2017

SV-Means: A Fast One-Class Support Vector Machine-Based Level Set Estimator

Anne M. Pavy
Wright State University

SV-MEANS: A FAST ONE-CLASS SUPPORT VECTOR MACHINE-BASED LEVEL SET ESTIMATOR

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy

by

ANNE M. PAVY
B.S.E.E., Wright State University, 2007
M.S.E.G., Wright State University, 2009

2017
Wright State University

WRIGHT STATE UNIVERSITY

GRADUATE SCHOOL

December 12, 2017

I HEREBY RECOMMEND THAT THE DISSERTATION PREPARED UNDER MY SUPERVISION BY Anne M. Pavy ENTITLED SV-Means: A Fast One-Class Support Vector Machine-Based Level Set Estimator BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Doctor of Philosophy

Brian D. Rigling, Ph.D.
Dissertation Director

Arnab K. Shaw, Ph.D.
Director, Electrical Engineering Ph.D. Program

Barry Milligan, Ph.D.
Interim Dean of the Graduate School

Committee on
Final Examination

Brian D. Rigling, Ph.D.

Fred D. Garber, Ph.D.

Kefu Xue, Ph.D.

Michael Bryant, Ph.D.

Randolph L. Moses, Ph.D.

ABSTRACT

Pavy, Anne M., Ph.D., Electrical Engineering Ph.D. Program, Department of Electrical Engineering, Wright State University, 2017. *SV-Means: A Fast One-Class Support Vector Machine-Based Level Set Estimator.*

In this dissertation, a novel algorithm, SV-Means, is developed motivated by the many functions needed to perform radar waveform classification in an evolving, contested environment. Important functions include the ability to: reject classes not in the library, provide confidence in the classification decision, adapt the decision boundary on-the-fly, discover new classes, and quickly add new classes to the library. The SV-Means approach addresses these functions by providing a fast algorithm that can be used for anomaly detection, density estimation, open set classification, and clustering, within a Bayesian generative framework. The SV-Means algorithm extends the quantile one-class support vector machine (q-OCSVM) density estimation algorithm into a classification formulation with inspiration from k-means and stochastic gradient descent principles. In addition, the algorithm can be trained at least an order of magnitude faster than the q-OCSVM and other OCSVM algorithms. SV-Means has been thoroughly tested with a phase-modulated radar waveform data set, and several data sets from the University of California Irvine (UCI) machine learning repository, in each application area except clustering. In clustering, a novel algorithm, SV-Means Level Set Clustering, was formulated using the SV-Means algorithm as a first step to determine the number of clusters per level set and distinguish overlapping clusters. Finally, an end-to-end demonstration from training, to testing, to clustering, to adding a new class to the library, was demonstrated using the SV-Means algorithm.

Abbreviations and Symbols

List of Abbreviations

2-D	2-Dimensional
AD	Anomaly Detection
ACF	Autocorrelation Function
AUC	Area Under Curve (area under ROC curve)
BW	Bandwidth
C	Classification
CwR	Classification with Rejection
DE	Density Estimation
EVT	Extreme Value Theory
FN	False Negative
FP	False Positive
GD	Gradient Descent
GSLS	Golden Section Line Search
I-OCSVM	Independent-OneClass Support Vector Machine
ID	Identification
ISOMAP	Isometric Mapping
KKT	Karush-Kuhn-Tucker
LS	Least Squares
MST	Minimum Spanning Tree
OCSVM	One-Class Support Vector Machine
OSR	Open Set Recognition
PCA	Principle Component Analysis
PI-SVM	Probability of Inclusion Support Vector Machine
POS-SVM	Probabilistic Open Space Support Vector Machine
q-OCSVM	Quantile One Class Support Vector Machine
QP	Quadratic Programming
RF	Random Features
ROC	Receiver Operating Characteristic
SGD	Stochastic Gradient Descent
SMO	Sequential Minimal Optimization
SNR	Signal-to-Noise Ratio
SVC	Support Vector Clustering
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
UB	Upper Bound
W-SVM	Weibull-calibrated Support Vector Machine

List of Symbols

$\mathbf{a}_{(e_{j,u},r)}$	Point along edge $e_{j,u}$
$A_{e_{j,u}}$	Set of equally spaced points $\mathbf{a}_{(e_{j,u},r)}$ along $e_{j,u}$
$\hat{b}$	Bias term in binary SVM
b	SV-Means outer index (number of initializations)
c	Class number
C	Soft-margin formulation SVM regularization term
d	Number of dimensions (spectral coefficients)
d_{RF}	Number of random features
$e_{j,u}$	Edge in minimum spanning tree S_j
E_j	Set of edges of longest length
F_j	Set of points associated with each ν_j
$\mathcal{F}$	High-dimensional Hilbert space
$g_G(\cdot)$	Decision function for minimum-volume set G
G	Minimum-volume set
$\mathbf{h}$	Gaussian distribution sampled vector
i	Data index
j	Quantile index
k_j	Number of clusters at level set j
$k(\cdot)$	Kernel function
$l_{e_{j,u}}$	Length of edge $e_{j,u}$
$\hat{l}_j$	Mean of the edge lengths in minimum spanning tree S_j
m	SV-Means inner index (number of ν values)
n	Number of data points
n_t	Number of subtrees
o	Quantile index
$O(\cdot)$	Order of operations
p	F_j index, $F_{j,p}$ is the pth index of F_j
P	Number of pulses
$\Pr$	Probability density function
q	Number of quantiles (minimum volume sets)
r	Point index along edge in minimum spanning tree
R	Radius of hypersphere
$\mathbb{R}$	Real number space
s	Data point index
S_j	Minimum spanning tree at level set j
$T_j^{n_t}$	Set of disjoint subtrees $S_j^v \in S_j$
u	Minimum spanning tree edge index
v	Subtree index
$\mathbf{w}$	SVM direction vector
$\mathbf{x}_i$	Data vector
$\mathbf{x}_{\text{sv}}$	Data vector that is a support vector
$\mathbf{x}'$	Test data vector

X	Data matrix
y_i	Class label corresponding to $\mathbf{x}_i$ in binary SVM
y_j	Cluster label for level set j
z	Explicit data transformation map via random feature function
α	Minimum volume density
β	Lagrange multiplier
δ	Normalized distance metric
Φ	Implicit data transformation map via a kernel function
η	Learning rate
λ	Lagrange multiplier
ν	Soft-margin formulation SVM regularization term
$\boldsymbol{\nu}^o$	ν vector defining output quantiles
$\boldsymbol{\nu}^t$	ν vector defining training quantiles
ρ	OCSVM margin width parameter
ρ_j	ρ corresponding to level set j
σ_j	Standard deviation of edge lengths in minimum spanning tree S_j
σ^2	Variance
Θ	SNR range
ξ	SVM slack variable
Ξ	SNR range index

.

Contents

List of Figures

List of Tables

Acknowledgments

I would like to thank my husband for his support and kindness during this process. In addition, I would like to thank my family for cheering me on and my friends who never gave up on me. Finally, I would like to thank my committee and my adviser, Brian Rigling, for their invaluable guidance.

Chapter 1

Introduction

1.1 Motivation

Radar waveform classification is an important task within various systems and applications, especially for future operations of cognitive radios and radars ([3], [4]). With increasing spectrum density, real-time situational awareness is imperative to maintain effective operation [5]. Waveform classification presents several key challenges. First, due to the crowded spectrum, waveform classification algorithms must operate in an open set framework [6] which means it is unlikely that every waveform encountered will be in the training library. Therefore, the ability to reject waveforms that are not in the library is critical to accurate waveform classification. Second, in order to take action based on the classification decision, it is important for the classifier to provide a likelihood or confidence in its answer to not only distinguish between known classes but to aid in rejection. Third, as the environment changes or as new information is discovered, a generative Bayesian formulation is needed to change the decision boundary on-the-fly as the prior information evolves. Fourth, as testing data is rejected, a robust clustering technique is needed to determine how many new classes are within the rejected data. Finally, timing is a significant factor. With the spectrum changing rapidly, it is necessary to have the ability to train a new waveform class

on-the-fly with large amounts of data.

These important and difficult challenges served as motivation in the development of a new fast algorithm, SV-Means, to address radar waveform classification for various applications: anomaly detection, density estimation, open set classification, and clustering. However, the challenges related to radar waveform classification are not unique and apply to several areas of research including network intrusion ([7], [8]), credit card fraud ([9], [10]), and image processing ([11], [12]).

1.2 Contributions

In this effort, two overarching contributions are made: (1) the development of the novel SV-Means algorithm and (2) the demonstration of SV-Means in several application areas. Specific contributions within the algorithm development, and within each application area are detailed below.

1.2.1 SV-Means Algorithm

A novel algorithm is developed, SV-Means, which was inspired by the quantile one-class support vector machine (q-OCSVM) density estimation algorithm in [13]. The SV-Means algorithm has several desirable properties that are considered to be significant contributions, specifically, a generative Bayesian formulation, an accurate boundary definition, and a fast training capability.

1.2.1.1 Generative Bayesian Formulation

Traditionally, support vector machines are discriminative classifiers. The SV-Means algorithm transforms the support-vector machine-based problem into a generative Bayesian formulation by modeling each class likelihood with multiple hierarchical boundaries accurately delineating probability quantiles.

1.2.1.2 Accurate Boundary Definition

The SV-Means algorithm's weight vector is estimated based on the importance of an accurate boundary for classification rejection, which is accomplished by level set estimation towards the extrema of the data. Typically, the outer boundary of the OCSVM is determined by a small amount of points, which could provide an unstable boundary in high dimensions. The SV-Means outer-level set estimation provides a form of regularization for a more accurate boundary.

1.2.1.3 Fast Training

The q-OCSVM density estimation algorithm is solved from the dual formulation with a global convex optimization using a Gaussian kernel where extra parameters are needed to optimize for each density level set. The SV-Means algorithm is solved from the primal formulation using a non-convex, k-means inspired algorithm based on stochastic gradient descent principles using random Fourier features to estimate the kernel. The SV-Means algorithm, therefore, trains at least an order of magnitude faster in comparison to q-OCSVM, and open set algorithms, as the number of training instances increase. Therefore, SV-Means is compatible with on-line training approaches. In addition, the training time for SV-Means does not increase as a function of the number of level sets as it does with the q-OCSVM algorithm.

1.2.2 SV-Means for Multiple Applications

In addition to the algorithmic contributions which advanced the state-of-the-art in OCSVMs. Contributions were made in applying this novel, improved algorithm to several important functions central to waveform classification and other application areas: anomaly detection, distribution estimation, open set classification (with rejection), and clustering. The SV-Means algorithm has been directly compared to state-of-the-art algorithms within each

application area, and has performed favorably, with the exception of the clustering algorithm. The clustering algorithm (SV-Means Level Set Clustering) emerged near the end of this research, and a comparison of this novel technique to other clustering algorithms remains as future work.

1.2.2.1 Anomaly Detection

SV-Means provides an accurate boundary, as described in a previous contribution in Section 1.2.1.2, and therefore, is a highly effective algorithm for anomaly detection, especially in higher dimensions. This is demonstrated in the experiments in Section 5.3.3.

1.2.2.2 Density Estimation

SV-Means, as mentioned previously, is inspired by the q-OCSVM density estimation algorithm. The q-OCSVM optimizes over all level sets, and hence, compromises an accurate boundary. SV-Means is able to focus on estimating the boundary level set and the remaining level sets are found via line search. As mentioned in a previous contribution in Section 1.2.1.3, SV-Means is faster than the q-OCSVM algorithm by two orders of magnitude as the number of training points and number of level sets increase.

1.2.2.3 Open Set Classification

SV-Means is novel in comparison to other open set algorithms in that it is the only algorithm that exclusively uses OCSVMs. Use of OCSVMs was avoided formerly due to issues distinguishing between known classes, which the SV-Means algorithm solves by density level set comparison. Another desirable property of the SV-Means algorithm is that each class is trained separately, so a new class can be added without re-training the entire classifier. For example, rejected samples can be retrained and then used by a clustering algorithm to group points together to add new classes to the library on-the-fly.

1.2.2.4 SV-Means Level Set Clustering

A novel algorithm, SV-Means Level Set Clustering, was developed, inspired by the Support Vector Clustering algorithm in [14]. The SV-Means Level Set Clustering algorithm provides multimodal level set boundaries that have a probabilistic meaning whereas the Support Vector Clustering algorithm provides multimodal boundaries with only structural meaning. The SV-Means Level Set Clustering algorithm is able to determine the number of clusters per level set and is able to distinguish overlapping clusters. These additional characteristics are not provided by the Support Vector Clustering algorithm.

1.2.3 Published/Submitted Papers Related to this Dissertation

A conference paper entitled "Phase-Modulated Radar Waveform Classification Using Quantile One-Class SVMs" was published in the *2015 IEEE Radar Conference Proceedings* which provided the first step in transforming the q-OCSVM density algorithm into a classification formulation. A journal paper entitled "SV-Means: A Fast SVM-Based Level Set Estimator for Phase-Modulated Radar Waveform Classification" was submitted to the *IEEE Journal of Selected Topics: Signal Processing for Machine Learning for Cognition in Radio Communications and Radar*. The journal paper details the novel SV-Means algorithm and compares the algorithm to the primal and dual q-OCSVM, and several open set algorithms.

1.3 Outline of Dissertation

In Chapter 2, a literature review is performed which surveys the binary SVM, one-class SVM, quantile one-class SVM, and several techniques that have been used to accelerate SVM-based algorithms. In Chapter 3, the SV-Means algorithm is described including its motivation and convergence properties. In Chapter 4, several application areas are described where SV-Means is a strong candidate for use: anomaly detection, density esti-

mation, open set classification, and clustering. In Chapter 5, experimentation results are shown for several different experiments including: comparing the SV-Means algorithm to existing algorithms in the application areas described in Chapter 4; an illustrative example demonstrating the novel SV-Means Level Set Clustering algorithm; and an end-to-end demonstration detailing training, testing, clustering, and adding new classes to the library using the SV-Means algorithm. Finally, in Chapter 6, concluding remarks are provided including recommendations for future work.

1.4 Notation

The following notational conventions are followed. Vectors are represented by lowercase bold characters (e.g., $\mathbf{x}$, $\mathbf{w}$, $\boldsymbol{\xi}$), scalars are denoted by lowercase characters (e.g., ξ), matrices are given by capital letters (e.g., X), and an apostrophe on a character represents a test point (e.g., $\mathbf{x}'$). There are a few exceptions to these notations when denoting transforms (e.g., $\Phi(\cdot)$, $z(\cdot)$). In addition, L denotes calculating the Lagrangian, P represents the Gaussian distribution, and any capital letter with super- or sub-scripts represent sets (e.g, X_j^+). Minimum volume sets are denoted by G, the Gaussian kernel is denoted by $k(\cdot, \cdot)$, and the decision functions are given by $g_{G_j}(\cdot)$.

Chapter 2

Background

This chapter discusses several research areas that provide a background and basis for the SV-Means algorithm developed in Chapter 3. In Section 2.1, a thorough literature review was performed on several support vector machine variations. In Section 2.2, algorithms discussed in the literature review are compared within an easy-to-view chart to highlight key elements of support vector machine algorithms.

2.1 Literature Review

The support vector machine (SVM) is a prominent technique in machine learning first developed by Vapnik [15]. The SVM finds a separating hyperplane between two sets of data points with maximum margin. Let $\{\mathbf{x}_i | i = 1, \ldots, n\}$ be the training set of n vectors where $\mathbf{x}_i \in \mathbb{R}^d$ and let $y_i \in \{-1, 1\}$ be the class label of each $\mathbf{x}_i$. Figure 2.1 depicts the binary support vector machine. The equation for the separating hyperplane is $\langle \mathbf{w}, \mathbf{x}_i \rangle + \hat{b} = 0$ where $\mathbf{w}$ is the weight vector perpendicular to the hyperplane and $\hat{b}$ is the bias. To find the maximum margin, two parallel hyperplanes are found that touch the closest data from each class. The closest data points are called support vectors (highlighted in yellow in Figure

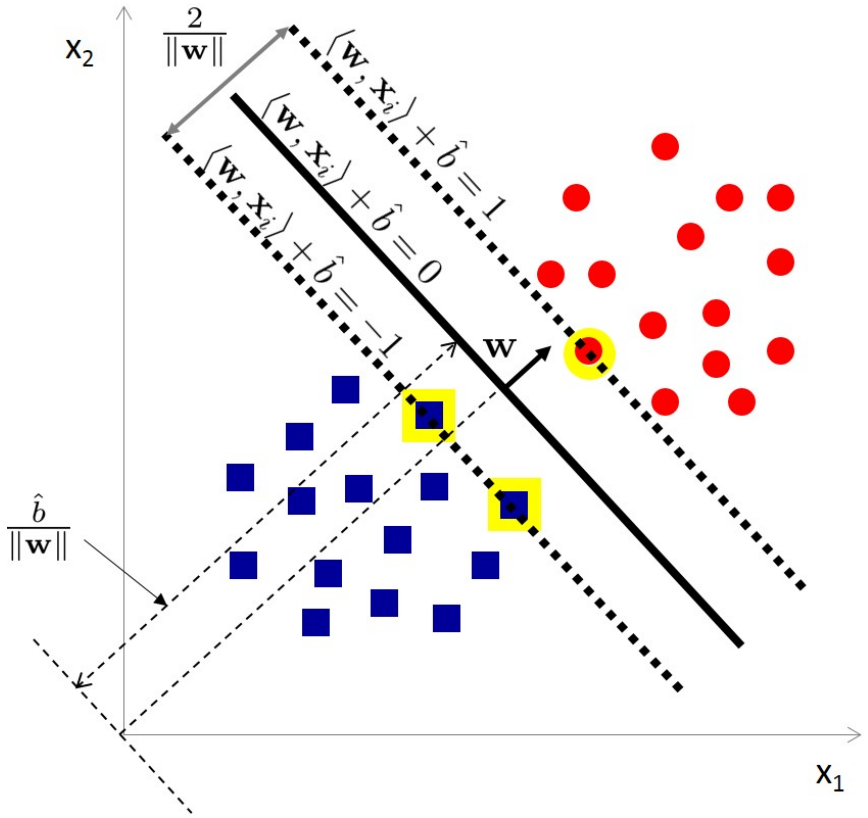


Figure 2.1: Hard margin, binary support vector machine.

2.1). The two parallel hyperplanes are defined by

$$\begin{aligned} \langle \mathbf{w}, \mathbf{x}_i \rangle + \hat{b} &= -1 \\ \langle \mathbf{w}, \mathbf{x}_i \rangle + \hat{b} &= 1 \end{aligned} \tag{2.1}$$

which gives a margin of $\frac{2}{\|\mathbf{w}\|}$. The problem simply becomes maximizing the margin $\frac{2}{\|\mathbf{w}\|}$, however, this is a non-convex problem. Instead, the equivalent problem of minimizing $\frac{\|\mathbf{w}\|}{2}$ is solved by substituting $\frac{\|\mathbf{w}\|^2}{2}$ as squaring is monotonic. The final optimization to find the maximum margin is given by the primal formulation

$$\begin{aligned} &\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2 \\ &\text{s.t. } y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + \hat{b}) \geq 1, \ i \in [n] \end{aligned} \tag{2.2}$$

where the contraints are formed using (2.1).

Equation (2.2) is used for separable data with a hard margin. For non-separable data, soft margins were developed by Cortes and Vapnik [16] to allow training points within the margin at a cost. This allows overlapping classes to develop an optimal boundary instead of a boundary determined by a very small margin. The soft margin is obtained by relaxing the constraint in (2.2) by adding slack variables $\xi_i \geq 0,\ i \in [n]$. The updated optimization problem with soft margins is given by

$$\begin{aligned} &\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n} \xi_i^2 \\ &\text{s.t. } y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + \hat{b}) \geq 1 - \xi_i,\ \xi_i \geq 0,\ i \in [n] \end{aligned} \tag{2.3}$$

where C is a variable that helps control the size of the margin.

Equation (2.2) and (2.3) are used for linear classifiers. The SVM algorithm was extended as a non-linear classifier by the use of kernels by Boser, Bernhard, and Vapnik [17]. To show how kernels are implemented to achieve a non-linear boundary, the dual formulation of the soft margin classifier is derived by taking the Lagrangian $L(\cdot)$ of the primal formulation in Equation (2.3). The Lagrangian is given by

$$\begin{aligned} L(\mathbf{w}, b, \xi, \lambda, \beta) =& \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i \\ &- \sum_{i=1}^{n} \lambda_i (y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + \hat{b}) - 1 + \xi_i) - \sum_{i=1}^{n} \beta_i \xi_i \end{aligned} \tag{2.4}$$

with multipliers $\lambda_i \geq 0$, $\beta_i \geq 0$. When the partial derivatives are set equal to zero with respect to the primal variables $\mathbf{w}$, b, and ξ_i, one obtains

$$\mathbf{w} = \sum_{i=1}^{n} \lambda_i y_i \mathbf{x}_i \tag{2.5a}$$

$$\sum_{i=1}^{n} \lambda_i y_i = 0 \tag{2.5b}$$

$$\beta_i = C - \lambda_i. \tag{2.5c}$$

Combining (2.5) into (2.4), the dual program is given by

$$\begin{aligned} &\min_{\lambda_i} \frac{1}{2} \sum_{i=1}^{n} \sum_{s=1}^{n} \lambda_i \lambda_s y_i y_s \langle \mathbf{x}_i, \mathbf{x}_s \rangle - \sum_{i=1}^{n} \lambda_i \\ &\text{s.t. } 0 \leq \lambda_i \leq C, \ \sum_{i=1}^{n} \lambda_i y_i = 0 \end{aligned} \tag{2.6}$$

where the constraint $0 \leq \lambda_i \leq C$ was formed by realizing the constraint $\beta_i \geq 0$ produces a new constraint on (2.5c) yielding $C - \lambda_i \geq 0$ or $\lambda_i \leq C$ [18].

The decision function $g_G(\mathbf{x}) = \text{sign}(\langle \mathbf{w}, \mathbf{x}' \rangle + \hat{b})$ determines if the testing point $\mathbf{x}'$ is in the positive or negative class. The decision function can be rewritten by substituting in the equation for $\mathbf{w}$ (2.5a) which gives

$$g_G(\mathbf{x}) = \text{sign}(\sum_{i=1}^{n} \lambda_i y_i \langle \mathbf{x}_i, \mathbf{x}' \rangle + \hat{b}). \tag{2.7}$$

and it is noted that the equation is based off of an inner product between two data points.

