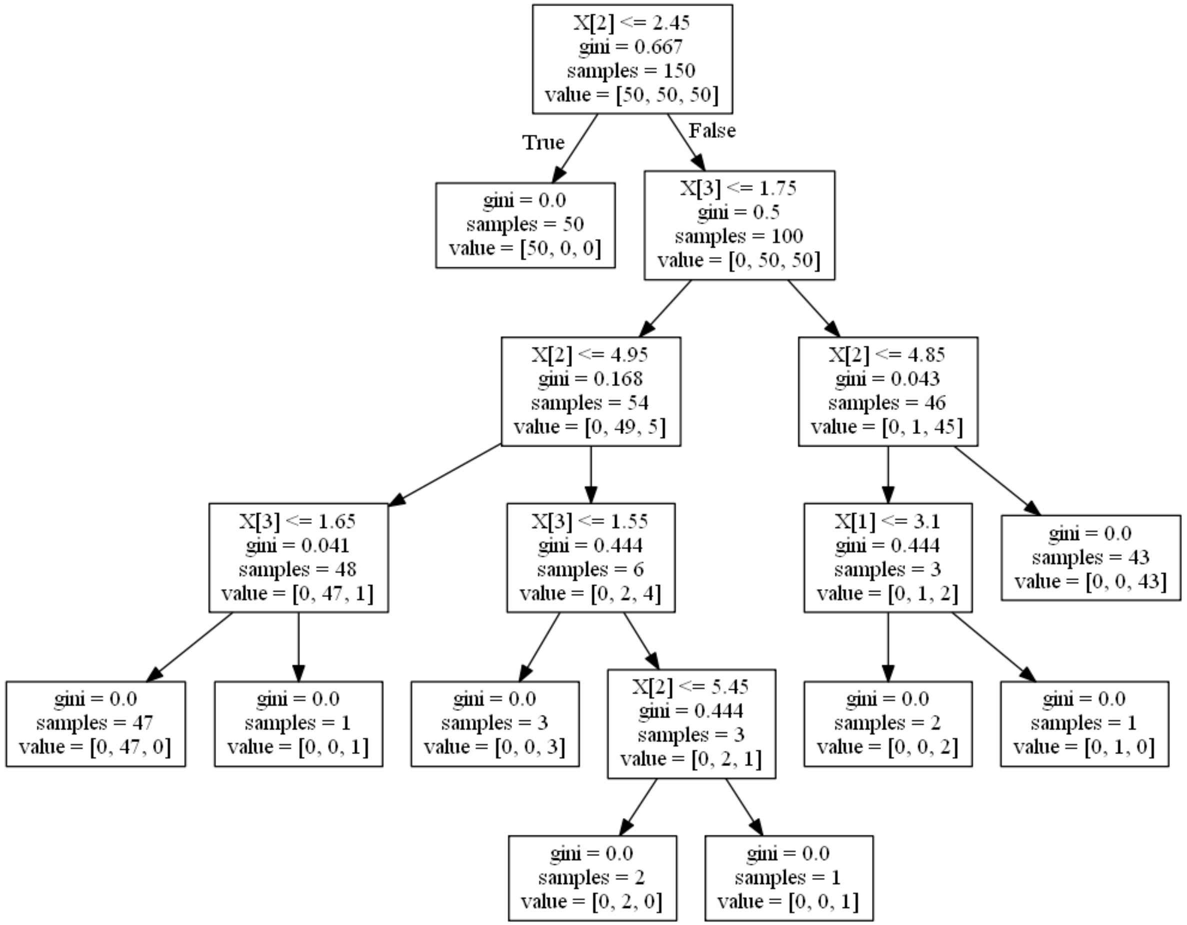
$$Gini: Gini(E) = 1 - \sum_{j=1}^{c} p_j^2$$

$$Entropy: H(E) = -\sum_{j=1}^{c} p_j \log p_j$$



Let’s get real!

1. **Building a decision tree**
2. **Kaggle evaluation**
3. **The test set**
4. **Hyperparameters**

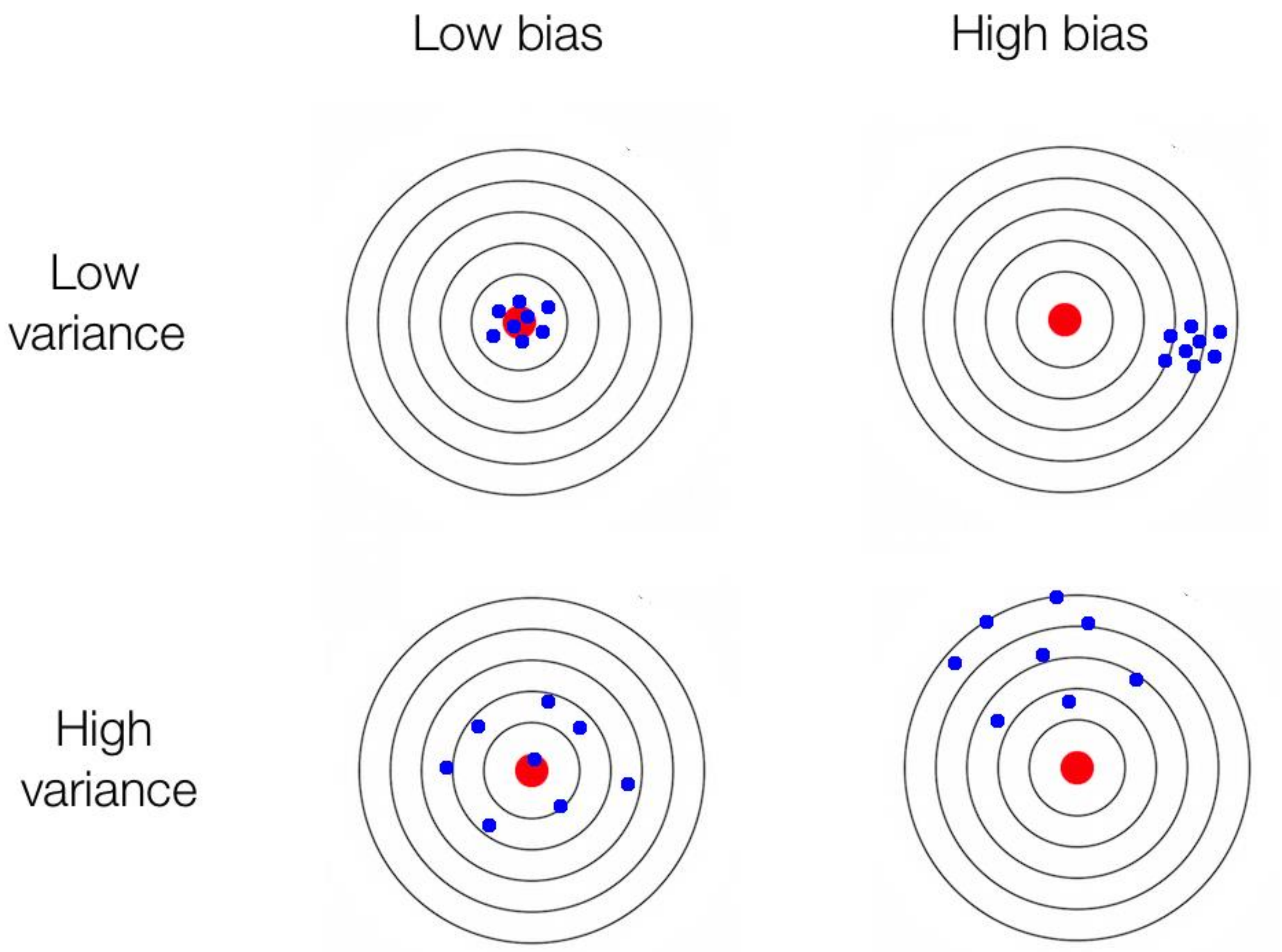


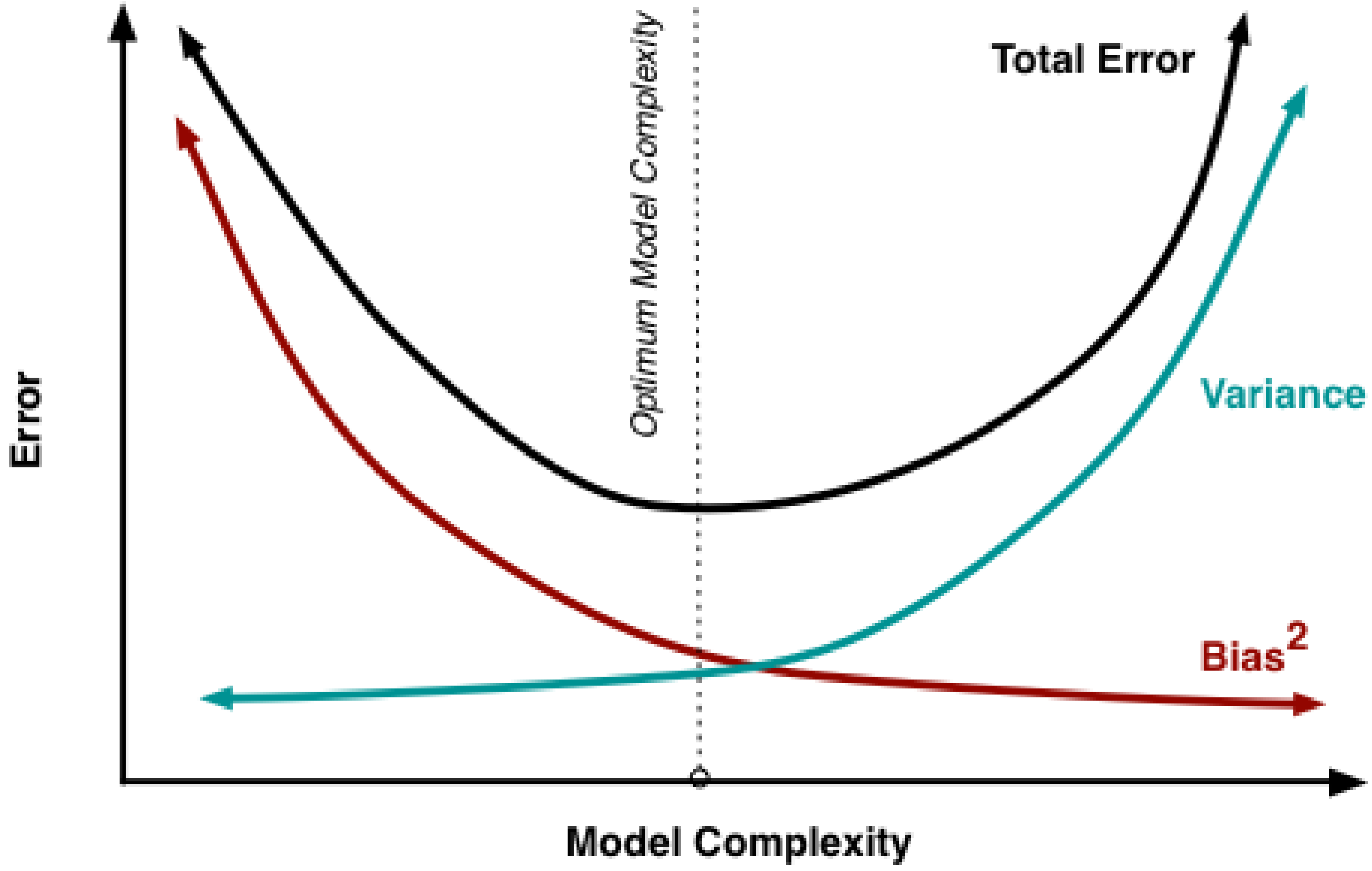
advantages:

