CHAPTER 4

Experimental Results and Discussions

This chapter is divided into two sections with the aim to evaluate all the proposed algorithms presented in Chapter 3. In the first section, the performance of the neSVDD is evaluated on one-class classification problems. After that, the second section proceeds to assess the performance of the TESVM on binary classification.

4.1 One-class Classification with Ellipsoidal Support Vector Data Description

All experiments conducted in this section are performed on the system equipped with Intel Core i7-4790 processor and 8 GB of ram under 64-bit Ubuntu 14.10. All classifiers were developed using MATLAB 2014a and the log-determinant minimization was solved by the SDPT3 solver [69] interfaced via YALMIP [70].

4.1.1 Standard datasets

Four one-class classification methods which are the target of the evaluation are SVDD, nSVDD, eSVDD, and the proposed neSVDD. That is, the comparison can be divided into two scenarios. The first scenario is the comparison between ellipsoidal and spherical boundaries, i.e. SVDD vs. eSVDD, and nSVDD vs. neSVDD. The second scenario is the benefit of the inclusion of negative examples into the formulations, i.e. SVDD vs. nSVDD, and eSVDD vs. neSVDD.

To simplify the experimental process, we used the implementation of SVDD and nSVDD from DDtools [77]. In short, DDtools is a MATLAB's toolbox which is specifically designed for testing one-class classification algorithms and is provided by D. Tax, the author of SVDD. In order to compare SVDD against eSVDD, we also add our own implementation of neSVDD

Table 4.1 Standard datasets for one-class classification

	Dataset	n	m_1	m_2
1	Balance-scale (left)	4	288	337
2	Balance-scale (middle)	4	49	576
3	Balance-scale (right)	4	288	337
4	Cancer wpbc (nonret)	33	151	47
5	Cancer wpbc (ret)	33	47	151
6	Ecoli (periplasm)	7	52	284
7	Glass (bldg. float)	9	70	144
8	Glass (bldg. nonfloat)	9	76	138
9	Glass (containers)	9	13	201
10	Glass (headlamps)	9	29	185
11	Glass (vehicle float)	9	17	197
12	Hepatitis (live)	19	123	32
13	Housing (MEDV>35)	13	48	458
14	Imports (low risk)	25	71	88
15	Iris (setosa)	4	50	100
16	Iris (versicolor)	4	50	100
17	Iris (virginica)	4	50	100
18	Liver (1)	6	145	200
19	Liver (2)	6	200	145
20	Sonar (mines)	60	111	97
21	Sonar (rocks)	60	97	111
22	Spectf (0)	44	95	254
23	Spectf (1)	44	254	95
24	Thyroid (normal)	21	93	3679
25	Wine (1)	13	59	119
26	Wine (2)	13	71	107
27	Wine (3)	13	48	130

to the toolbox. The implementation of eSVDD is, in fact, the same as neSVDD but without considering negative examples. The model of neSVDD is trained using the dual formulation (3.13) with the classification rule defined in (3.15).

To justify the performance of these classifiers, we conducted some experiments using the standard benchmark datasets as listed in Table 4.1 where n, m_1, and m_2 denote the number of features, target examples, and outliers, respectively. These datasets were also obtained from D. Tax's webpage [78] in prtools format [79] and were already preprocessed by the owner such as filling missing values. In addition, as a remark, the datasets are created from multiclass datasets by assigning one class of data to be the target class, and the rest to be outliers

In the experiments, the training data were further preprocessed by scaling to one before the training process. We use only target classes to train SVDD

and eSVDD. Furthermore, because of its popularity, we choose the RBF kernel as the kernel function for the experiment in this thesis. In addition, the RBF kernel is also a perfect choice for SVDD according to [31]. As a result, these ellipsoidal and spherical domain description algorithms are compared against one another under the same RBF kernel function. We select two hyperparameters for SVDD, nSVDD, eSVDD, and neSVDD by using grid search over 120 pairs of parameters. The best value of σ is searched from the set {0.5, 1, 5, 10, 15, 20, 25, 30, 35, 40, 45, 50} and C is searched from 1/(mr) for $r \in [0.1, 0.2, ..., 1.0]$.

For eSVDD, we set the search range of C to $N/(mr)$ where $r \in [0.1, 0.2, ..., 1.0]$. N is the approximate dimension of the empirical feature space. We factorize the kernel matrix $\mathbf{K}$ using eigendecomposition which provides $\mathbf{\Omega} = (\mathbf{L}\mathbf{D}^{1/2})^T$. After that kernel PCA map can be computed. Nevertheless, the rank of $\mathbf{D}$ is not always full. Therefore, we truncated its diagonal elements smaller than 10^{-5}. The dimension of the reduced $\mathbf{D}$ matrix is in fact the dimension of the empirical feature space. Moreover, a degenerate case of ellipsoids has to be avoided, when Assumption 2.1 does not hold, by adding a weighted identity matrix $\gamma\mathbf{I}$ of the appropriate dimension inside the logdet function. In the experiment, the weight γ is set to $\gamma = 10^{-5}$.

For each choice of the hyperparameter pair C and σ, ten-fold cross-validation was used. In order to ensure the consistency of the results, we evaluated all four ellipsoidal and spherical SVDDs on the same partitioned data of the cross-validation. However, in order to reduce the training time, for 9 out of 10 folds used for training nSVDD and neSVDD, negative examples were limited to only 20 examples, instead of $9m_2/10$. The best set of hyperparameters was determined using the maximum mean of the area under the Receiver Operating Characteristic (ROC) curve.

The ROC curve is generally the plot between false positive rates and true positive rates. The area under the curve is at most equal to one. To construct

Table 4.2 The comparison of the area under ROC for one-class classification

Dataset	Spherical SVDD		Ellipsoidal SVDD	
	SVDD	nSVDD	eSVDD	neSVDD
Balance-scale (left)	0.9665 (0.0204)	0.9671 (0.0201)*	0.9879 (0.0091)	**0.9896** (0.0102)*
Balance-scale (middle)	0.8009 (0.1099)	0.8014 (0.1082)*	0.9221 (0.0476)	**0.9845** (0.0377)*
Balance-scale (right)	0.9665 (0.0188)	0.9676 (0.0184)*	0.9870 (0.0088)	**0.9900** (0.0082)*
Cancer wpbc (nonret)	**0.5433** (0.1233)	0.5362 (0.1305)	0.5239 (0.1527)	0.5335 (0.1503)*
Cancer wpbc (ret)	0.6128 (0.1509)	0.6283 (0.1578)*	**0.6449** (0.1291)	0.6172 (0.1474)
Ecoli (periplasm)	0.9580 (0.0608)	**0.9599** (0.0586)*	0.9414 (0.0592)	0.9472 (0.0585)*
Glass (bldg. float)	0.8000 (0.0943)	0.8008 (0.0893)*	**0.8317** (0.0824)	0.8309 (0.0756)
Glass (bldg. nonfloat)	0.6541 (0.1244)	0.6841 (0.1281)*	**0.7502** (0.1125)	0.7083 (0.1293)
Glass (containers)	0.8269 (0.3286)	0.8269 (0.3286)	0.9787 (0.0371)	**0.9836** (0.0287)*
Glass (headlamps)	**0.9425** (0.0808)	**0.9425** (0.0808)	0.9124 (0.1200)	0.9108 (0.1247)
Glass (vehicle float)	0.7116 (0.1230)	0.7324 (0.1258)*	0.8600 (0.1430)	**0.8896** (0.1448)*
Hepatitis (live)	0.8183 (0.1252)	0.8183 (0.1252)	0.8182 (0.1369)	**0.8284** (0.1249)*
Housing (MEDV>35)	0.8523 (0.0936)	0.8604 (0.0935)*	**0.8905** (0.0766)	0.8867 (0.0822)
Imports (low risk)	0.8338 (0.0968)	0.8351 (0.0964)*	0.7618 (0.1299)	**0.8823** (0.0890)*
Iris (setosa)	**1.0000** (0.0000)	**1.0000** (0.0000)	**1.0000** (0.0000)	**1.0000** (0.0000)
Iris (versicolor)	0.9708 (0.0353)	0.9796 (0.0322)*	**0.9920** (0.0194)	0.9888 (0.0233)
Iris (virginica)	0.9688 (0.0456)	0.9692 (0.0459)*	**0.9796** (0.0337)	0.9704 (0.0369)
Liver (1)	0.5614 (0.0770)	0.5670 (0.0927)*	0.6155 (0.0782)	**0.6187** (0.0792)*
Liver (2)	0.5485 (0.1068)	0.6057 (0.0972)*	0.5771 (0.1059)	**0.6316** (0.1052)*
Sonar (mines)	0.7394 (0.0987)	0.7429 (0.0992)*	0.7879 (0.0919)	**0.8015** (0.0918)*
Sonar (rocks)	0.7179 (0.1109)	0.7209 (0.1103)*	0.6274 (0.1232)	**0.7327** (0.1122)*
Spectf (0)	0.8978 (0.0570)	0.9008 (0.0564)*	**0.9435** (0.0585)	0.9420 (0.0532)
Spectf (1)	0.7153 (0.0845)	**0.7327** (0.0834)*	0.6464 (0.0480)	0.6474 (0.0471)*
Thyroid (normal)	0.7928 (0.0735)	0.8453 (0.0676)*	0.9503 (0.0428)	**0.9644** (0.0497)*
Wine (1)	0.9989 (0.0047)	0.9989 (0.0047)	0.9978 (0.0076)	**0.9991** (0.0037)*
Wine (2)	0.9011 (0.0703)	0.9011 (0.0703)	0.9559 (0.0441)	**0.9713** (0.0411)*
Wine (3)	0.9949 (0.0181)	0.9949 (0.0181)	0.9986 (0.0062)	**0.9989** (0.0058)*

the ROC curve from the classifiers, it is necessary also output membership probabilities in addition to the classification result. For a training example, we define the membership value as $e^{-\|d\|}$ where d is the distance to the center of the descriptive boundary.