If the training data is non-linearly separable, the data may be transformed into a high-dimensional (possibily infinite) data space where it is likely a linear hyperplane exists to separate the data. The linear hyperplane formed in high-dimensional space is non-linear in ambient space. Each $\mathbf{x}_i$ is transformed via a map $\Phi : \mathbb{R}^d \to \mathcal{F}$ where $\mathcal{F}$ is a high dimensional Hilbert space generated by a positive-definite kernel $k(\mathbf{x}_i, \mathbf{x}_s)$. The kernel function represents an inner product in $\mathcal{F}$ through $k(\mathbf{x}_i, \mathbf{x}_s) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_s) \rangle$. The "kernel trick" gives an implicit mapping of the data into $\mathcal{F}$ without having to calculate the high-dimensional space itself. Therefore, the kernel replaces the inner product in (2.7) and throughout the dual SVM derivation to provide a non-linear classifier.

The one-class support vector machine (OCSVM) algorithm was inspired by the binary support vector machine with soft margins in Equation (2.3). OCSVMs have two main derivations developed by Tax and Duin [19] and Scholkopf [20]. In [19], instead of separating the data with a hyperplane, the data is surrounded by a hypersphere that acts as

its boundary. The volume of the hypersphere is optimized to handle outliers. In [20], the data is separated from the origin by a hyperplane with maximum margin. This derivation is commonly referred to as the ν-formulation as this parameter has an informative meaning. The ν parameter is an upper bound on the fraction of outliers and a lower bound on the fraction of support vectors. The Scholkopf derivation is formalized in the next section as it is the basis for the quantile one-class support vector machine (q-OCSVM) which is the basis for the SV-Means algorithm.

2.1.1 One-Class Support Vector Machine

In the one-class SVM classification problem based on the original Scholkopf formulation [20], training points $\{\mathbf{x}_i | i = 1, \ldots, n\}$ where $\mathbf{x}_i \in \mathbb{R}^d$ are separated from the origin in feature space by a hyperplane and the distance from this hyperplane to the origin is maximized. In this popular ν-SVM formulation [20], the algorithm finds a function g_G that returns $+1$ in a region capturing "most" of the data points if $\mathbf{x}_i \in G$ where G is a minimum-volume set, and -1 elsewhere. Each $\mathbf{x}_i$ is transformed via a map $\Phi : \mathbb{R}^d \to \mathcal{F}$ where $\mathcal{F}$ is a high dimensional Hilbert space generated by a positive-definite kernel $k(\mathbf{x}, \mathbf{x}')$. The kernel function represents an inner product in $\mathcal{F}$ through $k(\mathbf{x}, \mathbf{x}') = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle$. If the data is non-separable, slack variables ξ_i allow for some points to be within the margin and the parameter $\nu \in (0, 1)$ is the regularization parameter that sets an upper bound on the fraction of these margin errors. The ν-SVM formulation is solved using the following optimization:

$$\begin{aligned} &\min_{\mathbf{w}, \boldsymbol{\xi}, \rho} \frac{1}{2} \|\mathbf{w}\|^2 - \rho + \frac{1}{\nu n} \sum_{i=1}^{n} \xi_i \\ &\text{s.t. } \langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle \geq \rho - \xi_i, \ \xi_i \geq 0, \ i \in [n], \end{aligned} \tag{2.8}$$

where $\rho \in [0, 1]$ controls the width of the margin, which is equal to $\frac{\rho}{\|\mathbf{w}\|}$. The sign of function $g_G(\mathbf{x}') = \text{sgn}(\langle \mathbf{w}, \Phi(\mathbf{x}') \rangle - \rho)$ determines whether a point is in the positive or negative set. The minimum volume set $G(\alpha)$, where α is the density desired, is approximated when

the OCSVM is solved for $\nu = 1 - \alpha$ [13].

2.1.2 Quantile One-Class Support Vector Machine

In [13], a quantile one-class SVM (q-OCSVM) algorithm was introduced that gives an approach to estimate the distribution of the data. A sequence of quantiles is defined, $0 < \alpha_1 < \alpha_2, \ldots, < \alpha_q < 1$, where q is the number of minimum-volume sets to approximate. The q-OCSVM algorithm generalizes Equation (2.8) by approximating a set of minimum-volume sets $\{G_1, \ldots, G_q\}$ so that the hierarchy constraint $G_i \subseteq G_j$ is satisfied for $i < j$. The q-OCSVM algorithm [13] solves the following primal problem:

$$\begin{aligned} &\min_{\mathbf{w},\boldsymbol{\xi}_j,\rho_j} \frac{q}{2}\left\|\mathbf{w}\right\|^2 - \sum_{j=1}^{q}\rho_j + \sum_{j=1}^{q}\frac{1}{\nu_j n}\sum_{i=1}^{n}\xi_{j,i} \\ &\text{s.t. } \langle \mathbf{w}, \Phi(\mathbf{x}_i)\rangle \geq \rho_j - \xi_{j,i},\ \xi_{j,i} \geq 0,\ j \in [q],\ i \in [n] \end{aligned} \tag{2.9}$$

where $\nu_j = 1 - \alpha_j$. The program finds multiple, parallel half-space decision functions by searching for a global minimum over the sum of q objective functions while the programs share the same $\mathbf{w}$. The q half-spaces in the solution are only different by their bias terms, which makes them parallel to each other.

The derivation of the the q-OCSVM dual program is developed in [13], but is repeated here as a reference for the new SV-Means algorithm. The Lagrangian of (2.9) is

$$\begin{aligned} L(\mathbf{w}, \boldsymbol{\xi}, \boldsymbol{\rho}, \lambda, \beta) =& \frac{q}{2}\left\|\mathbf{w}\right\|^2 - \sum_{j=1}^{q}\rho_j + \sum_{j=1}^{q}\frac{1}{\nu_j n}\sum_{i=1}^{n}\xi_{j,i} \\ &- \sum_{j=1}^{q}\sum_{i=1}^{n}\lambda_{j,i}\left(\langle \mathbf{w}, \Phi(\mathbf{x}_i)\rangle - \rho_j + \xi_{j,i}\right) \\ &- \sum_{j=1}^{q}\sum_{i=1}^{n}\beta_{j,i}\xi_{j,i} \end{aligned} \tag{2.10}$$

with multipliers $\lambda_{j,i} \geq 0$, $\beta_{j,i} \geq 0$. When the partial derivatives are set equal to zero with

respect to the primal variables $\mathbf{w}$, ρ_j, and ξ_j, one obtains

$$\mathbf{w} = \frac{1}{q}\sum_{j,i}\lambda_{j,i}\Phi(\mathbf{x}_i) \tag{2.11a}$$

$$\sum_{i}\lambda_{j,i} = 1 \tag{2.11b}$$

$$0 \leq \lambda_{j,i} \leq \frac{1}{n\nu_j},\ i \in [n],\ j \in [q]. \tag{2.11c}$$

Combining (2.11) into (2.10), the dual program is given by

$$\min_{\lambda}\frac{1}{2q}\sum_{j,r\in[q]}\sum_{i,s\in[n]}\lambda_{j,i}\lambda_{r,s}k(\mathbf{x}_i,\mathbf{x}_s) \tag{2.12}$$

with constraints (2.11b) and (2.11c) and where the dot product $(\langle\Phi(\mathbf{x}_i),\Phi(\mathbf{x}_s)\rangle)_{\mathcal{F}}$ is commonly replaced with a kernel, $k(x_i,x_s)$. The resulting decision function for the jth estimate is

$$g_{G_j}(\mathbf{x}') = \text{sgn}\left(\frac{1}{q}\sum_{j,i}\lambda_{j,i}k(\mathbf{x}_i,\mathbf{x}') - \rho_j\right). \tag{2.13}$$

The values for ρ_j, using the condition $1 < \lambda_{j,i} < \frac{1}{n\nu_j}$, are found from a point x_{sv} by

$$\rho_j = \langle\mathbf{w},\Phi(\mathbf{x}_{sv})\rangle = \frac{1}{q}\sum_{j,i}\lambda_{j,i}k(\mathbf{x}_i,\mathbf{x}_{sv}). \tag{2.14}$$

This dual formulation for the one class SVM has significant advantages. It is derived from the ν-SVM formulation, which supports calibrated quantiles as ν approximately equals the fraction of support vectors and outliers ([21], [20]). And, the data only appears in the formulation as inner products supporting the "kernel trick" to provide non-linear decision boundaries. This formulation can be used to build a Bayesian probabilistic classifier that provides confidence and detects outliers ([22], [23]). Unfortunately, the dual formulation's quadratic solver has a complexity that grows with the size of the data, and hence,

large data sets are not feasible especially for on-line scenarios.

In order to address these complexity issues, in recent years, randomized projections of the data have been used to approximate kernel functions with great success across several different algorithm types ([24], [25], [26], [27]) including OCSVMs [28]. We use these Fourier random features, which give non-linear decision boundaries in linear formulations, as a first step in a OCSVM level set quantile solver, SV-Means.

2.1.3 SVM Speed-Up

Unfortunately, SVM techniques are burdened by computational complexity as they are solving a global convex quadratic programming problem and, in most cases, using a non-linear kernel where the complexity is magnified as the number of training samples increase. To compute the kernel for every training pair, it requires $O(n^2)$ computations and to solve the QP problem, $O(n^3)$ computations are required where n is the number of data points [29]. In the q-OCSVM case, additional optimization parameters are included to simultaneously solve for multiple hierarchal boundaries which also increases training time. For large data sets, this complexity is a bottleneck and several ideas to address this complexity issue have been formulated including: dimensionality reduction ([30], [31], [32]), decomposition ([33], [22]), primal formulations ([34], [35]), stochastic gradient descent and iterative methods ([36], [37], [38], [39]), and randomization techniques ([40], [41], [28]). These formulations are briefly discussed next.

Dimensionality reduction techniques, or pre-processing the data before performing the classification or optimization method, helps address scalability. A few examples of this approach are principle components analysis (PCA) [30], ISOMAP [31], and neural networks [32]. These approaches reduce the effective number of features but potentially at the cost of classification performance. They are important and can aid in reducing computation, but they do not specifically address the key limitation of the dual formulation of the SVM, which is that the QP solution scales predominantly with the number of data points not the

number of features. On the other hand, these techniques could prove more useful in the primal formulation which scales predominately with the number of features.

Decomposition methods, including sequential minimal optimization (SMO) [33], breaks the problem into workable subsets. Along the same vein, in [22], the phase-modulated radar waveform data was broken up into smaller overlapping chunks to have manageable class sizes while covering the variability of the data set. Primal formulations are popular with their implementation of non-linear kernels, which is normally handled with the dual formulation [34]. For primal formulations in [34] and [35], the QP problem is avoided with Newton's method by using a quadratic loss function. Other fast methods to solve SVM formulations are stochastic gradient descent and iterative methods ([36], [37], [38], [39]). However, if non-linear decision boundaries are desired, these methods still, however, must use SVM dual formulation and the kernel matrix, which scales poorly with large data sets.

In [28], the authors applied random features [40] to the anomaly detection problem using a OCSVM linear solver inspired by Rahimi and Recht [41]. The randomized projection of the data allows the use of primal linear formulations for significant speed-up in optimization, often two orders of magnitude faster than their dual formulation counterpart.

SV-Means addresses this complexity issue by reformulating the primal q-OCSVM problem into a non-convex optimization utilizing stochastic gradient descent and k-means principles. This formulation is unique as the weights are iteratively updated by taking the mean of the support vectors and outliers of the class. The support vectors and outliers are determined by a line search along the based on ν or the percentage of training instances. When the optimization is complete, a final weight vector is given and an additional line search is used to find the density level sets. In addition, the SV-Means algorithm is at least an order of magnitude faster than q-OCSVM as the number of training instances increase.

2.1.4 Random Fourier Features

Algorithms that use the kernel matrix, as in (2.12), incur larger computational and storage costs as the number of points increase. A kernel generates an implicit lifting of the data $\Phi : \mathbb{R}^d \to \mathcal{F}$ where $\mathcal{F}$ is a high-dimensional Hilbert inner product space that is applied to all pairs of data points. In [40], the authors propose an explicit mapping of the data $z : \mathbb{R}^d \to \mathbb{R}^{d_{RF}}$ to a low-dimensional Euclidean inner product space using a randomized Fourier feature map. The inner product between a pair of transformed points approximates the kernel, $k(\mathbf{x}_i, \mathbf{x}_s) = \Phi(\mathbf{x}_i)^\intercal \Phi(\mathbf{x}_s) \approx z(\mathbf{x}_i)^\intercal z(\mathbf{x}_s)$, where

$$z(\mathbf{x}_i) = \sqrt{\frac{2\sigma^2}{d_{RF}}} \begin{bmatrix} \sin(\mathbf{h}_1^\intercal \mathbf{x}_i) \\ \cos(\mathbf{h}_1^\intercal \mathbf{x}_i) \\ \vdots \\ \sin(\mathbf{h}_{d_{RF}/2}^\intercal \mathbf{x}_i) \\ \cos(\mathbf{h}_{d_{RF}/2}^\intercal \mathbf{x}_i) \end{bmatrix}, \ \mathbf{h}_u \overset{iid}{\sim} P(\mathbf{h}), \tag{2.15}$$

$P(\cdot)$ is the Gaussian distribution, σ is the kernel bandwidth, and $\mathbf{h}_u$ is $(d \times 1)$ where $u = 1, \ldots d_{RF}$. This positive-valued Fourier transformation was proved superior in [42] for the Gaussian kernel. In the SV-Means algorithm (and the primal q-OCSVM solved with QP in Section 5), the data set $X = [\mathbf{x}_1 \ldots \mathbf{x}_n]$, $(d \times n)$, is transformed via $z(X)$ in (2.15) where $z(\cdot)$ is applied to each column of X, $(d_{RF} \times n)$.

In Chapter 3, $\Phi(\mathbf{x}_i)$ is used to describe and relate the theory behind the algorithm. In order to perform the SV-Means algorithm, $\Phi(\mathbf{x}_i)$ is replaced with $z(\mathbf{x}_i)$. When referring to the the transformed data matrix, $z(X)$, the shorthand Z is used.

2.2 SVM Algorithm Comparison

Some of the aforementioned papers are categorized in Table 2.1. This SVM overview table compares each algorithm by listing key elements of SVM algorithms including their formu-

lation (closed or open), function (Classification (C), Classification with rejection (CwR), Anomaly Detection (AD), and Distribution Estimation (DE)), the number of classes, the objective function used (dual or primal), the parameterization used (C or ν), the type of transform (kernel, Random Features (RF), Principle Component Analysis (PCA), etc.), the loss function (Hinge, Quadratic, etc.), and the solver (Quadratic Programing (QP), iterative techniques, etc.). SV-Means is the only approach that is able to tackle all the functions listed, in an open set environment, while employing acceleration techniques including random features for kernel estimation and a unique fast solver, as described in Chapter 3.

2.3 Chapter Summary

In this chapter, a literature review was provided, highlighting topics including the one-class support vector machine, quantile one-class support vector machine, several algorithms developed to accelerate the support vector machine, and random Fourier features. All of these topics aided in the development of the novel SV-Means algorithm discussed in Chapter 3.

Table 2.1: SVM Algorithm Comparison

	Form -ulation	Function	# of classes	Objective Function	Para -meter	Transform	Loss Function	Solver
Chapelle [34]	Closed	C	2	Primal	C	Kernel	Quadratic	Newton
Glazer [13]	Closed	DE	1	Dual	ν	Kernel	Hinge	QP
Thomas [43]	Closed	DE	1	Dual	ν	Kernel	Hinge	Mult. dual solutions with different BWs
Liu [23]	Closed	C, DE	1	Dual	ν	Kernel	Hinge	OCSVM-based Bayesian
Erfani [28]	Closed	AD	1	Primal	C	RF	Hinge	Linear [41]
Wang [35]	Closed	AD	1	Primal	C	Kernel	Huber	Newton
Navia-Vazquez [36]	Closed	C	2	Primal	C	PCA	Hinge	Weighted LS
Gomez-Verdejo [37]	Closed	AD	1	Dual	C	Kernel	Hinge	Online Iterative Re-Weighted LS
Kivinen [38]	Closed	AD	1	Primal	ν	Kernel	Hinge	Online Stochastic GD
Shalev-Shwartz [39]	Closed	C	2	Primal	C	Kernel	Hinge	Stochastic Sub-GD
Pavy [this dissertation]	Open	CwR, AD, DE	1	Primal	ν	RF	Hinge	SV-Means

Chapter 3

SV-Means Algorithm

In this Chapter, the SV-Means algorithm is developed by expanding the q-OCSVM density estimation algorithm into a non-linear classification formulation using random Fourier features for kernel estimation and the primal objective function. SV-Means solves the optimization problem by using a non-convex, k-means inspired approach based on stochastic gradient descent principles. A detailed algorithm description is provided here, including insights into its convergence properties.

3.1 Algorithm Motivation

The development of the SV-Means algorithm is motivated by the fact that the SVM formulation is convex and that the inequality constraints are affine. In this case, strong duality (when the primal optimal objective and the dual optimal objective are equal) holds and Slater's conditions are met [44]. Given these restrictions and the fact that the objective function and constraints are differentiable, the Karush-Kuhn-Tucker (KKT) conditions are necessary and sufficient for an optimal solution [44].

It is observed that the KKT conditions in (2.11) can be reached by defining $\lambda_{j,i} = \frac{1}{n\nu_j}$ and redefining the summation of all $\Phi(\mathbf{x}_i)$ for $i = 1 \dots n$ in (2.11a) to a summation over only the $\Phi(\mathbf{x}_i)$ points associated with ν_j. The set of points, F_j, associated with each ν_j is

determined by calculating a ρ_j via a line search along the projection $\langle \mathbf{w}, \Phi(X) \rangle$ until $\lfloor n\nu_j \rceil$ points are outside the separating hyperplane where $\lfloor \cdot \rceil$ represents the nearest integer. The set of points associated with each ν_j is defined as $F_j = \{\Phi(\mathbf{x}_i) \in \Phi(X) : \langle \mathbf{w}, \Phi(\mathbf{x}_i) \rangle - \rho_j \leq 0, \quad i = 1 \dots n\}$ where $|F_j| = \lfloor n\nu_j \rceil$ and $F_{j,p}$ references the pth point in set F_j with $p = 1 \dots |F_j|$. These points satisfy the constraints in (2.9) and correspond to the ξ_i on the origin side of the separating hyperplane as depicted in Figure 3.1. Using the definitions of $\lambda_{j,i}$ and the set of points associated with each ν_j, equation (2.11a) is rewritten as

$$\mathbf{w} = \frac{1}{q} \sum_{j=1}^{q} \frac{1}{|F_j|} \sum_{p=1}^{|F_j|} F_{j,p} \tag{3.1}$$

and it is noted that F_j is a function of the previous $\mathbf{w}$. Iteratively, $\mathbf{w}$ is solved jointly over multiple ν values. Hence, to satisfy (3.1), the $\mathbf{w}$'s of each iteration are combined in a scheduled fashion using a learning rate as in [45]. This combination of calculating a mean of selected points and then calculating a new set of points associated with the new mean is a version of the k-means algorithm with $k = 1$, hence, the name SV-Means.

3.2 Convergence

Next the specifics of the solver are explained. K-means can be solved by stochastic gradient descent [46] and provably converges to a local optimum [47]. Using stochastic gradient descent principles, it has been shown that the stochastic k-means algorithm globally converges to a local optimum at an $O(1/m_{max})$ rate (where m_{max} is the number of iterations) with high probability even under the difficult conditions of the non-convex and non-differentiable k-means objective [45]. The iterative procedure in the SV-Means algorithm to optimize the q-OCSVM is a variant of the stochastic gradient k-means solution. Solving the OCSVM in this way essentially provides a slight relaxation of the objective so that the problem is no longer convex. This relaxation is mitigated in the SV-Means al-

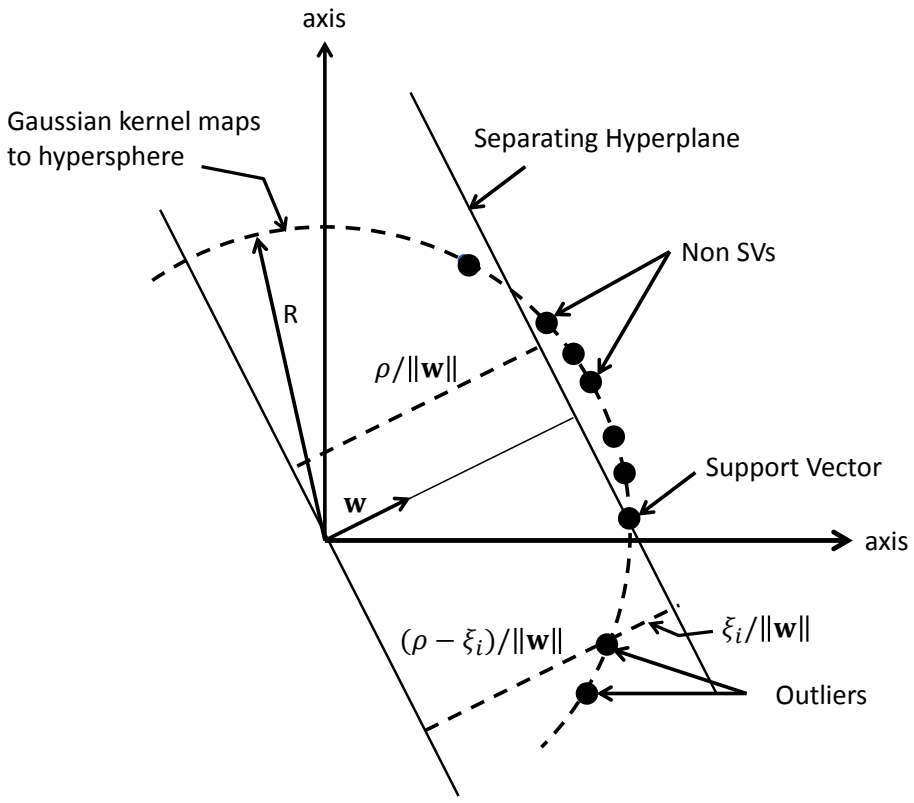


Figure 3.1: Illustration of SVM geometry. A Gaussian kernel maps points onto a hypersphere. (Figure based on depiction in [1].)

gorithm similar to other k-means solvers by seeding the algorithm with multiple starting points and then selecting the best solution. Although the solution is non-convex, the experiments in Section 5.3.2 show that this novel solution gives comparable performance to the dual q-OCSVM (modified to perform classification) whose formulation is convex, and sometimes better performance than the dual q-OCSVM due to the improved boundary. The k-means formulation developed here provides a significant speed-up compared to quadratic solvers.