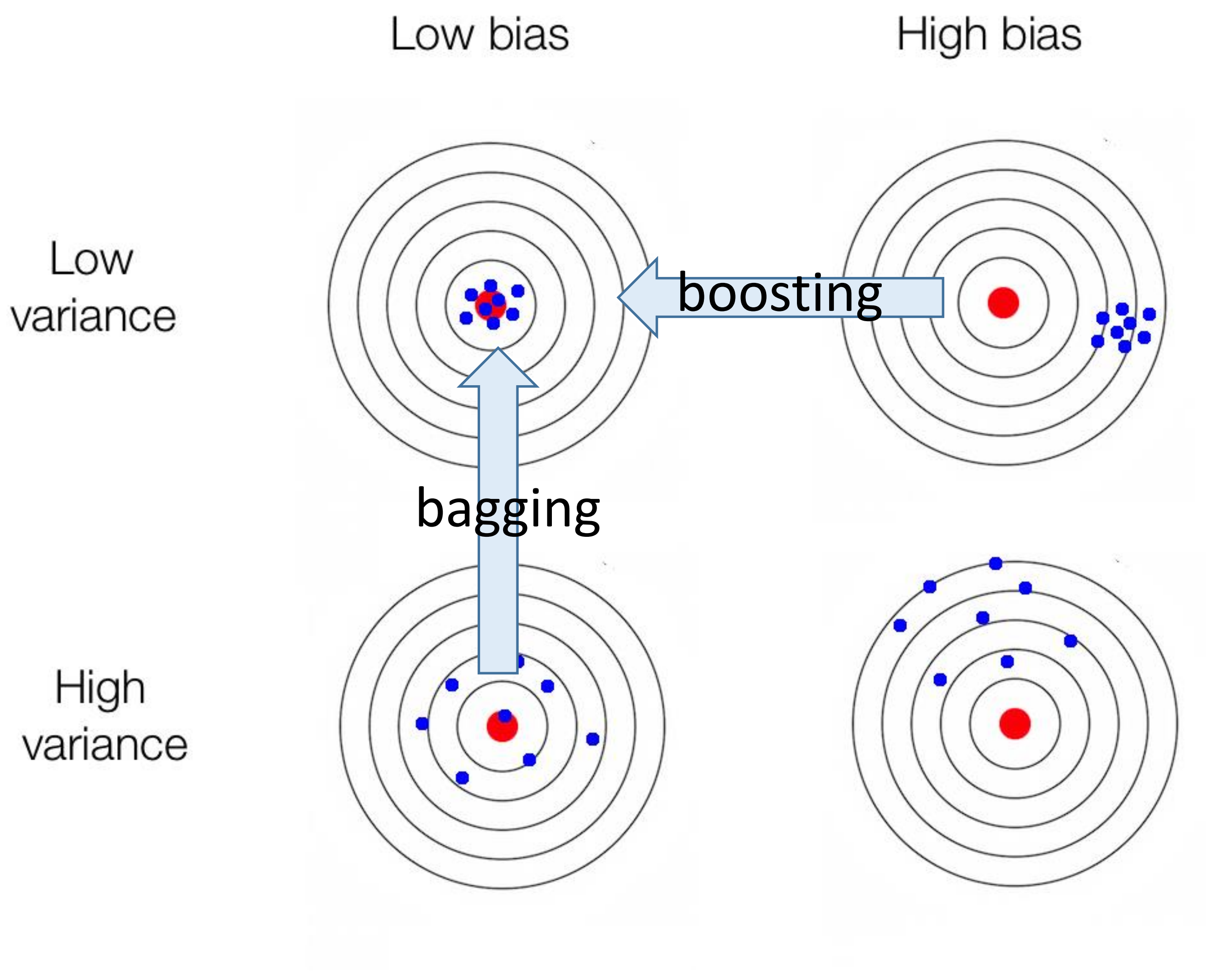
- ease of interpretation
- handles continuous and discrete features
- invariant to monotone transformation of features
- variable selection automated
- **low bias** (deep trees)

disadvantages:

- **high variance**
- overfitting


Low bias
High bias
Low variance
High variance


Total Error
Optimum Model Complexity
Variance
Error
Bias^2
Model Complexity


Low bias
High bias
Low variance
boosting
bagging
High variance

Bagging

- Simulate the notion of different train set samples:

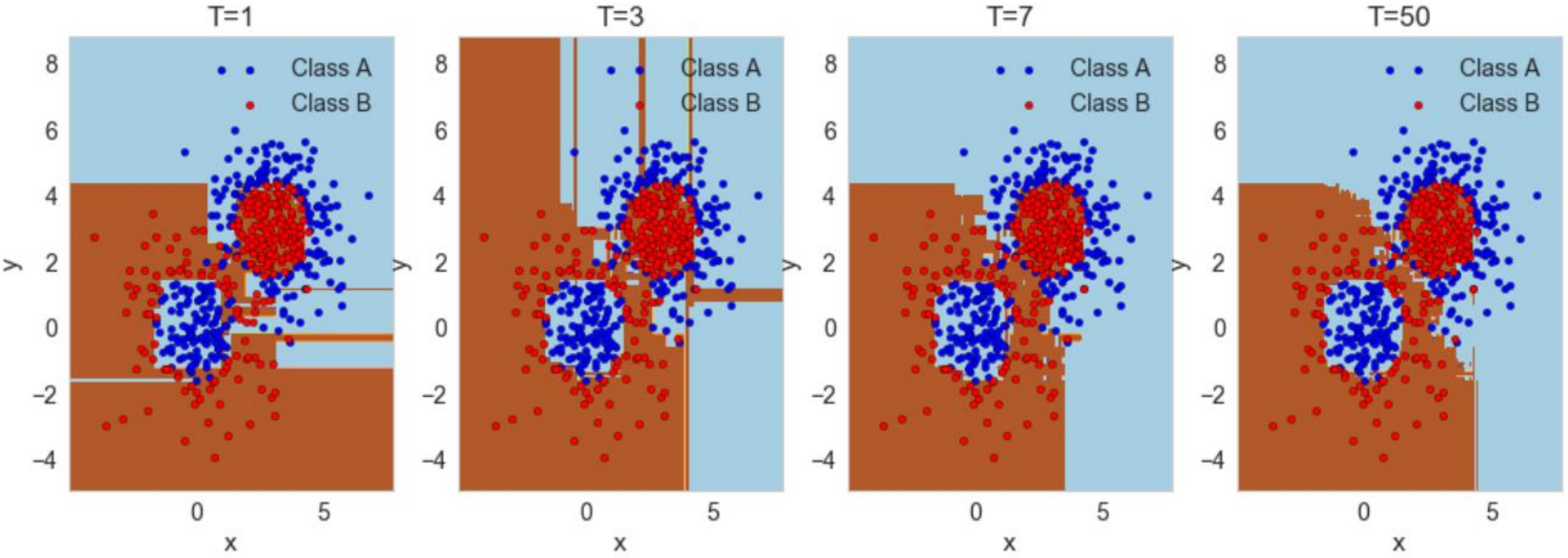
  1. sample data points from train set to create new train set
  2. fit low bias high variance model on new train set
  3. repeat steps (1) and (2) *T* times
  4. average the predictions of the *T* models

$$\hat{f}(x,\theta) = \frac{1}{T}\sum_{t=1}^{T} f_t(x,\theta)$$

Bagging: Random Forests

- Train set contains *n* data points with *m* features.
- Construct *T* low bias high variance decision trees by following these steps:
  1. Sample ***n*** data points at random **with replacement** from the train set.
  2. At each node, select $h \ll m$ features at random and compute the best split using only these *h* features.
  3. Each tree is grown to the largest extent possible. There is no pruning or early stopping.
- Step 3 ensures that the bagged models are low bias by learning deep complex decision trees.

Bagging: Random Forests



Let’s get real!

5. **Ensemble learning: bagging**

Boosting

- reduce the bias of a high bias low variance model
- turning an ensemble of weak learners into a strong learner
- the meta-model is additive, i.e. adaboost:

$$\hat{f}(x,\theta) = \sum_{t=1}^{T} \alpha_t f_t(x,\theta)$$

Boosting: adaboost

$\boxed{y_i \in \{-1, +1\}}$

Initialize the weights $D_1(i) = 1/n, i = 1, 2, \ldots, n$.

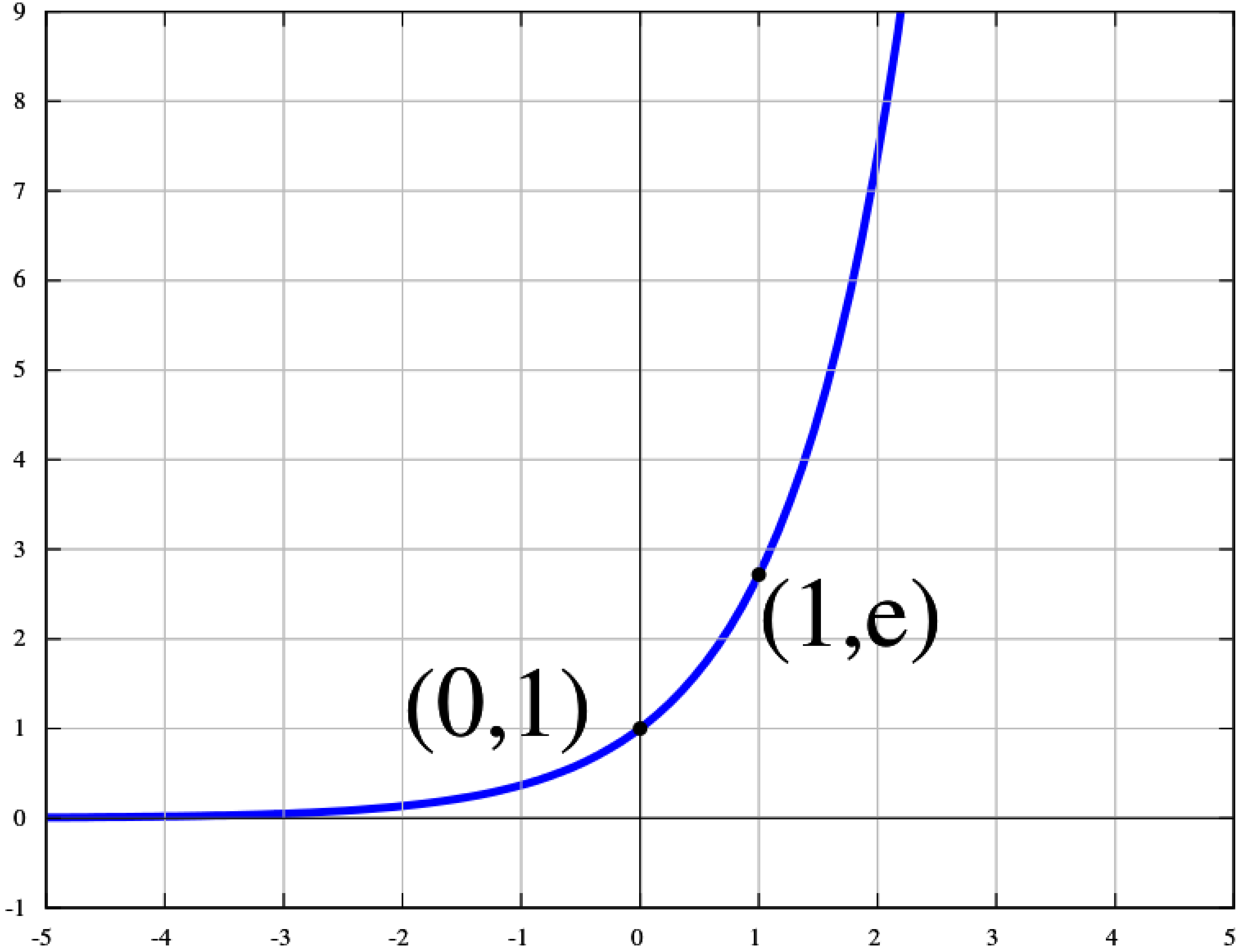
For $t = 1$ to T:

1. Fit a weak classifier $f_t(x, \theta)$ to the trainset data using weights $D_1(i)$.
2. Set $\alpha_t = \frac{1}{2} ln(\frac{1-error}{error})$.
3. Update weights:

$$D_{t+1}(i) = \frac{D_t(i) exp(-\alpha_t y_i f_t(x_i, \theta))}{Z_t},$$

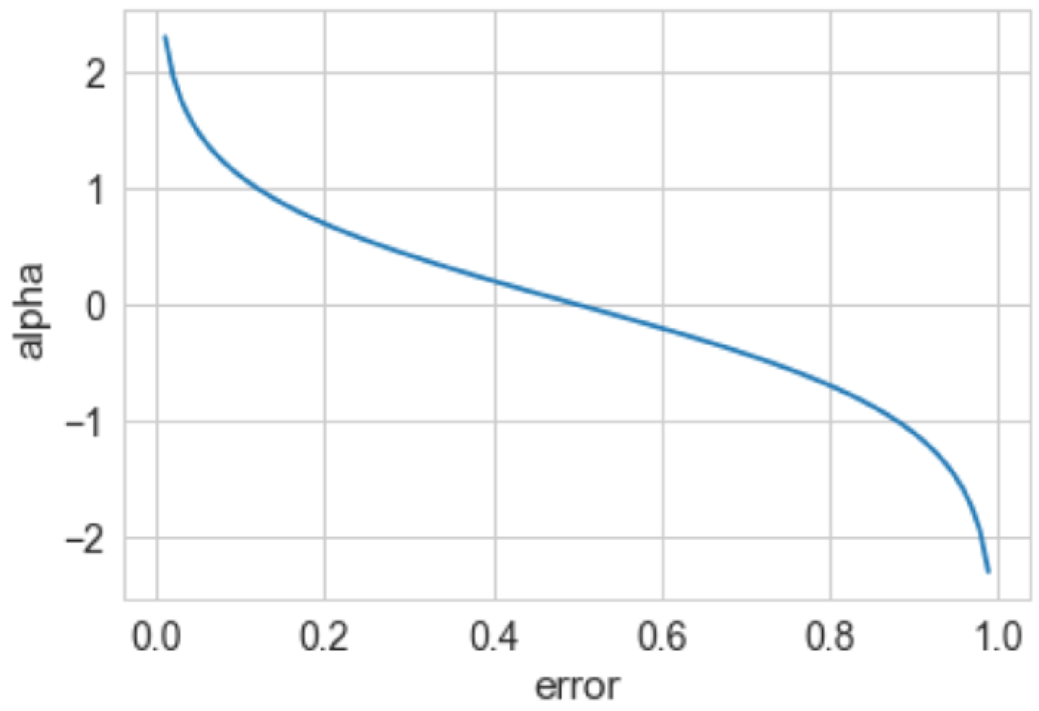
where Z_t is a normalizing factor that makes sure that $\sum D_{t+1}(i) = 1$.

$$\hat{f}(x, \theta) = \sum_{t=1}^{T} \alpha_t f_t(x, \theta)$$



9
8
7
6
5
4
3
2
1
0
-1
-5
-4
-3
-2
-1
0
1
2
3
4
5
(0,1)
(1,e)

Boosting: adaboost



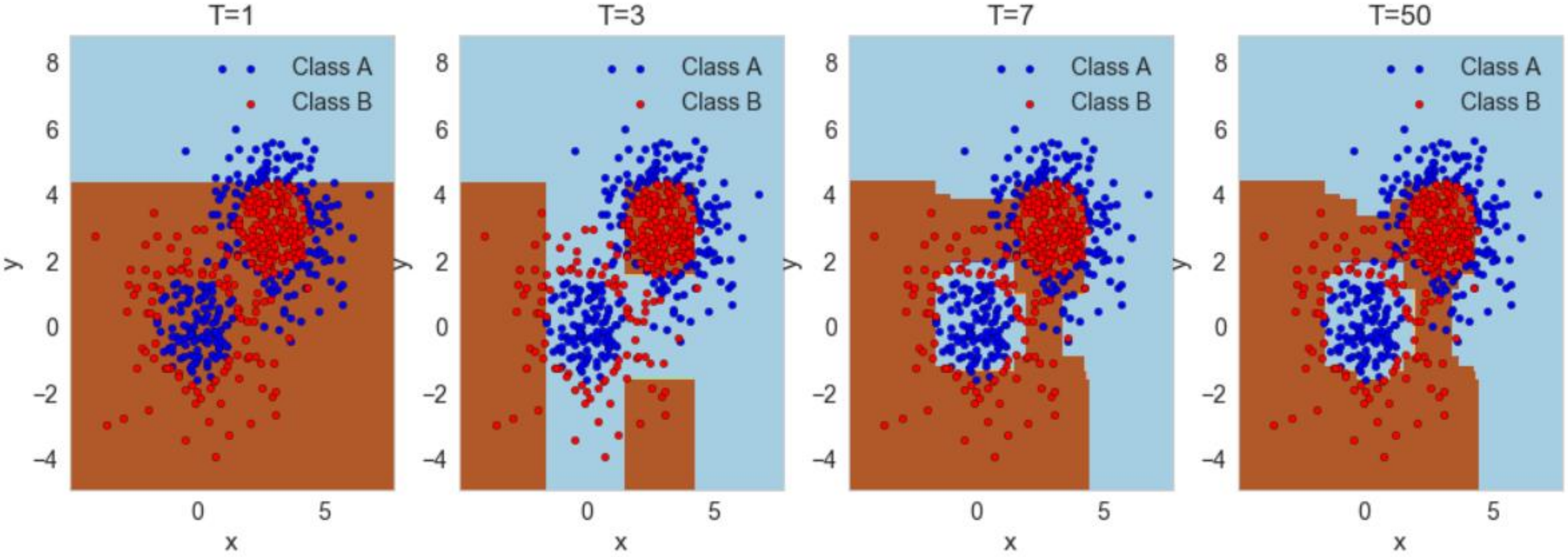
- The weight of a weak model in the boosted meta-model increases exponentially as the error approaches 0. Better models are given exponentially more weight.

- The weight is zero if the error rate is 0.5. A model with 50% accuracy is no better than random guessing, so it is ignored.

- The weight decreases exponentially as the error approaches 1. A negative weight is given to classifiers with worse than 50% accuracy. “Whatever that classifier says, do the opposite!”.

$$\hat{f}(x,\theta) = \sum_{t=1}^{T} \alpha_t f_t(x,\theta)$$

Boosting: adaboost



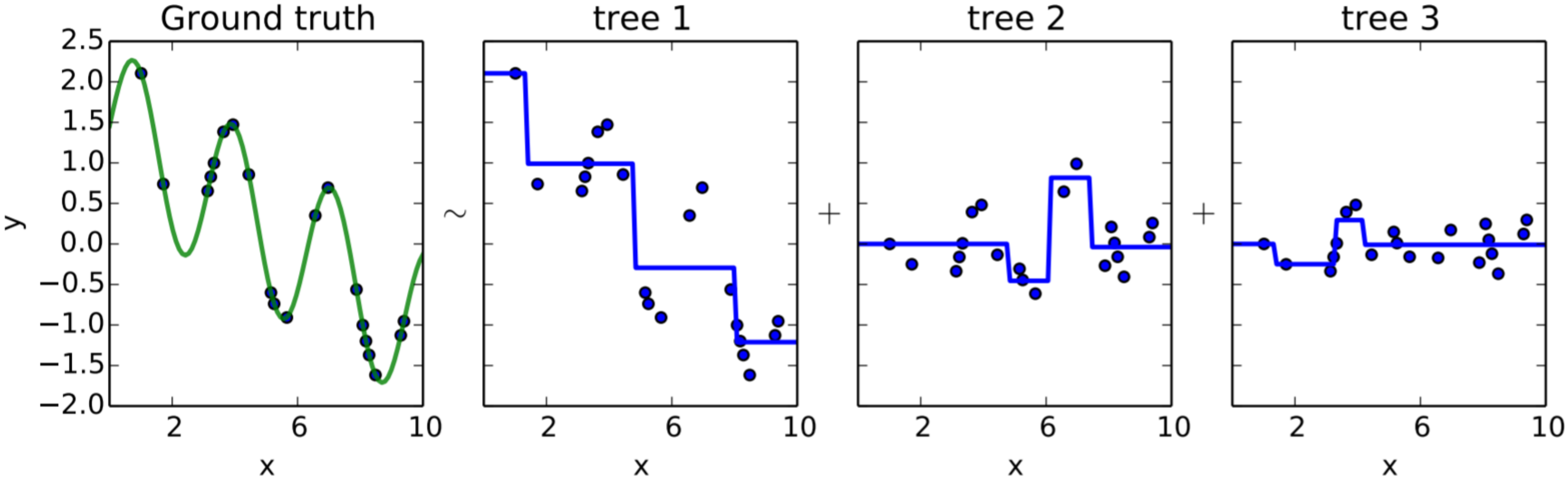
Boosting: Gradient Boosting

$$\hat{f}(x, \theta) = \sum_{t=1}^{T} f_t(x, \theta)$$

1. Fit a model $f_1(x, \theta) = y$
2. Fit a model to the **residuals** $h_1(x) = y - f_1(x, \theta)$
3. Create a new model $f_2(x, \theta) = f_1(x, \theta) + h_1(x)$

$$f_0(x, \theta) = \frac{1}{n} \sum_{i=1}^{n} y_i$$

$$f_{t+1}(x, \theta) = f_t(x, \theta) + h_t(x)$$

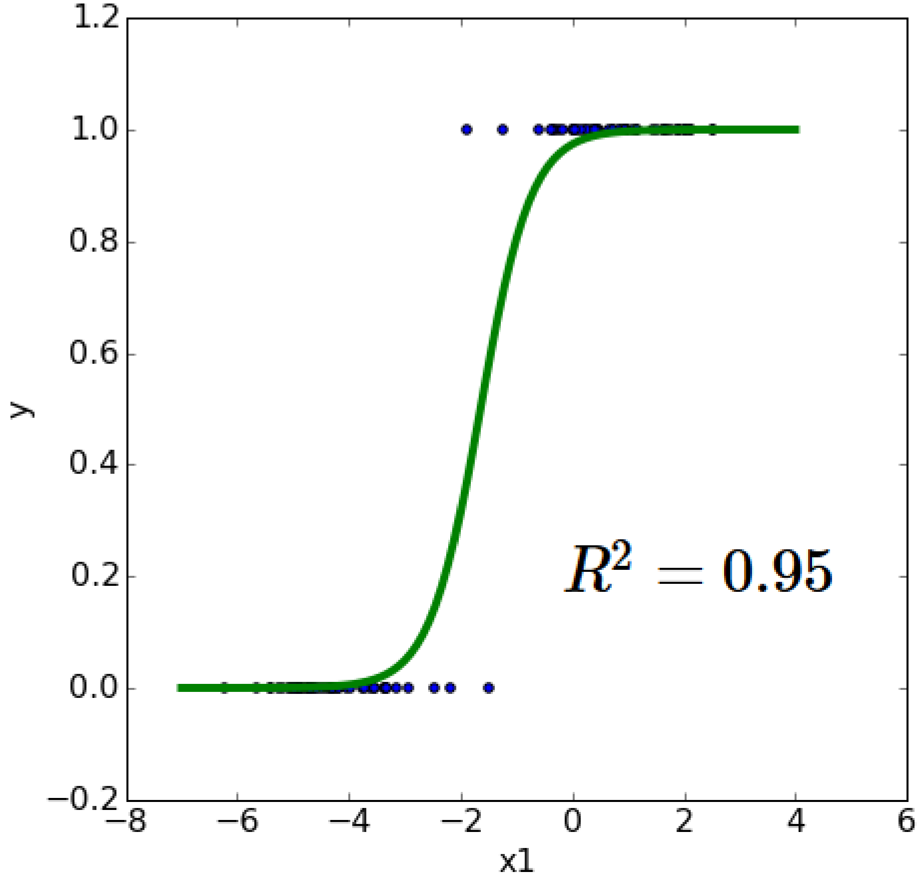


Ground truth
tree 1
tree 2
tree 3
y
x
~
+
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
2
6
10

Let’s get real!

