CHAPTER 3

Support Vector Classifiers based on Hyperellipsoids

This chapter is divided into three sections arranged by three main contributions of this thesis. The first section particularly focuses on one-class classification using a novel hyperellipsoid-based method as a data description tool. The proposed method is called "ellipsoidal support vector data description" (eSVDD) since it constructs an optimal soft-margin hyperellipsoid around the data, like in SVDD. The formulation of eSVDD is built upon the idea of MVCE introduced in the previous chapter with some additional features, especially the inclusion of negative examples.

In the second section, another support vector based classifier named the "twin hyper-ellipsoidal support vector machine" (TESVM) is proposed. The formulation is specifically designed for binary classification with the idea inspired by MVCE, SVDD, and THSVM. The formulations presented in the first two sections are in fact the basis of the proposed methods. In the last section, we further propose that, by using an empirical feature mapping technique, the proposed formulations can be extended to work in the feature space, allowing the formation of more complex decision boundary to deal with real-world data.

3.1 Ellipsoidal Support Vector Data Description

A soft-margin MVCE is directly inspired by other popular learning machines like SVM and SVDD. For a given set of m training examples, $\{\mathbf{x}_i\}_{i=1}^m$, the general idea of MVCE with soft margins, based on the MVCE formulation (2.17), can be formulated as

$$\begin{aligned} \min_{\mathbf{E},\mathbf{c},\xi} \quad & 2\log\det(\mathbf{E}^{-1}) + \frac{C}{m}\sum_{i=1}^{m}\xi_i \\ \text{s.t.} \quad & \left\|\mathbf{E}\mathbf{x}_i - \mathbf{c}\right\|^2 \le 1+\xi_i, \quad \xi_i \ge 0, \quad i=1,\ldots,m \end{aligned} \tag{3.1}$$

where the hyperellipsoid $\mathsf{E}_{\mathbf{E},\mathbf{c}}$ is the solution to (3.1) whose center is at $\mathbf{E}^{-1}\mathbf{c}$.

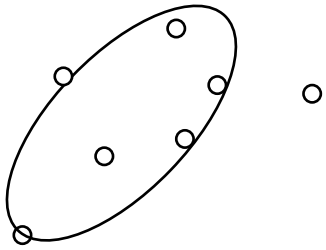

Figure 3.1 Rough sketch of eSVDD's descriptive boundary

Again, ξ_i is a slack variable to allow possible misclassification. $C > 0$ is the hyperparameter

Since the examples in the formulation (3.1) have no label, the formulation is also called "eSVDD with only positive examples" or simply eSVDD and can be illustrated as in Figure 3.1. It is worth noting that the formulation (3.1) already exists in the literature, such as in the works by Dolia et al. [58] and Wei et al. [60].

Among m training examples, it is also possible that few of them are erroneous or undesired examples. It is intuitively plausible to include the knowledge of the outliers to help improve the domain description of the data. Therefore, we are going to reformulate (3.1) to handle two possible labels, i.e. "target" and "outlier". The result is still a one-class classification problem under the assumption that the outlier class has very few number of examples.

Suppose each example has a label either $y_i = 1$ or -1 for target and outlier, respectively. We call (3.1) when reformulated with two possible labels as "eSVDD with negative examples" (neSVDD). The illustration of neSVDD is shown in Figure 3.2 where ○ and ● represent the target and outlier classes. When no outlier is presented, neSVDD is simply an eSVDD problem. It is natural to formulate the problem such that outliers are kept outside while targets are kept inside the hyperellipsoid. Some outliers and targets are minimally allowed to be inside and outside the descriptive boundary, respectively.

Thus, the formulation of neSVDD can be formulated as

$$\begin{aligned} \min_{\mathbf{E},\mathbf{c},\boldsymbol{\xi}} \quad & 2\log\det(\mathbf{E}^{-1}) + \frac{C}{m}\sum_{i=1}^{m}\xi_i \\ \text{s.t.} \quad & y_i\left\|\mathbf{E}\mathbf{x}_i - \mathbf{c}\right\|^2 \le y_i + \xi_i, \quad \xi_i \ge 0, \quad i = 1,\ldots,m. \end{aligned} \tag{3.2}$$

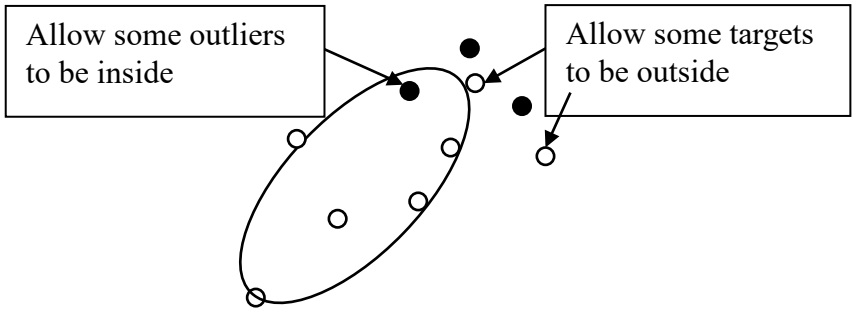

Figure 3.2 Rough sketch of neSVDD's descriptive boundary

Although the formulation of eSVDD as in (3.1) has no novelty in the literature, very few works have considered the inclusion of negative examples into the formulation. To the best of our knowledge, one formulation of MVCE without soft margins exists in the literature [74] where it is given to demonstrate an application of semidefinite programming. In fact, their formulation differs from our formulation (3.2) and has no soft margins. Another found literature on this subject is by Wei et al. [60] where they briefly mention the formulation of MVCE with negative examples. However, their formulation is also slightly different from our neSVDD in (3.2). Furthermore, their experimental results are reported based only on one dataset, i.e. from the iris dataset. In fact, they provide only the classification accuracy between Versicolor and Virginica classes, which is unlikely adequate to substantiate the efficiency of their method.