3.3 Algorithm Description

Again, it is noted that we are using the random Fourier feature transform described in Section 2.1.4, $z(\mathbf{x}_i)$, as an approximation to $\Phi(\mathbf{x}_i)$ for the SV-Means algorithm. The first step of the algorithm (Algorithm 1) is to initialize parameters b_{max}, m_{max}, ν^t_{min}, ν^t_{max}, and $\boldsymbol{\nu}^o$. The algorithm is initialized b_{max} times by randomly choosing points from the training data, forming an algorithm outer loop. The core of the SV-Means algorithm is within the inner loop (lines 5-14) where the parameter m_{max} is the number of inner loop iterations. It is important to distinguish between the ν values used to train, denoted by ν^t, which are randomly drawn from the uniform distribution bounded by ν^t_{min} and ν^t_{max} (line 6), where the superscript t is again used to denote training. The ν values included in $\boldsymbol{\nu}^o$ are used to calculate the final $\boldsymbol{\rho}_b$ output (corresponding to the final quantiles desired) of each outer loop intialization (line 16), hence, the superscript o, where $\boldsymbol{\nu}^o = [\nu^o_1, \nu^o_2, \ldots, \nu^o_q]$. For all the experiments in this dissertation, $b_{max} = 10$ and $m_{max} = 20$. For the experimentation in Sections 5.2 and 5.3.4, $\nu^t_{min} = 0$ and $\nu^t_{max} = 1$, were used for distribution estimation. For the radar waveform results in Sections 5.3.1, 5.3.2, and 5.3.5, $\nu^t_{min} = 0$ and $\nu^t_{max} = 0.05$, were used to estimate the OCSVM boundary for improved rejection capability.

A line-by-line descripion of the algorithm is as follows. A random training sample is chosen as the first estimate of the weight vector, $\mathbf{w}$. Throughout the algorithm execution, after an estimate of the weight vector is determined, the vector is normalized as we are only interested in the direction. The estimate of $\mathbf{w}$ is optimized via an iterative k-means stochastic gradient inspired approach where random ν^t_m values are chosen between ν^t_{min} and ν^t_{max} to ensure a joint optimization of multiple quantiles. This process begins with a random ν^t_m draw (line 6) which is used to compute ρ in a golden section line search $gs(\cdot,\cdot)$ (line 7). The line search is performed along the projection $\mathbf{w}^\intercal Z$ and the distance ρ is found when $n\nu^t_m$ of the points are outliers or support vectors (outside or on the boundary). These points are formalized as the set Z^- and the estimated weight vector $\mathbf{w}_m$ is computed by taking their mean (denoted by $\langle\cdot\rangle$) (line 9). Next, an iterative stochastic gradient descent approach

Algorithm 1: SV-Means

1 initialize: b_{max}, m_{max}, ν^t_{min}, ν^t_{max}, and $\boldsymbol{\nu}^o$
2 **for** $b := 1$ *to* b_{max} **do** *Random start*
3 $\quad \mathbf{w} = \mathbf{z}_t, \quad \mathbf{z}_t \in Z : t \sim U(1, n)$
4 $\quad \mathbf{w} = \mathbf{w}/||\mathbf{w}||_2$
5 $\quad$ **for** $m := 1$ *to* m_{max} **do** *Estimate* $\mathbf{w}$
6 $\quad\quad \nu^t_m = \hat{\nu}_m, \quad \hat{\nu}_m \sim U(\nu^t_{min}, \nu^t_{max})$
7 $\quad\quad \rho = gs(\mathbf{w}^\intercal Z, \nu^t_m)$
8 $\quad\quad Z^- = \{\mathbf{z}_i \in Z : \mathbf{w}^\intercal \mathbf{z}_i - \rho \leq 0, \ i = 1 \dots n\}$
9 $\quad\quad \mathbf{w}_m = \langle Z^- \rangle$
10 $\quad\quad \mathbf{w}_m = \mathbf{w}_m/||\mathbf{w}_m||_2$
11 $\quad\quad \eta = 1/m$
12 $\quad\quad \mathbf{w} = (1 - \eta)\mathbf{w} + \eta \mathbf{w}_m$
13 $\quad\quad \mathbf{w} = \mathbf{w}/||\mathbf{w}||_2$
14 $\quad$ **end**
15 $\quad \mathbf{w}_b = \mathbf{w}$
16 $\quad \boldsymbol{\rho}_b = gs(\mathbf{w}_b^\intercal Z, \boldsymbol{\nu}^o)$
17 **end**
18 $\mathbf{w}^* = \mathbf{w}_b$ corresponding to $\boldsymbol{\rho}_b$ with $\max(\sum_{j=1}^q \boldsymbol{\rho}_b)$
19 $\boldsymbol{\rho}^* = \boldsymbol{\rho}_b$ with $\max(\sum_{j=1}^q \boldsymbol{\rho}_b)$

is applied with a step size that gradually decreases the contribution of the estimated $\mathbf{w}_m$. After m_{max} iterations, a weight vector $\mathbf{w}_b$ is found and $\boldsymbol{\rho_b} = [\rho_1, \rho_2, \dots, \rho_q]$ is calculated via golden section line search using the final $\mathbf{w}_b$ and $\boldsymbol{\nu}^o$.

After repeating this process b_{max} times, b_{max} $\mathbf{w}_b$s have been calculated. The optimal $\mathbf{w}^*$ is chosen by selecting the $\mathbf{w}_b$ with the maximum sum of $\boldsymbol{\rho}_b$. This criteria was chosen as the primal q-OCSVM objective function minimizes the negative sum of ρ_j terms (2.9) (line 18). Finally, $\boldsymbol{\rho}^*$ denotes the $\boldsymbol{\rho}_b$ with maximum sum.

Note that the number of quantiles that are calculated for output adds minimal computation to the algorithm as it requires a single line search. This minimal computational increase is contrasted with algorithms based on quadratic programming, where there is a significant increase in computation with increasing predicted quantiles (see Section 5.3.2).

3.4 Chapter Summary

In this Chapter, the SV-Means algorithm was formalized in Algorithm 1 with a detailed line-by-line description. SV-Means provides a one-class support vector machine-based level set estimator that provides a generative Bayesian formulation, accurate boundary definition, and fast training. These unique features allow the algorithm to be used to address several application areas that are described in the next Chapter.

Chapter 4

Applications

In this chapter, several applications are explored: anomaly detection, density estimation, open set classification, and clustering. Within each research area, current SVM-based algorithms are discussed and compared to the SV-Means algorithm described in Chapter 3. Typically these algorithms are characterized as supervised, semi-supervised, and unsupervised. Supervised algorithms build models from training data that is completely labeled. Definitions of semi-supervised algorithms are different across applications. In classification, semi-supervised algorithm models are built from a combination of labeled and unlabeled data. In semi-supervised anomaly detection, labeled data is used to model the normal class, and test data that does not fit the model is classified as anomalous. In this work, we adopt the anomaly detection definition of semi-supervised learning. Finally, unsupervised algorithms train with unlabled data. We focus on semi-supervised algorithms except for in Section 4.4 which concentrates on unsupervised algorithms for clustering.

4.1 Anomaly Detection

In anomaly detection, data points are categorized as anomalous (or outliers) if they belong to classes that were not seen in training. Semi-supervised anomaly detection algorithms use labeled training data to provide a model for the normal class. Testing data is declared

anomalous if it is unlike the trained model. Global and local anomalies are characterized with respect to the entire data set or direct neighborhood, respectfully [48]. In general, a local anomaly may not be detected when using a global technique. Anomalies are also characterized as point (data point is anomalous with respect to the rest of the data), contextual (data point is anomalous in a specific context, but not otherwise), or collective (group of similar data is anomalous with respect to the entire data set) [49]. Only point anomalies are considered in this paper.

In [49], anomaly detection algorithms are described as nearest neighbor-based ([50], [51], [52]), clustering-based ([53], [54], [55]), statistical ([56], [57], [58]), information theoretic ([59], [60], [61]), spectral ([62], [63], [64]), and classifier-based ([65], [28], [66], [67]). In this work, we are interested in classifier-based anomaly detection. In closed set classification, labeled training data is used to optimize a classifier or model and testing data is classified as one of the trained classes. Classification-based anomaly detection develops models for labeled data, and testing data is classified either as normal or as an anomaly. Anomaly detection classification formulations are either one-class or multi-class. In one-class frameworks, the training data is from one class label and any testing data outside the model boundary is considered an anomaly. In multi-class frameworks, the training data is from multiple normal class labels and there is a classifier for each class to distinguish between normal classes. Testing data is considered an anomaly if it is not classified as a normal class. Multi-classification anomaly detection is similar to open set classification, described in Section 4.3, except that anomaly detection is referred to as rejection.

Although there are several classification-based anomaly detection algorithms (neural network-based [65], Bayesian networks-based [66], rule-based [67]), we are interested in one-class support vector machine (OCSVM) based anomaly detection to compare to the SV-Means algorithm. In [28], an OCSVM, based off a linear formulation in [41], is used for anomaly detection where random features are used for training acceleration and as a way to handle high-dimensional data without maintaining large amounts of data. How-

ever, in our experimentation, it was found that random features did not provide accurate boundaries for high-dimensional data. SV-Means is able to handle high-dimensional data by optimizing over multiple density level sets to provide accurate boundaries for anomaly detection, and its performance is comparable to the classic one-class SVM formulation as shown in Section 5.3.3.

4.2 Density Estimation

In density estimation, there are several types of methods, e.g., parametric, semi-parametric, and non-parametric. Parametric methods assume the distribution of the data is known (e.g., a Gaussian distribution), but the parameters of the distribution are not known and, therefore, need to be estimated. Two examples of parameter estimation techniques are maximum likelihood estimation and Bayesian estimation [68]. Semi-parametric methods do not assume a specific model for the underlying distribution but assume the model comes from a parametric family of distributions. For example, a mixture model is a semi-parametric method and the parameters from the mixture model are estimated [69]. In non-parametric methods, the density is estimated directly from the data without assuming a particular form of the underlying distribution. Examples of non-parametric methods include histograms ([70], [71]), kernel density estimation ([72], [73]) and support vector machines ([74], [75]).

Histograms are popular for their ease of use as the data is split into intervals or bins, and the data within those bins are counted. However, histograms start to fall apart with high-dimensional data sets due to the large number of bins, the relatively small amount of data in high-dimensions, and the computational complexity [76].

Kernel density estimation is a technique where a kernel is placed on each data point and then the kernels are summed for the final density estimation solution. Kernel density estimation is widely used but is not effective in high dimensions [77]. To be more effective in high dimensions, density estimation is relaxed and a minimum volume set approximation

is computed [78]. minimum volume sets determine the minimum volume or closed set containing a specified probability mass. Since the volume is minimized, type 1 error is controlled at the specified level, and type 2 errors are minimized, making the resulting boundaries useful for applications such as anomaly detection [79].

Support vector machines (SVMs) are proven to be powerful level set estimators [80]. However, they do not provide multiple hierarchical level sets to characterize the distribution further. In [13], the q-OCSVM estimates multiple hierarchical level sets by optimizing over a single weight vector. With a single weight vector, the hierarchical level sets are simply a resultant of a change in bias.

The SV-Means algorithm is inspired by q-OCSVM density estimation as shown in Chapter 3. For density estimation, SV-Means borrows the idea of optimizing over a single weight vector from the q-OCSVM algorithm but is significantly different in how the optimization is solved. The q-OCSVM is solved using a kernel and quadratic programing whereas SV-Means uses random Fourier features and a stochastic gradient descent k-means technique. The SV-Means optimization technique is not only faster for large data sets and more hierarchical boundaries (SV-Means determines final hierarchical boundaries by a simple line search performed on the optimal weight vector), but also provides a more accurate outer boundary.

4.3 Open Set Classification

An open set classification problem is developed under the assumption that there is incomplete knowledge of all waveforms present at training time. In most machine learning algorithms, closed set assumptions are used when a set of target classes are trained and then the algorithm tested with these same classes. The closed set formulation poses an issue when testing a new sample (unseen in training), as it is classified as the most likely trained class. In radar waveform classification, it is highly likely to detect a signal not in the al-

gorithm library so the open set classification problem is a good fit to reject these unknown waveforms. In [6], Scheirer formalized the open set problem calling it, open set recognition (OSR), and several other algorithms followed under this definition. The openness of a classification problem is defined in [6] as

$$\text{openness} = 1 - \sqrt{\frac{2 \times |\text{training classes}|}{|\text{testing classes}| + |\text{target classes}|}} \tag{4.1}$$

where $|\cdot|$ represents the number of instances in that set. When the classification problem is closed, openness is zero.

The first algorithm derived under the open set definition is the one-vs-set machine [6] which uses the separating hyperplane provided by a binary SVM and adds an additional parallel hyperplane on the other side of the data. If a testing point lies in between the slab (the area between the two parallel hyperplanes), it is labeled as the known class, and if it lies outside the slab, it is rejected for that class. In [81], the Weibull-calibrated SVM (W-SVM) was developed which uses a two-stage algorithm. The first stage includes a OCSVM fitted with a Weibull cumulative distribution function to provide a posterior estimate for a testing point. The Weibull distribution choice is based on extreme value theory (EVT). If the posterior estimate is less than a specified threshold, the testing point is rejected in the first stage; and if it is greater, the testing point is passed to a one-vs-all SVM in the second stage. The probabilities calculated in the second stage are also based on EVT.

In [82], the PI-SVM algorithm is developed where PI is shorthand for the unnormalized probability of inclusion. The algorithm starts with a one-vs-all binary SVM and then followed with fitting EVT distributions from positive training samples. Using this methodology, they are able to calculate an unnormalized posterior probability estimate for each class for a testing point. In [83], the probabilistic open space SVM or POS-SVM is formulated. The POS-SVM is also based on a one-vs-all binary SVM where a validation set is used to optimize individual class thresholds using Platt's method for posterior probability

estimation.

Note that the W-SVM used an OCSVM as part of their formulation. OCSVMs are a natural fit for the open set problem as they provide models for just the in-class data. However, these open set algorithms have avoided exclusively using OCSVMs because they have cited their inability to provide good generalization and separation between known classes. To help distinguish between known classes, binary SVMs are used in open set algorithms. In the SV-Means algorithm, this issue is tackled by estimating OCSVMs with hierarchical level sets for each class. If a testing point falls within the innermost level set of Class A and the outermost level set of Class B, Class A is chosen. If a testing point falls in the same level set for Class A and Class B, a normalized distance metric is calculated to decide a final class. More details on the calculation of confidence scores for each classification decision are provided in the next Section.

4.3.1 SVM Confidence Scores

It was briefly discussed how each open set algorithm was formulated and, except for the one-vs-set machine, how their data scores were turned into probabilities used for thresholding their final decisions. There were two posterior estimation techniques discussed: Platt's method and extreme value theory (EVT). In Platt's method [33], sometimes called Platt's calibration or scaling, a sigmoid function is used to estimate posteriors from SVM data scores. This approach is used in LIBSVM [84], a popular SVM software package, as well as in the open set POS-SVM described above. EVT models the extreme values of a score distribution as a reverse Weibull if the data is bounded from above, and as a Weibull if the data is bounded from below [85]. EVT is used by the W-SVM and PI-SVM open set algorithms.

Another technique to calculate the posterior involves first generating the likelihood by performing density estimation for each known class. OCSVMs can be used for estimation of distributions or minimum volume sets (equivalently density level sets [86]). In [87], it

is shown that the OCSVM is a consistent estimator of density level sets, and the solution provides an estimate of the tail of the density. In traditional SVMs, the half space boundary is calculated by achieving the maximum margin to the nearest training points or support vectors. Using a small set of data to determine the boundary limits the effectiveness of OCSVMs in high dimensions, as larger data sets more effectively span this space [19].

In [13], the q-OCSVM estimates multiple level sets to represent the distribution underlying the OCSVM. The q-OCSVM has an important property which includes estimating density level sets by optimizing over a single weight vector. This allows the density level set boundaries to be parallel to each other as they are only different with respect to their bias terms. The SV-Means algorithm extends the q-OCSVM by formulating this density estimation problem into a classification technique. Traditional SVMs are discriminative classifiers, but SV-Means is a generative classifier as it provides a non-parametric probability estimate for each test sample according to which density level set contains the sample. The SV-Means algorithm borrows the idea of estimating density level sets using a single weight vector from the q-OCSVM, but the weight vector is estimated based on the importance of classification rejection by estimating level sets near the extrema of the data set, which provides a more accurate boundary. Other density estimation techniques are described in [43] and [23].

The improved boundaries given by the SV-Means algorithm are also adaptable as the prior information changes. In discriminative algorithms, the posterior is calculated directly, and in generative algorithms, like SV-Means, the posterior is calculated via Bayes Rule. The calculated posteriors, however, are unnormalized as not all the classes are known at training. The unnormalized posterior is popular in computer vision and is also used in the PI-SVM algorithm ([88], [82]). A generative algorithm is more suitable for radar waveform classification since the prior probabilities of the known classes can change rapidly during a mission and the Bayesian formulation allows for adapting decision boundaries on-the-fly. Priors are normally set with domain knowledge, but in the case of the experiments in this

dissertation, the priors are equal.

4.4 Clustering

Clustering separates data into similar groups by following a set of criteria. Several popular clustering algorithms are described as hierarchical ([89], [90], [91]), centroid-based [92], and density estimation-based [93]. One clustering algorithm of recent interest is a non-parametric method, support vector clustering (SVC), which has the ability to generate cluster boundaries with arbitrary shapes [14]. SVC is a multi-resolutional approach which uses a hypershere-based support vector machine by varying C (helps controls margin width), Gaussian kernel width, and a constraint on the number of support vectors. Using this approach, clusters are formed with the enclosed boundaries. However, cluster membership cannot be determined from the boundaries alone. To determine cluster membership, a geometric approach is used where the line between two points is tested. If the points along the line fall outside of a boundary, the two points must belong in two different clusters. The lines between each pair of points are defined by an adjacency matrix which induces a graph. Instead of testing every pair of points, the overall testing is reduced to testing adjacencies with support vectors.

As described, the SVC method contains two main processes: (1) finding the enclosed boundaries and (2) assigning cluster membership. Several studies have been performed to improve how the cluster membership algorithm is solved as it is a computationally expensive process. Cluster membership improvement methods include: proximity graphs using a Delaunay diagram, minimum spanning trees (MST), or k-nearest neighbor ([94], [95], [96]); cell growing [97]; and cone cluster labeling [98].

All of these techniques use the approach in the seminal paper [14] to form the initial cluster boundaries, but as discussed, kernel based methods solved by quadratic programming do not scale well with large data sets. A few papers tackle ways to handle the

complexity of the support vector machine boundaries including chunking [99] and using ensembles [100]. In [101], the initial cluster boundary step is replaced by a hyperplane method based on a primal stochastic gradient descent (SGD) framework. Their use of SGD certainly speeds up the optimization, but the algorithm is still using a kernel which grows in computational complexity with the number of data samples.

4.4.1 SV-Means Level Set Clustering

We develop a novel clustering method using the SV-Means algorithm (Algorithm 1) for replacing the initial step of finding the enclosed boundaries. The new clustering algorithm, SV-Means Level Set Clustering, is shown in Algorithm 2. In the SVC algorithm [14], two parameters must be optimized in order to find the multi-resolutional levels to solit for closely spaced clusters. In the SV-Means Level Set Clustering algorithm, a similar effect is achieved by examining clusters at different level sets with the added benefit that these level set boundaries are hierarchical and have a probabilistic interpretation. With SVC, the boundaries have no constraints and, therefore, are arbitrary. In [101], a primal formulation with stochastic gradient descent principles is employed, but the use of the kernel requires the use of a budget algorithm to handle the size of the data. The SV-Means Level Set Clustering algorithm, in contrast, uses random features to approximate the kernel and can therefore handle large data set sizes.

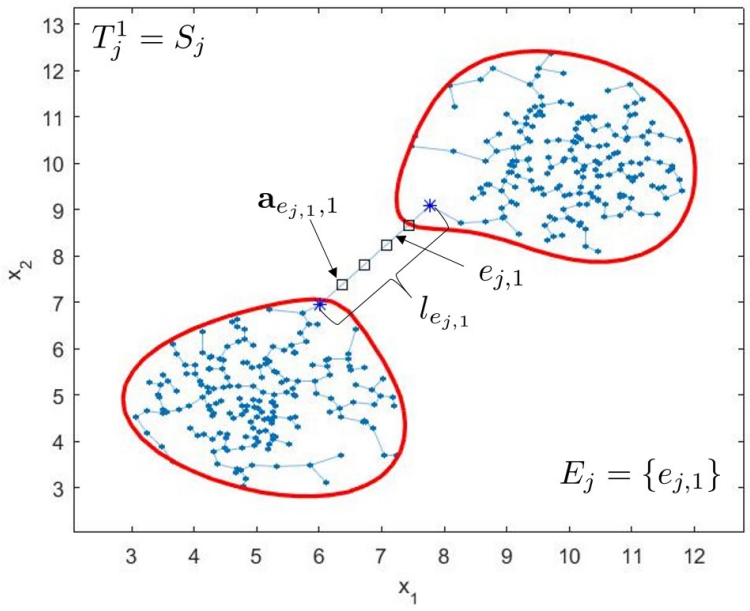
In the SV-Means Level Set Clustering Algorithm in Algorithm 2, level set clustering is accomplished by performing the SV-Means algorithm with $\boldsymbol{\nu}^o = [\nu_1^o, \ldots, \nu_q^o]$, which will return an optimum $\mathbf{w}$ and $\boldsymbol{\rho} = [\rho_1, \ldots, \rho_q]$ with $j = 1 \ldots q$ (line 2). Cluster membership is determined for each level set by using only the set of points Z_j^+ corresponding to that level set (line 5). A minimum spanning tree S_j is performed on X_j^+ (the points corresponding to the transformed points Z_j^+) and the average $\hat{l}_j$, and standard deviation σ_j, of all edge lengths are calculated. The set of longest edges, E_j, is determined by testing if each edge's length is greater than $(\hat{l}_j + \sigma_j)$ (line 9) [102]. In a similar fashion to the SVC algorithm,

r_{max} equally spaced testing points, $\mathbf{a}_{(e_{j,u},r)}$, along the edges in E_j, are examined (line 12). If one of the testing points along the edge is outside the boundary, the initial pair of nodes forming that edge are considered to be in different classes and the edge is removed (line 15). A set $T_j^{n_t}$ of disjoint subtrees S_j^v are created by removing the edge (line 17). The number of clusters k_j are determined by the cardinality of the set $T_j^{n_t}$. The labels $\mathbf{y}_j$ for nodes in $T_j^{n_t}$ (points in X_j^+) are determined via depth first search [103]. The output of the SV-Means Level Set Clustering algorithm is the number of clusters k_j, and labels $\mathbf{y}_j$, for each level set. A visual representation of some of the clustering terminology is provided in Figure 4.1.

