CSC 411
Machine Learning & Data Mining
ASSIGNMENT # 2
Out: 24th October
Due: 12th November

1 Class-Conditional Gaussians (4%)

In this question, you will derive the maximum likelihood estimates for class-conditional Gaussians with independent features (diagonal covariance matrices), i.e. Gaussian Naive Bayes, with shared variances. Start with the following generative model for a discrete class label $y \in (1, 2, ..., K)$ and a real valued vector of d features $\mathbf{x} = (x_1, x_2, ..., x_d)$:

$$p(y = k) = \alpha_k \tag{1}$$

$$p(\mathbf{x}|y = k, \boldsymbol{\mu}, \boldsymbol{\sigma}) = \left(\prod_{i=1}^{D} 2\pi\sigma_i^2 \right)^{-1/2} \exp\left\{ -\sum_{i=1}^{D} \frac{1}{2\sigma_i^2}(x_i - \mu_{ki})^2 \right\} \tag{2}$$

where α_k is the prior on class k, σ_i^2 are the shared variances for each feature (in all classes), and μ_{ki} is the mean of the feature i conditioned on class k. We write $\boldsymbol{\alpha}$ to represent the vector with elements α_k and similarly $\boldsymbol{\sigma}$ is the vector of variances. The matrix of class means is written $\boldsymbol{\mu}$ where the kth row of $\boldsymbol{\mu}$ is the mean for class k.

1. Use Bayes' rule to derive an expression for $p(y = k|\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\sigma})$. [**Hint**: Use the law of total probability to derive an expression for $p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\sigma})$.]

2. Write down an expression for the negative likelihood function (NLL)

$$\ell(\boldsymbol{\theta}; D) = -\log p(y^{(1)}, \mathbf{x}^{(1)}, y^{(2)}, \mathbf{x}^{(2)}, \cdots, y^{(N)}, \mathbf{x}^{(N)}|\boldsymbol{\theta}) \tag{3}$$

   of a particular dataset $D = \{(y^{(1)}, \mathbf{x}^{(1)}), (y^{(2)}, \mathbf{x}^{(2)}), \cdots, (y^{(N)}, \mathbf{x}^{(N)})\}$ with parameters $\boldsymbol{\theta} = \{\boldsymbol{\alpha}, \boldsymbol{\mu}, \boldsymbol{\sigma}\}$. (Assume that the data are iid.)

3. Take partial derivatives of the likelihood with respect to each of the parameters μ_{ki} and with respect to the shared variances σ_i^2.

4. Find the maximum likelihood estimates for $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$.

5*. Extra work for interested students: If you are familiar with Lagrange multipliers, show that the MLE for α_k is indeed given by equation 4:

$$\alpha_k = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}[y^{(i)} = k] \tag{4}$$

2 Handwritten Digit Classification (11%)

For this question you will build classifiers to label images of handwritten digits. Each image is 8 by 8 pixels and is represented as a vector of dimension 64 by listing all the pixel values in raster scan order. The images are grayscale and the pixel values are between 0 and 1. The labels y are $0, 1, 2, \cdots, 9$ corresponding to which character was written in the image. There are 700 training cases and 400 test cases for each digit; they can be found in a2digits.zip.

Starter code written in python is provided to help you load the data. A skeleton is also provided for each question that you should use to format your solution. You are welcome to use any other programming language, as long as your code is functional and modular.

Please note that if you are asked to report/compute quantities these should be clearly displayed in your written report. It is not sufficient to simply print these as an output of your code. The same applies to plots and figures.

0. Load the data and plot the means for each of the digit classes in the training data (include these in your report). Given that each image is a vector of size 64, the mean will be a vector of size 64 which needs to be reshaped as an 8×8 2D array to be rendered as an image. Plot all 10 means side by side using the same scale.

2.1 K-NN Classifier

1. Build a simple K nearest neighbor classifier using Euclidean distance on the raw pixel data.
   (a) For $K = 1$ report the train and test classification accuracy.
   (b) For $K = 15$ report the train and test classification accuracy.
2. For $K > 1$ K-NN might encounter ties that need to be broken in order to make a decision. Choose any (reasonable) method you prefer and explain it briefly in your report.
3. Use 10 fold cross validation to find the optimal K in the 1-15 range. You may use the KFold implementation in sklearn or your existing code from Assignment 1. Report this value of K along with the train classification accuracy, the average accuracy across folds and the test accuracy.

