Final Exam

SDS 383C: Statistical Modeling I

Due on Sunday 6th Dec by midnight
Please discuss the problems only with your TA or the instructor if needed

Problem 1 (10 pts)

Consider a linear regression with p parameters, fit by least squares to a set of training data $(x_1, y_1), \ldots, (x_N, y_N)$ drawn at random from a population. Let $\hat{\beta}$ be the least squares estimate. Suppose we have some test data $(\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_M, \tilde{y}_M)$ drawn at random from the same population as the training data. If $R_{\text{tr}}(\beta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \beta^{\mathrm{T}} x_i)^2$ and $R_{\text{te}}(\beta) = \frac{1}{M}\sum_{i=1}^{M}(\tilde{y}_i - \beta^{\mathrm{T}}\tilde{x}_i)^2$, prove that

$$E\left[R_{\text{tr}}(\hat{\beta})\right] \leq E\left[R_{\text{te}}(\hat{\beta})\right]$$

where expectations are over all that is random in each expression. Note that this setting is different from the setting we used for proving the optimism of training error in class, namely the in sample error. Here both x and y values are random, which makes the problem much easier.

Problem 2 (10 pts)

(1+1+1+1+1+1+2+2 pts) *The Transformation Trick*: Many variables have natural range restrictions, like being nonnegative, or being forced to be between 0 and 1. Kernel density estimators do not generally obey these restrictions, so they can give positive probability density to impossible values. One way around this is the *transformation method* or *transformation trick*: use an invertible function q to map the limited range of X to the whole real line, find the density of the transformed variable, and then undo the transformation.

In what follows, X is random variable with PDF f, Y is a random variable with PDF g, and $Y = q(X)$ for a known function q. You may assume that q is continuous, differentiable and monotonically increasing, inverse q^{-1} exists, and is also continuous, differentiable and monotonically increasing.

(a) Find $g(y)$ in terms of f and q.

(b) Find $f(x)$ in terms of g and q

(c) Suppose X is confined to the unit interval $[0,1]$ and $q(x) = \log \frac{x}{1-x}$. Find $f(x)$ in terms of g and this particular q

(d) The Beta distribution is confined to $[0,1]$. Draw 1000 random values from the Beta distribution with both shape parameters equal to $\frac{1}{2}$. Call this sample x and plot its histogram. (Hint: ?rbeta.).

(e) Fit a Gaussian kernel density estimate to x using density, npudens, or any other existing one-dimensional density estimator you like.

(f) Find a Gaussian kernel density estimate for logit(x).

(g) Using your previous results, convert the KDE for logit(x) into a density estimate for x.

(h) Make a plot showing (i) the true Beta density, (ii) the ”raw” kernel density estimate from 2e, and (iii) the transformed KDE from 2g. Make sure the plotting region shows all three curves adequately, and that the three curves are visually distinct.

Problem 3 (10 pts)

In class we saw density estimation using histograms. Let $X \in \mathbb{R}$ be a random variable with unknown density f_X supported on [0, 1]. The regular histogram estimator partitions the interval [0, 1] into D bins of equal length,

$$[0,1] = \bigcup_{j=1}^{D} \mathrm{Bin}(j) = \bigcup_{j=1}^{D} \left[\frac{j-1}{D}, \frac{j}{D}\right]$$

Denote $B(x) := \mathrm{Bin}(j)$ for which $x \in \mathrm{Bin}(j)$. The density estimate at x, based on n data points $\{X_i\}_{i=1}^{n}$, can be expressed as

$$\hat{f}_{X,D}(x) = \frac{D}{n} \sum_{i=1}^{n} \mathbf{1}\{X_i \in B(x)\} = \frac{D}{n} |\mathrm{Bin}(j)|,$$

where $|\mathrm{Bin}(j)|$ is the total number of datapoints falling in interval $\left[\frac{j-1}{D}, \frac{j}{D}\right]$. Note that $\hat{f}_{X,D}(x)$ estimates the density of f on interval $\left[\frac{j-1}{D}, \frac{j}{D}\right]$ by the fraction of points that fall in the $\mathrm{Bin}(j)$. The only parameter that has to be chosen in the above described procedure is the number of bins D of the regular histogram. We will be using the leave-one-out cross-validation (loocv), where the loss