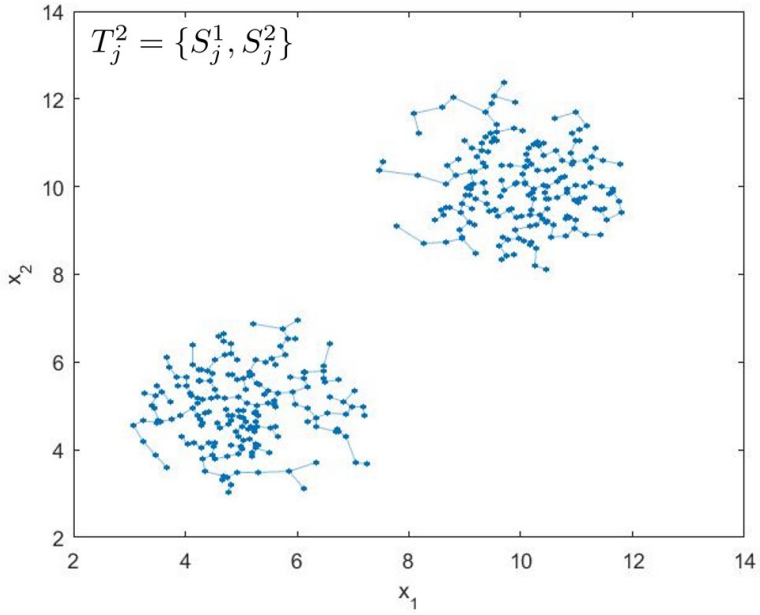
The SV-Means Level Set Clustering algorithm shows promising results as shown in Section 5.4 but it has not been fully tested and characterized. However, the algorithm provides a baseline for future extensions of this work.

4.5 Chapter Summary

In this chapter, SVM-based anomaly detection, density estimation, open set classification, and clustering techniques were described. The SV-Means algorithm is a viable candidate for all of these applications as it provides an accurate boundary, a confidence score, and a generative framework. Experimentation comparing the SV-Means algorithm against existing techniques in each application area is shown in Sections 5.3.3-5.3.5, and an example demonstrating the clustering algorithm is shown in Section 5.4.



(a)



(b)

Figure 4.1: Pictured in Figure 4.1a is the SV-Means boundary for level set j in red, the minimum spanning tree S_j constructed from X_j^+ in blue, an example of an edge $e_{j,1}$, the edge length $l_{e_{j,1}}$, the set of edges of longest length E_j, and a set of equally spaced testing points $A_{e_{j,1}} = \{\mathbf{a}_{e_{j,1},1}, \mathbf{a}_{e_{j,1},2}, \mathbf{a}_{e_{j,1},3}, \mathbf{a}_{e_{j,1},4}\}$ ($r_{max} = 4$) along $e_{j,1}$ represented by black squares. Each point along $e_{j,1}$ is tested to see if it falls outside the SV-Means boundary. If any of the points are outside of the boundary, the edge is removed. Pictured in Figure 4.1b shows the removal of edge $e_{j,1}$. The original minimum spanning tree S_j is now split into two subtrees, $T_j^2 = \{S_j^1, S_j^2\}$.

Algorithm 2: SV-Means Level Set Clustering

Step One: Estimate a support function

$Z = z(X)$, Transform X via (2.15)

Perform SV-Means in Algorithm 1 on Z with $b_{max} = 10$, $m_{max} = 20$, $\nu^t_{min} = 0$, $\nu^t_{max} = 0.05$, $\boldsymbol{\nu}^o = [\nu^o_1, \ldots, \nu^o_q]$, **return** $\mathbf{w}$, $\boldsymbol{\rho} = [\rho_1, \ldots, \rho_q]$ where $j = 1, \ldots, q$

Step Two: Assign cluster membership at each level set

Let X^+_j be the points corresponding to Z^+_j from Z in level set j

Let S_j be the minimum spanning tree constructed from X^+_j

Let $T^{n_t}_j$ be the set of disjoint subtrees $S^v_j \in S_j$ where $v = 1, \ldots, n_t$ and n_t is the number of subtrees

Let $e_{j,u}$ be an edge in S_j where $u = 1, \ldots, u_{max}$ and u_{max} is the number of edges

Let $l_{e_{j,u}}$ be the length of $e_{j,u}$

Let $\hat{l}_j$ be the mean of the edge lengths in S_j

Let σ_j be the standard deviation of the edge lengths in S_j

Let E_j be the set of edges in S_j of longest length

Let $A_{e_{j,u}}$ be a set of equally spaced points $\mathbf{a}_{(e_{j,u},r)}$ along $e_{j,u}$ where $r = 1, \ldots, r_{max}$ and r_{max} is the number of points

Initialize: $n_t = 1$, r_{max}

for $j = 1 : q$ **do** *Estimate clusters for each level set*

  Find X^+_j, the points corresponding to $Z^+_j = \{\mathbf{z}_i \in Z : \mathbf{w}^\intercal \mathbf{z}_i - \rho_j > 0,\ i = 1, \ldots, n\}$

  Form minimum spanning tree S_j from X^+_j via [104]

  $\hat{l}_j = \frac{1}{u_{max}} \sum_{u=1}^{u_{max}} l_{e_{j,u}}$

  $\sigma_j = \sqrt{\frac{1}{u_{max}} \sum_{u=1}^{u_{max}} (l_{e_{j,u}} - \hat{l}_j)^2}$

  $E_j = \{e_{j,u} \in S_j : l_{e_{j,u}} > \hat{l}_j + \sigma_j,\ u = 1, \ldots, u_{max}\}$

  $T^{n_c}_j = S_j$

  foreach $e_{j,u} \in E_j$ **do**

    Determine set $A_{e_{j,u}}$ of equally spaced points $\mathbf{a}_{(e_{j,u},r)}$ along $e_{j,u}$

    foreach $\mathbf{a}_{(e_{j,u},r)} \in A_{e_{j,u}}$ **do**

      if $\mathbf{w}^\intercal z(\mathbf{a}_{(e_{j,u},r)}) - \rho_j < 0$ **then**

        Remove $e_{j,u}$ from S_j

        $n_t = n_t + 1$

        $T^{n_t}_j = \cup_{v=1}^{n_t} S^v_j$

        break

      end

    end

  end

  Output: $T^{n_t}_j = \{S^1_j, \ldots, S^{n_t}_j\}$ for level set j

end

Output: $T^{n_t} = \{S^1, \ldots, S^{n_t}\}$ for every level set

$k_j = |T^{n_t}_j|$, number of clusters for level set j

Determine labels $\mathbf{y}_j$ for nodes in $T^{n_t}_j$ (points in X^+_j) via depth first search [103]

Chapter 5

Experimental Results

In this Chapter, several experiments were performed to test the capability and flexibility of the novel SV-Means algorithm. Before the results are shown, a Section is dedicated to detailing the data sets used for each experiment. The first experiment includes a brief visual representation of the convergence properties and quantile estimation. The second set of experiments includes an extensive comparison of the SV-Means algorithm to the primal q-OCSVM, dual q-OCSVM, anomaly detection algorithms, density estimation algorithms, and open set classification algorithms. The third experiment includes a visual representation of the SV-Means Level Set Clustering algorithm by clustering at two level sets. The fourth experiment includes an end-to-end demonstration from training, to testing, to clustering, to adding a new class to the library using the SV-Means algorithm.

5.1 Description of Data Sets

A variety of data sets were used to test the SV-Means algorithm against the primal and dual q-OCSVM (with classification capability) along with anomaly detection, density estimation, and open set algorithms. A few two-dimensional (2-D) data sets were used to visualize SV-Means convergence, SV-Means density estimation, SV-Means clustering, and an end-to-end demonstration.

5.1.1 2-D Data Sets

For the convergence and quantile estimation experiments in Section 5.2, two common 2-D data sets ([105], [43]) were tested using the SV-Means algorithm: double moon and double Gaussian. Both feature sets have $n = 1000$ training examples and $d = 2$ features. For the clustering and end-to-end demonstration, three separate Gaussians were used for each of the 3 classes with $n = 1000$ and $d = 2$ for each class.

5.1.2 UCI Machine Learning Repository Data Sets

Several data sets from the University of California Irvine (UCI) machine learning repository [106] are used for testing. For anomaly detection results, the following 24 data sets were used: abalone, balance scale, blood transfusion, breast cancer wisconsin, cmc, ecoli, glass, haberman, hayes-roth, ionosphere, iris, letter recognition, libras, magic, page blocks, parkinsons, pima indians diabetes, poker, spambase, tae, tic-tac-toe, wine, yeast, and zoo. For the density estimation results, the following 15 data sets were used: abalone, balance scale, blood transfusion, breast cancer wisconsin, cmc, haberman, ionosphere, letter recognition, magic, page blocks, pima indians diabetes, poker, spambase, tic-tac-toe, and yeast. For the open set results, the letter recognition data set was used.

5.1.3 Phase-modulated Radar Waveform Data Set

The phase-modulated radar waveform data in [107] and [22] was also used. The training data, X, described in Table 5.1, was generated for every combination of class (c), number of pulses averaged (p), and SNR range (Θ) using the ACF-based features formulated in [107]. The autocorrelation function (ACF)-based feature set is used to handle nuisance parameters (e.g., waveform alignment, varying pulse widths, unknown amplitudes, and SNR). The feature set is found via the autocorrelation of the pulse, followed by a Fourier transform of the log intensity. This feature set is similar to Mel frequency cepstral coefficients in the

speech recognition community. Exact details of the feature set are given in [107], which also describes how averaging over multiple pulses improves performance.

Table 5.1: Phase modulation types, training pulse width ranges, and testing pulse widths of the phase-modulated radar waveform data set.

c	Modulation Type	Code Length	Training τ (μsec)	Testing τ (μsec)
1	Barker	7	[0.875, 7]	1.75
2	Barker	11	[1.375, 11]	2.75
3	Barker	13	[1.625, 13]	3.25
4	Combined Barker	16	[1, 8]	2
5	Combined Barker	49	[3.08, 21.1]	6.13
6	Combined Barker	169	[10.58, 84.6]	21.1
7	Max. Length Pseudo Random	15	[1, 4.5]	1.5
8	Max. Length Pseudo Random	31	[0.235, 10.5]	1.5
9	Max. Length Pseudo Random	63	[4.221, 18.9]	6.3
10	Min. Peak Sidelobe	7	[1.05, 4.2]	1.4
11	Min. Peak Sidelobe	25	[1.25, 10]	2.5
12	Min. Peak Sidelobe	48	[2.4, 19.2]	4.8
13	T1	NA	[2, 16]	4
14	T2	NA	[1.5, 12]	3
15	T3	NA	[1, 8]	2
16	Polyphase Barker	7	[0.875, 7]	1.75
17	Polyphase Barker	20	[1, 8]	2
18	Polyphase Barker	40	[2, 16]	4
19	P1	NA	[5, 20]	10
20	P2	NA	[3.2, 25.6]	6.4
21	P3	NA	[3.2, 25.6]	6.4
22	P4	NA	[5, 20]	10
23	Minimum Shift Key	63	[2, 18.9]	4

For the following experiments, the training data (Table 5.1) consisted of $c = 23$ different classes with SNR ranging from $\Theta \in \{[-12, 12]\}$, number of pulses averaged $p = 20$, number of data points $n = 11000$, and number of features $d = 10$. FOr our experiments, the pulse widths were uniformly sampled from the intervals given in the fifth column of Table 5.1.

5.1.4 Data Preliminaries

For the primal q-OCSVM and SV-Means, the data sets described in Section 5.1 are first normalized and then transformed from d features to d_{RF} random features using $z(X)$ where $d_{RF} = 2000$ (2.15). The transformation includes selecting σ to define the Gaussian distribution. The σ parameter is calculated via quantiles of random distances in the training data, which is a technique used in [108].

5.2 SV-Means Convergence and Quantile Estimation

In this section, the convergence and quantile estimation of the SV-Means algorithm is explored via illustrative examples. The 2-D double moon and double Gaussian data set described in Section 5.1.1 are used to facilitate the visualization, and hence, understanding of the boundaries and quantiles. The double moon data set is shown in Figure 5.1 where $\nu_{min}^t = 0$ and $\nu_{max} = 1$ for density estimation, and $\nu^o = 0.05$, which captures 95% of the data. Note that all $b_{max} = 10$ initializations of the SV-Means algorithm are shown both on the convergence plot and on the two-dimensional boundary plot. As seen in Figure 5.1b, although all 10 initializations converged, the best initialization clearly defined a better boundary (in red) than the other initializations which often connected the two moons. This also confirms that the criteria to pick a final $\mathbf{w}^*$ (i.e., choose the $\mathbf{w}_b$ whose corresponding $\boldsymbol{\rho}_b$ has the minimum sum) is effective. This example also illustrates the non-convex nature of the SV-Means algorithm, motivating the need for the multiple random initializations, and also demonstrating that $m_{max} = 20$ inner loop iterations are enough for SV-Means to converge as shown in Figure 5.1a. The same number of outer and inner loop iterations are used for all experiments performed.

The second data set used is the double Gaussian, shown in Figure 5.2, where $\boldsymbol{\nu}^o = [0.8,\ 0.6,\ 0.4,\ 0.2,\ 0.05,\ 0]$. Both Figure 5.1 and 5.2 provide qualitative insight into the nature of the SV-Means quantile estimation properties.

5.3 SV-Means Algorithm Comparison Results

In this Section, the SV-Means algorithm is compared to the primal and dual form of the q-OCSVM extended for classification. The primal experiment directly compares SV-Means with the primal form of the q-OCSVM using random Fourier features for kernel estimation. The dual experiment tests the legitimacy of substituting a kernel estimation technique by comparing SV-Means to the dual form of the q-OCSVM using a traditional Gaussian kernel. Finally SV-Means is compared to algorithms within each application area: anomaly detection, density estimation, and open set classification.

5.3.1 SV-Means vs. Primal q-OCSVM

The SV-Means algorithm performance is compared to the primal q-OCSVM algorithm also using the radar waveform data described in Section 5.1.3. Three versions of the primal q-OCSVM (with added classification capability) are trained and are distinguished by the ν^o values used. The SV-Means algorithm is trained to accurately estimate the OCSVM boundary for better rejection capability with $\nu^t_{min} = 0$ and $\nu^t_{max} = 0.05$. The ν-parameters used for each experiment are shown in Table 5.2. The table highlights that the SV-Means algorithm uses different values of $\boldsymbol{\nu}^t$ to train than values of $\boldsymbol{\nu}^o$ used to compute the final or output $\boldsymbol{\rho}$ by performing a line search on the optimal $\mathbf{w}$.

The test data for this experiment used the pulse widths given in the fifth column of Table 5.1. The algorithms are tested at three SNRs: 0, 4, and 10 dB. To use the SV-Means and primal OCSVM as classifiers, the test data must be transformed with the same random Fourier feature transform, $z(\cdot)$, used in training. Each individual classifier consists of $q = 2, 5$ or 6 hierarchical quantiles that provide a natural probability distribution. For example, for $q > 1$, if a test point falls within the first quantile (the innermost quantile), the likelihood that the test point belongs to that class is high. The class chosen is either the unknown class, which is outside the boundaries for all classifiers, or the class that provides

Table 5.2: Summary of ν-parameters for the primal q-OCSVM and SV-Means for training ($\boldsymbol{\nu}^t$), output generation ($\boldsymbol{\nu}^o$), and testing thresholds (ν_q).

Algorithm	Train, $\boldsymbol{\nu}^t$	Output, $\boldsymbol{\nu}^o$	Test, ν_q
Primal q-OCSVM 2 ν^o	[0.05, 0]	[0.05, 0]	0.05
			0
Primal q-OCSVM 5 ν^o	[0.8, 0.6, 0.4, 0.2, 0.05]	[0.8, 0.6, 0.4, 0.2, 0.05]	0.05
	[0.8, 0.6, 0.4, 0.2, 0]	[0.8, 0.6, 0.4, 0.2, 0]	0
SV-Means	$\sim U(0, 0.05)$	[0.8, 0.6, 0.4, 0.2, 0.05, 0]	0.05
			0

the highest unnormalized posterior score. The likelihood score is computed at the quantile level unless the signal lies in the same quantile for two or more classes. In such a case, a normalized distance metric is computed to resolve the "ties" at the quantile level. The normalized distance metric is computed as follows:

$$\delta(x') = \frac{g_{G_j}(\mathbf{x}_{sv_j}) - g_{G_j}(\mathbf{x}')}{g_{G_j}(\mathbf{x}_{sv_j}) - g_{G_{j-1}}(\mathbf{x}_{sv_{j-1}})} \tag{5.1}$$

where $g_{G_j}(\cdot) = z(\mathbf{x}_i)\mathbf{w} - \rho_j$, and $\mathbf{x}_{sv_j}$ is the point corresponding to the support vector on the jth quantile boundary. This metric effectively interpolates within a quantile region to more accurately compare distance values. The class with the smallest normalized distance is chosen.

The testing results for each algorithm, $\boldsymbol{\nu}^o$ variation, and testing SNR are generated with 23 different classes determined by rotating the unknown class by leaving one class out for each experiment. The probability of correct classification for the 24 class problem (23 known and 1 unknown) at 0 dB, 4 dB, and 10 dB for the testing data set are reported in Table 5.3 for all algorithms. The probability of correct classification of all 24 classes and of just the unknown class are reported for both testing thresholds of $\nu_q = 0.05$ and $\nu_q = 0$. The result tables show that the SV-Means algorithm achieves similar and sometimes

better results than the primal q-OCSVM quadratic programming formulations. While each primal q-OCSVM excels in either overall classification (denoted 24 classes in Table 5.3) or classification rejection (denoted unknown class in Table 5.3), SV-Means excels in both.

As mentioned before in this Section, there were a variety of sets of ν^o values used. Initially, $\boldsymbol{\nu}^o$ was chosen to only contain ν^o values that spanned the whole distribution. It was then discovered, in the case of the primal q-OCSVM where $\boldsymbol{\nu}^o = [0.8,\ 0.6,\ 0.4,\ 0.2,\ 0]$ (denoted "Primal q-OCSVM - 5 ν^o and $\nu_q = 0$" in Table 5.3), that the unknown class performance drops. This result motivated additional testing to more accurately model the boundary. Therefore, the set $\boldsymbol{\nu}^o = [0.05,\ 0]$ (denoted "Primal q-OCSVM - 2 ν^o" in Table 5.3) was considered and this improved the probability of correct classification for the unknown class. In addition, the algorithms used two different testing thresholds corresponding to $\nu_q = 0.05$ and $\nu_q = 0$ in order to highlight the importance of where to draw the threshold to balance overall classification performance versus unknown class sensitivity.

Table 5.3: Probability of correct classification for all 24 classes (23 known and 1 unknown) and for just the unknown class with two boundary conditions $\nu_q = 0.05$ and $\nu_q = 0$ and 1 SNR range, Testing at 0, 4, and 10 dB

0dB SNR	24 classes		Unknown class	
Algorithm	ν_q = 0.05	ν_q = 0	ν_q = 0.05	ν_q = 0
Primal q-OCSVM - 2 ν^o	0.8831	0.8836	0.9067	0.7469
Primal q-OCSVM - 5 ν^o	0.9453	0.9764	0.8987	0.6385
SV-Means	0.9452	0.9735	0.9119	0.7269

4 dB SNR	24 classes		Unknown class	
Algorithm	ν_q = 0.05	ν_q = 0	ν_q = 0.05	ν_q = 0
Primal q-OCSVM - 2 ν^o	0.9076	0.9026	0.9306	0.7882
Primal q-OCSVM - 5 ν^o	0.9429	0.9812	0.9055	0.6977
SV-Means	0.9605	0.9845	0.9409	0.7725

10 dB SNR	24 classes		Unknown class	
Algorithm	ν_q = 0.05	ν_q = 0	ν_q = 0.05	ν_q = 0
Primal q-OCSVM - 2 ν^o	0.9283	0.9562	0.9834	0.8547
Primal q-OCSVM - 5 ν^o	0.9092	0.9832	0.9660	0.7580
SV-Means	0.9426	0.9898	0.9970	0.8540

5.3.2 SV-Means vs. Dual q-OCSVM

In the previous Section, it was shown that the primal q-OCSVM and SV-Means were comparable. However, both of these algorithm formulations use random Fourier features as an approximation to the kernel. Testing was performed to compare the primal q-OCSVM and SV-Means algorithms to the dual q-OCSVM which uses a traditional kernel. In [22], due to the complexity of the dual q-OCSVM algorithm, the data was broken up into smaller over-lapping chunks (i.e., small SNR ranges) for training, and the same strategy is used here. The phase-modulated radar waveform parameters described in Section 5.1.3 are used, but now the data is broken up into 11 over-lapping SNR ranges: $\Theta \in \{[-12,-8]$, $[-10,-6]$, $[-8,-4]$, $[-6,-2]$, $[-4,0]$, $[-2,2]$, $[0,4]$, $[2,6]$, $[4,8]$, $[6,10]$, $[8,12]\}$ with $n = 1000$ instances of each class in each SNR range.

Two versions of the dual q-OCSVM are trained with the 11 SNR ranges and are described in Table 5.4. The primal q-OCSVM and SV-Means algorithm use the same $\boldsymbol{\nu}^o$ variations as the previous experimentation given a single SNR range (described in Table 5.2). The same testing data and leave-one-out test strategy is used. Again, the probability of correct classification for the 24 class problem (23 known and 1 unknown) at 0 dB, 4 dB, and 10 dB for the testing data set is reported in Table 5.5 for all algorithms. These results show that using random Fourier features is a powerful substitute to a traditional kernel, as the performance results between the primal q-OCSVM, SV-Means, and the dual q-OCSVM are comparable.

Table 5.4: Summary of ν-parameters for the dual q-OCSVM and SV-Means for training ($\boldsymbol{\nu}^t$), output ($\boldsymbol{\nu}^o$), and testing thresholds (ν_q).

