Module 6: Maximizing the Margin of a Linear Classifier (Support Vector Machines)

Machine Learning Course

Contents

1 Introduction to Support Vector Machines

1.1 From Perceptron to Maximum-Margin Classifiers

In the previous lecture, we explored the Perceptron algorithm, which finds a linear decision boundary that correctly classifies all training examples (assuming the data is linearly separable). However, the Perceptron algorithm has a key limitation: it can find any linear separator that works, but it doesn't necessarily find the "best" one.

This raises an important question: Among all possible linear separators, which one should we choose? Support Vector Machines (SVMs) provide a principled answer to this question by finding the linear separator that maximizes the margin between the two classes.

1.2 Motivation for Maximum-Margin Classification

Why should we care about maximizing the margin? There are several compelling reasons:

- **Generalization**: A classifier with a larger margin is likely to generalize better to unseen data
- **Robustness**: Maximum-margin classifiers are more robust to small perturbations in the data
- **Theoretical guarantees**: The margin is directly related to bounds on the generalization error
- **Uniqueness**: Unlike the Perceptron, which can find many different solutions, the maximum-margin classifier is unique (for linearly separable data)

2 Review of Linear Classification

2.1 The Perceptron Algorithm

Let's briefly review the Perceptron algorithm:

The Perceptron Algorithm

1. Initialize $\mathbf{w} = \mathbf{0}$ and $b = 0$
2. Keep cycling through the training data $(\mathbf{x}, y)$:
   (a) If $y(\mathbf{w} \cdot \mathbf{x} + b) \leq 0$ (i.e., point misclassified):
      i. $\mathbf{w} = \mathbf{w} + y\mathbf{x}$
      ii. $b = b + y$

2.2 Perceptron Convergence

The Perceptron algorithm has an important convergence guarantee:

Perceptron Convergence Theorem
If the training data is linearly separable, then:

- The Perceptron algorithm will find a linear classifier with zero training error
- It will converge within a finite number of steps

However, as illustrated in the figure below, there can be many possible linear separators for a given dataset, and the Perceptron might find any one of them depending on initialization and the order of processing the training examples.

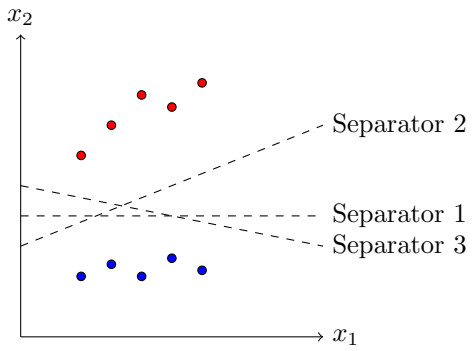

Figure 1: Multiple possible linear separators for a linearly separable dataset

3 The Maximum-Margin Approach

3.1 The Learning Problem

Let's formalize the learning problem for linear classification:

Given training data $(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(n)}, y^{(n)}) \in \mathbb{R}^d \times \{-1, +1\}$, find $\mathbf{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$ such that:

$$y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) > 0 \quad \text{for all } i = 1, 2, \ldots, n$$

This condition ensures that all training examples are correctly classified.

3.2 Scaling the Parameters

An important observation is that if $(\mathbf{w}, b)$ is a solution to the learning problem, then $(c\mathbf{w}, cb)$ is also a solution for any $c > 0$. This is because:

$$y^{(i)}(c\mathbf{w} \cdot \mathbf{x}^{(i)} + cb) = c \cdot y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) > 0$$

This scaling property allows us to reformulate the problem. We can equivalently ask for:

$$y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) \geq 1 \quad \text{for all } i = 1, 2, \ldots, n$$

This reformulation will be crucial for defining the margin.

3.3 Defining the Margin

The margin of a linear classifier is the distance from the decision boundary to the nearest training point. For a linear classifier defined by $\mathbf{w}$ and b, the decision boundary is the hyperplane:

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

The constraint $y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) \geq 1$ implies that:

- All positive points ($y^{(i)} = +1$) satisfy $\mathbf{w} \cdot \mathbf{x}^{(i)} + b \geq 1$
- All negative points ($y^{(i)} = -1$) satisfy $\mathbf{w} \cdot \mathbf{x}^{(i)} + b \leq -1$

This means that all positive points are on or above the hyperplane $\mathbf{w} \cdot \mathbf{x} + b = 1$, and all negative points are on or below the hyperplane $\mathbf{w} \cdot \mathbf{x} + b = -1$.