The derivation of the corresponding dual problem from (3.2) follows closely with the flow presented in the previous chapter for MVCE. That is from (3.2) the Lagrangian is

$$\begin{aligned} L(\mathbf{E},\mathbf{c},r,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta}) &= 2\log\det(\mathbf{E}^{-1}) + \frac{C}{m}\sum_{i=1}^{m}\xi_i \\ &\quad + \sum_{i=1}^{m}\alpha_i\left(y_i\left\|\mathbf{E}\mathbf{x}_i-\mathbf{c}\right\|^2 - y_i - \xi_i\right) - \sum_{i=1}^{m}\beta_i\xi_i \end{aligned} \tag{3.3}$$

where the Lagrange multiplier $\alpha_i, \beta_i \geq 0$ for $i = 1, 2, \ldots, m$. Hence, we have

$$\begin{aligned} L(\mathbf{E},\mathbf{c},r,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta}) &= -2\log\det(\mathbf{E}) \\ &\quad + \sum_{i=1}^{m}\left(\frac{C}{m} - \alpha_i - \beta_i\right)\xi_i + \sum_{i=1}^{m} y_i\alpha_i\left(\left\|\mathbf{E}\mathbf{x}_i-\mathbf{c}\right\|^2 - 1\right). \end{aligned} \tag{3.4}$$

Suppose $\mathbf{y} = [y_1, y_2, ..., y_m]$ and $\mathbf{Y} = \mathrm{diag}(\mathbf{y})$ with $y_i \in \{1, -1\}$ for $i = 1, 2, ..., m$. The first-order derivatives are

$$\frac{\partial L}{\partial \xi_i} = \frac{C}{m} - \alpha_i - \beta_i \tag{3.5}$$

$$\frac{\partial L}{\partial \mathbf{c}} = 2\left(\mathbf{c}\sum_{i=1}^{m} \alpha_i y_i - \mathbf{E}\sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i \right) = 2\left(\mathbf{c}\mathbf{y}^T\boldsymbol{\alpha} - \mathbf{EXY}\boldsymbol{\alpha} \right) \tag{3.6}$$

$$\frac{\partial L}{\partial \mathbf{E}} = -2\mathbf{E}^{-1} + \sum_{i=1}^{m} \alpha_i y_i \left[(\mathbf{E}\mathbf{x}_i - \mathbf{c})\mathbf{x}_i^T + \mathbf{x}_i(\mathbf{E}\mathbf{x}_i - \mathbf{c})^T \right]. \tag{3.7}$$

Under the first-order necessary condition of optimality, we have $0 \le \alpha_i \le C/m$ from (3.5) and the condition $\alpha_i, \beta_i \ge 0$. The optimal **c** can be obtained from (3.6) as

$$\mathbf{c} = \frac{\mathbf{EXY}\boldsymbol{\alpha}}{\mathbf{y}^T\boldsymbol{\alpha}}. \tag{3.8}$$

Substituting (3.8) into (3.7) and, (3.7) can be rewritten as

$$\mathbf{E}^{-1} = \frac{1}{2}(\mathbf{ES} + \mathbf{SE}) \tag{3.9}$$

where

$$\mathbf{S} = \mathbf{XAYX}^T - \frac{\mathbf{XY}\boldsymbol{\alpha}\boldsymbol{\alpha}^T\mathbf{YX}^T}{\mathbf{y}^T\boldsymbol{\alpha}}. \tag{3.10}$$

It is possible that **E** is not unique since **S** is not positive definite, however, we will assume that $\mathbf{E} = \mathbf{S}^{-1/2}$, similar to the solution in (2.15) and let the logarithm term in the objective of (3.2) act as a natural barrier function to drive the solution **E** to be positive definite. After further rewriting the Lagrangian (3.4) with $\sum_{i=1}^{m} y_i\alpha_i \left\|\mathbf{E}\mathbf{x}_i - \mathbf{c}\right\|^2 = n$, we obtain

$$L(\boldsymbol{\alpha}) = \log\det(\mathbf{XAYX}^T - \frac{\mathbf{XY}\boldsymbol{\alpha}\boldsymbol{\alpha}^T\mathbf{YX}^T}{\mathbf{y}^T\boldsymbol{\alpha}}). \tag{3.11}$$

In addition, by letting $\tilde{\mathbf{X}} = \begin{bmatrix} \mathbf{X} \\ \mathbf{e}^T \end{bmatrix}$ as in (2.22), we have the dual problem of neSVDD as

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \log\det \tilde{\mathbf{X}}\mathbf{AY}\tilde{\mathbf{X}}^T - \mathbf{y}^T\boldsymbol{\alpha} + n \\ \text{s.t.} \quad & \mathbf{0} \le \boldsymbol{\alpha} \le \frac{C}{m}. \end{aligned} \tag{3.12}$$

Furthermore, for more succinct dual formulation, we set $\mathbf{y}^T\boldsymbol{\alpha} = n$ so that the dual and primal problems have the same objective functions. Thus, we have

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \log\det \tilde{\mathbf{X}}\mathbf{A}\mathbf{Y}\tilde{\mathbf{X}}^T \\ \text{s.t.} \quad & \mathbf{y}^T\boldsymbol{\alpha} = n \\ & \mathbf{0} \le \boldsymbol{\alpha} \le \frac{C}{m}. \end{aligned} \tag{3.13}$$

The optimal $\mathsf{E}_{\mathbf{E},\mathbf{c}}$ can be computed from

$$\mathbf{E} = \left(\mathbf{X}\mathbf{Y}\left[\mathbf{A}\mathbf{Y} - \frac{\boldsymbol{\alpha}\boldsymbol{\alpha}^T}{\mathbf{y}^T\boldsymbol{\alpha}} \right]\mathbf{Y}\mathbf{X}^T \right)^{-1/2} \quad \text{and} \quad \mathbf{c} = \frac{\mathbf{E}\mathbf{X}\mathbf{Y}\boldsymbol{\alpha}}{\mathbf{y}^T\boldsymbol{\alpha}}.$$

The slack variables in the primal problem provides the box constraints on Lagrange multipliers in the dual problem (3.12). **c** is a linear combination of all training examples and each example is weighted by a Lagrange multiplier. The shape of hyperellipsoid **E** is a linear combination of outer products of training examples weighted by the same Lagrange multipliers. For the examples which have $\alpha_i > 0$ are the "support vectors" since they affect the shape and center of the hyperellipsoid.

