4: Linear Separators

CS1420: Machine Learning

Michael L. Littman
Spring 2018

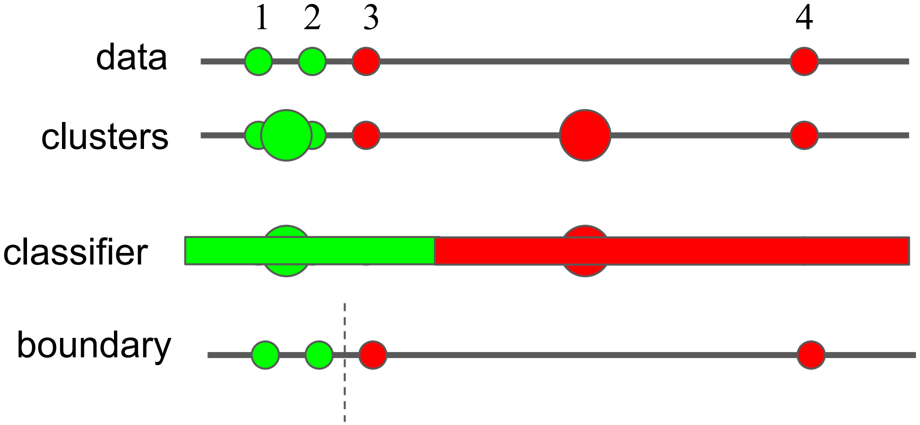
http://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/ . Chapter 9.1, 15, 16.

Generative Methods Struggle

If you focus on the central tendency of the class, it can be hard to get the boundary between classes right.

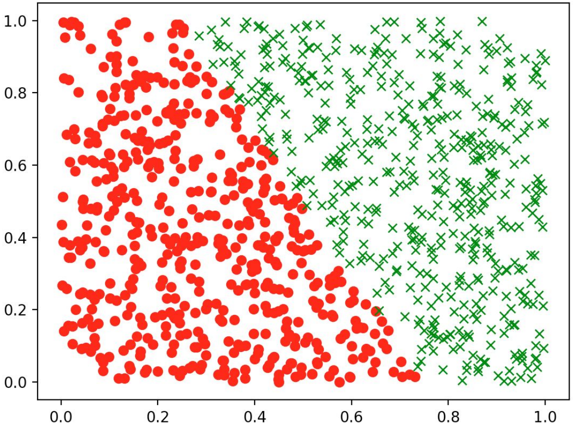
(Resulting classifier mislabels data point 3.)

Want to find the boundary that best *discriminates* between the classes.



Halfspace

$h_w(x) = 1$ if $<w,x> \geq 0$, 0 otherwise. (Assume we include a constant “1” column so the intercept can be learned.)



Data that can be labeled perfectly using a halfspace hypothesis is called *linearly separable*.

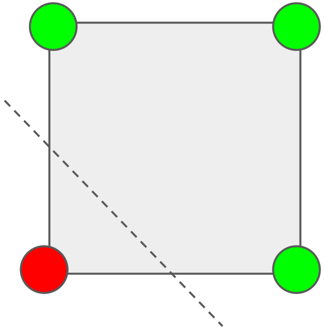
We’ll seek the ERM for linearly separable data. (Finding the ERM halfspace for non-linearly separable data is known to be hard, although logistic regression does ok by solving a different problem.)

VC dimension: d+1 for d dimensional space.

What’s Linearly Separable?

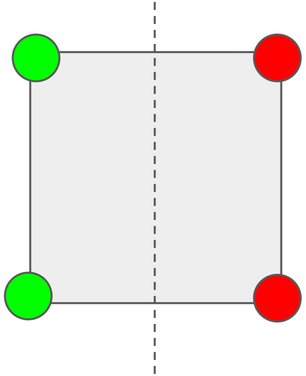
$f(x) = x^1 \text{ or } x^2$

$h(x) = x^1 + x^2 - ½ \geq 0$

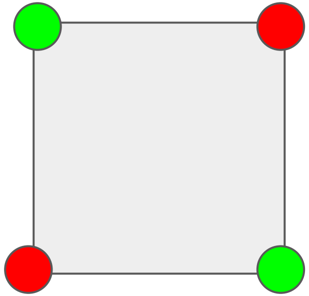


$f(x) = \text{not } x^1$

$h(x) = -x^1 + 0\,x^2 + ½ \geq 0$



But Not Exclusive Or



$f(x) = x^1 \text{ xor } x^2$

$h(x) = w^1 x^1 + w^2 x^2 + b \geq 0$

`00:` $b < 0$ (or $-b > 0$)

`10:` $w^1 + b \geq 0$

`01:` $w^2 + b \geq 0$

`11:` $w^1 + w^2 + b < 0$

Sum of first three constraints gives: $w^1 + w^2 + b > 0$, contradiction!