6. **Ensemble learning: boosting**

logistic regression



We need to make

assumptions

linearly separable

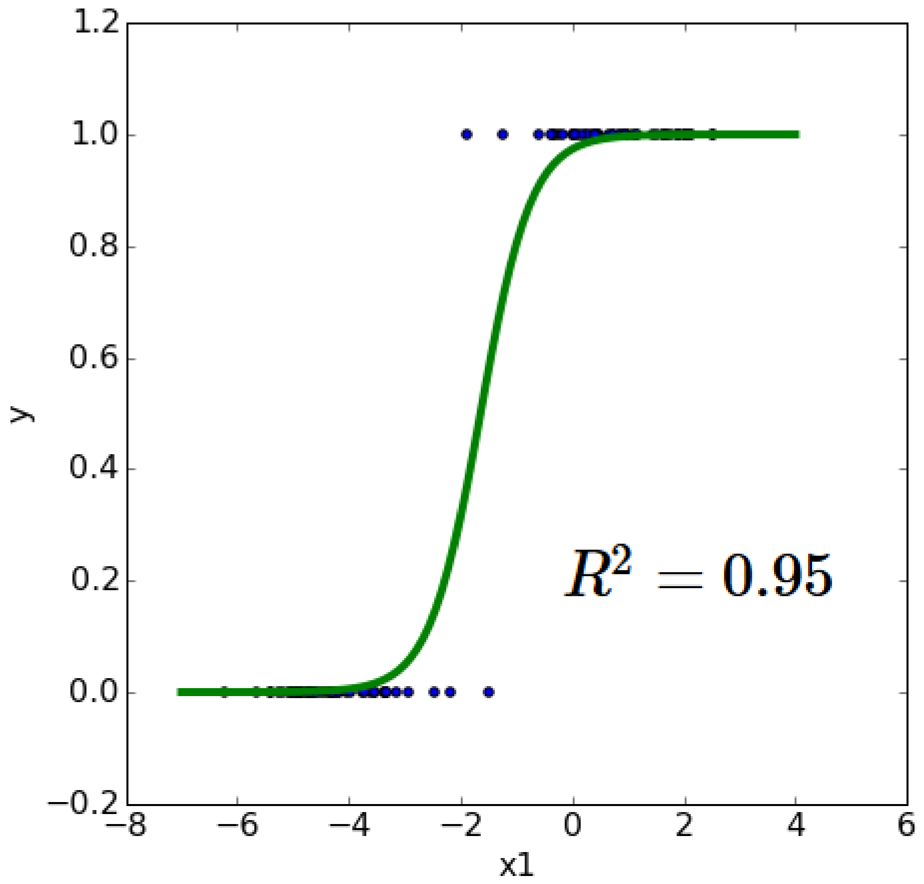
about the

model

logistic model

that generated the data.

logistic regression: logistic model



$$f(x, \theta) = g(\theta_0 + \theta_1 x_1)$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

logistic regression: cost function

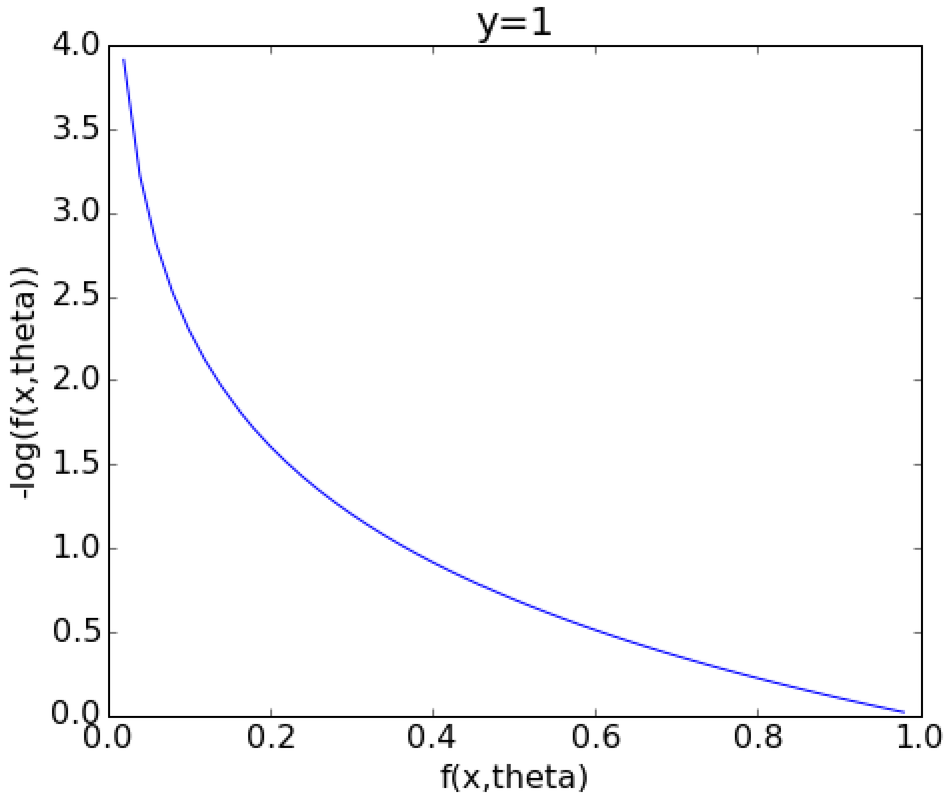
$$J(\theta) = -[\frac{1}{n}\sum_{i=1}^{n} y^{(i)} log(f(x^{(i)},\theta)) + (1 - y^{(i)}) log(1 - f(x^{(i)},\theta))]$$

We know that $y^{(i)}$ is either 0 or 1. If $y^{(i)} = 1$ then the cost function $J(\theta)$ is incremented by $-log(f(x^{(i)},\theta))$.

Similarly, if $y^{(i)} = 0$ then the cost function $J(\theta)$ is incremented by $-log(1 - f(x^{(i)},\theta))$.

logistic regression: cost function

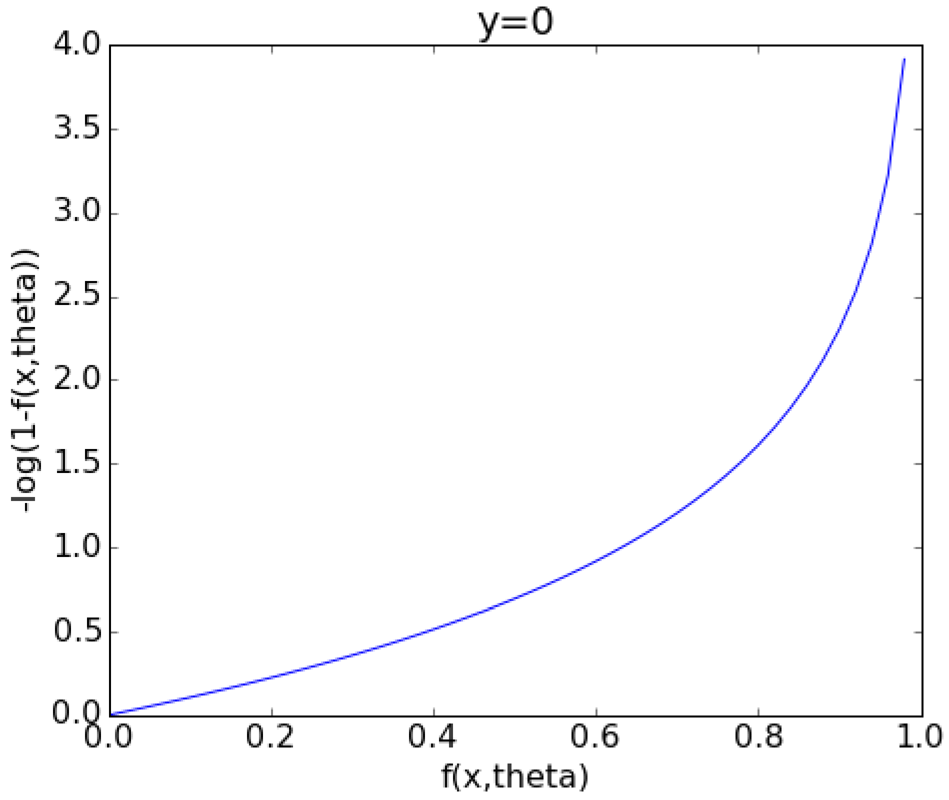
We know that $y^{(i)}$ is either 0 or 1. If $y^{(i)} = 1$ then the cost function $J(\theta)$ is incremented by $-log(f(x^{(i)}, \theta))$.