$$L_D = \int_0^1 \hat{f}_{X,D}(x)^2 dx - 2\int_0^1 \hat{f}_{X,D}(x) f_X(x) dx$$

is estimated using

$$\hat{L}_D = \int_0^1 \hat{f}_{X,D}(x)^2 dx - \frac{2}{n}\sum_{i=1}^{n} \hat{f}^{(i)}_{X,D}(X_i) \tag{1}$$

where $\hat{f}^{(i)}_{X,D}(X_i)$ is the density estimator at X_i obtained after removing the i^{th} observation. Since the loocv estimator (Eq. (1)) of the loss can be computationally expensive for a large number of samples n, your task is to prove that the following equation:

$$\hat{L}_D = \frac{2D}{n-1} - \frac{D(n+1)}{n-1}\sum_{j=1}^{D}\left(\frac{|\mathrm{Bin}(j)|}{n}\right)^2 \tag{2}$$

gives a closed form solution for the leave one out cross-validation estimator. You *do not* have to prove Eq. (1).

Problem 4 (10 pts)

In class we saw non-parametric bootstrap and subsampling. In this question we will work on some simulations to see cases where the bootstrap fails but subsampling works.

(a) (3 pts) In non-regular problems, the MLE is not asymptotically normal and the scaling constant is usually not $\sqrt{n}$. Prove that if $X_1, \ldots, X_n \sim U[0, \theta]$ then the MLE $\hat{\theta}$ satisfies $n(\theta - \hat{\theta}) \xrightarrow{d} Exp(\theta)$, where $Exp(\theta)$ is a exponential distribution with parameter θ. This is basically telling us that the variance of $\hat{\theta}$ should get close to θ^2/n^2 as $n \to \infty$.

(b) (1 pts) Simulate n datapoints from $U[0,1]$. Now forget that you know $\theta = 1$. Use non-parametric bootstrap to compute the asymptotic variance of $\hat{\theta}$. For $n = 500 : 500 : 10000$ plot the relative error (averaged over 50 random runs) between the bootstrap estimate of variance and $1/n^2$.

(c) (1 pts) Use parametric bootstrap to compute the asymptotic variance of $\hat{\theta}$. For $n = 500 : 500 : 10000$ calculate the relative error between the parametric bootstrap estimate of variance and $1/n^2$. Plot the average over 50 random runs.

(d) (4 pts) Use subsampling to compute the asymptotic variance of $\hat{\theta}$. For $n = 500 : 500 : 10000$ plot the relative error between the subsampling estimate of variance and $1/n^2$. In the same figure, plot different curves for different values of b (averaged over 50 random runs). Be creative about your choice of different values of b. Remember that b has to grow with n but $b/n \to 0$. Don't forget to scale the subsampling estimates of variance properly.

(e) (1 pts) Compare the 3 methods.

Problem 5

(10 pts) In this question we will compare logistic regression with k-nearest neighbors. Please download the dataset from `http://www.cs.cmu.edu/~psarkar/sds383c/X.txt`. The dataset has 2000 points generated from two gaussians in two dimensions. The first 1000 points come from one gaussian and the next 1000 from the other. In order to plot the decision boundary, you may need to use the meshgrid function in R or matlab.

(a) (1 pts) Estimate $\mu_1, \mu_2, \Sigma_1, \Sigma_2$ from the data. Plot the decision boundary of the underlying gaussians by plugging in the estimated values of the four parameters. Your figure should also include the scatter plot of the data.

(b) (1 pts) Train logistic regression (LR) and plot the decision boundary along with the scatter plot of the data on the same figure. Use different markers for the different classes. You can always label a point as class

1 if LR returns a value larger than .5 and 0 otherwise. Feel free to use your old logistic regression code.

(c) (2 pts) Now train k-nn with $k \in \{1, 10, 100, 300\}$ and plot the decision boundary for each k. Train logistic regression (LR) and plot the decision boundary along with the scatter plot of the data on the same figure. Use different markers for the different classes. You can always label a point as class 1 if LR returns a value larger than .5 and 0 otherwise. You can use the builtin functions in R or matlab.

(d) (1 pts) Compare k-nn with LR as we increase k in the last question. Give a brief explanation.

(e) (3 pts) Now find k using 10 fold cross validation. Describe how you will cross validate k and write your own code to do this.

(f) (2 pts) Is it possible to combine the ideas from k-nn and logistic regression to learn the decision boundary better? If yes, briefly describe your algorithm. If not, why not?