Module 6: Linear Boundaries for Binary Classification

Machine Learning Course

Contents

1 Introduction to Linear Classification

1.1 From Regression to Classification

In previous modules, we explored regression and conditional probability estimation, approaching each as an optimization task. We now extend this optimization mindset to classification problems, specifically focusing on linear boundaries for binary classification.

1.2 The Classification Framework

In binary classification:

- Input: Feature vectors $x \in \mathbb{R}^d$
- Output: Binary labels $y \in \{-1, +1\}$
- Goal: Learn a decision boundary that separates positive from negative examples

1.3 Linear vs. Non-linear Classification

Linear classification uses a hyperplane to separate the feature space:

- Simple, interpretable models
- Efficient to train and evaluate
- Limited expressivity (can only represent linearly separable patterns)
- Foundation for more complex models (e.g., neural networks)

2 Geometry of Linear Classification

2.1 Linear Decision Boundaries in Two Dimensions

A linear decision boundary in 2D is a line that separates the feature space into two regions.

Example: Line with y-intercept = 4, slope = -4/3

$$x_2 = -\frac{4}{3}x_1 + 4 \quad \text{(slope-intercept form)} \tag{1}$$

$$\Rightarrow 4x_1 + 3x_2 - 12 = 0 \quad \text{(standard form)} \tag{2}$$

2.2 Making Predictions with Linear Boundaries

To classify a point, we evaluate the linear function at that point:

- If the result is positive, predict $+1$
- If the result is negative, predict -1

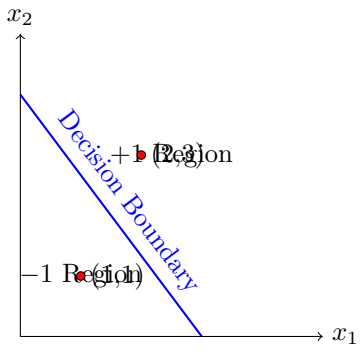

Figure 1: A linear decision boundary in 2D space with positive and negative regions

Example 1: Classifying the point (2,3)

$$\begin{aligned} 4x_1 + 3x_2 - 12 &= 4(2) + 3(3) - 12 && (3)\\ &= 8 + 9 - 12 && (4)\\ &= 5 > 0 && (5) \end{aligned}$$

Since the result is positive, we predict $+1$.

Example 2: Classifying the point (1,1)

$$\begin{aligned} 4x_1 + 3x_2 - 12 &= 4(1) + 3(1) - 12 && (6)\\ &= 4 + 3 - 12 && (7)\\ &= -5 < 0 && (8) \end{aligned}$$

Since the result is negative, we predict -1.

3 Linear Classification in D-Dimensional Space

3.1 Mathematical Formulation

In the general case with d-dimensional data:

- Data points: $\mathbf{x} \in \mathbb{R}^d$
- Labels: $y \in \{-1, +1\}$
- Weight vector: $\mathbf{w} \in \mathbb{R}^d$
- Bias term: $b \in \mathbb{R}$

The decision boundary is defined by:

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

3.2 Prediction Rule

For a new point $\mathbf{x}$, the prediction is:

$$\hat{y} = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b) = \begin{cases} +1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b > 0 \\ -1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b < 0 \end{cases}$$

3.3 Correctness Condition

A prediction is correct if and only if:

- True label $y = +1$ and $\mathbf{w} \cdot \mathbf{x} + b > 0$, OR
- True label $y = -1$ and $\mathbf{w} \cdot \mathbf{x} + b < 0$

This can be expressed compactly as:

$$y(\mathbf{w} \cdot \mathbf{x} + b) > 0$$

4 Geometric Interpretation

4.1 Properties of the Weight Vector

The weight vector $\mathbf{w}$ has several important geometric properties:

- $\mathbf{w}$ is perpendicular (normal) to the decision boundary
- The magnitude of $\mathbf{w}$ affects the "steepness" of the classifier but not the boundary itself
- The direction of $\mathbf{w}$ points toward the positive region

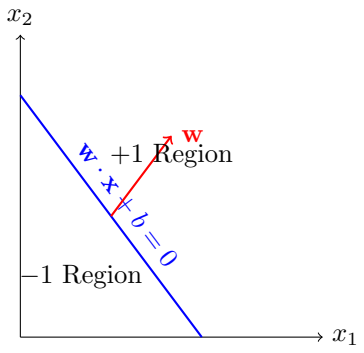

Figure 2: The weight vector $\mathbf{w}$ is perpendicular to the decision boundary

4.2 Role of the Bias Term

The bias term b determines the offset of the decision boundary from the origin:

- If $b = 0$, the boundary passes through the origin
- If $b > 0$, the boundary is shifted in the direction opposite to $\mathbf{w}$
- If $b < 0$, the boundary is shifted in the direction of $\mathbf{w}$

4.3 Distance to the Decision Boundary

The distance from a point $\mathbf{x}$ to the decision boundary is:

$$d(\mathbf{x}) = \frac{|\mathbf{w} \cdot \mathbf{x} + b|}{\|\mathbf{w}\|}$$

This has an important interpretation: points further from the boundary are classified with higher confidence.