logistic regression: cost function

Similarly, if $y^{(i)} = 0$ then the cost function $J(\theta)$ is incremented by

$$-log(1 - f(x^{(i)}, \theta)).$$



logistic regression

Fit a logistic model

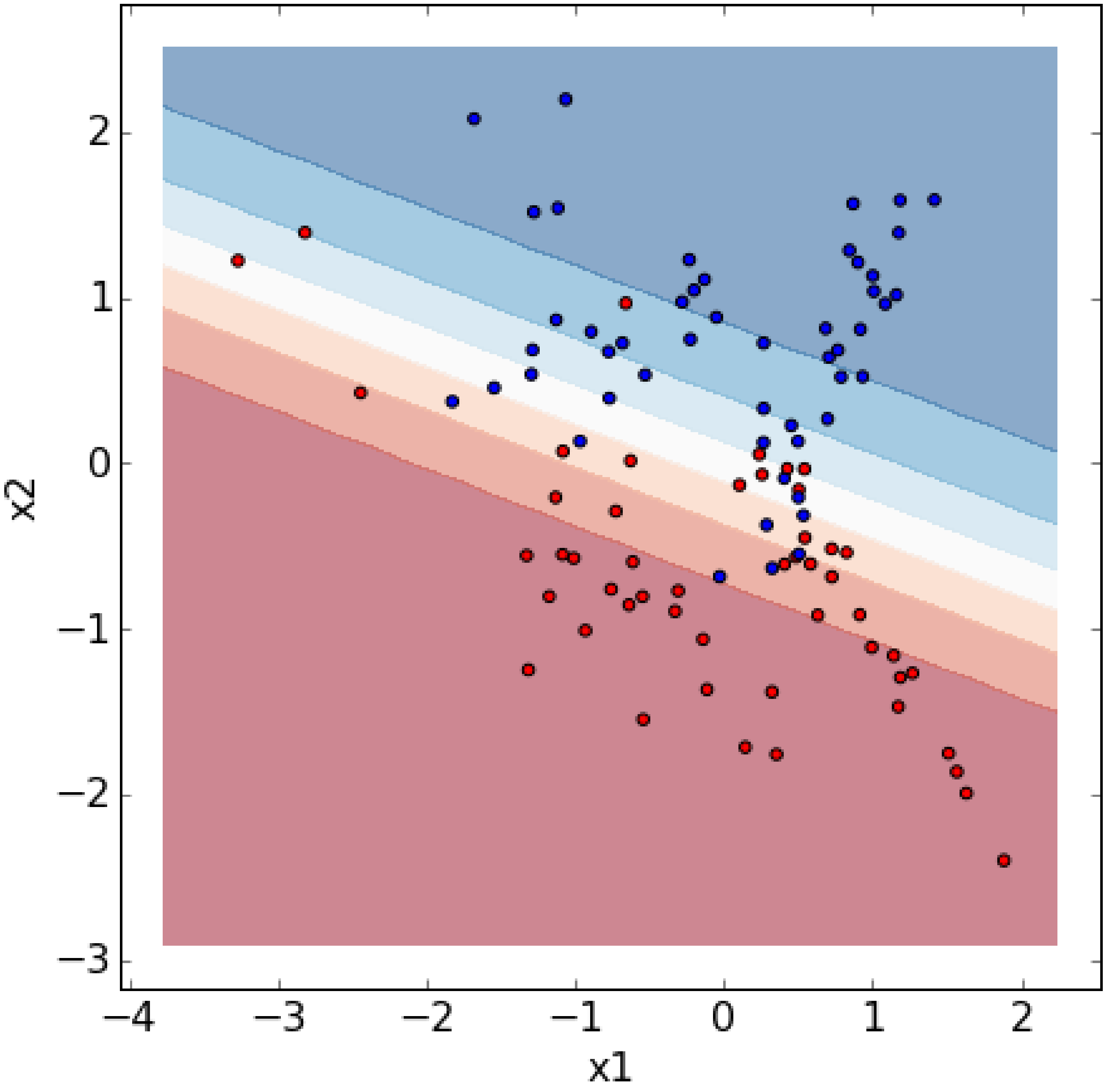
$$f(x,\theta) = g(\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_m x_m) = g(\theta' x)$$

to the data set such that the cost function

$$J(\theta) = -[\frac{1}{n}\sum_{i=1}^{n} y^{(i)} log(f(x^{(i)},\theta)) + (1 - y^{(i)}) log(1 - f(x^{(i)},\theta))]$$

is minimal using gradient descent

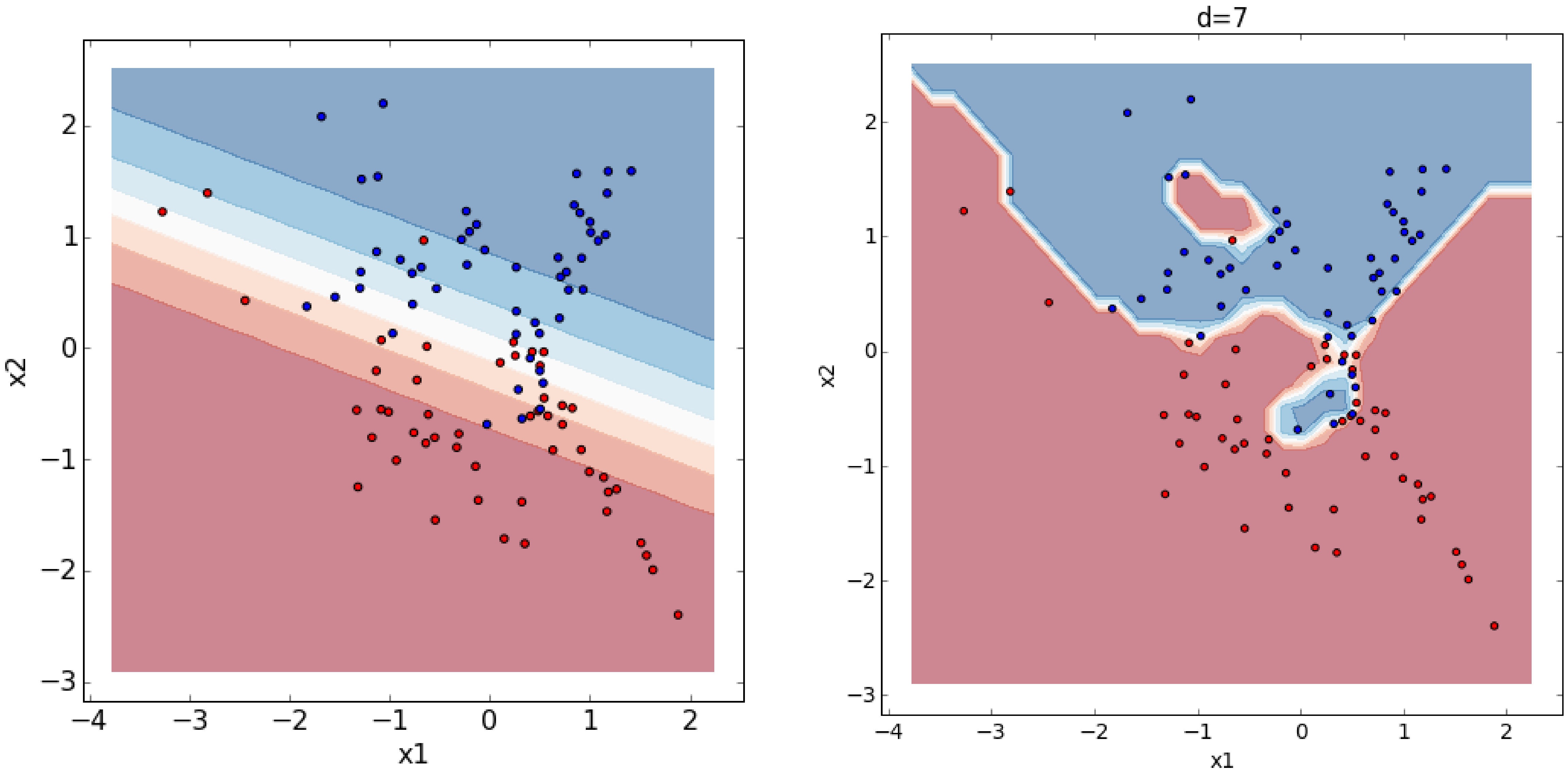
$$\theta_j := \theta_j - \alpha \frac{1}{n} \sum_{i=1}^{n} (f(x^{(i)},\theta) - y^{(i)}) x_j^{(i)}$$



x1
x2
−4
−3
−2
−1
0
1
2
−3

Let’s get real!

7. **Logistic regression**
8. **Non-linear transformations**



x1
x2
d=7
x1
x2

regularized logistic regression

$$f(x,\theta) = g(\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_m x_m)$$

