Lecture 3: Linear Methods for Classification

Readings: ESL (Ch. 4), ISL (Ch. 4); code

Soon Hoe Lim

September 2, 2025

Outline

1. Introduction
2. Generative Approach: Discriminant Analysis
3. Discriminative Approach: Logistic Regression
4. Direct Approach: The Perceptron Algorithm
5. Exercises

The Classification Problem

Given a feature vector X, we want to predict a qualitative response G that takes values in a discrete set $\mathcal{G} = \{1, 2, \ldots, K\}$. We seek a classification rule (or classifier) $G(X)$ that assigns a class label to any input X.

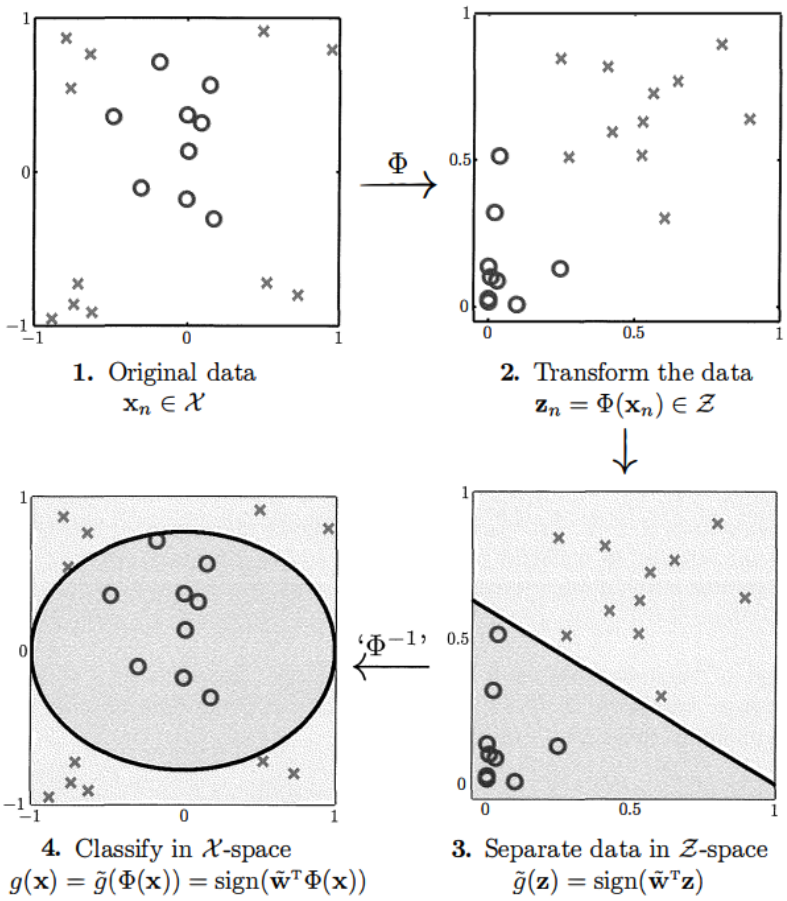
This lecture focuses on **linear methods for classification**, which produce **linear decision boundaries** in the input/feature space.

- A decision boundary is the surface in the input space $\mathcal{X}$ that separates points assigned to different classes.
- For a linear classifier, these boundaries are affine sets/hyperplanes.
- Despite its simplicity, this is a powerful and widely used idea:

  Nonlinear boundaries can be achieved by augmenting the feature space with transformations (e.g., squares, cross-products) and then applying linear methods in the new enlarged space.

Example: Linearly Separable vs. Nonlinearly Separable

The data set is not linearly separable, but separable by a circle on the original input space. By mapping them to a suitable feature space, the transformed samples are linearly separable in the feature space.



Linear Classifiers

- For classification, the Bayes optimal classifier assigns an observation x to the class with the largest posterior probability $P(G = k|X = x)$.
- The decision boundary between two classes k and l is the set of points in the input space where the posterior probabilities are equal:

$$\{x \in \mathcal{X} : P(G = k|X = x) = P(G = l|X = x)\}.$$

- This lecture focuses on classifiers that produce **linear decision boundaries**: divide the input space into regions labeled by class, and separate these regions by affine sets/hyperplanes.

⚠ Unlike the OLS and RLS regression, the ERM for classification rarely admits explicit solutions.

A Naive Method

Adapt linear regression for a classification problem.

