# Unsupervised Learning with Permuted Data

Technical Report UCI-ICS 03-03
School of Information and Computer Science
University of California, Irvine

Sergey Kirshner, Sridevi Parise, Padhraic Smyth,
Information and Computer Science
University of California, Irvine
Irvine, CA 92697-3425
{skirshne,sparise,smyth}@ics.uci.edu

February 21, 2003

**Abstract**

We consider the problem of unsupervised learning from a matrix of data vectors where in each row the observed values can be randomly permuted in an unknown fashion. Such problems arise naturally in areas such as computer vision and text modeling where measurements need not be in correspondence with the correct features. We provide a general theoretical characterization of the difficulty of "unscrambling" the values of the rows for such problems and relate the optimal error rate to the well-known concept of the Bayes classification error rate. For known parametric distributions we derive closed-form expressions for the optimal error rate that provide insight into what makes this problem difficult in practice. Finally, we show how the Expectation-Maximization procedure can be used to simultaneously estimate both a probabilistic model for the features as well as a distribution over the correspondence of the row values.

# 1 Introduction

There are a number of real-world machine learning problems that can be characterized as follows: there are $N$ objects of interest, where for each object we measure a number of features or attributes of the objects but we do not necessarily know the correspondence between the measurements for different objects. In this paper we focus specifically on the case where the feature values have been permuted in some unknown manner. Table 1 shows a simple example of this type of permutation problem. We would like to be able learn the joint probability density of the original data on the left given only the permuted data and knowledge of the type of permutations that may have been applied to the data (e.g., cyclic shifts). Two questions naturally arise: (a) how hard is this type of learning problem in general? and (b) what kinds of algorithms can we use to solve this problem in practice?

In considering the first problem, our intuition tells us that the "more different" the features in the original (unpermuted) table are then the "easier" the unscrambling problem may be. For example, in Table 1, the distributions of each individual feature in the table on the left appear quite different from each other, so that one hopes that given enough data one could eventually recover a model for the original data given only permuted data. In Section 2 we make this notion of learnability precise by introducing the notion of a Bayes-optimal permutation error rate. In Section 3 we show that under certain conditions this error rate is upper-bounded by an appropriately defined Bayes-optimal classification error rate, confirming the intuition that the ability to unmix the row-values should be related to the overlap of the column densities. In Section 4 we derive closed-form expressions for the permutation error rate for specific parametric models — finding for example that negative correlation among Gaussian columns can make the unmixing problem significantly harder, while positive correlation can make it quite easy. In Section 5 we address the second question (how to simultaneously unlearn the mixing and estimate the original joint density) using an EM framework, and provide various experimental results to illustrate how these learning algorithms work. Conclusions are presented in Section 6. The primary novel contribution of this paper is the introduction and analysis of the notion of Bayes-optimal error rates for "unscrambling" the permutations, providing a lower bound on the performance of any unsupervised learning algorithm for this problem.

Our interest in this "learning from permuted data" problem is motivated by recent work in applying machine learning techniques to astronomical image data [3]. In this problem each image consists of three intensity "blobs" that represent a particular type of galactic structure of interest to astronomers. For each blob a vector of features can be extracted, such as mean intensity, ellipticity, and so forth. Astronomers can visually identify a central "core" blob, and right and left "lobe" blobs in each image. However, in the training data the groups of features are not associated with

Table 1: An example of the permutation problem

| Original Data | | | | Permuted Data | | | | Permutations |
|---|---|---|---|---|---|---|---|---|
| 0.2 | 10 | 4 | 56 | 10 | 4 | 56 | 0.2 | $(2,3,4,1)$ |
| 0.1 | 12 | 4 | 62 | 62 | 0.1 | 4 | 12 | $(4,1,3,2)$ |
| 0.1 | 11 | 3 | 70 | 11 | 3 | 0.1 | 70 | $(2,3,1,4)$ |
| 0.3 | 14 | 4 | 61 | 61 | 0.3 | 14 | 4 | $(4,1,2,3)$ |

any labels that identify whether they are from the center, left, or right blob — this information is hidden and must be estimated from the data. This is a version of the permutation problem described earlier, but where sets of feature values (rather than individual feature values) are being permuted in each row the training data matrix. In our prior work [3] we developed an EM algorithm to solve this image analysis problem and focused on domain-specific aspects of the astronomy application.

Problems similar to the permutation problem occur in computer vision where, for example, features in the form of landmarks are calculated for an object of interest in an image (such as a face) but the features are not necessarily in correspondence across different images. Similar problems also arise in language modeling and information extraction, e.g., in learning models for bibliographic references in documents, where different text fields can occur in different positions depending on the bibliographic style being used [4]. A significant step forward has occurred in recent years with the realization that many of these types of correspondence issues can be cast as machine learning problems, viewing the unknown correspondences as hidden variables that can be estimated from the data using techniques such as Expectation-Maximization (EM), e.g., in vision [2]. Much of this prior work takes advantage of domain-specific information to help solve the correspondence problem, e.g., the use of prior knowledge of likely types of spatial deformations in images, or sequential constraints on text formation in information extraction. In contrast, in this paper we focus on a more abstract theoretical characterization of learning from permuted data.

# 2 Probabilistic Generative Models

## 2.1 Notation

Let $\mathbf{x} = (\vec{x}_1, \ldots, \vec{x}_i, \ldots, \vec{x}_c)$ be composed of $c$ feature-vectors $\vec{x}_i$, with each vector $\vec{x}_i$ taking values from the same $d$-dimensional set S. Thus, $\mathbf{x}$ has dimension $c \times d$ and takes values in the set $\mathrm{S}^c$.

**Example 1.** Let each $\vec{x}_i$ be a one-dimensional real-valued feature-vector. In this case S is the real line ($\mathrm{S} = \mathbb{R}$) with $d = 1$, each $\vec{x}_i$ is a scalar, and $\mathrm{S}^c = \mathbb{R}^c$.

**Example 2.** Let each feature-vector $\vec{x}_i$ take values on the vertices of a $d$-dimensional unit hypercube. Here $\mathrm{S} = \{0,1\}^d$ and $\mathrm{S}^c = \{0,1\}^{c \times d}$.

**Example 3.** Consider the case where $A = \{\mathrm{a}, \mathrm{b}, \ldots, \mathrm{z}\}$ is a set of letters in an alphabet and let $\mathrm{S} = A \times \mathbb{R}$. In this case each feature-vector takes values as pairs of a letter and a real number, i.e. $d = 2$, and the space $\mathrm{S}^c$ is $2c$-dimensional.

We define $p(\mathbf{x})$ as a probability density (distribution) function over the set $\mathrm{S}^c$. For example, in Example 1 above $p(\mathbf{x})$ could be a $c$-dimensional multivariate Gaussian density.

## 2.2 A Generative Model for Permuted Data

Our generative model consists of two parts:

- In the first part we generate samples from $p(\mathbf{x})$ in a standard manner — throughout this paper we assume independent and identically distributed random samples. In this manner we can generate a data matrix of $N$ rows and $c$ columns, where each column has dimension $d$.