After the grid search was performed, the best parameters then were selected. We report the areas under the ROC curves from 5 independent runs of 10-fold cross-validations and the results are presented in Table 4.2. From the table, the best value among four algorithms is emphasized using bold-faced font for each dataset. We also use asterisk (*) to highlight the case when adding negative examples help improve the results for both SVDD and eSVDD.

According to Table 4.2, the number of datasets that each algorithm is the best was 16, 8, 4, and 3 datasets, for neSVDD, eSVDD, nSVDD, and SVDD,

respectively. We observe that eSVDD provides better results than SVDD in 18 out of 26 datasets, and neSVDD is also better than nSVDD for 21 out of 26 datasets. In overall, the ellipsoidal boundary yields superior results to the spherical one for 22 out of 26 datasets. Therefore, changing the descriptive boundary from a hypersphere to a hyperellipsoid help improve the results.

Moreover, Table 4.2 also suggests that negative examples help improve the performance of both SVDD and eSVDD in many datasets. Precisely, according to the number of asterisks in Table 4.2, including negative examples into the formulation enhances the area under the ROC curve for 19 datasets in the case of SVDD, and 18 datasets in the case of eSVDD.

Incorporating negative examples may sometimes decrease the performance of the classifiers. This situation can be observed from Table 4.2 where nSVDD is worse than SVDD for one dataset, and neSVDD is worse than eSVDD for 8 datasets. This is because the presence of some negative examples might not be helpful in creating a better decision boundary, especially when only 20 examples of outliers were used in the experiment. In fact, what is important is the positions of examples which can provide meaningful information to the creation of descriptive boundaries.

Furthermore, in the case of neSVDD, we also address the performance decreases to occasional numerical errors during the simulation. In some cases, the matrix $\tilde{\mathbf{X}}\mathbf{A}\mathbf{Y}\tilde{\mathbf{X}}^T$ in neSVDD is indefinite making the optimization (3.13) unable to be solved. Therefore, to deal with such a problem, we always discarded any pair of cross-validation parameters whenever any training folds could not be trained. However, even with this issue, neSVDD is still superior to nSVDD.

In Appendix B, we provide additional results in Table B.1 which were obtained under the same experimental settings as in Table 4.2; however, instead of using the classification rule (3.15) for the optimization (3.13), the incorrect rule from (3.14) is used. Since (3.14) differs from (3.15) due to missing the proper normalization factor, the results in Table B.1 are unsurprisingly poorer than the ones in Table 4.2.

4.1.2 Effect of hyperparameters

For linear eSVDD, only one hyperparameter C is required to be specified. Since from the formulation of eSVDD (3.12), the sum of all Lagrange multiplies has to equal n. Therefore, eSVDD has the minimum value of C equal to n/m. When C increases, the size of the hyperellipsoid also increases. The size of the eSVDD's ellipsoid is limited by the maximum volume containing the examples as shown in Figure 4.1. The increase of C more than a certain value has no effect on the shape of the ellipsoid. However, this is different from neSVDD as shown in Figure 4.2. The reason is that the constraint $\mathbf{y}^T\mathbf{a} = n$ in neSVDD is still satisfied even for a very large C due to the subtraction between Lagrange multipliers. Thus, the value of a Lagrange multiplier can be as big as the value of C. For eSVDD, however, when we set C too large, a Lagrange multiplier cannot also be too large because of the constraint $\mathbf{e}^T\mathbf{a} = n$.

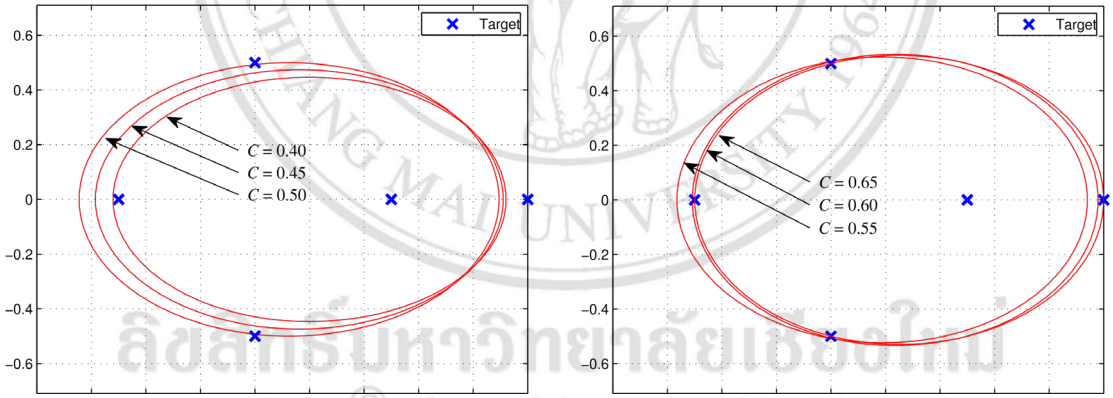


Figure 4.1 Effect of the parameter C on the boundary of eSVDD

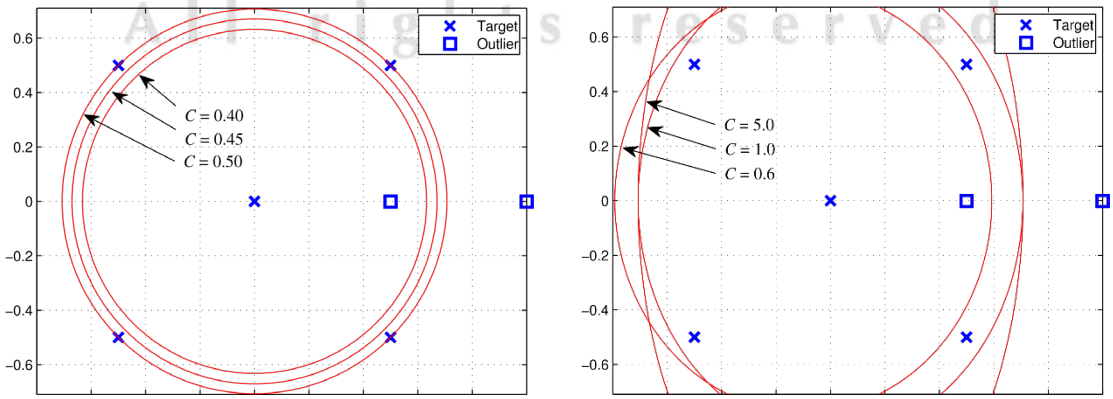
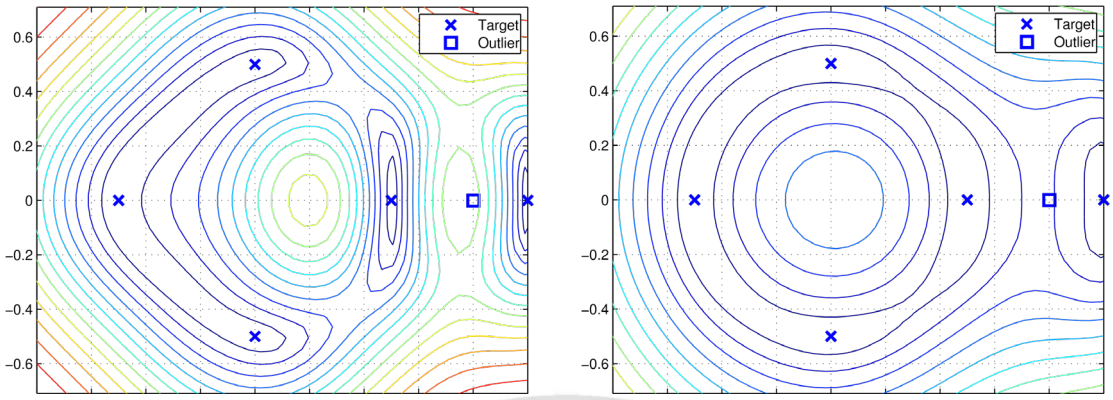


Figure 4.2 Effect of the parameter C on the boundary of neSVDD



(a) $\sigma = 0.50$ (b) $\sigma = 0.75$

Figure 4.3 Contour plots of the effect of σ parameter on neSVDD with RBF kernel

The RBF eSVDD has one more hyperparameter than the linear eSVDD. Figure 4.3 shows the effect of RBF kernels's parameter σ on neSVDD. In the figure, the target class 5 examples and the outlier class have one examples. By decreasing only kernel parameter σ, the data descriptive contours fit tighter to the examples, while the number of support vectors increases.

4.2 Two-class Classification with Twin Hyper-ellipsoidal Support Vector Machine

This section devotes to evaluating the performance of the proposed binary classifier, the twin hyper-ellipsoidal support vector machine (TESVM). Particularly, we compare the accuracy of the proposed method against THSVM and TWSVM, i.e. its spherical and planar counterparts as well as the state-of-the-art method like SVM. All classifiers are implemented using MATLAB 2017a on 64-bit Ubuntu 17.04 under 2.8GHz Intel Core i5-6402P with 8GB of RAM on both artificial and standard benchmark datasets.

Since TESVM is considered as an improvement of THSVM, head-to-head comparisons between the two are presented. All experiments are performed using 10-fold cross-validation and the cross-validation is repeated for 10 times with randomly shuffled examples. We use SDPT3 [69] through YALMIP [70] as a semidefinite programming solver for TESVM because of its support for the log-determinant objective function. The quadratic programming in THSVM and TWSVM is solved using the MATLAB's quadprog solver. For the implementation of SVM, libSVM [80] is used.