Regression for Classification

1: If G has K classes, create an **indicator response matrix** $\mathbf{Y}$ of size $N \times K$, where $Y_{ik} = 1$ if observation i is in class k, and 0 otherwise.
2: Fit a multivariate linear regression model, regressing $\mathbf{Y}$ on $\mathbf{X}$. This gives the fit $\hat{\mathbf{Y}} = \mathbf{X}\hat{\mathbf{B}}$, where $\hat{\mathbf{B}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$ is of size $(p+1) \times K$.
3: For a new input x, compute the vector of fitted values $\hat{f}(x)^T = (1, x^T)\hat{\mathbf{B}}$.
4: The classification rule is to choose the class with the largest fitted value:

$$\hat{G}(x) = \arg\max_{k \in \mathcal{G}} \hat{f}_k(x).$$

We can then view the regression as an estimate of conditional expectation: for the random variable Y_k, $\mathbb{E}[Y_k | X = x] = P(G = k | X = x)$.

Why This is (Usually) a Bad Idea

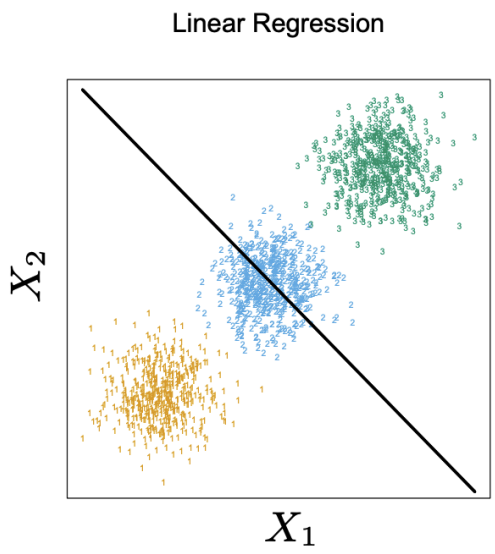
This approach seems simple, but it suffers from severe drawbacks.

1. Non-Probabilistic Predictions

- The fitted values $\hat{f}_k(x)$ are not probabilities.
- They are not constrained to be between 0 and 1.
- This makes interpretation difficult and can lead to unstable results.

2. The Masking Problem

- The rigid nature of the regression fit can lead to some classes being completely "masked" for $K \geq 3$ (natural for large K and small p).
- A decision region for a class might not exist, even if it is clearly present in the training data. This is a critical failure.



The Goal: The Bayes Optimal Classifier

The best possible classifier is the one that assigns an observation to the most probable class. This is the theoretical target we aim for.

Definition 1: The Bayes Optimal Classifier

Given the true class-conditional densities $f_k(x) = P(X = x|G = k)$ and priors $\pi_k = P(G = k)$, the class posterior probability is:

$$P(G = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{l=1}^{K} \pi_l f_l(x)}.$$

The **Bayes classifier** classifies to the class with the highest posterior probability: $\hat{G}^*(x) = \arg\max_k P(G = k|X = x)$.

Since the true densities $f_k(x)$ and priors π_k are unknown, the Bayes optimal classifier cannot be implemented in practice. Instead, we use our training data to model or estimate them. **Different assumptions about the form of $f_k(x)$ lead to different classification methods.**

Three Approaches to Linear Classification

We will explore principled strategies for finding linear decision boundaries.

1. **Generative Approaches**
   - Model the class-conditional densities $P(X|G = k)$. Use Bayes' rule to compute the posteriors $P(G = k|X)$.
   - **Discriminant Analysis:** Assume each density is a multivariate Gaussian.
2. **Discriminative Approaches**
   - No assumption on $P(X|G)$, directly models the posterior probabilities $P(G = k|X)$.
   - **Logistic Regression:** Model the posterior probabilities directly as a logistic function.
3. **Direct Approaches** (don't model probabilities at all)
   - Construct optimization algorithms to find separating hyperplanes.
   - **Perceptron**: Rosenblatt's model and algorithm that finds a separating hyperplane in the data (if one exists).
   - **Support Vector Classifier (SVC):** Vapnik's method to find an optimal hyperplane that directly separates the data with a maximal margin.

Generative Approach: Linear Discriminant Analysis (LDA)

LDA is a generative approach that results from applying the Bayes' rule under specific assumptions about the form of the $f_k(x)$.

Definition 2: LDA Assumptions

1. The class-conditional probability density for each class, $f_k(x)$, is a **multivariate Gaussian**: $X|G = k \sim \mathcal{N}(\mu_k, \Sigma_k)$. The density is given by:

$$f_k(x) = \frac{1}{(2\pi)^{p/2}|\Sigma_k|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1}(x-\mu_k)}.$$

2. All classes share a **common covariance matrix**: $\Sigma_k = \Sigma \quad \forall k \in \mathcal{G}$.

Proposition 1: Decision Boundaries under LDA

Under the LDA assumptions, all the decision boundaries produced are linear.

Proof.

