CS689: Machine Learning - Fall 2017

Homework 4

Assigned: Wednesday, Nov 15. Due: Wednesday, Nov 29 at 11:55pm

Getting Started: You should complete the assignment using your own installation of Python 2.7. Download the assignment archive from Moodle and unzip the file. The data files for this assignment are in the `data` directory. Starter code is in `blr.py` in the `code` directory.

Deliverables: This assignment has two types of deliverables: a report and code files.

- **Report:** The solution report will give your answers to the homework questions (listed below). The maximum length of the report is 5 pages in 11 point font, including all figures and tables. You can use any software to create your report, but your report must be submitted in PDF format. You will upload the PDF of your report to Gradescope under `HW04-Report` for grading. Access to Gradescope will be enabled one week before the assignment is due.
- **Code:** The second deliverable is your code. Your code must be Python 2.7 (no iPython notebooks, other formats, or code from other versions of Python). You will upload a zip file (not rar, bz2 or other compressed format) containing all of your code to Gradescope under `HW04-Programming`. Access to the autograder will be enabled one week before the assignment is due. When unzipped, your zip file should produce a directory called `code`. This folder should contain a file called `blr.py` that generates all of your results. You may include and call other code files. Make sure you upload all of your code. This assignment will not be auto-graded. Gradescope will only verify that you have correctly submitted a `blr.py` file. Your code will be manually verified during grading.

Academic Honesty Statement: Copying solutions from external sources (books, web pages, etc.) or other students is considered cheating. Sharing your solutions with other students is considered cheating. Posting your code to public repositories like GitHub is also considered cheating. Collaboration indistinguishable from copying is considered cheating. Any detected cheating will result in a grade of 0 on the assignment for all students involved, and potentially a grade of F in the course.

Introduction: Bayesian modeling is often most useful when the amount of data is low relative to the number of parameters, resulting in significant uncertainty about the optimal parameter values. In this question, you will experiment with a basic rejection sampling method for drawing posterior samples for a binary logistic regression model with a spherical multivariate normal prior on the parameters. The data vectors $\mathbf{x} = [x_1, x_2]$ are two-dimensional. The weight vector is thus $\mathbf{w} = [w_1, w_2]$ and the bias is b. The complete parameter vector is thus $\theta = [w_1, w_2, b]$. The prior is zero mean ($\mu_0 = [0, 0, 0]$), with a spherical covariance matrix $\Sigma_0 = 100 \cdot I$. The model is defined below.

$$P(Y=y|\mathbf{X}=x,\theta) = \left(\frac{1}{1+\exp(-(\mathbf{w}\mathbf{x}^\top+b))}\right)^{[y=1]}\left(\frac{\exp(-(\mathbf{w}\mathbf{x}^\top+b))}{1+\exp(-(\mathbf{w}\mathbf{x}^\top+b))}\right)^{[y=0]} \tag{1}$$

$$P(\theta|\mu_0,\Sigma_0) = \mathcal{N}(\theta;\mu_0,\Sigma_0) \tag{2}$$

The rejection sampling algorithm that you will implement is described in MLPP Section 23.3.3. Pseudo code for the sampler is listed below. This algorithm takes as input the number of desired samples S, a training data set $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{1:N}$ and the prior parameters μ_0 and Σ_0, and produces samples from the distribution $P(\theta|\mathcal{D},\mu_0,\Sigma_0)$. This sampler continues running until the desired number of samples S is obtained.

Algorithm 1 Smith and Gelfand Rejection Sampling

```
Inputs: $S, \mathcal{D}, \mu_0, \Sigma_0$
$\theta^{MLE} \leftarrow \max_\theta \mathcal{L}(\mathcal{D},\theta)$
$P_{MLE} = P(\mathcal{D}|\theta_{MLE})$
for ($s$ from 1 to $S$) do
    Accept ← False
    while (Accept=False) do
        $\theta_s \leftarrow$ mvnrnd$(\mu_0, \Sigma_0)$
        $P_s \leftarrow P(\mathcal{D}|\theta_s)$
        $u \leftarrow$ unirnd$(0,1)$
        if ($P_s/P_{MLE} \geq u$) then
            Accept ← True
        end if
    end while
end for
$\Theta^{RS} \leftarrow [\theta_1, ..., \theta_S]$
return $\Theta^{RS}$
```