Perceptron Training Algorithm

```
$w$ = [0,0,0,...,0]; α = 1
done = False
while not done:
    done = True
    for $(x,y)$ in $S$:
        $y^{hat}$ = $<w,x> \geq 0$
        $w = w + α(y-y^{hat})\ x$
        if $|y-y^{hat}|=1$: done = False
```

Get a point that is misclassified and add it (signed) to the weights.

Super simple! Converges! (Can be slow in the worst case.)

Delta Rule

Another approach. Want w such that $<w,x>$ is kind of like y.

Define loss to be the total squared difference: $\frac{1}{2} \sum_{(x,y)\in S} (y-<w,x>)^2$.

How can we choose w to optimize this loss?

What's the derivative with respect to weight w_i? $-\sum_{(x,y)\in S}(y-<w,x>)x_i$

Second derivative? $\sum_{(x,y)\in S} x_i^2$. Always positive!

Gradient descent: For $(x,y)\in S$: $w = w + \alpha(y-<w,x>)\ x$. Very similar to Perceptron rule!

Direct Minimization of Delta Rule Loss

$\frac{1}{2} \sum_{(x,y) \in S} (y - \langle w,x \rangle)^2$

Rewrite in matrix form. Let the rows of X be the input vectors in the training set. Let Y be a column vector of the target labels. Let w be a column vector of the weights. Then, we have Xw trying to match Y. Via least squares:

$Xw \cong Y$
$X^T Xw \cong X^T Y$
$(X^T X)^{-1} X^T Xw \cong (X^T X)^{-1} X^T Y$
$w = (X^T X)^{-1} X^T Y$

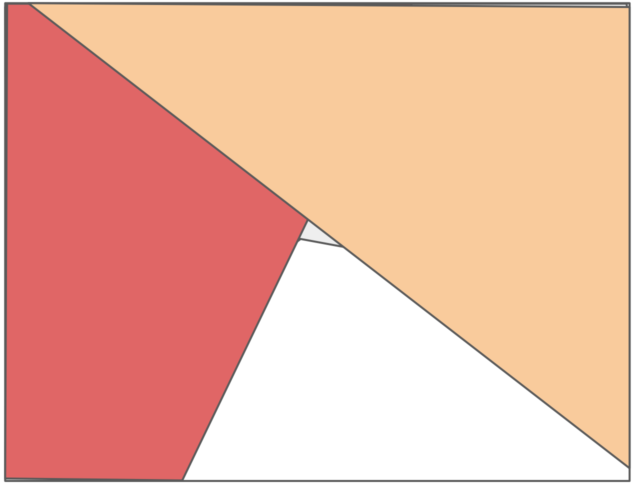
No search, no iteration. The power of Gaussian elimination.

Animation Example

https://www.youtube.com/watch?v=xpJHhHwR4DQ

Linear Programming

Find the best point w in a d-dimensional space where the objective is to maximize a linear scoring function on w ($<u,w>$) where w is subject to a set of linear constraints (halfspaces!).



$\max_w <u,w>$ s.t. $Aw \geq v$.

Comes up in zillions of settings:

- Operations research
- Game theory
- Decision theory

Efficiently solvable, even large scale, in theory and practice.

Finding a Linear Separator via LP

Want $h_w(x) = 1$ if $<w,x> \geq 0$, 0 otherwise.

Have sample S of x,y pairs. Each pair provides us with a constraint on w: If y=1, we want $<w,x> \geq 0$, if y=0, we want -$<w,x> \geq 0$.

Note that w are the variables and x and y are constants. So, these constraints are *linear* in the variables.

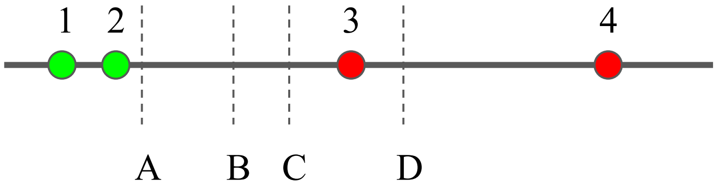
Any w that satisfies our set of linear constraints defines an ERM for the sample, so we can set any objective: say, u=[0,0,0,...,0].