See blackboard. □

Deriving the LDA Decision Rule

- We classify to the class that maximizes the log-posterior, $\log P(G = k|X = x)$. Since the denominator in Bayes' rule is the same for all classes, this is equivalent to maximizing:

$$\delta_k(x) = \log(\pi_k f_k(x)).$$

- Substituting the Gaussian density with a common covariance matrix Σ:

$$\delta_k(x) = \log \pi_k - \frac{1}{2}\log((2\pi)^p|\Sigma|) - \frac{1}{2}(x - \mu_k)^T\Sigma^{-1}(x - \mu_k).$$

- Expanding and dropping the terms that are constant across classes, we get the **linear discriminant function**:

$$\delta_k(x) = x^T\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + \log \pi_k.$$

The classifier is given by $g(x) = \arg\max_k \delta_k(x)$ with $\delta_k(x)$ the above linear discriminant function.

Parameter Estimation for LDA

The parameters of the Gaussian model are estimated from the training data:

- **Priors:** $\hat{\pi}_k = N_k/N$, the proportion of training samples in class k.
- **Class Means:** $\hat{\mu}_k = \frac{1}{N_k}\sum_{g_i=k} x_i$, the average of the inputs for class k.
- **Common Covariance:** A pooled estimate of the covariance matrix:

$$\hat{\Sigma} = \frac{1}{N-K}\sum_{k=1}^{K}\sum_{g_i=k}(x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T.$$

These estimates are then plugged into the discriminant function to obtain $\hat{\delta}_k(x)$. The decision boundary between each pair of classes k and l is then described by $\{x : \hat{\delta}_k(x) = \hat{\delta}_l(x)\}$.

Connections to the Naive Approach

The Two-Class Case ($K = 2$)

For two classes, the least-squares regression approach is closely related to LDA:

- The direction of the decision boundary (orientation of the hyperplane) is identical to the LDA direction, up to a scaling factor (see Exercise 3.1).
- However, the intercept (the position of the boundary) is generally different. The two decision rules only coincide if the training classes have equal sizes.

The Multi-Class Case ($K \geq 3$)

- The methods are not equivalent. Linear regression on indicators can be severely compromised by "masking," where the model fails to separate certain classes.
- LDA finds the optimal subspace for discriminating between the class centroids, whereas the regression approach finds a suboptimal one.

Quadratic Discriminant Analysis (QDA)

QDA

- Like LDA, we assume $X|G = k \sim \mathcal{N}(\mu_k, \Sigma_k)$.
- Unlike LDA, we allow each class to have its own covariance matrix Σ_k.

When we derive the discriminant function, the quadratic term $x^T \Sigma_k^{-1} x$ **no longer cancels**:

$$\delta_k(x) = -\frac{1}{2} \log |\Sigma_k| - \frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1}(x - \mu_k) + \log \pi_k.$$

The discriminant functions are now **quadratic** in x. This means the decision boundaries between classes are quadratic surfaces (conic sections like parabolas, hyperbolas, or ellipses).

QDA is more flexible than LDA but requires estimating many more parameters (esp. for large p), so it has higher variance.

LDA vs. QDA in Practice

- Both LDA and QDA have good track record, and perform well (likely due to the stability of the Gaussian model) on a large and diverse set of classification tasks.
- QDAs requires estimating $K \cdot p(p+1)/2$ covariance parameters, while LDA only requires $p(p+1)/2$.

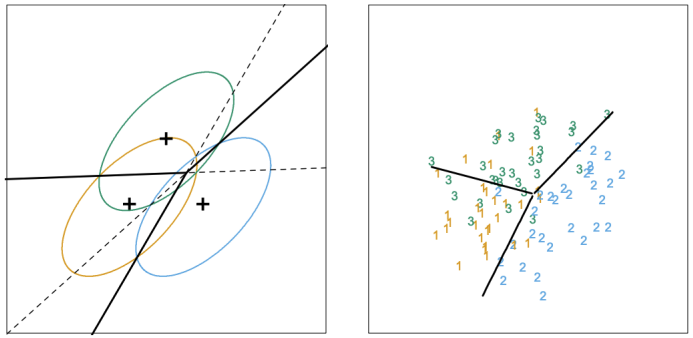


FIGURE 4.5. *The left panel shows three Gaussian distributions, with the same covariance and different means. Included are the contours of constant density enclosing 95% of the probability in each case. The Bayes decision boundaries between each pair of classes are shown (broken straight lines), and the Bayes decision boundaries separating all three classes are the thicker solid lines (a subset of the former). On the right we see a sample of 30 drawn from each Gaussian distribution, and the fitted LDA decision boundaries.*