5 Learning Linear Classifiers

5.1 Optimization Perspective

Learning a linear classifier involves finding the parameters $\mathbf{w}$ and b that minimize a loss function. Different algorithms use different loss functions:

- Perceptron: Uses a direct misclassification measure
- Logistic Regression: Uses log loss
- Support Vector Machines: Uses hinge loss

5.2 The Perceptron Algorithm

The perceptron is one of the earliest and most influential algorithms for linear classification.

Algorithm:

1. Initialize $\mathbf{w} = \mathbf{0}$, $b = 0$
2. Repeat until convergence:
   (a) For each training example $(\mathbf{x}^{(i)}, y^{(i)})$:
      i. If $y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) \leq 0$ (misclassified):
         A. $\mathbf{w} \leftarrow \mathbf{w} + y^{(i)}\mathbf{x}^{(i)}$
         B. $b \leftarrow b + y^{(i)}$

Convergence Property: If the data is linearly separable, the perceptron algorithm is guaranteed to converge to a solution in a finite number of steps.

5.3 Logistic Regression for Linear Classification

Logistic regression fits a linear boundary by modeling the probability of class membership:

$$P(Y = +1|\mathbf{x}, \mathbf{w}, b) = \frac{1}{1 + e^{-(\mathbf{w}\cdot\mathbf{x}+b)}}$$

The decision boundary is still $\mathbf{w} \cdot \mathbf{x} + b = 0$, but the model provides probabilities rather than just binary predictions.

5.4 Support Vector Machines (SVMs)

SVMs find the linear boundary that maximizes the margin (distance to the closest points from each class):

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \quad \text{subject to} \quad y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) \geq 1 \quad \forall i \tag{9}$$

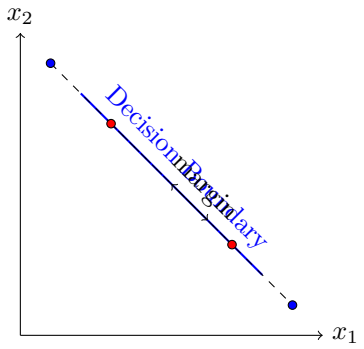

Figure 3: SVM finds the decision boundary that maximizes the margin

6 Limitations and Extensions

6.1 Linearly Non-separable Data

When data is not linearly separable, several approaches can be used:

- Soft-margin SVM: Allow some misclassifications with penalties
- Kernel methods: Implicitly map data to a higher-dimensional space where it becomes linearly separable
- Feature engineering: Create new features that make the data linearly separable

6.2 The Kernel Trick

The kernel trick allows us to implicitly work in a higher-dimensional space without explicitly computing the mapping:

- Linear kernel: $K(\mathbf{x}, \mathbf{z}) = \mathbf{x} \cdot \mathbf{z}$
- Polynomial kernel: $K(\mathbf{x}, \mathbf{z}) = (\mathbf{x} \cdot \mathbf{z} + c)^d$
- RBF kernel: $K(\mathbf{x}, \mathbf{z}) = \exp(-\gamma \|\mathbf{x} - \mathbf{z}\|^2)$

6.3 Multi-class Classification

Binary classification can be extended to multi-class problems using:

- One-vs-Rest: Train k binary classifiers, each separating one class from the rest
- One-vs-One: Train $\binom{k}{2}$ binary classifiers, one for each pair of classes
- Direct multi-class formulations (e.g., multinomial logistic regression)

7 Worked Examples

7.1 Example 1: Finding the Decision Boundary

Given points $(1, 2)$ labeled $+1$ and $(2, 1)$ labeled -1, find a linear decision boundary.

Solution: We need to find w_1, w_2, and b such that:

$$w_1 \cdot 1 + w_2 \cdot 2 + b > 0 \quad \text{(for the } +1 \text{ point)} \tag{10}$$
$$w_1 \cdot 2 + w_2 \cdot 1 + b < 0 \quad \text{(for the } -1 \text{ point)} \tag{11}$$

One possible solution is $w_1 = -1$, $w_2 = 1$, $b = 0$:

$$-1 \cdot 1 + 1 \cdot 2 + 0 = 1 > 0 \quad \checkmark \tag{12}$$
$$-1 \cdot 2 + 1 \cdot 1 + 0 = -1 < 0 \quad \checkmark \tag{13}$$

The decision boundary is $-x_1 + x_2 = 0$ or $x_2 = x_1$.