Best of the Best?

Are all ERM classifiers created equal?

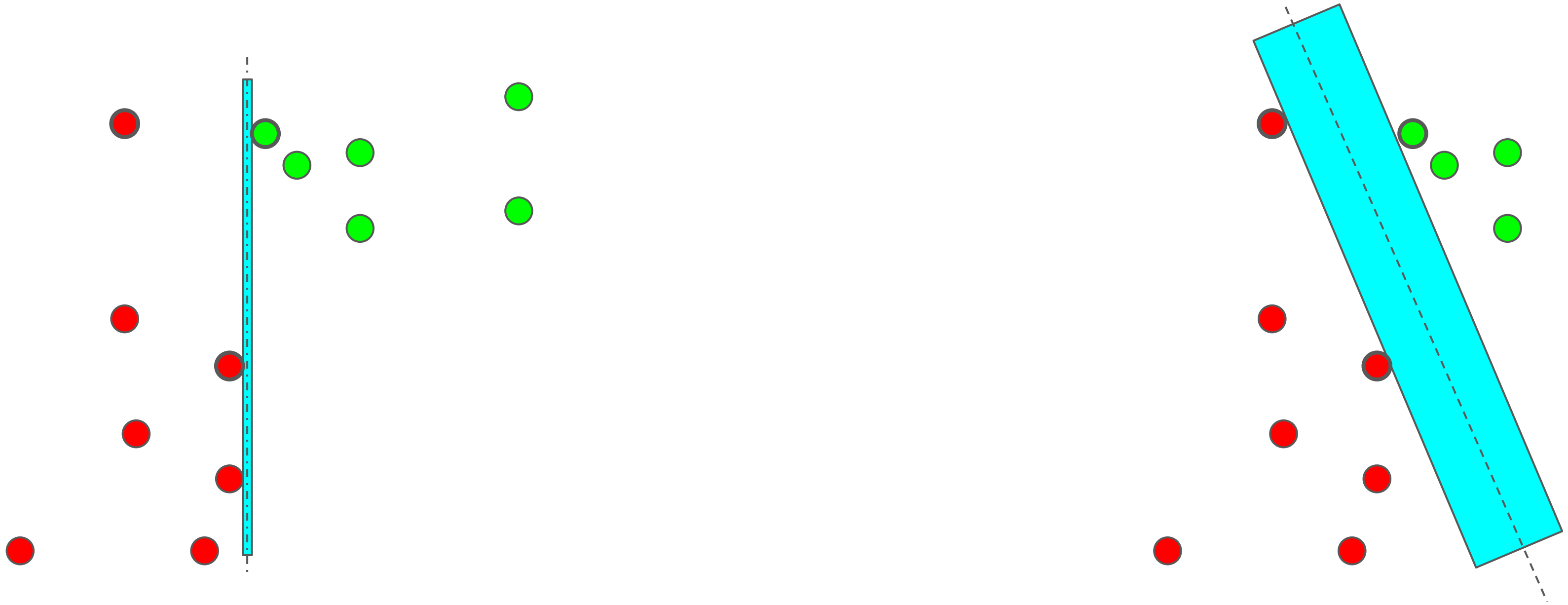
Which do you think ought to lead to the best generalization to new data?

A. A.
B. B.
C. C.
D. D.
E. All equivalent



Margin

Pick the separating line with the biggest margin of error, relative to perturbations of the points.



Calculating the Margin

Given a hyperplane, $h_w(x) = 1$ if $<w,x> \geq 0$, 0 otherwise, we can limit w so that $||w|| = 1$. For w', define $w = w'/||w'||$. (If $w'=0$, it's not a separator at all.)

Then, $||w|| = 1$ and $<w',x> \geq 0$ if and only if $<w,x> \geq 0$.

The distance between a point x and the hyperplane defined by w ($||w|| = 1$) is $|<w,x>|$.