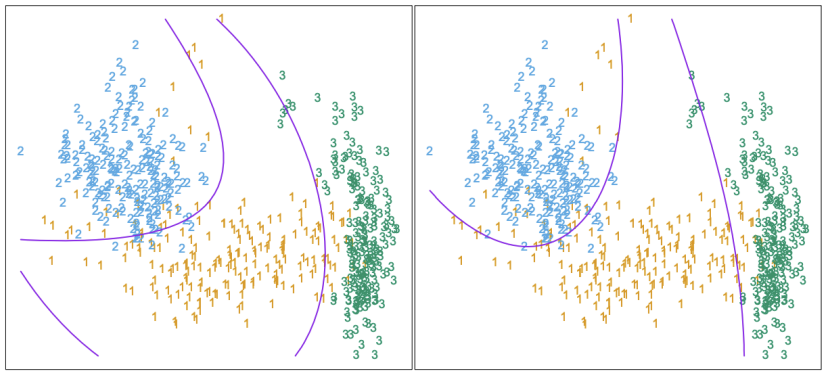


FIGURE 4.6. *Two methods for fitting quadratic boundaries. The left plot shows the quadratic decision boundaries for the data in Figure 4.1 (obtained using LDA in the five-dimensional space $X_1, X_2, X_1X_2, X_1^2, X_2^2$). The right plot shows the quadratic decision boundaries found by QDA. The differences are small, as is usually the case.*

Other Variants (See ESL Ch. 4.3.1-4.3.3)

Regularized Discriminant Analysis

A compromise between LDA and QDA that shrinks the separate covariances toward a common estimate, using the regularized covariance: $\hat{\Sigma}_k(\alpha) = \alpha\hat{\Sigma}_k + (1-\alpha)\hat{\Sigma}$.

- $\hat{\Sigma}_k$ is the covariance matrix of class k.
- $\hat{\Sigma}$ is the pooled covariance matrix (used in LDA).
- The tuning parameter $\alpha \in [0, 1]$ controls the amount of shrinkage.

Reduced-Rank Linear Discriminant Analysis

Performs dimension reduction by finding the linear combination $Z = a^T X$ that maximizes the ratio of between-class to within-class variance. This is achieved by solving Fisher's problem, maximizing the Rayleigh quotient $\max_a \frac{a^T \mathbf{B} a}{a^T \mathbf{W} a}$.

- $\mathbf{B}$ is the between-class covariance matrix of the class centroids.
- $\mathbf{W}$ is the pooled within-class covariance matrix.

The solutions v_ℓ are the eigenvectors from the generalized eigenvalue problem $\mathbf{B}v = \lambda \mathbf{W} v$. The resulting projections $Z_\ell = v_\ell^T X$ are the discriminant coordinates/canonical variates.

Discriminative Approach: Direct Probabilistic Models

- Instead of modeling $P(X|G)$ and using Bayes' rule (like LDA), directly models the posterior probabilities $P(G = k|X = x)$.
- This is a **discriminative** approach, as it focuses on the decision boundary without making assumptions about the distribution of the predictors.
- Recall that here we focus on models that produce linear decision boundaries. If some monotone transformation ϕ of $P(G = k|X = x)$ is linear in x, then the decision boundaries produced are linear.
- A popular model is the **logistic regression model**, which uses the logit transformation.

The Logistic Model (Two Classes)

For two classes (labeled 0 and 1), let $p = P(G = 0|X = x)$ and take $\phi(p) = \log(p/(1-p))$ (the logit/log-odds). This gives the logistic regression model.

Definition 3: The Logit Model

$$\log \frac{P(G = 0|X = x)}{P(G = 1|X = x)} = \beta_0 + \beta^T x.$$

By inverting this transformation, we can express the probabilities directly. Since $P(G = 0|X) + P(G = 1|X) = 1$, we get:

$$P(G = 0|X = x) = \frac{e^{\beta_0 + \beta^T x}}{1 + e^{\beta_0 + \beta^T x}} = \frac{1}{1 + e^{-(\beta_0 + \beta^T x)}},$$
$$P(G = 1|X = x) = \frac{1}{1 + e^{\beta_0 + \beta^T x}}.$$

The decision boundary, where $P(G = 0|X) = P(G = 1|X) = 0.5$, occurs when $\beta_0 + \beta^T x = 0$, which is a hyperplane.

? What if we change the labels to ± 1 instead? See Exercise 3.4.

The Logistic Model (Multi-Class)

For $K > 2$ classes, the model is extended by picking one class as a baseline (e.g., class K) and modeling the log-odds of each other class relative to it.