Table 2: Probability distributions for Example 5.

| $\mathbf{x}$ | $p(\mathbf{x})$ | $q(\mathbf{x})$ |
|---|---|---|
| $(0,0)$ | 0.28 | 0.28 |
| $(0,1)$ | 0.42 | 0.27 |
| $(1,0)$ | 0.12 | 0.27 |
| $(1,1)$ | 0.18 | 0.18 |

- The second part of the generative model randomly applies a permutation $\rho$ to each row of the data matrix in the following manner. Let $\mathcal{P} = \{\rho_1, \ldots, \rho_m\}$ be a set of permutations defined on $(1, \ldots, c)$. For example, $\mathcal{P}$ could be the set of all $c$ cyclic shifts, e.g., $c = 3, \rho_1 = (1,2,3)$, $\rho_2 = (2,3,1)$, and $\rho_3 = (3,1,2)$. For each row $\mathbf{x}$ of the data matrix a permutation $\rho \in \mathcal{P}$ is randomly selected according to a probability distribution $p(\rho)$ over $\mathcal{P}$. The components of $\mathbf{x}$ are then permuted according to the selected $\rho$ to obtain a permuted vector taking values in the same set $\mathrm{S}^c$.

The size of $\mathcal{P}$, $|\mathcal{P}|$, is denoted by $m$. Note that if all possible permutations are allowed then $m = c!$. Unless stated otherwise, in this paper we will generally assume that all permutations in $\mathcal{P}$ are equally likely, i.e. $p(\rho_j) = \frac{1}{m},\ j = 1, \ldots, m$.

### 2.3 Probability Densities for Permuted Data

It is useful to express the probability density of permuted vector, call it $q(\mathbf{x})$, as a function of (a) the density of the original data rows $p(\mathbf{x})$, and (b) the distribution over permutations $p(\rho)$. In the remainder of the paper whenever the symbol $q$ is used it is implicitly assumed that the argument $\mathbf{x}$ has been permuted. Note that $q(\mathbf{x})$ can be expressed as a finite mixture over all $m$ possible permutations that could have led to the generation of $\mathbf{x}$:

$$\begin{aligned} q(\mathbf{x}) &= \sum_{j=1}^{m} q(\mathbf{x}|\,\rho_j) p(\rho_j) = \sum_{j=1}^{m} q(\vec{x}_1, \ldots, \vec{x}_c|\,\rho_j)\, p(\rho_j) \\ &= \sum_{j=1}^{m} p\left(\rho_j^{-1}(\vec{x}_1), \ldots, \rho_j^{-1}(\vec{x}_c)\right) p(\rho_j) \\ &= \sum_{j=1}^{m} p\left(\vec{x}_{\rho_j^{-1}(1)}, \ldots, \vec{x}_{\rho_j^{-1}(c)}\right) p(\rho_j) \end{aligned} \tag{1}$$

where $\rho_j^{-1}$ is the unique inverse permutation for $\rho_j$.

**Example 4.** Let $c$=3, and $\mathcal{P} = \{\rho_1, \rho_2, \rho_3\}$ be a set of cyclic shifts: $\rho_1 = (1,2,3)$, $\rho_2 = (2,3,1)$, and $\rho_3 = (3,1,2)$. If $\mathbf{x} = (\vec{x}_1, \vec{x}_2, \vec{x}_3)$ is a permuted vector, it could have been obtained from one of three possible permutations, as reflected by the mixture model:

$$\begin{aligned} q(\mathbf{x}) &= q(\vec{x}_1, \vec{x}_2, \vec{x}_3|\rho_1)\, p(\rho_1) + q(\vec{x}_1, \vec{x}_2, \vec{x}_3|\rho_2)\, p(\rho_2) + q(\vec{x}_1, \vec{x}_2, \vec{x}_3|\rho_3)\, p(\rho_3) \\ &= p(\vec{x}_1, \vec{x}_2, \vec{x}_3)\, p(\rho_1) + p(\vec{x}_3, \vec{x}_1, \vec{x}_2)\, p(\rho_2) + p(\vec{x}_2, \vec{x}_3, \vec{x}_1)\, p(\rho_3)\,. \end{aligned}$$

An important point is that $p$ and $q$ are not the same distribution although both are defined over $\mathrm{S}^c$.

**Example 5.** Let $\mathrm{S} = \{0, 1\}$ with $c = 2$. Let $\mathcal{P} = \{\rho_1, \rho_2\}$ where $\rho_1 = (1, 2)$ and $\rho_2 = (2, 1)$. Let $p(\mathbf{x})$ be as defined in the Table 2. Assume $p(\rho_1) = p(\rho_2) = 0.5$. The resulting $q$ distribution on permuted vectors $\mathbf{x}$ is also listed in the Table 2. $p \neq q$ since, for example,

$$\begin{aligned} q(0,1) &= \sum_{j=1}^{2} p\left(0_{\rho_j^{-1}(1)}, 1_{\rho_j^{-1}(2)}\right) p(\rho_j) \\ &= p(0,1) \times 0.5 + p(1,0) \times 0.5 \\ &= 0.42 \times 0.5 + 0.12 \times 0.5 = 0.27 \\ &\neq 0.42 = p(0,1) . \end{aligned}$$

## 2.4 Inference and Learning

There are two problems of direct interest. In the first problem, we assume that $p(\mathbf{x})$ is known, and that the set of permutations $\mathcal{P}$ and their probabilities $p(\rho)$ are also known. Then, given a permuted vector $\mathbf{x}$, we can calculate

$$q(\rho_j|\mathbf{x}) = \frac{q(\mathbf{x}|\rho_j)\, p(\rho_j)}{\sum_i q(\mathbf{x}|\rho_i)\, p(\rho_i)}, \qquad j = 1, \ldots, m$$

using the mixture model in Equation 1 and Bayes rule. This allows us to identify (for example) the most likely permutation, $\arg\max_j q(\rho_j|\mathbf{x})$. It is straightforward to show that this decision rule is Bayes-optimal in that no other decision rule can achieve a lower average error in terms of identifying the permutations [1]. Of interest here is the probability that we make an error (on average) using this decision rule, i.e., what is the optimal error rate achievable in terms of unscrambling the row values. Here an "error" occurs whenever the most likely permutation is not the same as the true permutation that generated $\mathbf{x}$. We will refer to this error rate as the Bayes-optimal permutation error rate, defined as

$$E_P^\star = \int_{\mathrm{S}^c} q(\mathbf{x}) \times \left(1 - \max_j q(\rho_j|\mathbf{x})\right) \mathrm{d}\mathbf{x}$$

with the superscript $\star$ referring to "Bayes-optimal" and subscript $P$ referring to "permutation".

In the second problem, we are given the set of permuted vectors $D = \left\{\mathbf{x}^1, \ldots, \mathbf{x}^N\right\}$, a set of permutations $\mathcal{P}$, and an assumed functional form for $p(\mathbf{x})$. Our task in this case is to estimate the parameters of $p(\mathbf{x})$, the probabilities $p(\rho)$, and for each row $\mathbf{x}$ to estimate the probability that permutation $\rho_j$ was used to generate it. This problem is discussed in Section 5.

The two problems above are intimately related. The error rate of any learning algorithm in problem 2 (in terms of identifying permutations) will be lower-bounded by the Bayes-optimal permutation error rate $E_P^\star$ as defined in problem 1. Thus, $E_P^\star$ is a fundamental characteristic of the difficulty of learning in the presence of permutations and it is of direct interest to study it. In what follows we first show in Section 3 that $E_P^\star$ is it itself upper-bounded (under certain assumptions) by a well-known characteristic of density overlap (the Bayes-optimal classification error rate), and we then in Section 4 derive closed form expressions for $E_P^\star$ for specific simple forms for $p(\mathbf{x})$.

