CSC 535 – Probabilistic Graphical Models

Assignment Three

Due: 11:59pm (*) Friday, September 21.

(*) There is grace until 8am the next morning, as the instructor will not grade assignment before then. However, once the instructor starts grading assignments, no more assignments will be accepted.

Weight about 7 points

This assignment should be done individually

The purpose of this assignment is to solidify your understanding of conjugate priors, learn/review regression, learn about structuring computational experiments and writing about them, and learn a little bit about model selection in practice, which is an important and non-trivial topic in probabilistic modeling of the world. Key concepts that are featured include conjugate priors, Bayesian updating (for coin fairness), synthetic experiments where the truth is known, prediction error, log likelihood of observed data, AIC, and BIC, held out data error, and held out log-likelihood (not actually featured, but what it is should be clear from context).

Deliverables

Deliverables are specified below in more detail. For the high level perspective, you are to provide a program to output a few numbers and to create figures. You also need to create a PDF document that tells the story of the assignment including output, plots, and images that are displayed when the program runs. Even if the question does not explicitly remind you to put the resulting image into the PDF, if it is flagged with **($)**, you should do so. The instructor should not need to run the program to verify that you attempted the question. See

http://kobus.ca/teaching/grad-assignment-instructions.pdf

for more details about preparing write-ups. While it takes work, it is well worth getting better (and more efficient) at this. A substantive part of each assignment grade is reserved for exposition.

This assignment may be a bit more time consuming than some of the others, especially if you do not have a lot of programming experience. Start soon!

A key concept for this course is drawing samples from a distribution. Having the formula for the distribution tells you how likely a particular sample is, but it does not tell you how to create such samples. If you had a way to sample the distribution, then the histogram of the result should resemble the distributional formula. In the next problem, you will use the method in class to create a sampler for a particular distribution.

1.

a) Implement the method covered in class for sampling a generic distribution. Use this to sample a univariate Gaussian distribution with mean 0 and variance 1. Plot histograms normalized to unit area of 10, 100, 1000, and 10000 samples ($). Plot the source distribution (the univariate Gaussian) on top of the histograms **($)**.

b) Using your code from problem 1 as a starting point, construct a sampler for the Beta distributions with alpha=2 and beta=5. Provide a histogram of 1,000 samples, and plot the source distribution on top of the histogram as you did previously ($). For reference, the Beta distribution is given by: $Beta(u \mid a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} u^{a-1} \left(1-u\right)^{b-1}$, where $\Gamma(\bullet)$ is the gamma function which you can get Matlab to compute using GAMMA().

Note. Matlab implements sampling a univariate Gaussian (normrgnd()), as well as many others (e.g., icdf()), but here you are being asked to make your ***own*** *implementation which is generic and thus is easily extended to other distributions. More critically, implementing a crude version of a sampler will help you understanding of sampling*

2.

a) Consider the Beta distribution prior on coin flips with the two parameters both set to 3. Plot this distribution and comment about the nature of the prior ($). Now compute the posterior after 2, 4, 6, 8, and 10 coin flips, all of which happened to be tails. Plot the posterior distribution on top of the prior, and be sure to provide an informative caption. Such a caption does not simply tell me what the plot is, but tells me how I should interpret the results.

Hint. This is similar to an example in one of the lectures.

b) Remind yourself about the Poisson distribution. Show that a conjugate prior for it is the Gamma distribution ($).

The problems on this page are **optional** preparation for problems that follow. If you are familiar with classic regression, then you probably can skip this page. You should understand the material so that the problems that follow makes sense, but you do not need to write this one up.

Regression by example

Let $y(x)=1+2x+3x^2$
Express y(2) (i.e., x=2) as the dot product of two vectors of length 3, where the first vector is a function of x only.

Once you have accomplished the above, it may help you for upcoming parts to notice that while y(x) is not a linear function of x, if x is fixed, e.g., x=2, then y is a linear function of the coefficients of the polynomial, which will be the unknowns shortly.

Let $y(x)=4+3x+2x^2+1x^4$
Express: $\mathrm{y}_1=y(0),\ \mathrm{y}_2=y(\frac{1}{2}),\ \mathrm{y}_3=y(1),\ \mathrm{y}_4=y(2),\ \mathrm{y}_5=y(3)$
as a 5x4 matrix times a vector of length 4 ($). Your matrix **A** should be a function of the x values. This is like the previous question, but the polynomial is different, and we are doing multiple x values all together.