Definition 4: The Multinomial Logit Model

For $k = 1, \ldots, K-1$:

$$\log \frac{P(G = k|X = x)}{P(G = K|X = x)} = \beta_{k0} + \beta_k^T x.$$

This gives the probabilities via the **softmax transformation**:

$$P(G = k|X = x) = \frac{e^{\beta_{k0} + \beta_k^T x}}{1 + \sum_{l=1}^{K-1} e^{\beta_{l0} + \beta_l^T x}} \quad \text{for } k = 1, \ldots, K-1.$$

And for the baseline class K:

$$P(G = K|X = x) = \frac{1}{1 + \sum_{l=1}^{K-1} e^{\beta_{l0} + \beta_l^T x}}.$$

Fitting Logistic Regression via Maximum Likelihood

Fit the parameters by maximizing the conditional log-likelihood[1] of G given X.

Definition 5: Log-Likelihood (Two-Class Case)

For N observations with a binary response $y_i \in \{0, 1\}$, the log-likelihood is:

$$\ell(\beta) = \sum_{i=1}^{N} \{y_i \log p(x_i; \beta) + (1 - y_i) \log(1 - p(x_i; \beta))\}$$

where $p(x_i; \beta) = P(G = 1 | X = x_i; \beta) = \frac{e^{\beta_{10} + \beta_1^T x_i}}{1 + e^{\beta_{10} + \beta_1^T x_i}}$ and $\beta = (\beta_{10}, \beta_1)$.

- The log-likelihood is a **concave** function of β, which guarantees a unique maximum exists.
- No closed-form solution (why?) for the estimate $\hat{\beta}$: the solution must be found using an iterative algorithm like the **Newton-Raphson algorithm**.

[1] Note: $-\ell(\beta)$ is the cross-entropy loss (= binomial deviance/2).

The Newton-Raphson Algorithm (aka IRLS)

- Let $\mathbf{X}$ be the $N \times (p+1)$ design matrix, $\mathbf{y}$ be the vector of the y_i's, and $\mathbf{p}$ be the vector of fitted probabilities $p(x_i; \beta)$. The first and second derivatives are:

$$\frac{\partial \ell(\beta)}{\partial \beta} = \mathbf{X}^T(\mathbf{y} - \mathbf{p}) \quad \text{(Score vector)}$$
$$\frac{\partial^2 \ell(\beta)}{\partial \beta \partial \beta^T} = -\mathbf{X}^T \mathbf{W} \mathbf{X} \quad \text{(Hessian matrix)}$$

  where $\mathbf{W}$ is an $N \times N$ diagonal matrix with $W_{ii} = p(x_i; \beta)(1 - p(x_i; \beta))$.
- Starting with β^{old}, the updated estimate is:

$$\beta^{\text{new}} \leftarrow \beta^{\text{old}} - \left(\frac{\partial^2 \ell}{\partial \beta \partial \beta^T} \right)^{-1} \frac{\partial \ell}{\partial \beta} \bigg|_{\beta^{\text{old}}} = (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{W} \mathbf{z},$$

  where $\mathbf{z} = \mathbf{X}\beta^{\text{old}} + \mathbf{W}^{-1}(\mathbf{y} - \mathbf{p})$ (adjusted response).
- This is a weighted least squares step for the $z_i = x_i^T \beta^{\text{old}} + \frac{y_i - p_i}{p_i(1-p_i)}$ (aka **Iteratively Reweighted Least Squares (IRLS)**).

Regularization for Logistic Regression

⚠ Regularization is crucial when p is large or when the data are separable, which can cause the MLE estimates to be undefined (why? see Exercise 3.2).

L2-Regularized (Ridge) Logistic Regression

Adds an L2 penalty to the log-likelihood:

$$\max_{\beta} \left\{ \ell(\beta) - \frac{\lambda}{2} \|\beta\|^2 \right\}.$$

Equivalent to placing a Gaussian prior on the coefficients in a Bayesian setting.

L1-Regularized (Lasso) Logistic Regression

Using an L1 penalty allows for feature selection:

$$\max_{\beta} \left\{ \ell(\beta) - \lambda \sum_{j=1}^{p} |\beta_j| \right\}.$$

This objective is still concave but the penalty is non-differentiable at zero.

The LDA and Logistic Regression Connection

There is a deep formal connection between LDA and logistic regression.

LDA's Posterior is a Logit Model

Under the LDA model's specific Gaussian assumptions, the log-odds between any two classes k and l is:

$$\log \frac{P(G=k|X=x)}{P(G=l|X=x)} = \log \frac{\pi_k}{\pi_l} - \frac{1}{2}(\mu_k + \mu_l)^T \Sigma^{-1}(\mu_k - \mu_l) + x^T \Sigma^{-1}(\mu_k - \mu_l).$$

This is a linear function of x, exactly like the logistic regression model:

$$\log \frac{P(G=k|X=x)}{P(G=l|X=x)} = \beta_{kl,0} + \beta_{kl}^T x.$$

Key Difference: LDA fits the parameters by maximizing the full log-likelihood based on the joint density $P(X, G)$, while logistic regression maximizes the conditional log-likelihood based on $P(G|X)$. The logistic model is more general as it doesn't assume Gaussian data.

