Table S1: Contributions of the three top principal components of data extracted from chemo-dyes points via NIR-CDS system based on principal components analysis (PCA).

Dye points	**Chemo-dyes**	**% Principal Components (PC)**			**Sum of Top three PCs (%)**
		PC1	PC2	PC3	
1	5, 10, 15, 20 – Tetraphenyl - 21H, 23H-porphine manganese (III) chloride	79.00	1.50	1.40	81.90
2	5, 10, 15, 20 - Tetra phenyl-21H, 23H - porphine	77.00	1.59	1.20	79.79
3	2, 3, 7, 8, 12, 13, 17, 18 - Octaethyl-21H, 23H - porphine manganese (III) chloride	82.00	1.30	1.10	84.4
4	5, 10, 15, 20 – Tetrakis (4 – methoxyphenyl) - 21H, 23H - porphine iron (III) chloride	76.00	1.30	1.15	78.45
5	5, 10, 15, 20 – Tetraphenyl - 21H, 23H - porphine	80.00	2.80	1.30	84.10
6	5, 10, 15, 20 – Tetraphenyl - 21H, 23H - porphine copper (II)	75.60	5.20	1.57	82.37
7	5, 10, 15, 20 – Tetraphenyl - 21H, 23H - porphine zinc	76.40	4.40	3.20	84.00
8	5, 10, 15, 20 - Tetra phenyl - 21H, 23H - porphine iron (III) chloride	80.10	2.10	1.85	84.05
9	5, 10, 15, 20 - Tetrakis (4 – methoxyphenyl) - 21H, 23H - porphine cobalt(II)	98.36	1.02	0.42	99.80
10	Methyl red	78.00	1.15	1.10	80.25
11	Bromocresol green	70.00	5.50	2.10	77.60
12	Bromothymol blue	81.20	1.05	1.01	83.26

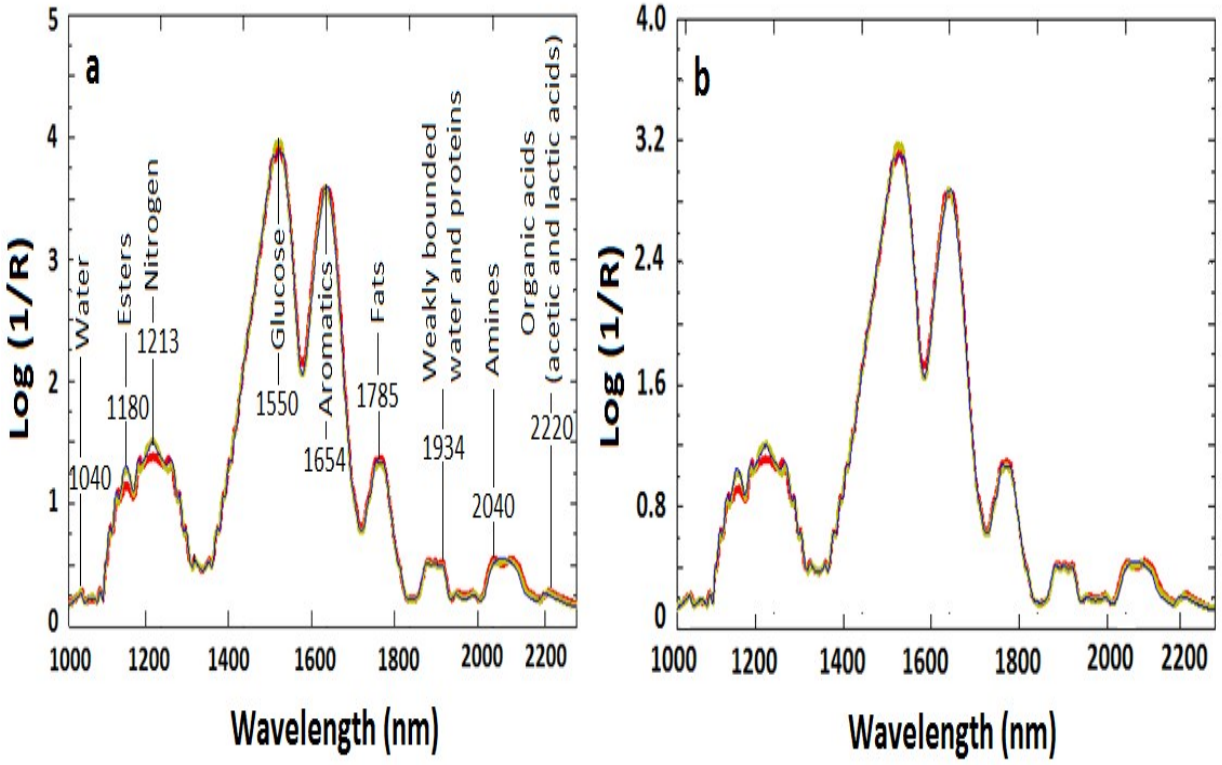

Fig. S1: SNV preprocessed spectra (a) and MSC preprocessed spectra (b) of cocoa beans samples acquired with NIRS- CDS based on TPP-Co

