CSE 544, Spring 2021, Probability and Statistics for Data Science

Assignment 3: Non-Parametric Inference Due: 3/18, 1:15pm, via Blackboard

(7 questions, 70 points total)

I/We understand and agree to the following:

(a) Academic dishonesty will result in an 'F' grade and referral to the Academic Judiciary.

(b) Late submission, beyond the 'due' date/time, will result in a score of 0 on this assignment.

(write down the name of all collaborating students on the line below)

---

1. MSE in terms of bias **(Total 5 points)**

For some estimator $\hat{\theta}$, show that MSE = bias$^2(\hat{\theta})$ + Var$(\hat{\theta})$. Show your steps clearly.

2. Practice with empirical CDF (eCDF) (Total 5 points)

Using the first 10 samples from the collisions.csv file on the class website, carefully draw the eCDF by hand. Make sure the x- and y-axis clearly indicate the sample points and their corresponding eCDF. Your plot must have y-limits from 0 to 1, and x-limits from smallest sample to the largest sample.

3. Programming fun with $\hat{F}$ (Total 15 points)

For this question, we require some programming; you should only use Python. You may use the scripts provided on the class website as templates. Do not use any libraries or functions to bypass the programming effort. Please submit your code as usual in your zip/tar file repo on BB. Provide sufficient documentation so the code can be evaluated. **Also attach each plot** as a separate sheet (or image) to your submission upload. All plots must be neat, legible (large fonts), with appropriate legends, axis labels, titles, etc.

(a) Write a program to plot $\hat{F}$ (empirical CDF or eCDF) given a list of samples as input. Your plot must have y-limits from 0 to 1, and x-limits from 0 to the largest sample. Show the input points as crosses on the x-axis. (2 points)

(b) Use an integer random number generator with range [1, 99] to draw n=10, 100, and 1000 samples. Feed these as input to (a) to **draw three plots**. What do you observe? (3 points)

(c) Modify (a) above so that it takes as input a collection of list of samples; that is, a 2-D array of sorts where each row is a list of samples (as in (a)). The program should now compute the average $\hat{F}$ across the rows and plot it. That is, for a given x point, first compute the $\hat{F}$ for each row (student), then average them all out across rows, and plot the average $\hat{F}$ for x. Repeat for all input points, x. Show all input points as crosses on the x-axis. (2 points)

(d) Use the same integer random number generator from (b) to draw n=10 samples for m=10, 100, 1000 rows. Feed these as input to (d) to **draw three plots**. What do you observe? (3 points)

(e) Modify the program from (a) to now also add 95% Normal-based CI lines for $\hat{F}$, given a list of samples as input. **Draw a plot** showing $\hat{F}$ and the CI lines for the a3_q3.dat data file (799 samples) on the class website. Use x-limits of 0 to 2, and y-limits of 0 to 1. (2 points)

(f) Modify the program from (e) to also add 95% DKW-based CI lines for $\hat{F}$. **Draw a single plot** showing $\hat{F}$ and both sets of CI lines (Normal and DKW) for the a3_q3.dat data. Which CI is tighter? (3 points)

4. Plug-in estimates **(Total 10 points)**

(a) Show that the plug-in estimator of the variance of X is $\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}(X_i - \bar{X}_n)^2$, where $\bar{X}_n$ is the sample mean, $\bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$. (2 points)

(b) Show that the bias of $\hat{\sigma}^2$ is $-\sigma^2/n$, where σ^2 is the true variance. (3 points)

(c) The kurtosis for a RV X with mean μ and variance σ^2 is defined as $Kurt[X] = E[(X-\mu)^4] \,/\, \sigma^4$. Derive the plug-in estimate of the kurtosis in terms of the sample data. (3 points)

(d) The plug-in estimator idea also extends to two RVs. Consider $\rho = E[XY] - E[X]E[Y]/\sigma_X\sigma_Y$, where σ_X is the standard deviation for RV X. Assuming n i.i.d. observations for X and Y that appear in pairs as $\{(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)\}$, derive the plug-in estimator for ρ. (Hint: What is the ePMF for the event $X=X_1$ AND $Y=Y_1$? What about for the event $X=X_1$ AND $Y=Y_2$?) (2 points)

5. Consistency of eCDF **(Total 10 points)**

Let D={X_1, X_2, …, X_n} be a set of i.i.d. samples with true CDF *F*. Let $\hat{F}$ be the eCDF for D, as defined in class.