Algorithm	Train, $\boldsymbol{\nu}^t$	Output, $\boldsymbol{\nu}^o$	Test, ν_q
Dual q-OCSVM 5 ν^o	[0.8, 0.6, 0.4, 0.2, 0.05]	[0.8, 0.6, 0.4, 0.2, 0.05]	0.05
	[0.8, 0.6, 0.4, 0.2, 0]	[0.8, 0.6, 0.4, 0.2, 0]	0

The 3 algorithms (SV-Means, Primal q-OCSVM, Dual q-OCSVM) are timed for sev-

Table 5.5: Probability of correct classification for all 24 classes (23 known and 1 unknown) and for just the unknown class, with two boundary conditions, $\nu = 0.05$ and $\nu = 0$, and 11 SNR ranges, and testing at 0, 4, and 10 dB SNR

0 dB SNR	24 classes		Unknown class	
Algorithm	$\nu_q = 0.05$	$\nu_q = 0$	$\nu_q = 0.05$	$\nu_q = 0$
Dual q-OCSVM - 5 ν^o	0.9164	0.9685	0.8984	0.7994
Primal q-OCSVM - 2 ν^o	0.8643	0.8932	0.8969	0.8520
Primal q-OCSVM - 5 ν^o	0.9186	0.9706	0.8800	0.6609
SV-Means	0.9356	0.9713	0.8886	0.8257

4 dB SNR	24 classes		Unknown class	
Algorithm	$\nu_q = 0.05$	$\nu_q = 0$	$\nu_q = 0.05$	$\nu_q = 0$
Dual q-OCSVM - 5 ν^o	0.9295	0.9740	0.9248	0.8287
Primal q-OCSVM - 2 ν^o	0.8950	0.9092	0.9322	0.8962
Primal q-OCSVM - 5 ν^o	0.9315	0.9781	0.9041	0.6972
SV-Means	0.9528	0.9792	0.9408	0.8686

10 dB SNR	24 classes		Unknown class	
Algorithm	$\nu_q = 0.05$	$\nu_q = 0$	$\nu_q = 0.05$	$\nu_q = 0$
Dual q-OCSVM - 5 ν^o	0.9440	0.9779	0.9746	0.8734
Primal q-OCSVM - 2 ν^o	0.9276	0.9487	0.9693	0.9501
Primal q-OCSVM - 5 ν^o	0.9377	0.9816	0.9441	0.7537
SV-Means	0.9664	0.9866	0.9697	0.9276

eral different groupings and values of ν^o on one class of $n = 11000$ for $|\Theta|$ = 1, and $n = 1000$ for $|\Theta|$ = 11. For the primal formulations, the single class had size $d_{RF} \times n$ and the dual formulations had size $d \times n$. The timing was measured using an Alienware (32 GB RAM, i7-7700HQ CPU@2.8 GHz) laptop with MATLAB 2017a and is recorded in Table 5.6 in seconds. It is shown that SV-Means is faster by two orders of magnitude when estimating multiple density level sets and for large training set sizes. The primal q-OCSVM timing using five ν^o values is shown for the larger SNR range, but the dual q-OCSVM is only shown up to two ν^o values, which illustrates the timing complexity for large amounts of data.

Table 5.6: Timing comparison of q-OCSVM variants (in seconds)

$\|\Theta\|$	SV-Means b_{max} = 10 m_{max} = 20	Primal q-OCSVM 2 ν^o	Primal q-OCSVM 5 ν^o	Dual q-OCSVM 1 ν^o	Dual q-OCSVM 2 ν^o
11	0.33	28.24	92.94	0.55	3.28
1	2.93	402.98	1674.07	738.10	3389.67

These experiments demonstrate that the SV-Means algorithm is able to extend the q-OCSVM density estimation algorithm in a powerful classification formulation with the following properties: distribution estimation for posterior calculation and distinction between known classes; rejection capability; and reduced computation speed. These illustrative experiments lead into a comparison of the SV-Means algorithm against other open set algorithms.

5.3.3 SV-Means vs. Anomaly Detection Algorithms

In this Section, the SV-Means algorithm is compared to the one-class support vector machine for anomaly detection. For experimentation, the 24 UCI data sets described in Section 5.1.2 are used as well as the radar waveform data described in Section 5.1.3. For each UCI data set, the class with the most data samples is chosen as the normal class (followed from [2]). For the radar waveform data, Class 8 from Table 5.1 is chosen as the normal class.

The remaining classes in each data set are used as anomalous data points. The data protocol for these experiments is to use 75% of the normal class randomly selected as training data and the remaining 25% is used for testing along with the defined anomalous classes. The SV-Means algorithm and the one-class support vector machine are compared to an upper bound (UB), which is a best case scenario if all of the data was known with a binary support vector machine. The upper bound provides intuition into how difficult the data is to separate.

In Table 5.7, the average of 25 AUC (area under receiver operating characteristic (ROC) curve) results are reported for each algorithm and for three different γ's. The first γ used is $\frac{1}{d}$ where d is the number of features, which is the default setting for the popular LIBSVM support vector machine solver [84]. The second and third γ are calculated via [108] (described in Section 5.1.4) using fine and course gamma settings, respectively. Several gammas are used to show the importance of gamma selection. The SV-Means algorithm is trained for better rejection capability with $\nu^t_{min} = 0$, $\nu^t_{max} = 0.05$, and $\nu^o = 0$.

Table 5.7: Average of 25 AUC results on UCI and radar waveform data using SV-Means and OCSVM. The data sets were modified according to [2] using the protocol where 75% of the normal class was randomly selected for training, and the anomalous classes combined with the remaining 25% of the normal class is used for testing. Each of the 25 tests consisted of a different random selection from the normal class.

Data Sets	**Examples**	**Features**	**UB**	**OCSVM**	**SV-Means**	γ
abalone	4,177	8	0.5426	0.6462	0.6178	0.1111
			0.6472	0.6496	0.6411	0.5780
			0.5900	0.6457	0.6386	0.03722
balance-scale	625	4	1	0.8303	0.8608	0.2500
			0.9998	0.8324	0.8366	0.3622
			0.9988	0.8268	0.9417	0.1073
blood-transfusion	748	4	0.7964	0.4990	0.5059	0.2500
			0.7901	0.4981	0.5603	0.0614
			0.7762	0.4975	0.5448	0.0094

Table 5.7 Continued

Data Sets	Examples	Features	UB	OCSVM	SV-Means	γ
breast-cancer-wisconsin	569	31	0.9989	0.9641	0.8442	0.0323
			0.9998	0.9613	0.9679	0.5878
			0.9998	0.9632	0.9627	0.2247
cmc	1,473	9	0.7469	0.4506	0.4468	0.1111
			0.7934	0.4361	0.4939	0.8750
			0.7742	0.4464	0.4710	0.2881
ecoli	336	7	1	0.9740	0.9715	0.1429
			1	0.9760	0.9717	0.1282
			1	0.9759	0.9638	0.0405
glass	214	7	0.8421	0.5721	0.6680	0.1111
			0.8684	0.5788	0.6471	0.2410
			0.8266	0.5681	0.6672	0.0304
haberman	306	3	0.8246	0.5427	0.4420	0.3333
			0.8183	0.5399	0.4648	0.1210
			0.7938	0.5384	0.5763	0.0217
hayes-roth	132	4	0.9731	0.8818	0.9028	0.2500
			0.9615	0.8886	0.8915	0.4644
			0.9731	0.8754	0.8949	0.1656
ionosphere	351	33	0.9798	0.9175	0.8572	0.0303
			1	0.9750	0.9562	1.3664
			0.9971	0.9384	0.9748	0.1834
iris	150	4	1	0.9804	0.9854	0.2500
			1	0.9802	0.9829	0.1644
			0.9967	0.9795	0.9815	0.0418
letter-recognition	20,000	16	0.9857	0.8712	0.9888	0.0625
			0.9994	0.9105	0.9035	0.6384
			0.9929	0.8778	0.9708	0.1510

Table 5.7 Continued

Data Sets	Examples	Features	UB	OCSVM	SV-Means	γ
libras	360	90	1	0.7530	0.5427	0.0111
			0.9990	0.8747	0.7695	7.2724
			1	0.9292	0.8776	2.0438
magic	19,020	10	0.8833	0.7036	0.6706	0.1000
			0.8920	0.7049	0.6290	0.1440
			0.8538	0.7016	0.7859	0.03603
page-blocks	5,473	10	0.9728	0.9190	0.9432	0.1000
			0.9589	0.9190	0.9520	0.0546
			0.9386	0.9190	0.9380	0.0090
parkinsons	195	22	0.9752	0.7510	0.7406	0.0455
			0.9820	0.7231	0.7569	0.6016
			0.9797	0.7347	0.7662	0.2194
pima-indians-diabetes	768	8	0.8835	0.6888	0.6637	0.1250
			0.8837	0.6896	0.6480	0.2099
			0.8807	0.6881	0.6660	0.0606
poker	25,010	10	0.6363	0.4978	0.5108	0.1000
			0.6480	0.5039	0.4989	1.1555
			0.6400	0.5007	0.5000	0.6451
radar-waveform	253,000	20	0.9976	0.9843	0.9263	0.0500
			0.9982	0.9822	0.9787	0.2022
			0.9977	0.9840	0.9540	0.0382
spambase	4,601	57	0.9497	0.7186	0.7620	0.0175
			0.9639	0.7177	0.7077	0.1587
			0.9533	0.7184	0.7424	0.0424
tae	151	5	0.8846	0.3681	0.5411	0.2000
			0.8677	0.4297	0.4412	0.6721
			0.8692	0.3587	0.5613	0.1062

Table 5.7 Continued

Data Sets	Examples	Features	UB	OCSVM	SV-Means	γ
tic-tac-toe	958	9	0.9991	0.6722	0.5056	0.0556
			1	1	0.7508	4
			1	1	0.7924	2.1649
wine	178	13	1	0.9316	0.9335	0.0769
			1	0.9352	0.9185	0.5165
			1	0.9329	0.9263	0.2329
yeast	1,484	8	0.7916	0.6878	0.6606	0.1250
			0.7803	0.6876	0.6711	0.0831
			0.7445	0.6875	0.6990	0.0251
zoo	101	16	1	0.9825	0.8921	0.0625
			1	0.9890	0.9947	1.0783
			1	0.9926	0.9935	0.5036

The results show that the SV-Means algorithm is a suitable replacement for the one-class support vector machine for anomaly detection. One point of note is the results for the tic-tac-toe data set. The SV-Means algorithm produces a low AUC score. This is a result of the tic-tac-toe data set being deterministic, i.e., the values of the data are either 0 or 1. The SV-Means algorithm is a density estimation technique and does not perform as well with deterministic data in lower-dimensional space as it does with non-deterministic data. Experiments were performed with $d_{RF} = 50,000$ random features and the AUC results for the SV-Means algorithm subsequently raised to 0.94.

5.3.4 SV-Means vs. Density Estimation Algorithms

The performance of the SV-Means algorithm for density estimation is evaluated with $\nu^t_{min} = 0$, $\nu^t_{max} = 1$, and $\boldsymbol{\nu}^o = [0.8,\ 0.6,\ 0.4,\ 0.2,\ 0.05,\ 0]$ using the radar waveform data described in Section 5.1.3. The accuracy of the model for likelihood prediction is verified by generating new data consistent with the training data, but with different noise and phase

realizations. An example of the accuracy of the quantile bin counts are shown in Figure 5.3 where each bin contains the averaged counts of all 23 classes combined. The number of training instances in each bin should correspond to $\alpha = 1 - \nu$. For example, for $\nu = 0.8$ bin, 20% of the training data should be contained within the quantile boundary.

Another experiment (repeated from [13]) was performed using the 16 UCI data sets described in Section 5.1.2. For each data set, 100 randomly selected points were chosen from the class with the largest amount of points for training. The remaining points from that class were used for testing. The SV-Means algorithm was compared to the q-OCSVM [13] and I-OCSVM (independent OCSVM) estimating level sets at $\alpha_1 = 0.05, \ldots, \alpha_{19} = 0.95$ for 19 total quantiles. I-OCSVM is simply calculating an independent single OCSVM boundary for each quantile. The accuracy of the level sets are determined by the coverage ratio (CR) where CR $= \frac{\alpha'}{\alpha}$; a perfect CR equals 1. The parameter α' is the number of testing points, and α is the number of expected testing points (αn) within the level set. In Figure 5.4a, α' is plotted against α averaged for all 16 data sets in a bar graph for each technique at each of the 19 level sets. In Figure 5.4b, α is plotted against the coverage ratio averaged over all 16 data sets. The SV-Means algorithm proves to be an accurate density estimation technique outperforming the q-OCSVM and I-OCSVM algorithms in coverage ratio for almost all level sets.

5.3.5 SV-Means vs. Open Set Algorithms

The open set algorithms mentioned in Section 4.3 (one-vs-Set Machine, W-SVM, PI-SVM, POS-SVM) and the SV-Means algorithm are compared using the phase-coded radar waveform set for Section 5.1.3. In order to compare the algorithms in a reasonable amount of time, the training data (with $n = 11000$ and $\Theta \in \{[-12, 12]\}$) was split 50/50 for training and testing with the comparison of open set algorithms relying on LIBSVM [84] to optimize their SVM formulations. For each algorithm, a 5-fold cross-validation procedure was performed to find the optimum parameter values of C and γ. The SV-Means algorithm

used the ν-parameters specified in Table 5.2. The algorithms were compared using varying degrees of openness defined in (4.1). The first experiment, in Figure 5.5a, used the first 13 classes defined in Table 5.1 as known classes in training and testing with 3 folds to obtain error bars. The varied openness levels are evaluated by adding subsets of the remaining 10 classes for testing. The second experiment in Figure 5.5b used the first 3 classes defined in Table 5.1 as known classes and a subset of the remaining 20 classes for testing.

The algorithms are compared using F-measure, which was proposed in [6] as a good statistic for comparing open set algorithm performance. F-measure is defined as

$$\text{F-measure} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5.2}$$

where $\text{Recall} = \frac{TP}{TP+FN}$ and $\text{Precision} = \frac{TP}{TP+FP}$ and TP, TN, FP, FN are defined as true positive, true negative, false positive, and false negative, respectively. A rejected sample that is from an unknown class is treated as a true negative, or if from a known class, as a false negative.

The two experiments show that when the problem is more "closed", the binary-SVM based algorithms perform well, but as the problem becomes more "open", the performance of existing algorithms drop. In Figure 5.5a, the POS-SVM algorithm follows the one-vs-All SVM performance curve. W-SVM and SV-Means, algorithms based on OCSVMs, are shown to give the best performance via high F-measure. In Table 5.8, the algorithms are timed on an HP (94.5 GB Intel(R) Xeon(R) CPU E5640@2.67GHz) computer with MATLAB 2017a. However, the comparison of open set algorithms is acheived via LIB-SVM which is written in C++, and SV-Means is purely in MATLAB. The timing includes training every class, whereas the previous timing in Table 5.6 trained a single class for comparison. It is shown that SV-Means is faster than W-SVM, PI-SVM, and one-vs-Set Machine (even though SV-Means is written in MATLAB). SV-Means is also shown to be faster than POS-SVM as the number of training examples and classes grow.

Table 5.8: Timing comparison of open set algorithms (in seconds)

# of training classes	SV-Means	POS-SVM	W-SVM	PI-SVM	One-vs-Set Machine
3	6.57	1.34	66.63	25.93	17.18
13	29.30	57.33	269.23	223.13	125.47

5.4 SV-Means Level Set Clustering

An illustrative example is given for the SV-Means Level Set Clustering algorithm described in Section 4.4 with two level sets. A 2-D data set is used to visualize the clustering process. The data consists of three separate Gaussians for each of the three classes, with $n = 1000$ and $d = 2$ for each class in a matrix X (2×3000).

Within SV-Means Level Set Clustering (Algorithm 2), the data X is first transformed via random features to form Z (line 1). The SV-Means algorithm is performed on the matrix Z with $b_{max} = 10$, $m_{max} = 20$, $\nu^t_{min} = 0$, $\nu^t_{max} = 0.05$, and $\boldsymbol{\nu}^o = [0.3,\ 0]$ (line 2). The vector $\boldsymbol{\nu}^o$ defines the level sets and clustering is performed at each level set. In Figure 5.7a, the three Gaussian classes are illustrated with the boundary corresponding to level set 1 ($\nu^o_1 = 0.3$) shown in red. In Figure 5.7b, the minimum spanning tree is shown determined by the level set 1 points in ambient space (line 2). Also pictured in this Figure are the three longest edges, $e_{j,u}$, in set E_j, and along each edge are the set of $r_{max} = 4$ points (red squares) in set $A_{e_{j,u}}$. For each longest edge, each point in $A_{e_{j,u}}$ is tested using the SV-Means model corresponding to level set 1. If any of the points in $A_{e_{j,u}}$ are outside of this model, the edge is removed and a separate tree is formed (line 17). The upper right corner Gaussian has a longest edge pictured, but it is not removed as the points along that edge were all within the SV-Means model. In this example, two edges were removed forming three separate trees. The points are associated via depth first search for each cluster. In Figure 5.7c, the green lines show the true class label and the black line shows the SV-Means Level Set Clustering label. In Figure 5.7d, the points are plotted by color to show three distinct classes.

For level set 2 ($\nu_2^o = 0$), the SV-Means boundary is shown in Figure 5.8a. The points corresponding to level set 2 include almost all of the points in X. However, to reduce the number of points used in the minimum spanning tree, only the points within the model and not in level set 1, are used as shown in Figure 5.8b. The same procedure is followed as when performing the level set 1 clustering. The final clusters for level set 1 are shown in Figure 5.8d. In Figure 5.8c, the green line shows the true class, while the black line shows the classes assigned by the SV-Means Level Set Clustering algorithm. At level set 2, only two clusters are found as two of the classes are overlapping.

The SV-Means Level Set Clustering algorithm illustrative example visually demonstrates how the algorithm works. The algorithm provides an informative way to cluster data at different level sets which gives the number of clusters, k_j, at each level set and provides a way to separate overlapping clusters. The level set boundaries also provide insight as they refer to probabilities rather than purely structure. The SV-Means Level Set Clustering algorithm is used in the final end-to-end demonstration in the next Section.

5.5 End-to-End Demonstration

Finally, an end-to-end demonstration is performed showing the capability and flexibility of the SV-Means algorithm. As in Section 5.4, which visually demonstrated the SV-Means Level Set Clustering algorithm, the three Gaussian 2-D data set is used to illustrate the end-to-end process of training, testing, clustering, and adding new classes to the library. At each step, the SV-Means algorithm is used. After the visual end-to-end demonstration, the radar waveform data set is used and only the final confusion matrices are shown for comparison.

For the 2-D Gaussian data set, there are three experiments for comparison. The first experiment provides a closed set example where the two upper right Gaussians (from the three Gaussian 2-D data set) are used to train, and a new realization of these two classes

are used in the testing data (openness = 0). In Figure 5.9, the training data is shown in the upper left corner in black and each class is provided with a model using SV-Means with six hierarchical boundaries corresponding to $\boldsymbol{\nu}^o = [0.8,\ 0.6,\ 0.4,\ 0.2,\ 0.05,\ 0]$, as shown in the bottom left figure. The testing data in blue is shown in the top middle figure and the testing points that were rejected are shown in the top right figure by blue squares. The confusion matrix on the bottom right shows the results from the closed set experiment with an additional column, designated by R, for the rejected samples. The yellow boxes around several parts of the overall figure signify the data that was rejected and that could be passed on to clustering. In this case, clustering was not performed on the seven rejected data points.

The second experiment uses the models from the two upper right Gaussians and adds the bottom left Gaussian class as an unknown class into the testing data (openness = 0.1). The first experiment's diagram flow is used in the second experiment and is shown in Figure 5.10. The top middle figure shows the additional class added to the testing data in blue. Using the models from the first experiment, the rejected data is pictured in the top right figure by blue squares. In this experiment, the rejected data is passed onto the SV-Means Level Set Clustering algorithm and is shown in the top left figure in Figure 5.11. The SV-Means Level Set Clustering algorithm is performed with only one level set (SV-Means with $\nu^o = 0.2$) and only one cluster was found, shown with green points in the top middle figure. The green points corresponding to the single cluster are then passed to the SV-Means algorithm with $\boldsymbol{\nu}^o = [0.8,\ 0.6,\ 0.4,\ 0.2,\ 0.05,\ 0]$ to create a model for the class and the six red boundaries are shown as well in the top middle figure. Finally, the new class is added to the library, as shown in the bottom right figure.

The third experiment provides a best-case-scenario baseline where all three Gaussian classes are known and their models are shown in the top left figure in Figure 5.12. The best-case-scenario baseline models are compared to the models found from the second experiment (two classes were known, one was discovered through clustering and added to

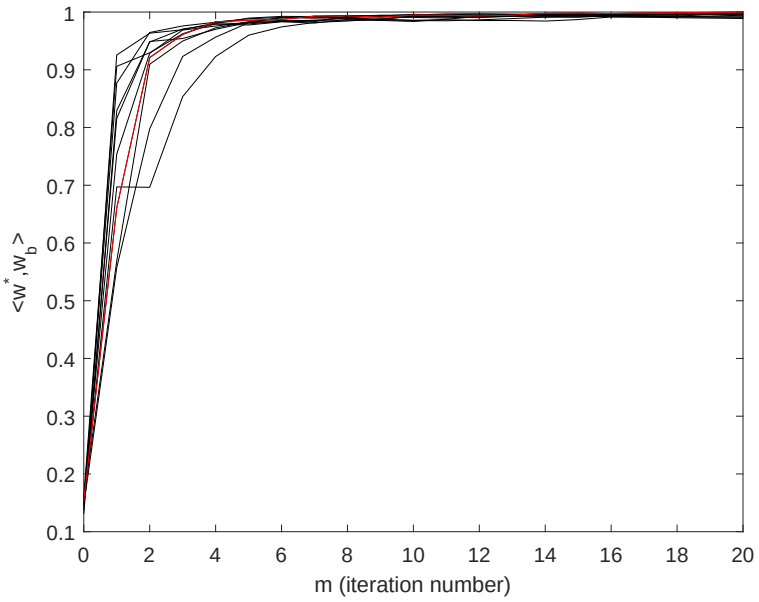
the library) and are shown in the bottom left figure. The testing data is shown in blue in the top middle figure. The best-case-scenario confusion matrix is shown on the top right and the second experiment confusion matrix is shown on the bottom right. The confusion matrices show favorable results for adding a new class to the library using SV-Means for training, clustering, and adding a new class to the library, even for overlapping clusters.