## 3 Analysis of Bayes-Optimal Error Rates

Recall that $\mathbf{x} = (\vec{x}_1, \ldots, \vec{x}_c)$. We can define a marginal distribution for each feature-vector $\vec{x}_i$, $i = 1, \ldots, c$ as

$$p(\vec{x}_i) = \int_{\mathrm{S}} \ldots \int_{\mathrm{S}} p(\vec{x}_1, \ldots, \vec{x}_{i-1}, \vec{x}_i, \vec{x}_{i+1}, \ldots, \vec{x}_c) \, \mathrm{d}\vec{x}_1 \ldots \mathrm{d}\vec{x}_{i-1} \mathrm{d}\vec{x}_{i+1} \ldots \mathrm{d}\vec{x}_c,$$

i.e., the marginal density for $\vec{x}_i$ defined on the set S. Each of the $c$ features has a similar marginal density on the same set S. We will use $p(\vec{x}|C_i) = p(\vec{x}_i)$ to denote the marginal density of $\vec{x}_i$ on the set S.

We now have $c$ different densities defined on S, which in turn defines a finite mixture $p_M(\vec{x})$ on S:

$$p_M(\vec{x}) = \sum_{i=1}^{c} p(\vec{x}|C_i) \times p(C_i)$$

where $p(C_i) = \frac{1}{c}$, since all marginals have equal weight in the process of defining the mixture. In the space S consider a classification problem with $c$ classes, where, given a measurement $\vec{x} \in \mathrm{S}$, we infer the most likely feature-vector $\vec{x}_j$ that it originated from, $j = 1, \ldots, c$. The Bayes-optimal classification rate for this problem is defined as

$$E_C^\star = \int_{\mathrm{S}} p_M(\vec{x}) \times \left(1 - \max_j p(C_j|\vec{x})\right) \mathrm{d}\vec{x}.$$

Intuitively, $E_C^\star$ is the error rate obtained if we were given vectors $\vec{x}_i$ one at a time, and asked to identify which of the $c$ "columns" they originated from, based on knowing each of the $p(\vec{x}|C_i)$ densities, $i = 1, \ldots, c$. $E_C^\star$ is proportional to the overlap of the individual feature densities $p(\vec{x}_i)$ in the space S. For example, for the data on the left in Table 1 we would expect the overlap of the 4 densities, as reflected by $E_C^\star$, to be quite small. Furthermore, we would expect intuitively that the permutation error rate $E_P^\star$ should also be low in this case, and more generally that it should be related to $E_C^\star$ in some manner. In what follows below we quantify this intuition. Specifically, under certain choices of $\mathcal{P}$, we show that $E_P^\star$ is upper-bounded by $E_C^\star$.

**Definition 1.** For $\mathcal{P}$, let $k$ be a **key** index if $(\rho_1(k), \ldots, \rho_m(k))$ is a permutation of $(1, \ldots, c)$.

Note that $\mathcal{P}$ having a key implies $m = |\mathcal{P}| = c$. The set of all cyclic shifts for example has a key.

**Example 6.** For $\mathcal{P} = \{\rho_1, \rho_2\}$ with $\rho_1 = (1, 2)$ and $\rho_2 = (2, 1)$, both indices 1 and 2 are keys.

**Example 7.** A set of permutations $\mathcal{P} = \{\rho_1, \rho_2, \rho_3, \rho_4\}$ with $\rho_1 = (1, 2, 3, 4)$, $\rho_2 = (2, 1, 3, 4)$, $\rho_3 = (1, 2, 4, 3)$, and $\rho_4 = (2, 1, 4, 3)$ does not have a key.

**Theorem 1.** If a set of permutations $\mathcal{P}$ for a permutation problem has a key, and if each permutation is equally likely, then

$$E_p^\star \leq E_c^\star.$$

*Proof.*

$$\begin{aligned} E_C^\star &= \int_{\mathrm{S}} p_M(\vec{x}) \times \left(1 - \max_i p(C_i|\vec{x})\right) \mathrm{d}\vec{x} = \int_{\mathrm{S}} p_M(\vec{x}) \, \mathrm{d}\vec{x} - \int_{\mathrm{S}} \max_i \left(p_M(\vec{x}) \, p(C_i|\vec{x})\right) \mathrm{d}\vec{x} \\ &= 1 - \int_{\mathrm{S}} \max_i \left(p(C_i) \, p(\vec{x}|C_i)\right) \mathrm{d}\vec{x} = 1 - \frac{1}{c} \int_{\mathrm{S}} p\left(\vec{x}|C_{I(\vec{x})}\right) \mathrm{d}\vec{x}, \end{aligned}$$

where

$$I(\vec{x}) = \arg\max_i p(\vec{x}|C_i).$$

Let $k$ be a key for $\mathcal{P}$. Let $Q(\vec{x}_k)$ be such that

$$\rho_{Q(\vec{x}_k)}(k) = I(\vec{x}_k).$$

By the definition of a key, existence of such $Q(\vec{x}_k)$ is guaranteed for all $\vec{x} \in \mathrm{S}$. Note that

$$\rho_{Q(\vec{x}_k)}(k) = I(\vec{x}_k) \Leftrightarrow \rho^{-1}_{Q(\vec{x}_k)}(I(\vec{x}_k)) = k.$$

Recalling the definition of $E^\star_P$,

$$\begin{aligned} E^\star_P &= \int_{\mathrm{S}^c} q(\mathbf{x}) \times \left(1 - \max_i q(\rho_i|\mathbf{x})\right) \mathrm{d}\mathbf{x} = \int_{\mathrm{S}^c} q(\mathbf{x})\,\mathrm{d}\mathbf{x} - \int_{\mathrm{S}^c} \max_i \left(q(\mathbf{x})\, q(\rho_i|\mathbf{x})\right) \mathrm{d}\mathbf{x} \\ &= 1 - \int_{\mathrm{S}^c} \max_i \left(p(\rho_i)\, q(\mathbf{x}|\rho_i)\right) \mathrm{d}\mathbf{x}. \end{aligned}$$

By Equation 1,

$$\begin{aligned} E^\star_P &= 1 - \int_{\mathrm{S}} \cdots \int_{\mathrm{S}} \max_i \left(p(\rho_i)\, p\left(\vec{x}_{\rho_i^{-1}(1)}, \ldots, \vec{x}_{\rho_i^{-1}(c)}\right)\right) \mathrm{d}\vec{x}_1 \ldots \mathrm{d}\vec{x}_c \\ &= 1 - \frac{1}{c} \int_{\mathrm{S}} \cdots \int_{\mathrm{S}} \max_i p\left(\vec{x}_{\rho_i^{-1}(1)}, \ldots, \vec{x}_{\rho_i^{-1}(c)}\right) \mathrm{d}\vec{x}_1 \ldots \mathrm{d}\vec{x}_c. \end{aligned}$$

