The p-Center Based Kernel Machines and Applications

Theodore Trafalis

School of Industrial and Systems Engineering
The University of Oklahoma

Overview

- The Support Vector Machine (SVM) is a powerful tool in machine learning for solving classification and regression problems.
- SVM Real-world applications: text categorization, pattern recognition, weather forecasting, and classifying DNA sequences, well known for its superior generalization ability and practical results.

Overview

- The Support Vector Machine (SVM) is a powerful tool in machine learning for solving classification and regression problems.
- SVM Real-world applications: text categorization, pattern recognition, weather forecasting, and classifying DNA sequences, well known for its superior generalization ability and practical results.

Overview

- The optimal solution of SVMs corresponds to the center of the largest hypersphere that can be inscribed in the set of consistent hypotheses called a version space.
- The SVM solution will be inaccurate if the version space is asymmetric or elongated.
- Several approaches have been proposed to use other possible centers of the version space that can improve the generalization performance.

Overview

- The optimal solution of SVMs corresponds to the center of the largest hypersphere that can be inscribed in the set of consistent hypotheses called a version space.
- The SVM solution will be inaccurate if the version space is asymmetric or elongated.
- Several approaches have been proposed to use other possible centers of the version space that can improve the generalization performance.

Overview

- The optimal solution of SVMs corresponds to the center of the largest hypersphere that can be inscribed in the set of consistent hypotheses called a version space.
- The SVM solution will be inaccurate if the version space is asymmetric or elongated.
- Several approaches have been proposed to use other possible centers of the version space that can improve the generalization performance.

Overview

- The Bayes point machines (BPMs) developed by Ruján (1997), Herbrich (2001), and Minka (2001) are based on an approximation of the center of mass of the version space.
- Trafalis and Malyscheff (2002) proposed the analytic center machine (ACM) based on the utilization of the analytic center of the version space.
- Another approach by Brückner (2005a) based on an approximation of the p-Center of the version space, the so-called p-Center machine (PCM).

Overview

- The Bayes point machines (BPMs) developed by Ruján (1997), Herbrich (2001), and Minka (2001) are based on an approximation of the center of mass of the version space.
- Trafalis and Malyscheff (2002) proposed the analytic center machine (ACM) based on the utilization of the analytic center of the version space.
- Another approach by Brückner (2005a) based on an approximation of the p-Center of the version space, the so-called p-Center machine (PCM).

Overview

- The Bayes point machines (BPMs) developed by Ruján (1997), Herbrich (2001), and Minka (2001) are based on an approximation of the center of mass of the version space.
- Trafalis and Malyscheff (2002) proposed the analytic center machine (ACM) based on the utilization of the analytic center of the version space.
- Another approach by Brückner (2005a) based on an approximation of the p-Center of the version space, the so-called p-Center machine (PCM).

Overview

- Another topic in machine learning is active learning.
- Objective: to choose the data to be labeled and included in the training set.
- Investigated by Campbell et al. (2000), Schohn and Cohn (2000), and Tong and Koller (2001) and applied to various applications.

Overview

- Another topic in machine learning is active learning.
- Objective: to choose the data to be labeled and included in the training set.
- Investigated by Campbell et al. (2000), Schohn and Cohn (2000), and Tong and Koller (2001) and applied to various applications.

Overview

- Another topic in machine learning is active learning.
- Objective: to choose the data to be labeled and included in the training set.
- Investigated by Campbell et al. (2000), Schohn and Cohn (2000), and Tong and Koller (2001) and applied to various applications.

Research Objectives

- The PCM is a relatively new approach and has not been developed and investigated extensively.
- Brückner (2005a) has introduced the formulation for solving binary classification problems based on the p-Center method initially proposed by Moretti (2003).
- We develop the p-Center based algorithms for solving regression and multiclass classification problems.
- We also propose an algorithm for performing active learning with the PCM.

Research Objectives

- The PCM is a relatively new approach and has not been developed and investigated extensively.
- Brückner (2005a) has introduced the formulation for solving binary classification problems based on the p-Center method initially proposed by Moretti (2003).
- We develop the p-Center based algorithms for solving regression and multiclass classification problems.
- We also propose an algorithm for performing active learning with the PCM.

Research Objectives

- The PCM is a relatively new approach and has not been developed and investigated extensively.
- Brückner (2005a) has introduced the formulation for solving binary classification problems based on the p-Center method initially proposed by Moretti (2003).
- We develop the p-Center based algorithms for solving regression and multiclass classification problems.
- We also propose an algorithm for performing active learning with the PCM.

Research Objectives

- The PCM is a relatively new approach and has not been developed and investigated extensively.
- Brückner (2005a) has introduced the formulation for solving binary classification problems based on the p-Center method initially proposed by Moretti (2003).
- We develop the p-Center based algorithms for solving regression and multiclass classification problems.
- We also propose an algorithm for performing active learning with the PCM.

Kernel Methods

- Kernel methods offer a solution by transforming or mapping an input $\mathbf{x}$ from the input space $\mathcal{X}$ into a possibly higher dimensional feature space $\mathcal{F}$ through a feature map $\phi : \mathcal{X} \to \mathcal{F}$ so that the nonlinear problems can be solved linearly.
- By introducing a kernel k, the existence of a feature space $\mathcal{F}$ and a map ϕ are induced.
- A positive definite kernel k can be expressed as $k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \cdot \phi(\mathbf{z}) \rangle$ for all $\mathbf{x}, \mathbf{z} \in \mathcal{X}$.

Kernel Methods

- Kernel methods offer a solution by transforming or mapping an input $\mathbf{x}$ from the input space $\mathcal{X}$ into a possibly higher dimensional feature space $\mathcal{F}$ through a feature map $\phi : \mathcal{X} \to \mathcal{F}$ so that the nonlinear problems can be solved linearly.
- By introducing a kernel k, the existence of a feature space $\mathcal{F}$ and a map ϕ are induced.
- A positive definite kernel k can be expressed as $k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \cdot \phi(\mathbf{z}) \rangle$ for all $\mathbf{x}, \mathbf{z} \in \mathcal{X}$.

Kernel Methods

- Kernel methods offer a solution by transforming or mapping an input $\mathbf{x}$ from the input space $\mathcal{X}$ into a possibly higher dimensional feature space $\mathcal{F}$ through a feature map $\phi : \mathcal{X} \to \mathcal{F}$ so that the nonlinear problems can be solved linearly.
- By introducing a kernel k, the existence of a feature space $\mathcal{F}$ and a map ϕ are induced.
- A positive definite kernel k can be expressed as $k(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}) \cdot \phi(\mathbf{z}) \rangle$ for all $\mathbf{x}$, $\mathbf{z} \in \mathcal{X}$.

Kernel Methods

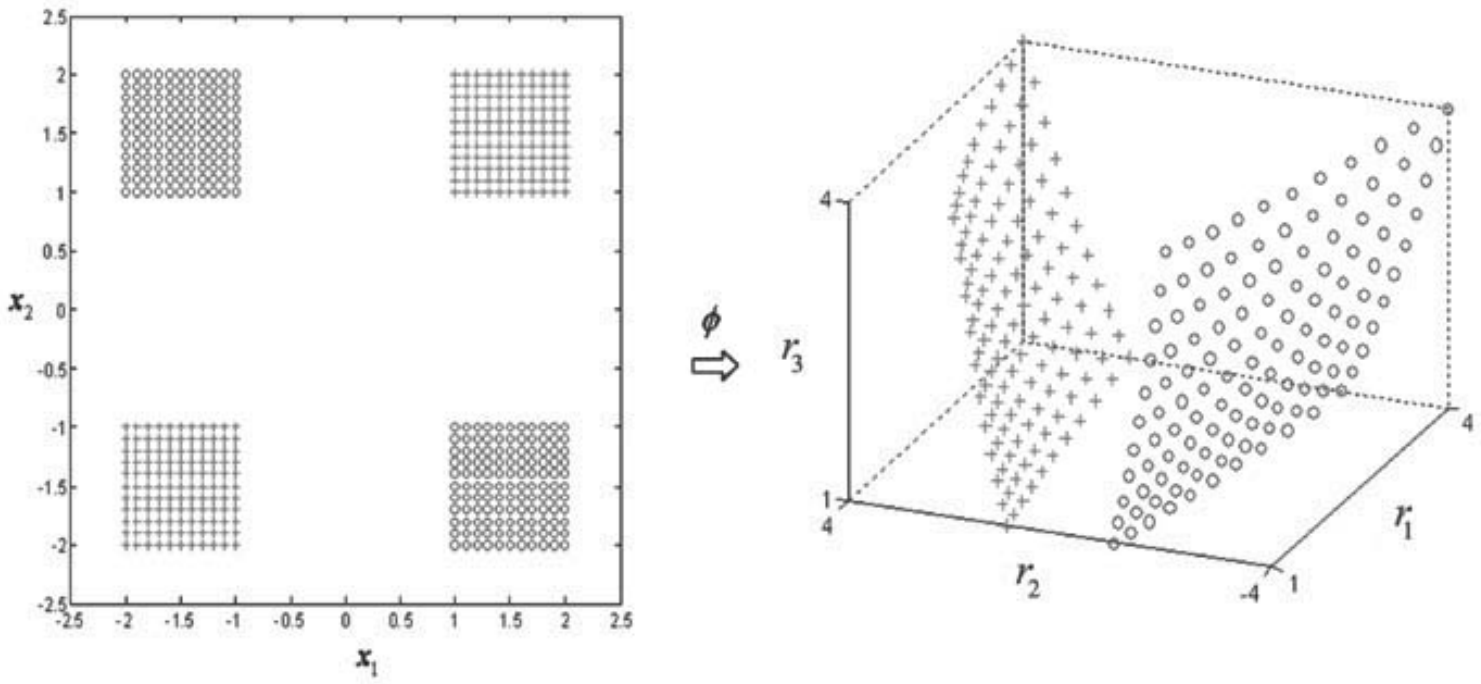

Figure: 1. A kernel map converts a nonlinear separable problem into a linear separable problem in the feature space, (+) belongs to positive class and (o) belongs to negative class.

Support Vector Classification

- In binary classification, the SVM algorithm constructs a hyperplane that separates a set of training vectors into two classes.
- The goals of SVMs are to maximize the margin of separation and minimize misclassification error.

Support Vector Classification

- In binary classification, the SVM algorithm constructs a hyperplane that separates a set of training vectors into two classes.
- The goals of SVMs are to maximize the margin of separation and minimize misclassification error.

Support Vector Classification

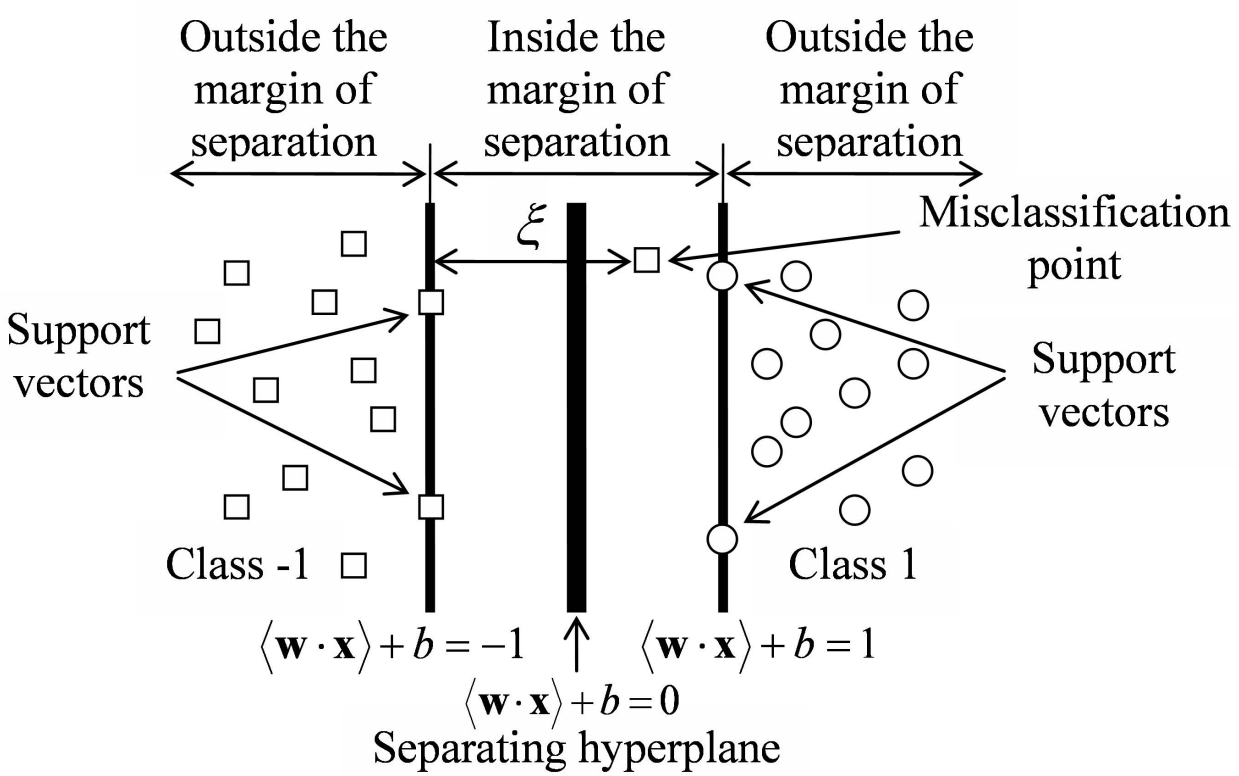

Figure: 2. Illustration of SVMs.

Support Vector Classification

- The SVM formulation can be represented as follows (Vapnik,1998):

$$\begin{aligned} \min \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l}\xi_i \\ \text{Subject to} \quad & y_i(\langle \mathbf{w}\cdot\mathbf{x}_i\rangle + b) \geq 1-\xi_i \\ & \xi_i \geq 0,\ i=1,...,l \end{aligned} \tag{1}$$

where $\mathbf{w}$ is the weight vector, b is a bias, ξ_i is a slack variable, and C is a trade-off parameter.

