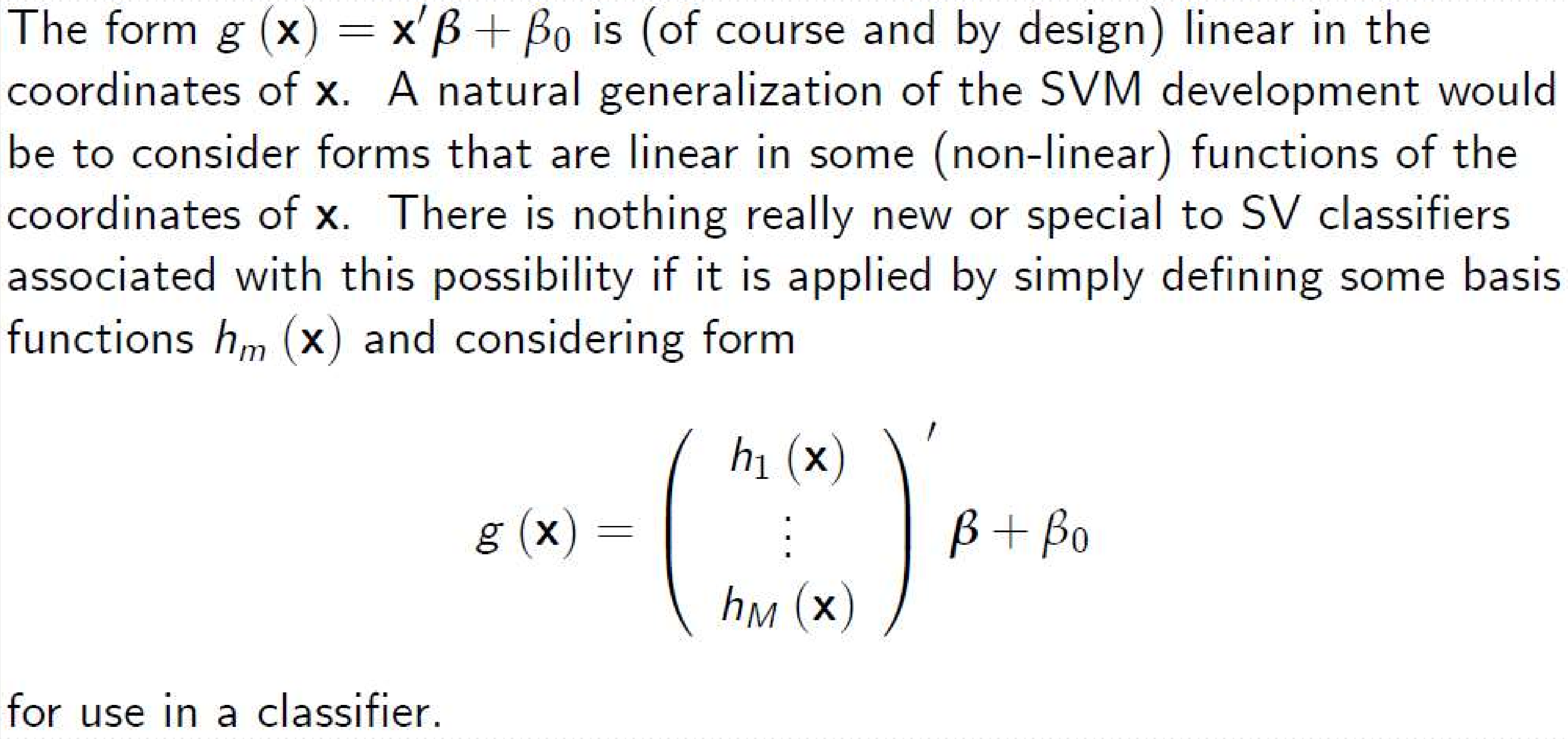
SVMs Part 3A: SVM Heuristics

Stephen Vardeman

Analytics Iowa LLC

ISU Statistics and IMSE

SV classifiers and basis functions

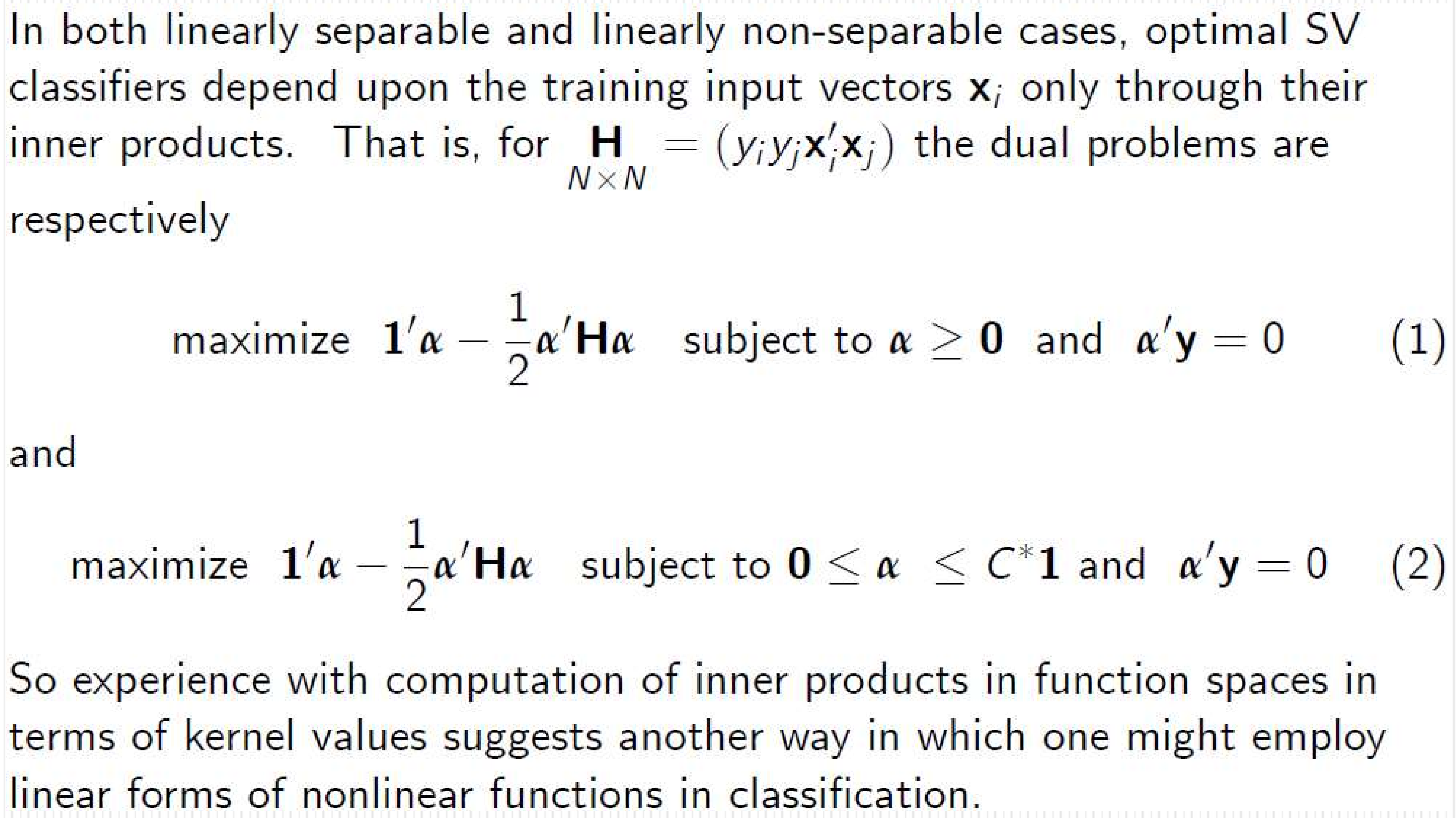


The form $g(\mathbf{x}) = \mathbf{x}'\boldsymbol{\beta} + \beta_0$ is (of course and by design) linear in the coordinates of $\mathbf{x}$. A natural generalization of the SVM development would be to consider forms that are linear in some (non-linear) functions of the coordinates of $\mathbf{x}$. There is nothing really new or special to SV classifiers associated with this possibility if it is applied by simply defining some basis functions $h_m(\mathbf{x})$ and considering form

$$g(\mathbf{x}) = \begin{pmatrix} h_1(\mathbf{x}) \\ \vdots \\ h_M(\mathbf{x}) \end{pmatrix}' \boldsymbol{\beta} + \beta_0$$

for use in a classifier.