At this point we can use the property of the key:

$$\max_i p\left(\vec{x}_{\rho_i^{-1}(1)}, \ldots, \vec{x}_{\rho_i^{-1}(c)}\right) \geq p\left(\vec{x}_{\rho^{-1}_{Q(\vec{x}_k)}(1)}, \ldots, x_{\rho^{-1}_{Q(\vec{x}_k)}(c)}\right).$$

$$\begin{aligned} E^\star_P &\leq 1 - \frac{1}{c} \int_{\mathrm{S}} \cdots \int_{\mathrm{S}} p\left(\vec{x}_{\rho^{-1}_{Q(\vec{x}_k)}(1)}, \ldots, \vec{x}_{\rho^{-1}_{Q(\vec{x}_k)}(c)}\right) \mathrm{d}\vec{x}_1 \ldots \mathrm{d}\vec{x}_c \\ &= 1 - \frac{1}{c} \int_{\mathrm{S}} \left( \int_{\mathrm{S}} \cdots \int_{\mathrm{S}} p\left(\vec{x}_{\rho^{-1}_{Q(\vec{x}_k)}(1)}, \ldots, \vec{x}_{\rho^{-1}_{Q(\vec{x}_k)}(I(\vec{x}_k)-1)}, \vec{x}_k, \vec{x}_{\rho^{-1}_{Q(\vec{x}_k)}(I(\vec{x}_k)+1)}, \ldots, x_{\rho^{-1}_{Q(\vec{x}_k)}(c)}\right) \right. \\ &\qquad \left. \mathrm{d}\vec{x}_1 \ldots \mathrm{d}\vec{x}_{k-1} \mathrm{d}\vec{x}_{k+1} \ldots \mathrm{d}\vec{x}_c \right) \mathrm{d}\vec{x}_k \\ &= 1 - \frac{1}{c} \int_{\mathrm{S}} p\left(\vec{x}_k | C_{I(\vec{x}_k)}\right) \mathrm{d}\vec{x}_k = E^\star_C. \end{aligned}$$

□

The theorem shows that under certain assumptions, the permutation problem is easier than a corresponding version of the classification problem. This generally agrees with our intuition since in the classification version of the problem we are classifying feature values one at a time in terms of which column they are thought to have originated from, whereas in the permutation version of the problem we are simultaneously classifying $c$ values together and have the additional information available that the $c$ values must all be assigned to different classes (in the presence of a key).

Note that if the set of permutations $\mathcal{P}$ does not have a key, $E^\star_P$ may in fact be larger than $E^\star_C$.

**Example 8.** Let $\mathcal{P}$ be defined as in Example 7. Let $\mathrm{S} = \{v_1, v_2\}$. Let $p(v_j|c_i)$ be defined as in Table 3. If we define the joint $p(\vec{x}_1, \vec{x}_2, \vec{x}_3, \vec{x}_4)$ as $p(\vec{x}_1)\, p(\vec{x}_2)\, p(\vec{x}_3)\, p(\vec{x}_4)$, it can be shown that in this case, $E^\star_C = 0.55 < 0.6975 = E^\star_P$.

Table 3: Distribution for Example 8.

| $p(v_j\|C_i)$ | $i=1$ | $i=2$ | $i=3$ | $i=4$ |
|---|---|---|---|---|
| $j=1$ | 0.9 | 0.8 | 0.2 | 0.1 |
| $j=2$ | 0.1 | 0.2 | 0.8 | 0.9 |

# 4 Analysis of Permutation Error Rates

In this section we derive closed-form expressions for the Bayes-optimal permutation error rate $E_P^\star$ for specific functional forms for $p(\mathbf{x})$ and we use these expressions to show how changes in the parameters of $p(\mathbf{x})$ can make learning and inference harder or easier.

## 4.1 Gaussian Features

We begin with the case of Gaussian features, since the Gaussian model is both amenable to analysis and widely used in practice.

### 4.1.1 Case 1: Two Independent Features

Consider the case when $c=2$, $d=1$ and $\mathrm{S}=\mathbb{R}$. Let $p(\mathbf{x})$ be a Gaussian with covariance matrix of the form $\sigma^2 I$ where $I$ is the identity matrix (the features are independent Gaussians with equal variances). Thus,

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

where

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \text{ and } \boldsymbol{\Sigma} = \begin{pmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{pmatrix}$$

We have $m=2$ with $\rho_1=(1,2)$ and $\rho_2=(2,1)$.

$$q(\mathbf{x}|\rho_1) = p(x_1, x_2) = \mathcal{N}((x_1, x_2)|\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

and

$$q(\mathbf{x}|\rho_2) = p(x_2, x_1) = \mathcal{N}((x_1, x_2)\,|\tilde{\boldsymbol{\mu}}, \boldsymbol{\Sigma}),$$

where

$$\tilde{\boldsymbol{\mu}} = \begin{bmatrix} \mu_2 \\ \mu_1 \end{bmatrix}$$

It is straightforward to show that

$$E_P^\star = \frac{1}{\sqrt{2\pi}} \int_{\frac{r}{2}}^{\infty} e^{-u^2/2} \mathrm{d}u \tag{2}$$

where

$$r^2 = (\boldsymbol{\mu} - \tilde{\boldsymbol{\mu}})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu} - \tilde{\boldsymbol{\mu}}).$$

Therefore, given the functional forms for $q(\mathbf{x}|\rho_1)$ and $q(\mathbf{x}|\rho_2)$,

$$E_P^\star = \frac{1}{\sqrt{2\pi}} \int_{\frac{|\mu_1-\mu_2|}{\sqrt{2}\sigma}}^{\infty} e^{-u^2/2} \mathrm{d}u.$$

The quantity $|\mu_1 - \mu_2| / \sigma$ is a measure of the overlap of the two Gaussians: as overlap increases $E_P^\star$ decreases, and vice-versa. This is exactly the same qualitative behavior as one gets with the Bayes-optimal classification error rate $E_C^\star$ for this problem, except that the range of integration is different. Specifically (using the results for $E_C^\star$ in [1]) we have

$$E_C^\star - E_P^\star = \frac{1}{\sqrt{2\pi}} \int_{\frac{|\mu_1-\mu_2|}{2\sigma}}^{\frac{|\mu_1-\mu_2|}{\sqrt{2}\sigma}} e^{-u^2/2} \mathrm{d}u.$$

In the cases of maximal overlap ($|\mu_1 - \mu_2| / \sigma$ is very large) and minimal overlap (the overlap expression is very small) the difference in the two error rates is very small. The difference between the two types of error is maximized when $\frac{|\mu_2-\mu_1|}{2\sigma} = \sqrt{\ln 2}$.

### 4.1.2 Case 2: Two Correlated Gaussian Features

Next, consider a generalization of Case 1 where the features are no longer assumed to be independent but are allowed to have non-zero correlation $\nu$:

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$$

where

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} \text{ and } \boldsymbol{\Sigma} = \begin{pmatrix} \sigma^2 & \nu \\ \nu & \sigma^2 \end{pmatrix}, \ -\sigma^2 < \nu < \sigma^2.$$

$q(\mathbf{x}|\rho_1)$ and $q(\mathbf{x}|\rho_2)$ are defined as in Case 1, but where $\boldsymbol{\Sigma}$ now has a covariance term $\nu$ in the off-diagonal positions. Using Equation 2 again, we get

$$E_P^\star = \frac{1}{\sqrt{2\pi}} \int_{\frac{|\mu_1-\mu_2|}{\sqrt{2}\sqrt{\sigma^2-\nu}}}^{\infty} e^{-u^2/2} \mathrm{d}u.$$

Thus, as in the independent case, $E_P^\star$ decreases as $|\mu_1 - \mu_2|$ increases and vice-versa.

As $\sigma \to \nu$, the lower limit of the integral approaches $\infty$ and $E_P^\star$ approaches zero. Thus, even though the two Gaussians could be heavily overlapped, as the correlation approaches 1, we can identify permuted *pairs of values* with accuracy approaching 1, in contrast to the Bayes-optimal classification error rate for the same problem which is defined based on classifying each value separately and cannot take advantage of the correlation information. This is a case where the permutation error rate can approach 0 even in cases where the classification error rate (proportional to the overlap of feature densities) can be as high as 0.5.