Support Vector Regression

- Consider a training set $\{\mathbf{x}_i, y_i\}_{i=1}^{l}$ of l instances, our objective is to construct a function for approximating y_i:

$$f(\mathbf{x}_i) = \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \approx y_i \tag{2}$$

  where $\mathbf{w}$ is the weight vector and b is a bias.
- Vapnik (1998) proposed the linear ε-insensitive loss function in the support vector regression (SVR) formulation. The linear ε-insensitive loss function is defined by:

$$L_\varepsilon(\mathbf{x}, y, f) = \begin{cases} 0 & \text{if } |y - f(\mathbf{x})| \leq \varepsilon \\ |y - f(\mathbf{x})| - \varepsilon & \text{otherwise.} \end{cases} \tag{3}$$

Support Vector Regression

- Consider a training set $\{\mathbf{x}_i, y_i\}_{i=1}^{l}$ of l instances, our objective is to construct a function for approximating y_i:

$$f(\mathbf{x}_i) = \langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b \approx y_i \tag{2}$$

  where $\mathbf{w}$ is the weight vector and b is a bias.
- Vapnik (1998) proposed the linear ε-insensitive loss function in the support vector regression (SVR) formulation. The linear ε-insensitive loss function is defined by:

$$L_\varepsilon(\mathbf{x}, y, f) = \begin{cases} 0 & \text{if } |y - f(\mathbf{x})| \leq \varepsilon \\ |y - f(\mathbf{x})| - \varepsilon & \text{otherwise.} \end{cases} \tag{3}$$

Support Vector Regression

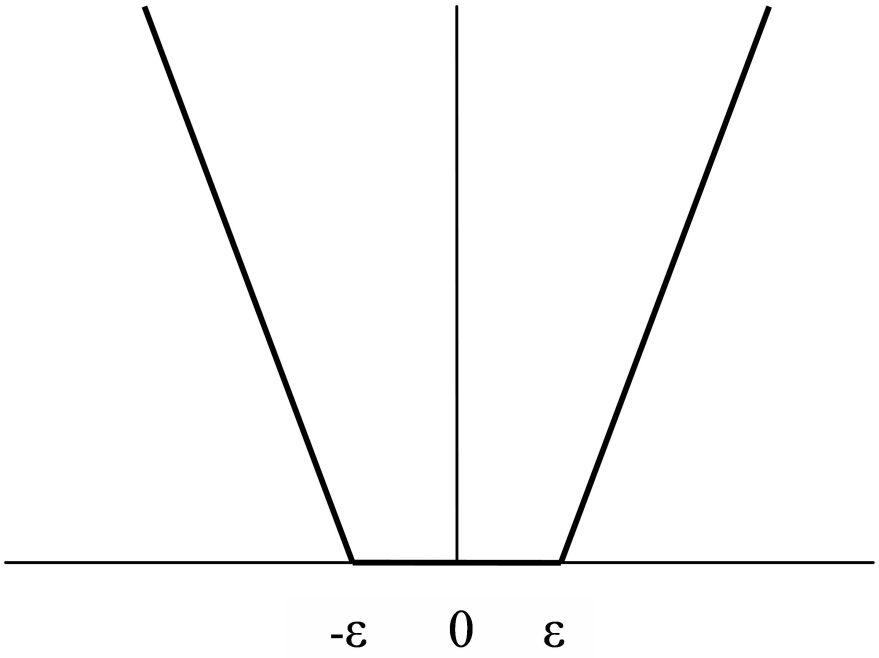

Figure: 3. The ε-insensitive loss function.

Support Vector Regression

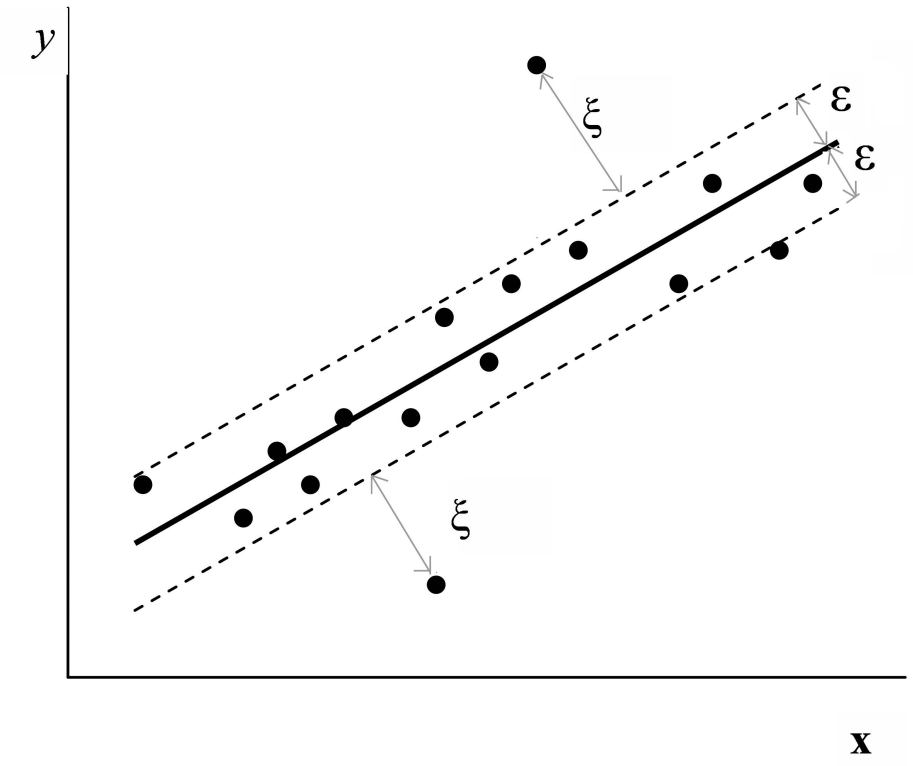

Figure: 4. Linear regression problem with some outliers.

Support Vector Regression

- The SVR formulation can be represented as follows (Vapnik, 1998):

$$\begin{aligned} \min \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i') \\ \text{Subject to} \quad & (\langle \mathbf{w}\cdot\mathbf{x}_i\rangle + b) - y_i \leq \varepsilon + \xi_i \\ & y_i - (\langle \mathbf{w}\cdot\mathbf{x}_i\rangle + b) \leq \varepsilon + \xi_i' \\ & \xi_i, \xi_i' \geq 0,\ i = 1, ..., l \end{aligned} \tag{4}$$

where $\mathbf{w}$ is the weight vector, b is a bias, C is a user-specified parameter, and ξ_i, ξ_i' are slack variables representing the deviations from the constraints of the ε-tube.

Version Space

- Given a training set $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ of l instances, where $\mathbf{x}_1$,..., $\mathbf{x}_l$ are n-dimensional vectors in $\mathcal{X} \subseteq \mathbb{R}^n$, y_i is the corresponding class, and a feature map $\phi : \mathcal{X} \rightarrow \mathcal{F}$, there exists a set of hyperplanes that separate the data in the feature space $\mathcal{F}$.
- This set of consistent hypotheses is called the version space (Mitchell, 1982).
- the version space is defined as (Tong & Koller, 2001):

$$\mathcal{V} = \{\mathbf{w} \in \mathcal{W} | \, \|\mathbf{w}\| = 1, y_i \langle \mathbf{w} \cdot \phi(\mathbf{x}_i) \rangle > 0, \forall i = 1, ..., l.\} \quad (5)$$

where the weight space $\mathcal{W}$ is equal to $\mathcal{F}$.

Version Space

- Given a training set $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ of l instances, where $\mathbf{x}_1$,..., $\mathbf{x}_l$ are n-dimensional vectors in $\mathcal{X} \subseteq \mathbb{R}^n$, y_i is the corresponding class, and a feature map $\phi : \mathcal{X} \rightarrow \mathcal{F}$, there exists a set of hyperplanes that separate the data in the feature space $\mathcal{F}$.
- This set of consistent hypotheses is called the version space (Mitchell, 1982).
- the version space is defined as (Tong & Koller, 2001):

$$\mathcal{V} = \{\mathbf{w} \in \mathcal{W} | \, \|\mathbf{w}\| = 1, y_i \langle \mathbf{w} \cdot \phi(\mathbf{x}_i) \rangle > 0, \forall i = 1, ..., l.\} \quad (5)$$

where the weight space $\mathcal{W}$ is equal to $\mathcal{F}$.

Version Space

- Given a training set $\{(\mathbf{x}_i, y_i)\}_{i=1}^{l}$ of l instances, where $\mathbf{x}_1$,..., $\mathbf{x}_l$ are n-dimensional vectors in $\mathcal{X} \subseteq \mathbb{R}^n$, y_i is the corresponding class, and a feature map $\phi : \mathcal{X} \rightarrow \mathcal{F}$, there exists a set of hyperplanes that separate the data in the feature space $\mathcal{F}$.
- This set of consistent hypotheses is called the version space (Mitchell, 1982).
- the version space is defined as (Tong & Koller, 2001):

$$\mathcal{V} = \{\mathbf{w} \in \mathcal{W} | \, \|\mathbf{w}\| = 1, y_i \langle \mathbf{w} \cdot \phi(\mathbf{x}_i) \rangle > 0, \forall i = 1, ..., l.\} \quad (5)$$

where the weight space $\mathcal{W}$ is equal to $\mathcal{F}$.

Version Space

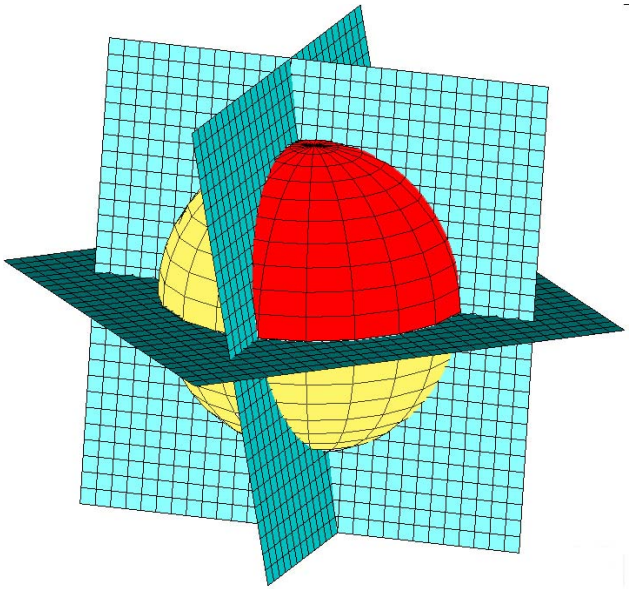

Figure: 5. The surface of the hypersphere represents unit weight vectors. The version space is the surface segment of the hypersphere in red color restricted by hyperplanes. Each hyperplane corresponds to a labeled training instance.

The p-Center Machine

- Moretti (2003) developed an efficient iterative algorithm to find the p-Center of a polytope using a weighted projection centering method.
- Based on Moretti's work (2003), Brückner (2005a) proposed a learning machine for binary classification by estimating the p-Center of the version space, the so-called p-Center machine (PCM).

The p-Center Machine

- Moretti (2003) developed an efficient iterative algorithm to find the p-Center of a polytope using a weighted projection centering method.
- Based on Moretti′s work (2003), Brückner (2005a) proposed a learning machine for binary classification by estimating the p-Center of the version space, the so-called p-Center machine (PCM).

The p-Center Machine

- Given a convex polytope $S = \{\mathbf{x} \in \mathbb{R}^n | \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$ and a feasible interior point $\mathbf{w} \in S \backslash bd(S)$, let $\mathbf{a}_i$ be the ith row of $\mathbf{A}$.
- There exists a straight line that intersects the surface of the polytope twice defined as:

$$\ell_i(\lambda) = \mathbf{w} + \lambda \mathbf{a}_i. \tag{6}$$

The p-Center Machine

- Given a convex polytope $S = \{\mathbf{x} \in \mathbb{R}^n | \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$ and a feasible interior point $\mathbf{w} \in S \backslash bd(S)$, let $\mathbf{a}_i$ be the ith row of $\mathbf{A}$.
- There exists a straight line that intersects the surface of the polytope twice defined as:

$$\ell_i(\lambda) = \mathbf{w} + \lambda \mathbf{a}_i. \tag{6}$$

The p-Center Machine

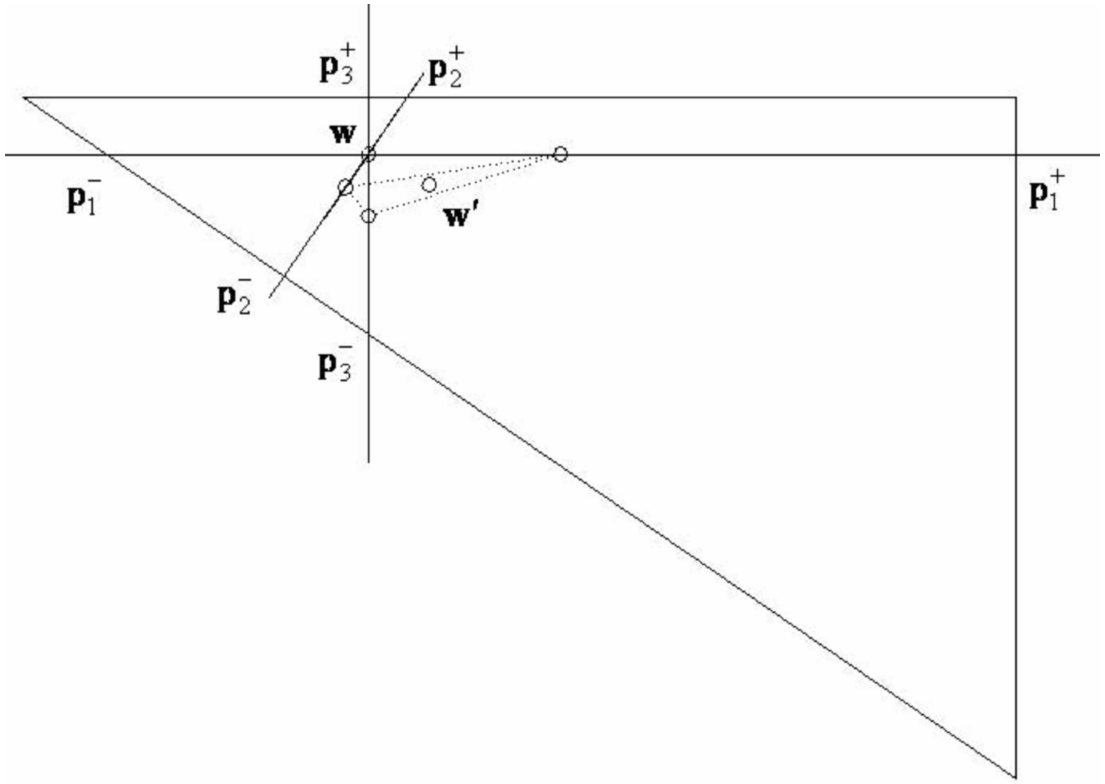

Figure: 6. Illustration of approximating the p-Center of a polytope.

