CSE517A – Homework 4

M. Neumann

Apr 9 2020

- Please keep your written answers brief and to the point. Incorrect or rambling statements can hurt your score on a question.
- If your hand writing is not readable, we **cannot give you credit**. We recommend you type your solutions in LaTeX and compile a .pdf for each answer. **Start every problem on a new page!**
- This will be due THU **April 23 2020** at **11:59pm** with an automatic 3-day extension.
- You may work in groups of at most 2 students.
- Submission instructions:
  - Start every problem on a **new page**.
  - Submissions will be exclusively accepted via **Gradescope**. Find instructions on how to get your Gradescope account and submit your work on the course webpage.

Note, that if you use *any* **resources** outside the course materials (which are the lecture notes and course books for this course and its prerequisite materials) to derive (part of) your solution, you will need to cite the source in your homework submission. This also holds for **online sources**. If you collaborate with anyone other than your partner, it is also your responsibility to indicate this in your submission.

This does not mean that citing the source allows you to *copy* existing solutions to any given problem. Please, refer to the **course syllabus** for more details.

Problem 1 (*5 points*) Gaussian Processes (GPs)

For simplicity you may assume zero-mean observations.

Assuming noise-free training data $D = \{(\mathbf{x}_i, f_i)\}_{i=1,\dots,n}$ with $f_i = f(\mathbf{x}_i)$, show that the variance cov_{f_i} for the GP prediction for a *training point* $\mathbf{x}_i$ is 0.

Problem 2 (*15 points*) Parameter Learning for Gaussian Processes (GPs)

For simplicity you may assume zero-mean observations for the entire problem.

(**a**) Despite being a non-parametric model, we still have to learn the kernel parameters $\boldsymbol{\theta}$ for a Gaussian process. Those so called *hyperparameters* can be learned by maximizing the probability of observing the training data given the GP prior. This can be formally expressed by the *marginal likelihood* $p(\mathbf{y} \mid X, \boldsymbol{\theta})$. Luckily for standard GPR, this marginal likelihood can be computed in closed form. Derive the **analytic log marginal likelihood expression** (assuming a GP prior, noisy observations with Gaussian i.i.d. noise $\epsilon \sim \mathcal{N}(0, \sigma_n)$, and a parameterized covariance/kernel function K_θ). By using K_θ we indicate that the kernel matrix K depends on the kernel parameters $\boldsymbol{\theta}$.

HINT: use the fact that $\mathbf{y} \sim \mathcal{N}(0, K_y)$, where $K_y = K_\theta + \sigma^2 I$.

(**b**) When implementing GPs (prediction and learning method), we aim to compute the required inverse matrix K_y^{-1} as efficient as possible. To do so, we leverage the Choleskey decomposition of $K_y = LL^T$, where L is a lower triangular matrix. State the log marginal likelihood in terms of $\boldsymbol{\alpha}$ and L, where $\boldsymbol{\alpha} = L^\top \backslash (L \backslash \mathbf{y})$ and $\backslash$ denotes left-division indicating that we solve a system of linear equations $L\mathbf{b} = \mathbf{y}$ for $\mathbf{b}$. (i.e., $L \backslash \mathbf{y} = L^{-1}\mathbf{y} = \mathbf{b} \Leftrightarrow L\mathbf{b} = \mathbf{y}$).

(**c**) For learning our goal is to pick the hyper-parameters $\boldsymbol{\theta}$ that maximize the log marginal likelihood (or minimize the negative log marginal likelihood). Derive the **derivative of the log marginal likelihood**. Again, state this equation in terms of $\boldsymbol{\alpha}$ and L. This expression is used in a gradient descent procedure in a practical implementation.

Problem 3 (*15 points*) k-Means Clustering

(a) (*5 pts*) Consider the following two possible termination conditions for k-means:

(i) STOP if assignments do not change

(ii) STOP if cluster centers do not change

Are these two conditions equivalent to each other? I.e. (i) $\Leftrightarrow$ (ii)? If your decision is *yes*, prove your answer. If your decision is *no*, provide a counter example and explain your reasoning.

(b) (*5 pts*) Does the k-means algorithm always converge? Argue why or why not.

(c) (*5 pts*) Is it possible that the k-means algorithm generates empty clusters? Argue why or why not.