Hard Support Vector Machine

Find the separating hyperplane that is farthest from its closest point:

$\mathrm{argmax}_{w:\|w\|=1}\ \min_{(x,y)\in S} |<w, x>|$ s.t. $\forall_{(x,y)\in S}\ (2y\text{-}1)<w, x> > 0.$

Equivalently:

$\mathrm{argmax}_{w:\|w\|=1}\ \min_{(x,y)\in S}\ (2y\text{-}1)<w, x>.$

More Rewriting

$\text{argmax}_{w:||w||=1} \min_{(x,y)\in S} (2y\text{-}1)<w, x>$.

$\text{argmax}_{w:||w||=1} M$ s.t. $\forall_{(x,y)\in S} (2y\text{-}1)<w, x> \geq M$. (Min is greatest lower bound.)

$\text{argmax}_{w:||w||=1} M$ s.t. $\forall_{(x,y)\in S} (2y\text{-}1)<1/M\, w, x> \geq 1$. (Multiply by $1/M$.)

$\text{argmin}_{w:||w||=1} 1/M$ s.t. $\forall_{(x,y)\in S} (2y\text{-}1)<1/M\, w, x> \geq 1$. ($1/M$ decreases with M.)

$\text{argmin}_{w} ||w||$ s.t. $\forall_{(x,y)\in S} (2y\text{-}1)<w, x> \geq 1$. (Move $1/M$ into norm of w.)

$\text{argmin}_{w} ||w||^2$ s.t. $\forall_{(x,y)\in S} (2y\text{-}1)<w, x> \geq 1$. (Squaring norm is monotonic.)

$\text{argmin}_{w} <w,w>$ s.t. $\forall_{(x,y)\in S} (2y\text{-}1)<w, x> \geq 1$. (Undo square root in norm.)

Some Technical Details

The last formulation is a type of quadratic program (QP):

$$\mathrm{argmin}_w \ ||w||^2 \text{ s.t. } \forall_{(x,y)\in S} \ (2y\text{-}1)<w, x> \geq 1,$$

which is a quadratic objective function and linear constraints.

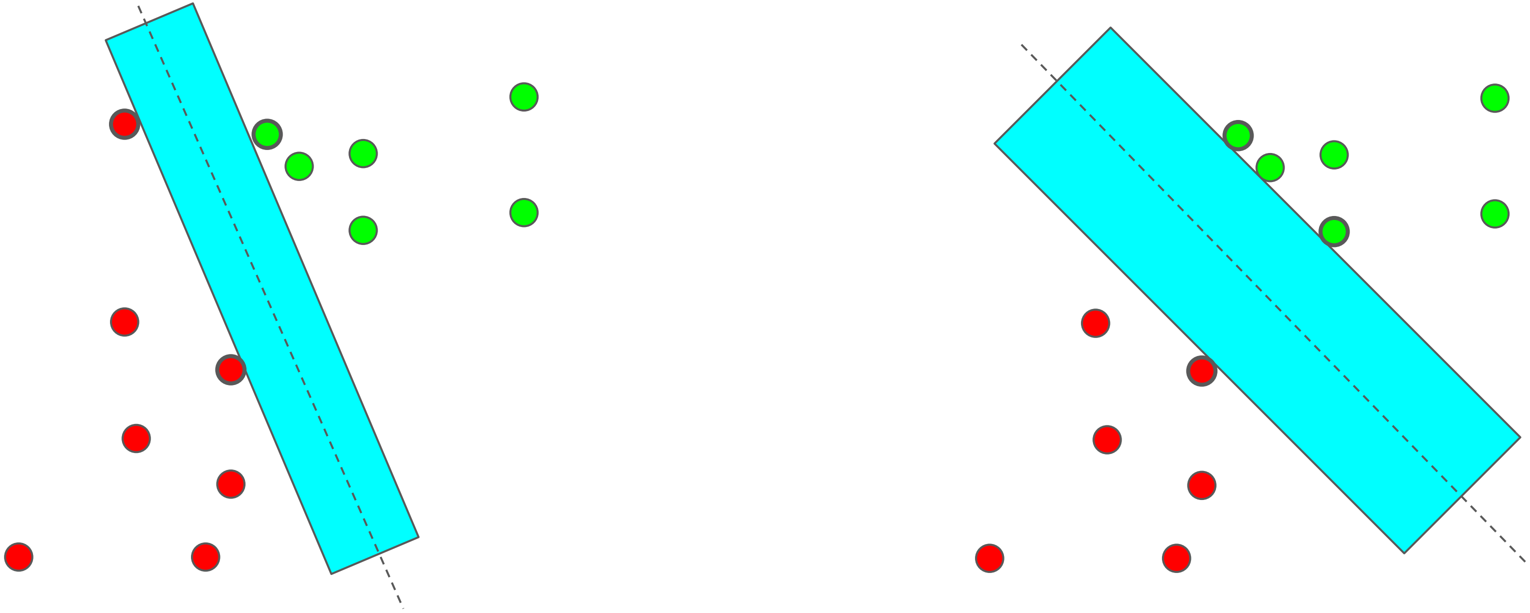
Such problems are computationally intractable, in general. But, solvable in polynomial time with interior point methods if objective is $w^{\mathrm{T}} Q\, w$, where Q is positive definite (like in our case).