7.2 Example 2: Perceptron Algorithm Trace

Consider the dataset:

- $(1, 1)$ with label $+1$
- $(0, 1)$ with label $+1$
- $(0, 0)$ with label -1
- $(1, 0)$ with label -1

Perceptron Trace:

1. Initialize $\mathbf{w} = (0, 0)$, $b = 0$
2. First example $(1, 1)$, $y = +1$:

$$y(\mathbf{w} \cdot \mathbf{x} + b) = 1 \cdot ((0, 0) \cdot (1, 1) + 0) = 0 \leq 0 \tag{14}$$

   Misclassified, so update:

$$\mathbf{w} \leftarrow (0, 0) + 1 \cdot (1, 1) = (1, 1) \tag{15}$$
$$b \leftarrow 0 + 1 = 1 \tag{16}$$

3. Second example $(0, 1)$, $y = +1$:

$$y(\mathbf{w} \cdot \mathbf{x} + b) = 1 \cdot ((1, 1) \cdot (0, 1) + 1) = 2 > 0 \tag{17}$$

   Correctly classified, no update.

4. Third example $(0, 0)$, $y = -1$:

$$y(\mathbf{w} \cdot \mathbf{x} + b) = -1 \cdot ((1, 1) \cdot (0, 0) + 1) = -1 < 0 \tag{18}$$

   Correctly classified, no update.

5. Fourth example $(1, 0)$, $y = -1$:

$$y(\mathbf{w} \cdot \mathbf{x} + b) = -1 \cdot ((1, 1) \cdot (1, 0) + 1) = -2 < 0 \tag{19}$$

   Correctly classified, no update.

After one pass through the data, all examples are correctly classified. The final decision boundary is $x_1 + x_2 + 1 = 0$ or $x_2 = -x_1 - 1$.

8 Connection to Neural Networks

8.1 The Perceptron as a Building Block

The perceptron can be viewed as a single neuron in a neural network:

- Inputs: $\mathbf{x} = (x_1, x_2, \ldots, x_d)$
- Weights: $\mathbf{w} = (w_1, w_2, \ldots, w_d)$
- Bias: b
- Activation function: $\text{sign}(z)$
- Output: $\hat{y} = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)$

8.2 From Linear to Non-linear Classification

Neural networks extend the perceptron by:

- Using multiple layers of neurons
- Employing non-linear activation functions (e.g., sigmoid, ReLU)
- Learning hierarchical representations of the data

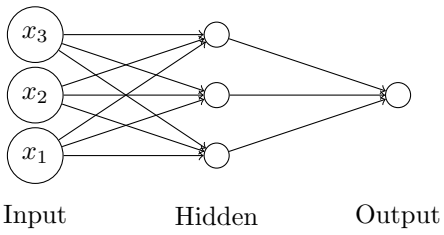

Figure 4: A simple neural network with one hidden layer

9 Practical Considerations

9.1 Feature Scaling

Linear classifiers are sensitive to the scale of features. Common scaling techniques include:

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

9.2 Regularization

To prevent overfitting, regularization terms can be added to the optimization objective:

- L1 regularization: $\lambda \|\mathbf{w}\|_1$ (promotes sparsity)
- L2 regularization: $\lambda \|\mathbf{w}\|_2^2$ (prevents large weights)

9.3 Evaluation Metrics

Common metrics for binary classification include:

- Accuracy: $\frac{\text{Correct predictions}}{\text{Total predictions}}$
- Precision: $\frac{\text{True positives}}{\text{True positives + False positives}}$
- Recall: $\frac{\text{True positives}}{\text{True positives + False negatives}}$
- F1 Score: $2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
- Area Under ROC Curve (AUC)

10 Summary and Key Takeaways

10.1 Core Concepts

- Linear classifiers separate the feature space using a hyperplane
- The decision boundary is defined by $\mathbf{w} \cdot \mathbf{x} + b = 0$
- Predictions are made based on the sign of $\mathbf{w} \cdot \mathbf{x} + b$
- A prediction is correct when $y(\mathbf{w} \cdot \mathbf{x} + b) > 0$

10.2 Algorithms

- Perceptron: Simple, iterative algorithm for linearly separable data
- Logistic Regression: Probabilistic approach to linear classification
- Support Vector Machines: Maximum-margin linear classifiers

10.3 Extensions

- Kernel methods for non-linear classification
- Multi-class classification techniques
- Neural networks as generalizations of perceptrons

10.4 Practical Tips

- Scale features appropriately
- Use regularization to prevent overfitting
- Choose evaluation metrics based on the problem context
- Consider non-linear methods if data is not linearly separable