Summary: LDA vs. Logistic Regression

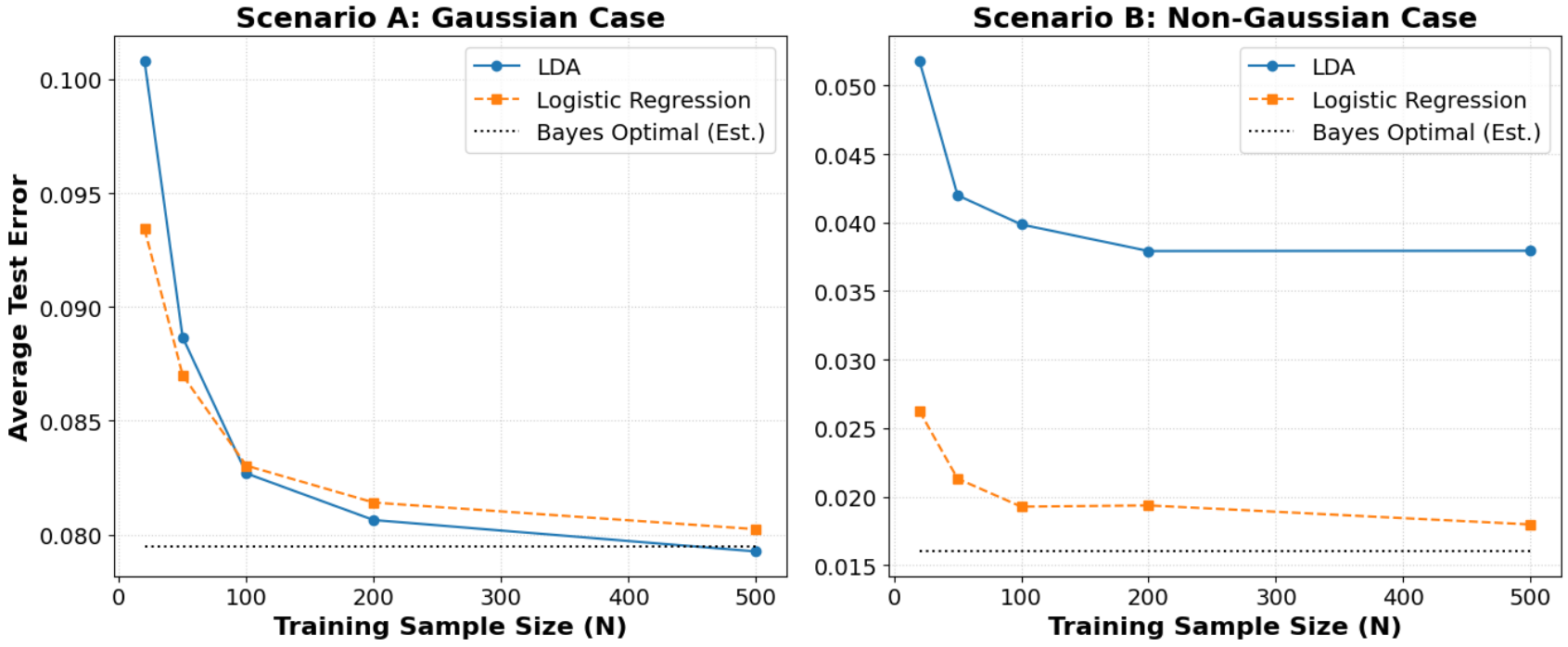
Both methods produce linear decision boundaries. How do they differ?

Linear Discriminant Analysis (LDA)	**Logistic Regression**
Generative model that models $P(X\|G)$ and $P(G)$.	Discriminative model that models $P(G\|X)$ directly.
Makes strong assumption that data is Gaussian with common covariance.	Makes minimal assumptions about the distribution of X.
More stable and efficient if the Gaussian assumption is met.	More robust and general. Often performs better if the LDA assumptions are not met but may need regularization.
Parameters found by maximizing the full log-likelihood (i.e., from moments).	Parameters found by maximizing the conditional log-likelihood.
Data points from all classes influence the boundary estimation.	Only the decision boundary matters; points far away from the boundary have little influence[2].

[2] See Exercise 3.5.

Example: LDA vs. Logistic Regression – Binary Case

- **Scenario A (Gaussian)**: Two Gaussian distributed classes with the same covariance matrix.
- **Scenario B (Non-Gaussian):** Class 0 is a mixture of two Gaussians, class 1 is a single Gaussian.