Interestingly, negative correlation has the opposite effect in that as the correlation coefficient becomes more negative, $E_P^\star$ increases and approaches $E_P^\star$. Intuitively, negative correlation makes the problem harder by effectively leading to more overlap between the two densities. To see this visually, in Figure 1 we plot simulated data from $q(\mathbf{x})$ for the case of very negative correlation, zero correlation, and very positive correlation.

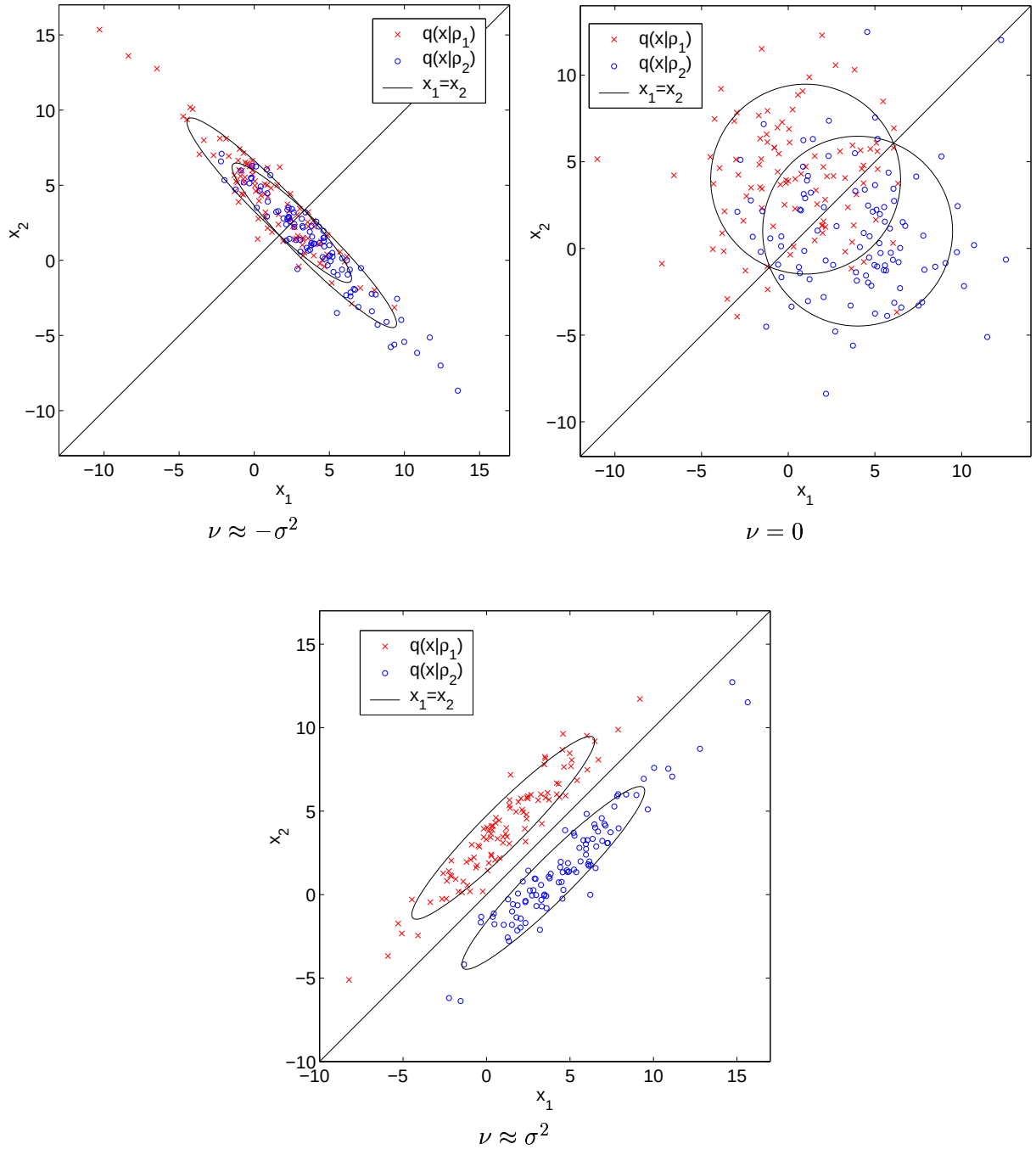


Figure 1: Simulated data (and covariance ellipses) from $q(\mathbf{x}|\rho_1)$ (x's) and $q(\mathbf{x}|\rho_2)$ (o's) for the correlated Gaussian case with equal variances (case 2). The optimal decision boundary corresponds to the line $x_1 = x_2$.

### 4.1.3 Case 3: Unequal Variances

Consider further the case where the Gaussian features have unequal variances $\sigma_1^2$ and $\sigma_2^2$ and covariance $\nu$. In this case, since $q(\mathbf{x}|\rho_1)$ and $q(\mathbf{x}|\rho_2)$ have unequal covariance matrices, there is no closed-form expression for $E_P^\star$ or $E_C^\star$ as we had before. Nevertheless, we can get an understanding of the variation of $E_P^\star$ as a function of $\sigma_1, \sigma_2$ and $\nu$ via simulations. Figure 2 shows some 2-D plots for various values of $\nu$, keeping $\sigma_1$ and $\sigma_2$ fixed. We see that variance inequality changes the nature of the overlap between $q(\mathbf{x}|\rho_1)$ and $q(\mathbf{x}|\rho_2)$ compared to the equal variance case in Figure 1.

We can also calculate empirical error rates (in terms of classifying permutations) using the true model for various values of the parameters (note that these are empirical estimates of $E_P^\star$ by definition). Figure 3 shows the variation of these empirical error rates with $\nu$ for different values of the variances keeping the ratio $\sigma_1\sigma_2$ constant. It can be seen from the plots that $E_P^\star$ depends on both $\nu$ as well as the difference $\left|\sigma_1^2 - \sigma_2^2\right|$, and that the variation of the error rate with $\nu$ is not monotonic.

## 4.2 Categorical Data

Consider the simple case when $c = 2$ and $d = 1$, i.e., $\mathbf{x}$ consists of two scalar features. Assume that the features are discrete and can take one of $V$ values. Let $m = 2$ with $\rho_1 = (1, 2)$ and $\rho_2 = (2, 1)$, and both permutations are assumed to be equally likely. We have,

$$\begin{aligned}
E_P^\star &= \sum_{\mathbf{x}} q(\mathbf{x}) \left(1 - \max_i q(\rho_i|\mathbf{x})\right) \\
&= \sum_{\mathbf{x}} q(\mathbf{x}) - \sum_{\mathbf{x}} q(\mathbf{x}) \max_i q(\rho_i|\mathbf{x}) \\
&= 1 - \frac{1}{2} \sum_{\mathbf{x}} \max_i q(\mathbf{x}|\rho_i) \qquad\qquad \text{where } \mathbf{x} = (x_1, x_2) \\
&= 1 - \frac{1}{2} \sum_{x_1, x_2} \max\{p(x_1, x_2), p(x_2, x_1)\} \\
&= 1 - \frac{1}{4} \sum_{x_1, x_2} \left(|p(x_1, x_2) - p(x_2, x_1)| + p(x_1, x_2) + p(x_2, x_1)\right) \\
&= \frac{1}{2} - \frac{1}{4} \sum_{x_1, x_2} |p(x_1, x_2) - p(x_2, x_1)| .
\end{aligned}$$

Thus, $E_P^\star$ is a function of the quantity $\sum_{x_1,x_2} |p(x_1, x_2) - p(x_2, x_1)|$, which we can call the permutation distance between $\rho_1$ and $\rho_2$. $E_P^\star$ decreases linearly with this distance, reflecting the fact that the more dissimilar the probabilities of each permuted pair of values are from the probabilities of the unpermuted pairs, the more $E_P^\star$ decreases.