For the radar waveform data, only the final confusion matrices are shown in Figure 5.13 as the data is 20-dimensional. The first ten phase-coded radar waveforms are used from Table 5.1. In the best-case-scenario experiment, all ten waveforms are known for training. In the open set comparison experiment, the first seven waveform classes were known in training and the remaining three classes were brought in at testing. The rejected points were sent to the SV-Means Level Set Clustering algorithm where one level set was used (SV-Means with $\nu^o = 0.2$) and three clusters were found. The three clusters were added to the library using SV-Means with $\boldsymbol{\nu}^o = [0.8,\ 0.6,\ 0.4,\ 0.2,\ 0.05,\ 0]$. The best-case-scenario confusion matrix is shown in Figure 5.13a and the open set comparison experiment (seven known and three unknown classes) is shown in Figure 5.13b. The confusion matrices are covered with yellow boxes highlighting the three classes to the library. The results show that SV-Means is a powerful and flexible algorithm that is able to train, cluster, and quickly add classes into to the library.

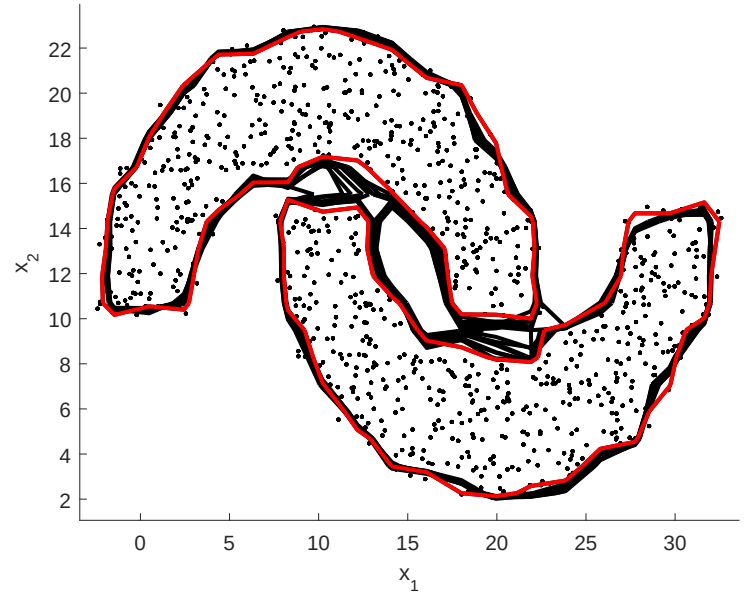
5.6 Chapter Summary

In this Chapter, several sets of experiments were performed to explore the capability and flexibility of the novel SV-Means algorithm using several different data sets. The first set of experimentation demonstrates the convergence properties of the SV-Means algorithm and demonstrates quantile estimation. The next large set of experimentation compares the SV-Means algorithm to the q-OCSVM algorithm modified to perform classification (primal and dual) and several applications: anomaly detection, density estimation, and open set

classification. The SV-Means Level Set Clustering algorithm is then visually demonstrated to show the capability to determine clusters at different level sets and distinguish overlapping clusters. Finally, an end-to-end demonstration is shown from training, to testing, to clustering, to adding new classes to the library using the SV-Means algorithm at each step.



(a)



(b)

Figure 5.1: SV-Means algorithm performed on the double moon 2-D data set where all $b_{max} = 10$ initializations are shown in both subfigures. In 5.1a, all 10 initializations converge within $m_{max} = 20$ inner loop iterations. In 5.1b, the figure illustrates all 10 boundaries corresponding to each initialization with $\nu^o = 0.05$. This demonstrates the non-convexity of the SV-Means algorithm and, therefore, the need for multiple random initializations. The best boundary, shown in red, confirms the criteria to pick a final $\mathbf{w}^*$ and $\boldsymbol{\rho}^*$ (lines 18-19 in Algorithm 1).

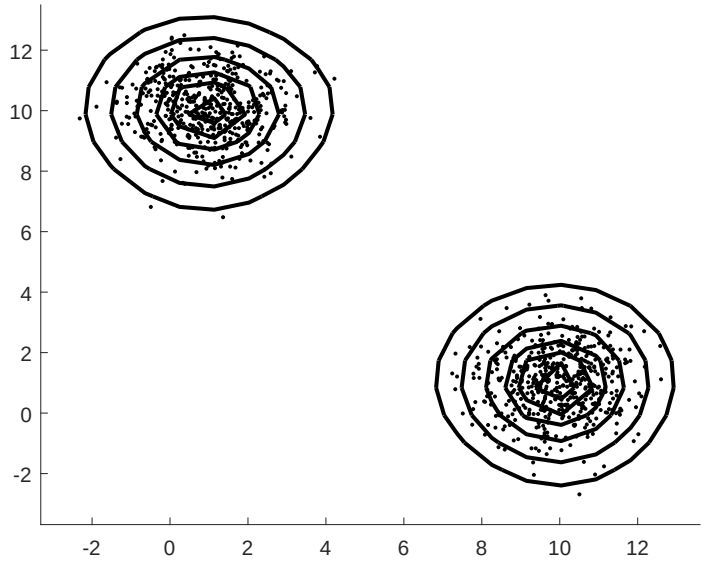


Figure 5.2: Double Gaussian data set density estimation is performed by the SV-Means algorithm where $\boldsymbol{\nu}^o = [0.8,\ 0.6,\ 0.4,\ 0.2,\ 0.05,\ 0]$.

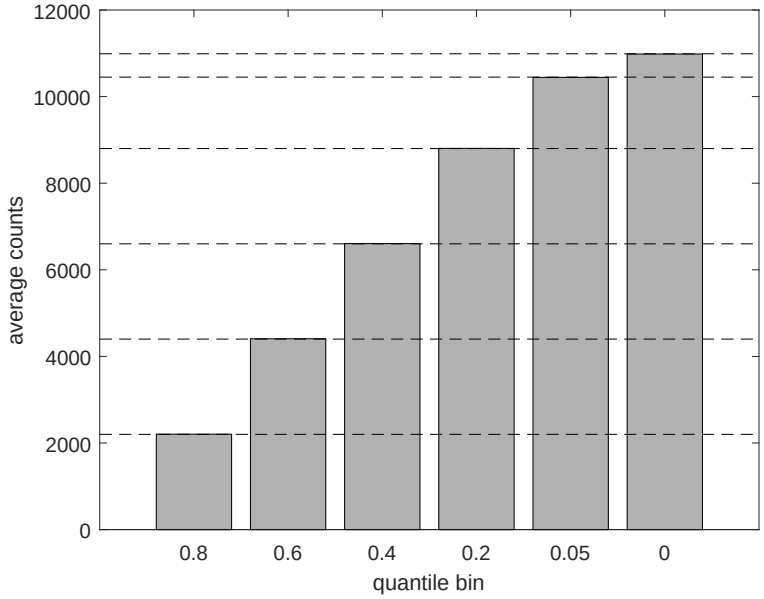
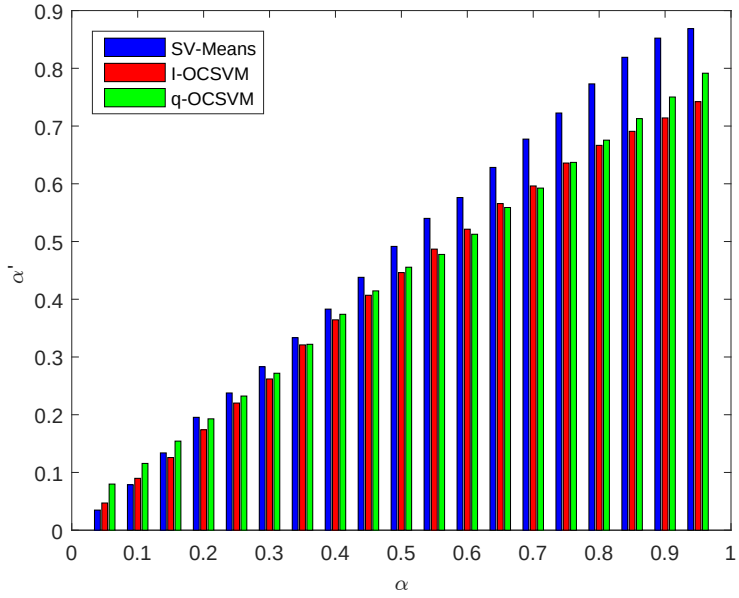
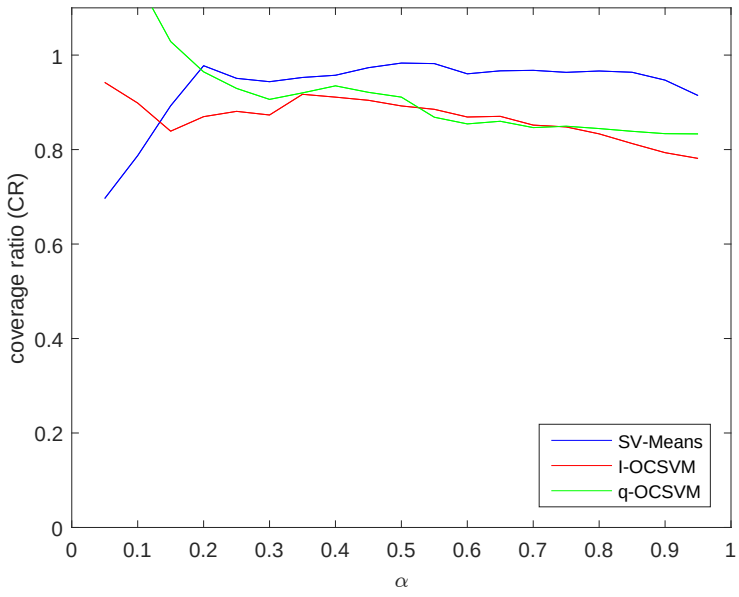


Figure 5.3: Histogram of average counts for each quantile bin showing the model accuracy for all 23 classes averaged. The lines show the bin counts corresponding to $\boldsymbol{\nu}^o = [0.8,\ 0.6,\ 0.4,\ 0.2,\ 0.05,\ 0]$.

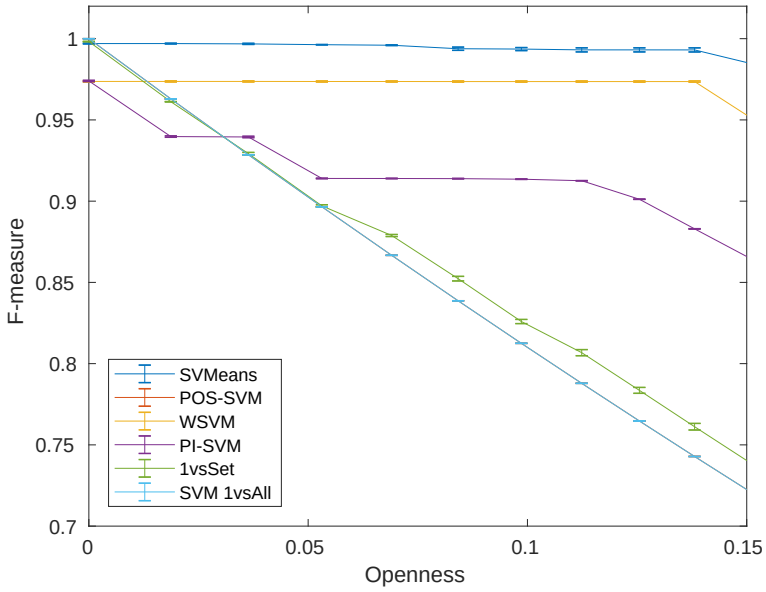


(a)

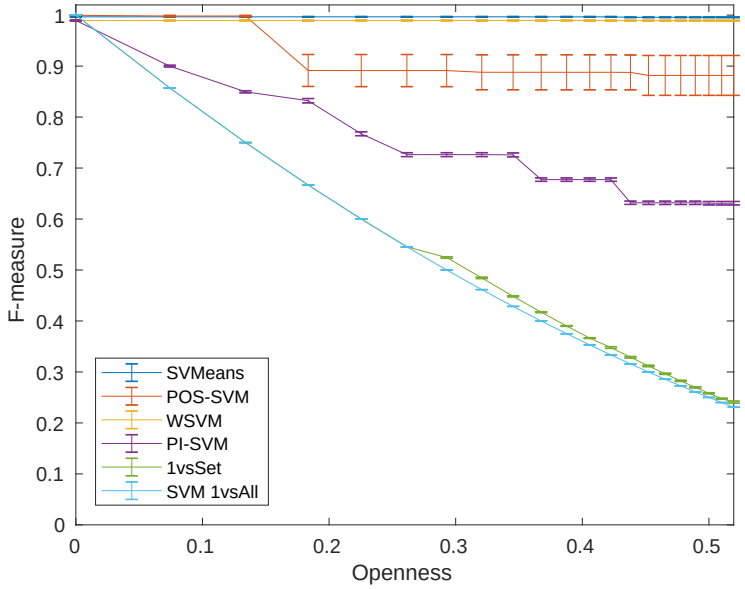


(b)

Figure 5.4: The SV-Means, q-OCSVM, and I-OCSVM algorithms were trained to estimate $\alpha_1 = 0.05, \ldots, \alpha_{19} = 0.95$, or 19 total quantiles, for the distribution using the largest class from each of the 15 UCI data sets desribed in Section 5.1.2. Figure 5.4a depicts α' as a function of α averaged over all data sets, and Figure 5.4b depicts the coverage ratio as a function of α averaged over all data sets.

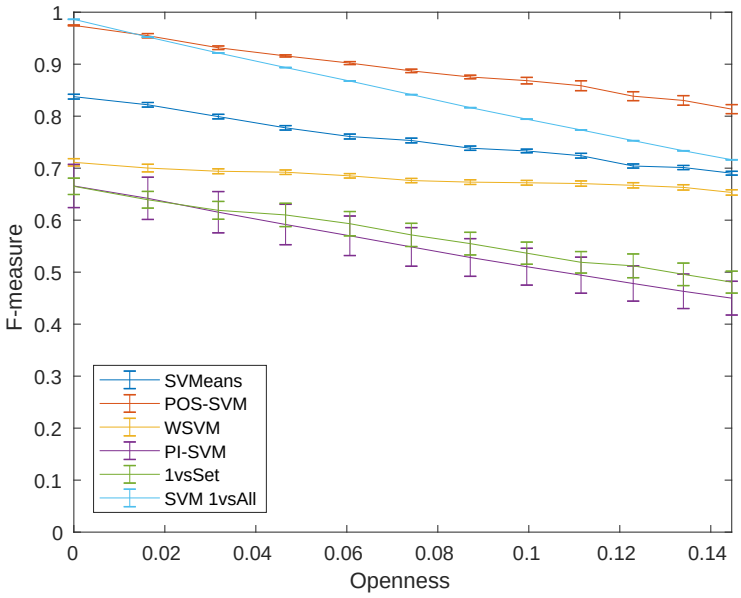


(a)

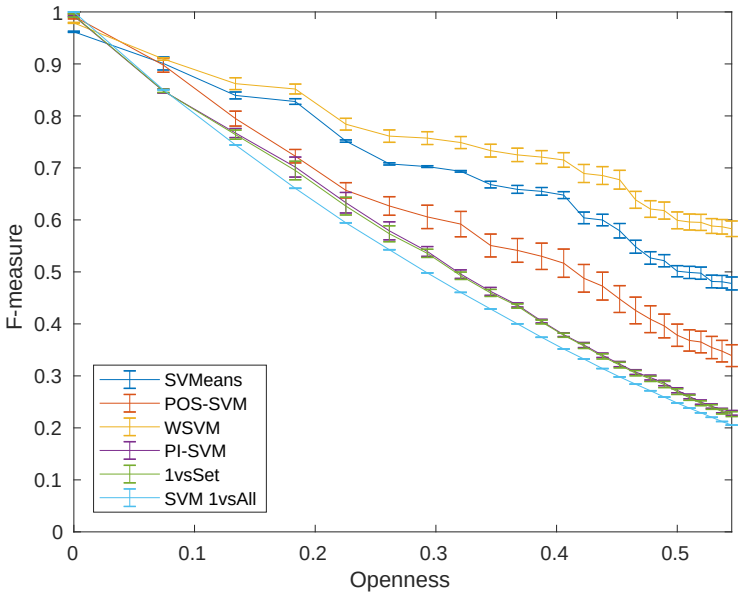


(b)

Figure 5.5: Comparison of multi-class open set classification algorithm F-measure scores for 13 classes in 5.5a, and 3 classes in 5.5b over 3 folds. The openness levels are tested by adding a subset of the remaining 10 and 20 classes, respectively.



(a)



(b)

Figure 5.6: Comparison of multi-class open set classification algorithm F-measure scores for 15 classes in 5.6a, and 3 classes in 5.6b over 3 folds. The openness levels are tested by adding a subset of the remaining 11 and 23 classes, respectively.

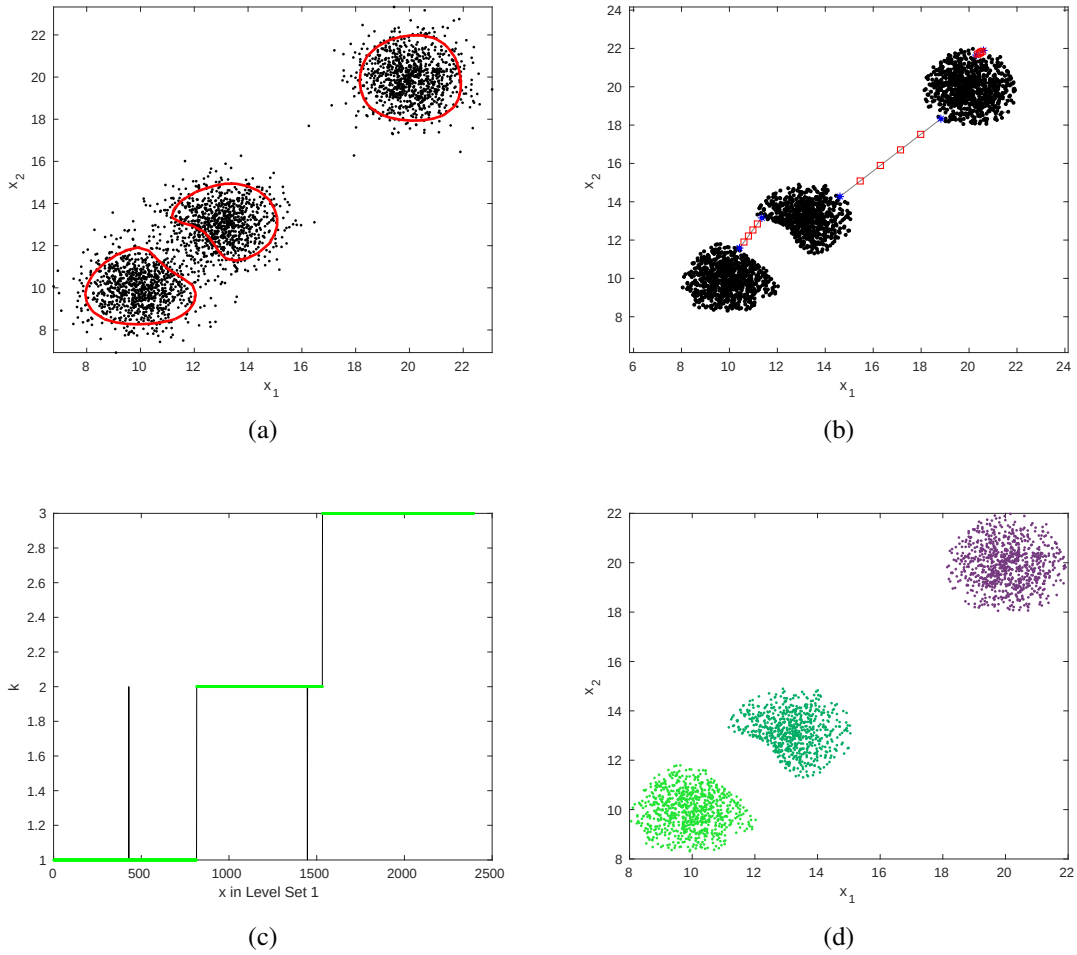


Figure 5.7: SV-Means Level Set Clustering 2-D example - level set 1: In 5.7a, the boundary corresponding to level set 1 ($\nu_1^o = 0.3$) is shown in red. In 5.7b, the minimum spanning tree is shown including 3 longest edges. Along the longest edges, testing points are shown as 4 red squares. In 5.7c, the correct cluster number for each point is shown in green and the cluster label given to each point by the SV-Means Level Set Clustering algorithm is shown in black. In 5.7d, the points are color-coded to which cluster they belong.