PAC bound decreases with increasing (assumptions about the) margin!

“Support vectors” are the points where the constraints are tight.

Support Vectors Define the Separator

3, in 2 dimensions.



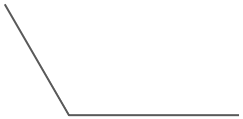
Soft Support Vector Machine

Relaxation, can be used even if data is not linearly separable. Slack variables ξ_i allow constraints to be violated (at a cost).

Given tunable parameter $\lambda>0$.

$\text{argmin}_{(w,\xi)}\ \lambda\ ||w||^2 + 1/m \sum_i \xi_i$ s.t. $\forall_{(xi,yi)\in S}\ (2y_i\text{-}1)<w, x_i> \geq 1\text{-}\xi_i$ and $\xi_i \geq 0$.

Can view the slack variables as a form of "hinge loss".



Duality and Dot Products

The quadratic program for SVMs can be rewritten in its dual form:

$$\max_{\alpha \in \Re n : \alpha \geq 0} \sum_i \alpha_i - \tfrac{1}{2} \sum_i \sum_j (2y_i-1)(2y_j-1)\alpha_i \alpha_j < x_i, x_j >$$

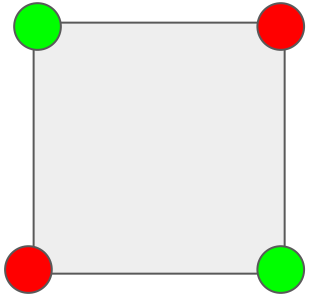
Note: $w = \sum_i (2y_i-1)\ \alpha_i x_i$. The data points for which $\alpha_i > 0$ are the support vectors!

Data dimensionality only relevant in the dot product between training data points.

Also, the sample complexity bounds depend on the margin, not the dimensionality.

Super clever trick lets us increase the dimensionality of the inputs nearly for free!

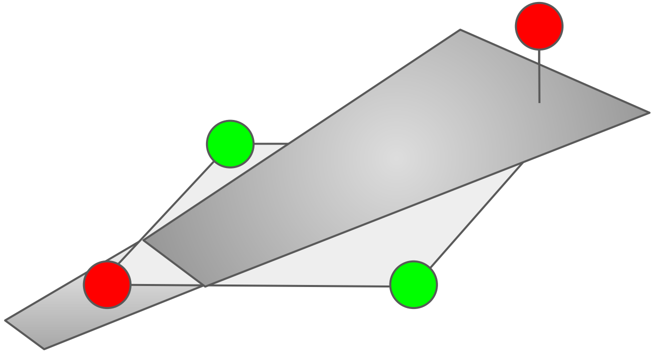
Why Increase Input Dimensionality?



(0,0) → 0, (0,1) → 1, (1,0) → 1, (1,1) → 0

No linear separator.

What happens if we include $x_1\,x_2$ as a separate feature?



(0,0,0) → 0, (0,1,0) → 1, (1,0,0) → 1, (1,1,1) → 0

Now, it’s linearly separable!

$w = (1,1,\text{-}2)$, $b = \text{-}½$

Costs of Adding New Features

Bigger hypothesis space, need more data.

- Depends on margin!

Bigger hypothesis space, more complex search.

- Kernel trick!

Kernel Trick

Dual SVM: $\max_{\alpha \in \Re n : \alpha \geq 0} \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j (2y_i-1)(2y_j-1)\alpha_i \alpha_j < x_i, x_j >$

Dimensionality of input only appears in dot product of points: $< x_i, x_j >$.

What if we simply redefine the dot product so that it includes the derived features?