For the special case when $V = 2$, i.e. $\mathrm{S} = \{v_1, v_2\}$, and the features are independent, i.e. using the notation defined in Section 3, $p(x_1, x_2) = p(x_1|C_1)\, p(x_2|C_2)$,

$$E_P^\star = \frac{1}{2} - \frac{1}{2} |p(v_1|C_1) - p(v_1|C_2)| .$$

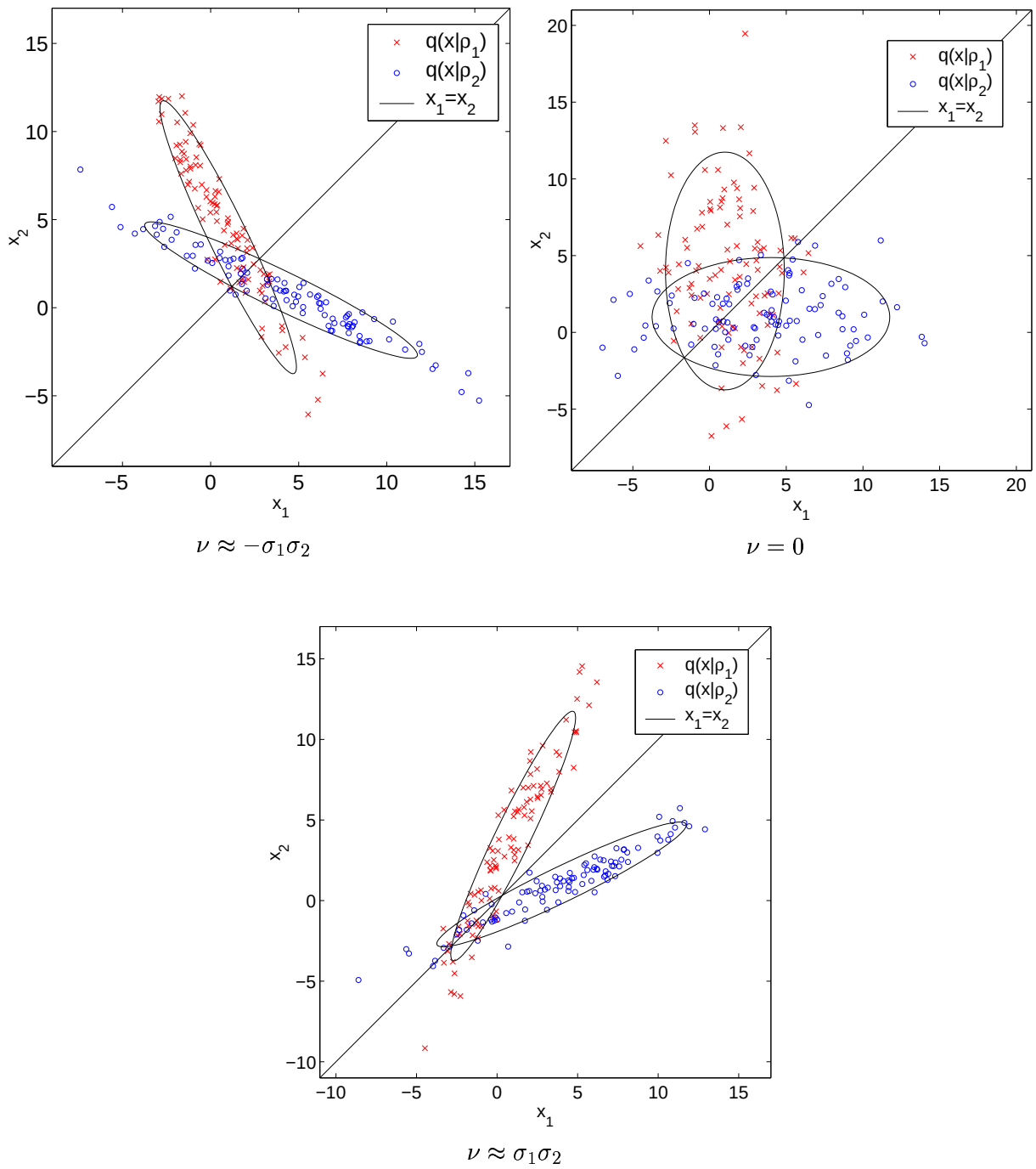


Figure 2: Simulated data (and covariance ellipses) from $q(\mathbf{x}|\rho_1)$ (x's) and $q(\mathbf{x}|\rho_2)$ (o's) for the correlated Gaussian case with unequal variances (case 3). The optimal decision boundary corresponds to the line $x_1 = x_2$.