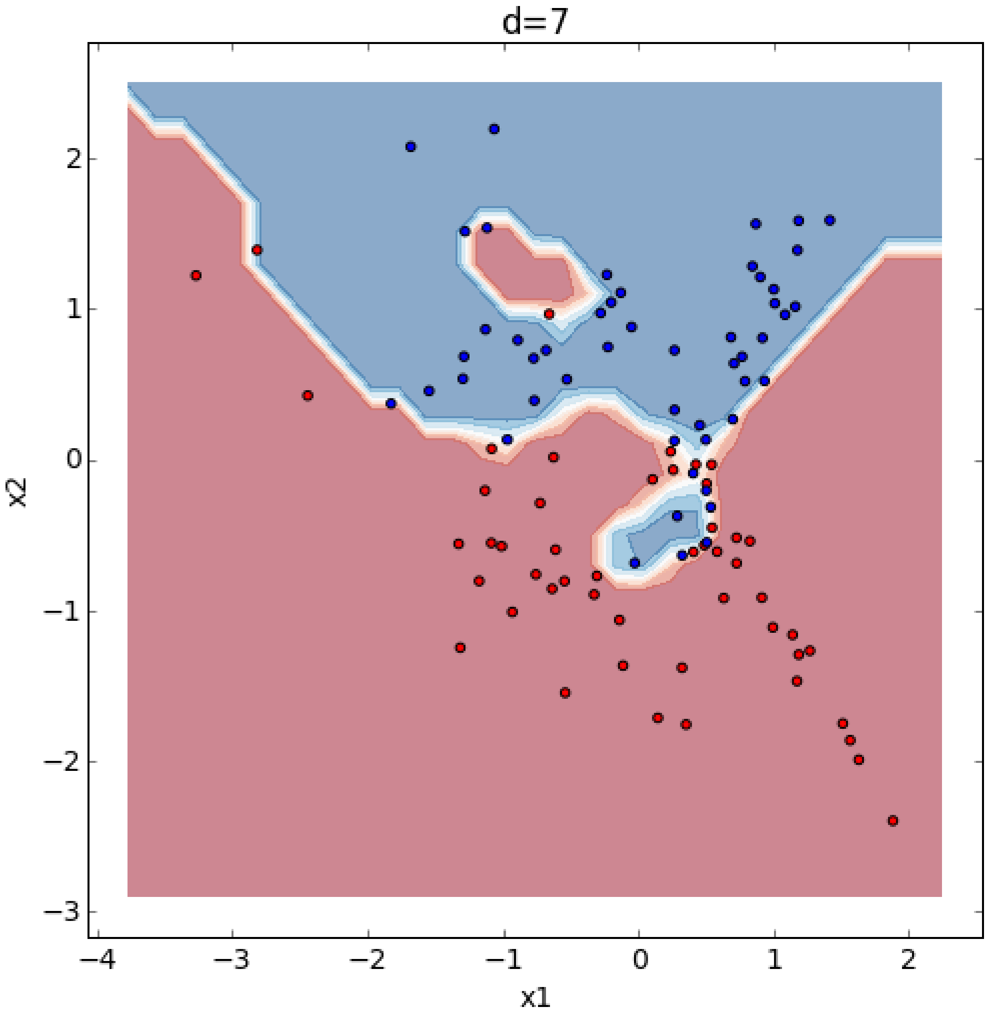
cost function

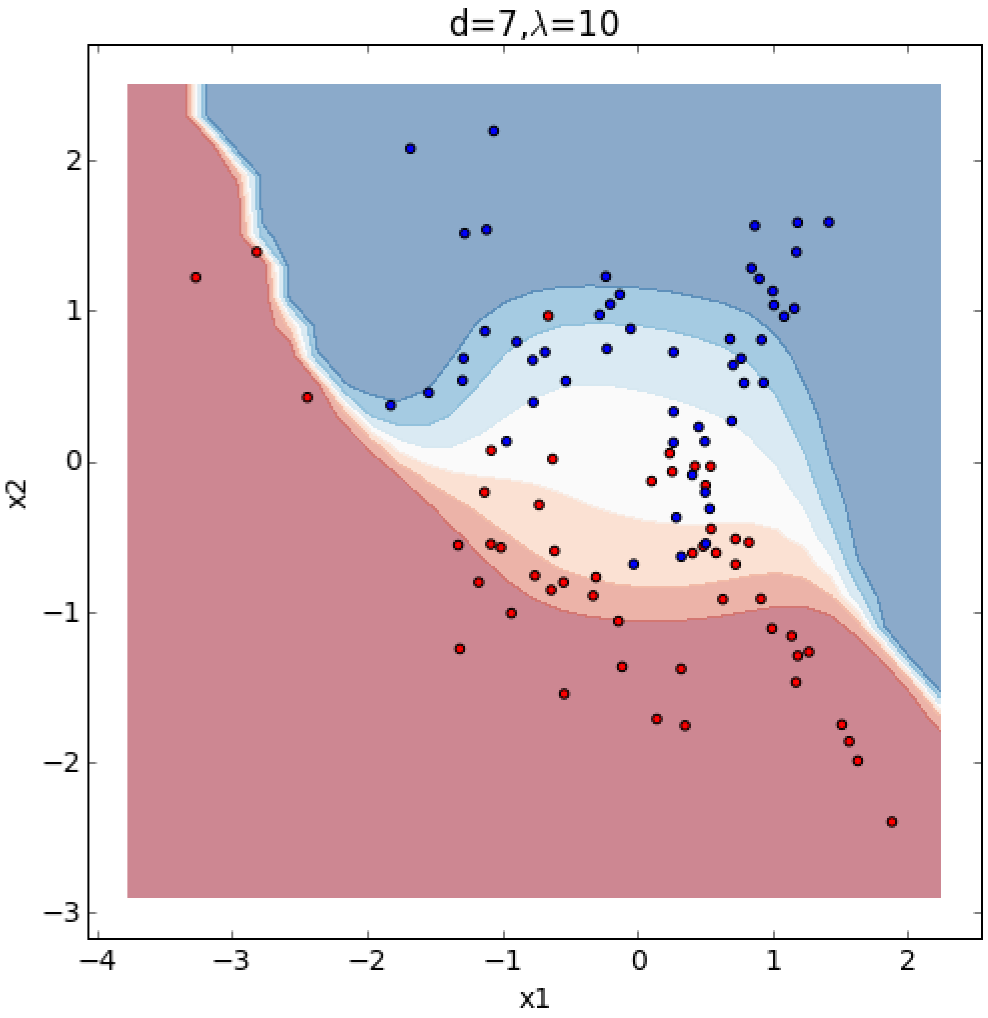
$$J(\theta) = -[\frac{1}{n}\sum_{i=1}^{n} y^{(i)} log(f(x^{(i)},\theta)) + (1-y^{(i)}) log(1-f(x^{(i)},\theta))]$$

$$J(\theta) = -[\frac{1}{n}\sum_{i=1}^{n} y^{(i)} log(f(x^{(i)},\theta)) + (1-y^{(i)}) log(1-f(x^{(i)},\theta))] + \frac{\lambda}{2m}\sum_{j=1}^{m}\theta_j^2$$

regularized cost function

regularized logistic regression





support vector machines

Fit a linear model

$$f(x,\theta) = \theta' x$$

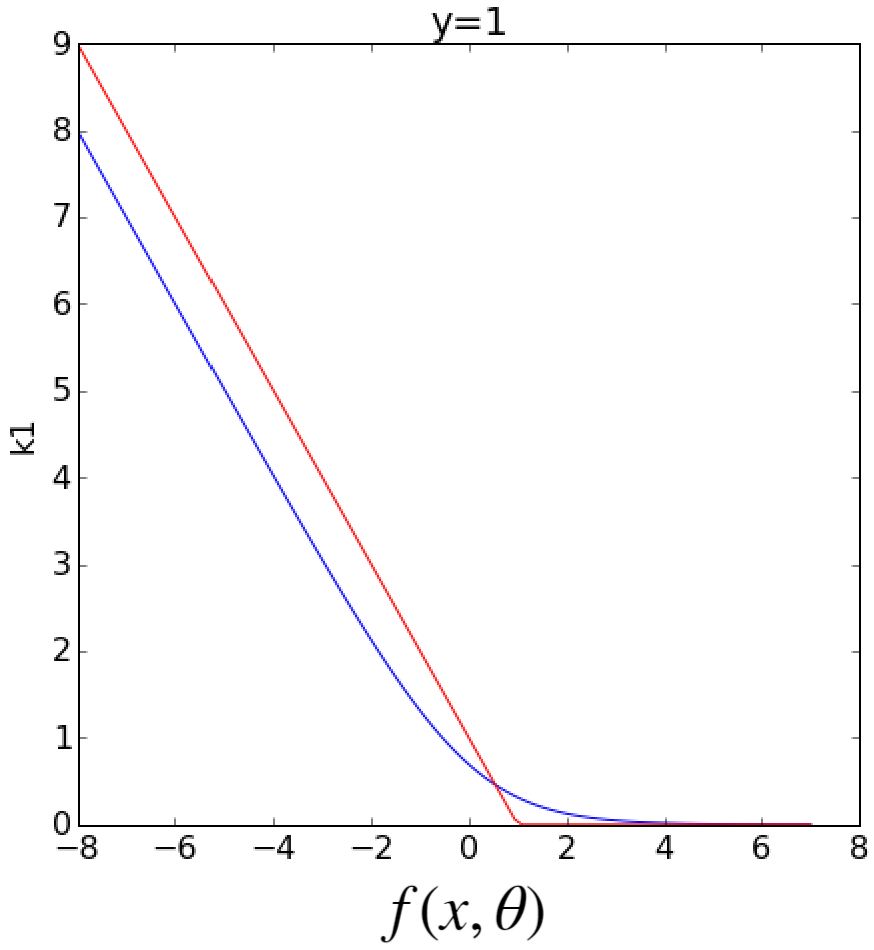
such that

$$J(\theta) = \left[C \sum_{i=1}^{n} y^{(i)} k_1(\theta' x^{(i)}) + (1 - y^{(i)}) k_0(\theta' x^{(i)}) \right] + \frac{1}{2} \sum_{j=1}^{m} \theta_j^2$$

with $k_1(\theta' x) = max(0, 1 - \theta' x)$ and $k_0(\theta' x) = max(0, 1 + \theta' x)$

is minimized.

support vector machines

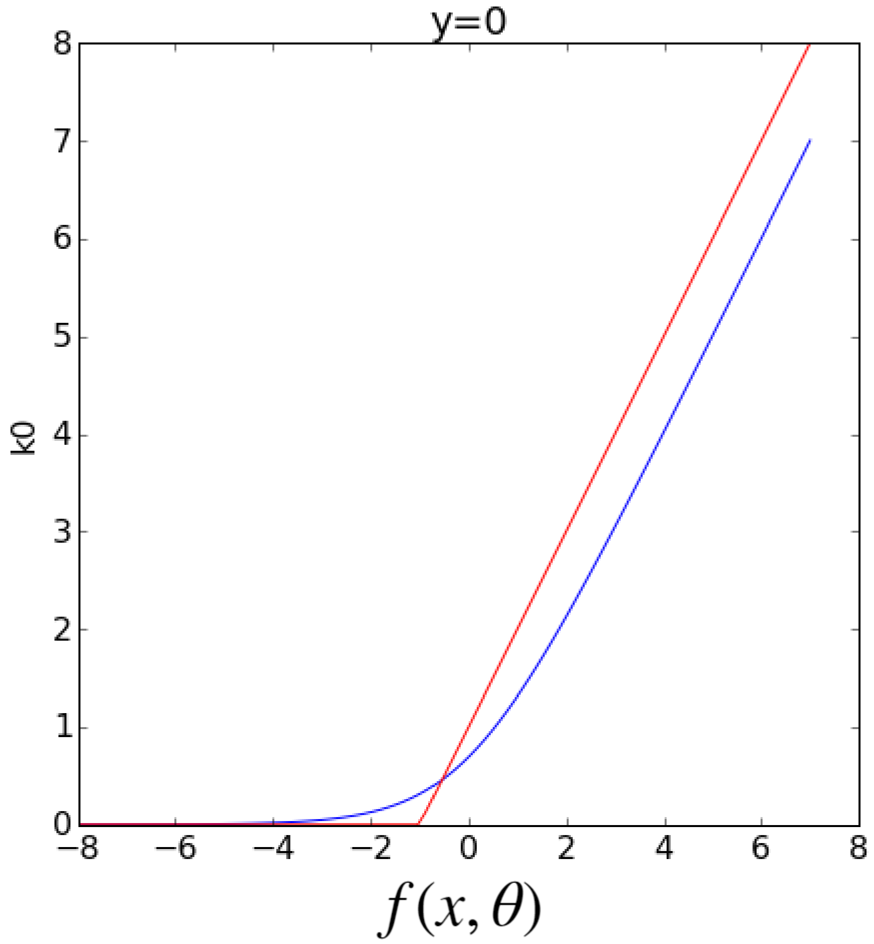


- replace cost function by piecewise linear function

- if *y* = 1 then the contribution to the cost is

$$k_1(f(x,\theta)) = max(0, 1 - f(x,\theta))$$

support vector machines



- replace cost function by piecewise linear function

- if *y* = 1 then the contribution to the cost is

$$k_1(f(x,\theta)) = max(0, 1 - f(x,\theta))$$

- if *y* = 0 then the contribution to the cost is

$$k_0(f(x,\theta)) = max(0, 1 + f(x,\theta))$$

kernel support vector machines

SVMs can also be formulated as a linear function of the samples (dual form) instead of the features as

$$f(x, \theta) = \sum_{i=1}^{n} \theta_i (x \cdot x^{(i)}) + \theta_0$$

that can be reformulated as a non-linear function using what is know as a kernel function