The p-Center Machine

- The intersection points are $\mathbf{p}_i^- = \ell_i(\lambda_i^-)$ and $\mathbf{p}_i^+ = \ell_i(\lambda_i^+)$, where λ_i^- is a negative value and λ_i^+ is a positive value that satisfies the following relations:

$$\begin{aligned}\left\langle \mathbf{a}_j \cdot \mathbf{p}_i^- \right\rangle &= \left\langle \mathbf{a}_j \cdot (\mathbf{w} + \lambda_i^- \mathbf{a}_i) \right\rangle = \left\langle \mathbf{a}_j \cdot \mathbf{w} \right\rangle + \lambda_i^- \left\langle \mathbf{a}_j \cdot \mathbf{a}_i \right\rangle \geq b_j, \forall j \\ \left\langle \mathbf{a}_j \cdot \mathbf{p}_i^+ \right\rangle &= \left\langle \mathbf{a}_j \cdot (\mathbf{w} + \lambda_i^+ \mathbf{a}_i) \right\rangle = \left\langle \mathbf{a}_j \cdot \mathbf{w} \right\rangle + \lambda_i^+ \left\langle \mathbf{a}_j \cdot \mathbf{a}_i \right\rangle \geq b_j, \forall j,\end{aligned} \tag{7}$$

where there exists a k such that:

$$\begin{aligned}\left\langle \mathbf{a}_k \cdot \mathbf{p}_i^- \right\rangle &= \left\langle \mathbf{a}_k \cdot (\mathbf{w} + \lambda_i^- \mathbf{a}_i) \right\rangle = \left\langle \mathbf{a}_k \cdot \mathbf{w} \right\rangle + \lambda_i^- \left\langle \mathbf{a}_k \cdot \mathbf{a}_i \right\rangle = b_k \\ \left\langle \mathbf{a}_k \cdot \mathbf{p}_i^+ \right\rangle &= \left\langle \mathbf{a}_k \cdot (\mathbf{w} + \lambda_i^+ \mathbf{a}_i) \right\rangle = \left\langle \mathbf{a}_k \cdot \mathbf{w} \right\rangle + \lambda_i^+ \left\langle \mathbf{a}_k \cdot \mathbf{a}_i \right\rangle = b_k.\end{aligned} \tag{8}$$

The p-Center Machine

- Based on these conditions, λ_i^- and λ_i^+ can be written as:

$$\lambda_i^- = \min_{j=1,..,l} \{\lambda_j\}, \; \lambda_i^+ = \max_{j=1,..,l} \{\lambda_j\}, \tag{9}$$

where

$$\lambda_j = \frac{b_j - \langle \mathbf{a}_j \cdot \mathbf{w} \rangle}{\langle \mathbf{a}_j \cdot \mathbf{a}_i \rangle}. \tag{10}$$

The p-Center Machine

- Since $\mathbf{p}_i^-$ and $\mathbf{p}_i^+$ are well defined for $i = 1, ..., l$, the p-Center of a given polytope can be calculated iteratively by

$$\begin{aligned} \mathbf{w}_{pCenter} &= \lim_{k \to \infty} \mathbf{w}^k \\ \mathbf{w}^k &= \frac{1}{2l} \sum_{i=1}^{l} (\mathbf{p}_i^- + \mathbf{p}_i^+) = \mathbf{w}^{k-1} + \frac{1}{2l} \sum_{i=1}^{l} (\lambda_i^- + \lambda_i^+) \mathbf{a}_i. \end{aligned} \tag{11}$$

- It was proved by Moretti (2003) that the p-Center exists, and thus it converges.

The p-Center Machine

- Since $\mathbf{p}_i^-$ and $\mathbf{p}_i^+$ are well defined for $i = 1, ..., l$, the p-Center of a given polytope can be calculated iteratively by

$$\begin{aligned} \mathbf{w}_{pCenter} &= \lim_{k\to\infty} \mathbf{w}^k \\ \mathbf{w}^k &= \frac{1}{2l}\sum_{i=1}^{l}(\mathbf{p}_i^- + \mathbf{p}_i^+) = \mathbf{w}^{k-1} + \frac{1}{2l}\sum_{i=1}^{l}(\lambda_i^- + \lambda_i^+)\mathbf{a}_i. \end{aligned} \tag{11}$$

- It was proved by Moretti (2003) that the p-Center exists, and thus it converges.

The p-Center Machine

- Brückner (2005a) modified the work by Moretti (2003) to approximate the p-Center of the version space on a curved polytope (portion of a hypersphere).

The p-Center Machine

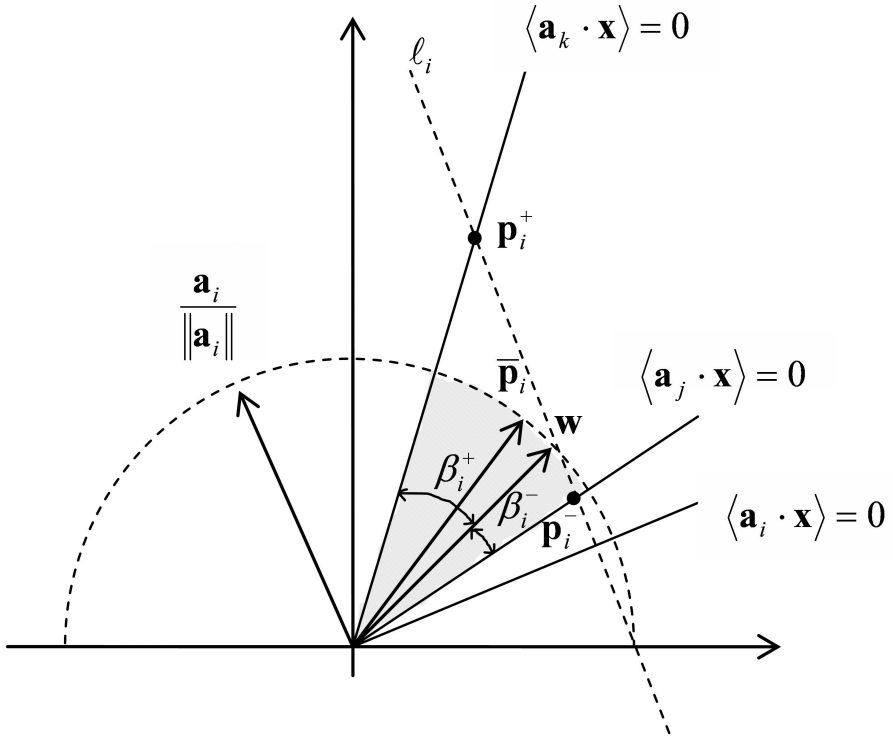

Figure: 7. Illustration of approximating the p-Center of a curved polytope (Brückner, 2005a).

The p-Center Machine

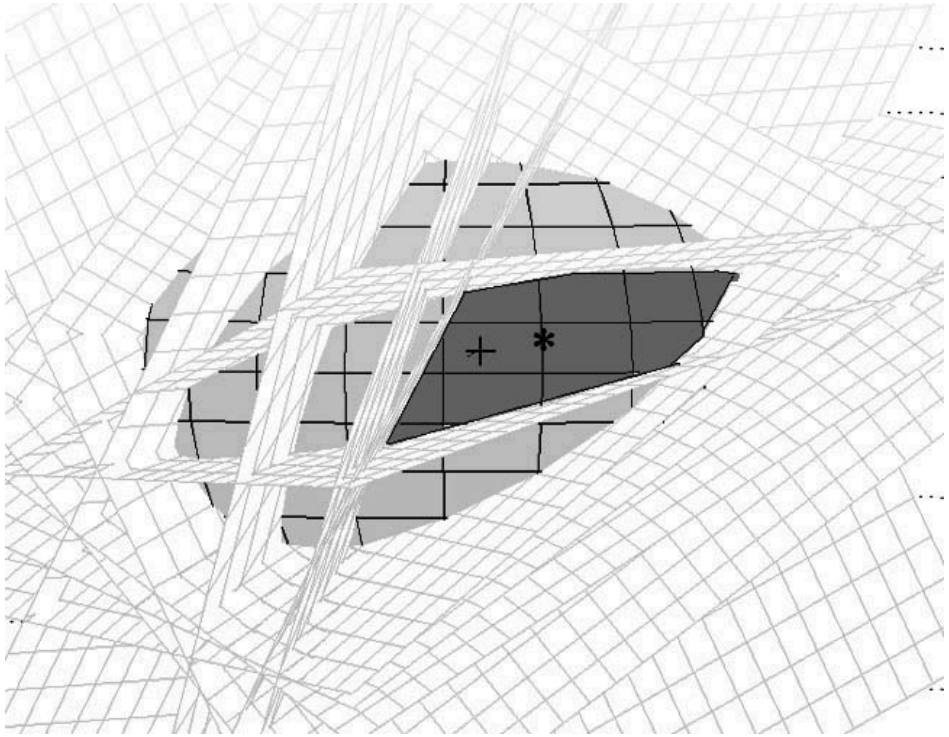

Figure: 8. Comparison of the PCM (*) and SVM (+) solutions in the version space. The classification error rates on the testing set are 8.92% and 14.26% for PCM and SVM, respectively.

The p-Center Machine

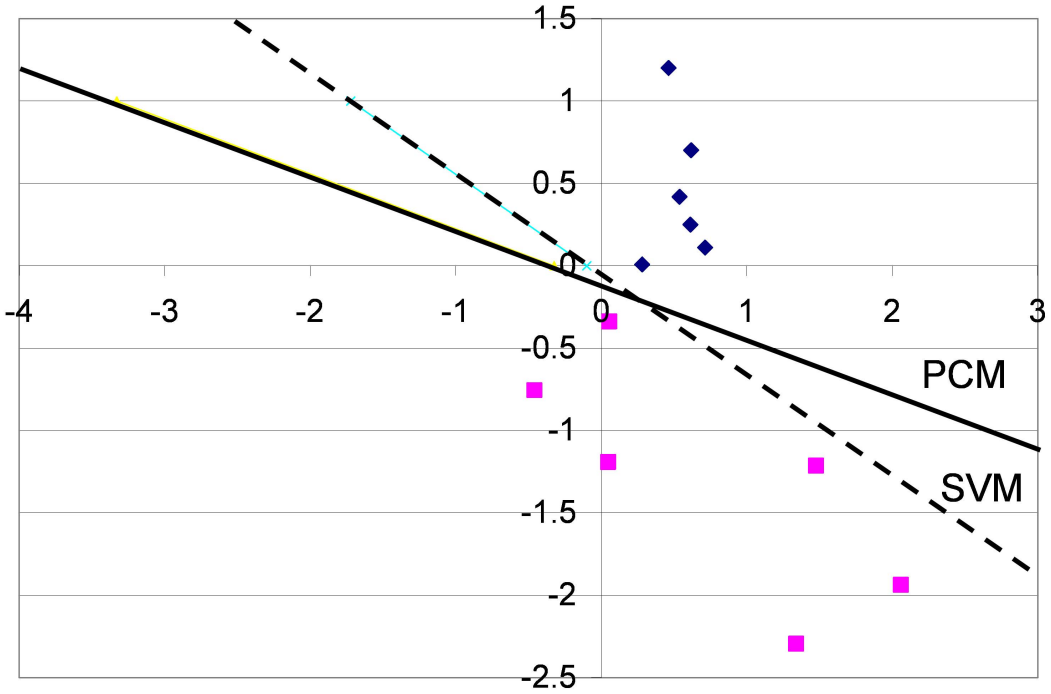

Figure: 9. Comparison of the PCM (*) and SVM (+) solutions in the input space.

