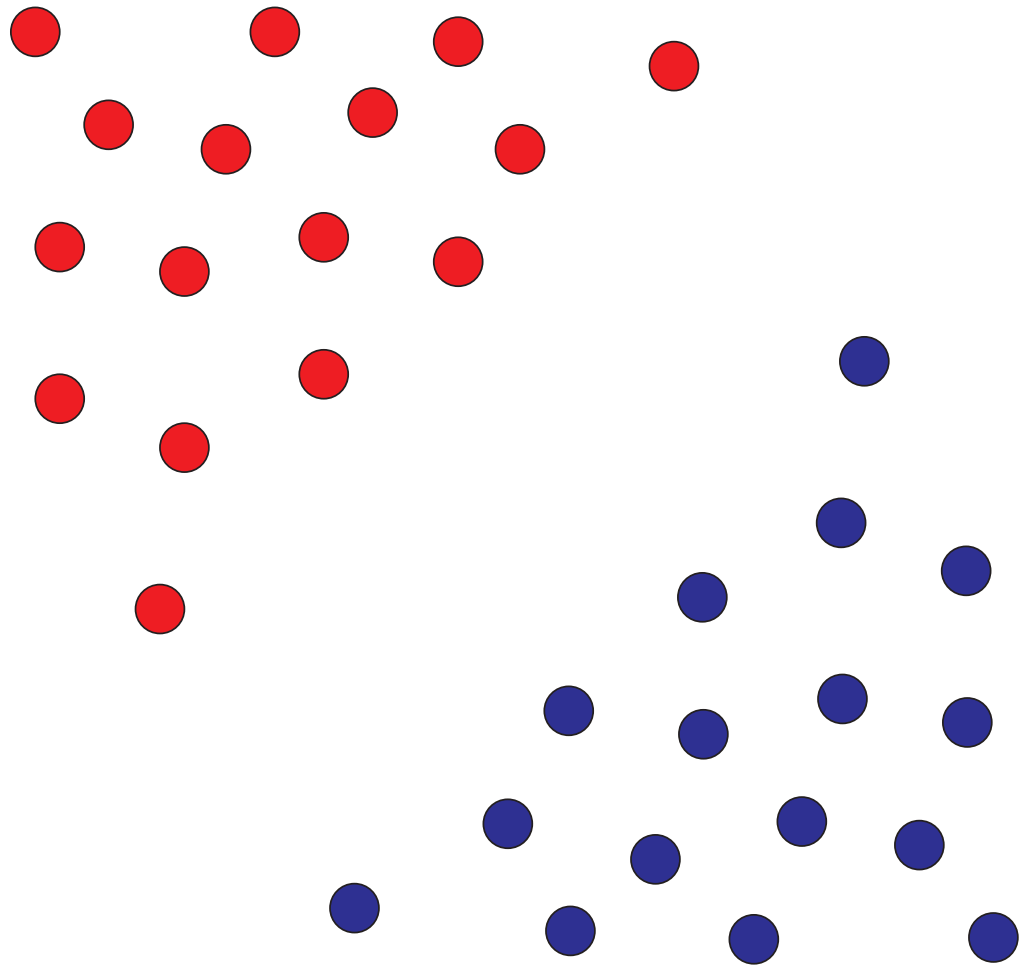
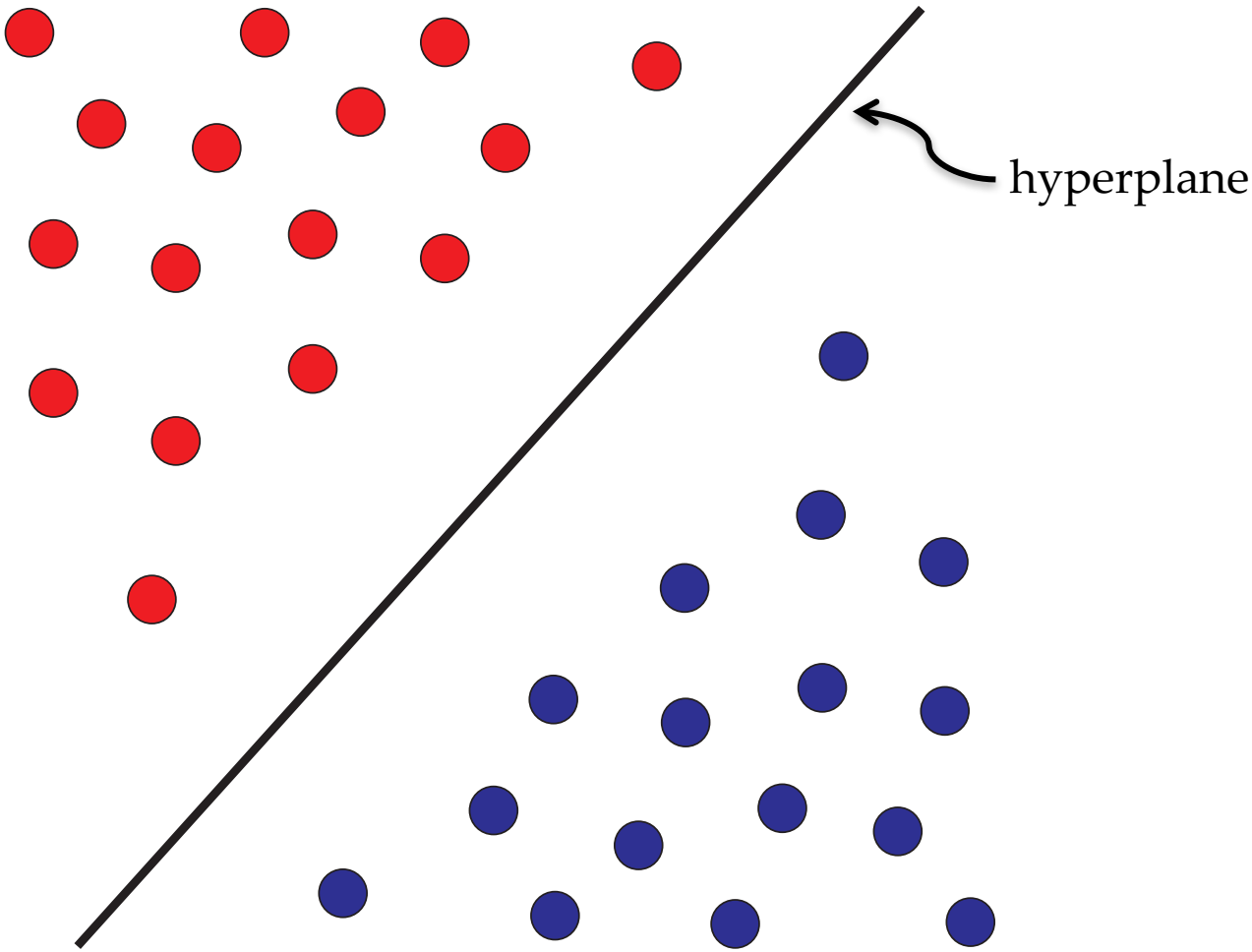
Support Vector Machines