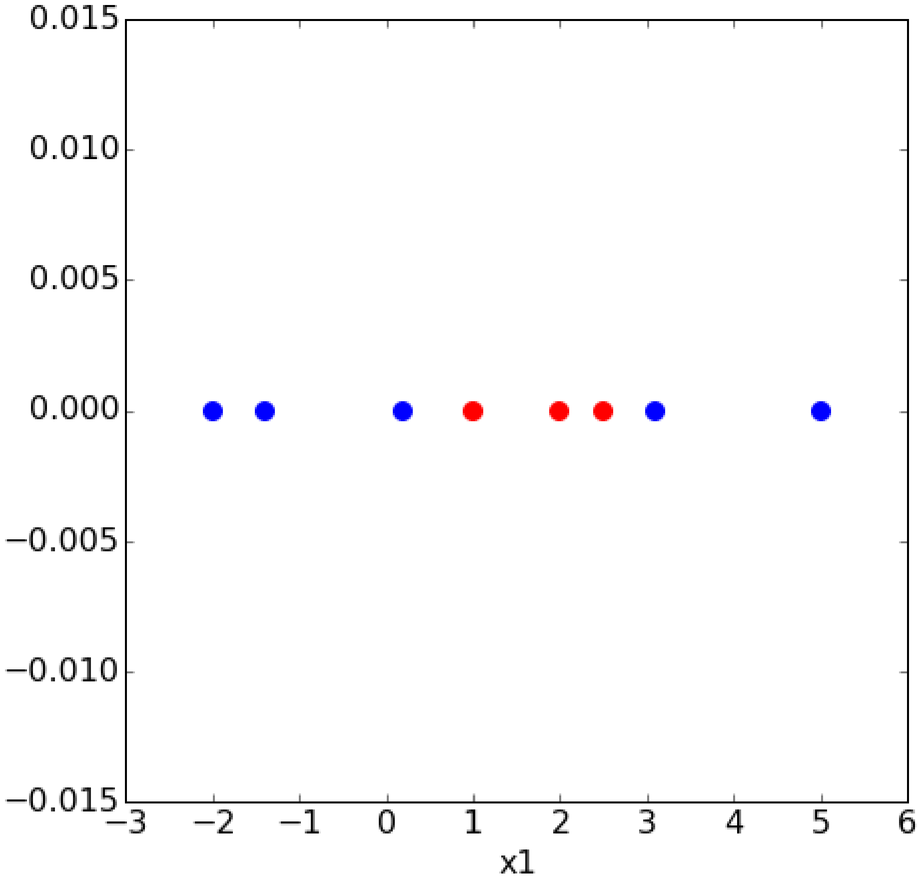
$$K(x^{(i)}, x^{(j)})$$

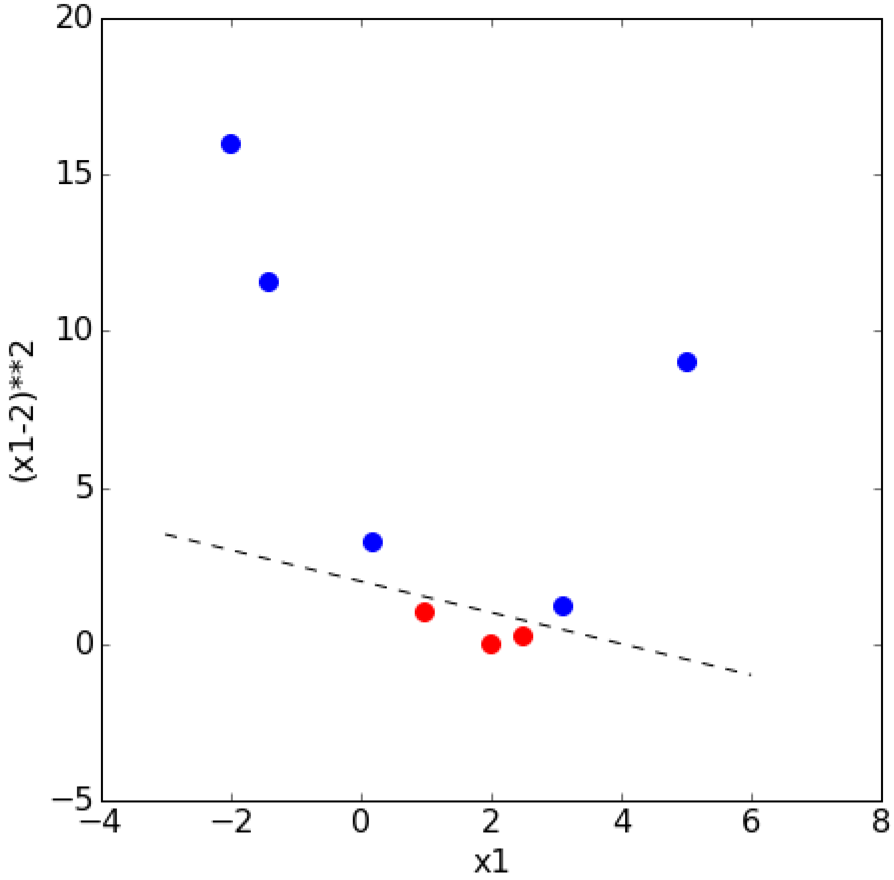
to become

$$f(x, \theta) = \sum_{i=1}^{n} \theta_i K(x, x^{(i)}) + \theta_0$$

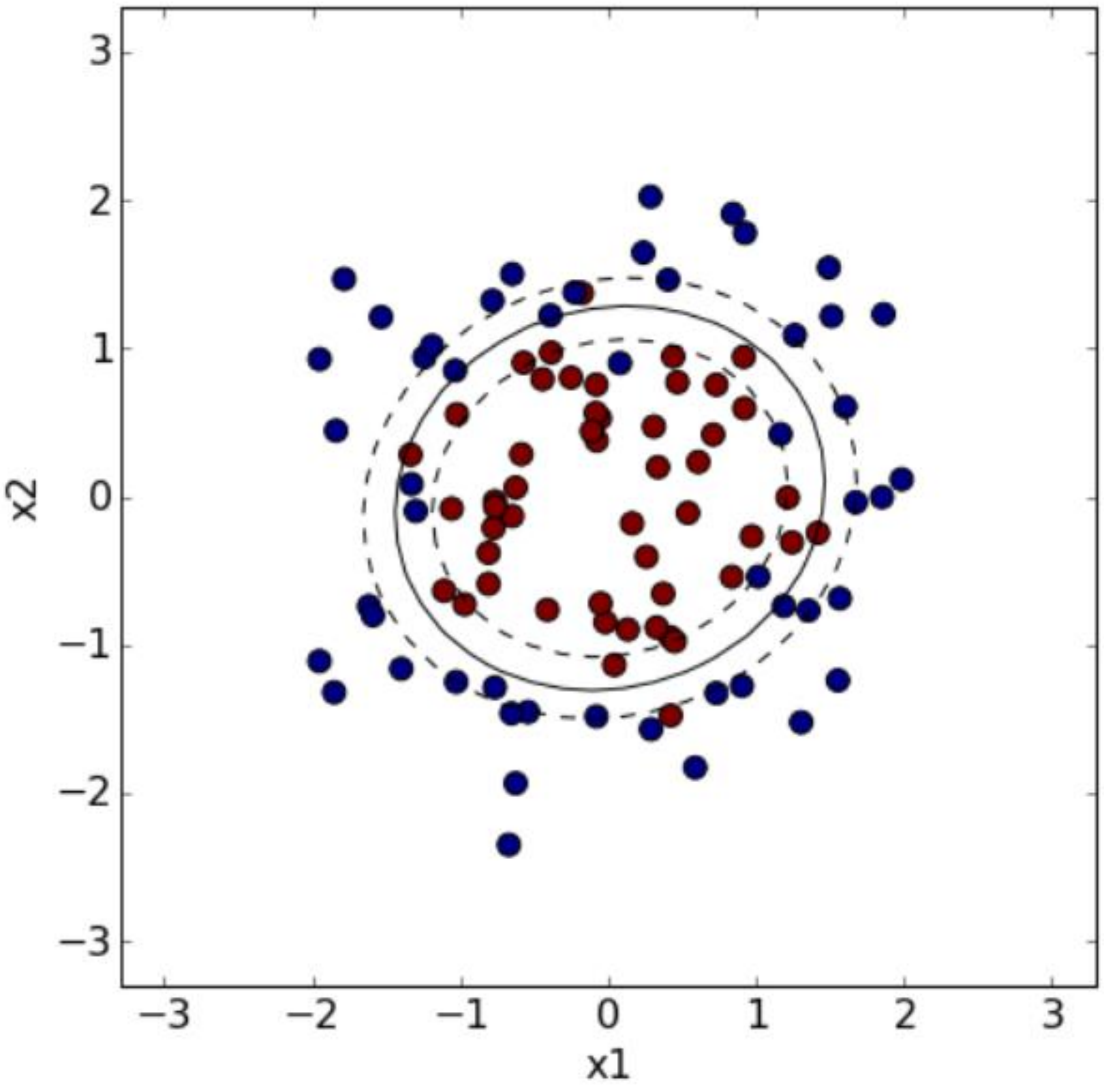
The data points $x^{(i)}$ for which $\theta_i > 0$ are called the support vectors.

kernel support vector machines





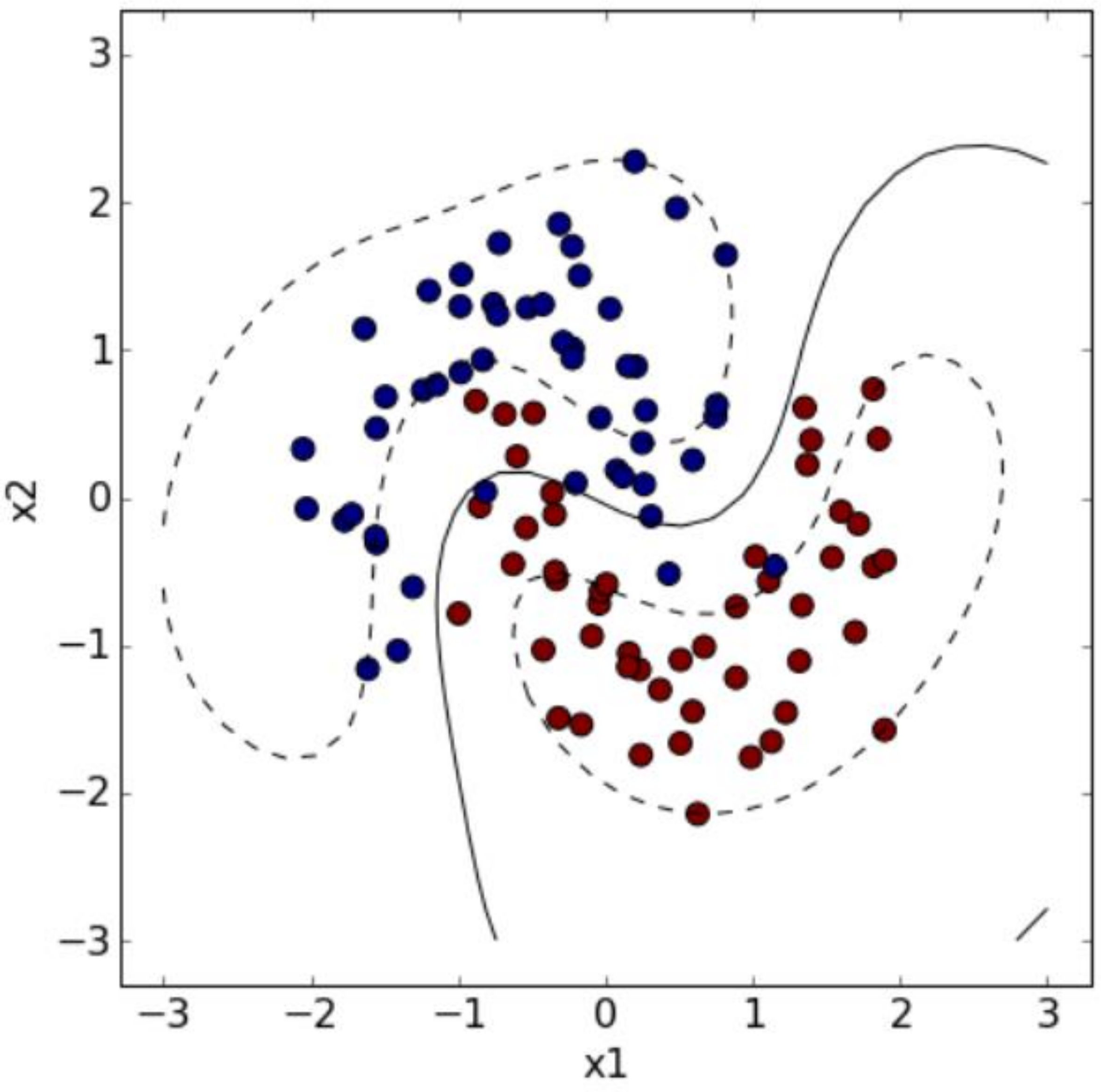
kernel support vector machines



$$f(x,\theta) = \sum_{i=1}^{n} \theta_i K(x, x^{(i)}) + \theta_0$$

$$K(x^{(i)}, x^{(j)}) = (x^{(i)} \cdot x^{(j)} + c)^d$$

kernel support vector machines



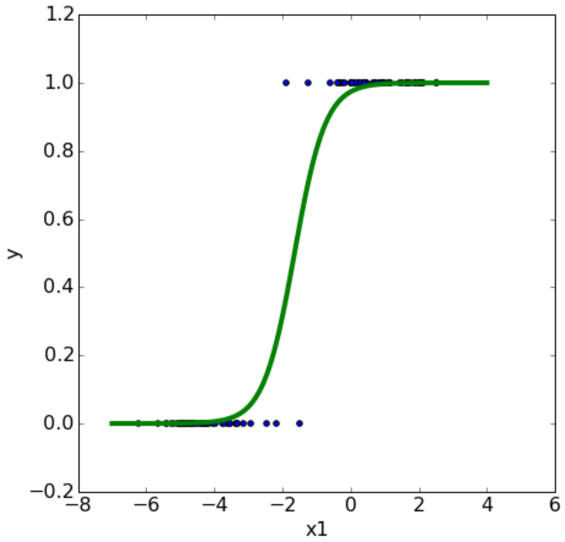
$$f(x, \theta) = \sum_{i=1}^{n} \theta_i K(x, x^{(i)}) + \theta_0$$