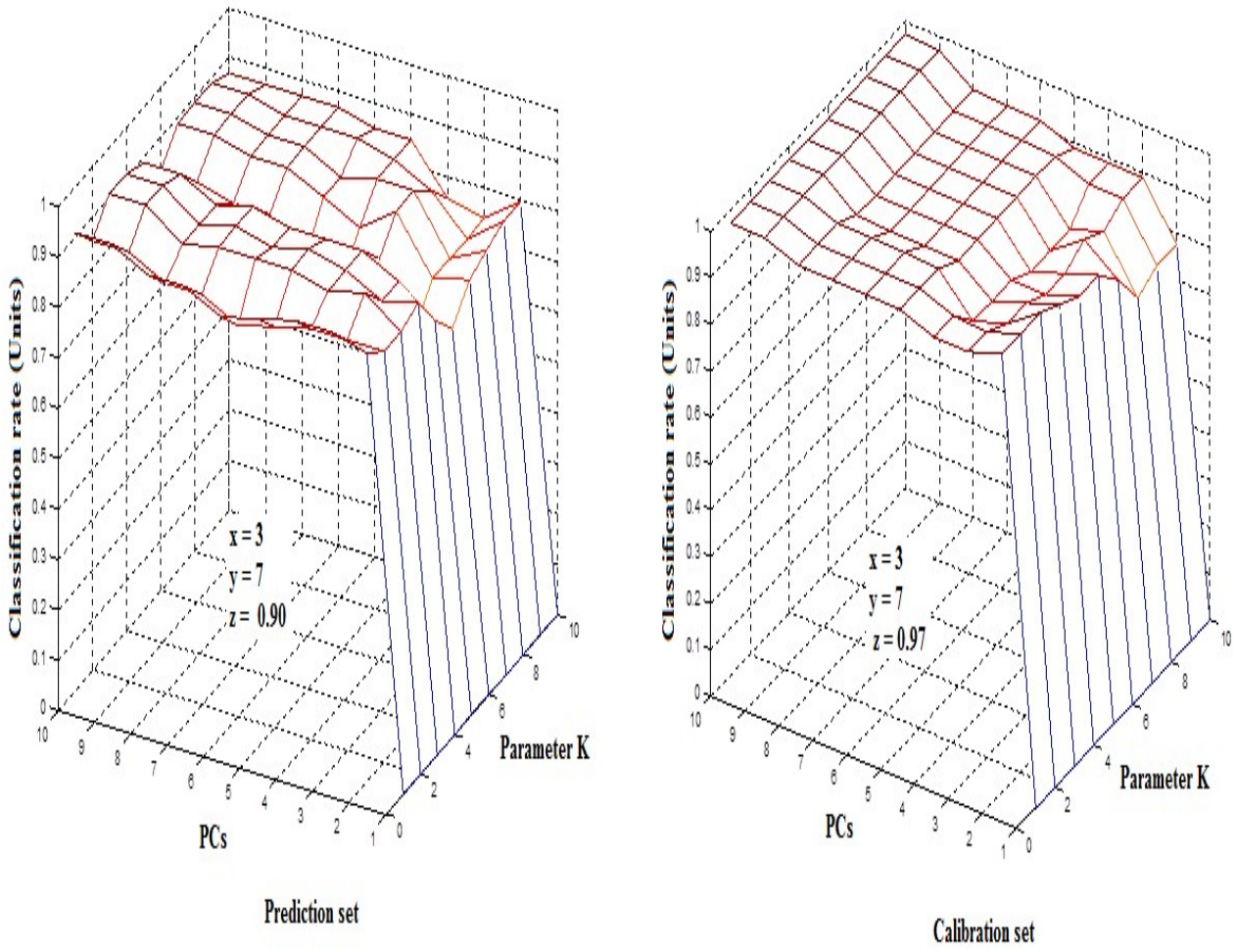

Fig. S2: The optimized kernel parameter (K) values of K-NN algorithm with their respective computed classification rates and principal components (PCs) numbers in both the prediction and calibration sets generated for CS e-nose.