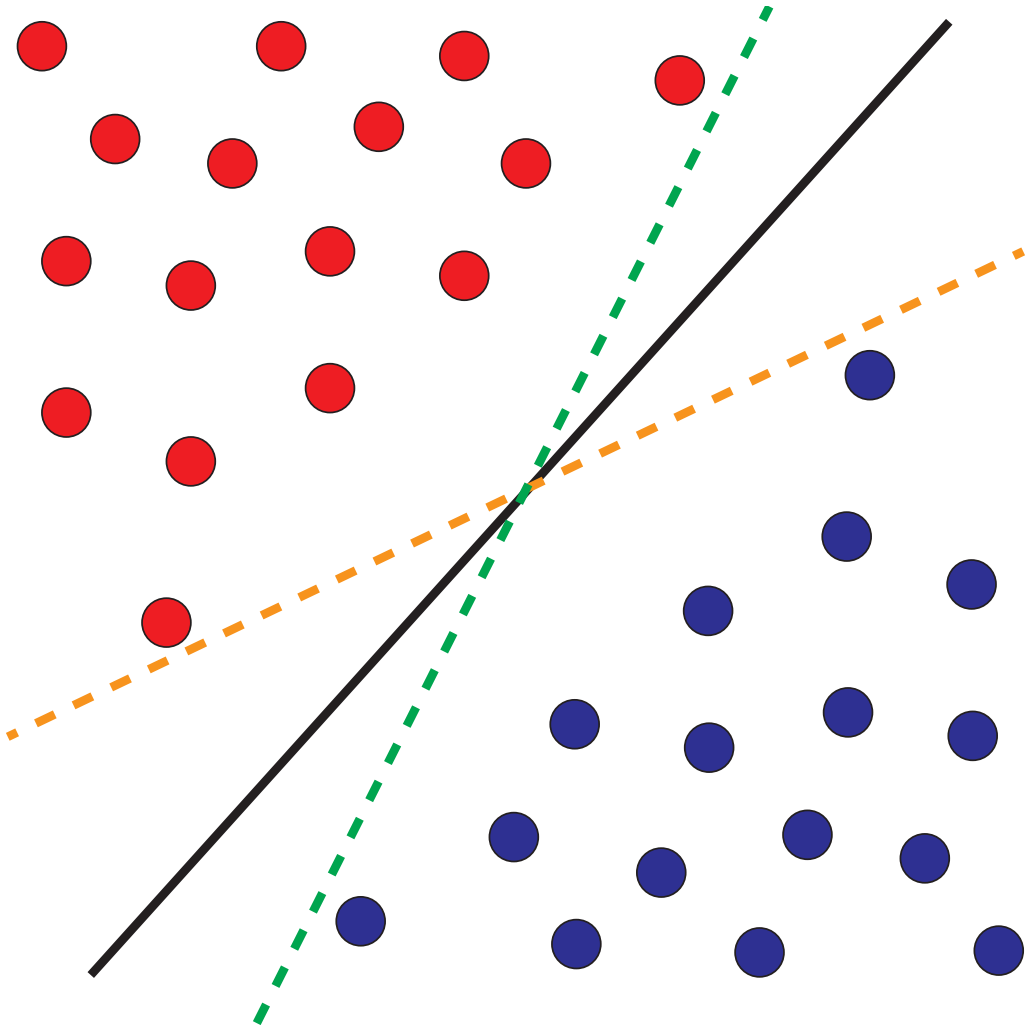
Linearly Separable Data



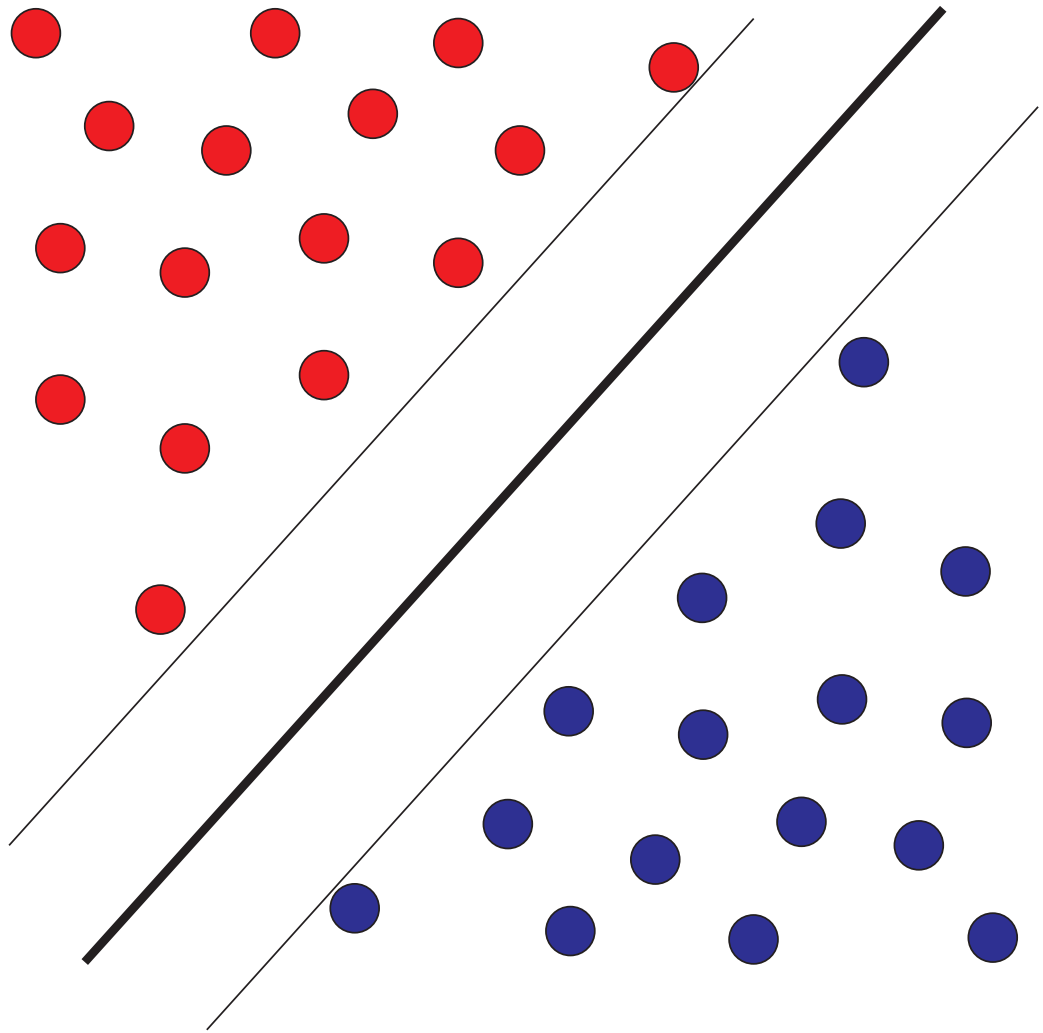
SVM: Simple Linear Separator



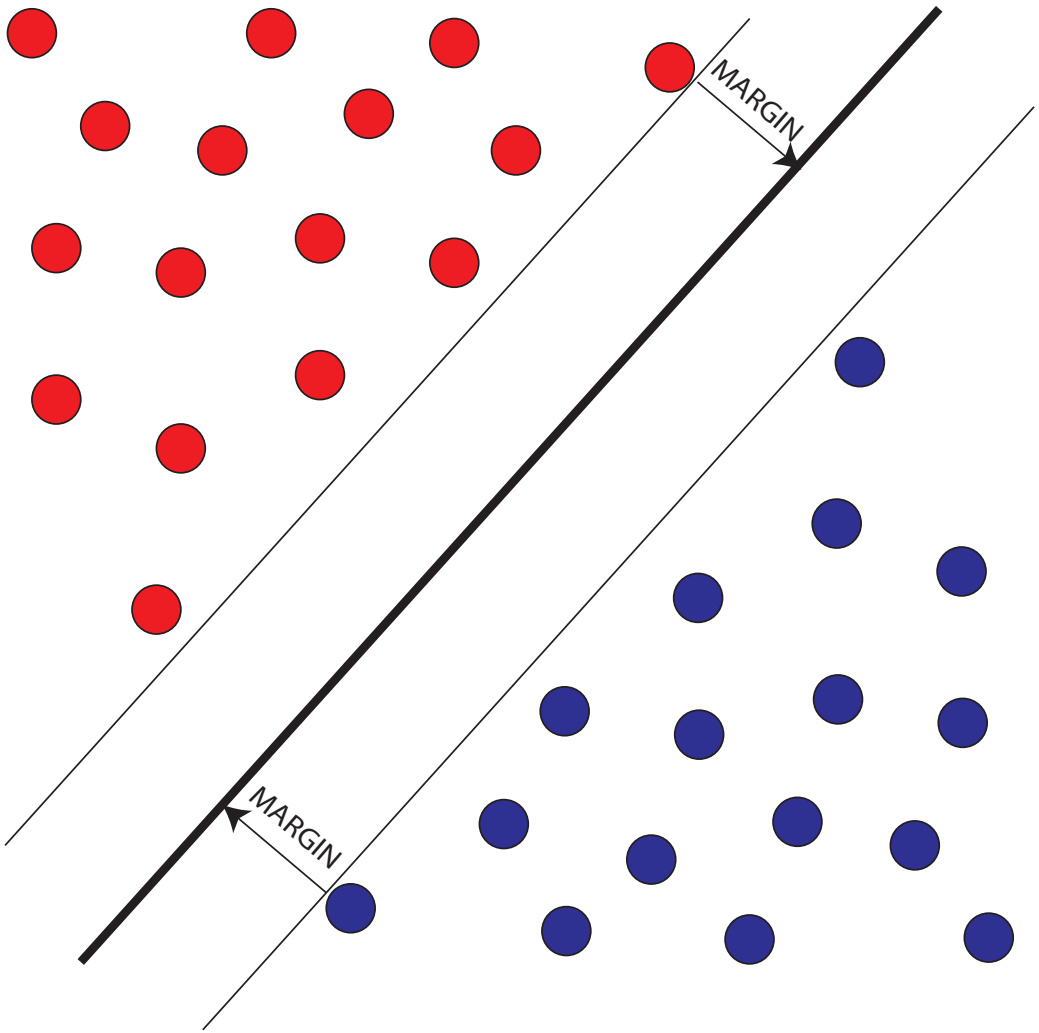
Which Simple Linear Separator?



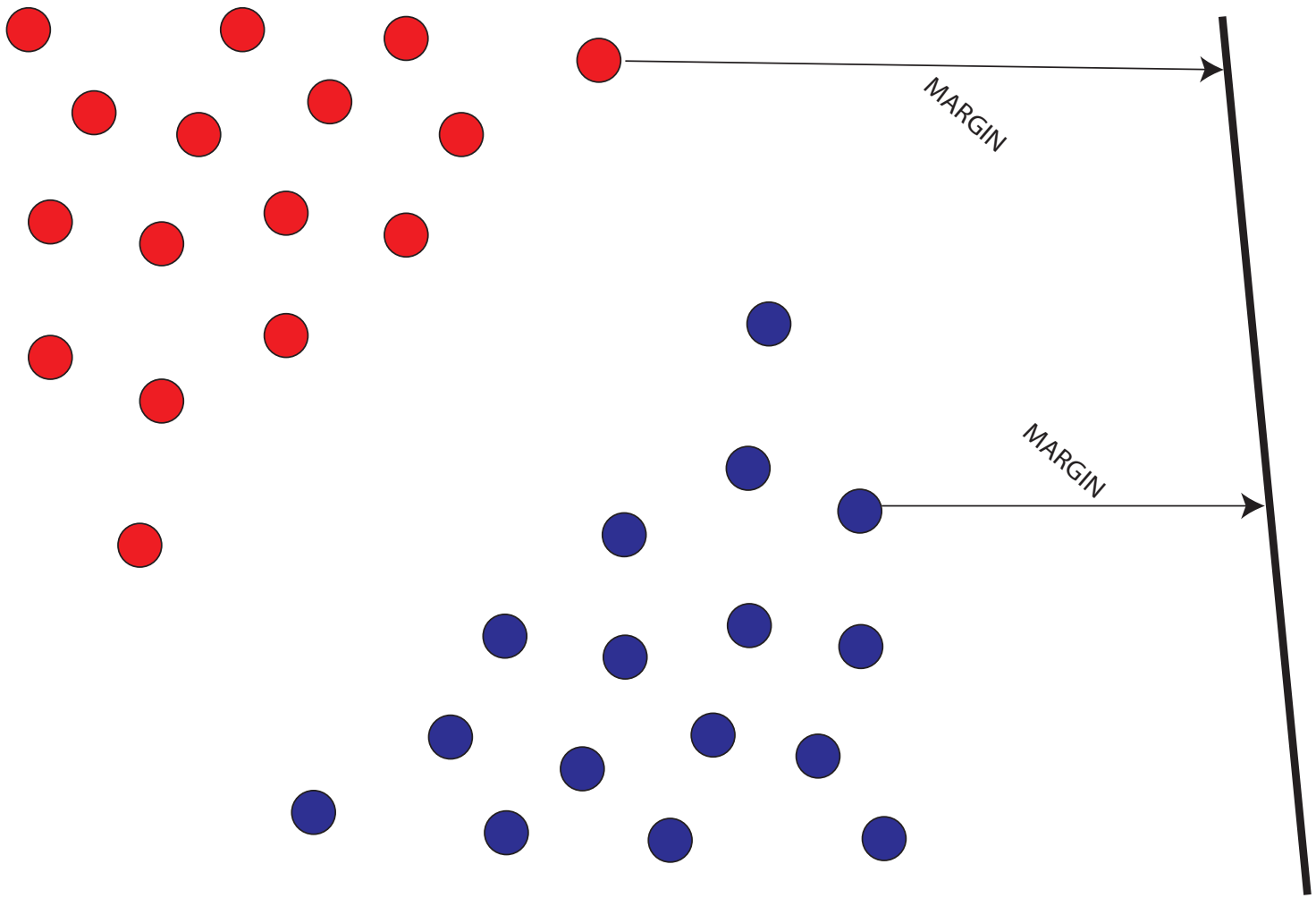
Classifier Margin



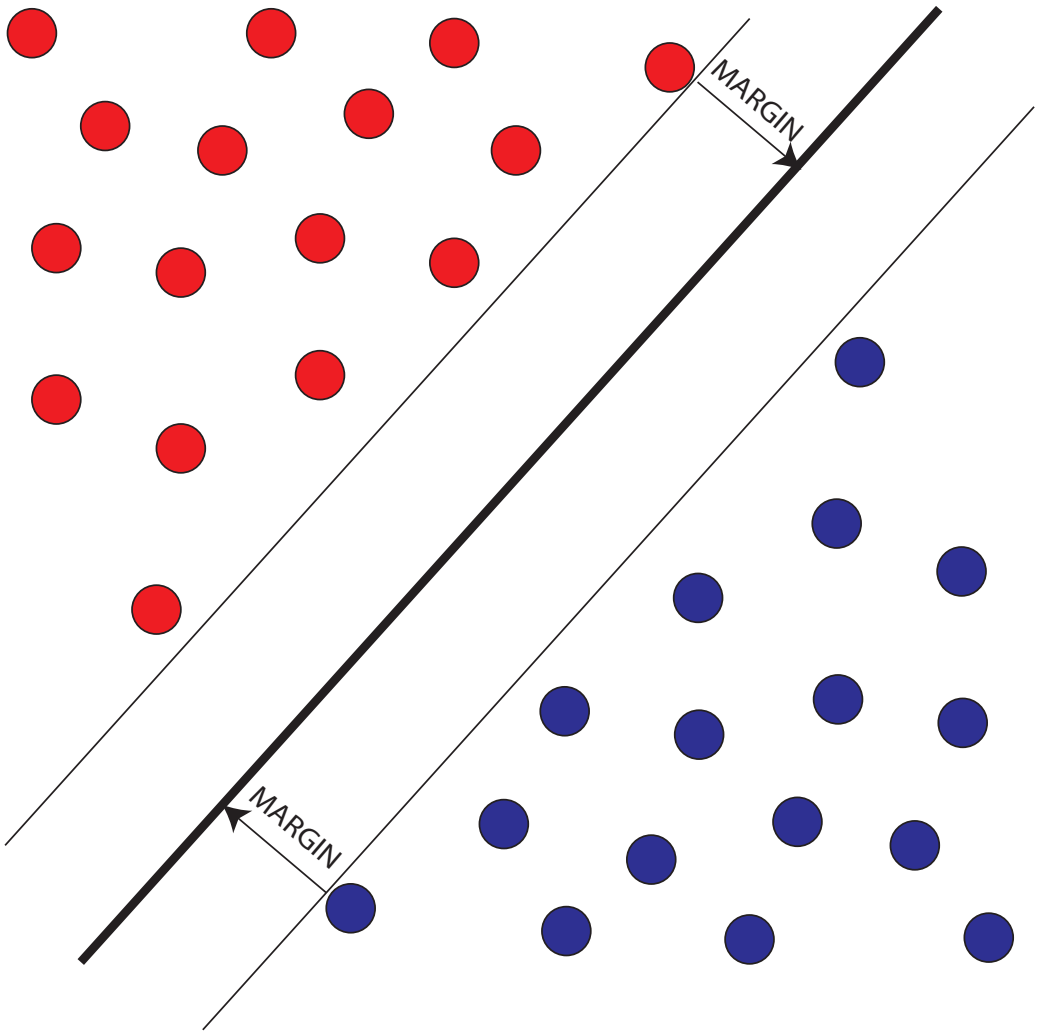
Objective #1: Maximize Margin



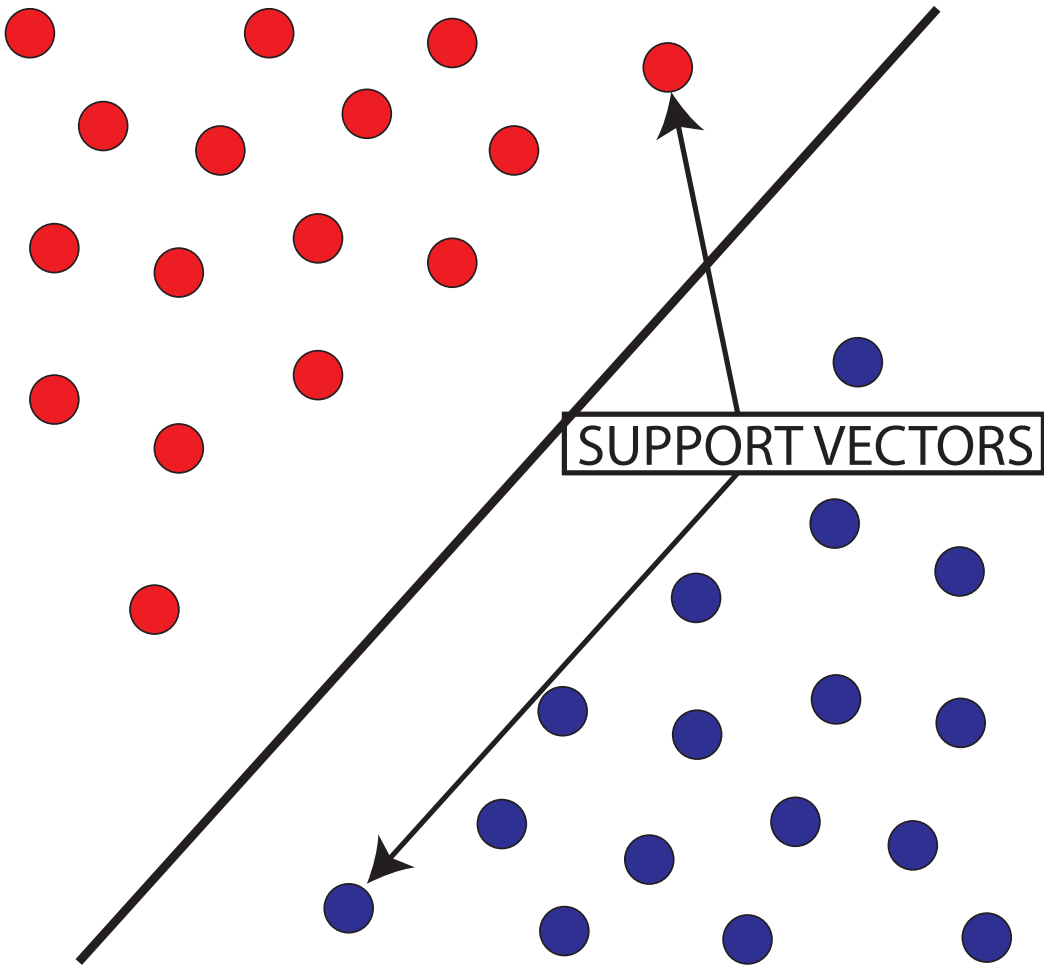
How's this look?