For the classification rule of neSVDD for the solution $\mathsf{E}_{\mathbf{E},\mathbf{c}}$, although the ellipsoid is defined as $\|\mathbf{E}\mathbf{x} - \mathbf{c}\| = 1$, it is **incorrect** to define the classification rule, for a given testing example $\mathbf{x} \in \mathbb{R}^n$, as

$$f(\mathbf{x}) = \begin{cases} 1, & \text{for } \|\mathbf{E}\mathbf{x} - \mathbf{c}\| \le 1 \\ -1, & \text{for } \|\mathbf{E}\mathbf{x}_i - \mathbf{c}\| > 1 \end{cases} \tag{3.14}$$

due to the effect of setting $\mathbf{y}^T\boldsymbol{\alpha} = n$ in the dual formulation (3.13). Since the corresponding primal problem of (3.13) is

$$\begin{aligned} \min_{\mathbf{E},\mathbf{c},r,\xi} \quad & nr^2 + 2\log\det(\mathbf{E}^{-1}) + \frac{C}{m}\sum_{i=1}^{m}\xi_i \\ \text{s.t.} \quad & y_i\|\mathbf{E}\mathbf{x}_i - \mathbf{c}\|^2 \le y_i r^2 + \xi_i, \quad \xi_i \ge 0, \quad i = 1,\ldots,m \end{aligned}$$

as shown in Lemma A.2 of Appendix A. The correct decision rule of neSVDD is

$$f(\mathbf{x}) = \begin{cases} 1, & \text{for } \|\mathbf{E}\mathbf{x} - \mathbf{c}\| \le r \\ -1, & \text{for } \|\mathbf{E}\mathbf{x}_i - \mathbf{c}\| > r \end{cases} \tag{3.15}$$

where r can be obtained from the complementary slackness conditions, according to (A.2). Precisely, those conditions are

$$0 = \beta_i \xi_i \tag{3.16}$$

$$0 = \alpha_i \left[y_i \|\mathbf{E}\mathbf{x}_i - \mathbf{c}\|^2 - y_i r^2 - \xi_i \right] \tag{3.17}$$

for i = 1, 2, ...m. By selecting an example with $0 < \alpha_i < \frac{C}{m}$, i.e. $\beta_i \neq 0$, we obtain $\xi_i = 0$ from (3.16). Therefore, r can be computed from (3.17).

3.2 Twin Hyper-ellipsoidal Support Vector Machine

For a given set of m training examples, $\{\mathbf{x}_i\}_{i=1}^{m}$, where each example is a member of either class I_1 or class I_2, and also let $m_1 = |\mathsf{I}_1|$ and $m_2 = |\mathsf{I}_2|$, TESVM solves the following pair of optimization problems to find two minimum volume hyperellipsoids, $\mathsf{E}_{\mathbf{E}_1,\mathbf{c}_1}$ and $\mathsf{E}_{\mathbf{E}_2,\mathbf{c}_2}$, where each hyperellipsoid is closest to one class, but also as far as possible from the other class.

$$\begin{aligned} \min_{\mathbf{E}_1,\mathbf{c}_1,r_1,\xi} \quad & r_1^2 + 2\log\det(\mathbf{E}_1^{-1}) - \frac{\nu_1}{m_2}\sum_{j\in\mathsf{I}_2}\|\mathbf{E}_1\mathbf{x}_j - \mathbf{c}_1\|^2 + \frac{C_1}{m_1}\sum_{i\in\mathsf{I}_1}\xi_i \\ \text{s.t.} \quad & \|\mathbf{E}_1\mathbf{x}_i - \mathbf{c}_1\|^2 \le r_1^2 + \xi_i \\ & \xi_i \ge 0,\ i \in \mathsf{I}_1 \end{aligned} \tag{3.18}$$

$$\begin{aligned} \min_{\mathbf{E}_2,\mathbf{c}_2,r_2,\xi} \quad & r_2^2 + 2\log\det(\mathbf{E}_2^{-1}) - \frac{\nu_2}{m_1}\sum_{i\in\mathsf{I}_1}\|\mathbf{E}_2\mathbf{x}_i - \mathbf{c}_2\|^2 + \frac{C_2}{m_2}\sum_{j\in\mathsf{I}_2}\xi_j \\ \text{s.t.} \quad & \|\mathbf{E}_2\mathbf{x}_j - \mathbf{c}_2\|^2 \le r_2^2 + \xi_j \\ & \xi_j \ge 0,\ j \in \mathsf{I}_2 \end{aligned} \tag{3.19}$$

where ν_1, C_1, ν_2, and C_2 are hyperparameters. r_1 and r_2 are the Mahalanobis distance from the center of $\mathsf{E}_{\mathbf{E}_1,\mathbf{c}_1}$ and $\mathsf{E}_{\mathbf{E}_2,\mathbf{c}_2}$, respectively.