4.2.1 Artificial datasets

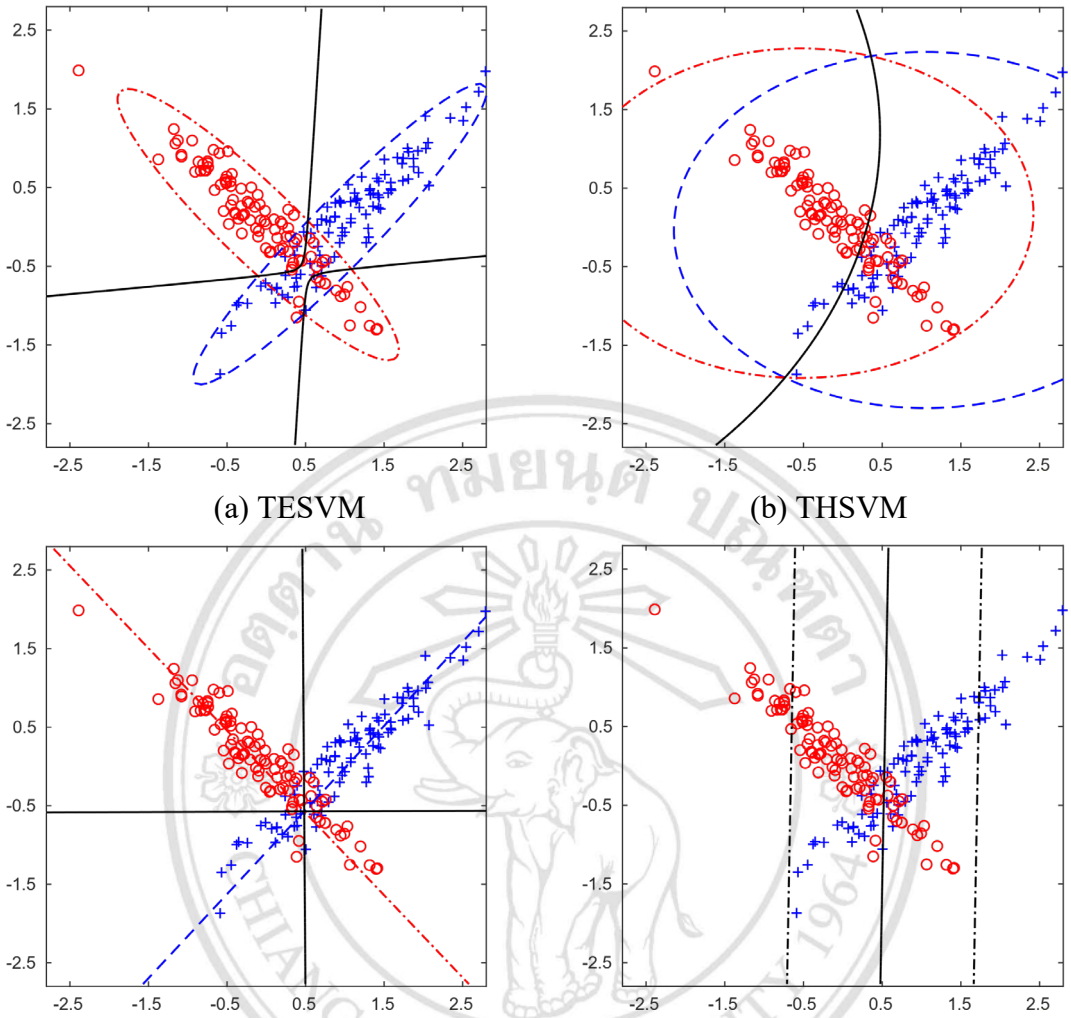
Three 2-dimensional artificial datasets are presented in this section in order to visually compare TESVM, THSVM, SVM, and TWSVM. The details of the data are summarized in Table 4.3. The first two datasets contain non-separable data generated based on Gaussian distributions and are to be classified by linear classifiers, while the third dataset is the Ho- Kleinberg's checkerboard dataset [81] consisting of a series of examples with uniform distributions to form a 4×4 tiles of checkerboard. This dataset is a challenging case for comparing the performance of nonlinear classifiers.

Table 4.3 Artificial datasets for binary classification

Dataset	n	m_1	m_2	m
Toy 1	2	100	100	200
Toy 2	2	50	200	250
Checkerboard	2	514	486	1000

The hyperparameters are chosen by performing a uniform grid search over ranges of parameters. The best set of parameters is the one which provides the best 10-fold cross-validation accuracy. In the case of THSVM, we set $c_1 = c_2 = c$ and $\nu_1 = \nu_2 = \nu$ where $c \in \{10^i \mid i = -5,-4,...,5\}$ and $\nu \in \{0.0, 0.1, 0.2, ..., 0.9\}$. In the case of TESVM, we also set $c_1 = c_2 = c$ and $\nu_1 = \nu_2 = \nu$ with $c \in \{2^i \mid i = -4,-3,...,4\}$ and $\nu \in \{10^{-9}, 10^{-12}\}$. The parameter σ of the RBF kernel for both methods is searched from $\{2^i \mid i = -4,-3,...,5\}$. For SVM and TWSVM, all parameters including C and σ are searched from $\{2^i \mid i = -5,-3,...,5\}$.

The first toy dataset is shown in Figure 4.4 where Figure 4.4(a)-4.4(d) show the results from TESVM, THSVM, TWSVM, and SVM, respectively. The examples in class 1 are represented by the red circles while the examples in class 2 are shown using the plus signs. In the cases of TESVM, THSVM, and TWSVM, the dash-dot lines ("-.") and dash ("--") lines represent the descriptive boundary for class 1 and 2, respectively, while for SVM the dash-dot lines indicating the margin of SVM. The solid lines illustrate the decision boundary between two classes.



(a) TESVM (b) THSVM

(c) TWSVM (d) SVM

Figure 4.4 Toy 1 dataset with linear kernel. The test accuracy on training set is (a) 92.5% (b) 80.5% (c) 92.0%. (d) 78.5%.

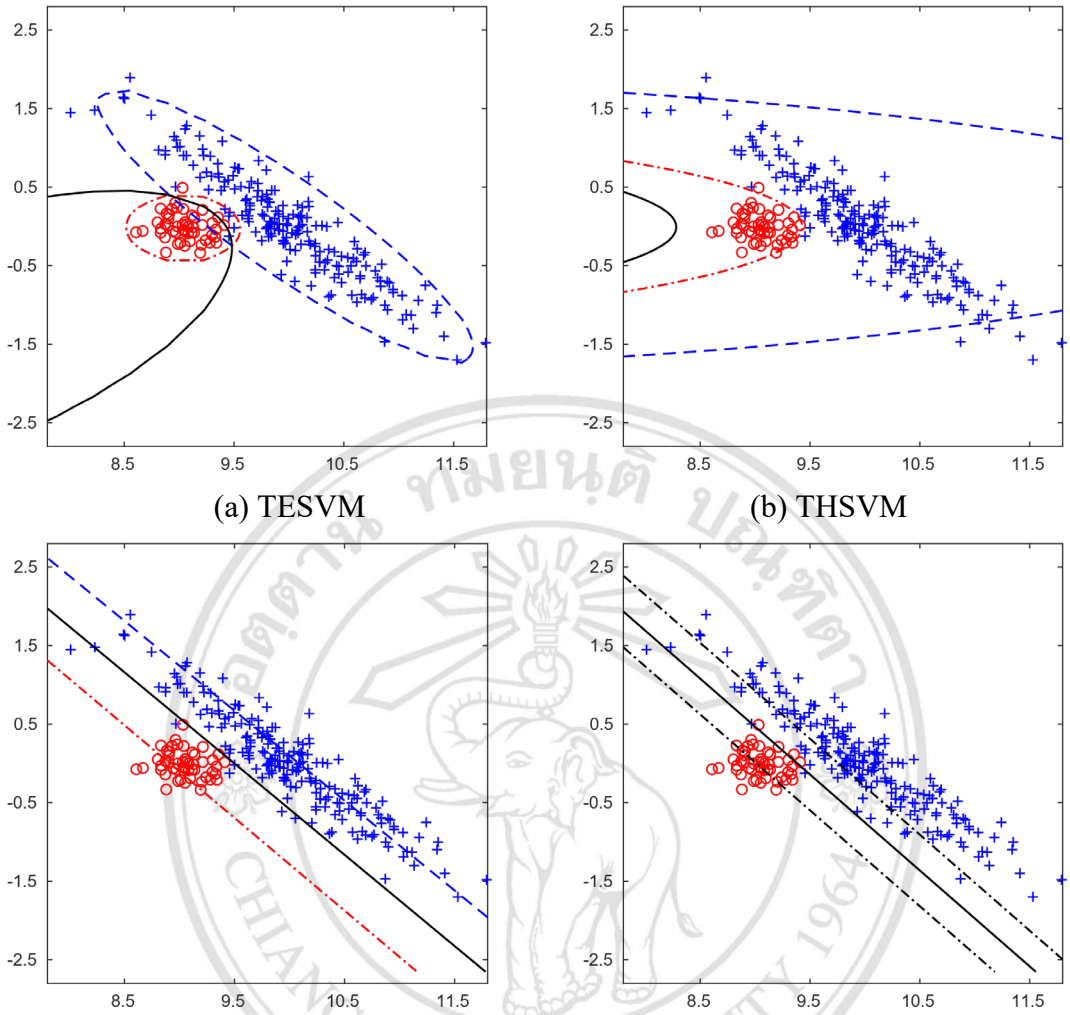
Each class in Toy 1 dataset contains 100 examples. Both classes are randomly generated from different center and rotation. Precisely, the Gaussian distributions are from $\mathsf{N}((0,0)^T, \text{diag}(1,1000^{-1/2}))$ and $\mathsf{N}((1,0)^T, \text{diag}(1,1000^{-1/2}))$ with the clockwise rotation angles at -45 and 45 degrees, respectively. The purpose of this dataset is to illustrate the circumstance when THSVM loses the spirit of TWSVM, even though it is said to be a successor. One benefit of linear TWSVM over the linear SVM is that linear TWSVM is inherently able to deal with the so-called "cross-planes" dataset [24] as seen in the comparison between Figure 4.4(c) and 4.4(d). However, THSVM has no such an ability as presented in Figure

Table 4.4 Binary classification accuracy on artificial datasets

Dataset	Kernel	TESVM	THSVM	TWSVM	SVM
Toy 1	linear	92.25 (0.42)	79.40 (0.51)	**92.40 (0.32)**	78.15 (1.08)
Toy 2	linear	**98.76 (0.35)**	80.44 (0.55)	97.28 (0.25)	97.36 (0.21)
Checkerboard	RBF	96.15 (0.37)	**96.45 (0.31)**	95.97 (0.30)	95.73 (0.56)

4.4(b). When the examples from two classes form an "×" shape, the accuracy obtained from THSVM on the training data can be as low as 50%. This is not the case for TESVM in Figure 4.4(a) as it offers less conservative class descriptors. According to Figure 4.4, TESVM significantly outperforms THSVM and SVM as can be seen from the test accuracy on the training set where TESVM, THSVM, and SVM achieve 92.5%, 80.5%, and 78.5% of accuracy, respectively. In contrast, the performance of TESVM is on par with TWSVM, i.e. 92.5% vs. 92.0%. As a remark, the decision rule in Figure 4.4 is tested on the entire training examples. Therefore, for better generalization, we provide the test accuracy from the 10-fold cross-validation in Table 4.4 where the highest accuracy is emphasized in bold showing that TESVM much better than THSVM in the case of linear classifiers.