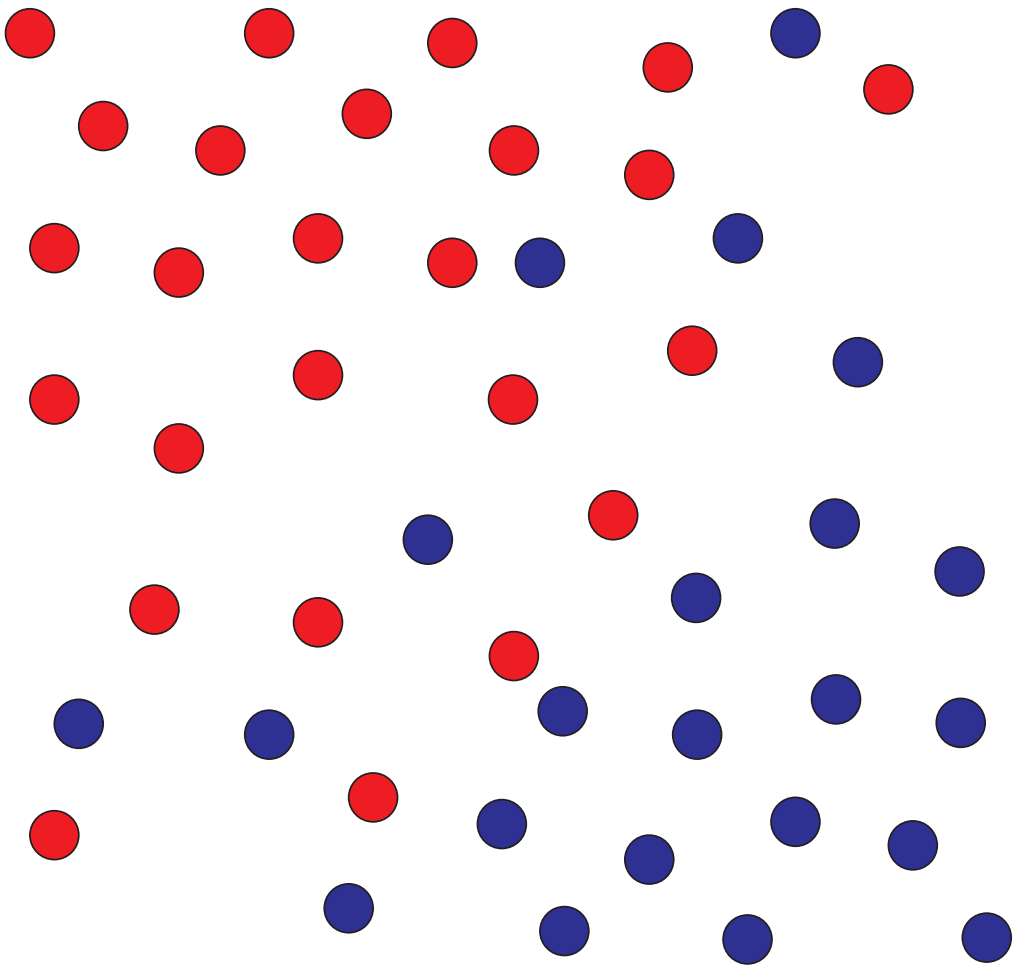
Objective #2:
Minimize Misclassifications



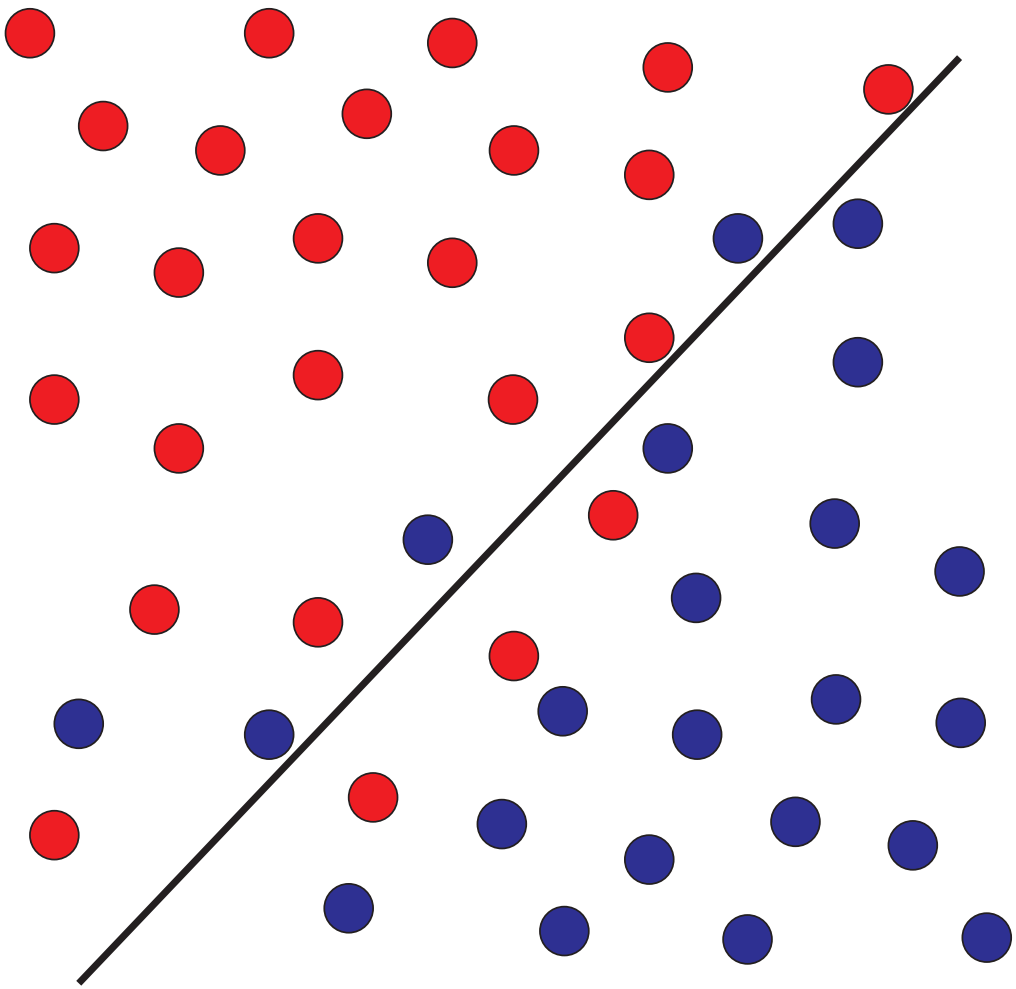
Support Vectors



Not Linearly Separable



SVM w/ “Soft Margin”



The model

- A hyperplane in $\mathbb{R}^n$ can be represented by a vector **w** with *n* elements (*n*=#variables), plus a **"bias"** term, w_0 which lifts it away from the origin.
  - $w_0 + \boldsymbol{w}^T \boldsymbol{x} = 0$ (equation of decision *boundary*)

- Any observation, **x**, 'above' the hyperplane has
  - $w_0 + \boldsymbol{w}^T \boldsymbol{x} > \mathbf{0}$

- Any observation, **x**, 'below' the hyperplane has
  - $w_0 + \boldsymbol{w}^T \boldsymbol{x} < \mathbf{0}$

The input…

- Input data and a class target.
- For best results, input data should be centered and standardized/normalized
- Hyperparameters for regularization and kernels.
  - (more on this in a minute…)

The output…

The output 'model' will be a set of parameters (i.e. a vector, $\mathbf{w}$, plus an intercept w_0)

For a new example, $\mathbf{x}$:

- If $w_0 + \boldsymbol{w}^T\boldsymbol{x} < \mathbf{0}$ then predict target $= -1$
- If $w_0 + \boldsymbol{w}^T\boldsymbol{x} > \mathbf{0}$ then predict target $= +1$

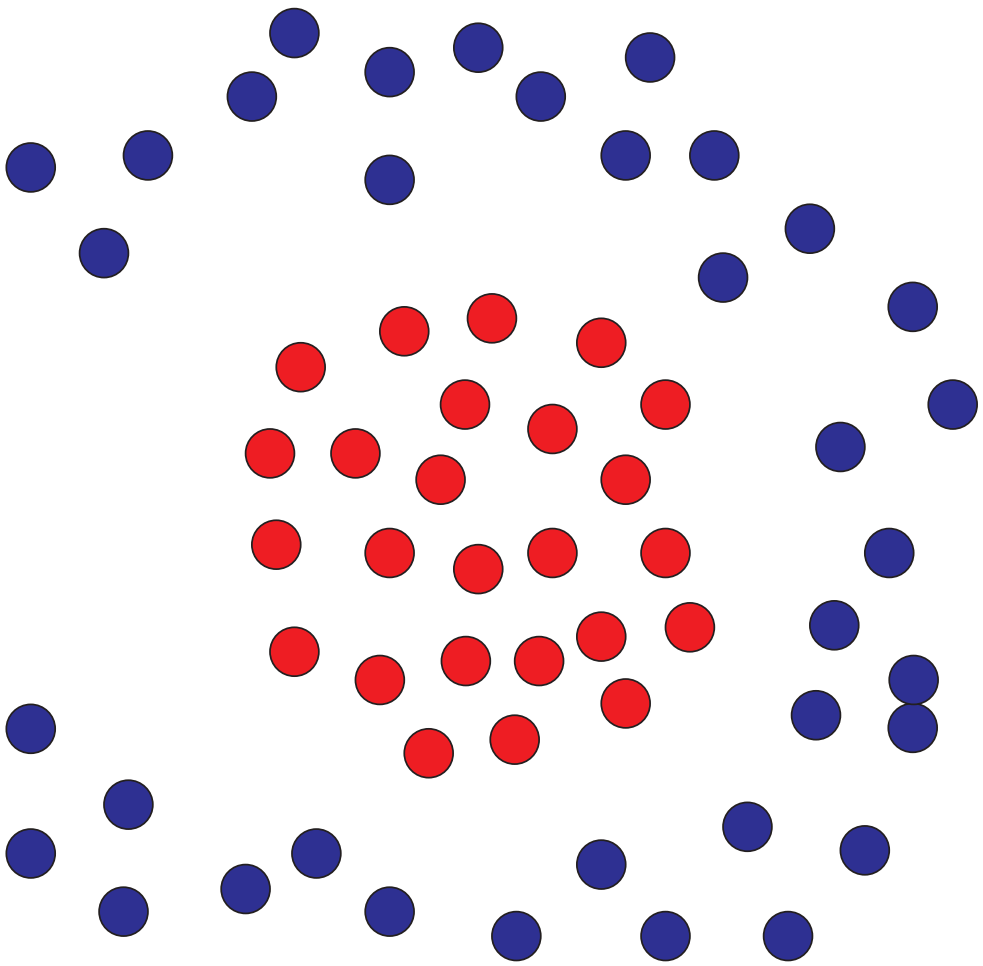
The above output changes when kernels are used, and it is best to use the model as an output object in that case.