The illustration of TESVM's decision boundary can be illustrated in Figure 3.3 where ○ and ● are the examples from class I_1 and I_2, respectively. The dash line is the

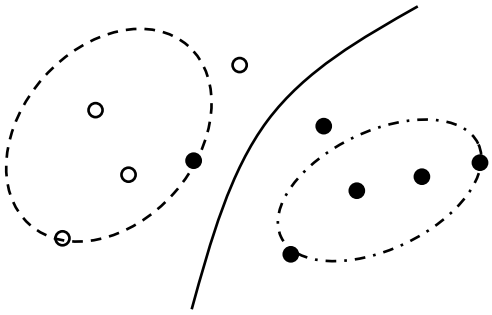

Figure 3.3 Rough sketch of TESVM's decision boundary

descriptive boundary for class I_1 and the dot-dash line is the boundary for class I_2. The decision boundary is shown in the solid line.

The optimization (3.18) and (3.19) are almost exactly the same where their minimizations are to find a hyperellipsoid around class I_1 and I_2, respectively. Therefore, we only focus on (3.18) for brevity.

The first two terms of (3.18) represent the volume of $E_{\mathbf{E}_1,\mathbf{c}_1}$ to be minimized. We include r_1^2 (also r_2^2) into the formulation to make it like THSVM as well as the formulation of MVCE in (2.17). The constraints of (3.18) define that $E_{\mathbf{E}_1,\mathbf{c}_1}$ must cover class I_1 while some examples are allowed to stay outside by the slack variables. The third term of (3.18) sets the objective that the optimal hyperellipsoid should be placed as far as possible from class I_2.

From (3.18), the Lagrangian can be obtained as

$$L(\mathbf{E}_1,\mathbf{c}_1,r_1,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\beta}) = r_1^2 - 2\log\det(\mathbf{E}_1) - \frac{\nu_1}{m_2}\sum_{j\in I_2}\left\|\mathbf{E}_1\mathbf{x}_j - \mathbf{c}_1\right\|^2 + \frac{C_1}{m_1}\sum_{i\in I_1}\xi_i + \sum_{i\in I_1}\alpha_i\left[\left\|\mathbf{E}_1\mathbf{x}_i - \mathbf{c}_1\right\|^2 - r_1^2 - \xi_i\right] - \sum_{i\in I_1}\beta_i\xi_i \qquad (3.20)$$

where $\alpha_i \geq 0$ and $\beta_i \geq 0$ for $i = 1,2,...,m_1$ are Lagrange multipliers. $\boldsymbol{\alpha}$ and $\boldsymbol{\beta} \in \mathbb{R}^{m_1}$ are their corresponding column vectors, respectively.

The formulation can be more compact by assigning the index for each example in class I_1 to be from 1 to m_1 and class I_2 to be from m_1+1 to m. We extend $\boldsymbol{\alpha}$ from $\mathbb{R}^{m_1}$ to $\mathbb{R}^{m}$ with

$$\alpha_{m_1+1} = \alpha_{m_1+2} = ... = \alpha_m = -\frac{\nu_1}{m_2}. \tag{3.21}$$

Therefore, (3.20) can be rewritten as

$$L(\mathbf{E}_1, \mathbf{c}_1, r_1, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = -2\log\det(\mathbf{E}_1) + \sum_{i=1}^{m} \alpha_i \left\| \mathbf{E}_1\mathbf{x}_i - \mathbf{c}_1 \right\|^2 + r_1^2 \left(1 - \sum_{i=1}^{m_1} \alpha_i \right) + \sum_{i=1}^{m_1} \left(\frac{C_1}{m_1} - \alpha_i - \beta_i \right) \xi_i \tag{3.22}$$

The first-order derivatives of (3.22) are

$$\frac{\partial L}{\partial r_1} = 2r_1 \left(1 - \sum_{i=1}^{m_1} \alpha_i \right) \tag{3.23}$$

$$\frac{\partial L}{\partial \xi_i} = \frac{C_1}{m_1} - \alpha_i - \beta_i, \qquad i = 1, 2, ..., m_1 \tag{3.24}$$

$$\frac{\partial L}{\partial \mathbf{c}_1} = 2\left(\mathbf{c}_1 \sum_{i=1}^{m} \alpha_i - \mathbf{E}_1 \sum_{i=1}^{m} \alpha_i \mathbf{x}_i \right) \tag{3.25}$$

$$\frac{\partial L}{\partial \mathbf{E}_1} = -2\mathbf{E}_1^{-1} + \sum_{i=1}^{m} \alpha_i \left[(\mathbf{E}_1\mathbf{x}_i - \mathbf{c}_1)\mathbf{x}_i^T + \mathbf{x}_i(\mathbf{E}_1\mathbf{x}_i - \mathbf{c}_1)^T \right] \tag{3.26}$$

Under the first-order necessary condition of optimality, (3.23) yields $\sum_{i=1}^{m_1} \alpha_i = 1$, and (3.24) yields $0 \le \alpha_i \le \frac{C_1}{m_1}$ for $i = 1, 2, ..., m_1$. Since we prefer the representation of the problem in a matrix form, let $\mathbf{A} = \text{diag}(\alpha_1, \alpha_2, ..., \alpha_m)$ and $\mathbf{X} = \left[\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m \right]$. Hence, from (3.25), we obtain

$$\mathbf{c}_1 = \mathbf{E}_1\mathbf{X}\boldsymbol{\alpha}. \tag{3.27}$$

By substituting (3.27) into (3.26), we also have

$$\mathbf{E}_1^{-1} = \frac{1}{2}(\mathbf{E}_1\mathbf{S} + \mathbf{S}\mathbf{E}_1) \tag{3.28}$$

where $\mathbf{S} = \mathbf{X}\mathbf{A}\mathbf{X}^T - \mathbf{X}\boldsymbol{\alpha}\boldsymbol{\alpha}^T\mathbf{X}^T$. The solution to (3.28) is $\mathbf{E}_1 = \mathbf{S}^{-1/2}$, following (2.15). Despite the fact that $\mathbf{E}_1$ may not be unique since $\mathbf{S}$ may not be positive definite, the log-

determinant term in the objective of (3.18) is, in fact, a natural barrier function to force $\mathbf{E}_1$ to a positive definite solution.