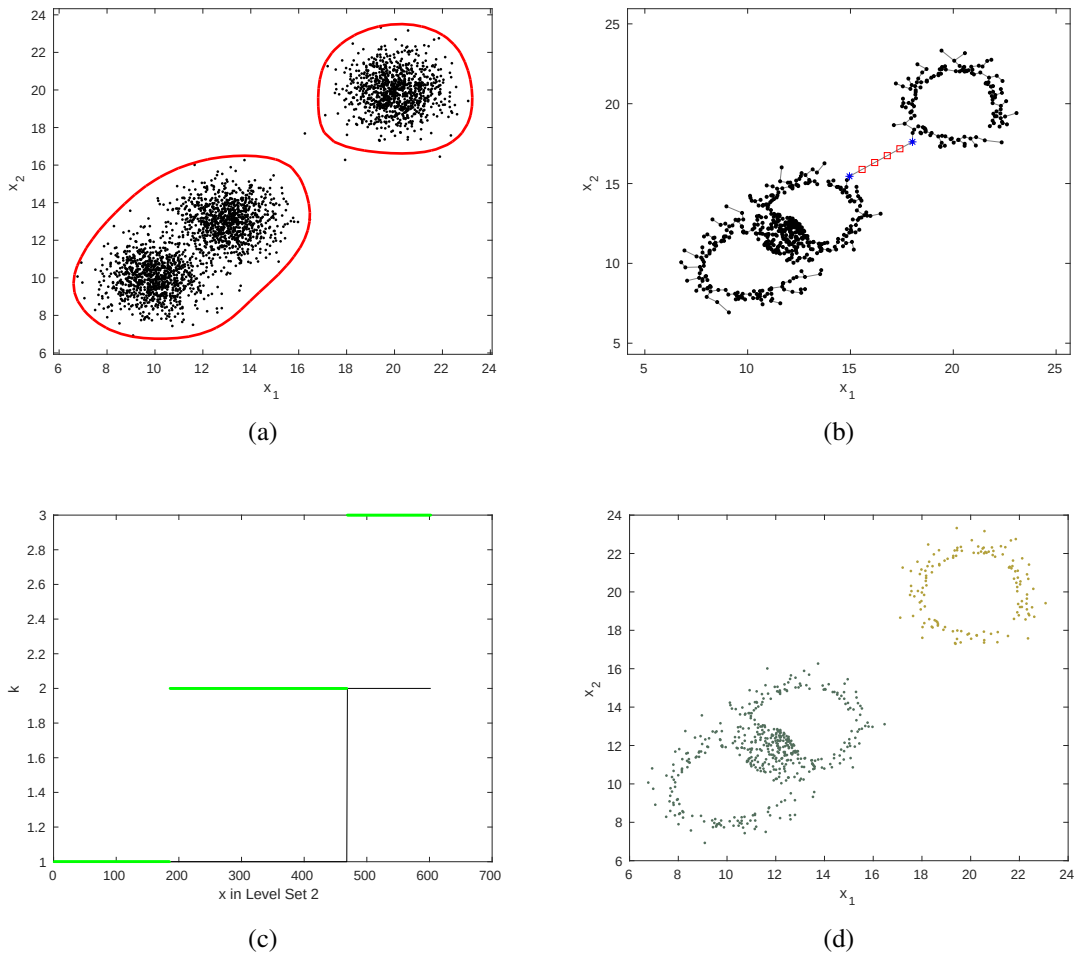


Figure 5.8: SV-Means Level Set Clustering 2-D example - level set 2: In 5.7a, the boundary corresponding to level set 2 ($\nu_2^o = 0$) is shown in red. In 5.7b, the minimum spanning tree is shown including 1 long edges. Along the longest edge, testing points are shown as 4 red squares. In 5.7c, the correct cluster number for each point is shown in green and the cluster label given to each point by the SV-Means Level Set Clustering algorithm is shown in black. In 5.7d, the points are color-coded to which cluster they belong. At level set 2, the overlapping clusters are not distinguishable.

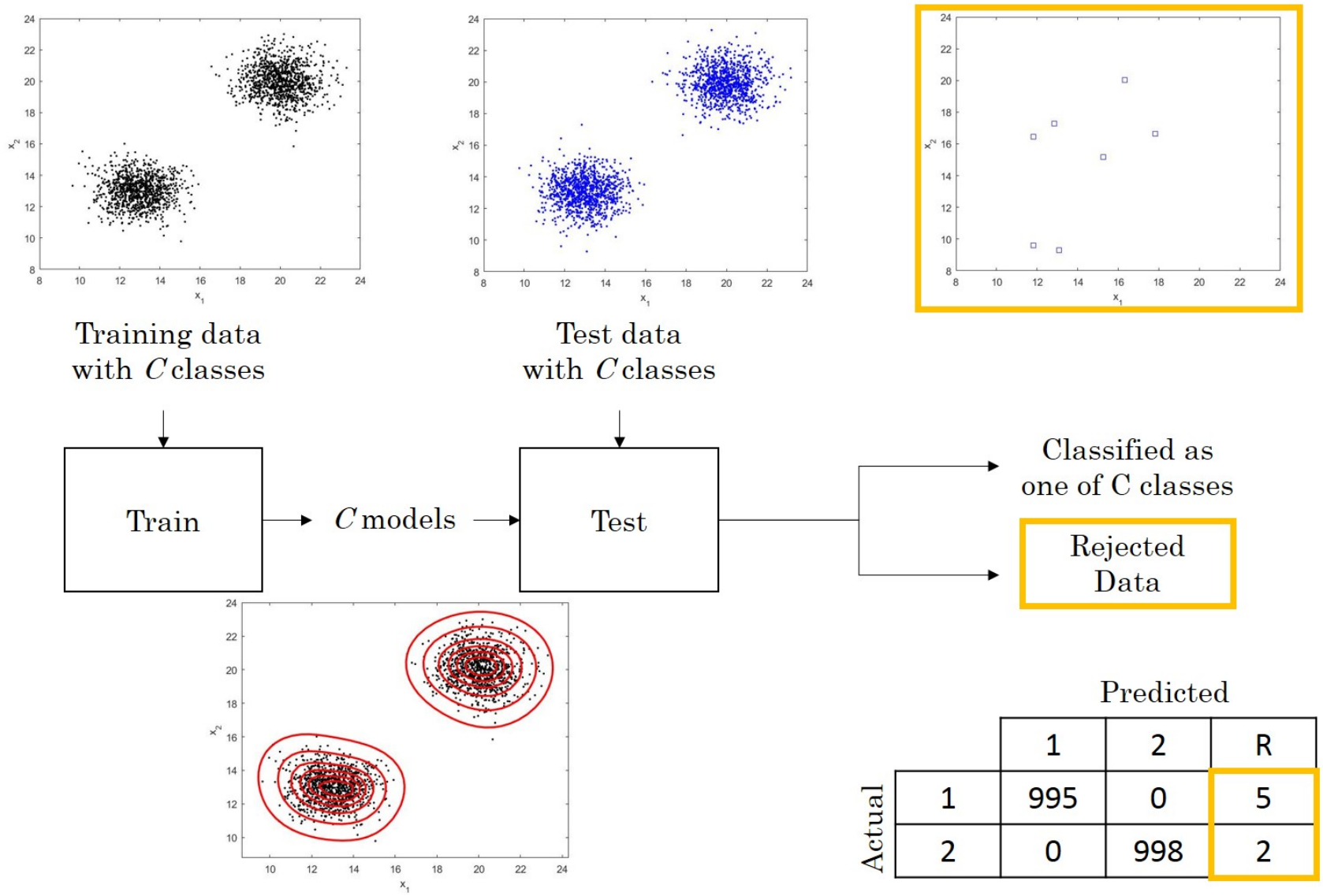


Figure 5.9: End-to-end demonstration: 2-D Gaussian data set. A closed set example (openness = 0) is given where two classes were used to train models using SV-Means. A different realization of the two classes were used for testing and the rejected points are highlighted in yellow in the top right figure. The confusion matrix is given in the bottom right corner.

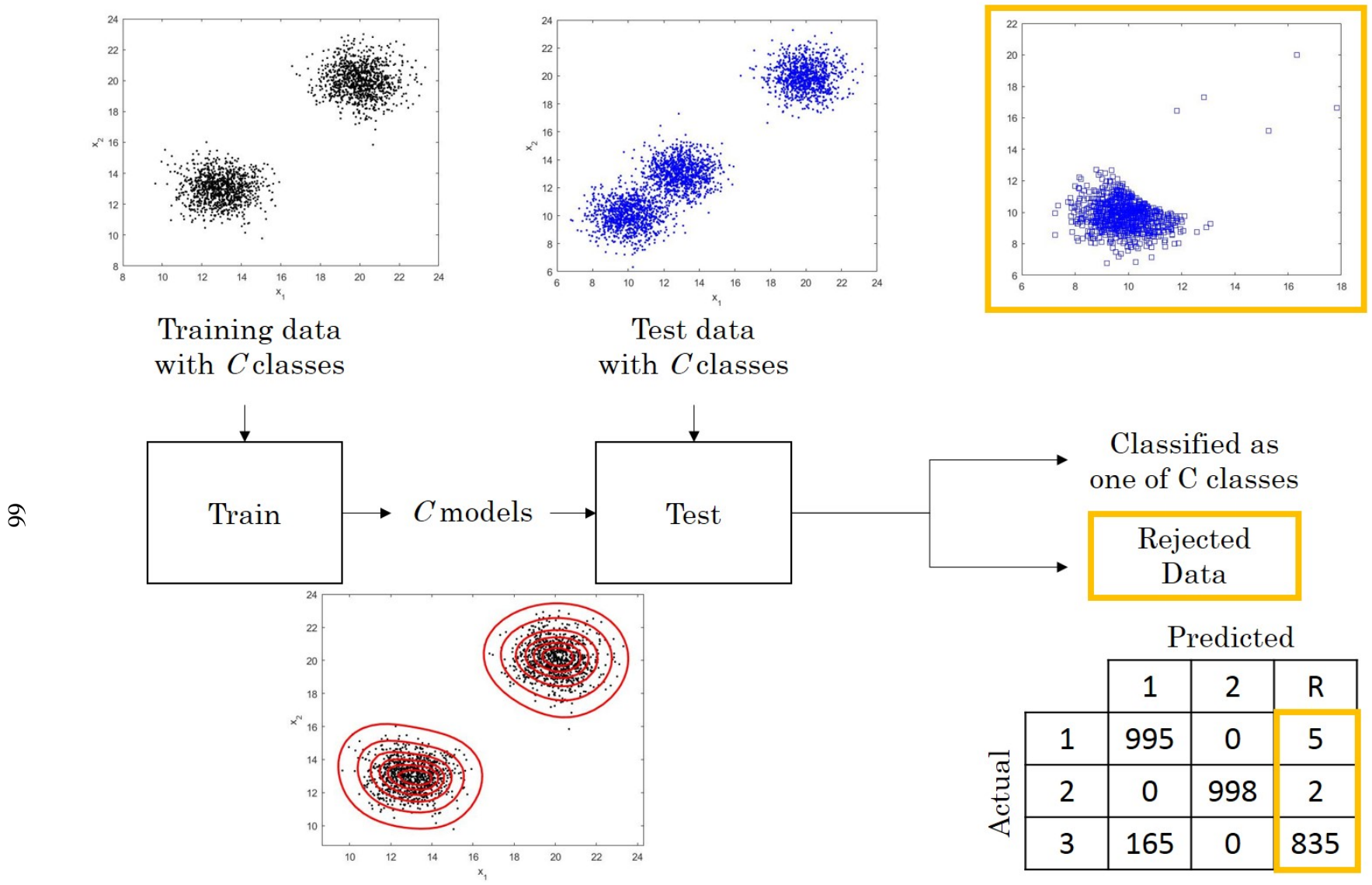


Figure 5.10: End-to-end demonstration: 2-D Gaussian data set. An open set example (openness = 0.1) is given where the same two classes/models are used as in Figure 5.9 except now the testing data includes a new class (picured in blue in the top middle figure). The rejected points are highlighted in yellow in the top right figure and the confusion matrix is given in the bottom right corner. The rejected points are passed to a clustering algorithm block shown in Figure 5.11.

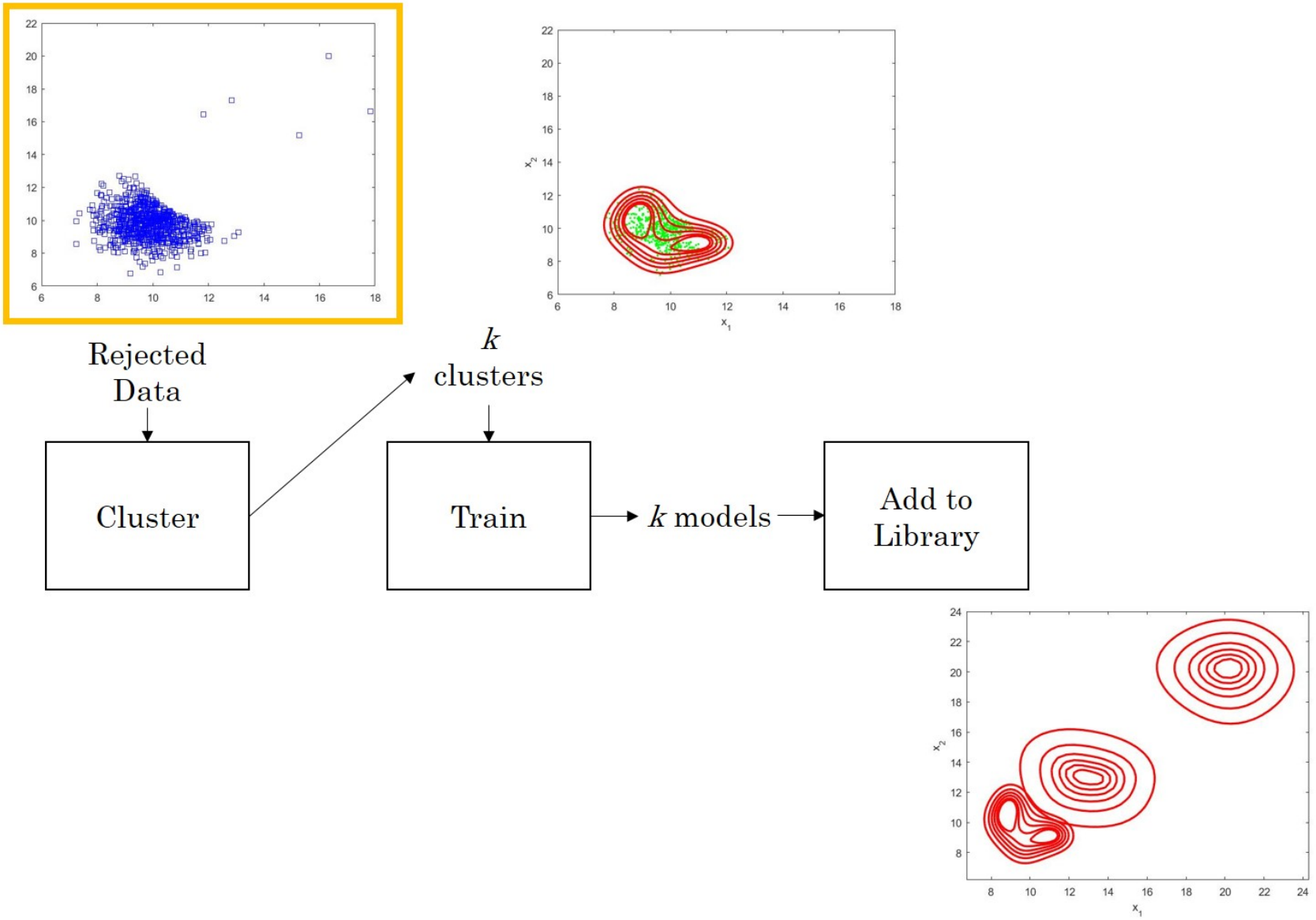


Figure 5.11: End-to-end demonstration: 2-D Gaussian data set. The rejected points from Figure 5.10 are sent to the SV-Means Level Set Clustering algorithm where one new cluster was determined (pictured in green in the top middle figure). The new cluster is trained as a new class using the SV-Means algorithm and the new class is added to the library (pictured in the bottom right figure).

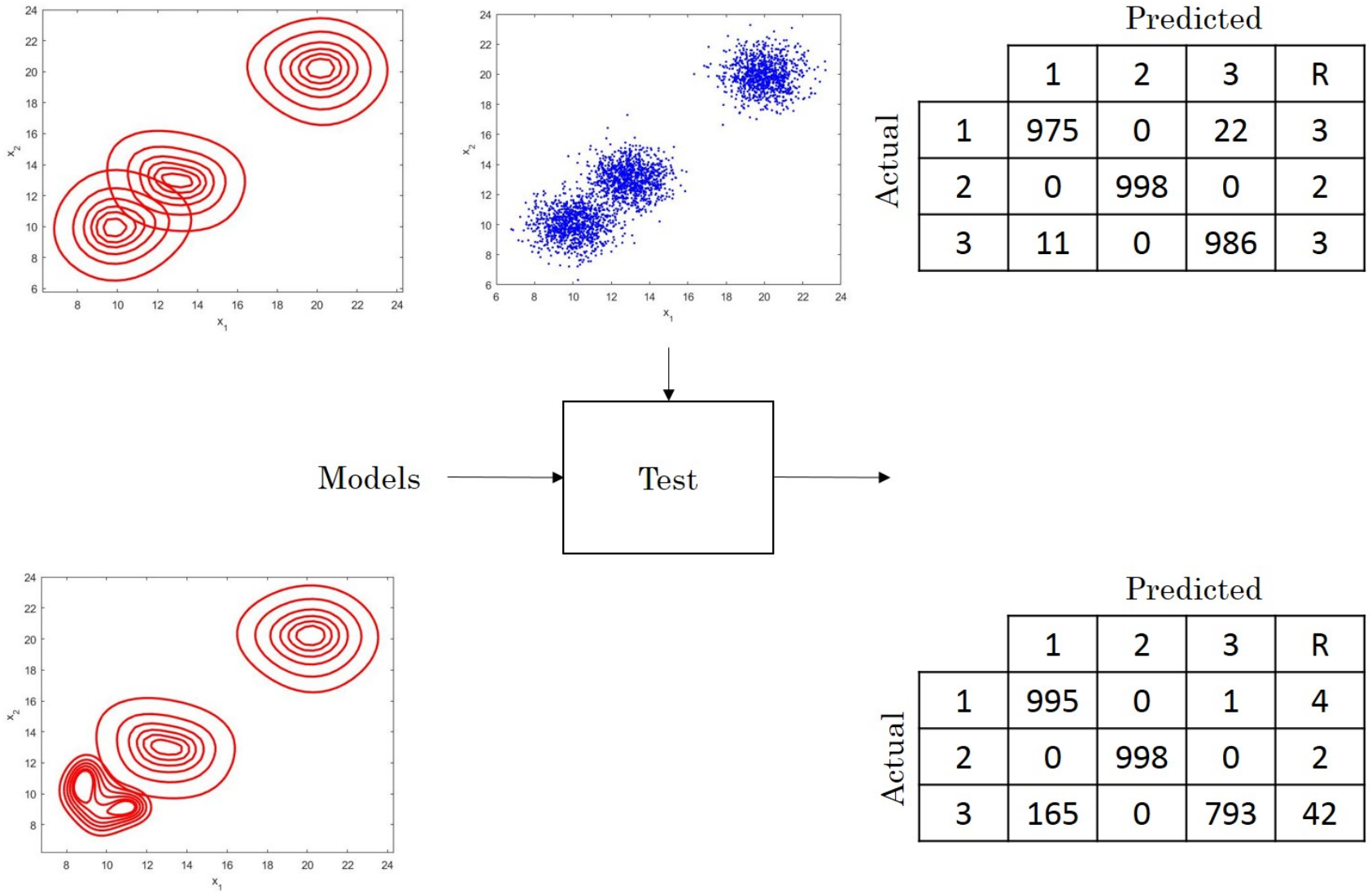


Figure 5.12: End-to-end demonstration: 2-D Gaussian data set. The performance of the open set experiment models (two known and one discovered via clustering) from Figure 5.11 is measured by comparing to the best-case-scenario models (assuming all three classes were known at training). The best-case-scenario models and corresponding confusion matrix are shown along the top, and the open set experiment models and corresponding confusion matrix are shown along the bottom.

Predicted

Actual	1	2	3	4	5	6	7	8	9	10	R
1	992	0	0	0	0	0	0	0	0	0	8
2	0	992	0	0	0	0	0	0	0	0	8
3	0	0	987	0	0	0	0	0	0	0	13
4	0	0	0	983	0	0	0	0	0	0	17
5	0	0	0	0	981	0	0	0	0	0	19
6	0	0	0	0	0	983	0	0	0	0	17
7	0	0	0	0	0	0	993	0	0	0	7
8	0	0	0	0	0	0	0	935	0	0	65
9	0	0	0	0	0	0	0	0	990	0	10
10	0	0	0	0	0	0	0	0	0	991	9

(a)

Predicted

Actual	1	2	3	4	5	6	7	8	9	10	R
1	992	0	0	0	0	0	0	0	0	0	8
2	0	992	0	0	0	0	0	0	0	0	8
3	0	0	987	0	0	0	0	0	0	0	13
4	0	0	0	983	0	0	1	0	0	0	16
5	0	0	0	0	981	0	0	0	0	0	19
6	0	0	0	0	0	983	0	0	0	0	17
7	0	0	0	0	0	0	993	0	0	0	7
8	0	0	0	0	0	0	0	864	0	0	136
9	0	0	0	0	0	0	0	0	805	0	195
10	0	0	0	0	0	0	0	0	0	763	237

(b)

Figure 5.13: End-to-end demonstration: phase-coded radar waveform data set. The performance of the open set experiment models (seven known and three discovered via clustering) from Figure 5.11 is measured by comparing to the best-case-scenario models (assuming all ten classes were known at training). The best-case-scenario confusion matrix is shown in Figure 5.13a and the open set experiment confusion matrix is shown in Figure 5.13b. The R column represents the rejected class.

Chapter 6

Closing Remarks

In this dissertation, the SV-Means algorithm is developed and has been proven to be a capable and flexible algorithm as shown in the extensive experimentation in the previous Chapter. In this Chapter, a summary of contributions, expected impacts, and recommended future research is discussed.

6.1 Summary of Contributions

This dissertation includes two overarching contributions: (1) the development of SV-Means algorithm and (2) demonstration of the SV-Means algorithm applied to several different applications. Within the SV-Means algorithm development, there are three main contributions: a generative Bayesian formulation, an accurate boundary definition, and fast training. The SV-Means algorithm transforms a one-class support vector machine-based problem from a traditionally discriminative algorithm into a generative algorithm by modeling each class likelihood with multiple hierarchical boundaries accurately delineating probability quantiles. The SV-Means algorithm provides an accurate boundary by estimating multiple level sets near the extrema of the data. The SV-Means algorithm provides fast training by solving the primal formulation using a non-convex, k-means inspired algorithm based on stochastic gradient descent principles using random Fourier features to estimate the kernel.

The second overarching contribution is demonstration of the SV-Means algorithm applied to several different application areas. There are four contributions attributed to the application areas explored with the SV-Means algorithm: anomaly detection, density estimation, open set classification, and clustering. In anomaly detection, the accurate boundary definition provided by SV-Means showed improved results in high dimensions. In density estimation, the SV-Means algorithm provides a more accurate outer boundary compared to the q-OCSVM which optimizes over equal level sets over the entire distribution. The SV-Means algorithm also executes significantly faster as the number of training data points and level sets increase. In open set classification, the SV-Means algorithm is the only algorithm that uses OCSVMs exclusively as compared to algorithms that include binary SVMs to distinguish between known classes. The SV-Means algorithm is a strong candidate for open set classification as the problem becomes more open, and SV-Means runs significantly faster than other open set algorithms. In clustering, a novel algorithm, SV-Means Level Set Clustering, was developed using SV-Means to determine the number of clusters at different level sets and to provide a way to distinguish between overlapping clusters.

The SV-Means algorithm has some additional desirable properties. Since each waveform class is trained separately, the addition of new waveforms to the library does not require the retraining of the other waveform classes. The SV-Means classification algorithm is an attractive adaptive classification framework for use with any feature extraction approach including deep learning architectures.