Nonlinear SVMs

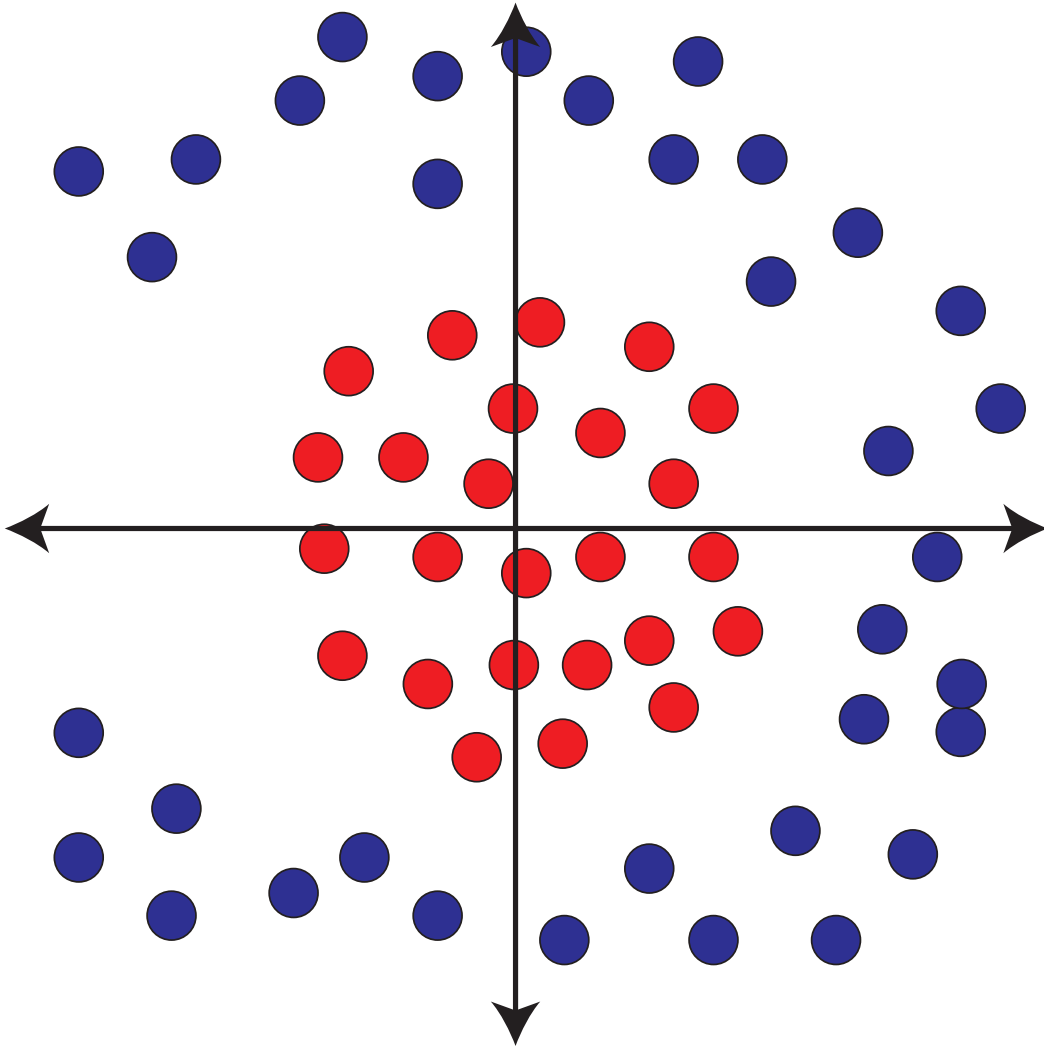
• • •

"The Kernel Trick"

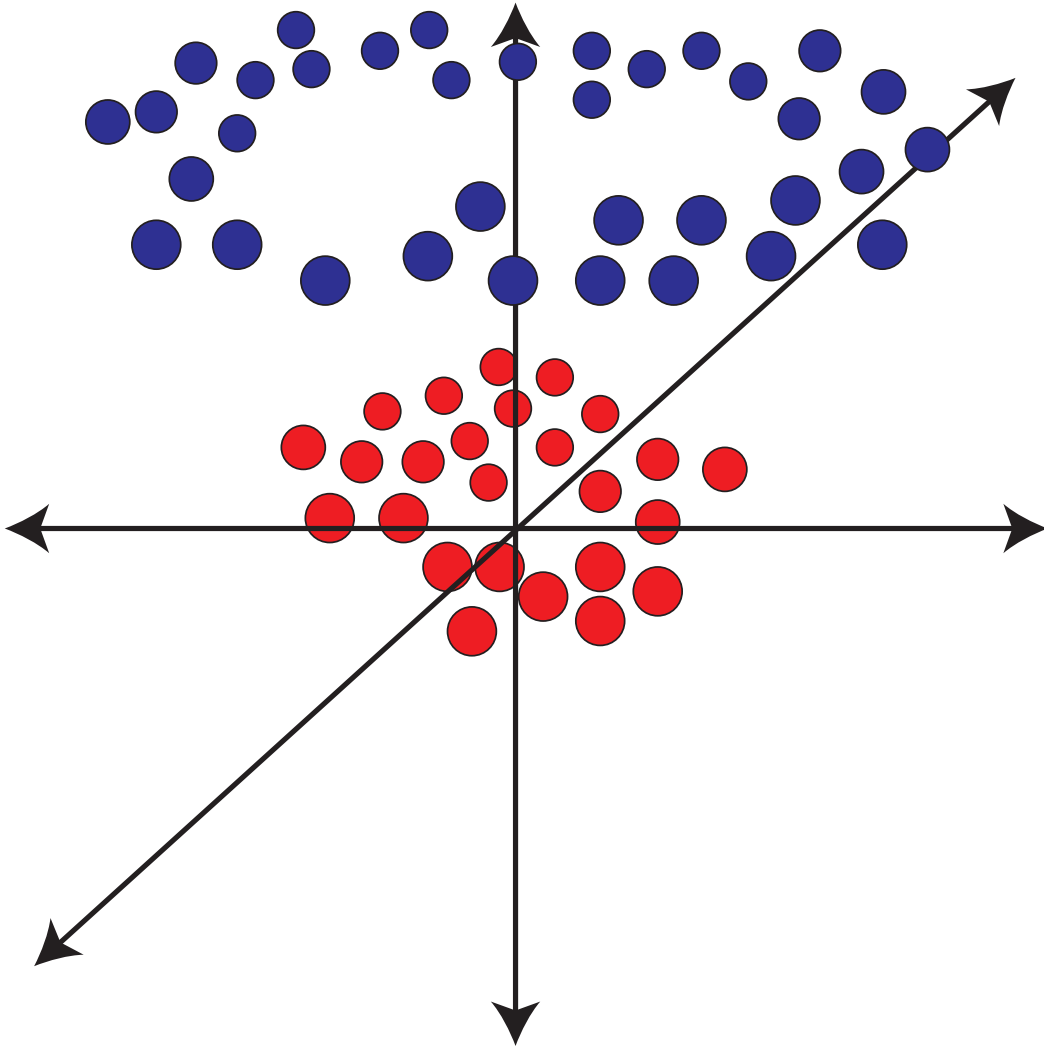
Not Linearly Separable



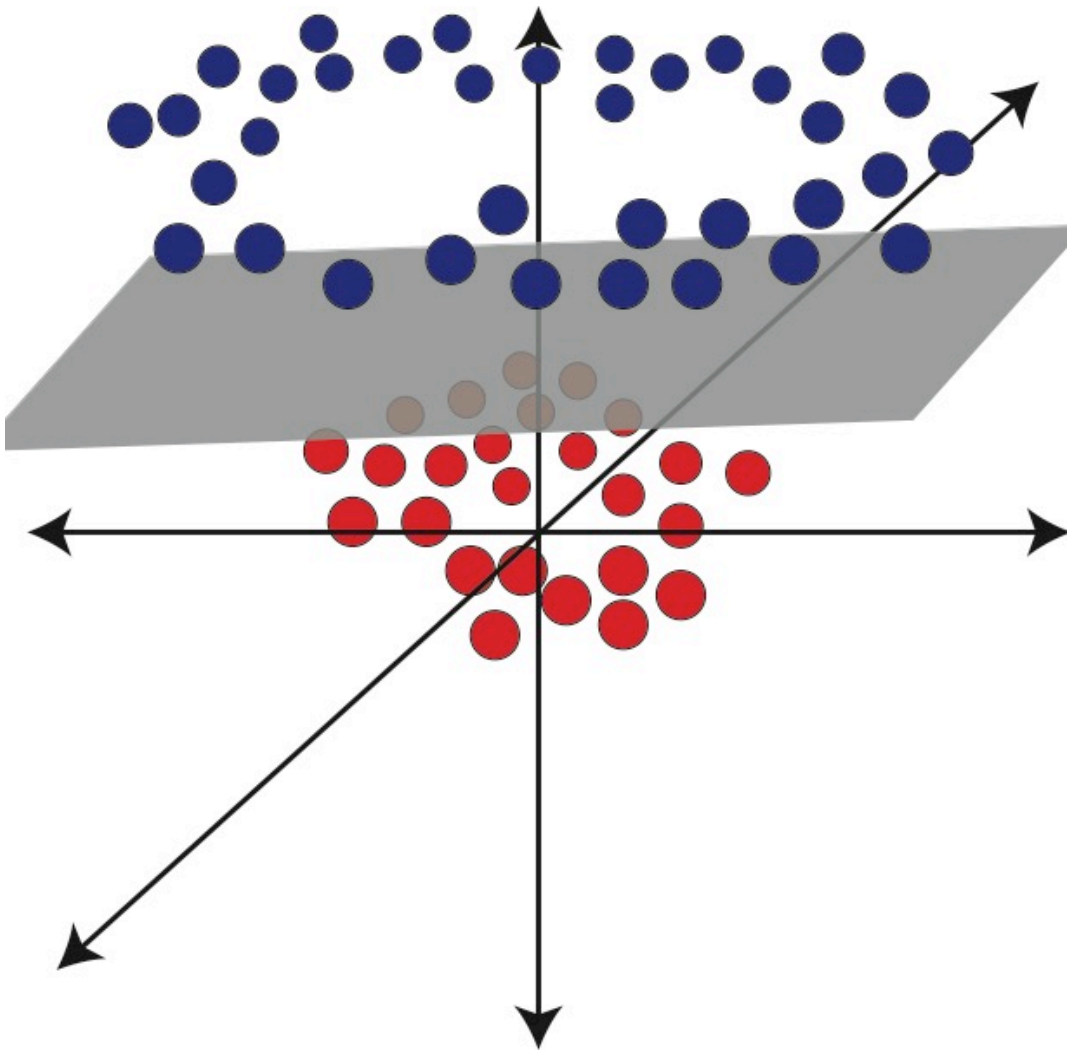
Create Additional Variables?



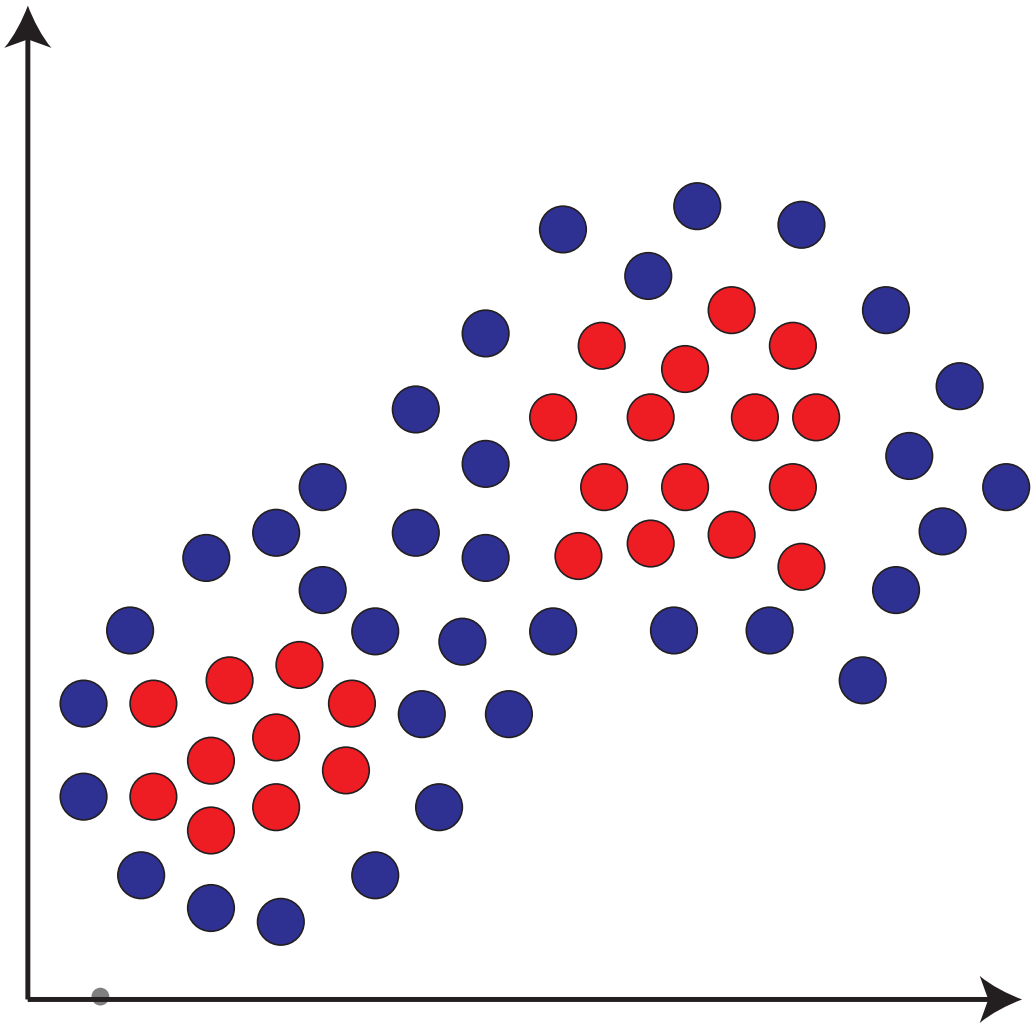
$$z = x^2 + y^2$$



New Data is Linearly Separable!



