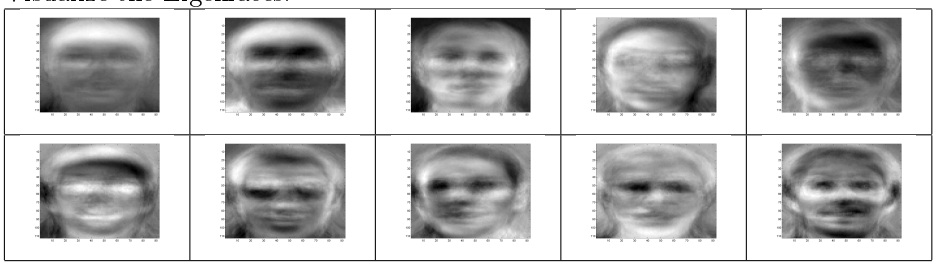
1 Principal Component Analysis: Eigenfaces

1.1 ORL Face Data

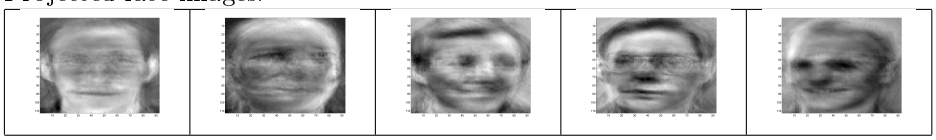
1.2 Calculating Eigenfaces

- Visualize the Eigenfaces:



- Face images 1, 5, 20, 30, 40:

Projected face images:



1.3 Using Eigenfaces to Classify a Face Image

Classification accuracy = 95%

2 Topic Models [Qirong Ho, 25 points]

Mixture models are an active area of Machine Learning research, with many extensions proposed over the years. For instance, in the previous homework we explored how to extend a K-Gaussians mixture model to an infinte Gaussian mixture model. In this question, we are going to explore the "mixtures-of-mixtures" concept, which forms the basis of a topic model.

2.1 K-bag-of-words Mixture Model

Before we talk about topic models, we first need to introduce the K-bag-of-words mixture model, which is used to model words in text documents. The K-bag-of-words model begins with K multinomial distributions (the bags of words). Each represents a probability distribution over words from some vocabulary of length V. Their parameter vectors are $\beta_1, \ldots, \beta_K$, where each β_k is a non-negative V-dimensional vector summing to 1.

Next, we have N documents numbered $1, \ldots, N$, where document i contains M_i words. Note that each document can have a different number of words, and we denote the j-th word of document i by $w_{ij} \in \{1, \ldots, V\}$ (we use integers to represent words). The words are modeled as follows: for each document i, we draw a mixture indicator $t_i \in \{1, \ldots, K\}$ from a prior π. This indicator t_i tells us which multinomial generates the words in document i. Finally, we draw each word w_{ij} from the multinomial parameter β_{t_i}, where the draws are made independently (i.e. we don't care about word order). This gives rise to the following generative process:

$$\begin{aligned} t_i &\sim \text{Multinomial}(\pi) \qquad \text{for } i \in \{1, \ldots N\} \\ w_{ij} &\sim \text{Multinomial}(\beta_{t_i}) \qquad \text{for } i \in \{1, \ldots, N\} \text{ and } j \in \{1, \ldots, M_i\}. \end{aligned}$$

Notice that this is similar to the K-multinomials mixture model, except that some of the observed data (the words w_{ij}) share the same mixture indicator t_i.

In all your answers, please use superscripts to denote vector indices. For example, π^k denotes the k-th element of π, and t_i^k denotes the k-th element of t_i.

1. **[2 points]** Using the definition of the multinomial distribution, explicitly write out $P(t_i \mid \pi)$.

   Answer:

   $$P(t_i \mid \pi) = \prod_k \pi_k^{t_i^k}$$

2. **[3 points]** Use conditional independence to simplify the expression $P(w_{ij} \mid t, \beta, \pi)$ as much as possible. In other words, derive an expression $P(w_{ij} \mid t, \beta, \pi) = P(w_{ij} \mid ...)$ where the '...' is some subset of $\{t, \beta, \pi\}$. The symbols t and β without subscripts represent all $t_1, \ldots, t_N$ and $\beta_1, \ldots, \beta_K$ respectively. **Hint: try drawing the model as a Bayes net (no need to show this). Your final answer should depend on every single $\beta_1, \ldots, \beta_K$.**

   Answer:

   $$P(w_{ij} \mid t, \beta, \pi) = P(w_{ij} \mid t_i, \beta) \tag{1}$$

   Given t_i, β, w_{ij} is conditionally independent of all other variables.