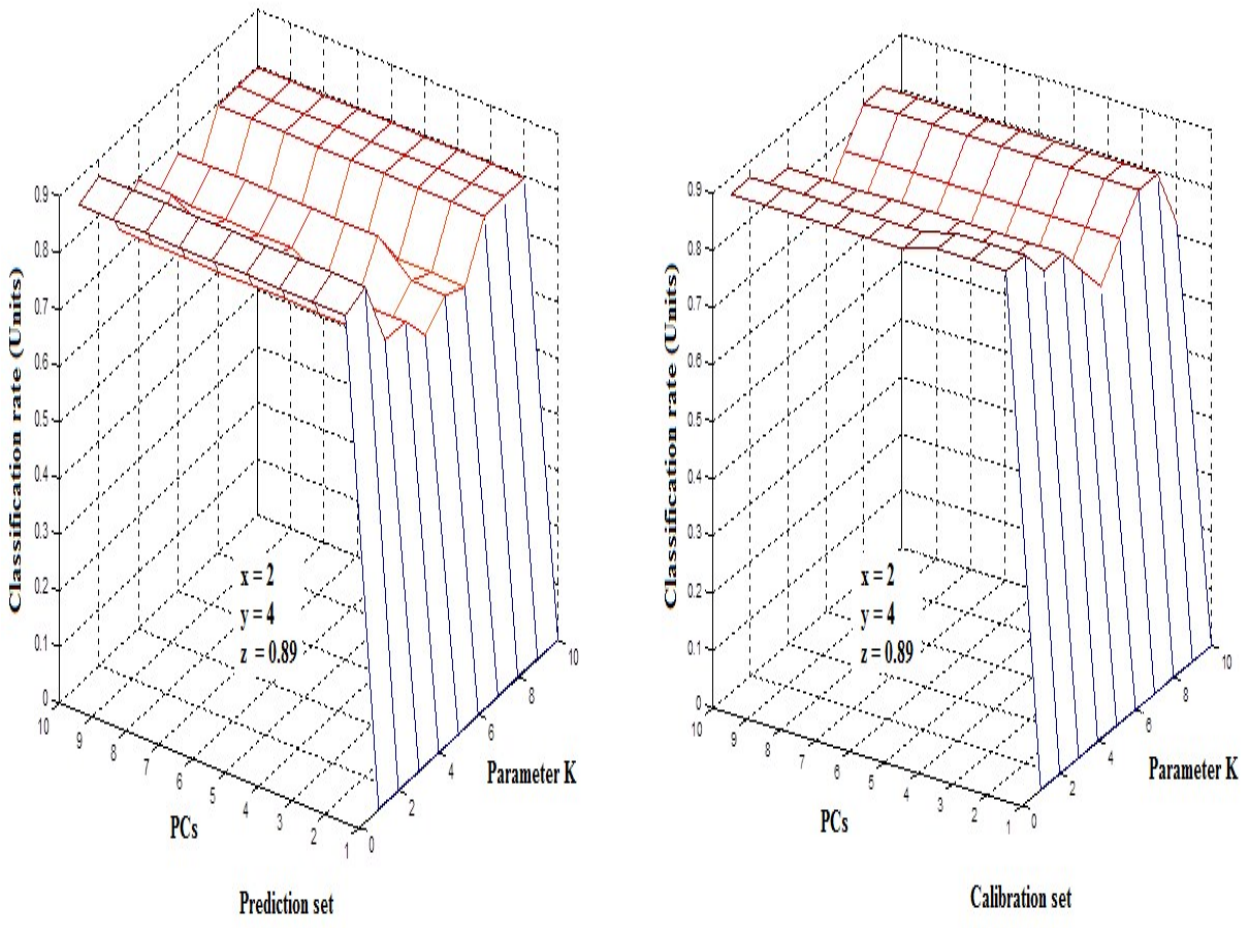

Fig. S3: The optimized kernel parameter (K) values of K-NN algorithm with their respective computed classification rates and principal components (PCs) numbers in both the prediction and calibration sets generated for NIR-CDS with point 9 (TPP-Co)

Formulae / mathematical expressions underlying Algorithms applied

Linear Discrimination Analysis (LDA)

This is based on Bayes rule with the assumption that characteristics or patterns of a class say k with a mean $\vec{x}_i$ that are normally distributed and have a covariance matrix H that relates to all sample classes under consideration.