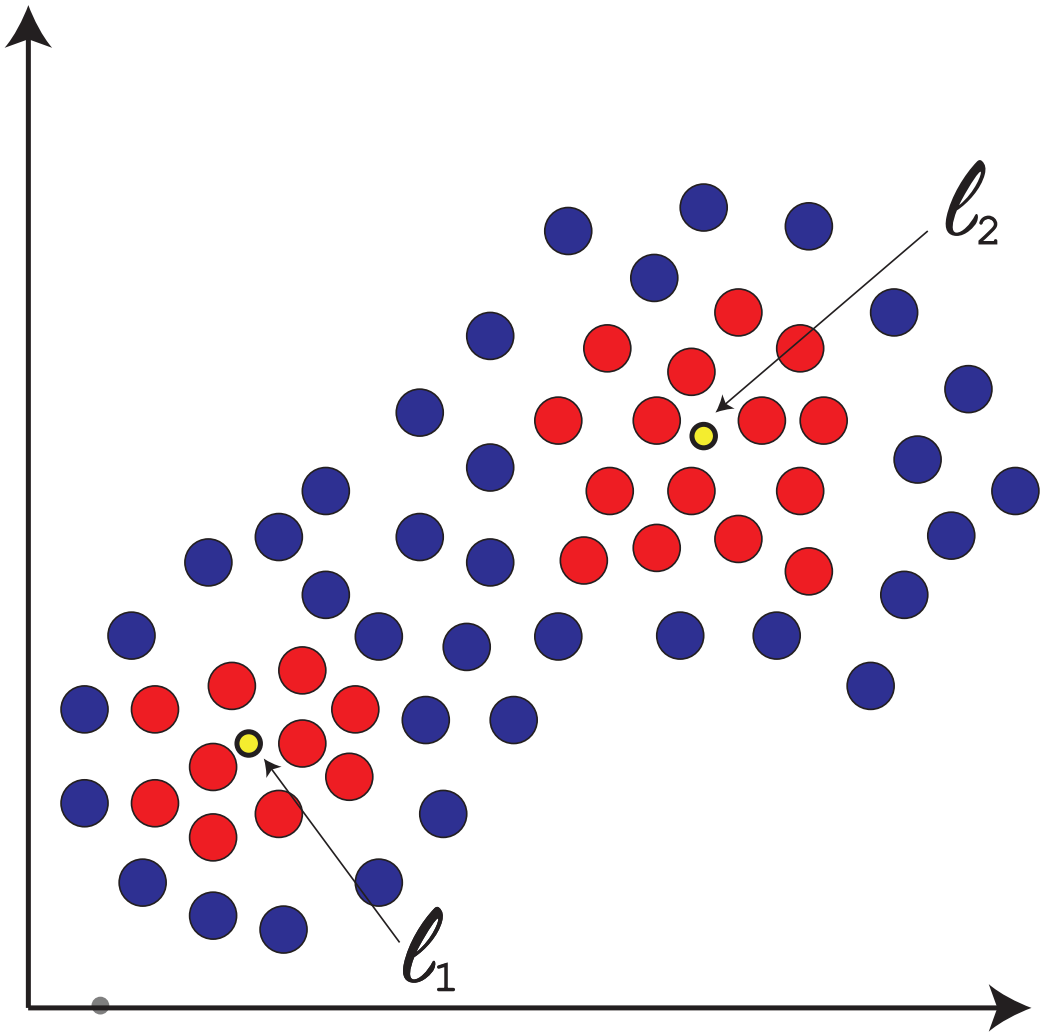
Another view…



The last 'trick' seems difficult in this case!

Not immediately clear what transformation will make this data linearly separable.

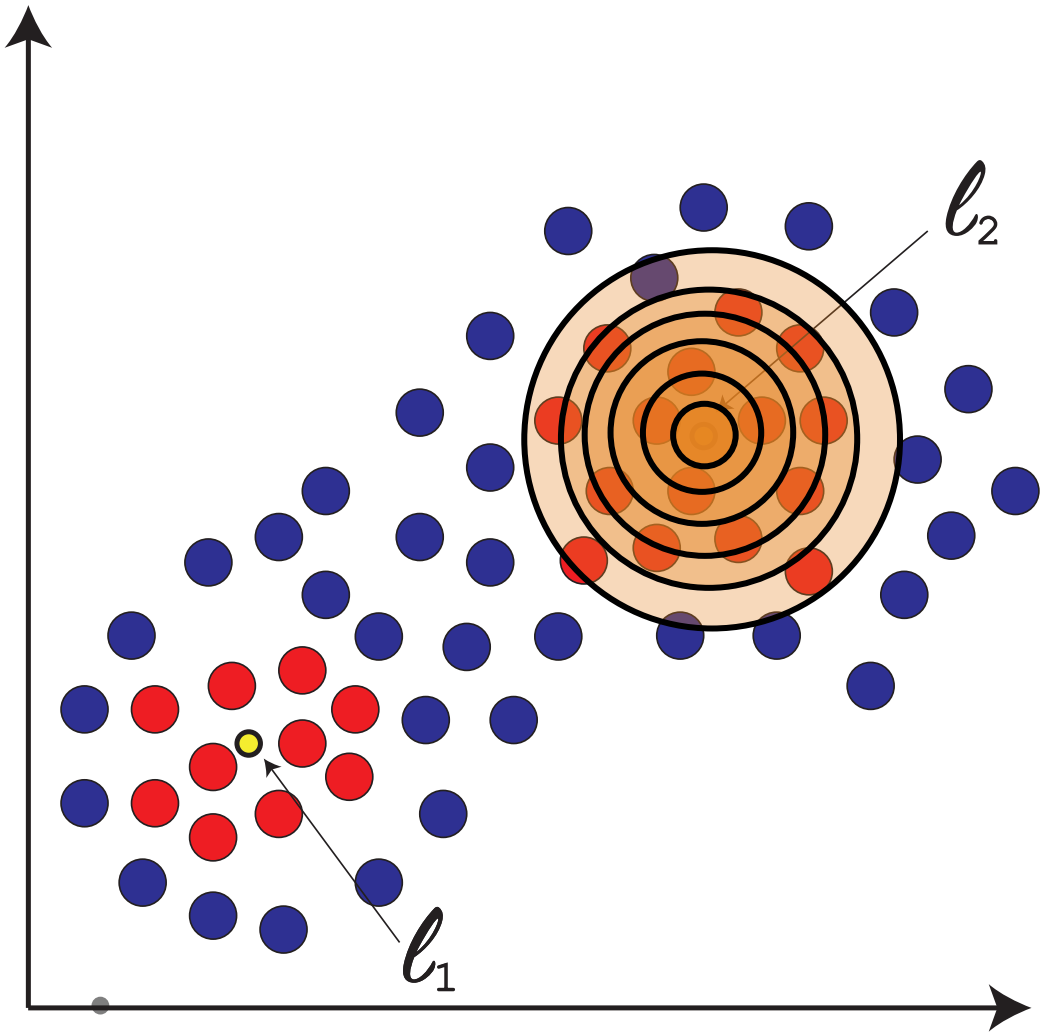
Kernels



- Suppose we add two points, which we'll call 'landmarks'.

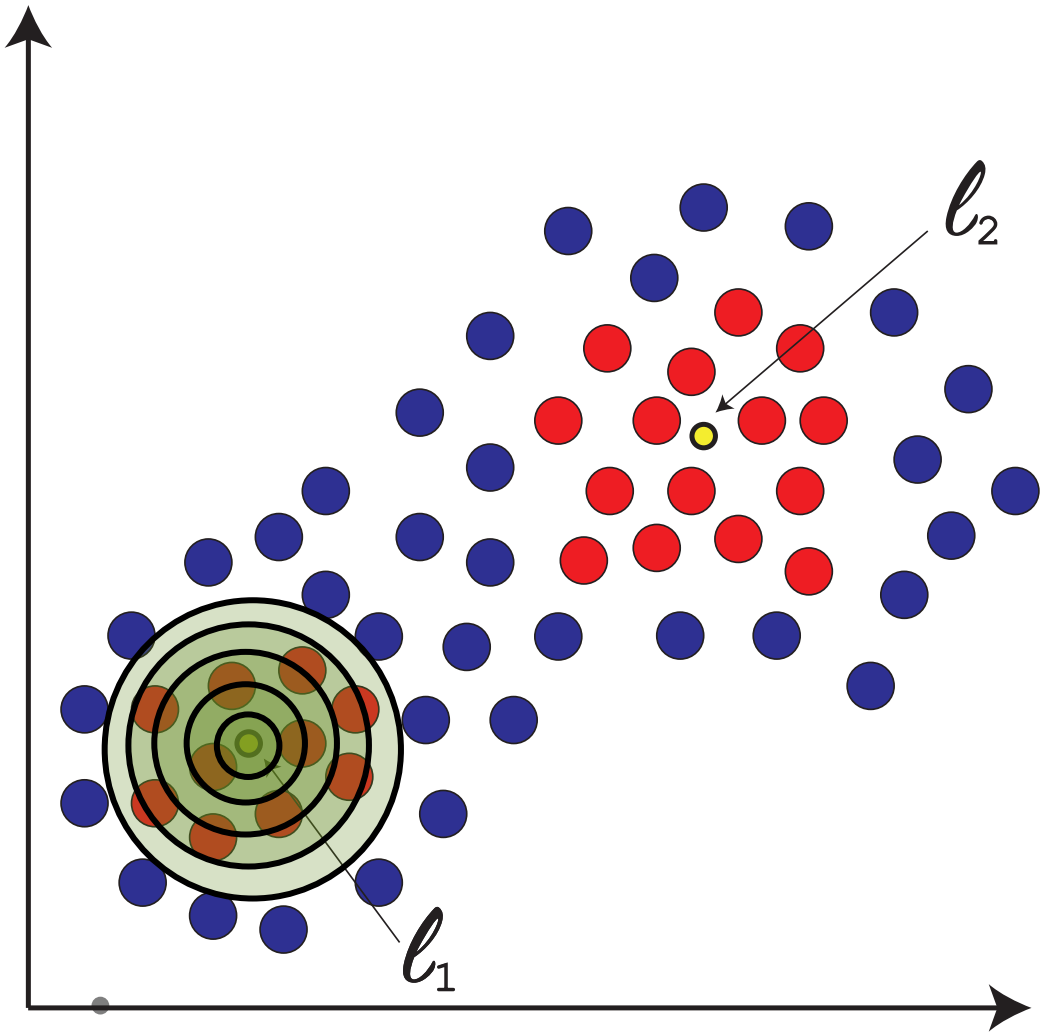
- Now suppose we create two new variables, f_1 and f_2, which measure the ***similarity*** of each point to those landmarks.

Kernels



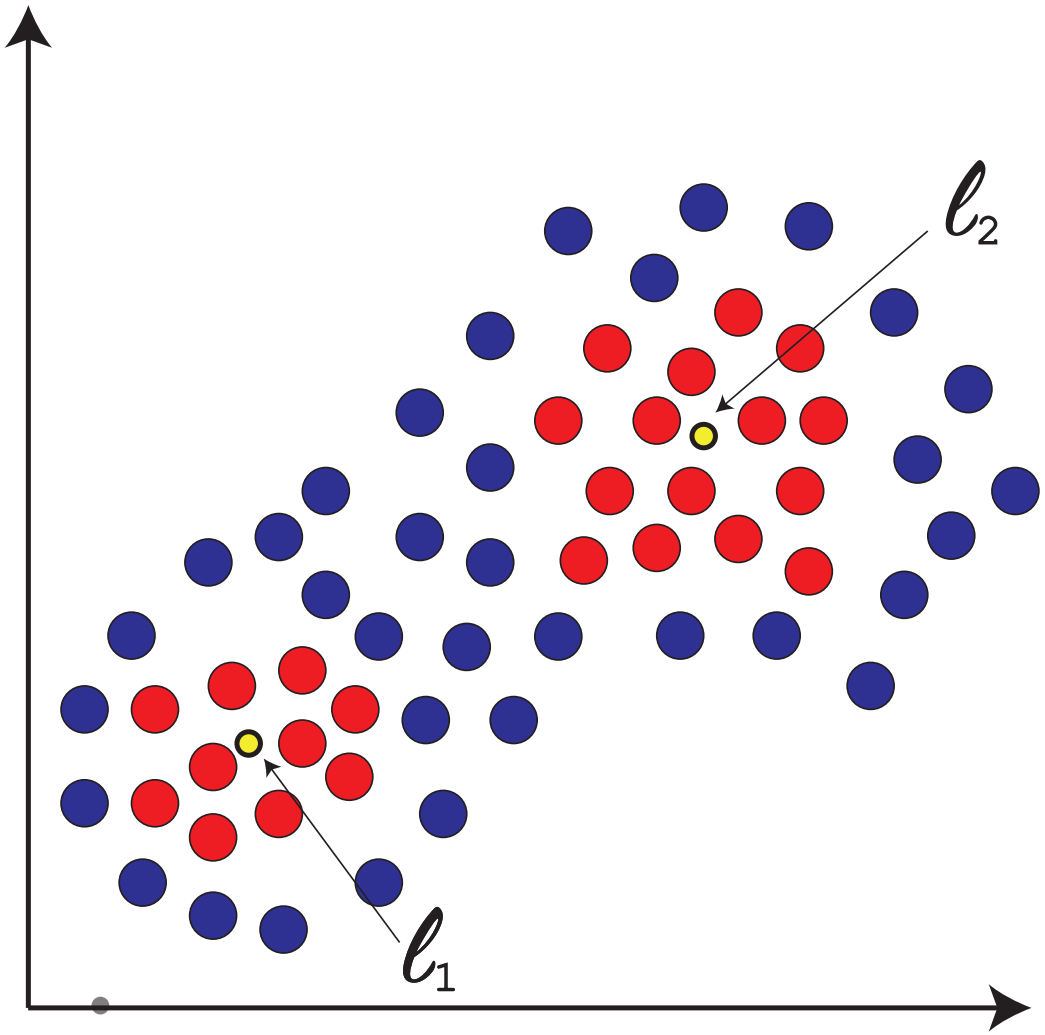
- f_1 is some measure of similarity (proximity) to l_1.
- It takes large values near l_1 and small values far from l_1.