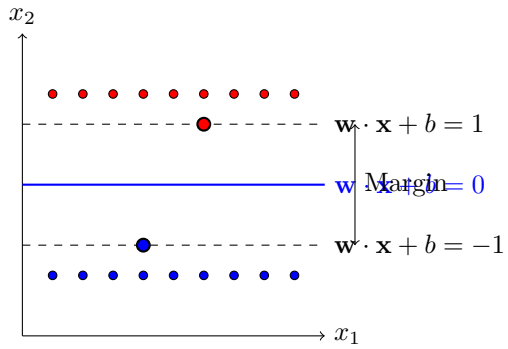

Figure 2: Illustration of the margin in a linearly separable dataset. The support vectors (highlighted) lie exactly on the margin boundaries.

3.4 Calculating the Margin

The distance from a point $\mathbf{x}$ to the hyperplane $\mathbf{w} \cdot \mathbf{x} + b = 0$ is given by:

$$\text{distance} = \frac{|\mathbf{w} \cdot \mathbf{x} + b|}{\|\mathbf{w}\|}$$

For points on the hyperplanes $\mathbf{w} \cdot \mathbf{x} + b = 1$ and $\mathbf{w} \cdot \mathbf{x} + b = -1$, this distance is:

$$\text{distance} = \frac{1}{\|\mathbf{w}\|}$$

Therefore, the margin γ is:

$$\gamma = \frac{1}{\|\mathbf{w}\|}$$

3.5 Maximizing the Margin

To maximize the margin $\gamma = \frac{1}{\|\mathbf{w}\|}$, we need to minimize $\|\mathbf{w}\|$. Since minimizing $\|\mathbf{w}\|$ is equivalent to minimizing $\|\mathbf{w}\|^2$ (and the latter is differentiable everywhere), we can formulate the maximum-margin classification problem as:

$$\min_{\mathbf{w}\in\mathbb{R}^d, b\in\mathbb{R}} \|\mathbf{w}\|^2$$

subject to:

$$y^{(i)}(\mathbf{w}\cdot\mathbf{x}^{(i)} + b) \geq 1 \quad \text{for all } i = 1, 2, \ldots, n$$

This is a convex quadratic optimization problem with linear constraints, which can be solved efficiently using standard optimization techniques.

4 Support Vector Machines

4.1 The Optimization Problem

The optimization problem for finding the maximum-margin linear classifier is:

Hard-Margin SVM Optimization Problem

$$\min_{\mathbf{w}\in\mathbb{R}^d, b\in\mathbb{R}} \|\mathbf{w}\|^2 \tag{1}$$

$$\text{subject to } y^{(i)}(\mathbf{w}\cdot\mathbf{x}^{(i)} + b) \geq 1 \quad \text{for all } i = 1, 2, \ldots, n \tag{2}$$

This formulation is known as the hard-margin Support Vector Machine (SVM).

4.2 Properties of the Solution

The solution to the SVM optimization problem has several important properties:

- It is a convex optimization problem, which means that it has a unique global minimum
- The solution depends only on a subset of the training points, called support vectors
- Support vectors are the points that lie exactly on the margin, i.e., $y^{(i)}(\mathbf{w}\cdot\mathbf{x}^{(i)} + b) = 1$

4.3 Support Vectors

The solution to the SVM optimization problem can be expressed as:

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y^{(i)} \mathbf{x}^{(i)}$$

where $\alpha_i \geq 0$ are Lagrange multipliers. Importantly, $\alpha_i > 0$ only for the support vectors (points that lie exactly on the margin). For all other points, $\alpha_i = 0$.

This means that the solution $\mathbf{w}$ is a linear combination of only the support vectors, and the decision boundary is determined entirely by these points.

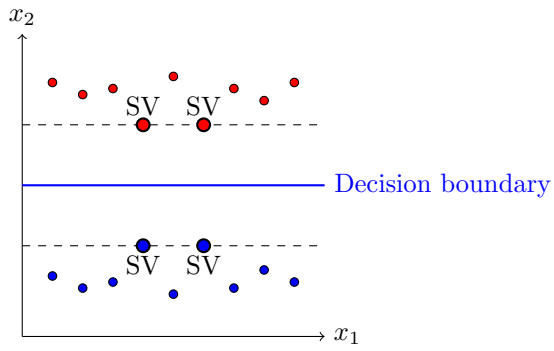

Figure 3: Support vectors (SV) are the points that lie exactly on the margin boundaries. The decision boundary is determined entirely by these points.