By rewriting (3.22) with KKT conditions, the dual formulation of (3.18) can be obtained as

$$\begin{aligned} \max_{\alpha_1,\ldots,\alpha_{m_1}} \quad & \log\det\left(\mathbf{X}\mathbf{A}\mathbf{X}^T - \mathbf{X}\boldsymbol{\alpha}\boldsymbol{\alpha}^T\mathbf{X}^T\right) \\ \text{s.t.} \quad & \sum_{i=1}^{m_1}\alpha_i = 1, \quad 0 \le \alpha_i \le \frac{C_1}{m_1}, \quad i = 1,2,\ldots,m_1 \end{aligned} \tag{3.29}$$

The optimization (3.29) is a standard semidefinite program and can be solved with a standard solver such as [69] which supports log-determinant in the objective function.

The complementary slackness conditions can also be derived, for $i = 1, 2, \ldots m_1$,

$$0 = \beta_i\xi_i \tag{3.30}$$

$$0 = \alpha_i\left[\left\|\mathbf{E}_1\mathbf{x}_i - \mathbf{c}_1\right\|^2 - r_1^2 - \xi_i\right] \tag{3.31}$$

By selecting an example with $0 < \alpha_i < \frac{C_1}{m_1}$, we obtain $\xi_i = 0$ from (3.30). Therefore, r can later be computed from (3.31).

After $\mathsf{E}_{\mathbf{E}_1,\mathbf{c}_1}$ is found, $\mathsf{E}_{\mathbf{E}_2,\mathbf{c}_2}$ also can be obtained from solving (3.19). Finally, the decision rule for the TESVM is defined by

$$\text{Class } k = \arg\min_{k=1,2}\frac{\left\|\mathbf{E}_k\mathbf{x} - \mathbf{c}_k\right\|}{r_k} \tag{3.32}$$

where $\mathbf{x} \in \mathbb{R}^n$ is a given testing example.

3.2.1 Connection to MVCE with negative examples

TESVM is the direct extension of THSVM. Its formulation is also related to the eSVDD with negative examples (3.2) proposed in Section 3.1. It is the same idea that THSVM can be viewed as double SVDDs with negative examples. In general, given I_1 and I_2 to be the positive and negative classes, respectively, a soft-margin MVCE problem with negative classes can be formulated as

$$\begin{aligned} \min_{\mathbf{E},\mathbf{c},r,\xi} \quad & r^2 + \log\det(\mathbf{E}^{-1}) + \frac{C}{m}\sum_{i\in \mathsf{I}_1\cup\mathsf{I}_2}\xi_i \\ \text{s.t.} \quad & y_i\left\|\mathbf{E}\mathbf{x}_i - \mathbf{c}_1\right\|^2 \le y_i r^2 + \xi_i \\ & \xi_i \ge 0,\ i \in \mathsf{I}_1\cup\mathsf{I}_2 \end{aligned} \tag{3.33}$$

where $y_i = 1$ for $i \in \mathsf{I}_1$ and $y_i = -1$ for $i \in \mathsf{I}_2$. The solution to (3.33) is the minimum volume covering hyperellipsoid $\mathsf{E}_{\mathbf{E},\mathbf{c}}$ which covers the positive class and excludes the negative class. Therefore, (3.33) differs from (3.18) only in that the excluding constraints of in (3.33) are moved into the objective of (3.18).

The dual formulation of (3.33) is

$$\begin{aligned} \max_{\alpha_1,\ldots,\alpha_m} \quad & \log\det\left(\mathbf{X}\mathbf{A}\mathbf{Y}\mathbf{X}^T - \mathbf{X}\mathbf{Y}\boldsymbol{\alpha}\boldsymbol{\alpha}^T\mathbf{Y}\mathbf{X}^T\right) \\ \text{s.t.} \quad & \sum_{i\in \mathsf{I}_1\cup\mathsf{I}_2} y_i\alpha_i = 1,\quad 0 \le \alpha_i \le \frac{C}{m},\quad i \in \mathsf{I}_1\cup\mathsf{I}_2 \end{aligned} \tag{3.34}$$

and it has the same form as (3.29) but with some modifications. That is the Lagrange multipliers belonging to the negative class are the optimization variables. They are not simply a constant as in (3.21). The sum of *m* Lagrange multipliers of (3.34) also must be one and the box constraints $0 \le \alpha_i \le C/m$ also confine the Lagrange multipliers. It is worth nothing that one subproblem of TESVM in the dual formulation requires to solve smaller numbers of optimization variables than one MVCE problem with negative examples.

3.3 Ellipsoids with Kernel Methods

In the previous sections, we directly formulate the minimal hyperellipsoids as a classification tool for both one-class and two-class classification problems. Even though hyperellipsoids provide less conservative boundary than hyperspheres, they are in some cases not adequate to describe complex patterns.

In general, learning machines utilize kernel methods to enhance more classification ability by replacing all inner product terms $\mathbf{x}_i^T\mathbf{x}_j$ in their formulation with a kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$. Because an inner product is a measurement of similarity between

two examples, by replacing it with a kernel function, we also obtain an alternative similarity measurement and hope that the training examples are mapped into a space with better class separability [75].