Kernels



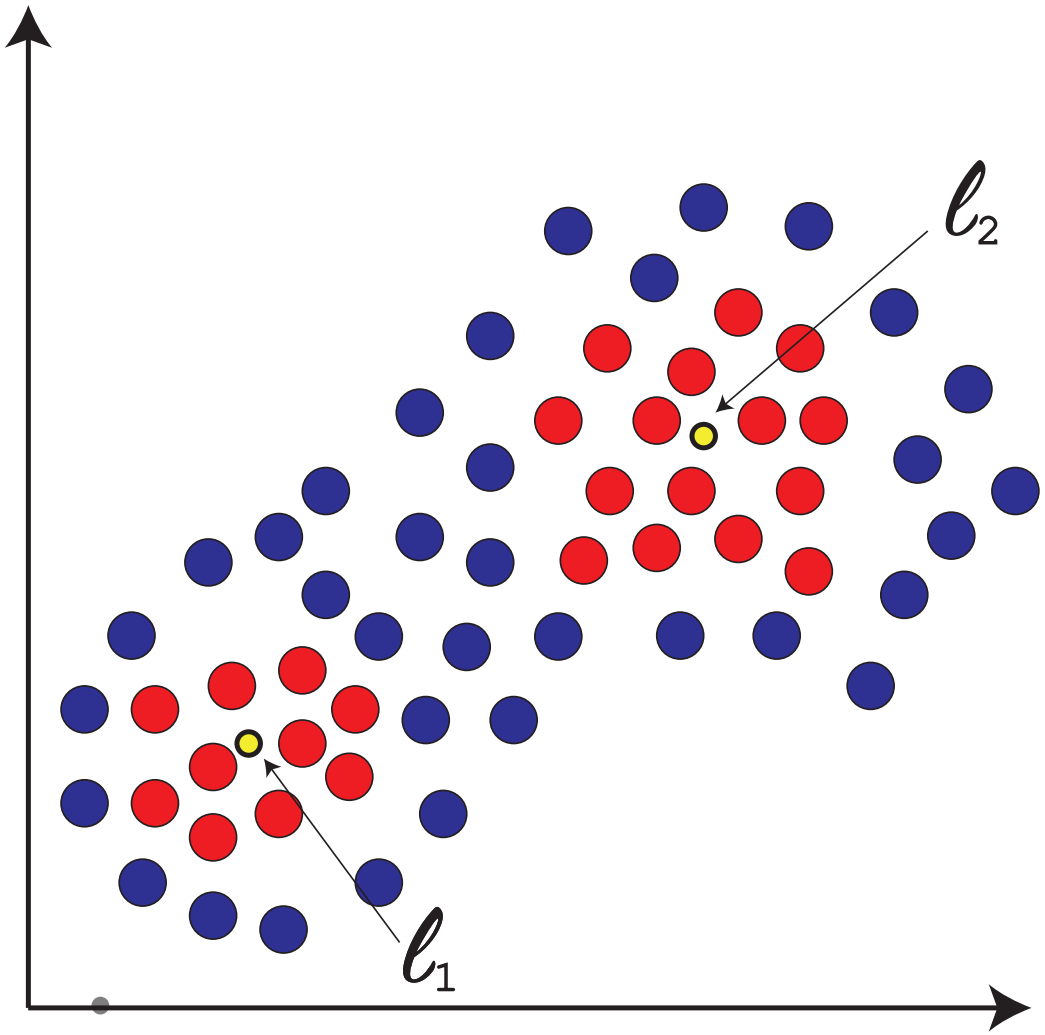
- f_2 is some measure of similarity (proximity) to l_2.
- It takes large values near l_2 and small values far from l_2.

Kernels



- Let's ignore our previous variables (the axis shown) and instead use f_1 and f_2.
- Where would the red and blue points be located if the axes were f_1 and f_2?
- Draw this picture

Kernels



- Next natural question – How do we choose the landmarks?
- You *could* choose a modest number of landmarks (using clustering or other methodology).
- In practice, a **kernel** uses *every* data point as a landmark.
- Essentially computes a similarity matrix to use in place of the data.

Summary of Kernels

- Kernels are similarity functions that measure some kind of proximity between data points.
- Number of data points becomes number of variables
  - So this is not good for large datasets! SAS has trouble running a kernel method with 50K data points!
- SVMs can use kernels in a very efficient way (similarity matrix never explicitly computed/stored).
- Kernels can improve the performance of SVMs in many situations.

Choosing Kernels

- Kernels embed data in a higher dimensional space (implicitly)
- Cannot typically know ahead of time which kernel function will work best
- Can try several, take best performer on validation data

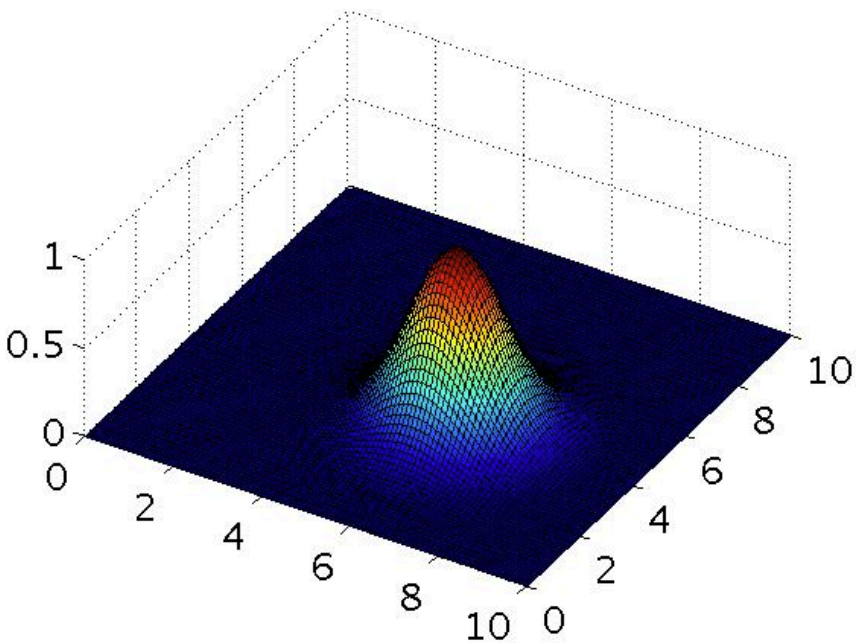
Popular Kernels

- Linear (➔ NO kernel)
- Radial Basis Functions (RBFs)
  - **Gaussian is most common and usually default**
  - $\exp\left(\frac{-\left\|x_i - x_j\right\|_2}{2\sigma^2}\right) = \exp\left(-\gamma \left\|x_i - x_j\right\|_2\right)$
  - $\gamma = \frac{1}{2\sigma^2}$ is hyper parameter controlling shape of function.
  - Some packages want you to specify gamma (γ).
    Some ask you to specify sigma (σ).
  - *NOT good for text classification. Typically linear is best for text*

RBF/Gaussian Kernel

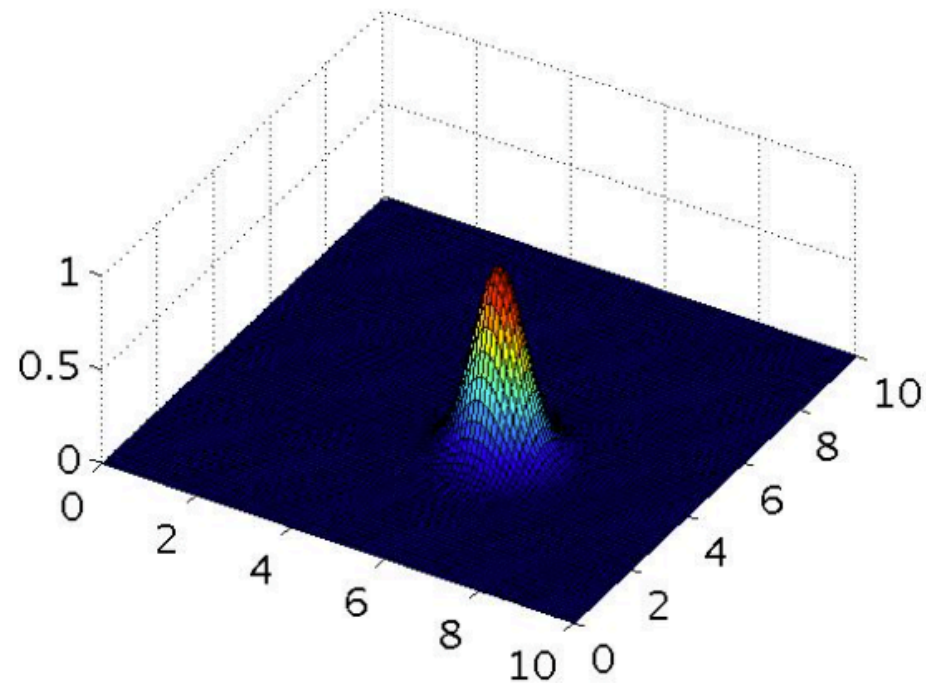
$$\exp\left(\frac{-||x_1-x_2||_2}{2\sigma^2}\right)$$