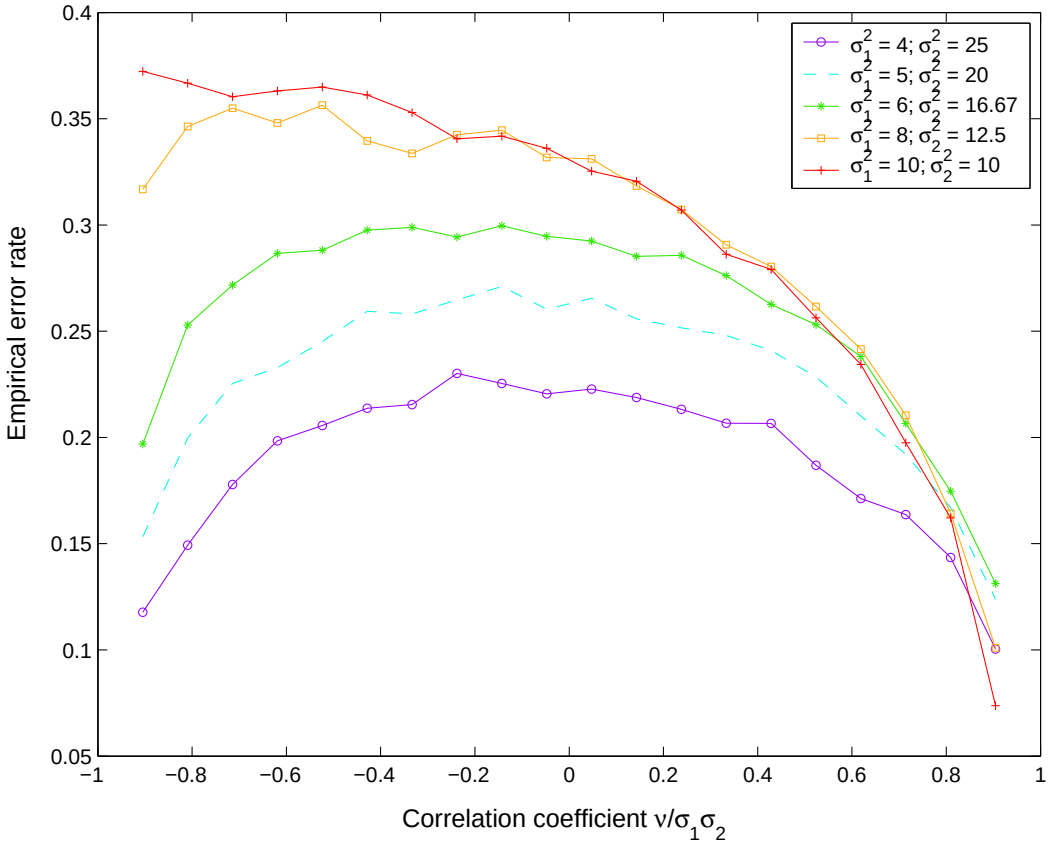


Figure 3: Empirical values of the error rates for Case 3

## 5 Learning from Permuted Data

In this section we briefly comment on the problem of unsupervised learning with permuted data. Space limitations do not permit a complete treatment of the topic: the goal is to illustrate that learning with permutations can be achieved in a practical sense, and to demonstrate that the Bayes-optimal permutation error rate provides an absolute lower bound on the error rate of practical learning algorithms. Just as for standard finite mixtures, mixtures of permutations can suffer from identifiability problems. The same probability distribution $q(\mathbf{x}^{\rho})$ for permuted vectors can be obtained from different sets of distinct probability distributions (up to a permutation of the feature-vectors) of unpermuted vectors. For example, consider $p(\mathbf{x})$ from Example 5 and change the values for $p(0,1)$ and $p(1,0)$ to different values while keeping their sum the same. The resulting distribution $q(\mathbf{x})$ would not change. For these situations, it is impossible to learn the original distribution $p(\mathbf{x})$ from a sample $D$ of distribution $q(\mathbf{x})$. Identifiability issues for mixtures of permutations need to be researched further [5]; however, we suspect that they would mirror that of ordinary mixtures of distributions of the same type as defined on the space of unpermuted vectors.

### 5.1 EM Algorithms for Permuted Data

Consider a "matrix" data set $D = \left\{\mathbf{x}^1, \ldots, \mathbf{x}^N\right\}$ with $N$ rows and $c$ columns (each column being $d$-dimensional) where we assume that $D$ was generated using the generative model described in Section 3. Assume that we know the set of permutations $\mathcal{P}$ and the functional form (but not the parameters) of $p(\mathbf{x})$. If we knew the permutations that generated each data vector $\mathbf{x}^i$, then presumably the problem of estimating the parameters of $p(\mathbf{x})$ using the "unscrambled" data would be straightforward. This suggests the use of the EM framework for this problem, treating the $m$ possible permutations as "hidden" information and using the mixture framework of Equation 1 as the basis for an EM learning algorithm for likelihood maximization.

Letting $\Theta$ be the unknown parameters of $p(\mathbf{x})$, the log-likelihood can be defined as

$$l(\Theta) \quad = \quad \ln q(D|\Theta) = \sum_{i=1}^{N} \ln q\left(\mathbf{x}^i|\Theta\right) = \sum_{i=1}^{N} \ln \sum_{j=1}^{m} p(\rho_j)\, p\left(\rho_j^{-1}\left(\mathbf{x}^i\right)\right).$$

After $\Theta$ has been initialized in some fashion, the parameters are changed iteratively, guaranteeing a non-decreasing log-likelihood at the end of each iteration. In the E-step, the probability of each permutation is estimated for each data vector given the current $\Theta$. In the M-step, new values for $\Theta$ are chosen to maximize the expected log-likelihood of the data with respect to the distribution over permutations as estimated in the E-step. As an example, the $p(\rho_j)$ terms can always be updated analytically as follows:

$$\hat{p}(\rho_j) = \frac{1}{N} \sum_{i=1}^{N} q\left(\rho_j|\mathbf{x}^i, \Theta\right)$$

where here (unlike in the analysis earlier in the paper) the probabilities of different permutations need not be equal and can be learned from the data.

### 5.2 Learning from Gaussian Data

We simulated data with $\mathrm{S} = \mathbb{R}$, $c = 2$, using the setup in Section 4.1.1. In the first experiment, we performed an empirical analysis of how the permutation error rate of models learned with EM depends on the number of training examples $N$. For this we set $\mu_1 = 1$, $\mu_2 = -1$, $\sigma^2 = 16$ which yields a Bayes-optimal permutation error rate of roughly 0.36 — thus, we know in advance that none of our learned models can have a lower error rate than this. 10 different training data sets were generated for each of the following sizes: $N = \{10, 20, 50, \ldots, 5000, 10000\}$ and for each training data set the best fitting (maximum likelihood) model was chosen from 10 random restarts of EM. Each of these best-fit models were then evaluated on a large independent test data set ($N = 2 \times 10^6$ data points).

The plot in Figure 4 shows that, as expected, the error rate of the predicted model approaches the Bayes-optimal error rate as the number of examples increases. For this particular problem, once the number of data points is on the order of 1000 or greater, EM is performing optimally in terms of identifying the permutations. We also investigated how the error rate of a learned model depends on the error rate of the true model which generated the training set. We fixed the number of examples in each training set ($N = 100$), and for several values of the Bayes-optimal permutation error rate we generated 10 sets of size $N$. Then we learned the best model for each of these sets with 10 random restarts. Each of these models was then evaluated on a large set ($N = 2 \times 10^6$

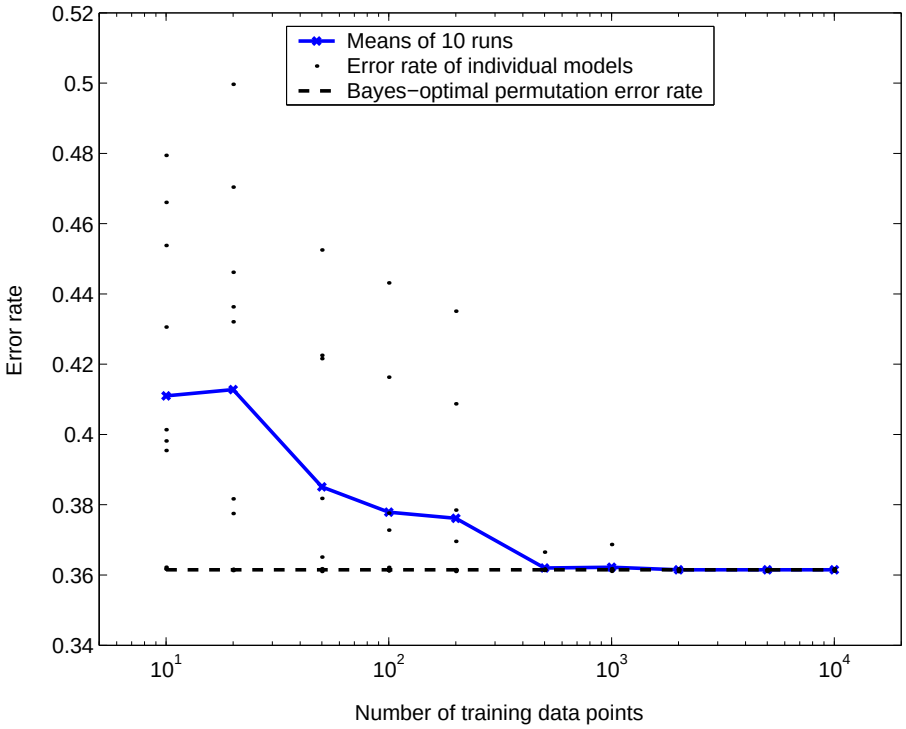


Figure 4: Error rates of models learned from permuted data as a function of the number of training examples.