For the second toy dataset, the examples in each class are generated from two different Gaussian distributions with unbalanced numbers of examples. The first class has 50 examples randomly drawn from $\mathsf{N}((9,0)^T, \text{diag}(1000^{1/2}, 1000^{-1/2}))$, while the second class has 200 examples drawn from $\mathsf{N}((10,0)^T, \text{diag}(1, 1000^{-1/2}))$. This dataset is specifically to demonstrate another weakness of THSVM. Despite the claim that THSVM is superior to TWSVM as the nonparallel hyperplanes of TWSVM cannot efficiently describe two classes when the examples are drawn from two distinct Gaussian distributions [19], our result shows that THSVM almost fails in this dataset as shown in Figure 4.5(b). From the figure, the accuracy on the training examples from THSVM is 80.0%. This implies that THSVM incorrectly identifies all the training examples to be entirely from the second class. On the other hand, according to Figure 4.5(a), 4.5(c), and 4.5(d), TESVM, TWSVM, and SVM achieve the accuracy of 99.2%, 97.6%, and 97.2%, respectively, showing no such a weakness as in the case of THSVM.



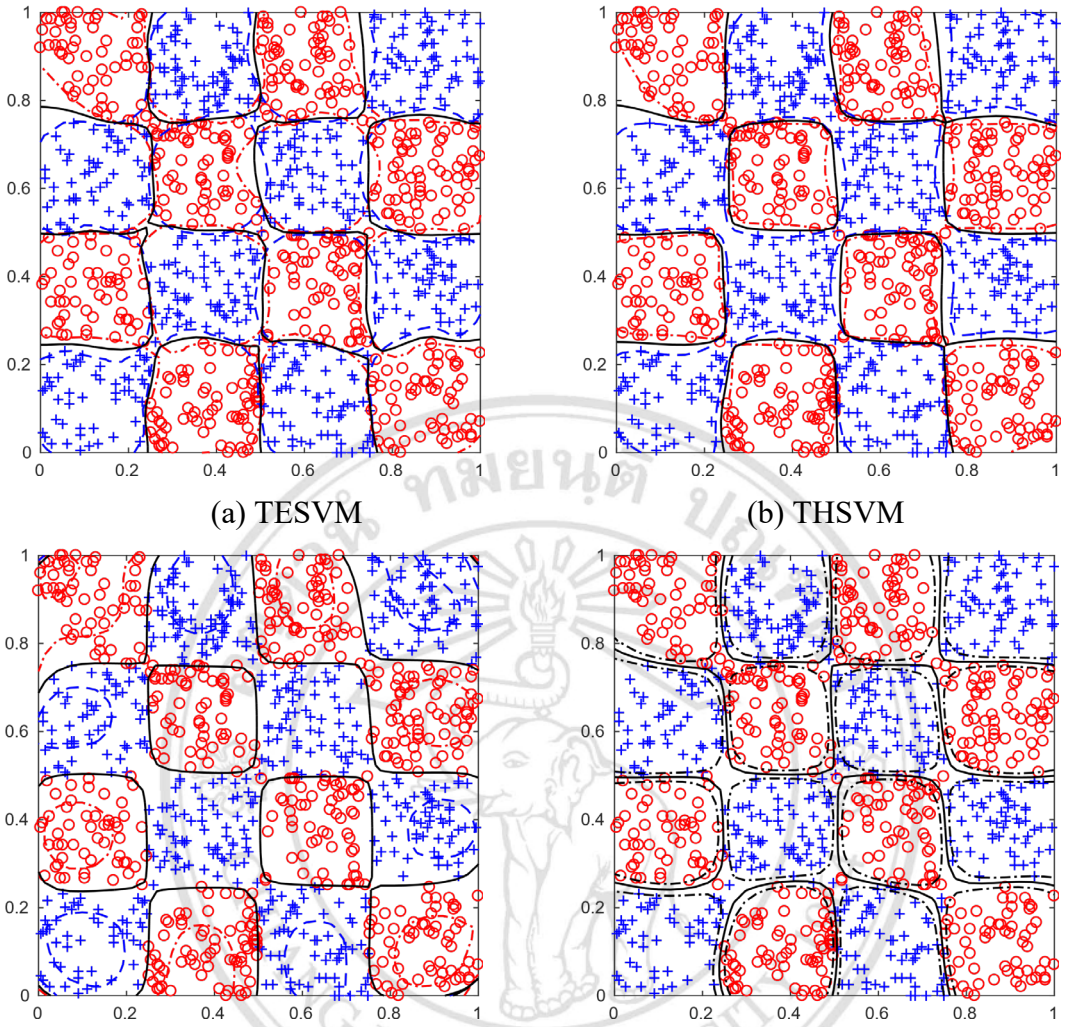
(a) TESVM (b) THSVM

(c) TWSVM (d) SVM

Figure 4.5 Toy 2 dataset with linear kernel. The test accuracy on training set is (a) 99.2% (b) 80.0% (c) 97.6% (d) 97.2%.

The accuracy from 10-fold cross-validation in Table 4.4 also further validates the result.

Finally, for the third toy dataset, the Ho-Kleinberg's checkerboard dataset is chosen to show the performance of the algorithms for a nonlinear separable case. We use the RBF kernel due to its popularity and success with real-world data. Figure 4.6(a)-4.6(d) displays the decision boundaries from THSVM, TESVM, TWSVM, and SVM. It can be observed that the decision boundary obtained from TESVM is rather complex with more curvature than from THSVM. Although it is reported from the figure that TESVM provides the better test accuracy on the training data than THSVM, i.e.,



(a) TESVM (b) THSVM

(c) TWSVM (d) SVM

Figure 4.6 Checkerboard dataset with RBF kernel. The test accuracy on training set is (a) 99.3% (b) 98.8% (c) 98.5% (d) 98.2%.

99.3% compared with 98.8%, TESVM gives slightly less accuracy from the 10-fold cross-validation, according to Table 4.4. In our view, all four methods when used with the RBF kernel are on par with one another in term of performance on this dataset. Therefore, additional datasets are required so as to further evaluate their performance.

4.2.2 Standard datasets

In this section, the standard datasets which are publicly available from the well-known UCI Machine Learning Repository [82] are used to compare the performances of THSVM, TESVM, SVM, and TWSVM. The details of the datasets are shown in Table 4.5. All the datasets contain two labels of data,

Table 4.5 Standard datasets for binary classification

Dataset	n	m	m_1	m_2
Bupa	6	345	145	200
Diabetes	8	768	500	268
Haberman	3	306	225	81
Heart	13	270	150	120
Hepatitis	19	155	32	123
Ionosphere	33	351	126	225
Iris (1)	4	150	50	100
Iris (2)	4	150	50	100
Iris (3)	4	150	50	100
Postop	8	86	62	24
Sonar	60	208	111	97
Thyroid (1)	5	215	150	65
Thyroid (2)	5	215	180	35
Thyroid (3)	5	215	185	30
Transfusion	4	748	570	178
WDBC	30	569	357	212

except for Iris and Thyroid which have three classes. Therefore, Iris and Thyroid datasets are formed two-class problems by using one-against-all strategy.

Since the aim of this thesis is on the advantage of using hyperellipsoids over hyperspheres, we first present the experimental results comparing THSVM vs. TESVM for the case of linear classifiers and nonlinear classifiers with the RBF kernel in Tables 4.6 and 4.7 respectively. Since THSVM separately finds two separate hyperspheres describing two classes, and TESVM also finds two separate hyperellipsoids, we thus consider two scenarios of hyperparameter selections. In Scenario I, the hyperparameters for solving two optimization subproblems are set to be the same, while in Scenario II, the hyperparameters can be different. As a result, the second scenario should reflex more flexibility of the decision boundary than the first scenario.