The p-Center Machine for Regression

- In this section, the proposed p-Center machine for regression analysis (PCR) is explained.
- Given a training set $\{(\hat{\mathbf{x}}_i, y_i)\}_{i=1}^{l}$ of l instances, our objective is to construct a function $f(\hat{\mathbf{x}}) = \langle \hat{\mathbf{w}} \cdot \hat{\mathbf{x}} \rangle + b$ for approximating y where $\hat{\mathbf{w}}$ is a weight vector and b is a bias.
- The feasibility constraints of a linear regression problem can be defined as:

$$\begin{aligned} &(\langle \hat{\mathbf{w}} \cdot \hat{\mathbf{x}}_i \rangle + b) - y_i \leq \varepsilon \\ &y_i - (\langle \hat{\mathbf{w}} \cdot \hat{\mathbf{x}}_i \rangle + b) \leq \varepsilon, \ i = 1, ..., l. \end{aligned} \tag{12}$$

The p-Center Machine for Regression

- In this section, the proposed p-Center machine for regression analysis (PCR) is explained.
- Given a training set $\{(\hat{\mathbf{x}}_i, y_i)\}_{i=1}^{l}$ of l instances, our objective is to construct a function $f(\hat{\mathbf{x}}) = \langle \hat{\mathbf{w}} \cdot \hat{\mathbf{x}} \rangle + b$ for approximating y where $\hat{\mathbf{w}}$ is a weight vector and b is a bias.
- The feasibility constraints of a linear regression problem can be defined as:

$$\begin{aligned} (\langle \hat{\mathbf{w}} \cdot \hat{\mathbf{x}}_i \rangle + b) - y_i &\leq \varepsilon \\ y_i - (\langle \hat{\mathbf{w}} \cdot \hat{\mathbf{x}}_i \rangle + b) &\leq \varepsilon, \ i = 1, ..., l. \end{aligned} \tag{12}$$

The p-Center Machine for Regression

- In this section, the proposed p-Center machine for regression analysis (PCR) is explained.
- Given a training set $\{(\hat{\mathbf{x}}_i, y_i)\}_{i=1}^{l}$ of l instances, our objective is to construct a function $f(\hat{\mathbf{x}}) = \langle \hat{\mathbf{w}} \cdot \hat{\mathbf{x}} \rangle + b$ for approximating y where $\hat{\mathbf{w}}$ is a weight vector and b is a bias.
- The feasibility constraints of a linear regression problem can be defined as:

$$\begin{aligned} &(\langle \hat{\mathbf{w}} \cdot \hat{\mathbf{x}}_i \rangle + b) - y_i \leq \varepsilon \\ &y_i - (\langle \hat{\mathbf{w}} \cdot \hat{\mathbf{x}}_i \rangle + b) \leq \varepsilon, \ i = 1, ..., l. \end{aligned} \tag{12}$$

The p-Center Machine for Regression

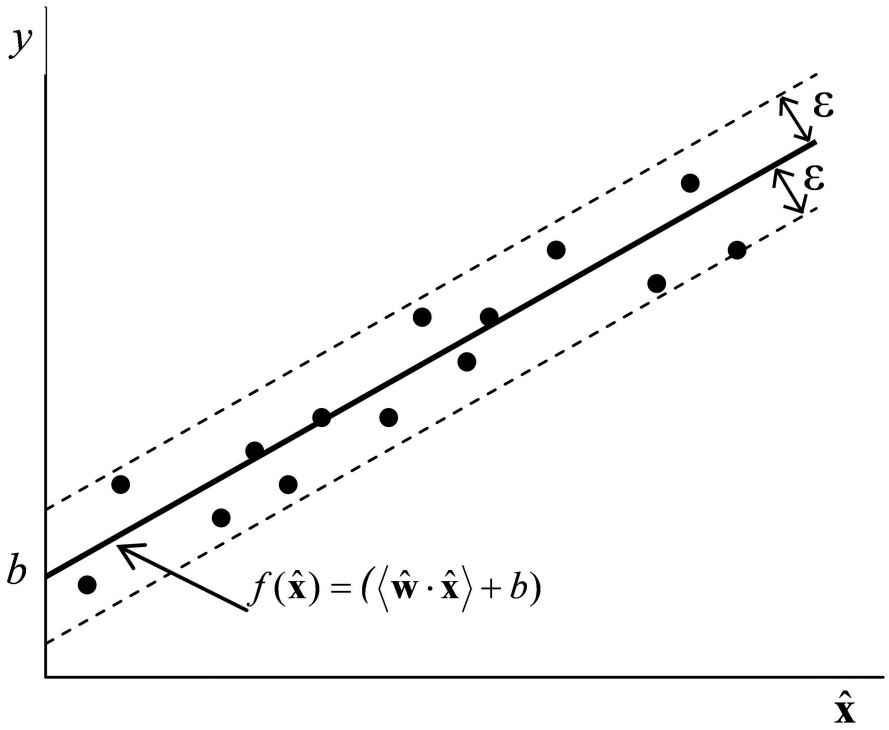

Figure: 10. General linear regression problem.

The p-Center Machine for Regression

- Any solution that satisfies the feasibility constraints provides feasible solutions for a regression problem. For simplicity, let $\mathbf{w} = (\hat{\mathbf{w}}, b) \in \mathbb{R}^{n+1}$ and $\mathbf{x}_i = (\hat{\mathbf{x}}_i, 1) \in \mathbb{R}^{n+1}$.
- Then, the feasibility constraints become:

$$\begin{aligned} \langle \mathbf{w} \cdot \mathbf{x}_i \rangle - y_i &\leq \varepsilon \\ y_i - \langle \mathbf{w} \cdot \mathbf{x}_i \rangle &\leq \varepsilon, \ i = 1, ..., l. \end{aligned} \tag{13}$$

- These constraints can be rewritten as:

$$\begin{aligned} -\langle \mathbf{w} \cdot \mathbf{x}_i \rangle &\geq -y_i - \varepsilon \\ \langle \mathbf{w} \cdot \mathbf{x}_i \rangle &\geq y_i - \varepsilon, \ i = 1, ..., l. \end{aligned} \tag{14}$$

The p-Center Machine for Regression

- Any solution that satisfies the feasibility constraints provides feasible solutions for a regression problem. For simplicity, let $\mathbf{w} = (\hat{\mathbf{w}}, b) \in \mathbb{R}^{n+1}$ and $\mathbf{x}_i = (\hat{\mathbf{x}}_i, 1) \in \mathbb{R}^{n+1}$.
- Then, the feasibility constraints become:

$$\begin{aligned} &\langle \mathbf{w} \cdot \mathbf{x}_i \rangle - y_i \leq \varepsilon \\ &y_i - \langle \mathbf{w} \cdot \mathbf{x}_i \rangle \leq \varepsilon, \ i = 1, ..., l. \end{aligned} \tag{13}$$

- These constraints can be rewritten as:

$$\begin{aligned} -\langle \mathbf{w} \cdot \mathbf{x}_i \rangle &\geq -y_i - \varepsilon \\ \langle \mathbf{w} \cdot \mathbf{x}_i \rangle &\geq y_i - \varepsilon, \ i = 1, ..., l. \end{aligned} \tag{14}$$

The p-Center Machine for Regression

- Any solution that satisfies the feasibility constraints provides feasible solutions for a regression problem. For simplicity, let $\mathbf{w} = (\hat{\mathbf{w}}, b) \in \mathbb{R}^{n+1}$ and $\mathbf{x}_i = (\hat{\mathbf{x}}_i, 1) \in \mathbb{R}^{n+1}$.
- Then, the feasibility constraints become:

$$\begin{aligned} &\langle \mathbf{w} \cdot \mathbf{x}_i \rangle - y_i \leq \varepsilon \\ &y_i - \langle \mathbf{w} \cdot \mathbf{x}_i \rangle \leq \varepsilon, \ i = 1, ..., l. \end{aligned} \tag{13}$$

- These constraints can be rewritten as:

$$\begin{aligned} -\langle \mathbf{w} \cdot \mathbf{x}_i \rangle &\geq -y_i - \varepsilon \\ \langle \mathbf{w} \cdot \mathbf{x}_i \rangle &\geq y_i - \varepsilon, \ i = 1, ..., l. \end{aligned} \tag{14}$$

Kernelization

- For nonlinear problems, given a feature map $\phi : \mathcal{X} \rightarrow \mathcal{F}$, the feasibility constraints can be defined as:

$$\begin{aligned} -\langle \mathbf{w} \cdot \phi(\mathbf{x}_i) \rangle &\geq -y_i - \varepsilon \\ \langle \mathbf{w} \cdot \phi(\mathbf{x}_i) \rangle &\geq y_i - \varepsilon, \ i = 1, ..., l. \end{aligned} \tag{15}$$

Kernelization

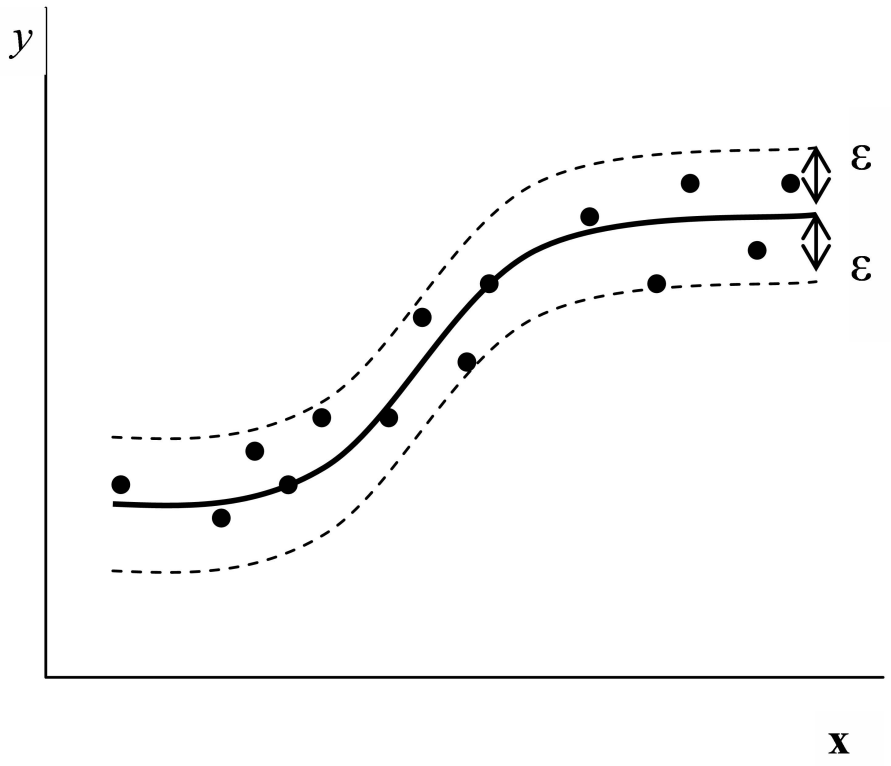

Figure: 11. Nonlinear regression problem.

Kernelization

- Given a kernel k with $k\left(\mathbf{x}_i, \mathbf{x}_j\right) = \left\langle \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) \right\rangle$, a vector of Lagrange multipliers $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, ..., \alpha_l, b]^T$, and $\mathbf{w} = \sum_{i=1}^{l} \alpha_i \phi(\mathbf{x}_i)$, the feasible constraints can be described as:

$$\begin{aligned} -\left\langle \mathbf{k}_i \cdot \boldsymbol{\alpha} \right\rangle &\geq -y_i - \varepsilon \\ \left\langle \mathbf{k}_i \cdot \boldsymbol{\alpha} \right\rangle &\geq y_i - \varepsilon, \end{aligned} \tag{16}$$

where $\mathbf{k}_i = [k_{1i}, k_{2i}, ..., k_{li}, 1]^T$, $k_i \in \mathbb{R}^{l+1}$, $i = 1, ..., l$.

Kernelization

- Since the feasible region is unbounded, the additional constraints to restrict the feasible region are needed.
- In our approach, we use the same approach proposed by Malyscheff and Trafalis (2002) to isolate the solution in the feasible region.

Kernelization

- Since the feasible region is unbounded, the additional constraints to restrict the feasible region are needed.
- In our approach, we use the same approach proposed by Malyscheff and Trafalis (2002) to isolate the solution in the feasible region.

Kernelization

- Given the vectors $\mathbf{q}_i$ that form a basis for the nullspace of the kernel matrix $\mathbf{K}$ and v is the dimension of this nullspace, we can define $\mathbf{Q} = [\mathbf{q}_1, ..., \mathbf{q}_v] \in \mathbb{R}^{(l+1)\times v}$.
- Hence, we have the additional constraints:

$$\begin{aligned} -\langle \mathbf{q}_i \cdot \boldsymbol{\alpha} \rangle &\geq \varepsilon \\ \langle \mathbf{q}_i \cdot \boldsymbol{\alpha} \rangle &\geq \varepsilon, \; i = 1, ..., v. \end{aligned} \tag{17}$$

Kernelization

- Given the vectors $\mathbf{q}_i$ that form a basis for the nullspace of the kernel matrix $\mathbf{K}$ and v is the dimension of this nullspace, we can define $\mathbf{Q} = [\mathbf{q}_1, ..., \mathbf{q}_v] \in \mathbb{R}^{(l+1)\times v}$.
- Hence, we have the additional constraints:

$$\begin{aligned} -\langle \mathbf{q}_i \cdot \boldsymbol{\alpha} \rangle &\geq \varepsilon \\ \langle \mathbf{q}_i \cdot \boldsymbol{\alpha} \rangle &\geq \varepsilon, \ i = 1, ..., v. \end{aligned} \tag{17}$$

Finding an Initial Feasible Solution

- In order to find an initial feasible solution, we need to solve the following LP problem:

$$\begin{aligned} \min \quad & \varepsilon \\ \text{Subject to} \quad & \mathbf{k}_i^T \boldsymbol{\alpha} \geq y_i - \varepsilon \\ & -\mathbf{k}_i^T \boldsymbol{\alpha} \geq -y_i - \varepsilon \\ & \mathbf{q}_i^T \boldsymbol{\alpha} \geq -\varepsilon \\ & -\mathbf{q}_i^T \boldsymbol{\alpha} \geq -\varepsilon, \ i = 1, ..., v. \end{aligned} \tag{18}$$

- From this LP problem, we denote the solution as $\boldsymbol{\alpha}_{init}$ and the minimum value of the objective function as ε_{min}.

The p-Center Approach

- The feasible constraints described can be rewritten in a matrix form as:

$$\begin{bmatrix} \mathbf{K}^T \\ -\mathbf{K}^T \\ \mathbf{Q}^T \\ -\mathbf{Q}^T \end{bmatrix} \boldsymbol{\alpha} \geq \begin{bmatrix} \mathbf{Y} - \varepsilon \mathbf{1} \\ -\mathbf{Y} - \varepsilon \mathbf{1} \\ -\varepsilon \mathbf{1} \\ -\varepsilon \mathbf{1} \end{bmatrix}, \tag{19}$$

where $\mathbf{K} = [\mathbf{k}_1, \mathbf{k}_2, ..., \mathbf{k}_l] \in \mathbb{R}^{(l+1)\times l}$, $\mathbf{Q} = [\mathbf{q}_1, ..., \mathbf{q}_v] \in \mathbb{R}^{(l+1)\times v}$, $\mathbf{Y}$=$[y_1, y_2, ..., y_l] \in \mathbb{R}$, and $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, ..., \alpha_l, b]^T$, for $i = 1, ..., l$.