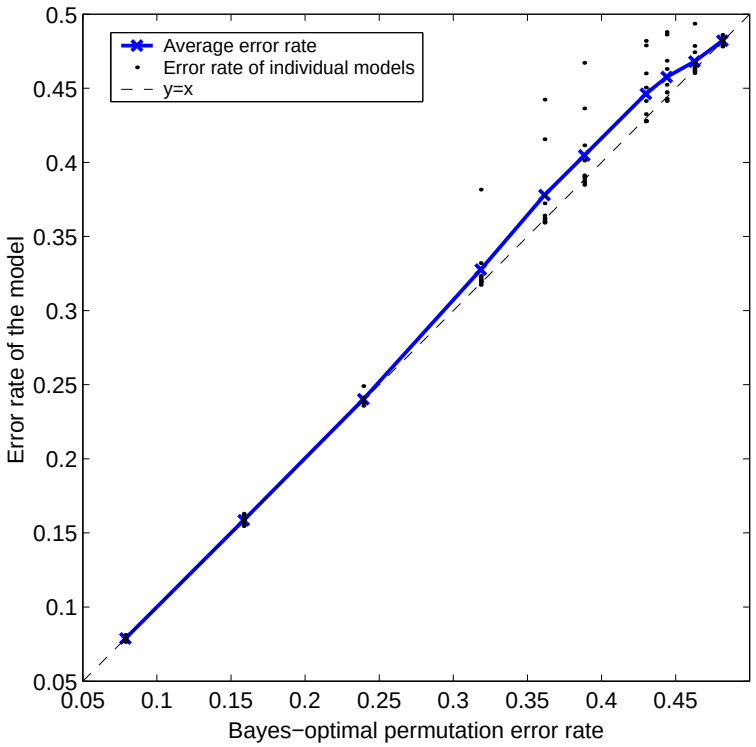


Figure 5: Error rates of models learned from permuted data as a function of Bayes-optimal permutation error rate.

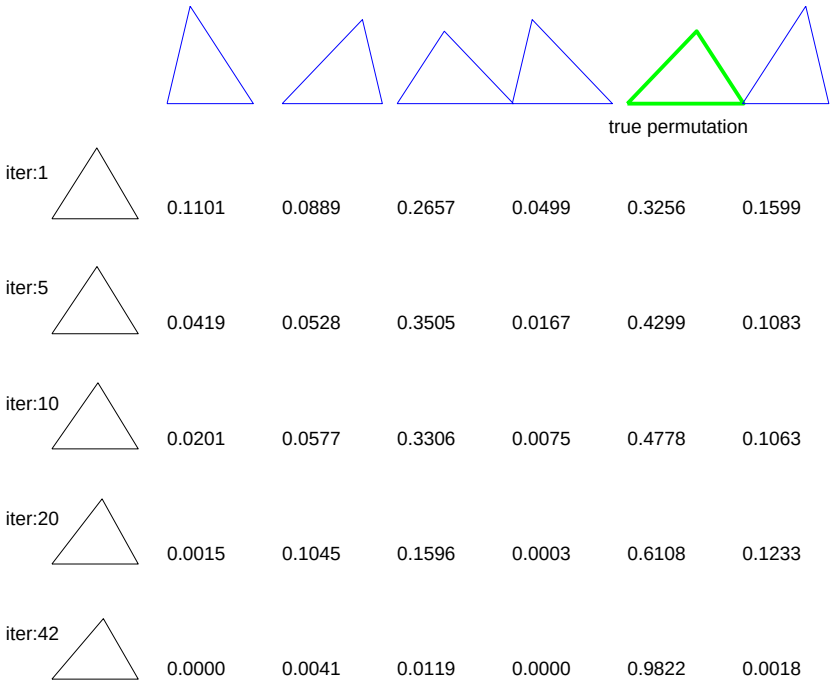


Figure 6: Illustration of EM learning with triangle data.

data points) generated from a true model. Figure 5 shows that as the error rate of the true model increases, so does the increase in error over the true error. The increase begins to diminish once the true error rate gets close to its maximum value of 0.5 since the error rate of any model in this case is bounded by 0.5.

### 5.3 Learning with Rotated Triangles

This problem is a simplified version of the image analysis problem considered in [3], presented here for illustrative purposes. We simulated 100 triangles from a distribution over angles (corresponding to $p(\mathbf{x})$), and then rotated and reflected the triangles in a manner corresponding to a set of random permutations. The learning problem is to learn back the distribution which generated the triangles and put the triangles in geometric correspondence. For this problem, $c = 3$ (3 angles), $\mathrm{S} = \mathbb{R}$, and $\mathcal{P} = \{\rho_1, \ldots, \rho_6\}$ (all six possible permutations of $(1, 2, 3)$). We set $p(\rho)$ as uniform. $p(\mathbf{x})$ is defined as $p(x_1, x_2, x_3) \propto \mathcal{N}\left(x_1|\mu_1, \sigma_1^2\right) \times \mathcal{N}\left(x_2|\mu_2, \sigma_2^2\right)$ if $x_1 + x_2 + x_3 = \pi$, and 0 otherwise. For $\mu_1, \mu_2 > 0$ such that $\mu_1 + \mu_2 < \pi$, and with small $\sigma_1^2$ and $\sigma_2^2$, this distribution generates triangles with angles $x_1, x_2, x_3$.

Figure 6 demonstrates how EM learns both the underlying density model for angle generation and a distribution over rotations and reflections for each triangle. The rows represent different iterations of EM and the leftmost column is the learned density model as represented by the "mean triangle" at each iteration. The columns represent the 6 possible permutations for one of the simulated triangles in the training data, and the numbers in each row are the probability distribution $p(\rho_j|\mathbf{x}), j = 1, \ldots, 6$ for a specific iteration of EM. Starting from a random triangle model (upper left corner) and considerable uncertainty about the likely permutation (row 1), EM gradually learns both the correct "mean shape" and identifies the most likely orientation for this

particular triangle (row 5).

# 6 Conclusions

We analyzed the problem of unsupervised learning in the presence of unknown permutations of feature values. We introduced and analyzed the Bayes-optimal permutation error rate $E_P^\star$. Motivated by the fact that $E_P^\star$ is a lower bound on the error rate of any model that tries to identify permutations, we derived a general bound and closed-form expressions for $E_P^\star$ for specific learning problems and found (for example) that negative and positive correlation among the feature variables can lead to very different learning problems in the presence of permuted data. The paper concluded with a brief empirical illustration of how EM can be used to perform unsupervised learning from permuted data. There are several possible extensions of this work including further analysis of the relationship between $E_C^\star$ and $E_P^\star$ and analysis of learning algorithms for more general transformations than permutations.

# References


[1] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* John Wiley & Sons, New York, second edition, 2000.

[2] Brendan J. Frey and Nebojsa Jojic. Estimating mixture models of images and inferring spatial transformations using the EM algorithm. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1999.

[3] Sergey Kirshner, Igor V. Cadez, Padhraic Smyth, and Chandrika Kamath. Learning to classify galaxy shapes using the EM algorithm. In *Advances in Neural Information Processing Systems 15.* MIT Press, 2003.

[4] Andrew McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000.

[5] D. M. Titterington, A. F. M. Smith, and U. E. Makov. *Statistical Analysis of Finite Mixture Distributions.* John Wiley & Sons, 1985.