$\sigma = 1$

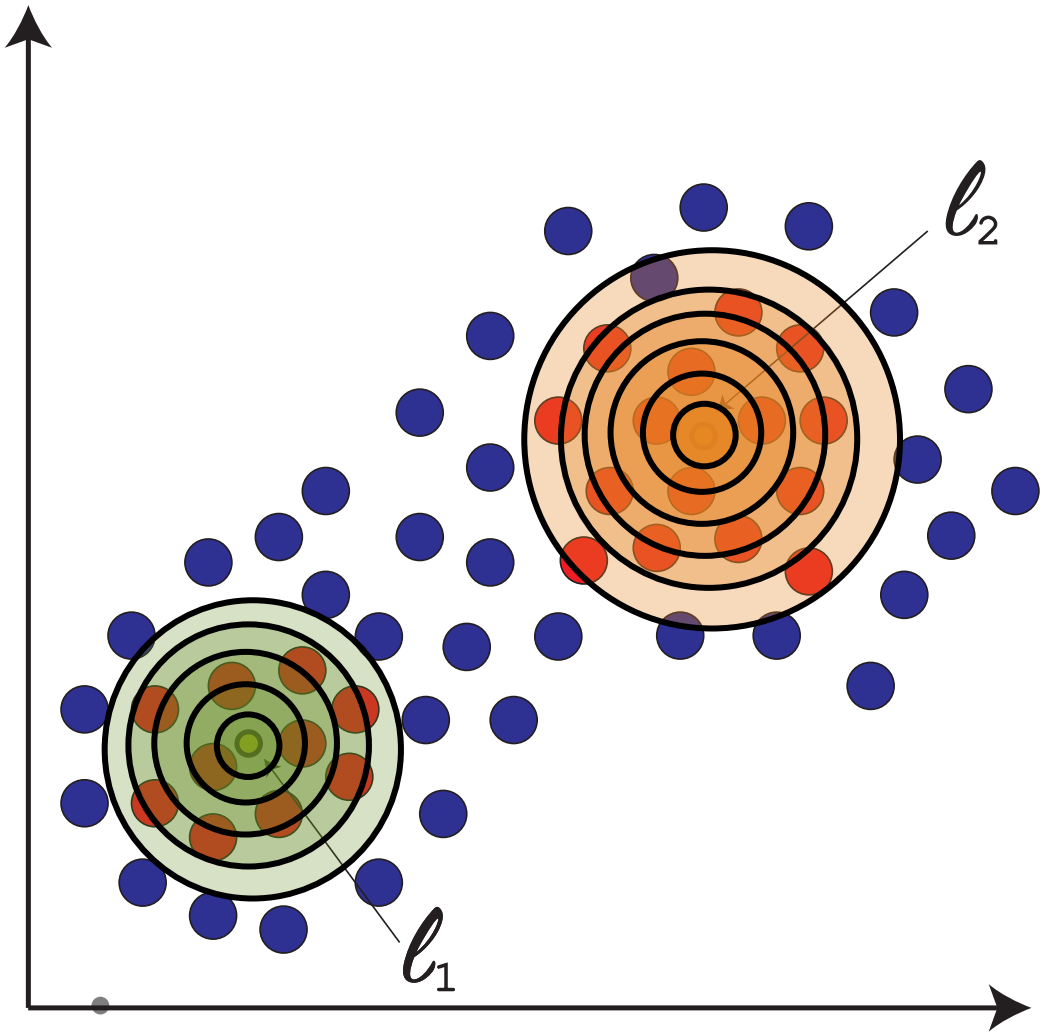


$$\exp\left(\frac{-||x_1-x_2||_2}{2\sigma^2}\right)$$

$\sigma = 0.5$



Kernels

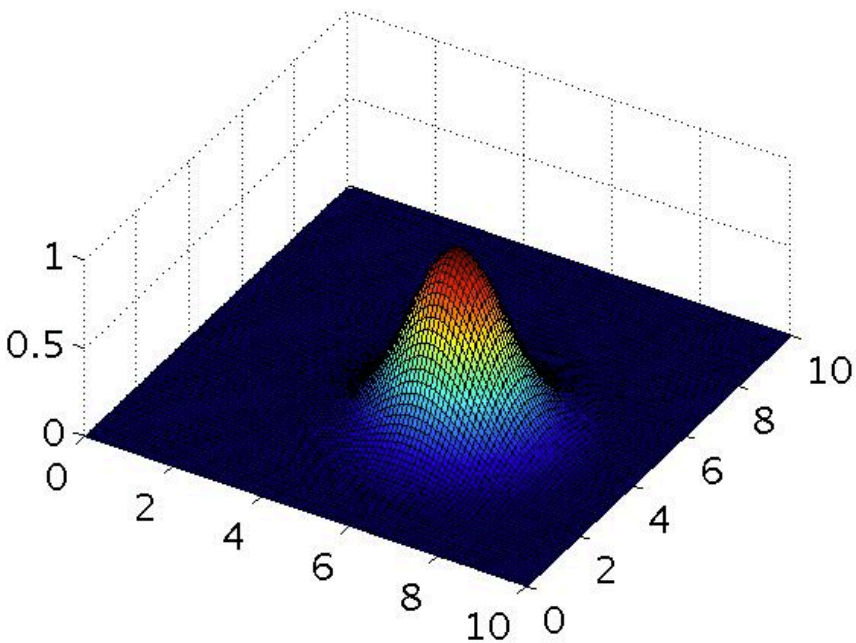


- The circles shown are meant to represent contours of those Gaussian functions.
- For which kernel fuction is σ larger, f_1 or f_2?
- (In practice, σ is the same for each point)

RBF/Gaussian Kernel

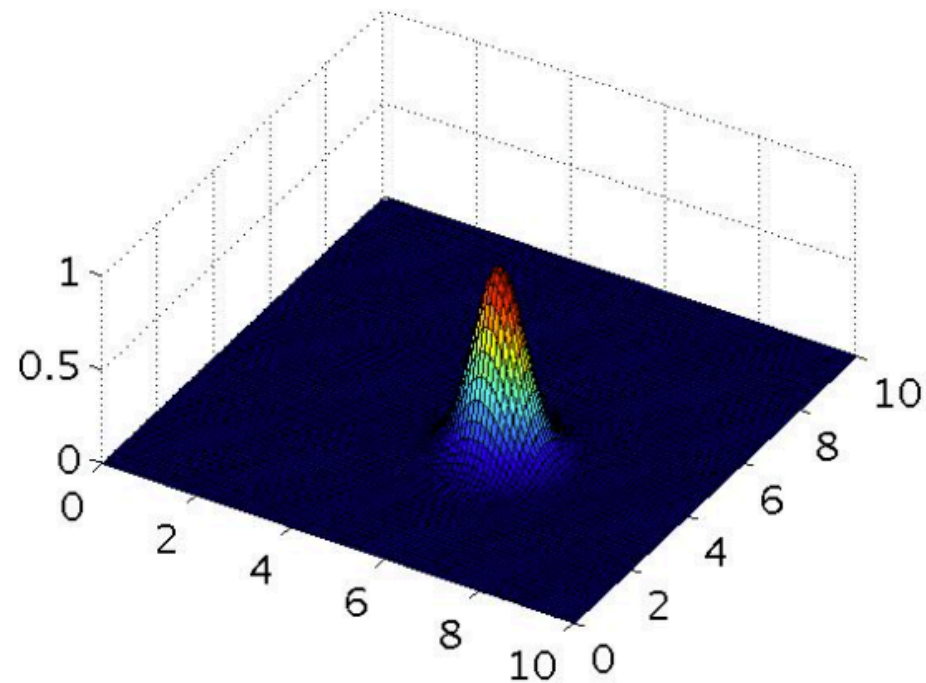
$$\exp\left(\frac{-||x_1-x_2||_2}{2\sigma^2}\right)$$

$\sigma = 1$



$$\exp\left(\frac{-||x_1-x_2||_2}{2\sigma^2}\right)$$

$\sigma = 0.5$



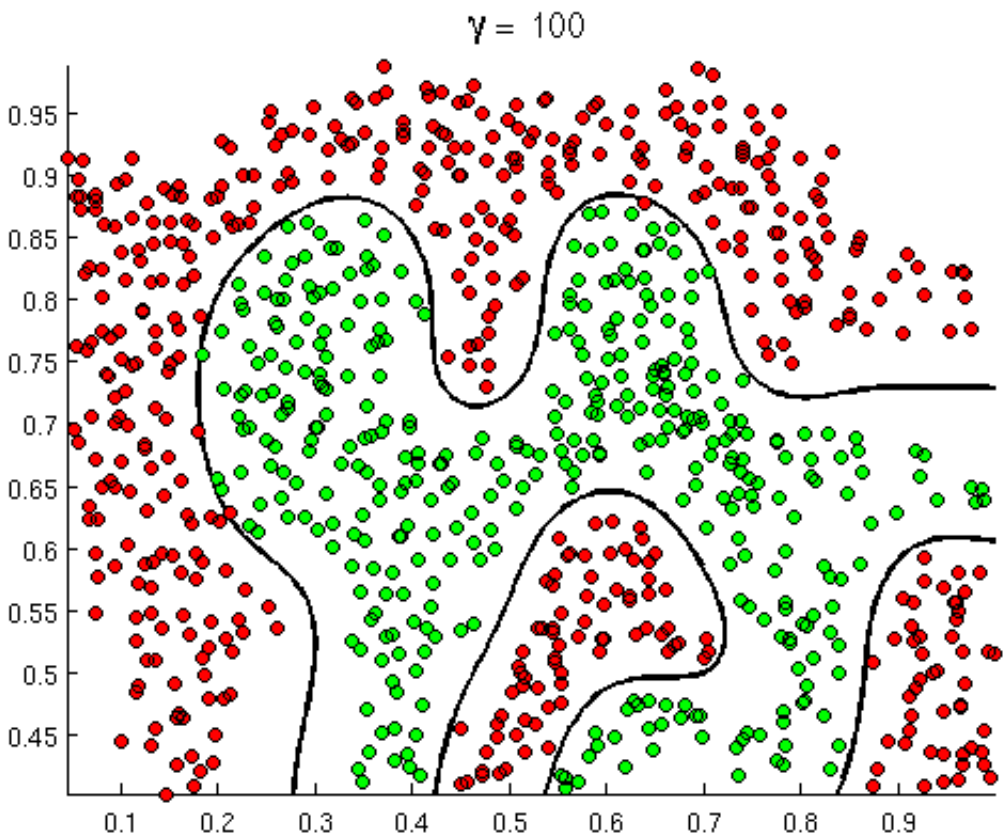
Tuning σ (or equivalently, γ)

- This hyperparameter controls the 'influence' of each training observation.
- A **larger value of σ** (equivalently, a smaller value of γ) means that basis functions are wider – the influence of a single point reaches far.
  - Smoother decision boundary **=> Reduce potential for overfitting.**
- A **smaller value of σ** (equivalently, a larger value of γ) means that basis functions are slimmer – the influence of a single point is more local.
  - More localized/jagged decision boundary **=> Overfitting more likely**
  - Consider: if σ were small enough, every point might be identified individually!

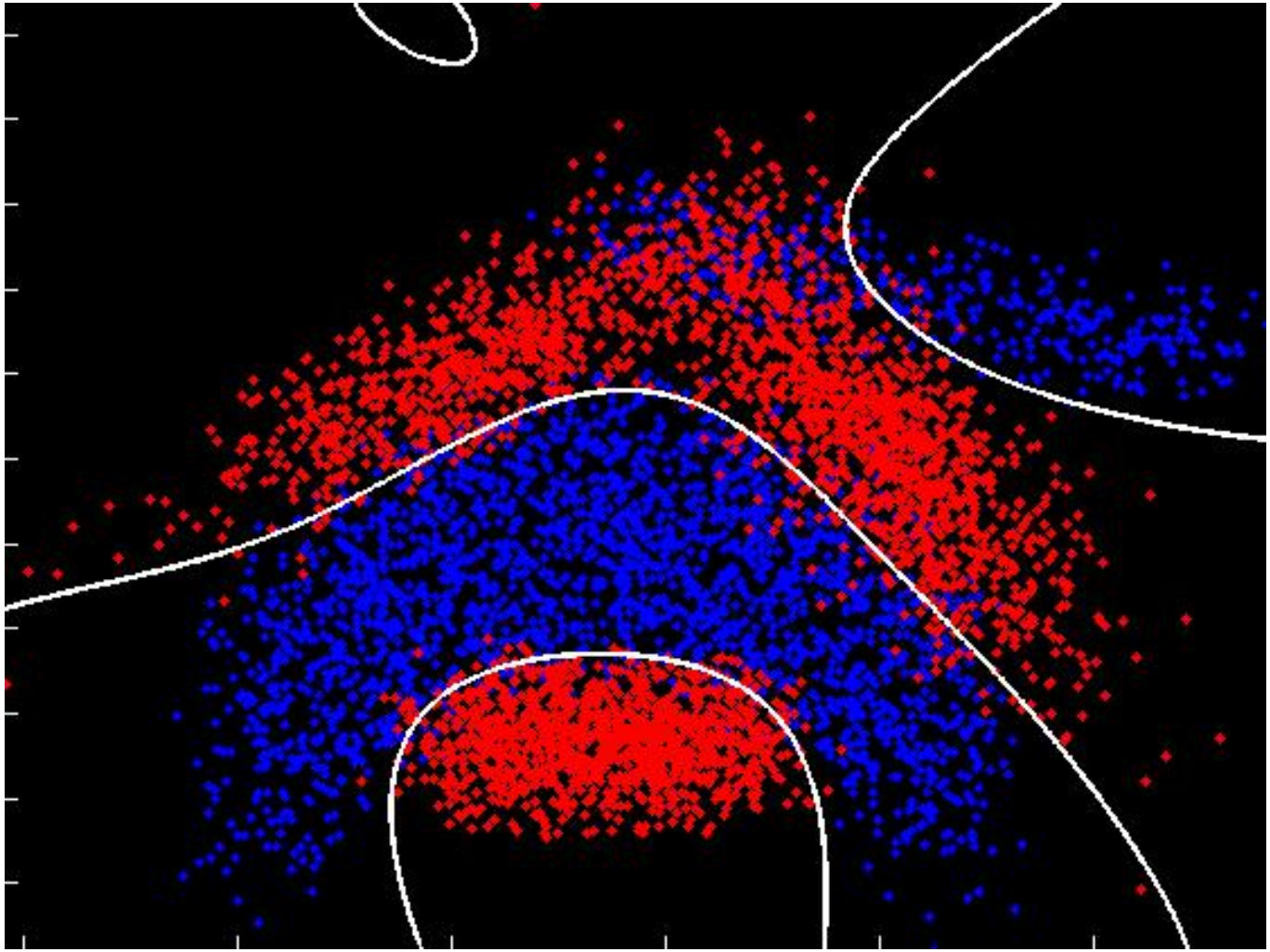
Other Kernels

- Polynomial
  - $\left(ax_i^T x_j + c\right)^d$ where a and c are constants and d is degree of polynomial
- Sigmoid
  - $tanh\left(ax_i^T x_j + c\right)$ where a and c are constants
- Both much less popular than linear/RBF

What kernels can do



What kernels can do



Regularization

- As with most machine learning algorithms, a *regularization penalty* on $\mathbf{w}$ can be added. ($\lambda||\mathbf{w}||$)

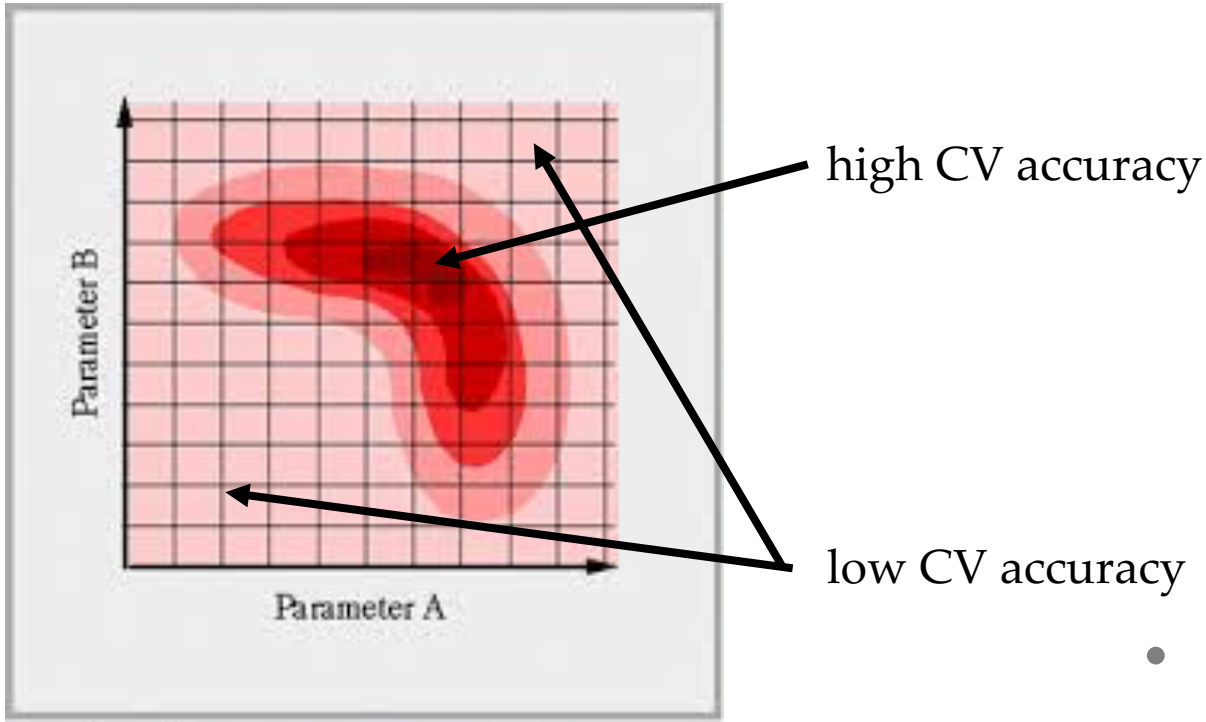
- Rather than specifying λ, SVMs are coded to expect

$$C = \frac{1}{\lambda}$$

  - C controls the tradeoff between a smooth decision boundary (bias/underfitting) and classifying training points correctly (variance/overfitting).
  - Larger C aims to classify all training points correctly.
  - Smaller C aims to make decision surface more smooth.