5 The Dual Formulation

5.1 Lagrangian Formulation

To solve the SVM optimization problem, we can use Lagrange multipliers. The Lagrangian is:

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i [y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) - 1]$$

where $\alpha_i \geq 0$ are the Lagrange multipliers.

5.2 Dual Problem

By taking derivatives of the Lagrangian with respect to $\mathbf{w}$ and b and setting them to zero, we get:

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y^{(i)} \mathbf{x}^{(i)}$$

$$\sum_{i=1}^{n} \alpha_i y^{(i)} = 0$$

Substituting these back into the Lagrangian, we get the dual problem:

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y^{(i)} y^{(j)} \mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)}$$

subject to:

$$\alpha_i \geq 0 \quad \text{for all } i = 1, 2, \ldots, n$$

$$\sum_{i=1}^{n} \alpha_i y^{(i)} = 0$$

This dual formulation is often easier to solve than the primal problem, especially when the number of features is large.

6 Example: Iris Dataset

6.1 Dataset Description

The Iris dataset is a classic dataset in machine learning, collected by the botanist Edgar Anderson and made famous by the statistician Ronald Fisher. It contains measurements of 150 iris flowers from three different species:

- Iris setosa
- Iris versicolor
- Iris virginica

For each flower, four measurements were taken:

- Sepal length
- Sepal width
- Petal length
- Petal width

6.2 Binary Classification Example

For simplicity, let's consider a binary classification problem using only two of the species (setosa and versicolor) and two of the features (sepal width and petal width).

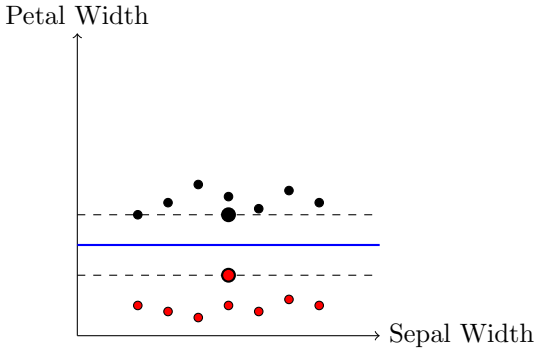

Figure 4: SVM classification of Iris setosa (red circles) and Iris versicolor (black triangles) using sepal width and petal width features.

In this example, the data is linearly separable, and the SVM finds the maximum-margin linear classifier. The support vectors are the points that lie exactly on the margin boundaries.

7 Soft-Margin SVM

7.1 Handling Non-linearly Separable Data

The hard-margin SVM assumes that the data is linearly separable. However, in real-world scenarios, data is often not perfectly separable due to noise or outliers. To handle such cases, we can use a soft-margin SVM, which allows for some misclassifications.

7.2 Slack Variables

We introduce slack variables $\xi_i \geq 0$ for each training point, which measure the degree of misclassification:

$$y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) \geq 1 - \xi_i \quad \text{for all } i = 1, 2, \ldots, n$$

If $\xi_i = 0$, the point is correctly classified and on or beyond the margin. If $0 < \xi_i \leq 1$, the point is correctly classified but within the margin. If $\xi_i > 1$, the point is misclassified.

7.3 Optimization Problem

The soft-margin SVM optimization problem is:

Soft-Margin SVM Optimization Problem

$$\min_{\mathbf{w}, b, \xi} \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i \tag{3}$$

$$\text{subject to } y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) \geq 1 - \xi_i \quad \text{for all } i = 1, 2, \ldots, n \tag{4}$$

$$\xi_i \geq 0 \quad \text{for all } i = 1, 2, \ldots, n \tag{5}$$

The parameter $C > 0$ controls the trade-off between maximizing the margin and minimizing the classification error. A larger C places more emphasis on correctly classifying all training points, while a smaller C places more emphasis on maximizing the margin.

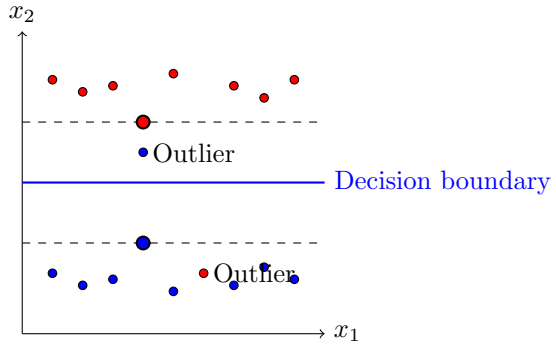

Figure 5: Soft-margin SVM allows for some misclassifications to handle outliers or non-linearly separable data.