SV classifiers and kernels



In both linearly separable and linearly non-separable cases, optimal SV classifiers depend upon the training input vectors $\mathbf{x}_i$ only through their inner products. That is, for $\underset{N\times N}{\mathbf{H}} = (y_i y_j \mathbf{x}_i' \mathbf{x}_j)$ the dual problems are respectively

$$\text{maximize} \quad \mathbf{1}'\boldsymbol{\alpha} - \frac{1}{2}\boldsymbol{\alpha}'\mathbf{H}\boldsymbol{\alpha} \quad \text{subject to } \boldsymbol{\alpha} \geq \mathbf{0} \text{ and } \boldsymbol{\alpha}'\mathbf{y} = 0 \qquad (1)$$

and

$$\text{maximize} \quad \mathbf{1}'\boldsymbol{\alpha} - \frac{1}{2}\boldsymbol{\alpha}'\mathbf{H}\boldsymbol{\alpha} \quad \text{subject to } \mathbf{0} \leq \boldsymbol{\alpha} \ \leq C^*\mathbf{1} \text{ and } \boldsymbol{\alpha}'\mathbf{y} = 0 \qquad (2)$$

So experience with computation of inner products in function spaces in terms of kernel values suggests another way in which one might employ linear forms of nonlinear functions in classification.

Replacing Euclidean inner products

Let $\mathcal{K}$ be a non-negative definite kernel and consider the possibility of using N functions $\mathcal{K}(\mathbf{x}, \mathbf{x}_1), \ldots, \mathcal{K}(\mathbf{x}, \mathbf{x}_N)$ to build new (N-dimensional data-dependent) feature vectors

$$\mathbf{k}(\mathbf{x}) = \begin{pmatrix} \mathcal{K}(\mathbf{x}, \mathbf{x}_1) \\ \vdots \\ \mathcal{K}(\mathbf{x}, \mathbf{x}_N) \end{pmatrix}$$

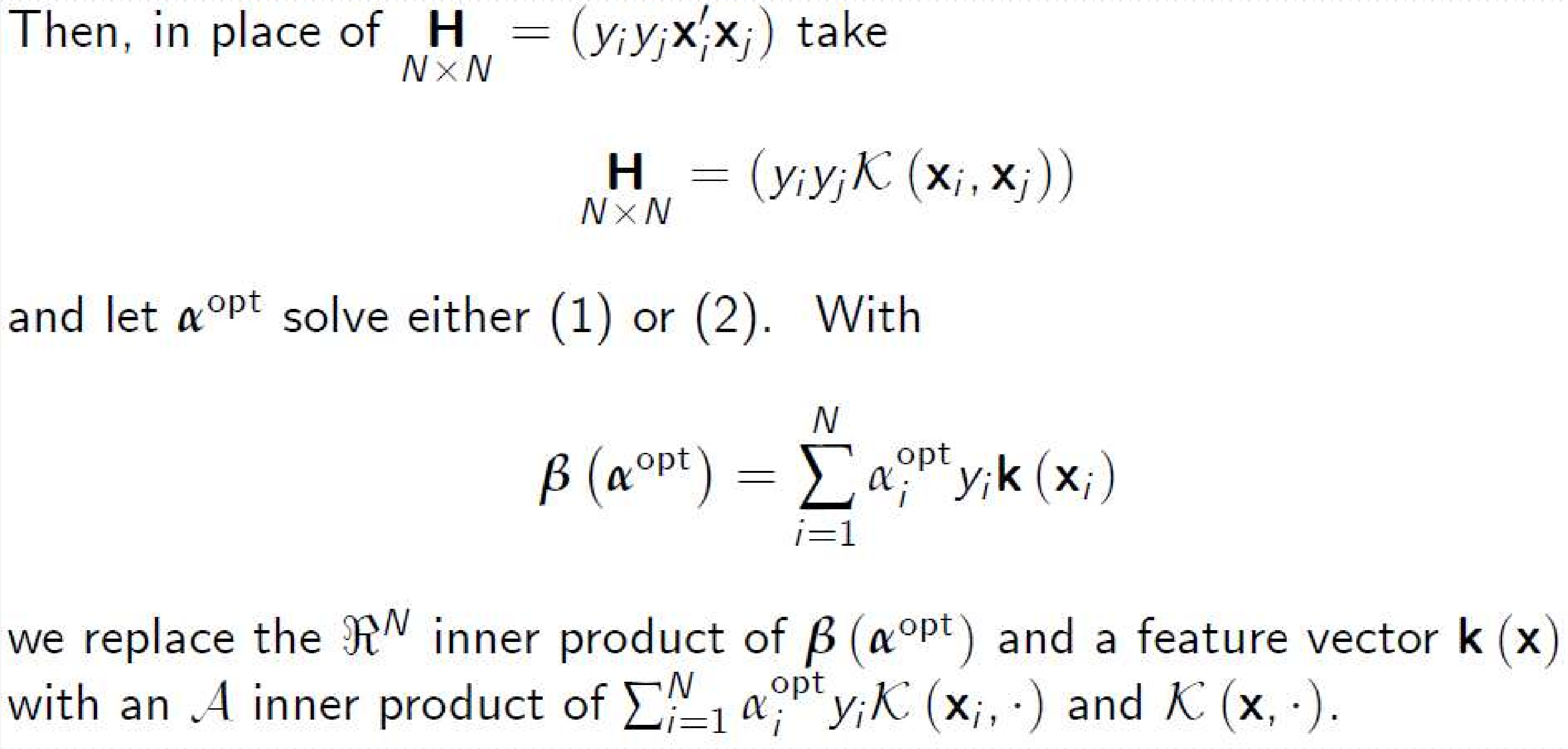
for any input vector $\mathbf{x}$ (including the $\mathbf{x}_i$ in the training set) and rather than defining inner products for new feature vectors (for input vectors $\mathbf{x}$ and $\mathbf{z}$) in terms of $\Re^N$ inner products