(a) Derive E($\hat{F}$) in terms of *F*. Start by writing the expression for $\hat{F}$ at some α. (3 points)

(b) Show that bias($\hat{F}$) = 0. (2 points)

(c) Derive se($\hat{F}$) in terms of *F* and n. (3 points)

(d) Show that $\hat{F}$ is a consistent estimator. (2 points)

6. Properties of estimators **(Total 10 points)**

(a) Find the bias, se, and MSE in terms of θ for $\hat{\theta} = \frac{1}{n}\sum_{i=1}^{n} X_i$, where X_i are i.i.d. $\sim$ Bernoulli(θ). Hint: Follow the same steps as in class, assuming the true distribution is unknown. Only at the end use the fact that the unknown distribution is Bernoulli(θ) to get the final answers in terms of θ. (5 points)

(b) Derive the Normal-based (1-α) CI for $\hat{\theta}$. Explain why Normal-based CIs are applicable here.(5 points)

7. Kernel density estimation (Total 15 points)

This question asks you to implement Kernel density estimator (KDE) from scratch and evaluate it for a sample dataset, a3_q7.csv. Do not use inbuilt KDE functions. But, you can use inbuilt pdf functions to estimate pdf at a point. The formal definition of KDE, which estimates pdf, is:

$$\widehat{f_h}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x-x_i}{h}\right) \quad (1)$$

where K(.) is called the kernel function which should be a smooth, symmetric and a valid density function. Parameter h > 0 is called the smoothing bandwidth that controls the amount of smoothing.

(a) For the a3_q7.csv dataset, the true distribution is Normal(0.5, 0.01) (the mean value μ is 0.5 , the variance σ^2 is 0.01). The task here is to implement a KDE function using the Normal distribution as the kernel, ***normal_kde(x,h,D)*** in python, where x is the point at which the pdf is to be estimated, h is the bandwidth and D is the list of data points. Implement the function as normal_kde.py by first computing $K\left(\frac{x-x_i}{h}\right)$ for all data points x_i in given dataset, where K(u) is the pdf of the standard Normal at point **u** = $\frac{x-x_i}{h}$, and then summing up all K() values and dividing by nh, where n is number of data points, as in Equation (1) above. Submit your code. (3 points)

(b) Obtain the p.d.f. for x = {0, 0.01, 0.02, ...,1} and compute the sample mean and sample variance (use result of Q4(a) as needed) for h=0.0001, 0.0005, 0.001, 0.005, 0.05. Report the deviation (as a percentage difference with respect to true mean or variance) of the estimates from the original distribution (Normal(0.5, 0.01)) in each of the 5 cases. Show on a single plot the pdf of the original Normal and the KDE estimates of the pdf for all 5 bandwidths. Include this plot in your submission. Which of the h values performs best? (6 points)

(c) Repeat (a) and (b) above when using the uniform kernel (implement as ***uniform_kde(x,h,D)*** as uniform_kde.py) with the function **K(u)** described as $\mathrm{K(u)} = \begin{cases} ½ & for -1 \le u \le 1 \\ 0 & otherwise \end{cases}$, where **u** = $\frac{x-x_i}{h}$, and Triangular distribution, ***triangular_kde(x,h,D)*** (implement as triangular_kde.py), using triangle kernel described as **K(u)** = 1-|u| for |u| ≤ 1 (and K(u) = 0 otherwise), where **u** = $\frac{x-x_i}{h}$. Repeat all parts of (b) for these two kernels for all 5 bandwidth values and report the percentage deviation from original mean and variance, plot the KDE estimates, and report the best bandwidth for each kernel choice. (6 points)