In this algorithm, θ^{MLE} is the maximum likelihood estimate for the parameters θ. $P_{mle} = P(\mathcal{D}|\theta^{MLE}) = \prod_{n=1}^{N} P(Y=y_n|\mathbf{X}=x_n,\theta^{MLE})$ is the probability of the data under the parameters θ^{MLE}. mvnrnd(μ,Σ) is a function that returns a sample from a multivariate normal distribution with parameters μ, Σ. θ_s is a parameter vector sampled from the prior. P_s is the probability of the data under the parameters θ_s. unirnd(a,b) is a function that draws a sample from the continuous uniform distribution on $[a,b]$. $\Theta^{RS} = [\theta_1,...,\theta_S]$ is the set of samples produced by the algorithm.

Questions:

1. (*30 points*) **Implementation:** In this question, you will implement the rejection sampler and several other methods that will be used to evaluate the sampler.

a. (*10 pts*)As with many machine learning algorithms, a straightforward implementation of Algorithm 1 may be numerically unstable. In particular, when N is large, a naive computation of $P(\mathcal{D}|\theta^{MLE})$ and $P(\mathcal{D}|\theta_s)$ may underflow. Explain why and describe a numerically stable computation for the ratio P_s/P^{MLE} that is required in the rejection step of the algorithm.

b. (*10 pts*) Using your numerically stable implementation of P_s/P^{MLE}, implement Algorithm 1. You should use scikit-learn's logistic regression code to obtain θ^{MLE} (be careful to set the estimator's C parameter properly during learning to obtain the MLE, and use a convergence tolerance of 1e-2). You should also use methods from `numpy.random` or `scipy.stats` to draw the required random samples. As you answer to this question, include the corresponding code in `blr.py` in a method named `rejection_sampler`.

c. (*10 pts*) To help visualize the output of the sampling process, write a function that takes a set of parameters θ and plots the corresponding decision boundary. To accomplish this, your code should convert the implicit function representation of the decision boundary $w_1x_1 + w_2x_2 + b = 0$ to an explicit equation $x_2 = ax_1 + c$ for the line corresponding to the boundary, and then plot this line. As you answer to this question, derive and report the formulas for a and c in the equation $x_2 = ax_1 + c$ given w_1, w_2, b. Include the corresponding code in your code submission in `blr.py` in a method named `plot_boundary`.

2. (*40 points*) **Learning:** The training set $\mathcal{D}^{Tr}$ contains 50 data cases. Let $\mathcal{D}^{Tr}_M$ be a data set consisting of the first M data cases in $\mathcal{D}^{Tr}$. For $M \in \{10, 20, 30, 40, 50\}$, use your sampler implementation to draw a set of 100 samples Θ^{RS}_M from the posterior $P(\theta|\mathcal{D}^{Tr}_M, \mu_0, \Sigma_0)$. Use the five sets of 100 samples $\Theta^{RS}_{10}, ..., \Theta^{RS}_{50}$ to answer the following questions.[1]

a. (*5 pts*) For each $M \in \{10, 20, 30, 40, 50\}$, produce one scatter plot showing the data set $\mathcal{D}^{Tr}_M$ with points colored by class. Overlay the decision boundaries corresponding to the samples in Θ^{RS}_M returned by your sampler.

b. (*5 pts*) For each $M \in \{10, 20, 30, 40, 50\}$, produce a scatter plot showing the data set $\mathcal{D}^{Tr}_M$ with points colored by class. Overlay the decision boundary corresponding to the posterior mean of the parameters estimated as $\theta^{Mean}_M = \frac{1}{S}\sum_{\theta \in \Theta^{RS}_M} \theta$.

c. (*5 pts*) For each $M \in \{10, 20, 30, 40, 50\}$, produce a scatter plot showing the data set $\mathcal{D}^{Tr}_M$ with points colored by class. Overlay the decision boundary corresponding to the posterior mode (the MAP estimate of the parameters) θ^{MAP}_M computed using $\mathcal{D}^{Tr}_M$. The posterior mode θ_{MAP} can be obtained by calling scikit-learn's logistic regression code with an appropriate value of C derived from the prior. As your answer to this question, provide the requested plots along with the formula for C in terms of μ_0 and Σ_0.