In the first scenario, the search ranges for the best hyperparameters are identical to the ranges specified in Section 4.2.1. However, for the second scenario, we set $c_1 \neq c_2$ and $\nu_1 \neq \nu_2$ for both THSVM and TESVM, where $c_1, c_2 \in \{10^i \mid i = -3, -2, ..., 3\}$ and $\nu_1, \nu_2 \in \{0.0, 0.1, 0.2, ..., 0.9\}$ for THSVM, and $c_1, c_2 \in \{2^i \mid i = -4, -3, ..., 4\}$ and $\nu_1, \nu_2 \in \{10^{-9}, 10^{-12}\}$ for TESVM. The kernel parameter σ is always set the same for one pair of optimizations and is searched from the set $\{2^i \mid i = -4, -3, ..., 5\}$. In addition, for TESVM to

Table 4.6 Classification accuracy (and its standard deviation) for linear THSVM vs. linear TESVM under two different scenarios

Dataset	Scenario I		Scenario II	
	TESVM	THSVM	TESVM	THSVM
Bupa	**67.7391 (1.2377)**	56.9565 (1.0248)	**68.6957 (1.0584)**	57.3913 (1.1674)
Diabetes	72.1875 (0.6175)	**73.1771 (1.5321)**	**75.8464 (0.7474)**	74.2708 (0.5428)
Haberman	69.0523 (1.1221)	**71.1765 (2.3606)**	**75.3268 (0.5167)**	73.0392 (0.9774)
Heart	**81.2963 (1.1077)**	79.9630 (1.6420)	**81.8519 (1.1581)**	80.4074 (2.2928)
Hepatitis	**82.9032 (1.9054)**	79.0323 (1.0201)	**82.9677 (1.8044)**	80.1935 (1.9486)
Ionosphere	**68.7464 (0.7603)**	37.7493 (0.4505)	**92.1937 (0.3344)**	87.7208 (0.6363)
Iris (1)	**100.0000 (0.0000)**	**100.0000 (0.0000)**	**100.0000 (0.0000)**	**100.0000 (0.0000)**
Iris (2)	**82.9333 (0.6441)**	66.8000 (0.4216)	**88.0667 (1.4555)**	86.6667 (1.5396)
Iris (3)	**96.6667 (0.0000)**	93.4000 (0.6630)	96.7333 (0.6630)	**96.8000 (0.8195)**
Postop	**61.3953 (2.0363)**	55.4651 (3.4689)	**69.3023 (1.9915)**	69.1860 (1.5744)
Sonar	**75.8654 (1.4476)**	69.3269 (1.2980)	**77.1635 (1.7148)**	69.5673 (1.8419)
Thyroid (1)	**83.6744 (1.0842)**	34.5116 (1.7096)	**92.1860 (0.7844)**	87.9535 (3.1120)
Thyroid (2)	98.0465 (0.1961)	**98.3721 (0.8277)**	98.4651 (0.4412)	**98.9302 (0.4927)**
Thyroid (3)	**97.6279 (0.2640)**	97.3953 (1.6142)	98.0465 (0.1961)	**98.6977 (0.7532)**
Transfusion	74.1578 (0.5199)	**75.9358 (0.7377)**	**76.5909 (0.4980)**	75.5882 (0.4725)
WDBC	**95.4482 (0.4097)**	94.2179 (0.4719)	**95.7821 (0.3416)**	94.7100 (0.8707)

Table 4.7 Classification accuracy (and its standard deviation) for THSVM vs. TESVM with RBF kernel under two different scenarios

Dataset	Scenario I		Scenario II	
	TESVM	THSVM	TESVM	THSVM
Bupa	**69.7971 (1.2876)**	69.7391 (1.1928)	**69.7971 (1.2876)**	69.7391 (1.4081)
Diabetes	76.1849 (0.5402)	**76.3411 (0.5760)**	**76.8359 (0.3157)**	76.3411 (0.5760)
Haberman	**76.1438 (0.3774)**	74.1176 (2.3906)	**76.2092 (0.6855)**	71.4706 (4.0174)
Heart	**83.9259 (0.6098)**	83.0000 (0.9147)	**83.9259 (0.6098)**	83.6667 (0.5367)
Hepatitis	**83.4839 (1.2613)**	81.6129 (2.5672)	84.7097 (1.2558)	**85.2258 (0.7100)**
Ionosphere	94.3020 (0.3290)	**95.1567 (0.6579)**	94.4444 (0.3077)	**95.1567 (0.6579)**
Iris (1)	**100.0000 (0.0000)**	**100.0000 (0.0000)**	**100.0000 (0.0000)**	**100.0000 (0.0000)**
Iris (2)	**97.4667 (0.2811)**	96.9333 (0.5622)	**97.6667 (0.6479)**	97.6000 (0.7166)
Iris (3)	**98.0667 (0.4919)**	96.7333 (0.7337)	**98.0667 (0.4919)**	96.3333 (1.0541)
Postop	**71.7442 (1.2318)**	69.1860 (1.8385)	**72.7907 (0.9806)**	70.3488 (1.4759)
Sonar	**89.6635 (1.3264)**	87.2115 (1.2861)	**89.6635 (1.3264)**	88.1731 (0.9122)
Thyroid (1)	**96.9302 (0.4493)**	96.4651 (0.9093)	97.2093 (0.5370)	**97.2558 (0.4073)**
Thyroid (2)	**99.4884 (0.1471)**	98.8372 (0.3289)	**99.4884 (0.1471)**	98.5116 (0.7532)
Thyroid (3)	**98.6512 (0.3432)**	98.5116 (0.8987)	**98.6512 (0.3432)**	98.3721 (1.0342)
Transfusion	**77.3128 (0.7047)**	76.7914 (1.0992)	**77.4332 (0.1381)**	77.3930 (0.5653)
WDBC	**97.1880 (0.1172)**	96.5026 (0.3925)	**97.1880 (0.2029)**	96.8014 (0.3868)

avoid degenerate cases when the matrix inside log-determinant objective function (3.25) has some zero eigenvalues, we also add a small diagonal matrix $\gamma\mathbf{I}$ to it where γ is a small scalar and $\mathbf{I}$ is an identity matrix of appropriate dimensions. We consider γ as another hyperparameter and search for its optimal value from the range $\{10^i \mid i = -1, -2, -3, -4\}$.

From Tables 4.6 and 4.7, the results indicate that the second scenario likely provides better classification accuracy since the decision boundary is

formed by searching a larger search space of hyperparameters. However, that is not always true, as can be seen from the case of linear THSVM with Transfusion dataset where Scenario II has lower accuracy than Scenario I. That is because the best hyperparameters are determined based only on one run of 10-fold cross validations while the results are reported from ten independent runs with randomly shuffled data.

Additionally, similar to the results in Section 4.2.1, it can be observed that linear TESVM tested with the standard datasets also performs better than linear THSVM on both scenarios as reported in Table 4.6 on most of the datasets. That is 11 and 12 datasets out of 15 datasets for Scenario I and II, respectively. While linear TESVM performs worse in some datasets, the differences are rather small. As a result, we conclude that linear TESVM provides better prediction results than linear THSVM. For further insights into Table 4.6, we also summarize the ranking of each algorithm for each dataset in Appendix B's Table B.2 where linear TESVM with Scenario II has the best average rank as well as the best average accuracy.

The classification accuracy from both THSVM and TESVM with the RBF kernel on the standard datasets also further shows the advantage of using hyperellipsoids over hyperspheres. According to Table 4.7, we observe that TESVM provides better accuracy on 13 and 12 datasets out of 15 datasets on the first and the second scenarios, respectively. Although the accuracies are not drastically improved from Scenario I to Scenario II and also from THSVM to TESVM, slight enhancements in the accuracies in many datasets are still a good indicator to show that the less conservative class descriptors of hyperellipsoids can help squeeze the performance from the less flexible spherical shape of THSVM, even together with the RBF kernel. We also summarize Table 4.7 by ranking them for each dataset as shown in Appendix B's Table B.3. According to the table, TESVM with RBF kernel under Scenario II has the best average rank as well as the best average accuracy.

Table 4.8 Classification accuracy (and its standard deviation) for linear binary classifiers

Dataset	TESVM	THSVM	TWSVM	SVM
Bupa	68.6957 (1.0584)	57.3913 (1.1674)	66.6667 (0.6111)	**69.1014 (0.8334)**
Diabetes	75.8464 (0.7474)	74.2708 (0.5428)	**76.9401 (0.3806)**	76.9271 (0.3178)
Haberman	75.3268 (0.5167)	73.0392 (0.9774)	**75.4575 (0.5856)**	73.5294 (0.0000)
Heart	81.8519 (1.1581)	80.4074 (2.2928)	83.8519 (0.5843)	**84.1481 (0.5179)**
Hepatitis	**82.9677 (1.8044)**	80.1935 (1.9486)	80.1290 (2.3122)	79.4839 (1.5146)
Ionosphere	**92.1937 (0.3344)**	87.7208 (0.6363)	82.0798 (0.7288)	87.5499 (0.8709)
Iris (1)	**100.0000 (0.0000)**	**100.0000 (0.0000)**	**100.0000 (0.0000)**	**100.0000 (0.0000)**
Iris (2)	**88.0667 (1.4555)**	86.6667 (1.5396)	74.8667 (1.3717)	72.0000 (1.4055)
Iris (3)	96.7333 (0.6630)	**96.8000 (0.8195)**	94.8000 (0.8777)	95.8000 (0.6325)
Postop	69.3023 (1.9915)	69.1860 (1.5744)	70.5814 (1.2318)	**72.0930 (0.0000)**
Sonar	77.1635 (1.7148)	69.5673 (1.8419)	75.5288 (1.9945)	**78.2692 (2.0623)**
Thyroid (1)	**92.1860 (0.7844)**	87.9535 (3.1120)	81.3488 (0.4625)	86.6512 (0.4412)
Thyroid (2)	98.4651 (0.4412)	**98.9302 (0.4927)**	92.2791 (0.8262)	98.6047 (0.0000)
Thyroid (3)	98.0465 (0.1961)	**98.6977 (0.7532)**	97.3488 (0.2247)	98.0465 (0.4274)
Transfusion	76.5909 (0.4980)	75.9358 (0.7377)	**77.9278 (0.6253)**	76.2032 (0.0000)
WDBC	95.7821 (0.3416)	94.7100 (0.8707)	94.6749 (0.3214)	**97.6977 (0.1934)**
Average	**85.5762**	83.2169	82.7801	84.1315

Table 4.9 Classification accuracy (and its standard deviation) for classifiers with RBF kernel