3. **[2 points]** Using the definition of the multinomial distribution, explicitly write out your simplified probability statement from part 2.

   Answer:

$$\begin{aligned} P(w_{ij} \mid t, \beta, \pi) &= P(w_{ij} \mid t_i, \beta) \\ &= \prod_{k=1}^{K} \left(\prod_{v=1}^{V} (\beta_k^v)^{w_{ij}^v} \right)^{t_i^k} \\ &= \prod_{k=1}^{K} \prod_{v=1}^{V} (\beta_k^v)^{w_{ij}^v t_i^k} \end{aligned}$$

2.2 Topic Model with K topics

To develop a topic model[1], we need to make two changes. First, instead of letting document i take on a single topic t_i, let's allow it to have a *mixture* over topics θ_i, where θ_i is a non-negative K-dimensional vector summing to 1. Think of θ_i as a probability distribution over the K topics represented by $\beta_1, \ldots, \beta_K$. The appropriate prior distribution for θ_i is a *Dirichlet distribution*, which will be described shortly.

Second, we now introduce a *topic indicator* z_{ij} for each word w_{ij}, which determines word w_{ij}'s topic. Notice how this differs from the K-bag-of-words model: we're now allowing each word to have its own topic, instead of restricting it to follow the document's topic t_i. Naturally, we shall draw z_{ij} from document i's topic distribution θ_i.

These two changes give rise to the following generative process:

$$\begin{aligned} \theta_i &\sim \text{Dirichlet}(\alpha) && \text{for } i \in \{1, \ldots N\} \\ z_{ij} &\sim \text{Multinomial}(\theta_i) && \text{for } i \in \{1, \ldots, N\} \text{ and } j \in \{1, \ldots, M_i\} \\ w_{ij} &\sim \text{Multinomial}(\beta_{z_{ij}}) && \text{for } i \in \{1, \ldots, N\} \text{ and } j \in \{1, \ldots, M_i\}. \end{aligned}$$

[1] For more information, refer to *Latent Dirichlet Allocation* (Blei et al., 2003).

$\alpha > 0$ is a scalar parameter for the (symmetric) Dirichlet distribution, defined as

$$P(\theta_i \mid \alpha) = \frac{[\Gamma(\alpha)]^K}{\Gamma(K\alpha)} \prod_{k=1}^{K} (\theta_i^k)^{\alpha-1}$$

where $\Gamma(\alpha)$ is the Gamma function[2]. Pay attention to how this model is a "mixture of mixtures": each θ_i represents a mixture over topic vocabularies $\beta_1, \ldots, \beta_K$, and there are N such mixtures $\theta_1, \ldots, \theta_N$, that together constitute the mixture of mixtures.

1. **[2 points]** Use conditional independence to simplify the expression $P(z_{ij} \mid \theta, \alpha, \beta)$ as much as possible. The symbols θ and β without subscripts represent all $\theta_i, \ldots, \theta_N$ and $\beta_1, \ldots, \beta_K$ respectively.

   Answer:

   $$P(z_{ij} \mid \theta, \alpha, \beta) = P(z_{ij} \mid \theta_i) \tag{2}$$

   Given θ_i, z_{ij} is conditionally independent of α, β and $\theta \setminus \{\theta_i\}$.

2. **[2 points]** Using the definition of the multinomial distribution, explicitly write out your simplified probability statement from part 1.

   Answer:

   $$\begin{aligned} P(z_{ij} \mid \theta, \alpha, \beta) &= P(z_{ij} \mid \theta_i) \\ &= \prod_{k=1}^{K} (\theta_i^k)^{z_{ij}^k} \end{aligned}$$

3. **[2 points]** Use conditional independence to simplify the expression $P(w_{ij} \mid z, \theta, \alpha, \beta)$ as much as possible.

   Answer:

   $$P(w_{ij} \mid z, \theta, \alpha, \beta) = P(w_{ij} \mid z_{ij}, \beta) \tag{3}$$

   Given z_{ij}, β, w_{ij} is conditionally indpendent of all other variables.