Generally, a kernel is a positive definite function $k(\cdot,\cdot):\Re^n \times \Re^n \mapsto \Re$ satisfying Mercer's conditions. These conditions guarantee the explicit ability to factorize a kernel to be an inner product between two vectors, i.e., $k(\mathbf{x}_i,\mathbf{x}_j)=\boldsymbol{\varphi}(\mathbf{x}_i)^T\boldsymbol{\varphi}(\mathbf{x}_j)$, where $\boldsymbol{\varphi}(\mathbf{x}_i):\Re^n \mapsto \mathsf{H}$, and H is the feature space. Therefore, a kernel also defines an inner product in H, alternatively to the inner product in the input space $\Re^n$. The mapping $\boldsymbol{\varphi}(\mathbf{x})$ is generally unknown and the dimension of space H is usually much higher than *n*. In fact, for some kernel functions, it is possible that the dimension of feature space is infinite.

Given an *m* training examples $\{\mathbf{x}_i\}_{i=1}^m$, we want to map all examples into the feature space using an unknown mapping function $\boldsymbol{\varphi}(\mathbf{x})$. Let's also denote

$$\boldsymbol{\Phi}=[\boldsymbol{\varphi}(\mathbf{x}_1),\boldsymbol{\varphi}(\mathbf{x}_2),...,\boldsymbol{\varphi}(\mathbf{x}_m)] \tag{3.35}$$

and define the so-called the kernel matrix $\mathbf{K}\in\mathsf{S}_+^m$ to be $\mathbf{K}=\boldsymbol{\Phi}^T\boldsymbol{\Phi}$. The kernel matrix is simply the composition of $k(\mathbf{x}_i,\mathbf{x}_j)$ for all *i* and *j*, i.e.

$$\mathbf{K}=\begin{bmatrix} k(\mathbf{x}_1,\mathbf{x}_1) & k(\mathbf{x}_1,\mathbf{x}_2) & \cdots & k(\mathbf{x}_1,\mathbf{x}_m) \\ k(\mathbf{x}_2,\mathbf{x}_1) & k(\mathbf{x}_2,\mathbf{x}_2) & \cdots & k(\mathbf{x}_2,\mathbf{x}_m) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_m,\mathbf{x}_1) & k(\mathbf{x}_m,\mathbf{x}_2) & \cdots & k(\mathbf{x}_m,\mathbf{x}_m) \end{bmatrix}. \tag{3.36}$$

Since most if not all learning machines in the family of support vector classifiers are compatible with kernel methods, it is also interesting to see the construction of the MVCE in the feature space using kernel methods. However, applying kernel tricks to MVCE is not a straightforward task since the formulation of MVCE consists of outer products, instead of inner products. In the following subsections, we will first review the current state of the art in kernelizing the MVCE, followed by our proposed method which is a simpler framework to construct the MVCE in the feature space.

3.3.1 Existing literature on kernelizing MVCE

To the best of our knowledge, there are very few research studies working on kernelizing the MVCE. More specifically, those publications are by Dolia et al. [57-58] and Wei et al. [59-60] where they try to rewrite the MVCE formulation in the form of inner products with the help of some matrix factorizations. In fact, both methods from Dolia et al. and Wei et al. are conceptually the same and can be summarized as follows.

First, rewrite the objective of (2.20) i.e. the objective of

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \log\det(\tilde{\mathbf{X}}\mathbf{A}\tilde{\mathbf{X}}^T) \\ \text{s.t.} \quad & \mathbf{e}^T\boldsymbol{\alpha} = 1 \\ & \boldsymbol{\alpha} \geq 0. \end{aligned} \tag{2.20}$$

by mapping $\mathbf{X}$ to $\boldsymbol{\Phi}$ where $\boldsymbol{\Phi}$ represents the matrix of training examples in the feature space. As $\tilde{\mathbf{X}} = [\mathbf{X}^T \quad \mathbf{e}]^T$, we also denote $\tilde{\boldsymbol{\Phi}} = [\boldsymbol{\Phi}^T \quad \mathbf{e}]^T$ as the image of $\tilde{\mathbf{X}}$ in the feature space. Hence, we have

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \log\det(\tilde{\boldsymbol{\Phi}}\mathbf{A}\tilde{\boldsymbol{\Phi}}^T) \\ \text{s.t.} \quad & \mathbf{e}^T\boldsymbol{\alpha} = 1 \\ & \boldsymbol{\alpha} \geq 0. \end{aligned} \tag{3.37}$$

However, $\boldsymbol{\Phi}$ is normally unknown. By utilizing the fact that, for any matrices $\mathbf{M}$ and $\mathbf{N}$, $\mathbf{MN}$ and $\mathbf{NM}$ have the same nonzero eigenvalues. In addition, together with using the Cholesky decomposition of the kernel matrix in the augmented feature space, we have

$$\begin{aligned} \log\det(\tilde{\boldsymbol{\Phi}}\mathbf{A}\tilde{\boldsymbol{\Phi}}^T) &= \log\det((\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2})(\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2})^T) \\ &= \log\det((\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2})^T(\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2})) \\ &= \log\det(\mathbf{A}^{1/2}\tilde{\boldsymbol{\Phi}}^T\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2}) \\ &= \log\det(\mathbf{A}^{1/2}\tilde{\mathbf{K}}\mathbf{A}^{1/2}) \qquad (\text{Note: } \tilde{\mathbf{K}} = \mathbf{K} + \mathbf{e}\mathbf{e}^T) \\ &= \log\det(\mathbf{A}^{1/2}\mathbf{C}^T\mathbf{C}\mathbf{A}^{1/2}) \qquad (\text{Note: } \tilde{\mathbf{K}} = \mathbf{C}^T\mathbf{C}) \\ &= \log\det((\mathbf{C}\mathbf{A}^{1/2})^T(\mathbf{C}\mathbf{A}^{1/2})) \\ &= \log\det((\mathbf{C}\mathbf{A}^{1/2})(\mathbf{C}\mathbf{A}^{1/2})^T) \\ &= \log\det(\mathbf{C}\mathbf{A}\mathbf{C}^T). \end{aligned} \tag{3.38}$$

Hence, (3.37) becomes

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \log\det(\mathbf{C}\mathbf{A}\mathbf{C}^T) \\ \text{s.t.} \quad & \mathbf{e}^T\boldsymbol{\alpha} = 1 \\ & \boldsymbol{\alpha} \geq 0. \end{aligned} \tag{3.39}$$

As a result, directly solving (3.37) can be avoid by using the formulation (3.39) with the fact that $\mathbf{C}$ can be obtained by factorizing $\mathbf{K} + \mathbf{e}\mathbf{e}^T$.