Direct Approaches: Construct Separating Hyperplanes

For a two-class problem with labels $y_i \in \{-1, 1\}$, consider linear classifiers of the form $\hat{G}(x) = \text{sign}(x^T\beta + \beta_0)$. The decision boundary is a hyperplane defined by $\{x : x^T\beta + \beta_0 = 0\}$.

Definition 6: Linear Separability

A dataset $\{(x_i, y_i)\}_{i=1}^N$ with $y_i \in \{-1, 1\}$ is called linearly separable if there exists at least one hyperplane (β, β_0) that correctly classifies all input samples, i.e., such that $y_i(x_i^T\beta + \beta_0) > 0$ for all i.

? When the data are separable, there are infinitely many such hyperplanes. How do we find one? And which one should we choose?

- The **Perceptron Learning Algorithm** finds *a* separating hyperplane, if one exists (this lecture).
- The **Optimal Separating Hyperplane** (Support Vector Classifier) finds the *best* one by maximizing the margin (next lecture).

Rosenblatt's Perceptron Learning Algorithm (1958)

This algorithm[3] attempts to find a separating hyperplane by minimizing the total distance of misclassified points to the decision boundary.

Definition 7: The Perceptron Criterion

The algorithm seeks to minimize:

$$D(\beta, \beta_0) = -\sum_{i \in \mathcal{M}} y_i(x_i^T \beta + \beta_0),$$

where $\mathcal{M}$ indexes the set of misclassfied points.
This criterion is non-negative and proportional to the sum of distances of the misclassified points to the boundary $x^T \beta + \beta_0 = 0$.

The signed distance of a point x_i to the hyperplane $L = \{x : \beta^T x + \beta_0 = 0\}$ is given by $\text{dist}_{\pm}(x_i, L) = \frac{\beta^T x_i + \beta_0}{\|\beta\|}$. The sign indicates on which "side" of the hyperplane the point lies (relative to the orientation of the normal vector β).

[3] Not widely used these days (why?), mostly of historical and pedagogical interest only as it is the simplest example of online learning.