4. **[2 points]** Using the definition of the multinomial distribution, explicitly write out your simplified probability statement from part 3.

   Answer:

   $$\begin{aligned} P(w_{ij} \mid z, \theta, \alpha, \beta) &= P(w_{ij} \mid z_{ij}, \beta) \\ &= \prod_{k=1}^{K} \left(\prod_{v=1}^{V} (\beta_k^v)^{w_{ij}^v} \right)^{z_{ij}^k} \\ &= \prod_{k=1}^{K} \prod_{v=1}^{V} (\beta_k^v)^{w_{ij}^v z_{ij}^k} \end{aligned}$$

[2] http://en.wikipedia.org/wiki/Gamma_function

2.3 Interpreting Topic Models

The topic model, like the K-bag-of-words mixture model, is a *latent variable* model: some of the variables are unobserved, and we are interested in finding their values. For the K-bag-of-words model, we are interested in finding the hidden document topics t_i. For the topic model, we are mostly interested in the hidden document topic distributions θ_i (and to some extent the word topics z_{ij}).

1. **[2 points]** Both t_i from the K-BoW model and θ_i from the topic model say something about document i's topical content. In one sentence, state the main difference between t_i, θ_i.

   Answer:

   t_i is a single integer between 1 and K, while θ_i is a vector of K components that sum to 1.

2. **[3 points]** Discuss the implications of your answer to part 1. How is topic modeling more useful than K-BoW? Your answer should be no more than a few sentences.

   Answer:

   The observation about the differences implies that t_i represents the single topic that the document is about while θ_i represents the document as a mixture of K topics. Therefore topic modeling allows a document to be a mixture of multiple topics which is arguably a more natural representation of reality than the K-BoW which restricts a document to be only about a single topic. Documents more than a few sentences long are likely to be talking about more than just a single topic, making the assumption of K-BoW unrealistic.

3. **[3 points]** In both K-BoW and topic models, the β_k parameters represent vocabularies for each topic k. We didn't talk about learning the values of β, but it turns out that the common learning strategies (Gibbs sampling or Variational EM) will sometimes produce topics that share words — in other words, $\beta_k^v > 0$ and $\beta_\ell^v > 0$ for some topic k and some topic ℓ. Why is this word sharing useful? Again, keep your answer brief.

   Answer:

   Word sharing is useful because a word could have multiple meanings each of which is used commonly in a different context. For example, the word "web" could be commonly used in technical articles about the internet (topic:internet) or in articles about spiders (topic:spiders).

4. **[2 points]** PCA (Principal Component Analysis) can also be used to learn "topics" from a set of documents. Give at least two differences between PCA and topic models. You don't have to explain the differences, just list them.

   Answer:

   - The contributions of words to a topic can be negative in PCA while they are constrained to be non-negative and < 1 in topic models.
   - The topics obtained from PCA are orthogonal to each other while there is no such restriction on topics in topic models.

3 Adaboost

3.1 True or False

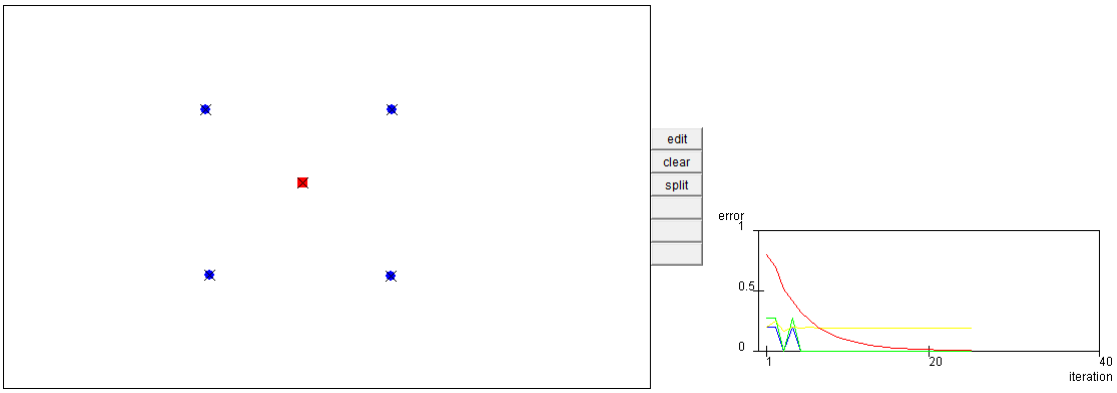
- (a) False. Adaboost guarantees that an upper bound on the training error never increases, but does not guarantee that the training error itself never increases.
- (b) False. If the weak learners h_t do not perform better than random guessing, i.e. $\epsilon_t = 0.5$, then Adaboost will assign them zero weight:

$$\begin{aligned} \alpha_t &= \frac{1}{2} \ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right) \\ &= \frac{1}{2} \ln 1 \\ &= 0. \end{aligned}$$