$$\mathbf{k}(\mathbf{x})' \mathbf{k}(\mathbf{z}) = \sum_{k=1}^{N} \mathcal{K}(\mathbf{x}, \mathbf{x}_k) \mathcal{K}(\mathbf{z}, \mathbf{x}_k)$$

we consider the abstract space inner products of corresponding functions

$$\langle \mathcal{K}(\mathbf{x}, \cdot), \mathcal{K}(\mathbf{z}, \cdot) \rangle_{\mathcal{A}} = \mathcal{K}(\mathbf{x}, \mathbf{z})$$

SVM heuristics



Then, in place of $\underset{N\times N}{\mathbf{H}} = \left(y_i y_j \mathbf{x}_i'\mathbf{x}_j\right)$ take

$$\underset{N\times N}{\mathbf{H}} = \left(y_i y_j \mathcal{K}\left(\mathbf{x}_i, \mathbf{x}_j\right)\right)$$

and let $\boldsymbol{\alpha}^{\mathrm{opt}}$ solve either (1) or (2). With

$$\boldsymbol{\beta}\left(\boldsymbol{\alpha}^{\mathrm{opt}}\right) = \sum_{i=1}^{N} \alpha_i^{\mathrm{opt}} y_i \mathbf{k}\left(\mathbf{x}_i\right)$$

we replace the $\Re^N$ inner product of $\boldsymbol{\beta}\left(\boldsymbol{\alpha}^{\mathrm{opt}}\right)$ and a feature vector $\mathbf{k}\left(\mathbf{x}\right)$ with an $\mathcal{A}$ inner product of $\sum_{i=1}^{N} \alpha_i^{\mathrm{opt}} y_i \mathcal{K}\left(\mathbf{x}_i, \cdot\right)$ and $\mathcal{K}\left(\mathbf{x}, \cdot\right)$.

SVM heuristics cont.

That is, $\mathbf{k}(\mathbf{x})' \boldsymbol{\beta}(\boldsymbol{\alpha}^{\text{opt}})$ is replaced with

$$\left\langle \sum_{i=1}^{N} \alpha_i^{\text{opt}} y_i \mathcal{K}(\mathbf{x}_i, \cdot), \mathcal{K}(\mathbf{x}, \cdot) \right\rangle_{\mathcal{A}} = \sum_{i=1}^{N} \alpha_i^{\text{opt}} y_i \mathcal{K}(\mathbf{x}, \mathbf{x}_i)$$