8 Kernel Methods for Non-linear Classification

8.1 Limitations of Linear Classifiers

Linear classifiers, including hard-margin and soft-margin SVMs, can only learn linear decision boundaries. However, many real-world datasets require non-linear decision boundaries for effective classification.

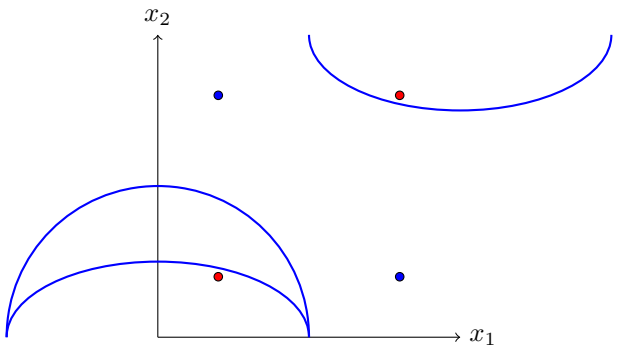

Figure 6: Example of data that requires a non-linear decision boundary (XOR pattern)

8.2 The Kernel Trick

The kernel trick is a clever technique that allows SVMs to learn non-linear decision boundaries without explicitly mapping the data to a higher-dimensional space. It relies on the observation that the dual formulation of the SVM optimization problem only depends on inner products between data points.

8.2.1 Feature Maps

Let $\phi : \mathbb{R}^d \to \mathbb{R}^D$ be a feature map that maps data from the original d-dimensional space to a higher-dimensional space $\mathbb{R}^D$. The idea is to find a linear boundary in this higher-dimensional space, which corresponds to a non-linear boundary in the original space.

8.2.2 Kernel Functions

A kernel function $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ computes the inner product in the higher-dimensional space without explicitly computing the mapping:

$$K(\mathbf{x}, \mathbf{z}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{z})$$

This is powerful because:

- We never need to explicitly compute $\phi(\mathbf{x})$, which could be very high-dimensional or even infinite-dimensional
- We only need to define the kernel function K, which computes the inner product directly

8.3 Common Kernel Functions

Several kernel functions are commonly used in practice:

- **Linear kernel**: $K(\mathbf{x}, \mathbf{z}) = \mathbf{x} \cdot \mathbf{z}$
- **Polynomial kernel**: $K(\mathbf{x}, \mathbf{z}) = (\mathbf{x} \cdot \mathbf{z} + c)^d$, where $c \geq 0$ and $d \in \mathbb{N}$
- **Radial Basis Function (RBF) kernel**: $K(\mathbf{x}, \mathbf{z}) = \exp(-\gamma \|\mathbf{x} - \mathbf{z}\|^2)$, where $\gamma > 0$
- **Sigmoid kernel**: $K(\mathbf{x}, \mathbf{z}) = \tanh(\alpha \mathbf{x} \cdot \mathbf{z} + c)$, where $\alpha > 0$ and $c \geq 0$

8.4 Kernel SVM Formulation

The dual formulation of the SVM optimization problem with kernels is:

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y^{(i)} y^{(j)} K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$$

subject to:

$$\alpha_i \geq 0 \quad \text{for all } i = 1, 2, \ldots, n$$

$$\sum_{i=1}^{n} \alpha_i y^{(i)} = 0$$

The decision function becomes:

$$f(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y^{(i)} K(\mathbf{x}^{(i)}, \mathbf{x}) + b \right)$$

8.5 Example: RBF Kernel

The RBF kernel is particularly popular because it can model complex non-linear decision boundaries. It effectively measures the similarity between two points based on their Euclidean distance:

$$K(\mathbf{x}, \mathbf{z}) = \exp(-\gamma \|\mathbf{x} - \mathbf{z}\|^2)$$

The parameter γ controls the "width" of the kernel:

- Small γ: Wide kernel, smooth decision boundary
- Large γ: Narrow kernel, more complex decision boundary

9 Multi-class SVMs

9.1 Extending SVMs to Multiple Classes

SVMs are inherently binary classifiers, but many real-world problems involve multiple classes. Several strategies exist for extending SVMs to multi-class classification:

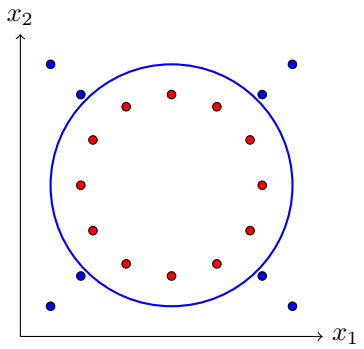

Figure 7: Example of a non-linear decision boundary learned using an RBF kernel

9.1.1 One-vs-Rest (OVR)

In the One-vs-Rest approach:

- Train k binary SVMs, where k is the number of classes
- Each SVM distinguishes one class from all other classes
- For a new point, evaluate all k SVMs and assign the class with the highest confidence score

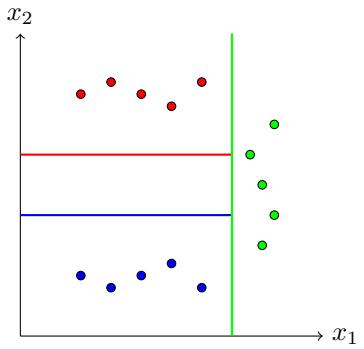

Figure 8: One-vs-Rest approach for multi-class SVM with three classes

9.1.2 One-vs-One (OVO)

In the One-vs-One approach:

- Train $\binom{k}{2} = \frac{k(k-1)}{2}$ binary SVMs, one for each pair of classes
- For a new point, each binary SVM votes for one class
- Assign the class with the most votes

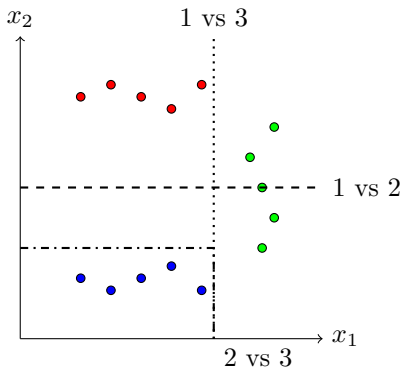

Figure 9: One-vs-One approach for multi-class SVM with three classes

9.1.3 Direct Multi-class Formulation

There are also direct formulations of multi-class SVMs that solve a single optimization problem:

- Crammer and Singer's multi-class SVM
- Weston and Watkins' multi-class SVM

These formulations are more complex but can sometimes yield better results than the OVR or OVO approaches.

9.2 Comparison of Multi-class Strategies

- **One-vs-Rest**:
  - Pros: Simple, only requires k classifiers
  - Cons: Class imbalance, ambiguous regions
- **One-vs-One**:
  - Pros: Better handling of class imbalance, smaller training sets for each classifier
  - Cons: Requires $\binom{k}{2}$ classifiers, which can be large for many classes
- **Direct Multi-class**:
  - Pros: Theoretically more principled, considers all classes simultaneously
  - Cons: More complex optimization problem, computationally expensive

10 Practical Considerations

10.1 Hyperparameter Selection

SVMs have several hyperparameters that need to be tuned:

- C: The regularization parameter in soft-margin SVMs
- Kernel parameters (e.g., γ in RBF kernel, d in polynomial kernel)

These hyperparameters are typically selected using cross-validation:

- Split the data into training and validation sets
- Train SVMs with different hyperparameter values on the training set
- Evaluate performance on the validation set
- Select the hyperparameters that yield the best validation performance

10.2 Feature Scaling

SVMs are sensitive to the scale of the features. It's generally recommended to scale the features before training an SVM:

- Standardization: $x' = \frac{x-\mu}{\sigma}$
- Min-max scaling: $x' = \frac{x-\min(x)}{\max(x)-\min(x)}$

10.3 Handling Large Datasets

Standard SVM implementations have time complexity $O(n^2)$ to $O(n^3)$, where n is the number of training examples. This can be prohibitive for large datasets. Several approaches exist for scaling SVMs to large datasets:

- Chunking: Solve the optimization problem in smaller chunks
- Sequential Minimal Optimization (SMO): Optimize two Lagrange multipliers at a time
- Stochastic gradient descent: Approximate the SVM solution using SGD
- Linear SVMs: For linear kernels, specialized algorithms like LIBLINEAR can scale to millions of examples

10.4 Probabilistic Outputs

Standard SVMs output only the class label, not a probability. However, there are methods to convert SVM outputs to probabilities:

- Platt scaling: Fit a logistic regression model to the SVM scores
- Isotonic regression: A more flexible non-parametric approach

11 Worked Examples

11.1 Example 1: Linear SVM on a 2D Dataset

Consider a simple 2D dataset with two classes:

- Positive examples: $(1, 1)$, $(2, 3)$, $(3, 2)$
- Negative examples: $(1, 3)$, $(2, 1)$, $(3, 3)$

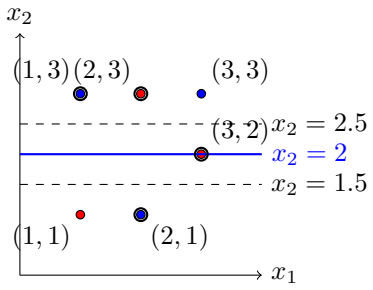

Figure 10: Linear SVM on a 2D dataset. The decision boundary is $x_2 = 2$, and the support vectors are circled.