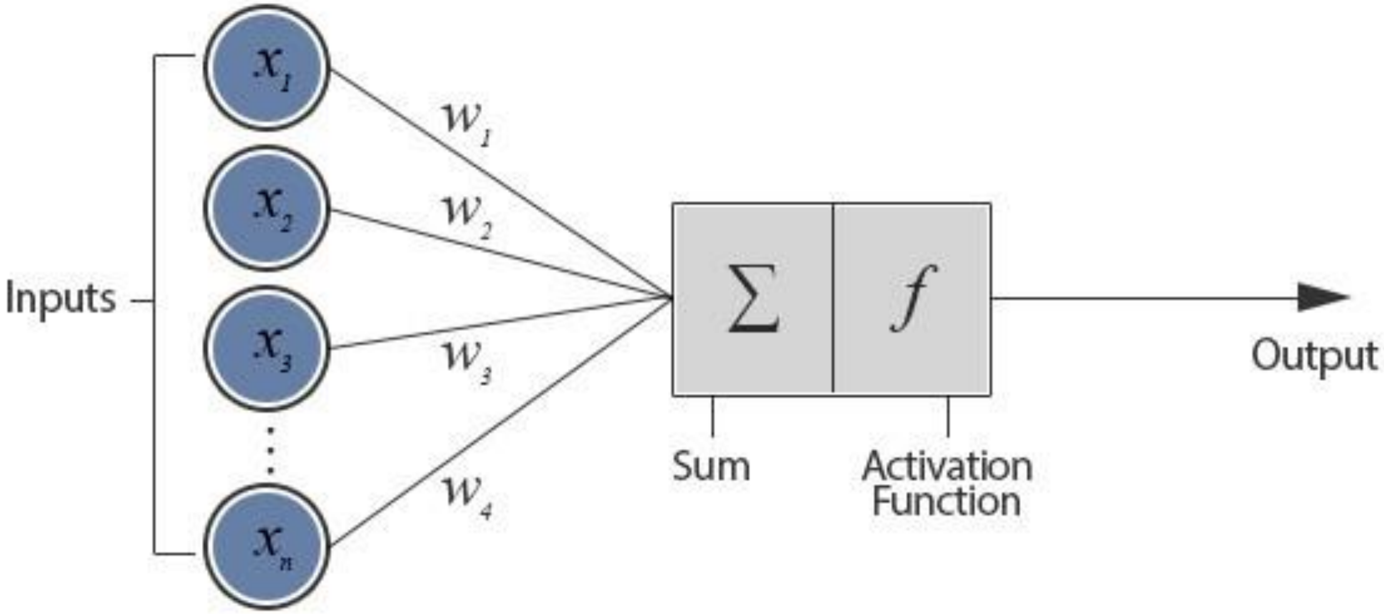
$$K(x^{(i)}, x^{(j)}) = \exp\left[-\frac{\|x^{(i)} - x^{(j)}\|^2}{2\sigma^2}\right]$$

Let’s get real!

9. Support Vector Machines

$$f(x, \theta) = g(\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_m x_m) = g(\theta' x)$$



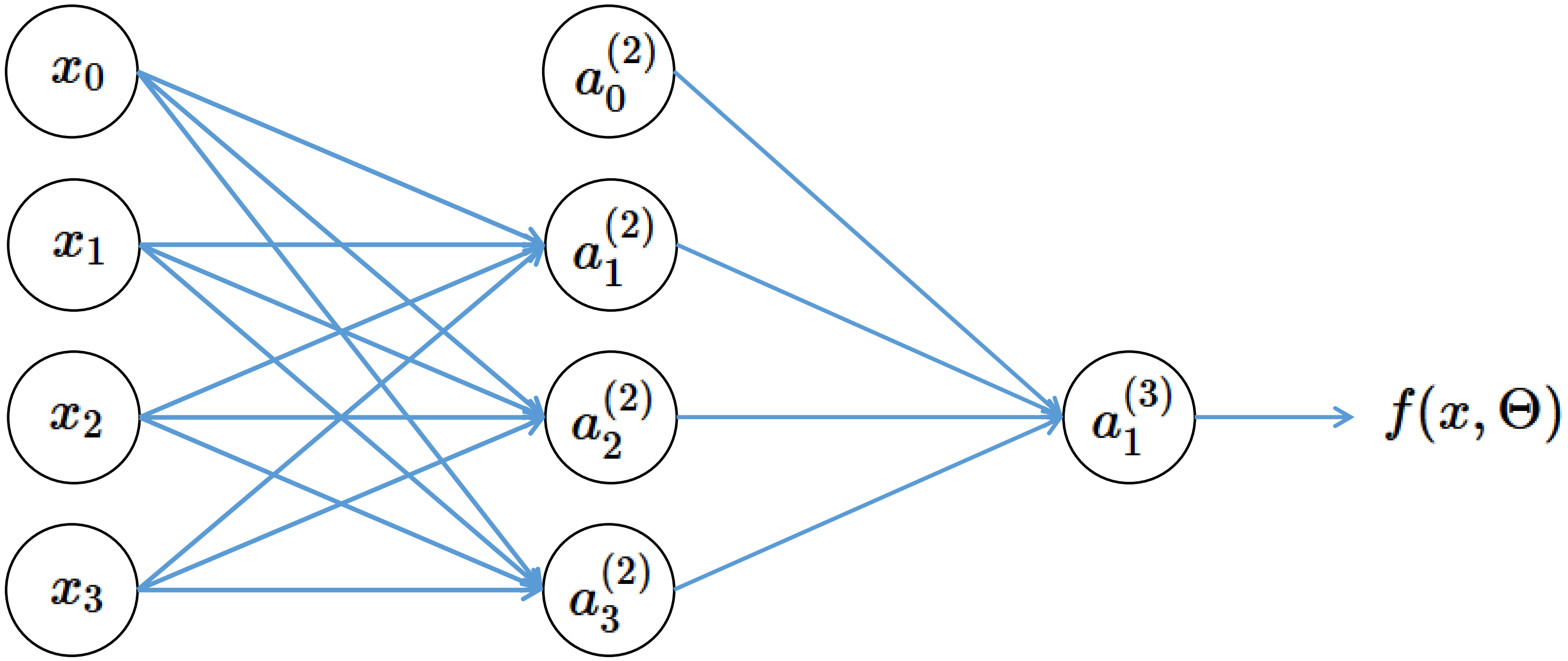


Model: $f(x,\Theta) = g(\Theta_{10}^{(2)} a_0 + \Theta_{11}^{(2)} a_1^{(2)} + \Theta_{12}^{(2)} a_2^{(2)} + \Theta_{13}^{(2)} a_3^{(2)})$

$$a_1^{(2)} = g(\Theta_{10}^{(1)} x_0 + \Theta_{11}^{(1)} x_1 + \Theta_{12}^{(1)} x_2 + \Theta_{13}^{(1)} x_3)$$

$$a_2^{(2)} = g(\Theta_{20}^{(1)} x_0 + \Theta_{21}^{(1)} x_1 + \Theta_{32}^{(1)} x_2 + \Theta_{23}^{(1)} x_3)$$

$$a_3^{(2)} = g(\Theta_{30}^{(1)} x_0 + \Theta_{31}^{(1)} x_1 + \Theta_{32}^{(1)} x_2 + \Theta_{33}^{(1)} x_3)$$



Cost function logistic regression:

$$J(\theta) = -[\frac{1}{m}\sum_{i=1}^{n} y^{(i)}log(f(x^{(i)},\theta) + (1-y^{(i)})log(1-f(x^{(i)},\theta))] + \frac{\lambda}{2n}\sum_{j=1}^{n}\theta^2$$

Cost function feedforward neural network:

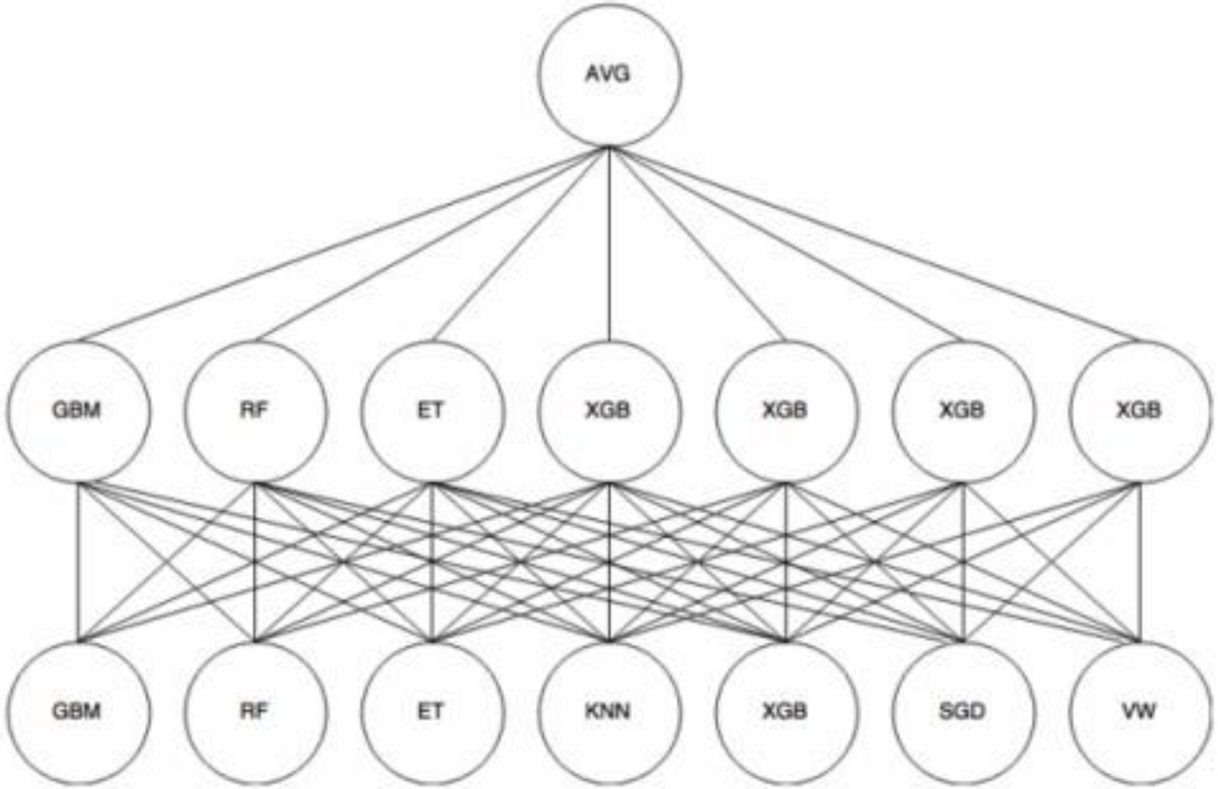
$$J(\theta) = -[\frac{1}{m}\sum_{i=1}^{n}\sum_{k=1}^{K} y_k^{(i)}log(f(x^{(i)},\Theta)_k) + (1-y_k^{(i)})log(1-f(x^{(i)},\Theta)_k)] + \frac{\lambda}{2n}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}(\Theta_{ji}^{(l)})^2$$

https://playground.tensorflow.org/

Let’s get real!

10. Neural Networks

Blending/Stacking



https://mlwave.com/kaggle-ensembling-guide/