Dataset	TESVM	THSVM	TWSVM	SVM
Bupa	69.7971 (1.2876)	69.7391 (1.4081)	**73.2174 (0.5829)**	72.6087 (0.8560)
Diabetes	76.8359 (0.3157)	76.3411 (0.5760)	77.0964 (0.7751)	**77.3958 (0.3373)**
Haberman	**76.2092 (0.6855)**	74.1176 (2.3906)	73.2026 (1.4035)	74.2484 (0.7193)
Heart	83.9259 (0.6098)	83.6667 (0.5367)	83.7778 (0.6246)	**84.3704 (0.4553)**
Hepatitis	84.7097 (1.2558)	**85.2258 (0.7100)**	84.3871 (1.5745)	83.4839 (1.9521)
Ionosphere	94.4444 (0.3077)	95.1567 (0.6579)	93.9031 (0.8191)	**95.2137 (0.3984)**
Iris (1)	**100.0000 (0.0000)**	**100.0000 (0.0000)**	**100.0000 (0.0000)**	**100.0000 (0.0000)**
Iris (2)	**97.6667 (0.6479)**	97.6000 (0.7166)	97.2667 (0.5837)	97.2667 (0.2108)
Iris (3)	**98.0667 (0.4919)**	96.7333 (0.7337)	97.2000 (0.5259)	95.8667 (0.5259)
Postop	**72.7907 (0.9806)**	70.3488 (1.4759)	69.0698 (3.2980)	72.6744 (1.2560)
Sonar	89.6635 (1.3264)	88.1731 (0.9122)	88.7981 (1.2215)	**89.7596 (1.2831)**
Thyroid (1)	97.2093 (0.5370)	**97.2558 (0.4073)**	95.1628 (0.9607)	96.0465 (0.5481)
Thyroid (2)	**99.4884 (0.1471)**	98.8372 (0.3289)	**99.4884 (0.1471)**	98.9767 (0.1961)
Thyroid (3)	**98.6512 (0.3432)**	98.5116 (0.8987)	97.8605 (0.2402)	97.8140 (0.6951)
Transfusion	77.4332 (0.1381)	77.3930 (0.5653)	**79.2112 (0.4332)**	78.9706 (0.6335)
WDBC	97.1880 (0.2029)	96.8014 (0.3868)	**98.0492 (0.1934)**	97.8383 (0.1862)
Average	**88.3800**	87.8688	87.9807	88.2834

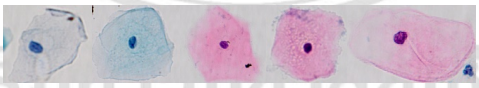
Moreover, we further compare both THSVM and TESVM with SVM and TWSVM. The setups of SVM and TWSVM are the same as described in Section 4.2.1. The comparisons of all four classifiers are provided in Table 4.8 and 4.9 for the linear and RBF kernels, respectively. The accuracy values from THSVM and TESVM in Table 4.8 are extracted from the best scenario in Table 4.6 for each dataset. Likewise, the results from THSVM and TESVM in Table 4.9 are also obtained from Table 4.7. We observe that

TESVM and SVM provide better results than THSVM and TWSVM in term of the number of datasets for both linear and RBF kernels. For the average accuracy across all 16 standard datasets, TESVM is superior to other methods. Furthermore, we further summarize Table 4.8 and 4.9 by reporting the corresponding ranks as presented in Appendix B's Table B.4 and B.5. From the tables, TESVM achieves the best ranks as well as the best average accuracies. In fact, these ranks of TESVM are considered close to SVM's.

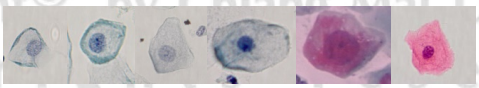
One drawback of TESVM that we can observe during the experiments is the proper choices of (ν_1, ν_2). According to (3.25), the term $\mathbf{XAX}^T - \mathbf{X}\boldsymbol{\alpha}\boldsymbol{\alpha}^T\mathbf{X}^T$ is equivalent to $\sum_{i\in \mathrm{I}_1} \alpha_i \mathbf{x}_i \mathbf{x}_i^T - \frac{\nu_1}{m_2} \sum_{j\in \mathrm{I}_2} \mathbf{x}_j \mathbf{x}_j^T$ .which implies that if ν_1 is specified too large, the overall term will become an indefinite matrix. Thus, the optimization cannot be solved. As a result, the values of ν_1 and ν_2 in this paper are kept being rather small to avoid such a pitfall and the proper selection method is subjected to further research.

4.2.3 Private dataset

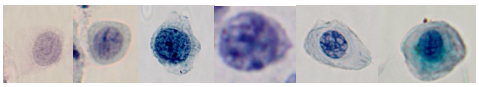
Although standard datasets are widely used to gauge the performance of a classification algorithm, we are also interested in applying TESVM to other datasets whose data collection and feature extraction processes are



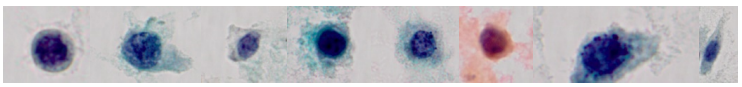
(a) Class 1: Normal and ASC-US cells



(b) Class 2: LSIL cells



(c) Class 3: HSIL cells



(d) Class 4: SCC cells

Figure 4.7 Examples of cervical cancer cell (Courtesy of T. Chankong)

Table 4.10 LCH pap smear dataset

Class	1	2	3	4	Total
No. of examples	150	64	38	48	300
No. of features	9				

Table 4.11 Classification accuracy (and its standard deviation) for classifiers with RBF kernel on LCH dataset

Dataset	TESVM	THSVM	TWSVM	SVM
1 vs the rest	89.3333 (0.5212)	85.8667 (1.1353)	**92.3333 (0.8607)**	91.3333 (1.1547)
2 vs the rest	88.9333 (0.7503)	86.0667 (0.6441)	**93.5667 (0.2250)**	92.8667 (0.3220)
3 vs the rest	**89.7000 (0.3668)**	86.6333 (1.9717)	88.6333 (0.7445)	87.5000 (0.5270)
4 vs the rest	89.5667 (0.8020)	89.3000 (1.0476)	**89.8333 (0.5932)**	89.7667 (0.8614)
One-against-one	77.4667 (1.1675)	71.3333 (1.0887)	**81.2000 (0.8043)**	79.2000 (0.8777)

transparent to us. In this section, we deliberately select the proprietary dataset known as the "LCH pap smear dataset" by Chankong et al. [83].

The LCH data named after Lampang Cancer Hospital (LCH) who was also the source of data, consists of 300 cervical cancer cell images categorized into 4 classes by expert as shown in Figure 4.7. The process of data collection was also approved by the Lampang Cancer Hospital Ethics Committee. The number of data in each class is summarized in Table 4.10 and the details of each class of cells are as follows. Class 1 contains 150 cells where 80 of them are normal cells and 70 are atypical squamous cells of undetermined significance (ASC-US). Class 2 and 3 consist of 64 low-grade squamous intraepithelial lesion (LSIL) and 38 high-grade squamous intraepithelial lesion (HSIL), respectively. Finally, Class 4 consists of squamous cell carcinoma (SCC) cells. Among 4 classes, only Class 1 is considered normal, while the rest is abnormal classified into different degrees of abnormalities. The raw images of the cells were undergone a feature extraction process resulted in totally 9 features. For more details on the dataset and its extraction process, see the reference [83].

The experimental setup for training and testing the LCH dataset is the same as in Section 4.2.2. Since the dataset has four classes, we form four binary classification scenarios based on the one-against-all fashion as shown in Table 4.11. The multiclass classification with the one-against-one strategy is also given. Some of the results for SVM, TWSVM, and THSVM presented in the table were previously reported in [84]. According to the table,

TWSVM exhibits the best performance among the others. Our proposed TESVM, however, does outperform THSVM for all cases.

4.2.4 TESVM with instance reduction

SVM and TWSVM solve one or two QPPs to find the best hyperplanes from the given training data to create a decision rule. Likewise, SVDD and THSVM also require solving QPPs to create hyperspheres. For TESVM, since its formulation is based on the semidefinite program of MVCE, it is more computational expensive due to a larger number of unknown variables to be solved.

Table 4.12 shows some selected datasets with at least 1000 examples where n_c, n, and m are the number of classes, features, and examples, respectively. Among these datasets, we conduct some experiments similar to the experiments in Section 4.2.2 (Scenario II). However, as shown in Table 4.13, where the classification accuracy (%) and the training time (seconds) are

Table 4.12 Large datasets with at least 1000 examples

Dataset	n_c	n	m
Adult	2	14	32561
Banana	2	2	5300
Checkerboard	2	2	1000
German	2	20	1000
HTRU2	2	8	17898
Occupancy	2	5	20560
Pen-based	10	16	7494
Page-blocks0	2	10	5472
Phoneme	2	5	5404
Shuttle	7	9	43500
Skin	2	3	245057
Twonorm	2	20	7400
Wilt	2	5	4339

Table 4.13 Classification accuracy and training time (in seconds) and their standard deviations for some selected datasets from Table 4.12

Dataset	TESVM	THSVM	TWSVM	SVM
Checkerboard	96.15 (0.37) *114.1 (123.4)*	96.45 (0.31) *0.293 (0.110)*	95.97 (0.30) *0.245 (0.045)*	95.73 (0.56) *0.050 (0.007)*
HTRU2	Out of memory	N/A	N/A	98.04 (0.02) *9.666 (1.301)*
Occupancy detection	Out of memory	N/A	N/A	98.99 (0.01) *3.437 (0.289)*
Shuttle	Out of memory	N/A	N/A	99.90 (0.01) *5.098 (0.490)*
Skin segmentation	Out of memory	N/A	N/A	N/A

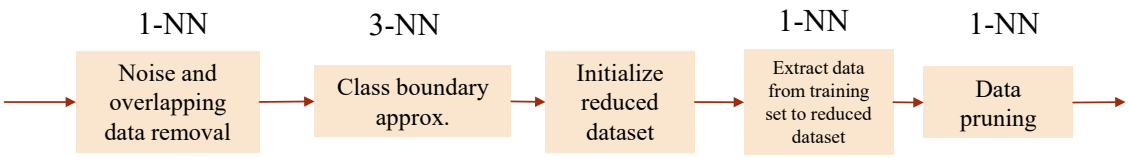


Figure 4.8 Instance reduction procedure

shown in normal and italic texts, respectively, TESVM fails to be trained on many datasets except for the checkerboard dataset which has only 2 classes, 2 features, and 1000 examples. Hence, this problem reflects a major drawback of TESVM. In fact, any MVCE-based classifiers attempting at solving an SDP problem using a generic SDP solver are unable to be trained under large datasets.

There are many possibilities to enable TESVM to handle larger data. For example, active set strategies are proposed in [41] as a technique to solve MVCE problem with more than 10000 examples. Wang and Xiao [63] also suggest a procedure based on Mahalanobis distance in the kernel space to select a subset of data which contains possible supports for the hyperellipsoid.