The SVM finds the decision boundary $x_2 = 2$, which corresponds to $w = (0, 1)$ and $b = -2$. The margin is $\gamma = 1/\|w\| = 1$, and the support vectors are the points closest to the decision boundary: $(2, 1)$, $(3, 2)$, $(1, 3)$, and $(2, 3)$.

11.2 Example 2: Soft-Margin SVM with Outliers

Now, let's add an outlier to the dataset:

- Positive examples: $(1, 1)$, $(2, 3)$, $(3, 2)$, $(2, 0.5)$ (outlier)
- Negative examples: $(1, 3)$, $(2, 1)$, $(3, 3)$

With a hard-margin SVM, this dataset would not be linearly separable due to the outlier. However, a soft-margin SVM can handle the outlier by allowing it to be misclassified (with a penalty determined by the parameter C).

11.3 Example 3: Non-linear SVM with RBF Kernel

Consider a dataset with a circular pattern:

- Positive examples: Points inside a circle
- Negative examples: Points outside the circle

This dataset is not linearly separable, but an SVM with an RBF kernel can learn the circular decision boundary. The support vectors are the points closest to the decision boundary.

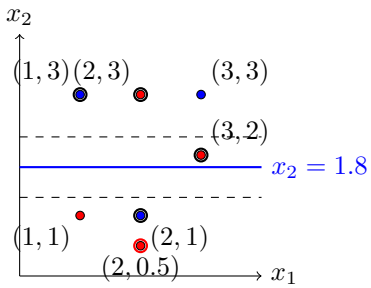

Figure 11: Soft-margin SVM with an outlier. The decision boundary shifts slightly to accommodate the outlier.

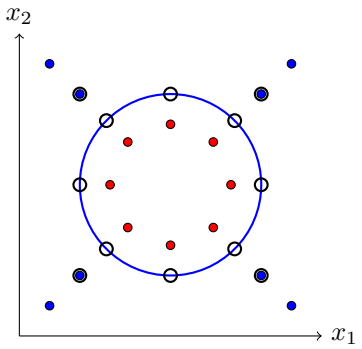

Figure 12: Non-linear SVM with RBF kernel. The decision boundary is a circle, and the support vectors are circled.

12 Summary and Key Takeaways

12.1 Core Concepts

- Support Vector Machines (SVMs) find the linear classifier with the maximum margin between classes
- The margin is the distance from the decision boundary to the nearest training points
- Maximizing the margin leads to better generalization to unseen data
- The optimization problem for hard-margin SVMs is to minimize $\|w\|^2$ subject to $y^{(i)}(w \cdot x^{(i)} + b) \geq 1$ for all training points

12.2 Extensions

- Soft-margin SVMs allow for some misclassifications to handle non-linearly separable data
- Kernel SVMs use the kernel trick to learn non-linear decision boundaries
- Multi-class SVMs extend binary SVMs to handle multiple classes

12.3 Strengths of SVMs

- Effective in high-dimensional spaces
- Memory efficient (only support vectors are needed for prediction)
- Versatile (different kernel functions for different types of data)
- Robust to overfitting, especially in high-dimensional spaces

12.4 Limitations of SVMs

- Computationally intensive for large datasets
- Sensitive to the choice of kernel and hyperparameters
- No direct probability estimates (though methods exist to convert scores to probabilities)
- Can be challenging to interpret, especially with non-linear kernels

12.5 Practical Tips

- Scale features before training SVMs
- Use cross-validation to select hyperparameters
- Start with a linear kernel and move to more complex kernels if needed
- For large datasets, consider specialized implementations or approximations

12.6 Connections to Other Methods

- SVMs are related to logistic regression, but focus on the margin rather than probabilistic interpretation
- The kernel trick used in SVMs is also applicable to other methods (e.g., kernel PCA, kernel k-means)
- SVMs can be viewed as a special case of regularized empirical risk minimization
- The concept of maximum margin has influenced other methods, such as boosting algorithms