The p-Center Approach

- Since this form is equivalent to the one of a n-dimensional convex polytope $S = \{\mathbf{x} \in \mathbb{R}^n | \mathbf{A}\mathbf{x} \geq \mathbf{b}\}$, we can obtain the solution by finding the p-Center of a convex polytope using the initial LP solution, α_{init}.

Experiments

- Suppose we would like to approximate the following function:

$$f(x) = \frac{\sin(10x)}{10x}. \tag{20}$$

Results

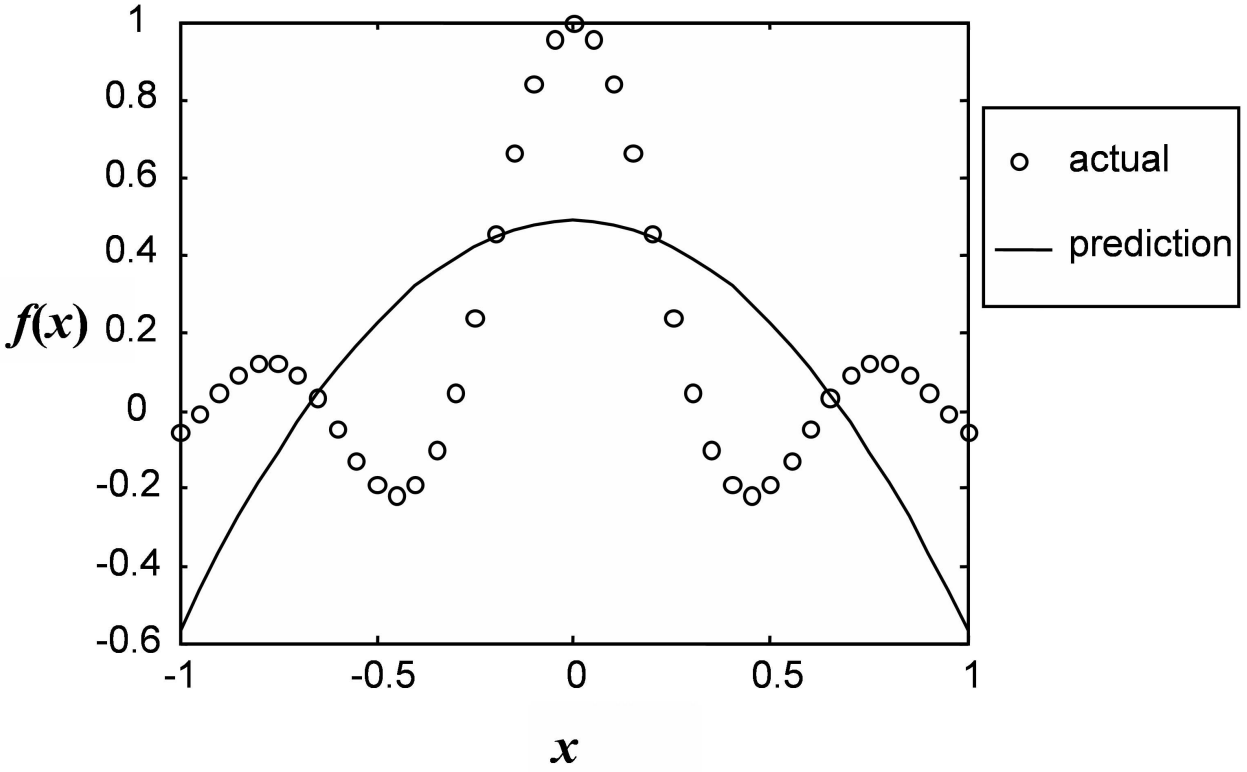

Figure: 12. The p-Center machine for regression using a 2nd degree polynomial kernel, and $\varepsilon = 0.5102$.

Results

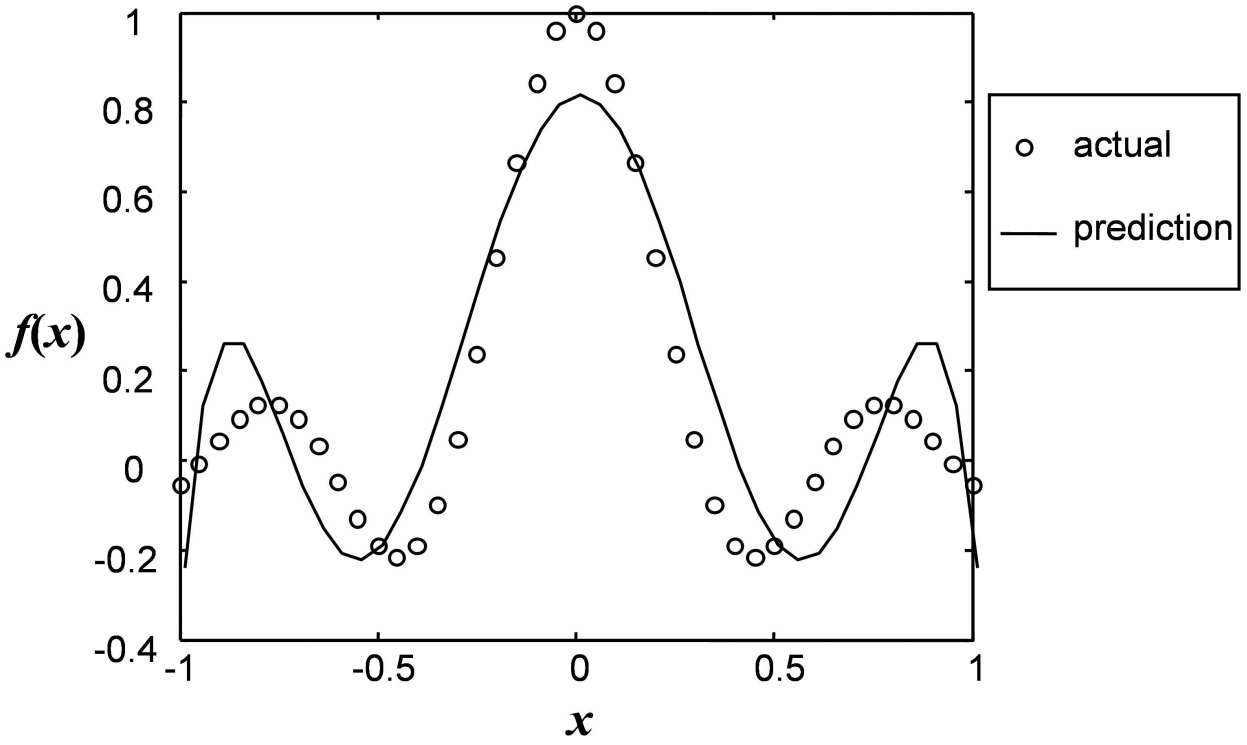

Figure: 13. The p-Center machine for regression using a 6th degree polynomial kernel, and $\varepsilon = 0.2004$.

Results

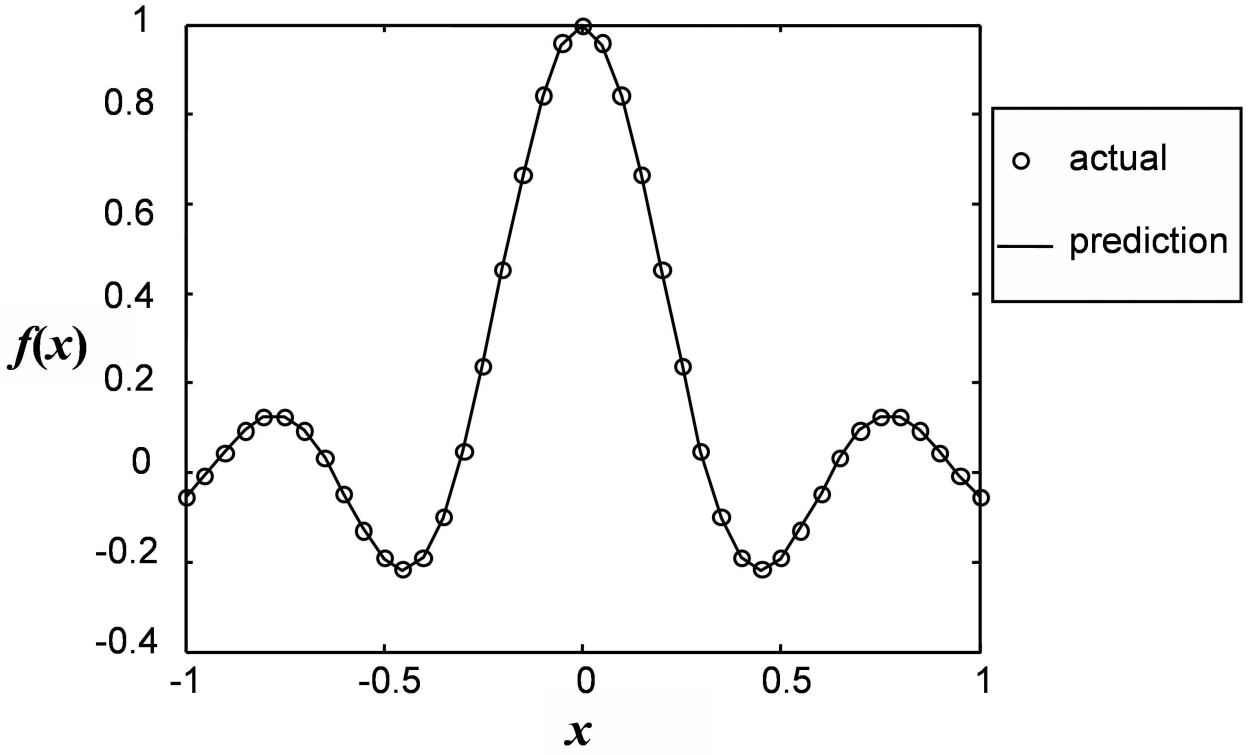

Figure: 14. The p-Center machine for regression using a 12th degree polynomial kernel and $\varepsilon = 0.0018$.

Experiments

Table: 1. Data sets used in the experiments taken from the DELVE (Data for Evaluating Learning in Valid Experiments) project at the Univesity of Toronto, Canada (Rasmussen et al., 1996).

Data Set	Total Instances	Training Instances	Testing Instances	Number of Attributes
Boston	506	304	202	13
Pumadyn	8192	128	8064	8
Add10	9792	256	9536	10

Results

Table: 2. Results on the testing set after 100 runs using the Boston data set with 95% confidence intervals.

Method	Kernel	Parameters	MSE
PCR	Multiquadric	$\sigma = 1$	25.15 ± 0.65
SVR	Polynomial	$p = 1,\ \varepsilon = 4,\ C = 1$	25.51 ± 0.85
Stepwise Regression	-	-	25.47 ± 0.71

Results

Table: 3. Results on the testing set after 100 runs using the Pumadyn data set with 95% confidence intervals.

Method	Kernel	Parameters	MSE
PCR	Multiquadric	$\sigma = 4$	6.91 ± 0.22
SVR	RBF	$\gamma = 0.1,\ \varepsilon = 1,\ C = 100$	6.97 ± 0.15
Stepwise Regression	-	-	14.92 ± 0.09

Results

Table: 4. Results on the testing set after 100 runs using the Add10 data set with 95% confidence intervals.

Method	Kernel	Parameters	MSE
PCR	Multiquadric	$\sigma = 1$	3.63 ± 0.05
SVR	RBF	$\gamma = 0.5,\ \varepsilon = 1,\ C = 100$	3.66 ± 0.04
Stepwise Regression	-	-	7.17 ± 0.03

One-against-all Method

- Using the one-against-all (OAA) method, for an m-class classification problem, we construct m binary classifiers where $f_k,\ k = 1, ..., m$, separates instances of the class k from all other instances.
- We train f_k such that the kth classifier provides separating surface that separates between the instances in the kth class ($f_k(\mathbf{x}) > 0$) and all other instances in the rest of the classes ($f_k(\mathbf{x}) < 0$).
- Then, the class of instance $\mathbf{x}_i$ corresponds to the maximal value of the function $f_k(\mathbf{x}_i), k = 1, ..., m$:

$$c = \arg\max_{k=1,..,m} \{f_k(\mathbf{x}_i)\}, \tag{21}$$

where c is the class of $\mathbf{x}_i$.

One-against-all Method

- Using the one-against-all (OAA) method, for an m-class classification problem, we construct m binary classifiers where $f_k,\ k = 1, ..., m$, separates instances of the class k from all other instances.
- We train f_k such that the kth classifier provides separating surface that separates between the instances in the kth class ($f_k(\mathbf{x}) > 0$) and all other instances in the rest of the classes ($f_k(\mathbf{x}) < 0$).
- Then, the class of instance $\mathbf{x}_i$ corresponds to the maximal value of the function $f_k(\mathbf{x}_i), k = 1, ..., m$:

$$c = \arg\max_{k=1,..,m} \{f_k(\mathbf{x}_i)\}, \tag{21}$$

where c is the class of $\mathbf{x}_i$.

One-against-all Method

- Using the one-against-all (OAA) method, for an m-class classification problem, we construct m binary classifiers where $f_k,\ k = 1, ..., m$, separates instances of the class k from all other instances.
- We train f_k such that the kth classifier provides separating surface that separates between the instances in the kth class ($f_k(\mathbf{x}) > 0$) and all other instances in the rest of the classes ($f_k(\mathbf{x}) < 0$).
- Then, the class of instance $\mathbf{x}_i$ corresponds to the maximal value of the function $f_k(\mathbf{x}_i), k = 1, ..., m$:

$$c = \arg\max_{k=1,..,m} \{f_k(\mathbf{x}_i)\}, \tag{21}$$

where c is the class of $\mathbf{x}_i$.