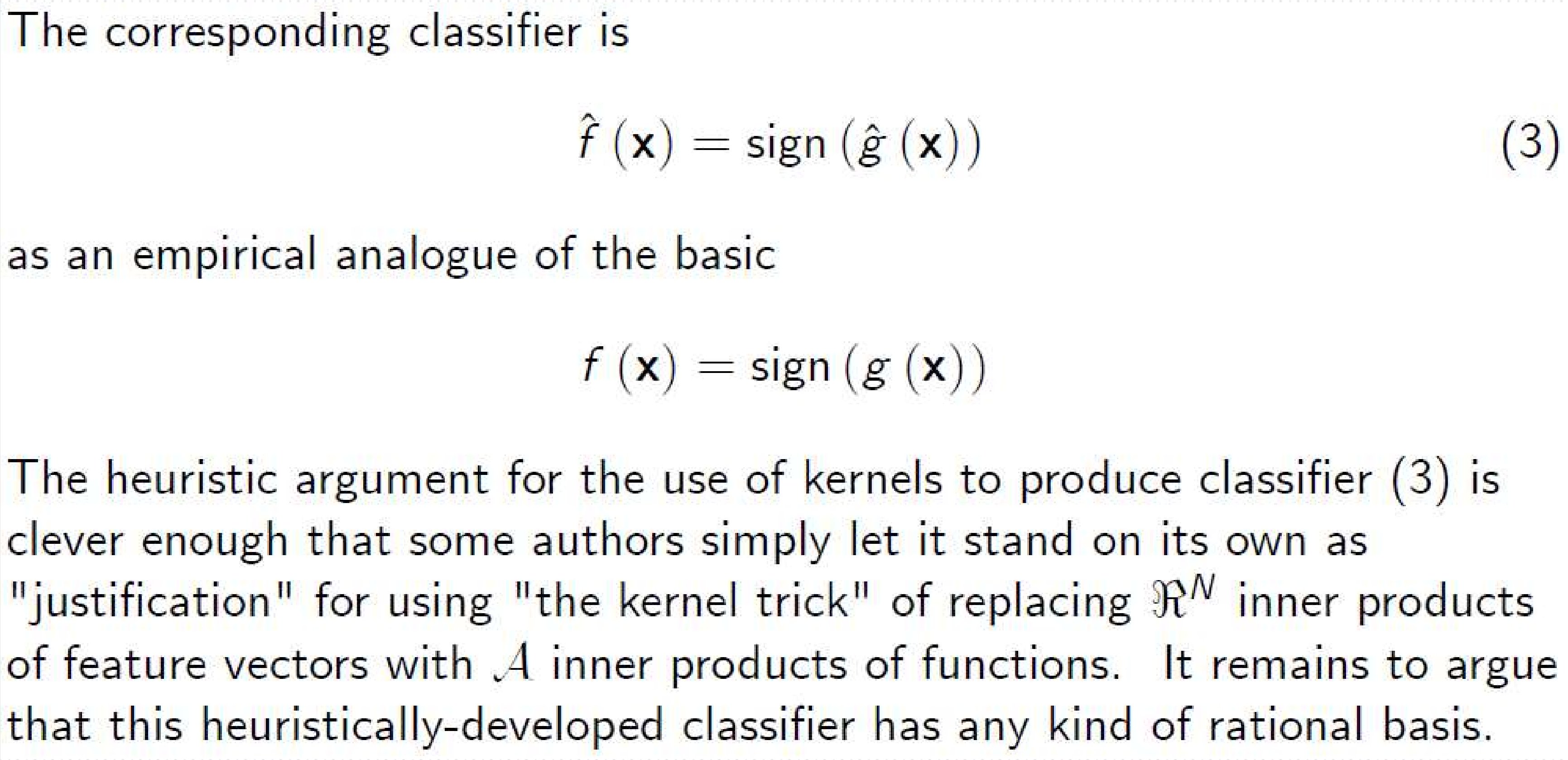
Then for any i for which $\alpha_i^{\text{opt}} > 0$ (an index corresponding to a **support** feature **vector** in this context) set

$$\beta_0(\boldsymbol{\alpha}^{\text{opt}}) = y_i - \sum_{j=1}^{N} \alpha_j^{\text{opt}} y_j \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$$

and have an (empirical) analogue of $g(\mathbf{x}) = \mathbf{x}'\boldsymbol{\beta} + \beta_0$ (for the kernel case)

$$\hat{g}(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i^{\text{opt}} y_i \mathcal{K}(\mathbf{x}, \mathbf{x}_i) + \beta_0(\boldsymbol{\alpha}^{\text{opt}})$$

SVM heuristics cont.



The corresponding classifier is

$$\hat{f}(\mathbf{x}) = \operatorname{sign}(\hat{g}(\mathbf{x})) \tag{3}$$

as an empirical analogue of the basic

$$f(\mathbf{x}) = \operatorname{sign}(g(\mathbf{x}))$$

The heuristic argument for the use of kernels to produce classifier (3) is clever enough that some authors simply let it stand on its own as "justification" for using "the kernel trick" of replacing $\Re^N$ inner products of feature vectors with $\mathcal{A}$ inner products of functions. It remains to argue that this heuristically-developed classifier has any kind of rational basis.