Now suppose you have observed values in a vector **y** from a model similar to the one in (b) (coefficients might be different) evaluated at the same x values, specifically $(0,\frac{1}{2},1,2,3)$,

Let the 5x4 matrix be **A**, and the vector of unknown coefficients be **w**. Express the sum of the squared error between the estimate and the data using these matrices and vectors. Notice that your answer does not depend on the particular coefficients or data.

As a concrete example, provide the value for sum of squared error for the **A** and **w** from (b) with observed $\mathbf{y}=\left(3,6,12,32,120\right)^T$.

Back to the regularly scheduled problems that need to be written up ☹.

The next four problems are interconnected, and will involve a bit of time to code up and also to write up. I suggest understanding them as a whole first, so you can plan bits of code that can be shared. I also suggest constructing plot output files with names constructed from the parameters and choices that they illustrate. If you are using latex, the file names can be put into your latex source, and then, if you improve the figures (fix bugs, improve titles, beautify colors and fonts, etc.), you can rebuild your writeup without much effort. Finally, remember the adages that the sooner you start coding, the longer it takes.

In a nutshell, we will generate data sets generated from a known function using three different error models. We will then fit polynomials of degree 0 through 7 (1 through 8 coefficients). We will evaluate these by a number of methods: 1) RMS fitting error; 2) RMS error from the truth (oracle); 3) log likelihood; 4) AIC; 5) BIC; 6) RMS error of held out data (using leave-one-out (LOO) cross validation) using the observed data; and 7) RMS error of the held out data using the truth.

Part of the point of this exercise is to learn (more) about organizing computational experiments. If you are organized in your coding and your thinking, it need not be an epic, and there is a lot to learn about model selection, which is a extremely important

3. Now consider the general case of a polynomial with coefficient vector, **w**, i.e., each successive component of w is for an increasing degree of *x*. Consider the case where there is no error in the *x* values, but observed y values are distributed normally around the values predicted by the model (the mean) with some known variance. Assume that the length of **w** is given, but that its values are not known. Show that the MLE for **w** is the value that minimizes the sum of squared error (i.e., exactly what you expressed in the previous problem) ($).

   Hint. You need to use the equation for the Gaussian distribution in this problem. Also, material in the lectures may be helpful.

   *Note: You do **not** need to show that the pseudo inverse, as used in the next problem, is the minimizer, although doing so is something you might want to do for your own enlightenment (or extra credit).*

4. Consider the degree 3 polynomial function $f(x) = x^3 - x$, with *x* in [-1,1]. Generate a data set of 11 data points from this model by generating 11 values of x uniformly spaced out, computing the value of *y=f(x)*, and adding normally distributed noise with sigma=0.2. Set the random number seed to 535 to reduce the variety of results that I get.

   Hint. To get normally distributed noise you could use your sampler from the previous assignment, but now that you have implemented it, you have earned the right to use the Matlab function randn() or the equivalent in your preferred platform. Note that you simply multiply the result by the standard deviation to get the appropriate values.

   I recommend experimenting with the value of sigma, and if you have anything interesting to say about it, please include it in your report. However, I am only expecting results for one value of sigma.

   Call the synthetically generated data "observed". Next your program should find the MLE **w** for lengths 1 to 8 (i.e., with 1 through 8 coefficients, or degree 0 through 7). This can be done using $\mathbf{w} = \mathbf{A}^{\dagger}\mathbf{y}$ where $\mathbf{A}^{\dagger} = \left(\mathbf{A}^{T}\mathbf{A}\right)^{-1}\mathbf{A}^{T}$ (pseudoinverse). Plot the 8 curve approximations on top of the true curve and the observed data **($)**.

Now plot the RMS error (the square root of the average of the squared error) as a function of the length of **w** using the corresponding observed data **($)**. Based on lowest error, what is the best value for the length of **w? ($)** In addition, compute the RMS error of your estimates with respect to "truth" in *f(x)* **($)**. What is the best value for the length of **w** based on this test (which relies on an oracle to compare your points to the "truth"? **($)**. Note that the truth error is estimating the difference of the curve that you estimated and the true curve by evaluating them both at your sample points.