In this thesis, however, we attempt to solve the problem from another perspective. We introduce an instance selection technique as proposed by A. Yodjaiphet [85] to be a preprocessing step before solving the MVCE-based classifiers. Although many instance reduction methods have been presented in literature, the technique proposed in [85] offers exceptional reduction speed while retaining the significant portion of data. The rough idea of the instance selection procedure is that, among all training examples, the most important part of the data is only located at the boundary where two different classes face against or overlap each other. Therefore, it is intuitively plausible to ignore the examples located far from the boundary. The implementation of this idea can be achieved entirely based on comparing the in-class and between-class distances of various subsets of data. In fact, most parts of the algorithm rely merely on *k*-nearest neighbors where the reduction procedure can be described as in Figure 4.8. For the exacted reduction algorithm, see [85]. The illustrations of the data reduction

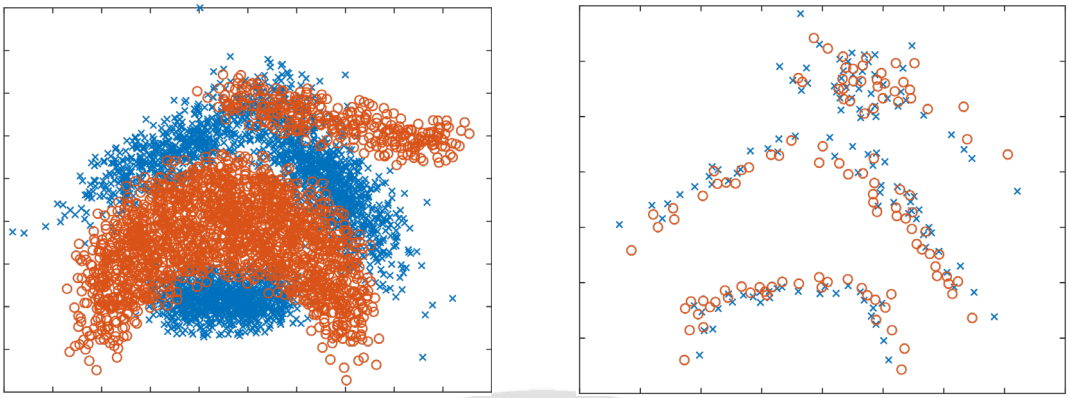


Figure 4.9 Banana dataset and its reduced version

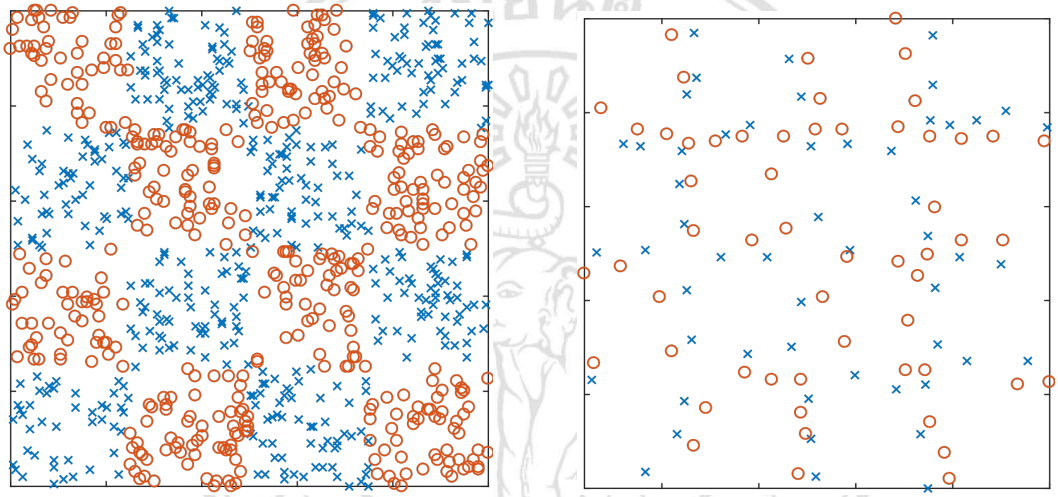


Figure 4.10 Ho-Kleinberg's checkerboard dataset and its reduced version

results are shown in Figure 4.9 and 4.10 for Banana and Ho-Kleinberg's checkerboard datasets, respectively. According to the figures, the reduced data are mainly extracted from the boundary of the original data.

The goodness of the instance reduction technique will allow the proposed TESVM classifier to be trained on larger datasets and be solved using a generic SDP solver. Since the complexity of the introduced reduction scheme is much lower than solving an SDP problem, incorporating the reduction step into TESVM does not add much time complexity. Although the instance reduction does sacrifice some classification accuracy, Table 4.14 shows show that the accuracy loss is acceptable in the case of SVM.

To demonstrate the performance of TESVM with data reduction, 10 independent runs of 10-fold cross validation are performed on the reduced version of datasets listed in Table 4.12. Because 10-fold cross validation

Table 4.14 Classification accuracy (with its standard deviation) of SVM: With vs. without instance reduction

Dataset	SVM (without instance reduction)	SVM (with instance reduction)	Accuracy loss (%)
Checkerboard	95.2600 (0.3239)	93.5700 (0.5438)	-1.77
Banana	90.6038 (0.0755)	90.3717 (0.1543)	-0.26
HTRU2	98.0417 (0.0192)	97.9009 (0.0201)	-0.14
Occupancy detection	98.9942 (0.0050)	98.7359 (0.0360)	-0.26

Table 4.15 The average number of examples (and its standard deviation) of 9 training folds in 10-fold cross validation

Dataset	Original training folds	Reduced training folds
Adult	29304.9 (0.3)	174.32 (10.79)
Banana	4770.0 (0.0)	236.61 (7.53)
Checkerboard	900.0 (0.0)	109.39 (2.60)
German	900.0 (0.0)	129.54 (5.25)
HTRU2	16108.2 (0.4)	168.91 (7.98)
Occupancy	18504.0 (0.0)	141.72 (6.29)
Pen-based (1)	6744.6 (0.5)	37.28 (1.66)
Pen-based (2)	6744.6 (0.5)	132.47 (4.04)
Pen-based (3)	6744.6 (0.5)	90.25 (3.08)
Pen-based (4)	6744.6 (0.5)	73.50 (3.17)
Pen-based (5)	6744.6 (0.5)	66.35 (2.98)
Pen-based (6)	6744.6 (0.5)	80.93 (2.78)
Pen-based (7)	6744.6 (0.5)	38.19 (1.66)
Pen-based (8)	6744.6 (0.5)	87.46 (3.47)
Pen-based (9)	6744.6 (0.5)	63.50 (2.43)
Pen-based (10)	6744.6 (0.5)	77.95 (2.64)
Page-blocks0	4924.8 (0.4)	143.71 (5.24)
Phoneme	4863.6 (0.5)	36.84 (4.54)
Shuttle (1)	39150.0 (0.0)	71.48 (2.57)
Shuttle (4)	39150.0 (0.0)	53.98 (1.55)
Shuttle (5)	39150.0 (0.0)	13.62 (1.12)
Skin	220551.3 (0.5)	170.10 (3.76)
Twonorm	6660.0 (0.0)	59.81 (5.67)
Wilt	3905.1 (0.3)	69.22 (4.65)

constructs 10 models from different nine folds of data, we report the average sizes as well as their standard deviations of 9-fold training data for both the original and reduced datasets in Table 4.15. It is important to note that, although the reduced 9 folds are used for training, the resulted models are tested using the normal testing fold which does not undergo data reduction. Moreover, the datasets whose name is followed by a parenthesized number such as “Pen-based (3)” are two-class datasets created by using one-against-all fashion. That is “Pen-based (3)” is the dataset derived from the third class against the rest from the 10-classes Pen-based dataset.

Similar to the experimental setup in Scenario II of Section 4.2.2, each hyperellipsoid in TESVM are independently adjusted their hyperparameters. This setting is also the case for each hypersphere of THSVM. In TWSVM, however, the hyperparameters for each hyperplane are set to be identical. In order to find the best hyperparameters, we perform a grid search over given ranges of parameters where the best parameters are the ones which provides the best 10-fold cross-validation accuracy.

The percentages of classification accuracy from four classifiers with RBF kernel, i.e., TESVM, THSVM, TWSVM, and SVM, on the reduced datasets are shown in Table 4.16. For each dataset, the average classification accuracy (%) as well as its standard deviation is shown using the normal text, while the average training time (in seconds) is shown in italics. In order to interpret the result, we also provide Table B.6 in Appendix B to summarize the rank of each classifier on each dataset as well as the average rank and average accuracy. According to those tables, when taking only TESVM and THSVM into account, TESVM performed better than THSVM for 18 out of 24 datasets. In addition, among four classifiers, TESVM achieved the second place for the average rank with 9 datasets being the best among the others. Although SVM is the best in terms of the average rank, average accuracy, and attaining the best accuracy for 12 datasets, TESVM can still find a niche in some datasets, better than both THSVM and TWSVM which are from the same family of twin support vector classifiers. In fact, upon a closer comparison between TESVM and SVM, we observe that TESVM is better than SVM for 10 out of 24 datasets.