After the objective of MVCE can be rewritten as a function of the kernel matrix, Dolia et al. and Wei et al. then also try to rewrite the weighted norm in the augmented feature space without explicitly using the function $\boldsymbol{\varphi}$. For a given $\mathbf{x} \in \mathbb{R}^n$, the weighted norm weighted by $\tilde{\mathbf{E}}$ of appropriated dimensions is defined as $f(\mathbf{x}) = \tilde{\mathbf{x}}^T\tilde{\mathbf{E}}\tilde{\mathbf{x}}$. In the feature space, the norm becomes $f(\mathbf{x}) = \tilde{\boldsymbol{\varphi}}^T\tilde{\mathbf{E}}\tilde{\boldsymbol{\varphi}}$. However, in fact, $\tilde{\mathbf{E}}$ comes in the form of an inverse matrix, i.e. $\tilde{\mathbf{E}}^{-1} = \tilde{\boldsymbol{\Phi}}\mathbf{A}\tilde{\boldsymbol{\Phi}}^T$. Dolia et al. and Wei et al. utilize the singular value decomposition (SVD) of $\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2}$ as

$$\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T \tag{3.40}$$

where $\mathbf{U}$, $\mathbf{V}$, and $\boldsymbol{\Sigma}$ are the eigenvector matrix of $\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2}(\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2})^T$, the eigenvector matrix of $(\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2})^T\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2}$, and singular values of $\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2}$, respectively. It follows that

$$\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2}(\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2})^T = \tilde{\boldsymbol{\Phi}}\mathbf{A}\tilde{\boldsymbol{\Phi}}^T = \mathbf{U}\boldsymbol{\Sigma}\boldsymbol{\Sigma}^T\mathbf{U}^T = \tilde{\mathbf{E}}^{-1} \tag{3.41}$$

and

$$(\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2})^T\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2} = \mathbf{A}^{1/2}\tilde{\boldsymbol{\Phi}}^T\tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2} = \mathbf{A}^{1/2}\tilde{\mathbf{K}}\mathbf{A}^{1/2} = \mathbf{V}\boldsymbol{\Sigma}^T\boldsymbol{\Sigma}\mathbf{V}^T. \tag{3.42}$$

Therefore, (3.41) and (3.42) imply that $\tilde{\mathbf{E}}^{-1}$ and $\mathbf{A}^{1/2}\tilde{\mathbf{K}}\mathbf{A}^{1/2}$ share the same set of non-zero eigenvalues. Since $\tilde{\mathbf{K}}$ is known from the training examples and $\mathbf{A}$ is also known from the solution of (3.39), we can easily compute $\mathbf{A}^{1/2}\tilde{\mathbf{K}}\mathbf{A}^{1/2}$. As a result, the eigendecomposition of $\mathbf{A}^{1/2}\tilde{\mathbf{K}}\mathbf{A}^{1/2}$ yields the matrices $\mathbf{V}$ and $\boldsymbol{\Sigma}^T\boldsymbol{\Sigma}$.

Since, from (3.41), we have $\tilde{\mathbf{E}} = \mathbf{U}(\boldsymbol{\Sigma}\boldsymbol{\Sigma}^T)^{-1}\mathbf{U}^T$, the weight norm can be rewritten as

$$f(\mathbf{x}) = \tilde{\boldsymbol{\varphi}}^T \mathbf{U}\left(\boldsymbol{\Sigma}\boldsymbol{\Sigma}^T\right)^{-1}\mathbf{U}^T\tilde{\boldsymbol{\varphi}}. \tag{3.43}$$

Also from (3.40), $\mathbf{U} = \tilde{\boldsymbol{\Phi}}\mathbf{A}^{1/2}(\boldsymbol{\Sigma}\mathbf{V}^T)^+$, where $(\cdot)^+$ is a pseudoinverse operator. Therefore, it is possible to entirely eliminate the unknown function $\boldsymbol{\varphi}$ from the weighted norm, $f(\mathbf{x}) = \tilde{\boldsymbol{\varphi}}^T\tilde{\mathbf{E}}\tilde{\boldsymbol{\varphi}}$, by simply replacing any multiplication pairs between the term $\tilde{\boldsymbol{\varphi}}$ in $f(\mathbf{x})$ and the term $\tilde{\boldsymbol{\Phi}}$ in $\mathbf{U}$ with the vector of kernel functions.

3.3.2 Constructing MVCE in the feature space via empirical feature mapping

Although Dolia et al. [57-58] and Wei et al. [59-60] successfully proposed a method to kernelize MVCE as described in Section 3.3.1, it is important to emphasize that their works rely on the factorization of $\mathbf{A}^{1/2}\tilde{\mathbf{K}}\mathbf{A}^{1/2}$. Although we believe that the factorization is an indispensable step toward kernelizing the MVCE, their methods are over-complicated. Its formulation relies on the factorization of the constituent of the kernel matrix $\mathbf{K}$ and the Lagrange multiplier's matrix $\mathbf{A}$. Precisely, the fact that their approach must factorize the matrix $\mathbf{A}^{1/2}\tilde{\mathbf{K}}\mathbf{A}^{1/2}$ makes it rely on the structure of the problem.