2.2 Conditional Gaussian Classifier Training

Using maximum likelihood, fit a set of 10 class-conditional Gaussians with a separate, full covariance matrix for each class. Remember that the conditional multivariate Gaussian probability density is given by,

$$p(\mathbf{x}|y = k, \boldsymbol{\mu}, \Sigma_k) = (2\pi)^{-d/2}|\Sigma_k|^{-1/2} \exp\left\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_k)^T \Sigma_k^{-1}(\mathbf{x} - \boldsymbol{\mu}_k)\right\} \tag{5}$$

You should take $p(y = k) = \frac{1}{10}$. You will compute parameters μ_{kj} and Σ_k for $k \in (0...9), j \in (1...64)$. You should implement the covariance computation yourself (i.e. without the aid of 'np.cov'). **[Hint:** To ensure numerical stability you may have to add a small positive value to the diagonal of each covariance matrix. For this assignment you can add $0.01I$ to each matrix.]

1. Plot an 8 by 8 image of the log of the diagonal elements of each covariance matrix Σ_k. Plot all ten classes side by side using the same grayscale.
2. Using the parameters you fit on the training set and Bayes rule, compute the average conditional log-likelihood, i.e. $\frac{1}{N}\sum_{i=1}^{N} \log(p(y^{(i)}|\mathbf{x}^{(i)}, \theta))$ on both the train and test set and report it.
3. Select the most likely posterior class for each training and test data point as your prediction, and report your accuracy on the train and test set.

4*. Extra work for interested students: Compute the leading eigenvectors (largest eigenvalue) for each class covariance matrix (can use np.linalg.eig) and plot them side by side as 8 by 8 images.

2.3 Naive Bayes Classifier Training

1. Convert the real-valued features $\mathbf{x}$ into binary features $\mathbf{b}$ using 0.5 as a threshold: $b_j = 1$ if $x_j > 0.5$ otherwise $b_j = 0$.
2. Using these new binary features $\mathbf{b}$ and the class labels, train a Bernoulli Naive Bayes classifier using MAP estimation with prior $Beta(\alpha, \beta)$ with $\alpha = \beta = 2$. In particular, fit the model below on the training set.

$$p(y = k) = \frac{1}{10} \tag{6}$$
$$p(b_j = 1|y = k) = \eta_{kj} \tag{7}$$
$$p(\mathbf{b}|y = k, \eta) = \Pi_{j=1}^{d} (\eta_{kj})^{b_j} (1 - \eta_{kj})^{(1-b_j)} \tag{8}$$
$$P(\eta_{kj}) = Beta(2, 2) \tag{9}$$

You should compute parameters η_{kj} for $k \in (0...9), j \in (1...64)$

Regularization: Instead of the prior, you could add two training cases to your data set for each class, one which has every pixel off and one which has every pixel on. Make sure you understand why this is equivalent to using a prior. You may use either scheme in your own code.

3. Plot each of your $\boldsymbol{\eta}_k$ vectors as an 8 by 8 grayscale image. These should be presented side by side and with the same scale.

4. Given your parameters, sample one new data point for each of the 10 digit classes. Plot these new data points as 8 by 8 grayscale images side by side.

5. Using the parameters you fit on the training set and Bayes rule, compute the average conditional log-likelihood, i.e. $\frac{1}{N}\sum_{i=1}^{N}\log(p(y^{(i)}|\mathbf{x}^{(i)},\theta))$ on both the train and test set and report it.

6. Select the most likely posterior class for each training and test data point, and report your accuracy on the train and test set.

2.4 Model Comparison

Briefly (in a few sentences) summarize the performance of each model. Which performed best? Which performed worst? Did this match your expectations?

Submission

Assignments must be submitted by **10:00 pm on November 12th**; a late penalty of 10% per day will be assessed thereafter (up to 3 days, then submission is blocked). We highly recommend using Python 3.6 embedded in Anaconda 3.4 to solve the programming questions. However, you are welcome to use any programming language you like, given that your code solves the problem, and is functional and modular.

What to submit

- All work to be submitted via Markus.
- All mathematical work has to be clearly marked by the question it belongs to, we highly recommend using a math editor such as MathType, Word Equations or LaTex. However, if you prefer to use paper and pen(cil), you are welcome to do so as long as your hand writing is readable. Unreadable work will result in losing the full mark of the question.
- Your written report should be submitted in PDF format. Note that Markus has filesize limits and it is your responsibility to ensure that your work is uploaded on time.
- Your code must be clearly structured with an obvious entry point.
- All graphs and plots have to be clearly marked by their question. Use readable grids whenever applicable.