Dot Products of Transformed Vectors

$\phi(q) = [q_1{}^2, q_2{}^2, \sqrt{2}\, q_1\, q_2]$

$<\phi(x), \phi(y)>$

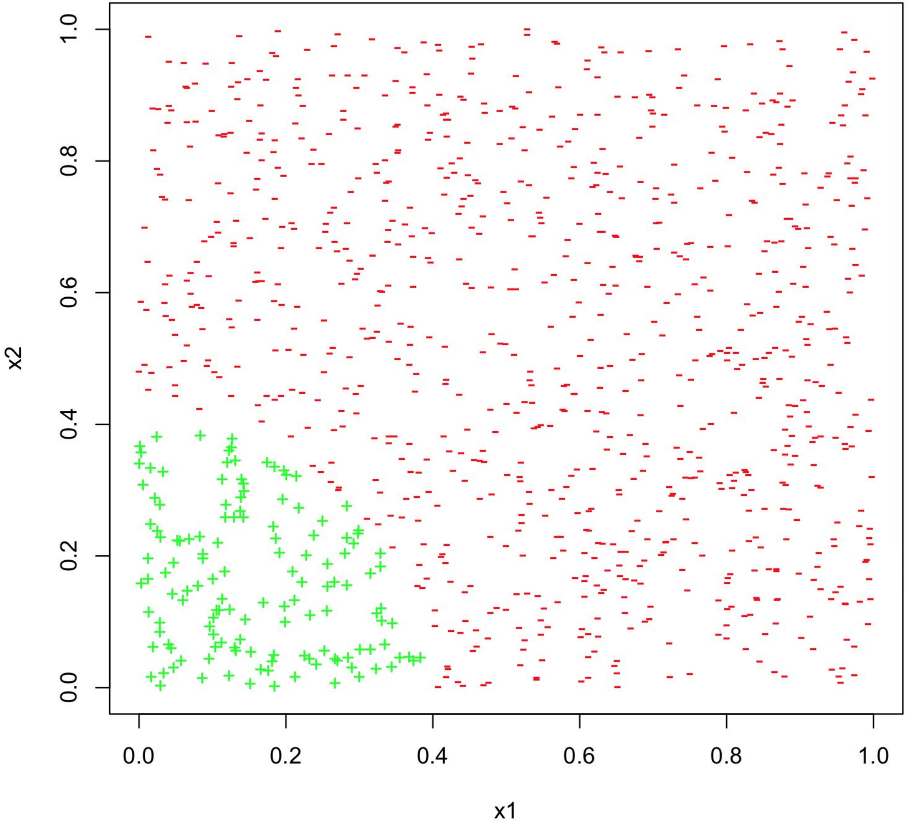
$= <[x_1{}^2, x_2{}^2, \sqrt{2}\, x_1\, x_2], [y_1{}^2, y_2{}^2, \sqrt{2}\, y_1\, y_2]>$

$= x_1{}^2\, y_1{}^2 + 2\, x_1\, x_2\, y_1\, y_2 + x_2{}^2\, y_2{}^2$

$= (x_1\, y_1 + x_2\, y_2)^2$

$= <x, y>^2$

Not linear separable in original space. Linearly separable in the transformed space!



String Kernel

Represent a string as a vector by creating a dimension for every possible substring and putting in the count of how many times that sequence appears in the string. So, the vector for “banana” has:

$v_{\mathrm{b}} = 1, v_{\mathrm{a}} = 2, v_{\mathrm{n}} = 2, v_{\mathrm{ba}} = 1, v_{\mathrm{an}} = 2, v_{\mathrm{na}} = 2, v_{\mathrm{ban}} = 1, v_{\mathrm{ana}} = 2, v_{\mathrm{nan}} = 1, v_{\mathrm{bana}} = 1, v_{\mathrm{anan}} = 1, v_{\mathrm{nana}} = 1, v_{\mathrm{banan}} = 1, v_{\mathrm{anana}} = 1, v_{\mathrm{banana}} = 1$, otherwise, $v_i = 0$

Infinite dimensional! But, can compute the kernel value efficiently: How many substrings are in common between two strings?

Kernel Gallery

http://crsouza.com/2010/03/17/kernel-functions-for-machine-learning-applications/

Polynomial kernel: $K(x,y) = (c_1 <x,y> + c_2)^{c3}$

Gaussian kernel: $K(x,y) = \exp(-||x-y||^2 / c^2)$

Sigmoid kernel: $K(x,y) = \tanh(c_1 <x,y> + c_2)$

Many more.