Tuning Hyperparameters

- How do we choose the *specific* values of the hyperparameters σ (or γ) and C?
- One option is a **grid search**. See how the algorithm performs for all combinations of σ and C within a certain range:



Extensions of SVMs

• • •

Multiclass classification

Regression

Multiclass Classification with SVM

- Most straightforward approach: **One vs. All method**
  1. Starting with k classes
  2. Train one SVM for each class, separating the points in that class (code as +1) from all other points (code as -1).
  3. For SVM on class i, result is a set of parameters $\boldsymbol{w_i}$
  4. To classify a new data point $\boldsymbol{d}$, compute $\boldsymbol{w_i^T}\,\boldsymbol{d}$ and place $\boldsymbol{d}$ in the class for which $\boldsymbol{w_i^T}\,\boldsymbol{d}$ is largest.
- This is still an ongoing research issue: how to define a larger objective function efficiently to avoid several binary classifiers.
- New methods/packages constantly being developed. Most existing packages ***can*** handle multiclass targets.

Support Vector Regression

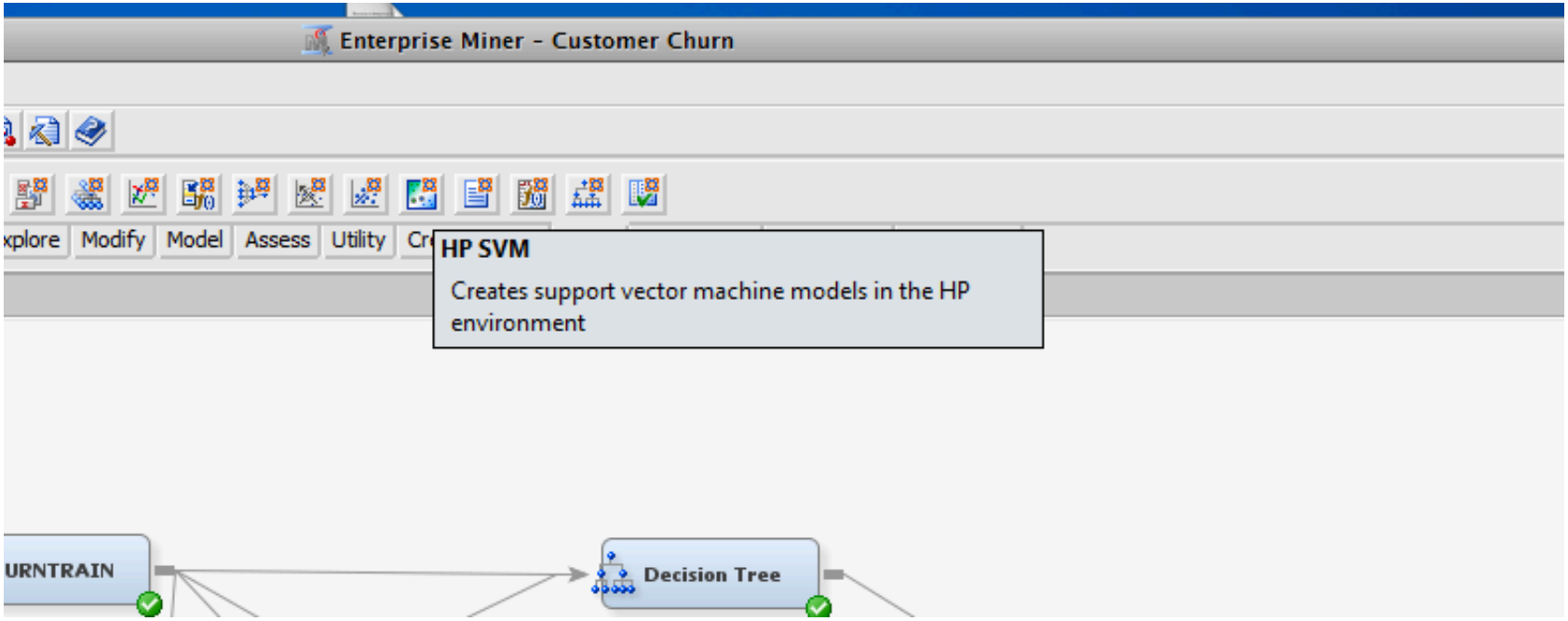
- The methodology behind SVMs has been extended to the regression problem.
- Essentially, the data is imbedded in a very high dimensional space via kernels and then a regression hyperplane is determined via optimization.

Creating an SVM in SAS EM

In my experience, this algorithm does not work as effectively as those implemented in R or Python. You also don't have the flexibility of hyperparameter tuning via cross validation.

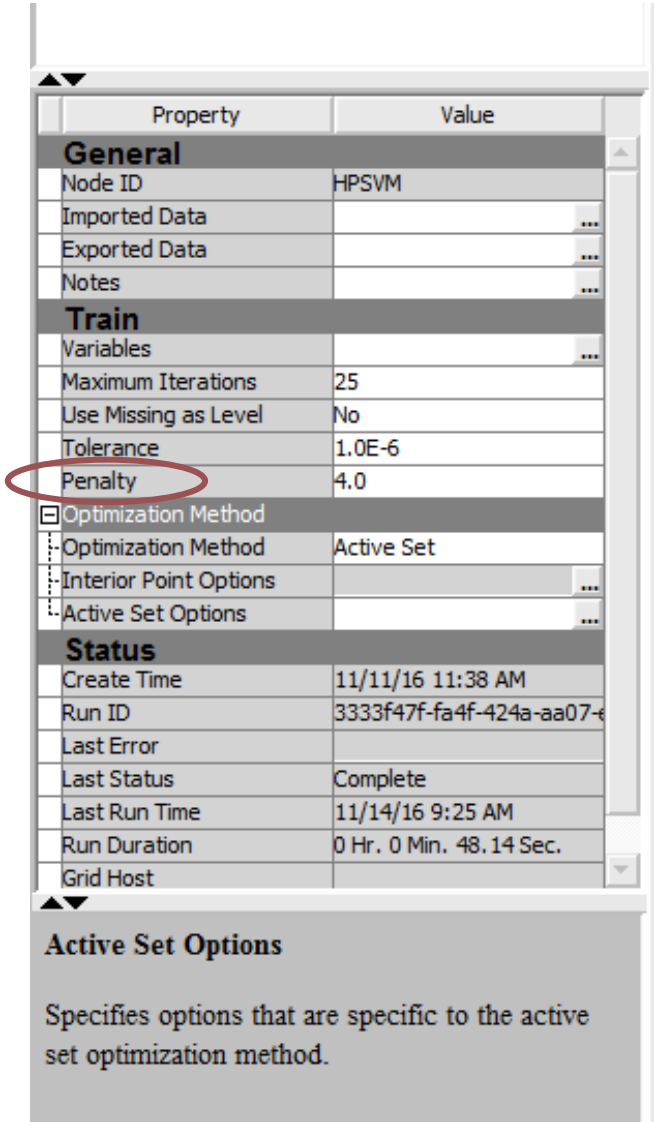
SVM in SAS EM

Under the HPDM tab, find HP SVM node



SVM in SAS EM

The parameter C is called the *Penalty* and is listed under the option panel "Train"



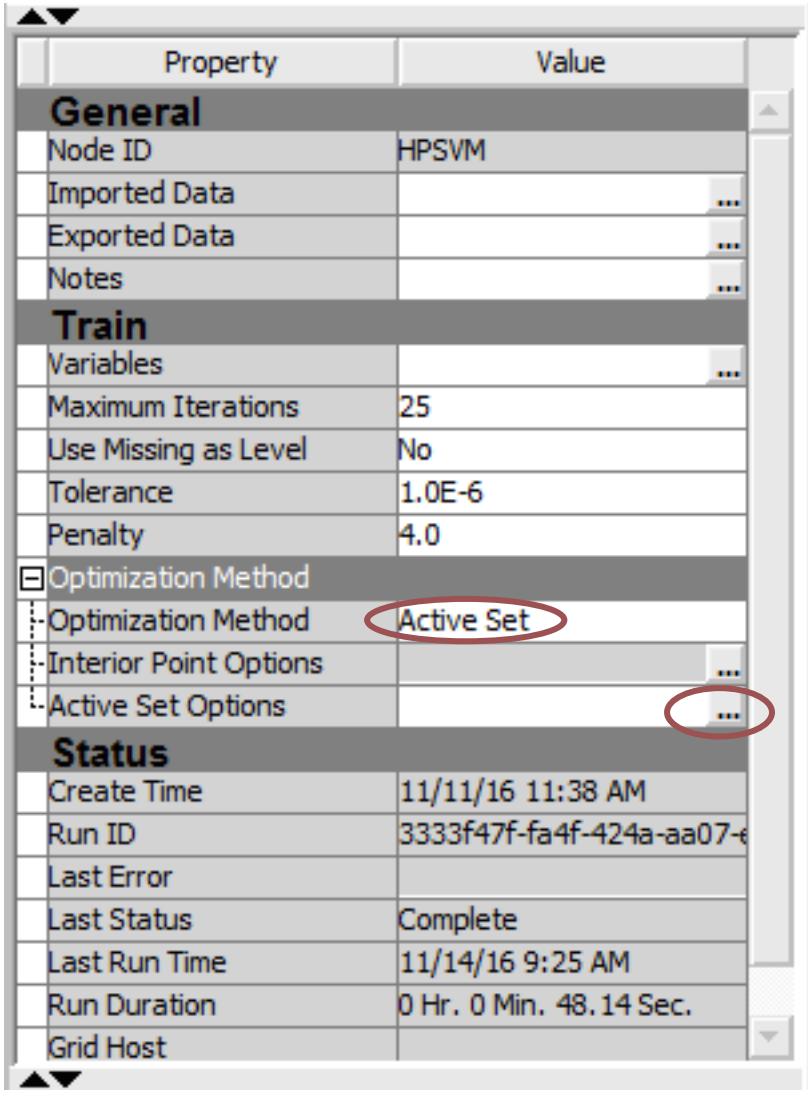
Property	Value
General	
Node ID	HPSVM
Imported Data	...
Exported Data	...
Notes	...
Train	
Variables	...
Maximum Iterations	25
Use Missing as Level	No
Tolerance	1.0E-6
Penalty	4.0
Optimization Method	
Optimization Method	Active Set
Interior Point Options	...
Active Set Options	...
Status	
Create Time	11/11/16 11:38 AM
Run ID	3333f47f-fa4f-424a-aa07-
Last Error	
Last Status	Complete
Last Run Time	11/14/16 9:25 AM
Run Duration	0 Hr. 0 Min. 48.14 Sec.
Grid Host	

Active Set Options

Specifies options that are specific to the active set optimization method.

SVM in SAS EM

To use SVM with kernels, change the optimization method to "Active Set" and click the ellipses for more options.



Property	Value
General	
Node ID	HPSVM
Imported Data	...
Exported Data	...
Notes	...
Train	
Variables	...
Maximum Iterations	25
Use Missing as Level	No
Tolerance	1.0E-6
Penalty	4.0
Optimization Method	
Optimization Method	Active Set
Interior Point Options	...
Active Set Options	...
Status	
Create Time	11/11/16 11:38 AM
Run ID	3333f47f-fa4f-424a-aa07-e
Last Error	
Last Status	Complete
Last Run Time	11/14/16 9:25 AM
Run Duration	0 Hr. 0 Min. 48.14 Sec.
Grid Host	

SVM in SAS EM

See the various options for the kernel used and the parameters. The parameter for the RBF kernel is gamma not sigma.