Problem 4 (*15 points*) Clustering with Kernel k-Means

(a) (*10 pts*) Kernelize the k-means algorithm.

(b) (*5 pts*) Why is it not possible to visualize the cluster centers (in the case of 2D or 3D input data) or use them as prototypes for learning.

Problem 5 (*30 points*) Expectation-Maximization (EM) for Mixture Model Clustering

We are given a set of data points $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ with $\mathbf{x}_i \in \mathbb{R}^d$, which are assumed to be drawn from a probabilistic model consisting of k probability distributions:

$$g(\mathbf{x} \mid \Theta) = \sum_{j=1}^{k} \pi_j p(\mathbf{x} \mid \boldsymbol{\theta}_j)$$

where π_j is the probability of drawing from the j-th distribution (i.e. $\sum \pi_j = 1$), and Θ is our collection of parameters (i.e. $\Theta = \{(\pi_j, \boldsymbol{\theta}_j)\}_{j=1}^{k}$).
For a Gaussian mixture component:

$$p(\mathbf{x} \mid \boldsymbol{\theta}_j) \sim \mathcal{N}_d(\boldsymbol{\mu}_j, \Sigma_j)$$

we have $\boldsymbol{\theta}_j = \{\boldsymbol{\mu}_j, \Sigma_j\}$, where $\boldsymbol{\mu}_j \in \mathbb{R}^d$ and $\Sigma_j \in \mathbb{R}^{d \times d}$ are the mean and covariance of the multivariate normal distribution.

(a) (*5 pts*) Warm-up: what is the probability that some point $\mathbf{x}$ belongs to the j-th distribution for a *Gaussian mixture model*?

(b) (*5 pts*) Our Expectation-Maximization (EM) clustering algorithm needs a criteria for convergence. One method would be to assess the *likelihood of the data* under our previously estimated parameters Θ and under our current parameters Θ'. With some chosen $\epsilon > 0$, if our updated parameters have a likelihood that is ϵ greater than the likelihood of the previous estimated parameters, we would replace the current estimates with our updated parameters and continue with our algorithm. Otherwise, we exit. State the likelihood $\mathcal{L}(X \mid \Theta)$, for the data X given the parameters Θ for a **general mixture model** g.

(c) (*5 pts*) How could you use the GMM likelihood to select the number of clusters k for GMM clustering?

(d) (*15 pts*) Despite it's intuitive interpretation, it turns out that expectation maximization performs MLE. More specifically, it maximizes the following lower bound of the likelihood:

$$\mathcal{B} = \sum_{i=1}^{n} \sum_{j=1}^{k} z_{ij} \log \left(\frac{\pi_j p(\mathbf{x}_i \mid \boldsymbol{\theta}_j)}{z_{ij}} \right),$$

where $\sum_{j=1}^{k} z_{ij} = 1$ and $z_{ij} = [\mathbf{z}_i]_j$.[1]
Now, let's assume a mixture model for d-dimensional **binary input data**, where each mixture

[1] To be able to use this upper bound (also know as *Jensen's inequality*) we had to cast the likelihood as an expectation. That's why the z_{ij}'s appear in the equation.

component is represented as a product of Bernoulli distributions:

$$p(\mathbf{x}_i \mid \boldsymbol{\theta}_j) = \prod_{m=1}^{d} (\theta_{jm})^{x_{im}} (1 - \theta_{jm})^{(1-x_{im})}$$

with $\boldsymbol{\theta}_j$ being the vector of dimension-specific probabilities for the jth mixture component. Start by writing down the log-likelihood for this specific distribution. Now, instead of maximizing this log-likelihood, we maximize the lower bound given as $\mathcal{B}$. Based on this maximization, derive the MLE estimate for the parameters θ_{jm}, which corresponds to the update rule used in the M-step in the EM algorithm.

Problem 6 (*20 points*) Dimensionality Reduction and Feature Transformation

(a) (*10 pts*) Show that the PCA solution is *optimal*. That is, there is no *linear method* that produces a smaller (squared) reconstruction error.

(b) (*5 pts*) What is the difference between PCA feature transformation, feature normalization/standardization, and feature scaling (min-max scaling)?

(c) (*5 pts*) Why is feature scaling and/or feature normalization an important pre-processing step for the following ML methods:

- logistic regression
- ridge regression