One-against-all Method

- Given an m-class classification problem, we construct the set of functions:

$$f_k(\mathbf{x}) = \left\langle \mathbf{w}^k \cdot \phi(\mathbf{x}) \right\rangle, \; k = 1, ..., m. \tag{22}$$

- Hence, for $k = 1, ..., m$, the version space becomes:

$$\mathcal{V} = \left\{ \mathbf{w}^k \in \mathcal{W} | y_i \left\langle \mathbf{w}^k \cdot \phi(\mathbf{x}_i) \right\rangle > 0, \; i = 1, ..., l \right\}. \tag{23}$$

- Using this version space, we can find the solution by finding the p-Center of a curved polytope.

One-against-all Method

- Given an m-class classification problem, we construct the set of functions:

$$f_k(\mathbf{x}) = \left\langle \mathbf{w}^k \cdot \phi(\mathbf{x}) \right\rangle, \; k = 1, ..., m. \tag{22}$$

- Hence, for $k = 1, ..., m$, the version space becomes:

$$\mathcal{V} = \left\{ \mathbf{w}^k \in \mathcal{W} | y_i \left\langle \mathbf{w}^k \cdot \phi(\mathbf{x}_i) \right\rangle > 0, \; i = 1, ..., l \right\}. \tag{23}$$

- Using this version space, we can find the solution by finding the p-Center of a curved polytope.

One-against-all Method

- Given an m-class classification problem, we construct the set of functions:

$$f_k(\mathbf{x}) = \left\langle \mathbf{w}^k \cdot \phi(\mathbf{x}) \right\rangle, \; k = 1, ..., m. \tag{22}$$

- Hence, for $k = 1, ..., m$, the version space becomes:

$$\mathcal{V} = \left\{ \mathbf{w}^k \in \mathcal{W} | y_i \left\langle \mathbf{w}^k \cdot \phi(\mathbf{x}_i) \right\rangle > 0, \; i = 1, ..., l \right\}. \tag{23}$$

- Using this version space, we can find the solution by finding the p-Center of a curved polytope.

One-against-one Method

- For an m-class classification problem, the one-against-one (OAO) method constructs $m(m-1)/2$ classifiers, where each classifier is trained on the training set from two classes.
- We want to find the decision functions for all combinations of class pairs.
- Given an m-class classification problem, the decision functions for class q against class r can be written as:

$$f_{qr}(\mathbf{x}) = \langle \mathbf{w}^{qr} \cdot \phi(\mathbf{x}) \rangle, \; q = 1, ..., m, \; r = 1, ..., m, \; q \neq r. \tag{24}$$

One-against-one Method

- For an m-class classification problem, the one-against-one (OAO) method constructs $m(m-1)/2$ classifiers, where each classifier is trained on the training set from two classes.
- We want to find the decision functions for all combinations of class pairs.
- Given an m-class classification problem, the decision functions for class q against class r can be written as:

$$f_{qr}(\mathbf{x}) = \langle \mathbf{w}^{qr} \cdot \phi(\mathbf{x}) \rangle, \; q = 1, ..., m, \; r = 1, ..., m, \; q \neq r. \tag{24}$$

One-against-one Method

- For an m-class classification problem, the one-against-one (OAO) method constructs $m(m-1)/2$ classifiers, where each classifier is trained on the training set from two classes.
- We want to find the decision functions for all combinations of class pairs.
- Given an m-class classification problem, the decision functions for class q against class r can be written as:

$$f_{qr}(\mathbf{x}) = \langle \mathbf{w}^{qr} \cdot \phi(\mathbf{x}) \rangle, \; q = 1, ..., m, \; r = 1, ..., m, \; q \neq r. \tag{24}$$

One-against-one Method

- Thus, for $q = 1, ..., m,\ r = 1, ..., m,\ q \neq r$ the version space becomes:

$$\mathcal{V} = \{\mathbf{w}^{qr} \in \mathcal{W} | y_i \langle \mathbf{w}^{qr} \cdot \phi(\mathbf{x}_i) \rangle > 0,\ i = 1, ..., l.\} \qquad (25)$$

- Using this version space, we can obtain the solution by finding the p-Center of a curved polytope.
- After all classifiers are determined, we apply a max-vote strategy to decide the class of the corresponding data.
- For the instance $\mathbf{x}_i$, if $\text{sign}(f_{qr}(\mathbf{x}_i)) > 0$ or $\mathbf{x}_i$ is in the qth class, the vote for the qth class is increased by one. Otherwise, the vote for the rth class is added by one.
- Then, we classify the instance $\mathbf{x}_i$ into the class with the highest vote.

One-against-one Method

- Thus, for $q = 1, ..., m,\ r = 1, ..., m,\ q \neq r$ the version space becomes:

$$\mathcal{V} = \{\mathbf{w}^{qr} \in \mathcal{W} | y_i \langle \mathbf{w}^{qr} \cdot \phi(\mathbf{x}_i) \rangle > 0,\ i = 1, ..., l.\} \qquad (25)$$

- Using this version space, we can obtain the solution by finding the p-Center of a curved polytope.
- After all classifiers are determined, we apply a max-vote strategy to decide the class of the corresponding data.
- For the instance $\mathbf{x}_i$, if $\text{sign}(f_{qr}(\mathbf{x}_i)) > 0$ or $\mathbf{x}_i$ is in the qth class, the vote for the qth class is increased by one. Otherwise, the vote for the rth class is added by one.
- Then, we classify the instance $\mathbf{x}_i$ into the class with the highest vote.

One-against-one Method

- Thus, for $q = 1, ..., m,\ r = 1, ..., m,\ q \neq r$ the version space becomes:

$$\mathcal{V} = \{\mathbf{w}^{qr} \in \mathcal{W} | y_i \langle \mathbf{w}^{qr} \cdot \phi(\mathbf{x}_i) \rangle > 0,\ i = 1, ..., l.\} \qquad (25)$$

- Using this version space, we can obtain the solution by finding the p-Center of a curved polytope.
- After all classifiers are determined, we apply a max-vote strategy to decide the class of the corresponding data.
- For the instance $\mathbf{x}_i$, if $\text{sign}(f_{qr}(\mathbf{x}_i)) > 0$ or $\mathbf{x}_i$ is in the qth class, the vote for the qth class is increased by one. Otherwise, the vote for the rth class is added by one.
- Then, we classify the instance $\mathbf{x}_i$ into the class with the highest vote.

One-against-one Method

- Thus, for $q = 1, ..., m,\ r = 1, ..., m,\ q \neq r$ the version space becomes:

$$\mathcal{V} = \{\mathbf{w}^{qr} \in \mathcal{W} | y_i \langle \mathbf{w}^{qr} \cdot \phi(\mathbf{x}_i) \rangle > 0,\ i = 1, ..., l.\} \qquad (25)$$

- Using this version space, we can obtain the solution by finding the p-Center of a curved polytope.
- After all classifiers are determined, we apply a max-vote strategy to decide the class of the corresponding data.
- For the instance $\mathbf{x}_i$, if $\text{sign}(f_{qr}(\mathbf{x}_i)) > 0$ or $\mathbf{x}_i$ is in the qth class, the vote for the qth class is increased by one. Otherwise, the vote for the rth class is added by one.
- Then, we classify the instance $\mathbf{x}_i$ into the class with the highest vote.

One-against-one Method

- Thus, for $q = 1, ..., m,\ r = 1, ..., m,\ q \neq r$ the version space becomes:

$$\mathcal{V} = \{\mathbf{w}^{qr} \in \mathcal{W} | y_i \langle \mathbf{w}^{qr} \cdot \phi(\mathbf{x}_i) \rangle > 0,\ i = 1, ..., l.\} \qquad (25)$$

- Using this version space, we can obtain the solution by finding the p-Center of a curved polytope.
- After all classifiers are determined, we apply a max-vote strategy to decide the class of the corresponding data.
- For the instance $\mathbf{x}_i$, if $\text{sign}(f_{qr}(\mathbf{x}_i)) > 0$ or $\mathbf{x}_i$ is in the qth class, the vote for the qth class is increased by one. Otherwise, the vote for the rth class is added by one.
- Then, we classify the instance $\mathbf{x}_i$ into the class with the highest vote.

DDAG

- For an m-class classification problem, this method constructs $m(m-1)/2$ classifiers.
- At each node, a binary classifier is constructed.
- The advantages of DDAG are fast computational time especially for large-scale problems and a risk bound (Platt et al., 2000).
- We propose to use the p-Center approach for constructing a binary classifier for each DDAG node.

DDAG

- For an m-class classification problem, this method constructs $m(m-1)/2$ classifiers.
- At each node, a binary classifier is constructed.
- The advantages of DDAG are fast computational time especially for large-scale problems and a risk bound (Platt et al., 2000).
- We propose to use the p-Center approach for constructing a binary classifier for each DDAG node.

DDAG

- For an m-class classification problem, this method constructs $m(m-1)/2$ classifiers.
- At each node, a binary classifier is constructed.
- The advantages of DDAG are fast computational time especially for large-scale problems and a risk bound (Platt et al., 2000).
- We propose to use the p-Center approach for constructing a binary classifier for each DDAG node.

DDAG

- For an m-class classification problem, this method constructs $m(m-1)/2$ classifiers.
- At each node, a binary classifier is constructed.
- The advantages of DDAG are fast computational time especially for large-scale problems and a risk bound (Platt et al., 2000).
- We propose to use the p-Center approach for constructing a binary classifier for each DDAG node.

DDAG

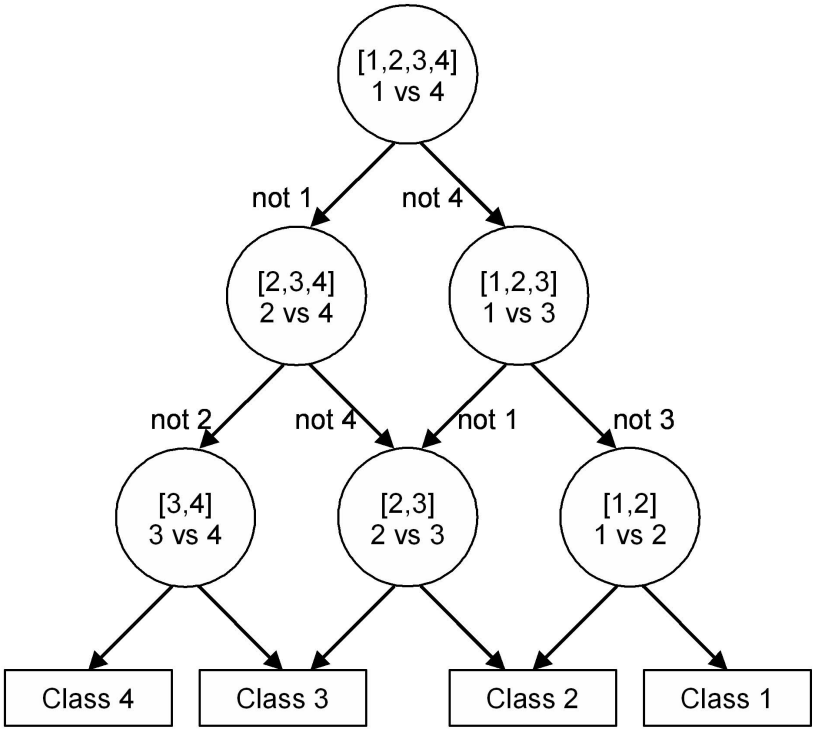

Figure: 15. The DDAG diagram for the 4-class classification problem.

Implementation

Table: 5. Data sets used in the experiments taken from the UC Irvine Machine Learning Repository (Asuncion & Newman, 2007).

Data Set	Total Instances	Number of Attributes	Number of Classes
Iris	150	4	3
Wine	178	13	3
Glass	214	13	6
Dermatology	358	34	6

Results

Table: 6. The accuracy with 95% confidence intervals and computation time results for OAA after conducting a 10-fold cross-validation (best results in bold).

Data Set	MPCM (in MATLAB®)		MSVM (C-based MOSEK solver)		MSVM (MATLAB®solver)	
	Accuracy (%)	Time (sec.)	Accuracy (%)	Time (sec.)	Accuracy (%)	Time (sec.)
Iris	**98.00 ± 2.24**	6.0	97.33 ± 2.58	**2.1**	93.33 ± 3.99	99.0
Wine	**98.92 ± 1.52**	8.5	**98.92 ± 1.52**	**2.7**	**98.92 ± 1.52**	93.1
Glass	**70.00 ± 6.13**	**8.6**	69.90 ± 6.13	8.7	68.53 ± 6.21	435.4
Dermatology	97.50 ± 1.62	103.3	**97.51 ± 1.61**	**31.9**	**97.51 ± 1.61**	1739.9

Results

Table: 7. The accuracy with 95% confidence intervals and computation time results for OAO after conducting a 10-fold cross-validation (best results in bold).

Data Set	MPCM (in MATLAB®)		MSVM (C-based MOSEK solver)		MSVM (MATLAB®solver)	
	Accuracy (%)	Time (sec.)	Accuracy (%)	Time (sec.)	Accuracy (%)	Time (sec.)
Iris	**98.00 ± 2.24**	4.0	**98.00 ± 2.24**	**1.1**	94.00 ± 3.80	29.5
Wine	**99.44 ± 1.09**	6.5	98.92 ± 1.52	**1.3**	98.92 ± 1.52	22.2
Glass	**71.84 ± 6.01**	8.2	71.73 ± 6.02	**3.9**	69.85 ± 6.13	78.3
Dermatology	**98.07 ± 1.43**	34.7	97.51 ± 1.61	**10.9**	97.51 ± 1.61	57.1

Results

Table: 8. The accuracy with 95% confidence intervals and computation time results for DDAG after conducting a 10-fold cross-validation (best results in bold).