5. Repeat the second part of previous question (error versus number of coefficients), but instead of fitting error, plot (a) the log of the likelihood, (b) the AIC value, and (c) the BIC value for the observed data **($)**. To do this, you will need the variance associated with your model, for which you can compute from the deviations from your fit of the model. Which value for the length of **w** is suggested in each case? **($)**

6. Again consider lengths of **w** 1 through 8, but now hold out each point in turn. More specifically, for each of 11 data points, exclude that point from data, and fit the model on the other 10, and compute the error for (A) the training data, and for (B) the held out point on that model. (This is often called leave-one-out (LOO) cross-validation). Plot the RMS error for (A) and (B) aggregated over the 11 training/test partitions as a function of the length of **w** as before **($)**. What is a good value for the length of **w** based on the average of the RMS errors for the training data, A, and the held out data, B? **($)**

7. Conclusions that you might have drawn from the previous three questions are a function of the true underlying model ($f(x) = x^3 - x$), the error model that we tried, and the data randomly generated from these. It is critical not to rely on conclusions based on only one experiment. In this question we will address the third issue by aggregating the results over many generated data sets, and we will focus on a single question: How promising are the various evaluation approaches for determining the model class, i.e., the number of coefficients.

   Using all the code from the previous problems to an advantage, implement a loop over 101 versions of the generated data (101 is like 100, except it is odd which creates less mystery about the semantics of median values). Each time through the loop, your sampling of the errors will be different, creating new versions the synthetic data set. Do not reset the random seed inside the loop! For each of the seven evaluation methods (*), record the number of polynomial coefficients that gave the best (smallest) error or best (largest) log likelihood. Provide the median of these numbers for each of the methods in a table **($)** and discuss your results **($)**. For the discussion, try to be thoughtful and methodical.

   (*) The seven evaluation methods introduced in the above three problems were: 1) RMS error of the fit curve from the truth (oracle); 2) RMS fitting error; 3) log likelihood; 4) AIC; 5) BIC; 6) RMS error of training data in the leave-one-out (LOO) cross validation experiment; and 7) RMS of the held out (test) data for the .

If have done all the questions until here, then congratulations, you are done! However, if you are looking for alternatives, or extra problems, feel free to keep reading.

8. (**) Put a prior on the coefficients of **w** in the regression equation. Use a simple multivariate normal distribution with **0** mean and diagonal covariance (equal for each coefficient). Hence the precision matrix is simply a parameter, alpha, times the identify matrix. Derive an expression for the posterior distribution. It should depend on both the original precision (or variance) and the precision (or variance) for the prior ($).

9. (**) Derive the decision boundary between two classes with univariate Gaussian posteriors for given means and variances ($). In other words, you are laying out the algorithm for deciding between two models consisting of Gaussians each with their own mean and variance. Go beyond simply setting up the question and solve for the boundary in terms of means and variances ($). These means and variances (four numbers) would be initial input to a program that decided which class subsequent values belonged to (no need to write the program). For concreteness, you can think of distinguishing male/female based on height. Finally, suppose that mistaking the first class for the second is N times more costly than the reverse. Provide a revised expression for the decision boundary ($).

10. (**) Augment the error model of the regression problems to add an outlier. Specifically, one of the points, randomly chosen is set to a value that is significantly different from the model, and one would never get it from the model with normally distributed error. For example, a sensor is compromised by a voltage surge. Experiment enough with outliers to convince yourself that fitting a regression model, which relies on the assumption of Gaussian noise, is quite unstable!

    With outliers, you can get arbitrarily bad fits. So, even if the model degree is reasonable, the prediction error can be terrible. If the numbers of parameters in a model is not too large (a reasonable limit depends on several factors such as the number of data points), a good way to fit models with outliers is RANSAC. Computer vision survivors already know RANSAC. I can supply notes, or a quick surfing session will provide some ideas about how to proceed.

    For this problem explore RANSAC as a way to get good fits, and good model selection, in the presence of outliers. (I have not implemented this myself, but it *should* work!)

11. (**) Minimize the expression developed in problem 3 to derive an expression for the MAP estimate ($).

12. (**) Most familiar distributions are in the exponential family, which means that they have the form:

    $$p(\mathbf{x} \mid \boldsymbol{\eta}) = h(\mathbf{x}) g(\boldsymbol{\eta}) \exp\{\boldsymbol{\eta}^T \mathbf{u}(\mathbf{x})\}$$

    where x may be scalar or vector, discrete or continuous. The parameters η are called the natural parameters of the distribution. $g(\eta)$ is the the normalizing constant.

    a) Show that the multinomial, Gaussian, gamma, and Poisson distributions are part of this family ($)

    b) Suggest a generalized conjugate prior for this family, and show that it has the conjugacy property ($)

What to Hand In

Hand in a program hw3.<suffix> (e.g., hw3.m if you are working in Matlab) and the PDF file hw3.pdf with the story of your efforts into D2L.