Hypothetically, applying of the Bayes rule assigns a test pattern denoted by (a) to a class k with the largest posterior probability $P(k \mid a)$ as expressed mathematically below:

$$- 2\log\ [P(k \mid a)] = (a - \vec{x}_i)^T H^{-1}(a - \vec{x}_i) - 2log\pi_i + log|H|$$

Hence the determinant of H, which is $|H|$ to the class of k maximizes the linear function given as:

$$L_i(a) = 2\vec{x}_i^{\,T} H^{-1} a - \vec{x}_i^{\,T} H^{-1} \vec{x}_i + 2log\pi_i$$

Approximating the matrix H by the intra-class covariance matrix (I) becomes:

$$I = \left(\frac{A - CF)^T (A - CF)}{n - F}\right)$$

$where\ A\ is\ the\ N\ x\ n - order\ calibration\ set\ matrix;$

$F\ is\ the\ F\ x\ n\ matrix\ with\ the\ class\ means;$

$C\ is\ the\ N\ x\ \ F - order\ matrix\ of\ class\ indicators\ (C_{ij} = 1, when\ the\ calibration$

$set\ pattern\ of\ sample\ a_i\ belongs\ to\ class\ j\ or\ otherwise)$

These mathematical expressions stated by González-Rufino *et al.*[17] were used in the development of the k-NN algorithms which were written in matlab codes and applied.

K-nearest neighbour (k-NN)

K-NN algorithm was written in Matlab codes based on the following mathematical expressions used by O'farrell *et al.* [16]

K-NN algorithm finds or locates the k-nearest patterns to sample (a) among the calibration data set within the classification set (G), such that $G = \{(x_i, y_i), i = 1...n\}$

Where $x_i\ denote\ a\ training\ pattern\ in\ the\ calibration\ \ data\ set, y_i\ its\ corresponding$

$class,$ $and\ \ i$
$the\ amount\ of\ training\ patterns - number\ of\ samples\ x\ number\ of\ classes\ (eg.$
$90\ cocoa\ beans\ 30\ x\ 3)$.

The classification of the samples into the different groups is algorithmically decided based on the highest vote of the nearest neighbours, with the selected class for the sample assigned the code of 1 or otherwise 0. Thus for completely dissimilar or disjoint classes, the kernel parameter K = 1. The higher the K values in a mixed sample, the lesser the cases of misclassifications.

Support Vector Machine (SVM)

This algorithm is underpinned by structural risk minimization, which minimizes the chances of sample misclassification of an unseen data among a fixed distribution with an unknown probability. The study adopted and applied the SVM algorithm based on mathematical expressions used by Chen *et al*. [15]

Taking for instance, a training data set with k number of samples denoted as $\{a_i, b_i\}, i = 1, 2, \ldots., p; in\ which\ case\ a \epsilon R^n, and\ is\ n - dimensional\ vector; and\ with\ a\ class\ given\ as\ b \epsilon \{-1, +1\}$. The various patterns can be considered as linearly separable given that vector $\omega\ and\ scalar\ \gamma\ are\ defined\ to\ satisfy\ the\ conditions\ of\ the\ inequalities\ below$:

$$\omega.a_i + \geq \gamma + 1, for\ all\ b = +1 \quad (1)$$

$$\omega.a_i + \gamma \geq -1, \ for\ all\ b = -1 \quad (2)$$

These inequality conditions create a hyperplane that separates data points such that those with the same label are located on the same side. For this to occur, $\omega\ and\ \gamma\ must\ be\ such\ that\ the$ condition below is:

$$b_i(\omega.a_i + \gamma) > 0 \quad (3)$$

Satisfying condition (3) enables the creation of a hyperplane that divides the samples into different linearly separable classes. Rescaling of $\omega\ and\ \gamma$ to achieve $\min_{1 \leq i \leq p} b_i(\omega.a + \gamma) \geq 1$, leads to the distance from the data point closest to the hyperplane to be $1/||\omega||$. This therefore modifies condition (3) as:

$$b_i(\omega.a_i + \gamma) \geq 1 \quad (4)$$

The resulting hyperplane from the adjustment under expression (4), for which the distance to the closest point is maximal becomes an optimal separating hyperplane (OSH). The OSH can be obtained via minimizing $||\omega||^2$ based on constraint (4) as the distance to the closest point equals $1/||\omega||$. This minimization procedure is achieved using Lagrange multipliers and quadratic programming optimization methods. However, this can lead to maximizing when $\alpha_i \geq 0, i = 1,2,\ldots,p$ are positive Lagrange multipliers linked with constraint (4) such that:

$$L(\alpha) = \sum_i \alpha_i - 1/2 \sum_{i.j} \alpha_i \alpha_j b_i b_j (a_i a_j) \quad (5)$$

Given that $\alpha^m = (\alpha_1^m, \alpha_2^m, \ldots \alpha_p^m)$ leads to obtaining an optimal maximization solution for the problem under (5). The resultant optimal separating hyperplane can be expressed as:

$$\omega^m = \sum_i a_i b_i \alpha_i^m \quad (6)$$

Thus the support vectors then becomes the point where $\alpha_1^m > 0, when \ \ the \ equality \ in \ under \ constraint \ (4) \ holds$.