Data Set	MPCM (in MATLAB®)		MSVM (C-based MOSEK solver)		MSVM (MATLAB®solver)	
	Accuracy (%)	Time (sec.)	Accuracy (%)	Time (sec.)	Accuracy (%)	Time (sec.)
Iris	**98.00 ± 2.24**	3.6	**98.00 ± 2.24**	**1.0**	94.00 ± 3.80	28.3
Wine	**99.44 ± 1.09**	5.9	98.92 ± 1.52	**1.3**	98.92 ± 1.52	22.2
Glass	**71.84 ± 6.01**	8.0	71.73 ± 6.02	**3.9**	69.85 ± 6.13	77.2
Dermatology	**98.07 ± 1.43**	33.1	97.51 ± 1.61	**9.9**	97.51 ± 1.61	55.62

Active Learning with Kernel Machines

- Using SVMs, the basic idea of the active learning algorithms of Campbell et al. (2000), Schohn and Cohn (2000), and Tong and Koller (2001) is to choose the unlabeled instance for the next query closest to the separating hyperplane in the feature space that is the instance with the smallest margin.
- Tong and Koller (2001) suggested selecting the next unlabeled instances that approximately bisect the version space.

Active Learning with Kernel Machines

- Using SVMs, the basic idea of the active learning algorithms of Campbell et al. (2000), Schohn and Cohn (2000), and Tong and Koller (2001) is to choose the unlabeled instance for the next query closest to the separating hyperplane in the feature space that is the instance with the smallest margin.
- Tong and Koller (2001) suggested selecting the next unlabeled instances that approximately bisect the version space.

Active Learning with Kernel Machines

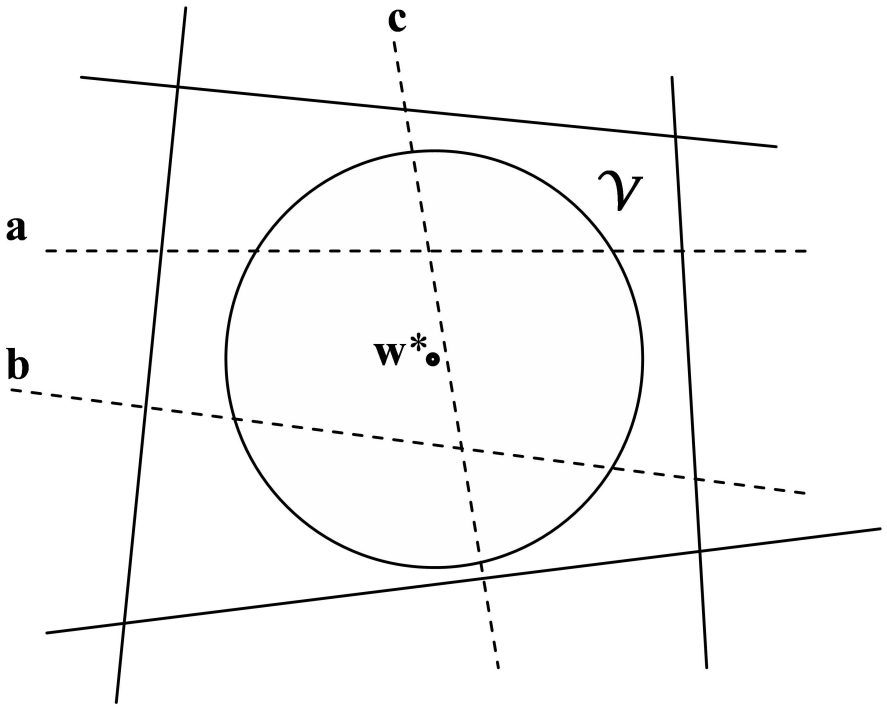

Figure: Illustration of active learning. The active learning algorithm will choose **c** since it is the closest to $\mathbf{w}^*$.

Active Learning with Kernel Machines

- Using the approach of Tong and Koller (2001), we develop an active learning that utilizes the p-Center of the version space.
- Hence, we select the unlabeled instances that are closed to the p-Center solution, $\mathbf{w}^*_{p-Center}$.

Active Learning with Kernel Machines

- Using the approach of Tong and Koller (2001), we develop an active learning that utilizes the p-Center of the version space.
- Hence, we select the unlabeled instances that are closed to the p-Center solution, $\mathbf{w}^*_{p-Center}$.

Active Learning for Tornado Prediction

- In tornado prediction, labeling data is considered costly and time consuming since we need to verify which storm-scale circulations produce tornadoes in the ground.
- The tornado events can be verified from facts in the ground including photographs, videos, damage surveys, and eyewitness reports.
- Based on tornado verification, we then determine and label which circulations produce tornadoes or not.

Active Learning for Tornado Prediction

- In tornado prediction, labeling data is considered costly and time consuming since we need to verify which storm-scale circulations produce tornadoes in the ground.
- The tornado events can be verified from facts in the ground including photographs, videos, damage surveys, and eyewitness reports.
- Based on tornado verification, we then determine and label which circulations produce tornadoes or not.

Active Learning for Tornado Prediction

- In tornado prediction, labeling data is considered costly and time consuming since we need to verify which storm-scale circulations produce tornadoes in the ground.
- The tornado events can be verified from facts in the ground including photographs, videos, damage surveys, and eyewitness reports.
- Based on tornado verification, we then determine and label which circulations produce tornadoes or not.

Measuring the Quality of the Forecasts for Tornado Prediction

Table: 9. Confusion matrix.

	"Yes" Observation	"No" Observation
"Yes" Forecast	a (hit)	b (false alarm)
"No" Forecast	c (miss)	d (correct null)

Measuring the Quality of the Forecasts for Tornado Prediction

- Those skill scores are defined as:

$$CSI = \frac{a}{a+b+c} \tag{26}$$

$$POD = \frac{a}{a+c} \tag{27}$$

$$FAR = \frac{b}{a+b} \tag{28}$$

$$Bias = \frac{a+b}{a+c} \tag{29}$$

$$HSS = \frac{2(ad-bc)}{(a+c)(c+d)+(a+b)(b+d)} \tag{30}$$

Experiments

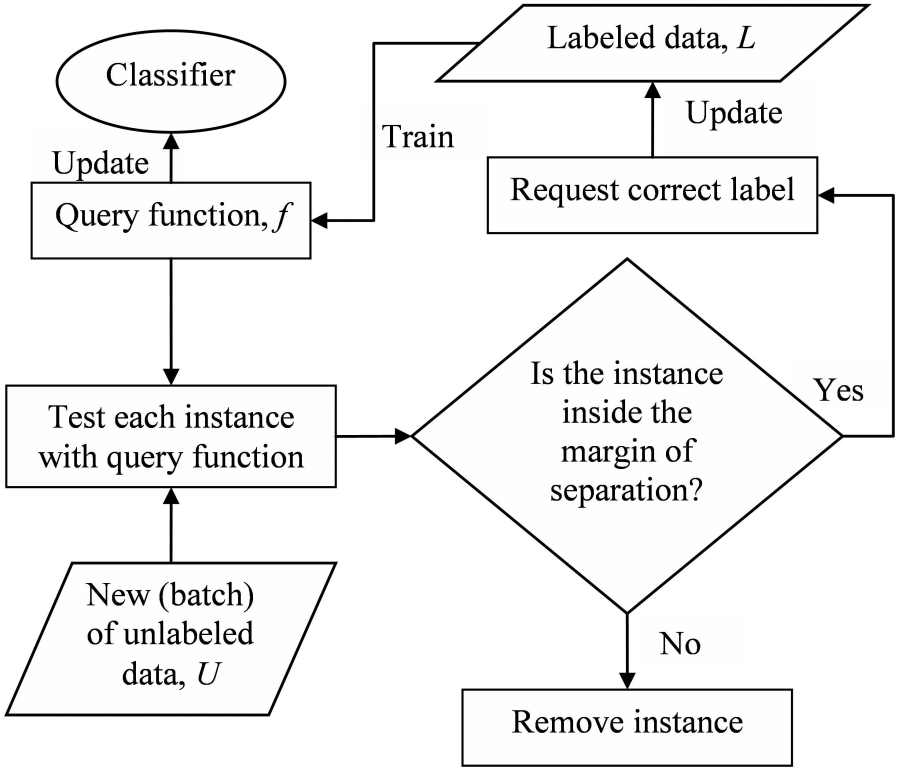

Figure: 17. The scheme of active learning.

Results

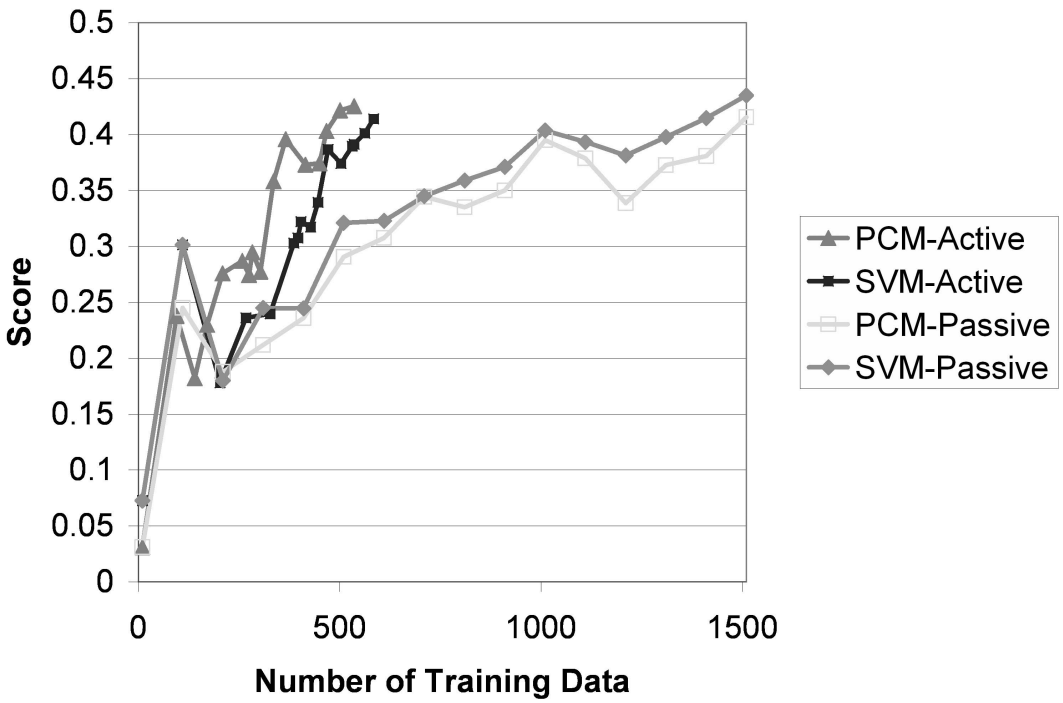

Figure: 18. The results of CSI on the testing set using active and passive learning at all iterations.

Results

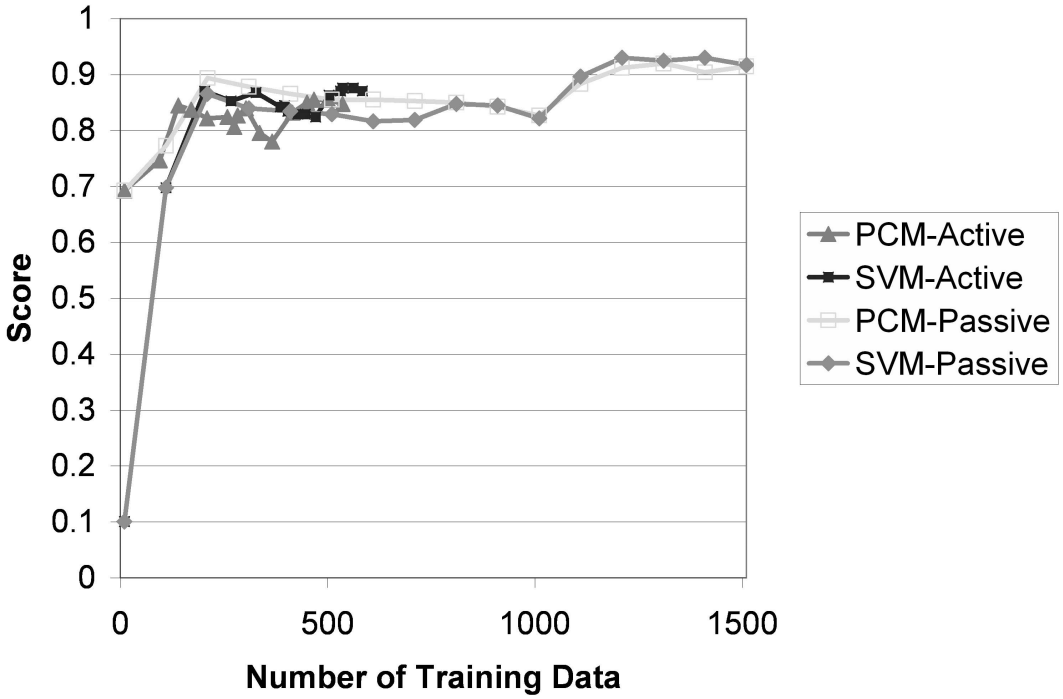

Figure: 19. The results of POD on the testing set using active and passive learning at all iterations.