6.2 Expected Impact

The SV-Means algorithm provides several advantages over a typical hyperplane OCSVM (and q-OCSVM) that makes it a viable choice for anomaly detection, density estimation, open set classification, and clustering. These advantages make it possible to use the SV-means algorithm as the processing engine for an end-to-end adaptive classification system

in operational time frames as illustrated in Chapter 5.5. The simplicity and speed of the algorithm allows new classes to be added to the library without retraining the entire library, and these library additions can be made quickly for large data sets. These attributes will have a significant impact on any area where the environment is constantly evolving and the classes in the training set are only a small subset of the classes in the real world (e.g., waveform classification).

6.3 Future Research

Future work will pursue using deep learning architectures for adaptive waveform feature design since it is believed that the SV-Means algorithm's ability to work with large data sets will be compatible with the large data sets used to train deep learning algorithms. Also, research in deep learning networks, due to their very long training times, is continually improving stochastic gradient descent techniques and applying these techniques to the SV-Means algorithm could provide an even faster algorithm. In addition, the SV-Means algorithm has complete control over the number of support vectors and outliers as it is based on the ν-OCSVM formulation. Fitting an EVT model to these boundary points, like the W-SVM and PI-SVM open set algorithms, could provide even stronger results. Finally, the SV-Means Level Set Clustering algorithm is a novel algorithm with promising results, but portions of the algorithm could be further tested and optimized, in particular, the second major step of the algorithm, i.e., cluster membership.

Bibliography

[1] G. Cohen, M. Hilario, and C. Pellegrini, *One-Class Support Vector Machines with a Conformal Kernel. A Case Study in Handling Class Imbalance*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 850–858. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-27868-9_93

[2] K. Noto, C. Brodley, and D. Slonim, "Frac: a feature-modeling approach for semi-supervised and unsupervised anomaly detection," *Data Mining and Knowledge Discovery*, vol. 25, no. 1, pp. 109–133, Jul 2012. [Online]. Available: https://doi.org/10.1007/s10618-011-0234-x

[3] J. Lunden and V. Koivunen, "Automatic radar waveform recognition," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 1, no. 1, pp. 124–136, 2007.

[4] M. T. Mushtaq, F. A. Butt, and A. Malik, "An overview of spectrum sensing in cognitive radar systems," in *2014 IEEE Microwaves, Radar and Remote Sensing Symposium (MRRS)*, Sept 2014, pp. 115–118.

[5] F. Gini, A. D. Maio, and L. Patton, Eds., *Waveform Design and Diversity for Advanced Radar Systems*, ser. Radar, Sonar & Navigation. Institution of Engineering and Technology, 2012.

[6] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult, "Toward open set recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 7, pp. 1757–1772, July 2013.

[7] W. Eskin, E.; Lee and S. S., "Mmodeli system call for intrusion detection using dynamic window sizes," in *Proceedings of DARPA Information Survivability Conference and Exposition (DISCEX)*, 2001.

[8] N. Wu and J. Zhang, "Factor analysis based anomaly detection," in *IEEE Systems, Man and Cybernetics SocietyInformation Assurance Workshop, 2003.*, June 2003, pp. 108–115.

[9] R. J. Bolton, D. J. Hand, and D. J. H, "Unsupervised profiling methods for fraud detection," in *Proc. Credit Scoring and Credit Control VII*, 2001, pp. 5–7.

[10] R. Brause, T. Langsdorf, and M. Hepp, "Neural data mining for credit card fraud detection," in *Proceedings of the 11th IEEE International Conference on Tools with Artificial Intelligence*, ser. ICTAI '99. Washington, DC, USA: IEEE Computer Society, 1999, pp. 103–. [Online]. Available: http://dl.acm.org/citation.cfm?id=850950.853684

[11] M. Davy and S. Godsill, "Detection of abrupt spectral changes using support vector machines an application to audio signal segmentation," in *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, May 2002, pp. II–1313–II–1316.

[12] B. Scarth G.; McIntyre, M.; Wowk and R. Somorjai, "Detection of novelty in functional images using fuzzy clustering," in *Proceedings of the 3rd Meeting of the International Society for Magnetic Resonance in Medicine*, 1995, p. 238.

[13] A. Glazer, M. Lindenbaum, and S. Markovitch, "q-ocsvm: A q-quantile estimator for high-dimensional distributions," in *Advances in Neural Information Processing Systems*, 2013, pp. 503–511.

[14] A. Ben-Hur, D. Horn, H. T. Siegelmann, and V. Vapnik, "Support vector clustering," *Journal of Machine Learning Research*, vol. 2, pp. 125–137, 2001.

[15] A. Vapnik, V.; Lerner, "Recognition of patterns with the help of generalized portraits," *Avtomat. i Telemath*, vol. 24, no. 6, pp. 774–780, 1963.

[16] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[17] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, ser. COLT '92. New York, NY, USA: ACM, 1992, pp. 144–152. [Online]. Available: http://doi.acm.org/10.1145/130385.130401

[18] R. Khardon, "Deriving support vector machines: Optimizing functions under constrants," Lecture Notes: Advanced Topics in Machine Learning, April 2008.

[19] D. M. J. Tax and R. P. W. Duin, "Support vector domain description," *Pattern Recognition Letters*, vol. 20, pp. 1191–1199, 1999.

[20] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[21] B. Schölkopf, R. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt, "Support vector method for novelty detection," in *Proceedings of the 12th International Conference on Neural Information Processing Systems*, ser. NIPS'99. Cambridge, MA, USA: MIT Press, 1999, pp. 582–588.

[22] A. M. Pavy and B. D. Rigling, "Phase modulated radar waveform classification using quantile one-class svms," in *2015 IEEE Radar Conference (RadarCon)*, May 2015, pp. 0745–0750.

[23] Q. Liu, "Efficient radar emitters scheme recognition based on a novel svm algorithm," in *Advanced Engineering Forum*, vol. 4. Trans Tech Publ, 2012, pp. 232–237.

[24] R. Chitta, R. Jin, and A. K. Jain, "Efficient kernel clustering using random fourier features," in *Proceedings of the 2012 IEEE 12th International Conference on Data Mining*, ser. ICDM '12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 161–170. [Online]. Available: http://dx.doi.org/10.1109/ICDM.2012.61

[25] E. G. Bazavan, F. Li, and C. Sminchisescu, "Fourier kernel learning." in *ECCV (2)*, ser. Lecture Notes in Computer Science, A. W. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, Eds., vol. 7573. Springer, 2012, pp. 459–473.

[26] R. Brault, F. d'Alché-Buc, and M. Heinonen, "Random fourier features for operator-valued kernels," *CoRR*, vol. abs/1605.02536, 2016. [Online]. Available: http://arxiv.org/abs/1605.02536

[27] L. Deng, M. Hasegawa-Johnson, and X. He, "Random features for kernel deep convex network." IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), May 2013. [Online]. Available: https://www.microsoft.com/en-us/research/publication/random-features-for-kernel-deep-convex-network/

[28] S. Erfani, M. Baktashmotlagh, S. Rajasegarar, S. Karunasekera, and C. Leckie, "R1svm: a randomised nonlinear approach to large-scale anomaly detection," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[29] R. Collobert, S. Bengio, and Y. Bengio, "A parallel mixture of svms for very large scale problems," *Neural Comput.*, vol. 14, no. 5, pp. 1105–1114, May 2002. [Online]. Available: http://dx.doi.org/10.1162/089976602753633402

[30] I. Jolliffe, *Principal Component Analysis*. Springer Verlag, 1986.

[31] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, p. 2319, 2000.

[32] G. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504 – 507, 2006.

[33] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Advances in Large Margin Classifiers*. MIT Press, 1999, pp. 61–74.

[34] O. Chapelle, "Training a support vector machine in the primal," *Neural computation*, vol. 19, no. 5, pp. 1155–1178, 2007.

[35] L. Wang and Y. Yang, "Training one-class support vector machines in the primal space," in *Electronic Computer Technology, 2009 International Conference on*. IEEE, 2009, pp. 157–160.

[36] A. Navia-Vazquez, F. Perez-Cruz, A. Artes-Rodriguez, and A. R. Figueiras-Vidal, "Weighted least squares training of support vector classifiers leading to compact and adaptive schemes," *IEEE Transactions on Neural Networks*, vol. 12, no. 5, pp. 1047–1059, Sep 2001.

[37] V. Gomez-Verdejo, J. Arenas-Garcia, M. Lazaro-Gredilla, and . Navia-Vazquez, "Adaptive one-class support vector machine," *IEEE Transactions on Signal Processing*, vol. 59, no. 6, pp. 2975–2981, June 2011.

[38] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2165–2176, Aug 2004.

[39] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, "Pegasos: primal estimated sub-gradient solver for svm," *Mathematical Programming*, vol. 127, no. 1, pp. 3–30, 2011.

[40] A. Rahimi, B. Recht *et al.*, "Random features for large-scale kernel machines." in *NIPS*, vol. 3, no. 4, 2007, p. 5.

[41] A. Rahimi and B. Recht, "Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning," in *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds. Curran Associates, Inc., 2009, pp. 1313–1320.

[42] D. J. Sutherland and J. G. Schneider, "On the error of random fourier features," *CoRR*, vol. abs/1506.02785, 2015.

[43] A. Thomas, V. Feuillard, and A. Gramfort, "Calibration of one-class svm for mv set estimation," in *Data Science and Advanced Analytics (DSAA), 2015. 36678 2015. IEEE International Conference on*. IEEE, 2015, pp. 1–9.

[44] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[45] C. Tang and C. Monteleoni, "Convergence rate of stochastic k-means," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, A. Singh and J. Zhu, Eds., vol. 54. Fort Lauderdale, FL, USA: PMLR, 20–22 Apr 2017, pp. 1495–1503.

[46] L. Bottou, "On-line learning in neural networks," D. Saad, Ed. New York, NY, USA: Cambridge University Press, 1998, ch. On-line Learning and Stochastic Approximations, pp. 9–42.

[47] L. Bottou and Y. Bengio, "Convergence properties of the k-means algorithms," in *Advances in Neural Information Processing Systems 7*. MIT Press, 1995, pp. 585–592.

[48] M. Goldstein and S. Uchida, "A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data." *PloS one*, vol. 11 4, p. e0152173, 2016.

[49] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 15:1–15:58, Jul. 2009. [Online]. Available: http://doi.acm.org/10.1145/1541880.1541882

[50] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: Identifying density-based local outliers," *SIGMOD Rec.*, vol. 29, no. 2, pp. 93–104, May 2000. [Online]. Available: http://doi.acm.org/10.1145/335191.335388

[51] J. Tang, Z. Chen, A. W.-C. Fu, and D. W.-L. Cheung, "Enhancing effectiveness of outlier detections for low density patterns," in *Proceedings of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining*, ser. PAKDD '02. London, UK, UK: Springer-Verlag, 2002, pp. 535–548. [Online]. Available: http://dl.acm.org/citation.cfm?id=646420.693665

[52] M. Wu and C. Jermaine, "Outlier detection by sampling with accuracy guarantees," in *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '06. New York, NY, USA: ACM, 2006, pp. 767–772. [Online]. Available: http://doi.acm.org/10.1145/1150402.1150501

[53] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large

spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96. AAAI Press, 1996, pp. 226–231. [Online]. Available: http://dl.acm.org/citation.cfm?id=3001460.3001507

[54] T. Kohonen, M. R. Schroeder, and T. S. Huang, Eds., *Self-Organizing Maps*, 3rd ed. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2001.

[55] Z. He, X. Xu, and S. Deng, "Discovering cluster-based local outliers," *Pattern Recognition Letters*, vol. 24, no. 9, pp. 1641 – 1650, 2003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167865503000035

[56] N. Ye and Q. Chen, "An anomaly detection technique based on a chi-square statistic for detecting intrusions into information systems," *Quality and Reliability Engineering International*, vol. 17, no. 2, pp. 105–112, 3 2001.

[57] P. Galeano, D. Pea, and R. S. Tsay, "Outlier detection in multivariate time series via projection pursuit," Universidad Carlos III de Madrid. Departamento de Estadstica, DES - Working Papers. Statistics and Econometrics. WS, 2004. [Online]. Available: https://EconPapers.repec.org/RePEc:cte:wsrepe:ws044211

[58] D. Agarwal, "Detecting anomalies in cross-classified streams: a bayesian approach," *Knowledge and Information Systems*, vol. 11, no. 1, pp. 29–44, Jan 2007. [Online]. Available: https://doi.org/10.1007/s10115-006-0036-4

[59] S. Ando, "Clustering needles in a haystack: An information theoretic analysis of minority and outlier detection," in *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, Oct 2007, pp. 13–22.

[60] C. C. Noble and D. J. Cook, "Graph-based anomaly detection," in *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '03. New York, NY, USA: ACM, 2003, pp. 631–636. [Online]. Available: http://doi.acm.org/10.1145/956750.956831

[61] S. Lin and D. E. Brown, *An Outlier-based Data Association Method For Linking Criminal Incidents*, pp. 326–330. [Online]. Available: http://epubs.siam.org/doi/abs/10.1137/1.9781611972733.39

[62] H. Dutta, C. Giannella, K. D. Borne, and H. Kargupta, "Distributed top-k outlier detection from astronomy catalogs using the demac system." in *SDM*. SIAM, 2007, pp. 473–478. [Online]. Available: http://dblp.uni-trier.de/db/conf/sdm/sdm2007.html#DuttaGBK07

[63] T. IDÉ and H. KASHIMA, "Eigenspace-based anomaly detection in computer systems," in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '04. New York, NY, USA: ACM, 2004, pp. 440–449. [Online]. Available: http://doi.acm.org/10.1145/1014052.1014102

[64] M. ling Shyu, S. ching Chen, K. Sarinnapakorn, and L. Chang, "A novel anomaly detection scheme based on principal component classifier," in *in Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop, in conjunction with the Third IEEE International Conference on Data Mining (ICDM03*, 2003, pp. 172–179.

[65] S. Hawkins, H. He, G. J. Williams, and R. A. Baxter, "Outlier detection using replicator neural networks," in *Proceedings of the 4th International Conference on Data Warehousing and Knowledge Discovery*, ser. DaWaK 2000. London, UK, UK: Springer-Verlag, 2002, pp. 170–180. [Online]. Available: http://dl.acm.org/citation.cfm?id=646111.679466

[66] D. Janakiram, V. A. Reddy, and A. V. U. P. Kumar, "Outlier detection in wireless sensor networks using bayesian belief networks," in *2006 1st International Conference on Communication Systems Software Middleware*, 2006, pp. 1–6.

[67] G. Tandon and P. K. Chan, "Weighting versus pruning in rule validation for detecting network and host anomalies," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '07. New York, NY, USA: ACM, 2007, pp. 697–706. [Online]. Available: http://doi.acm.org/10.1145/1281192.1281267

[68] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1993.

[69] L. Bordes, C. Delmas, and P. Vandekerkhove, "Semiparametric estimation of a two-component mixture model where one component is known," *Scandinavian Journal of Statistics*, vol. 33, no. 4, pp. 733–752, 2006. [Online]. Available: http://dx.doi.org/10.1111/j.1467-9469.2006.00515.x

[70] "X. contributions to the mathematical theory of evolution.—ii. skew variation in homogeneous material," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 186, pp. 343–414, 1895. [Online]. Available: http://rsta.royalsocietypublishing.org/content/186/343

[71] C. Pana, S. Severi, and G. T. F. de Abreu, "An adaptive approach to non-parametric estimation of dynamic probability density functions," in *2016 13th Workshop on Positioning, Navigation and Communications (WPNC)*, Oct 2016, pp. 1–4.

[72] M. Rosenblatt, "Remarks on some nonparametric estimates of a density function," *Ann. Math. Statist.*, vol. 27, no. 3, pp. 832–837, 09 1956. [Online]. Available: https://doi.org/10.1214/aoms/1177728190

[73] Y.-C. Chen, "A Tutorial on Kernel Density Estimation and Recent Advances," *ArXiv e-prints*, Apr. 2017.

[74] J. Weston, A. Gammerman, M. O. Stitson, V. Vapnik, V. Vovk, C. Watkins, and R. Holloway, "Support vector density estimation," 1999.

[75] V. Vapnik and S. Mukherjee, "Support vector method for multivariate density estimation," *Advances in Neural Information Processing Systems*, pp. 659–665, 2000.

[76] D. Gunopulos, G. Kollios, V. J. Tsotras, and C. Domeniconi, "Approximating multi-dimensional aggregate range queries over real attributes," *SIGMOD Rec.*, vol. 29, no. 2, pp. 463–474, May 2000. [Online]. Available: http://doi.acm.org/10.1145/335191.335448

[77] D. W. Scott and S. R. Sain, "9 - multidimensional density estimation," in *Data Mining and Data Visualization*, ser. Handbook of Statistics, C. Rao, E. Wegman, and J. Solka, Eds. Elsevier, 2005, vol. 24, no. Supplement C, pp. 229 – 261. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0169716104240093

[78] A. Glazer, M. Lindenbaum, and S. Markovitch, "Learning high-density regions for a generalized kolmogorov-smirnov test in high-dimensional data," in *Proceedings of The 26th Conference on Neural Information Processing Systems (NIPS-2012)*, Lake Tahoe, Nevada, 2012. [Online]. Available: http://www.cs.technion.ac.il/~shaulm/papers/pdf/Glazer-Lindenbum-Markovitch-NIPS2012.pdf

[79] C. Park, J. Z. Huang, and Y. Ding, "A computable plug-in estimator of minimum volume sets for novelty detection," *Operations Research*, vol. 58, no. 5, pp. 1469–1480, 2010. [Online]. Available: https://EconPapers.repec.org/RePEc:inm:oropre:v:58:y:2010:i:5:p:1469-1480

[80] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2001.

[81] W. J. Scheirer, L. P. Jain, and T. E. Boult, "Probability models for open set recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2317–2324, Nov 2014.

[82] L. P. Jain, W. J. Scheirer, and T. E. Boult, *Multi-class Open Set Recognition Using Probability of Inclusion*. Cham: Springer International Publishing, 2014, pp. 393–409. [Online]. Available: https://doi.org/10.1007/978-3-319-10578-9_26

[83] M. D. Scherreik and B. D. Rigling, "Open set recognition for automatic target classification with rejection," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 2, pp. 632–642, April 2016.

[84] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[85] S. Kotz and S. Nadarajah, *Extreme Value Distributions: Theory and Applications*. Imperial College Press, 2000. [Online]. Available: https://books.google.com/books?id=b40P_o3yXuUC

[86] W. Polonik, "Minimum volume sets and generalized quantile processes," *Stochastic Processes and their Applications*, vol. 69, no. 1, pp. 1 – 24, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0304414997000288

[87] R. Vert and J.-P. Vert, "Consistency and convergence rates of one-class svms and related algorithms," *J. Mach. Learn. Res.*, vol. 7, pp. 817–854, Dec. 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1248547.1248576

[88] A. Jepson and R. Mann, "Qualitative probabilities for image interpretation," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, 1999, pp. 1123–1130 vol.2.

[89] D. Mllner, "Modern hierarchical, agglomerative clustering algorithms," *CoRR*, vol. abs/1109.2378, 2011. [Online]. Available: http://dblp.uni-trier.de/db/journals/corr/corr1109.html#abs-1109-2378

[90] A. Gordon, "A review of hierarchical classification," vol. 150, pp. 119–137, 01 1987.

[91] F. Murtagh and P. Contreras, "Algorithms for hierarchical clustering: An overview, ii," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, August 2017, article ID: WIDM1219. [Online]. Available: http://eprints.hud.ac.uk/id/eprint/32552/

[92] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: analysis and implementation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 7, pp. 881–892, Jul 2002.

[93] H.-P. Kriegel, P. Krger, J. Sander, and A. Zimek, "Density-based clustering," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, no. 3, pp. 231–240, 2011. [Online]. Available: http://dx.doi.org/10.1002/widm.30

[94] V. Estivill-castro and I. Lee, "Amoeba: Hierarchical clustering based on spatial proximity using delaunay diagram," in *in Proceedings of the 9th International Symposium on Spatial Data Handling*, 2000, pp. 7–26.

[95] J. Yang, V. Estivill-Castro, and S. K. Chalup, "Support vector clustering through proximity graph modelling," in *Neural Information Processing, 2002. ICONIP '02. Proceedings of the 9th International Conference on*, vol. 2, Nov 2002, pp. 898–903 vol.2.

[96] V. Estivill-Castro, I. Lee, and A. T. Murray, *Criteria on Proximity Graphs for Boundary Extraction and Spatial Clustering*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 348–357. [Online]. Available: https://doi.org/10.1007/3-540-45357-1_37

[97] J.-H. Chiang and P.-Y. Hao, "A new kernel-based fuzzy clustering approach: support vector clustering with cell growing," *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 4, pp. 518–527, Aug 2003.

[98] S.-H. Lee and K. M. Daniels, *Cone Cluster Labeling for Support Vector Clustering*, 2006, pp. 484–488. [Online]. Available: http://epubs.siam.org/doi/abs/10.1137/1.9781611972764.45

[99] T. Ban and S. Abe, "Spatially chunking support vector clustering algorithm," in *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541)*, vol. 1, July 2004, p. 418.

[100] C. L. F. Z. W.J. Puma-Villanueva, G.B. Bezerra, "Improving support vector clustering with ensembles," in *Proc. Int'l Joint Conf. Neural Networks*, 2005.

[101] T. Pham, T. Le, and H. Dang, "Scalable support vector clustering using budget," *CoRR*, vol. abs/1709.06444, 2017. [Online]. Available: http://arxiv.org/abs/1709.06444

[102] O. Grygorash, Y. Zhou, and Z. Jorgensen, "Minimum spanning tree based clustering algorithms," in *2006 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06)*, Nov 2006, pp. 73–81.

[103] R. Tarjan, "Depth first search and linear graph algorithms," *SIAM JOURNAL ON COMPUTING*, vol. 1, no. 2, 1972.

[104] R. C. Prim, "Shortest connection networks and some generalizations," *The Bell System Technical Journal*, vol. 36, no. 6, pp. 1389–1401, Nov 1957.

[105] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1998.

[106] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[107] B. D. Rigling and C. Roush, "Acf-based classification of phase modulated waveforms," in *2010 IEEE Radar Conference*, May 2010, pp. 287–291.

[108] B. Caputo, K. Sim, F. Furesjo, and A. Smola, "Appearance-based object recognition using svms: Which kernel should i use?" in *Proc of NIPS workshop on Statistical methods for computational experiments in visual processing and computer vision, Whistler*, vol. 2002, 2002.