A slack variable $\eta_i \geq 0, i = 1,2 \ldots, p$ can be introduced when the data are not linearly separable leading to:

$$b_i(\omega.a_i + \gamma) \text{ - } 1 + \eta_i \geq 0 \quad (7)$$

Thus a generalized OSH is obtained from the difference of expressions (8) – (9).

$$\min_{\omega, \gamma, \eta_i, \ldots \eta_p} [\frac{1}{2}||\omega||^2 + K \sum_{i=1}^{p} \xi_i] \quad (8)$$

$$\eta_i \geq 0, i = 1,2 \ldots, p \quad (9)$$

The first and second terms in expression 8, is the same as the linearly separable case, in which case they respectively control the algorithm learning capacity and the number of misclassified samples; whereas the parameter C is selected by user. Selecting higher value for C suggests assigning higher penalty parameter to error.

However, when it is impossible to define a hyperplane with linear equations on the training data, the data is transformed to a higher dimensional space in order to spread it out with the aim of finding a linear hyperplane that is computed by using the concept of kernel function (K), by mapping (a) it to its feature space $\varphi(a_i)$ using expression 5 modified as:

$$L(\alpha) = \sum_i \alpha_i - 1/2 \sum_{i,j} \alpha_i \alpha_j b_i b_j (\varphi(a_i)\varphi(a_j)) \quad (10)$$

The kernel function facilitates easier computations in the feature space and it is denoted by:

$$K(a_i, b_i) = (\varphi(a_i)\varphi(a_j))$$

Hence the kernel function is used as a classification function. The SVM was applied to the data collected from the samples and transformed them into a higher dimensional feature space, which then allows for their classification based on the maximal hyperplane generated.

Extreme Learning Machine (ELM)

For the ELM algorithm which is a single hidden feed forward network, employs the random selection of input weights with its resulting output analytically computed. The mathematical basis underlying this algorithm as presented by Zheng *et al.* [14] was adopted.

Given the variable a, an input vector; K, an activation function; δ, the weight vector linking the hidden node and the output nodes; $f(a)$, the output vector; w, weight vector connecting the hidden node and the input nodes; and b, the bias associated to the hidden node.

Considering C calibration samples of $(a_i, t_i) \epsilon R^d x R^m$, where t_i denotes the output of $f(a)$ labeled with the codes $(0,1)^m$ as the category vector, the ELM algorithm can be expressed and modeled using its activation function $K(w,b,a)$ and P hidden nodes as:

$$\sum_{i=1}^{P} \delta_i K_i(w_i a_i + b_1) = t_i, j = 1,2,\ldots N. \quad (1)$$

The equation (1) can be simplified as $T = B\delta$ with B and δ expressed with the equations (2) and (3) respectively below:

$$\delta = \begin{bmatrix} \delta_1^T \\ \vdots \\ \delta_C^T \end{bmatrix}_{L\ x\ m}, T = \begin{bmatrix} t_1^T \\ \vdots \\ t_C^T \end{bmatrix}_{P\ x\ m} \quad (2)$$

$$B = \begin{bmatrix} K_i(w_i a_i + b_1) & .. & K_i(w_P a_i + b_P) \\ : & .. & : \\ K_i(w_i a_C + b_1) & .. & K_i(w_P a_C + b_P) \end{bmatrix}_{CxP} \qquad (3)$$

The hidden nodes parameters are randomly selected and the weight matrix (δ) estimated with expression (4) :

$$\hat{\delta} = B^T T(B^T B + \lambda I)^{-1} \qquad (4)$$

where $\lambda > 0$ is considered a regularized parameter

Based on above expression, the output weights of ELM can be computed. Thus a class of an unknown sample (cocoa bean sample) $\check{a}$ can be predicted as below:

$$class(\check{a}) = \arg max(\check{B}\hat{\delta}), \qquad where\ \check{B} = K(w_1\check{a} + b_1)...K(w_P\check{a} + b_P)$$