Perceptron Learning Algorithm

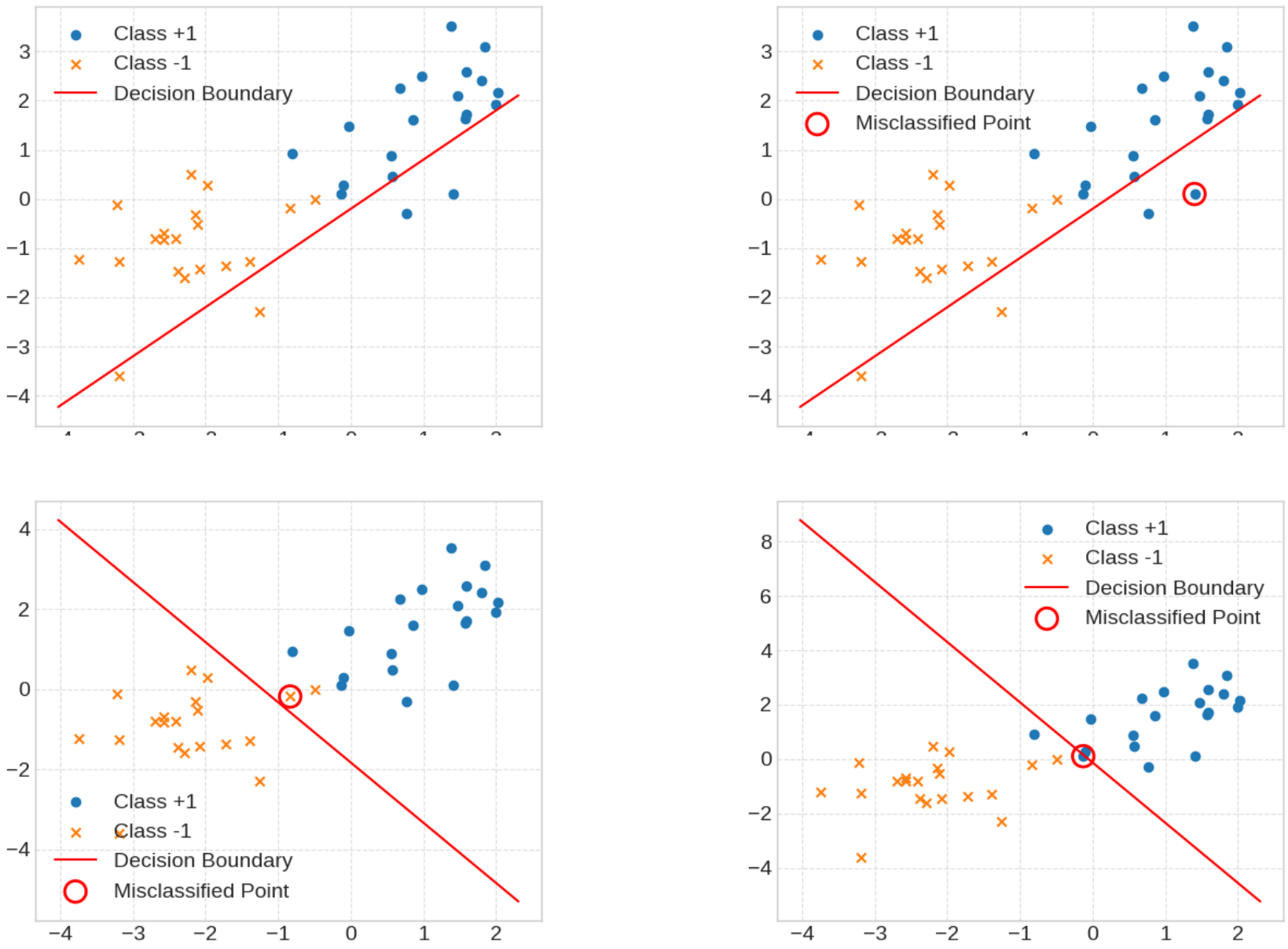
Algorithm 1 Perceptron Algorithm

Input: Training set $\{(x_i, y_i)\}_{i=1}^N$ with $x_i \in \mathbb{R}^d$, $y_i \in \{-1, +1\}$, $\eta =$ learning rate
Initialize: $\beta \leftarrow \beta^{init} \in \mathbb{R}^d$, $\beta_0 \leftarrow \beta_0^{init}$
repeat
  for $i = 1, \ldots, N$ **do**
    Prediction: $\hat{y}_i \leftarrow \mathrm{sign}(x_i^\top \beta + \beta_0)$
    if $\hat{y}_i \neq y_i$ **then**
      $\beta \leftarrow \beta + \eta\, y_i x_i \quad \beta_0 \leftarrow \beta_0 + \eta\, y_i$
    end
  end
until *convergence or maximum iterations*;

- **Convergence:** For linearly separable data, it is guaranteed to converge to a separating hyperplane in a finite number of steps (Exercise 3.3).
- **Non-Uniqueness:** The solution found depends on the starting values. For separable data, there are many possible solutions.

⚠ If the data are *not* linearly separable, the algorithm will not converge (it will instead cycle through different solutions).

Visualizing the Perceptron Algorithm ($\mathcal{X} = \mathbb{R}^2$)



Exercises

Exercise 3

1. Solve Exercise 4.2 in ESL.
2. Solve Exercise 4.5 in ESL.
3. Solve Exercise 4.6 in ESL.
4. Let the class labels be $Y \in \{-1, 1\}$. The logistic regression model defines the probability of the positive class ($Y = 1$) as

$$P(Y = 1|X = x) = \sigma(\beta_0 + \beta^T x),$$

where $\sigma(z) = \frac{1}{1+e^{-z}}$ (sigmoid function). Show that the negative log-likelihood (NLL) for a single observation (x, y), defined as $-\log P(Y = y|X = x)$, simplifies to the logistic loss: $L(\beta_0, \beta) = \log(1 + e^{-y(\beta_0+\beta^T x)})$.

5. Consider the same setup as the previous exercise. Compute the gradient of the NLL with respect to β, and use the result to deduce that for any $\epsilon > 0$, if $y\left(\beta_0 + \beta^\top x\right) > \log\left(\frac{\|x\|}{\epsilon}\right)$, then $\|\nabla_\beta L(\beta_0, \beta)\| < \epsilon$.

Exercises

Exercise 3 (Experimental)

Generate a 2D synthetic dataset with two classes ($Y = 0$ and $Y = 1$), each following a multivariate Gaussian distribution: $X|Y = 0 \sim \mathcal{N}(\mu_0, \Sigma_0)$, $X|Y = 1 \sim \mathcal{N}(\mu_1, \Sigma_1)$, where $\mu_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, $\Sigma_0 = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 1 \end{bmatrix}$, $\mu_1 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$, $\Sigma_1 = \begin{bmatrix} 1 & -0.3 \\ -0.3 & 1 \end{bmatrix}$. Generate 200 samples per class and then split the data into training (70%) and test sets (30%).

1. Implement a classification model using: LDA, QDA, logistic regression, and perceptron. Fit each model on the training data and compute the misclassification error on the test data.
2. Plot the decision boundaries of each model together with the test data points.
3. Find the optimal Bayes classifier for this synthetic setting and compare its misclassification error on the test set to the models above.
4. Discuss how the assumptions of each model affect their performance relative to the Bayes classifier. Explore other settings such as when the ground truth class means are further apart and the effect of regularization tuning.