d. (*5 pts*) For each $M \in \{10, 20, 30, 40, 50\}$, produce a scatter plot showing the data set $\mathcal{D}^{Tr}_M$ with points colored by class. Overlay the decision boundary corresponding to the MLE θ^{MLE}_M computed using $\mathcal{D}^{Tr}_M$. The MLE θ_{MLE} can be also obtained by calling scikit-learn's logistic regression code with an appropriate value of C (use a convergence tolerance of 1e-2).

e. (*5 pts*) The above plots show estimated and sampled parameters using their decision boundaries in the space of the data. Another useful visualization is to look at the estimated and sampled parameters in parameter space. In this question we will focus on plotting the weight parameters in weight space. For each $M \in \{10, 20, 30, 40, 50\}$, produce one scatter plot showing the sampled values of the weights contained in Θ^{RS}_M. In the same plot, also include points that correspond to the estimated weights in θ^{Mean}_M, θ^{MAP}_M, and

[1] For each of parts (a)-(d), the answer requires a total of five plots, one each for $M \in \{10, 20, 30, 40, 50\}$. The horizontal axis of each of these plots should correspond to the value of x_1, and the vertical axis should correspond to x_2. The x and y axis limits for these plots should be fixed to $[-1, 1]$. For part (e), the answer requires a total of five plots, one each for $M \in \{10, 20, 30, 40, 50\}$. The horizontal axis of each plot should correspond to the value of w_1, and the vertical axis should correspond to w_2. All five plots should have the same x axis limits as well as the same y axis limits.

θ_M^{MLE} computed using $\mathcal{D}_M^{Tr}$. Make sure to color code the points or use different marker styles. Also include a legend.

f. (*5 pts*) Referring to your visualizations, explain in what way θ_M^{MLE} differs from θ_M^{Mean} when M is small. Explain why this is the case.

g. (*5 pts*) Referring to your visualizations, explain what effect increasing the size M of the training data set has on the set of samples produced in Θ_M^{RS}. Explain why this is the case.

h. (*5 pts*) Explain what effect increasing the amount of training data M should have on the relationship between $\theta_M^{MLE}, \theta_M^{MAP}$, and θ_M^{Mean} according to theory. Referring to your visualizations, do you see this effect? Explain why or why not.

3. (*30 points*) **Prediction:** In this question, we will look at the properties of the Bayesian logistic regression posterior predictive distribution relative to making predictions using different plug-in estimators.

a. (*5 pts*) In order to use the S samples in Θ_M^{RS} to compute the posterior predictive distribution for test data, you will implement a simple Monte Carlo approximation as shown below. As you answer to this question, include the corresponding code in `blr.py` in a method named `predictive_distribution`.

$$\begin{aligned} P(Y=y|\mathbf{X}=\mathbf{x}, \mathcal{D}_M^{Tr}, \mu_0, \Sigma_0) &= \int P(Y=y|\mathbf{X}=\mathbf{x}, \theta)P(\theta|\mathcal{D}_M^{Tr}, \mu_0, \Sigma_0)d\theta && (3) \\ &\approx \frac{1}{S}\sum_{\theta\in\Theta_M^{RS}} P(Y=y|\mathbf{X}=\mathbf{x}, \theta) && (4) \end{aligned}$$

b. (*20 pts*) Next, for $M \in \{10, 20, 30, 40, 50\}$, compute the following estimates of test set log likelihood. Provide a single plot showing the test set log likelihood versus the number of number of training cases for all of the methods. The result will be a single plot with four trend lines. Color code the lines and provide a legend.

- Bayesian inference: $L_M^B = \sum_{n=1}^N \log P(Y=y_n|\mathbf{X}=\mathbf{x}_n, \mathcal{D}_M^{Tr}, \mu_0, \Sigma_0)$
- Plug-in Mean: $L_M^{Mean} = \sum_{n=1}^N \log P(Y=y_n|\mathbf{X}=\mathbf{x}_n, \theta_M^{Mean})$
- Plug-in MAP: $L_M^{MAP} = \sum_{n=1}^N \log P(Y=y_n|\mathbf{X}=\mathbf{x}_n, \theta_M^{MAP})$
- Plug-in MLE: $L_M^{MLE} = \sum_{n=1}^N \log P(Y=y_n|\mathbf{X}=\mathbf{x}_n, \theta_M^{MLE})$

c. (*5 pts*) Which method appears to perform best in terms of test set log likelihood? Explain the effect of increasing the amount of training data M on the relative performance of this collection of methods.