CSE 544, Spring 2022, Probability and Statistics for Data Science

Assignment 3: Non-Parametric Inference Due: 3/4, 8:15pm, 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 a3_q2.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. 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)

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

For this question, we require some programming; 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 $\hat{F}$ (eCDF) given a list of samples as input. Use 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 lists 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 $\widehat{F}$ 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_q4.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)

5. 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)

6. Properties of estimators **(Total 5 points)**

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. $\sim$ Poisson(θ). Show your work. Hint: Follow the same steps as in class, assuming the true distribution is unknown. Only at the very end use the fact that the unknown distribution is Poisson(θ) to get the final answers in terms of θ.

7. Functionals of the empirical distribution function **(Total 6 points)**

Let $X_1, X_2, \ldots, X_n$ be i.i.d. RVs with true CDF F. Let $\hat{F}$ be their empirical distribution function. Let a and b be some numbers with a < b. Define $\theta = F(b) - F(a)$. Let $\hat{\theta} = \hat{F}_n(b) - \hat{F}_n(a)$.

(a) Find $\widehat{se}(\hat{\theta})$ (4 points)

(b) Find an approximate (1-α) CI for θ. (2 points)

8. Kernel density estimation (Total 14 points)

This question asks you to implement Kernel density estimator (KDE) and kernel functions from scratch. Do not use inbuilt KDE functions. But you can use inbuilt functions for getting the p.d.f. of known distributions, mean, variance, and integrations. 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. n is the number of samples in the dataset.

(a) Generate a dataset by getting 800 samples (n=800) from a random variable distributed as Normal(0.5, 0.01) (mean of 0.5 and variance of 0.01). *scipy.stats.norm can be used for this.* The task here is to implement a KDE estimator **kde_estimate(x_list,data,h,kernel)** that uses the Normal distribution **normal_kernel(x,data,h)** as the kernel, where **x_list** is a list of points at which the pdf is to be estimated, **h** is the bandwidth, **data** is the list of points in the dataset, **x** is one of the points in x_list and **kernel** is a python function that should be used as K(.). The **normal_kernel** function computes $K\left(\frac{x-x_i}{h}\right)$ for all data points $\mathbf{x_i}$ in the dataset for a given x and h, and returns a list of size n. Where K(u) is the pdf of the standard Normal at point **u** = $\frac{x-x_i}{h}$. Then **kde_estimate** function iterates over all the values of x in x_list and calculates the $\widehat{f_h}(x)$ at every x by summing up all K() values and dividing by nh, where n is number of data points, as in Equation (1) above. kde_estimate returns a list of size equal to that of x_list and this list is considered as the estimated p.d.f. (5 points)

(b) Obtain the estimated p.d.f. for x_list = [0, 0.01, 0.02, ...,1] for different values of h = 0.0001, 0.0005, 0.001, 0.005, 0.05. Then

i. Show on a single plot the pdf of the original Normal (get this from scipy.stats.norm.pdf) and the KDE estimates of the pdf for all 5 bandwidths. Include this plot in your submission.

ii. For each estimate, find the mean and variance using the estimated p.d.f. Report the deviation of the estimates as a percentage difference of mean and variance from the original distribution (Normal(0.5, 0.01)) in each of the 5 cases. Hint: use $E[X] = \int_{-\infty}^{\infty} xf(x)\,dx$ and $Var(X) = E[X^2] - (E[X])^2$ to get the mean and variance of the KDE estimates. Use scipy.integrate.simps to integrate.

iii. Which of the h values performs best? (3 points)

(c) Repeat (a) and (b) above when using the uniform kernel (implement as ***uniform_kernel(x,data,h)***) with the function **K(u)** described as $K(u) = \{½ \; for -1 \leq u \leq 1 \; 0 \; otherwise$, where **u** = $\frac{x-x_i}{h}$, and Triangular distribution, ***triangular_kde(x,data,h)***, 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 new kernels for all 5 bandwidth values and

i. Plot the KDE estimates for different kernels separately.

ii. Report the percentage deviation from the original mean and variance.

iii. Report the best bandwidth for each kernel choice. Report the best kernel and bandwidth combination. (3 points)

(d) Repeat (a), (b) and (c) but with the dataset **a3_q8.csv**. Then

i. Plot the KDE estimates for the three different kernels separately. Instead of plotting the original p.d.f of the dataset, plot the histogram of the dataset. Use matplotlib.pyplot.hist with density parameter to plot the histogram.
ii. Report the percentage deviation from the sample mean and variance. For sample mean and variance, use numpy's mean and standard deviation functions.
iii. Report the best bandwidth for each kernel choice. Report the best kernel and bandwidth combination. (3 points)

Report all the deviations in a single table in your submission PDF with the column names - "dataset", "kernel", "bandwidth", "% deviation mean", "% deviation variance". Add caption to all your plots as "<dataset> <kernel>". Please submit your code as usual in your zip/tar file repo on BB with filename a3_q8.py. Provide sufficient documentation so the code can be evaluated.