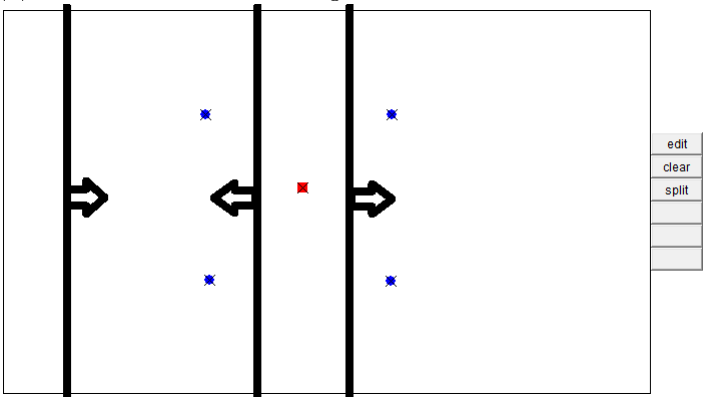
  Thus, the weak learner contributes nothing to the learnt function $f(x)$, so the true error does not decrease.

3.2 Boosting

- (a) Training samples on the left, error graph (after adding additional test samples) on the right:

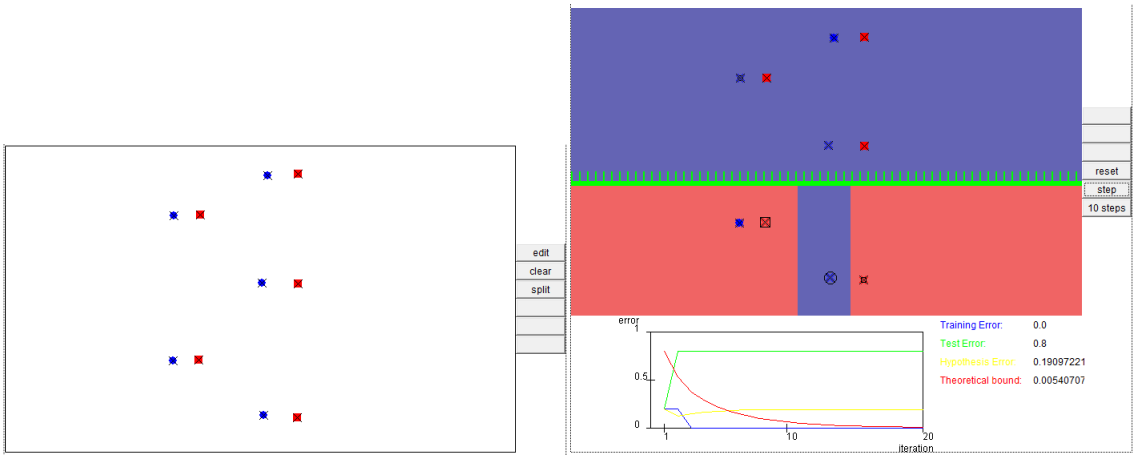


- (b) We can achieve 0 training error in 3 iterations, i.e. 3 weak learners (shown below).



  Proof: Clearly, 1 weak learner cannot achieve 0 training error. Moreover, if we only use the 2 weak learners in the middle, then we cannot get the learnt Adaboost function $f(x)$ to be positive on all 4 blue points; the 3rd weak learner on the left is required for this to happen.

- (c) Adaboost overfits on the following dataset (left). The train/test split and error graph are shown on the right. Test points are solid blue/red shapes, while training points are dithered blue/red shapes. Only 0.2 of the test points are classified correctly.



- (d) Terminate Adaboost early, before the training error reaches zero. This produces a lower-complexity classifier (i.e. fewer weak learners), which has more bias but less variance — and hence is less prone to overfitting. For the dataset in part (c), just one iteration would have given 0.8 test accuracy, but two or more iterations will give only 0.2 test accuracy.