As a result, in this section, we propose an alternative solution based on the empirical feature mapping, or the kernel principle component analysis (kernel PCA) mapping which is a more elegant but simple answer to kernelizing the MVCE. The empirical feature mapping in fact is not a new concept in the literature. It has been studied by many researchers, including [75-76]; however, its use in the MVCE problem is still unexplored.

For a given set of m training examples $\{\mathbf{x}_i\}_{i=1}^m$, we would like to define a map from the input space $\mathbb{R}^n$ to a space called "empirical feature space H_E" such that the inner product in H_E is equal to the one in H.

Definition 3.1: Given the training examples $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m \in \mathbb{R}^n$. The kernel PCA map from $\mathbb{R}^n$ to H_E is defined as

$$\boldsymbol{\varphi}_E : \mathbf{x} \mapsto (\boldsymbol{\Omega}^+)^T \mathbf{k}(\mathbf{x})$$

where $\boldsymbol{\Omega}$ satisfies $\mathbf{K} = \boldsymbol{\Omega}^T \boldsymbol{\Omega}$ and $\mathbf{k}(\mathbf{x}) = [k(\mathbf{x}, \mathbf{x}_1), k(\mathbf{x}, \mathbf{x}_2), ..., k(\mathbf{x}, \mathbf{x}_m)]^T$.

Corollary 3.1: The empirical feature space H_E and the feature space H have the same inner product and Euclidean distance.

Proof: Let $\mathbf{k}_i = \mathbf{k}(\mathbf{x}_i)$, $\boldsymbol{\varphi}_i = \boldsymbol{\varphi}(\mathbf{x}_i)$, and $\boldsymbol{\varphi}_{E_i} = \boldsymbol{\varphi}_E(\mathbf{x}_i)$. It follows that $\boldsymbol{\varphi}_{E_i}^T \boldsymbol{\varphi}_{E_j} = \mathbf{k}_i^T \boldsymbol{\Omega}^+ (\boldsymbol{\Omega}^+)^T \mathbf{k}_j = \mathbf{k}_i^T \mathbf{K}^+ \mathbf{k}_j = \boldsymbol{\varphi}_i^T \boldsymbol{\Phi} (\boldsymbol{\Phi}^T \boldsymbol{\Phi})^+ \boldsymbol{\Phi}^T \boldsymbol{\varphi}_j = \boldsymbol{\varphi}_i^T \boldsymbol{\varphi}_j = k(\mathbf{x}_i, \mathbf{x}_j)$. The Euclidean distance in both space is also the same according to $\left\| \boldsymbol{\varphi}_i - \boldsymbol{\varphi}_j \right\|^2 = k(\mathbf{x}_i, \mathbf{x}_i) - 2k(\mathbf{x}_i, \mathbf{x}_j) + k(\mathbf{x}_j, \mathbf{x}_j)$. □

From the definition, kernel PCA mapping explicitly defines a map from the input space to an empirical feature space H_E. The term "empirical" indicate that the map is created from the given empirical measures. The space H_E is a Euclidean space with a finite dimension and it is different from the feature space H which can possibly have infinite dimensions such as in the case of the RBF kernel.

Even though the dimension of H may be infinite, learning machines usually perform only in a subspace of H spanned by the images of the training examples $\{\boldsymbol{\varphi}(\mathbf{x}_i)\}_{i=1}^m$. Since the inner product and the Euclidean distance in H_E are the same as in H as in Corollary 3.1, the separability between examples is preserved between both spaces. Mathematically, we say that H is isomorphic with H_E [75].

As a result, in this thesis, we suggest that the MVCE in the kernel-defined feature space should be constructed via the use of the empirical feature map, avoiding the need in trying to reformulate the MVCE problem in terms of inner products. That is, from a given set of training examples $\{\mathbf{x}_i\}_{i=1}^m$, we

first compute the kernel matrix $\mathbf{K}$. Then, factorize it so that we obtain $\mathbf{\Omega}$. After that, follow the mapping defined in Definition 3.1. Hence, the image of an example in the empirical feature space can be obtained.

One benefit of the empirical feature map is that it allows the existing formulations to seamlessly work in the feature space. Hence, the formulations of the proposed eSVDD with negative examples (3.12) and TESVM (3.34) can now readily be equipped with the kernel methods. Furthermore, with the empirical feature map, an example in the input space can be visualized in the feature space. The center and boundary of the hyperellipsoid $\mathsf{E}_{\mathbf{E},\mathbf{c}}$ can also be computed.

It is worth noting that, although a kernel matrix $\mathbf{K}$ which satisfies Mercer's condition can always be factorized as $\mathbf{K} = \mathbf{\Omega}^T\mathbf{\Omega}$, the decomposed matrix $\mathbf{\Omega}$ is not unique for one $\mathbf{K}$. There is more than one approach in factorizing $\mathbf{K}$, such as eigenvalue decomposition and LDL decomposition, and each method results in different $\mathbf{\Omega}$. In eigendecomposition, suppose $\mathbf{K} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$, then, we have $\mathbf{\Omega} = (\mathbf{V}\mathbf{\Lambda}^{1/2})^T$, where $\mathbf{\Lambda}$ is the diagonal eigenvalue matrix corresponding to the eigenvector matrix $\mathbf{V}$. For LDL decomposition, $\mathbf{K} = \mathbf{L}\mathbf{D}\mathbf{L}^T$, we have $\mathbf{\Omega} = (\mathbf{L}\mathbf{D}^{1/2})^T$ where $\mathbf{L}$ is the lower triangular matrix whose diagonal elements are ones and $\mathbf{D}$ is a diagonal matrix. In general, both $\mathbf{\Lambda}$ and $\mathbf{D}$ are not full-rank because $\mathbf{K}$ is positive semidefinite. In this thesis, therefore, it is necessary to use a reduced or truncated version of the decomposition.