Moreover, as further references, classification accuracy and training time (in seconds) are also reported for the four classifiers with RBF kernel on smaller datasets as shown in Table 4.17 with their ranking summarized in Table B.7 in Appendix B. According to the tables, among 12 datasets, TESVM obtained the best accuracy for 3 out of 12 datasets, while SVM achieved totally 8 out of 12 datasets. This suggests that the strength of SVM in that it can provide good prediction accuracy, especially in the datasets with smaller number of examples. However, similar to the results from

Table 4.16 Classification accuracy and training time (in seconds) and their standard deviations on the large datasets with instance reduction

Dataset	TESVM	THSVM	TWSVM	SVM
Adult	80.7411 (0.1949) *6.8040 (0.9743)*	81.5703 (0.2944) *0.0137 (0.0010)*	82.7481 (0.1507) *0.0103 (0.0020)*	81.5279 (0.2359) *0.0095 (0.0010)*
Banana	90.3453 (0.1571) *5.3456 (0.7702)*	88.1321 (0.2268) *0.0190 (0.0016)*	89.4642 (0.9107) *0.0132 (0.0014)*	90.3717 (0.1543) *0.0121 (0.0012)*
Checkerboard	93.6000 (0.4372) *0.9411 (0.1884)*	94.9400 (0.5147) *0.0103 (0.0009)*	91.4500 (0.9301) *0.0071 (0.0005)*	93.5700 (0.5438) *0.0059 (0.0014)*
German	71.3300 (0.6993) *7.2037 (1.1181)*	70.8500 (1.0638) *0.0106 (0.0009)*	74.3200 (0.5940) *0.0085 (0.0021)*	73.6000 (0.8602) *0.0079 (0.0025)*
HTRU2	97.9070 (0.0487) *2.8546 (0.3729)*	97.8333 (0.0609) *0.0126 (0.0003)*	97.8634 (0.0658) *0.0085 (0.0009)*	97.9009 (0.0201) *0.0054 (0.0007)*
Occupancy	98.8205 (0.1635) *7.9758 (1.5885)*	98.0034 (0.3991) *0.0090 (0.0006)*	98.6629 (0.1167) *0.0078 (0.0008)*	98.7359 (0.0360) *0.0068 (0.0010)*
Pen-based (1)	99.7972 (0.0320) *1.1629 (0.1794)*	99.7732 (0.0218) *0.0066 (0.0004)*	99.7251 (0.0362) *0.0060 (0.0006)*	99.7451 (0.0172) *0.0012 (0.0001)*
Pen-based (2)	99.3355 (0.0402) *6.8744 (1.0637)*	99.2060 (0.0675) *0.0115 (0.0006)*	98.7950 (0.1603) *0.0080 (0.0006)*	99.5850 (0.0528) *0.0068 (0.0005)*
Pen-based (3)	99.6410 (0.0473) *1.4653 (0.2253)*	99.5103 (0.0345) *0.0079 (0.0005)*	99.4876 (0.0651) *0.0067 (0.0007)*	99.8225 (0.0289) *0.0030 (0.0004)*
Pen-based (4)	99.6771 (0.0407) *0.8786 (0.1729)*	99.6130 (0.0308) *0.0078 (0.0008)*	99.4489 (0.0545) *0.0063 (0.0005)*	99.7585 (0.0364) *0.0021 (0.0002)*
Pen-based (5)	99.8706 (0.0282) *0.7942 (0.1496)*	99.6744 (0.0454) *0.0076 (0.0009)*	99.6851 (0.0804) *0.0061 (0.0005)*	99.8612 (0.0268) *0.0021 (0.0002)*
Pen-based (6)	99.7838 (0.0453) *1.2034 (0.1762)*	99.7638 (0.0252) *0.0082 (0.0006)*	98.8671 (0.1697) *0.0065 (0.0006)*	99.7758 (0.0338) *0.0027 (0.0004)*
Pen-based (7)	99.7825 (0.0907) *0.4321 (0.1149)*	99.8479 (0.0432) *0.0064 (0.0003)*	99.7638 (0.0413) *0.0058 (0.0005)*	99.8812 (0.0336) *0.0012 (0.0002)*
Pen-based (8)	99.5957 (0.0383) *1.2566 (0.1981)*	99.5250 (0.0766) *0.0084 (0.0005)*	98.9685 (0.1428) *0.0064 (0.0005)*	99.7758 (0.0551) *0.0033 (0.0003)*
Pen-based (9)	99.7251 (0.0328) *0.7202 (0.1331)*	99.6878 (0.0229) *0.0078 (0.0006)*	99.5036 (0.1149) *0.0061 (0.0005)*	99.7758 (0.0343) *0.0022 (0.0003)*
Pen-based (10)	99.6531 (0.0680) *1.0655 (0.1684)*	99.6437 (0.0683) *0.0084 (0.0016)*	99.2167 (0.3654) *0.0065 (0.0006)*	99.8305 (0.0295) *0.0030 (0.0003)*
Page-blocks0	96.3359 (0.1002) *7.6180 (1.6104)*	92.7814 (0.6407) *0.0099 (0.0008)*	95.6232 (0.1596) *0.0081 (0.0008)*	95.8553 (0.2031) *0.0080 (0.0012)*
Phoneme	74.9112 (0.9248) *0.3806 (0.0327)*	74.7150 (2.5087) *0.0065 (0.0007)*	69.3949 (2.5887) *0.0064 (0.0011)*	73.0477 (2.0270) *0.0009 (0.0001)*
Shuttle (1)	99.7175 (0.0758) *0.8790 (0.0833)*	95.4816 (0.6618) *0.0067 (0.0004)*	97.9097 (0.4675) *0.0058 (0.0006)*	99.7149 (0.0227) *0.0046 (0.0008)*
Shuttle (4)	99.8246 (0.0583) *0.7044 (0.1214)*	99.8531 (0.0362) *0.0064 (0.0006)*	95.9285 (0.2519) *0.0061 (0.0010)*	99.9559 (0.0043) *0.0016 (0.0004)*
Shuttle (5)	99.9269 (0.0059) *1.5511 (0.2659)*	99.8947 (0.0579) *0.0063 (0.0004)*	99.4186 (0.3942) *0.0060 (0.0003)*	99.9434 (0.0057) *0.0011 (0.0003)*
Skin	99.9045 (0.0303) *4.8346 (0.6326)*	98.2431 (0.0533) *0.0112 (0.0005)*	99.6005 (0.4138) *0.0103 (0.0027)*	99.5800 (0.0739) *0.0061 (0.0005)*
Twonorm	94.6000 (0.4659) *0.6062 (0.1164)*	96.5838 (0.1354) *0.0073 (0.0012)*	95.1203 (0.6232) *0.0058 (0.0006)*	96.6419 (0.2164) *0.0023 (0.0005)*
Wilt	99.2279 (0.0714) *0.6909 (0.0580)*	99.4031 (0.0619) *0.0079 (0.0006)*	99.0873 (0.2373) *0.0072 (0.0016)*	99.4330 (0.0607) *0.0021 (0.0003)*

Table 4.16 for larger datasets, TESVM still ranks at the second place over THSVM and TWSVM. In fact, among 12 datasets, there is no dataset where THSVM performed better than TESVM.

Table 4.17 Classification accuracy and training time (in seconds) and their standard deviations on the small datasets with instance reduction

Dataset	TESVM	THSVM	TWSVM	SVM
Bupa	68.3768 (1.7891) *0.6809 (0.1119)*	64.6667 (1.2923) *0.0100 (0.0039)*	67.9130 (1.6569) *0.0066 (0.0022)*	68.8406 (1.4732) *0.0011 (0.0005)*
Diabetes	74.8047 (0.9334) *1.0374 (0.2619)*	72.4219 (0.8784) *0.0097 (0.0026)*	76.0938 (0.4034) *0.0070 (0.0017)*	76.6406 (0.6205) *0.0022 (0.0006)*
Haberman	74.4771 (1.8322) *1.2488 (0.3378)*	70.9150 (1.7016) *0.0084 (0.0034)*	73.2026 (0.9243) *0.0079 (0.0038)*	73.4641 (0.6680) *0.0009 (0.0015)*
Heart	80.3704 (1.6002) *1.3773 (0.3822)*	78.1852 (1.1509) *0.0075 (0.0021)*	83.5185 (1.2128) *0.0060 (0.0008)*	83.6667 (1.4125) *0.0006 (0.0001)*
Hepatitis	81.0968 (1.6108) *1.5119 (0.4153)*	80.4516 (1.2558) *0.0071 (0.0012)*	81.2258 (1.2705) *0.0060 (0.0007)*	78.9677 (1.8299) *0.0004 (0.0000)*
Ionosphere	68.3191 (2.8861) *1.5099 (0.4049)*	40.2279 (0.9478) *0.0071 (0.0012)*	60.6553 (2.0826) *0.0060 (0.0011)*	87.3504 (1.2896) *0.0006 (0.0001)*
Sonar	83.9904 (1.5710) *1.7955 (0.4553)*	75.7212 (2.4956) *0.0075 (0.0011)*	78.9904 (1.7268) *0.0063 (0.0017)*	82.6442 (2.4246) *0.0013 (0.0007)*
Thyroid (1)	95.2093 (1.0296) *2.1585 (0.5578)*	82.8372 (1.2099) *0.0065 (0.0011)*	93.4419 (1.1695) *0.0058 (0.0010)*	95.6744 (0.8219) *0.0003 (0.0001)*
Thyroid (2)	97.6744 (0.5801) *2.1455 (0.5524)*	93.0233 (1.0963) *0.0065 (0.0010)*	92.7442 (1.3551) *0.0057 (0.0008)*	98.3721 (0.3953) *0.0002 (0.0001)*
Thyroid (3)	95.8140 (1.9364) *2.2227 (0.5513)*	91.3953 (1.6877) *0.0062 (0.0010)*	94.5116 (1.8187) *0.0065 (0.0043)*	94.1860 (4.1673) *0.0003 (0.0001)*
Transfusion	73.8235 (2.6914) *2.0786 (0.5180)*	73.3021 (0.8121) *0.0084 (0.0011)*	72.2326 (4.0969) *0.0064 (0.0010)*	77.1123 (0.5151) *0.0011 (0.0003)*
WDBC	95.7645 (0.6804) *2.0930 (0.5358)*	84.0598 (2.4212) *0.0071 (0.0013)*	96.4851 (0.5039) *0.0058 (0.0008)*	96.6784 (0.7792) *0.0007 (0.0002)*

In conclusion, this section shows that it is impossible to evaluate TESVM against its peers on the datasets larger than 1000 examples without the instance reduction. With the introduction of the data reduction technique, we can conclude that TESVM is superior to THSVM and TWSVM. Although, for smaller datasets with instance reduction, TESVM seems weaker than SVM in term of the number of the best datasets, such weakness is lessened in the large datasets.