Stat 406 Spring 2010: homework 6

1 EM for MAP estimation of Bernoulli mixtures

Consider a Bernoulli mixture model:

$$p(z = k) = \pi_k, \quad p(\mathbf{x}|z = k) = \prod_{j=1}^{d} \text{Beta}(x_j|\mu_{kj}), \tag{1}$$

where μ_{kj} is the probability that feature j turns on in class k. Let us put a uniform prior on π and a product of Beta priors on μ:

$$p(\theta) = \prod_{k=1}^{K} \prod_{j=1}^{d} \text{Beta}(\mu_{kj}|\alpha, \beta) \tag{2}$$

We can use EM to find the MAP estimate of $\boldsymbol{\theta} = (\boldsymbol{\mu}, \boldsymbol{\pi})$.

1. Show that the E step is

$$r_{nk} \stackrel{\text{def}}{=} p(z_n = k|x_n, \theta^t) = \frac{\pi_k \prod_j \mu_{kj}^{x_{nj}} (1 - \mu_{kj})^{1-x_{nj}}}{\sum_{k'} \pi_{k'} \prod_j \mu_{k'j}^{x_{nj}} (1 - \mu_{k'j})^{1-x_{nj}}} \tag{3}$$

   for $n = 1 : N$.

2. Show that the M step is

$$\mu_{kj} = \frac{[\sum_n r_{nk}(x_{nj} - \mu_{kj})] + (1 - \mu_{kj})(\alpha - 1) - \mu_{kj}(\beta - 1)}{\mu_{kj}(1 - \mu_{kj})} \tag{4}$$

   (Note: If $\alpha = \beta = 1$, we recover the MLE.)

2 Shrinkage in linear regression

(Source: Jaakkola)
Consider performing linear regression with an orthonormal design matrix, so $||\mathbf{x}_{:,k}||_2^2 = 1$ for each column (feature) k, and $\mathbf{x}_{:,k}^T \mathbf{x}_{:,j} = 0$, so we can estimate each parameter w_k separately.
Figure 1 plots $\hat{w}_k$ vs $c_k = 2\mathbf{y}^T \mathbf{x}_{:,k}$, the correlation of feature k with the response, for 3 different esimation methods: ordinary least squares (OLS), ridge regression with parameter λ_2, and lasso with parameter λ_1.

1. Unfortunately we forgot to label the plots. Which method does the solid (1), dotted (2) and dashed (3) line correspond to? Hint: see Section 4.4.5 of 19feb edition.

2. What is the value of λ_1?

3. What is the value of λ_2?

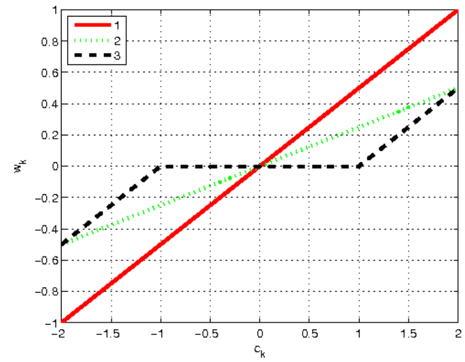

Figure 1: Plot of $\hat{w}_k$ vs amount of correlation c_k for three different estimators.

3 Nonlinear regression for inverse dynamics

In this question, we fit a model which can predict what torques a robot needs to apply in order to make its arm reach a desired point in space. The data was collected from a SARCOS robot arm with 7 degrees of freedom. The input vector $\mathbf{x} \in \mathbb{R}^{21}$ encodes the desired position, velocity and accelaration of the 7 joints. The output vector $\mathbf{y} \in \mathbb{R}^7$ encodes the torques that should be applied to the joints to reach that point. The mapping from $\mathbf{x}$ to $\mathbf{y}$ is highly nonlinear.
We have $N = 48,933$ training points and $N_{test} = 4,449$ testing points. For simplicity, we following standard practice and focus on just predicting a scalar output, namely the torque for the first joint.
Download the data (with name `sarcosData.mat`) from `http://people.cs.ubc.ca/~murphyk/Data/`. Standardize the inputs so they have zero mean and unit variance on the training set, and center the outputs so they have zero mean on the training set. Apply the corresponding transformations to the test data. Below we will describe various models which you should fit to this transformed data. Then make predicitons. and compute the standardized mean squared error on the test set as follows

$$SMSE = \frac{\sum_{i=1}^{N_{test}} (y_i - \hat{y}_i)^2}{\sigma^2} \tag{5}$$

where $\sigma^2 = \sum_{i=1}^{N} (y_i - \overline{y})^2$ is the variance of the output computed on the training set.

1. The first method you should try is standard linear regression. Turn in your numbers and code. (According to [RW06, p24], you should be able to achieve a SMSE of 0.075 using this method.)

2. Now try running K-means clustering (using cross validation to pick K). Then fit an RBF network to the data, using the $\boldsymbol{\mu}_k$ estimated by K-means. Use CV to estimate the RBF bandwidth. What SMSE do you get? Turn in your numbers and code. (Use the latest version (12March) of pmtk for this; see `linregFitTest` for examples of how to fit kernelized lienar regression models using CV.) (According to [RW06, p24], Gaussian process regression can get an SMSE of 0.011, so the goal is to get close to that.)

3. Now try fitting a feedforward neural network. Use `mlpRegressFitNetlab` and `mlpRegressPredictNetlab`. Use CV to pick the number of hidden units and the strength of the ℓ_2 regularizer. What SMSE do you get? Turn in your numbers and code.

References

[RW06] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.