Results

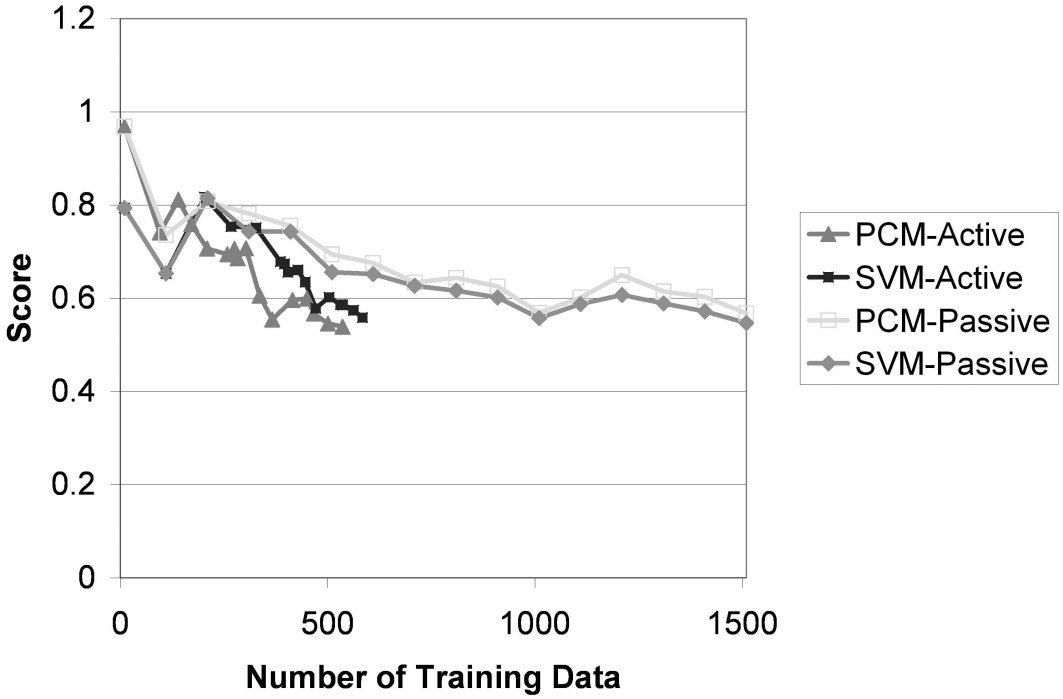

Figure: 20. The results of FAR on the testing set using active and passive learning at all iterations.

Results

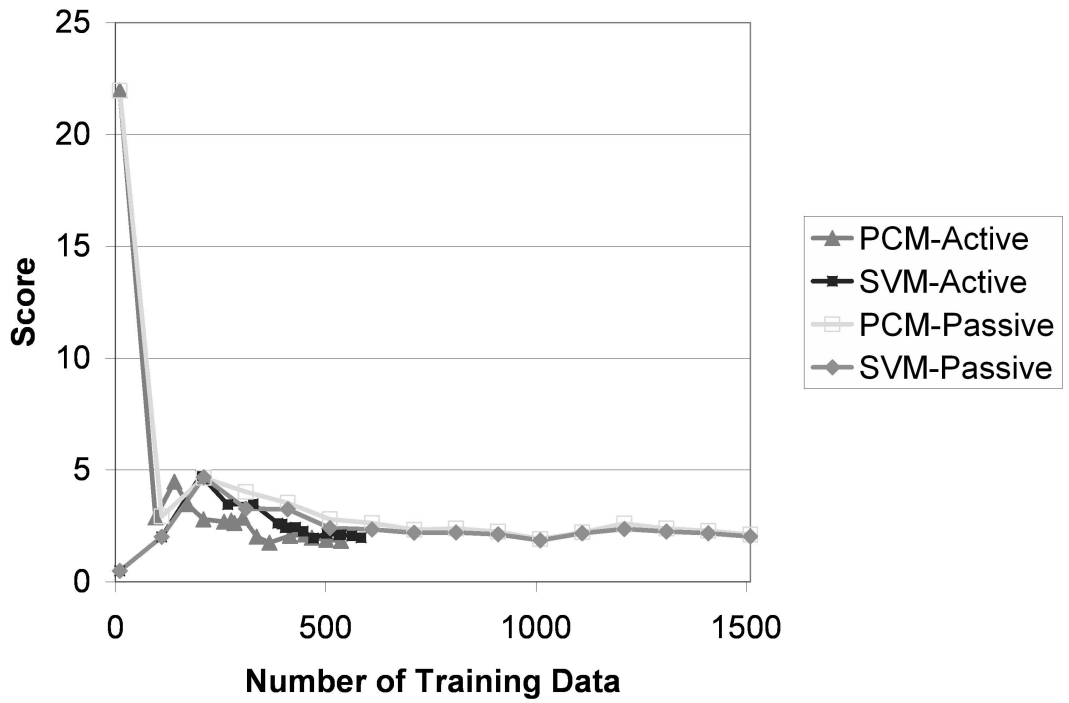

Figure: 21. The results of Bias on the testing set using active and passive learning at all iterations.

Results

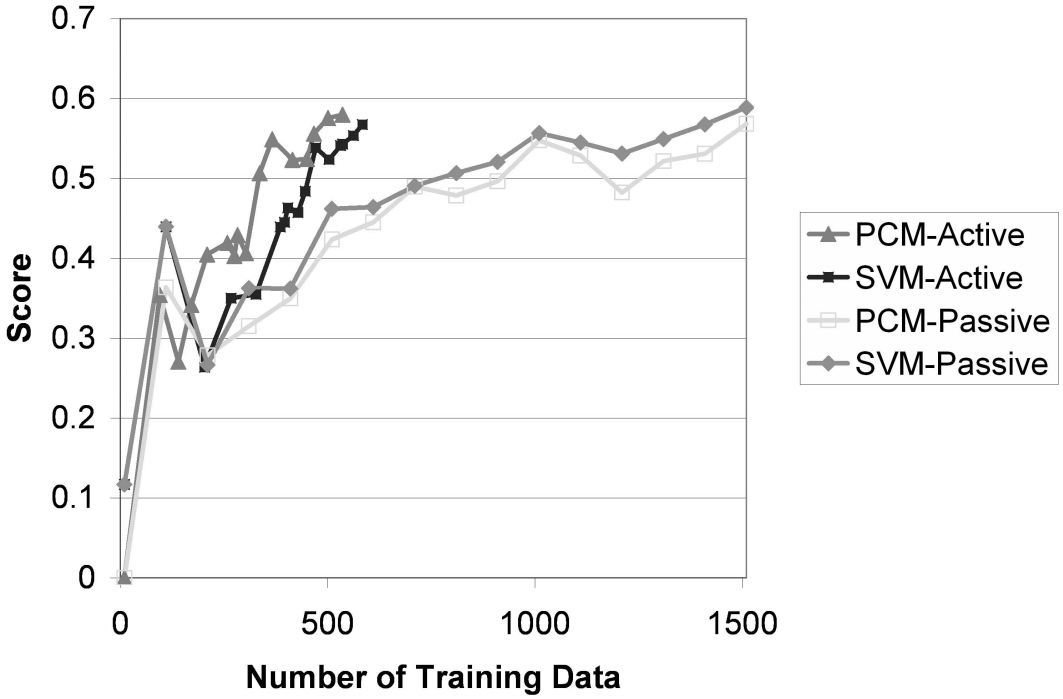

Figure: 22. The results of HSS on the testing set using active and passive learning at all iterations.

Results

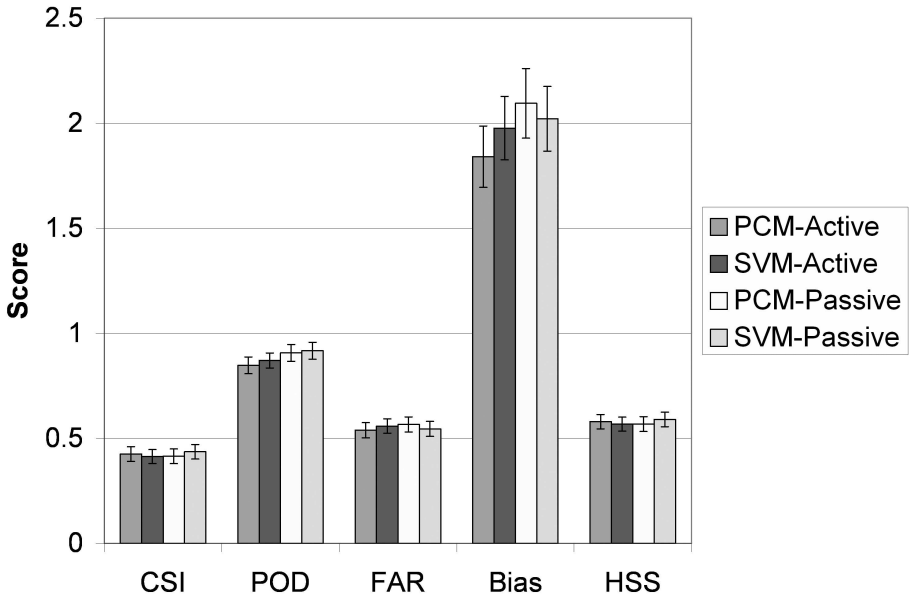

Figure: 23. The last iteration results with 95% confidence intervals on the testing set after conducting bootstrap resampling with 1000 replications.

Results

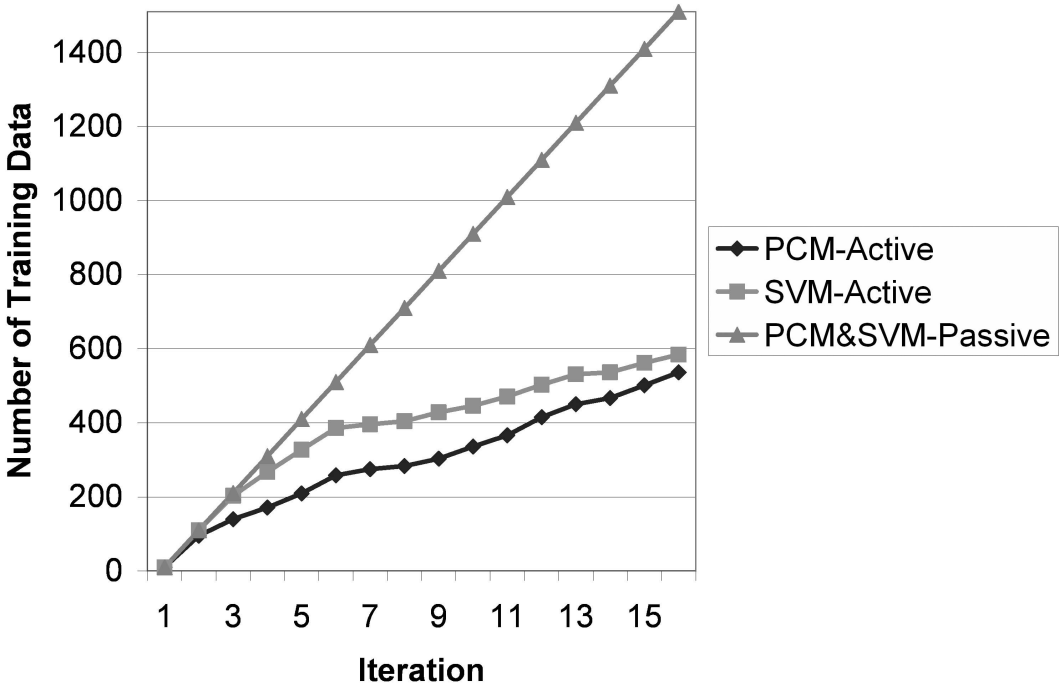

Figure: 24. The diagram of the number of training data vs. iteration.

Conclusions

- Motivated by finding an alternative approach to SVMs for solving problems with asymmetric or elongated version spaces, we develop new algorithms for regression analysis, multiclass classification, and active learning with the p-Center approach.
- The results of the proposed algorithms show promising performance compared to the SVMs.
- The proposed p-Center machine for regression analysis show competitive performance on solving nonlinear regression problems.

Conclusions

- Motivated by finding an alternative approach to SVMs for solving problems with asymmetric or elongated version spaces, we develop new algorithms for regression analysis, multiclass classification, and active learning with the p-Center approach.
- The results of the proposed algorithms show promising performance compared to the SVMs.
- The proposed p-Center machine for regression analysis show competitive performance on solving nonlinear regression problems.

Conclusions

- Motivated by finding an alternative approach to SVMs for solving problems with asymmetric or elongated version spaces, we develop new algorithms for regression analysis, multiclass classification, and active learning with the p-Center approach.
- The results of the proposed algorithms show promising performance compared to the SVMs.
- The proposed p-Center machine for regression analysis show competitive performance on solving nonlinear regression problems.

Conclusions

- For solving multiclass classification problems, our proposed algorithm also performs as good as the multiclass SVM.
- An algorithm for conducting active learning with the PCM is found to be effective on reducing the instances that need to be labeled and included in the training set.

Conclusions

- For solving multiclass classification problems, our proposed algorithm also performs as good as the multiclass SVM.
- An algorithm for conducting active learning with the PCM is found to be effective on reducing the instances that need to be labeled and included in the training set.

Recommendations

- The future development of this research might be concentrated on removing redundant or non-binding constraints for further improvement in performance and computational time.
- In our current approach, we use all of the constraints for computing the p-Center of the version space.
- The impact of uncertainty on the data to the performance of our proposed algorithms is also interesting to investigate.
- Another possible development is to investigate the implementation of our approach for very large scale problems and in real time applications.

Recommendations

- The future development of this research might be concentrated on removing redundant or non-binding constraints for further improvement in performance and computational time.
- In our current approach, we use all of the constraints for computing the p-Center of the version space.
- The impact of uncertainty on the data to the performance of our proposed algorithms is also interesting to investigate.
- Another possible development is to investigate the implementation of our approach for very large scale problems and in real time applications.

Recommendations

- The future development of this research might be concentrated on removing redundant or non-binding constraints for further improvement in performance and computational time.
- In our current approach, we use all of the constraints for computing the p-Center of the version space.
- The impact of uncertainty on the data to the performance of our proposed algorithms is also interesting to investigate.
- Another possible development is to investigate the implementation of our approach for very large scale problems and in real time applications.

Recommendations

- The future development of this research might be concentrated on removing redundant or non-binding constraints for further improvement in performance and computational time.
- In our current approach, we use all of the constraints for computing the p-Center of the version space.
- The impact of uncertainty on the data to the performance of our proposed algorithms is also interesting to investigate.
- Another possible development is to investigate the implementation of our approach for very large scale problems and in real time applications.

- Thank You