STAT 542 / CS 598: Homework 3

Fall 2019, by Ruoqing Zhu (rqzhu)

Due: Monday, Oct 7 by 11:59 PM Pacific Time

Contents

Directions

Students are encouraged to work together on homework. However, sharing, copying, or providing any part of a homework solution or code is an infraction of the University's rules on Academic Integrity. Any violation will be punished as severely as possible. Final submissions must be uploaded to your homework submission portal. No email or hardcopy will be accepted. For late submission policy and grading rubrics, please refer to the course website.

- You are required to submit two files:
  - Your `.rmd` RMarkdown (or Python) file which should be saved as `HW3_yourNetID.Rmd`. For example, `HW3_rqzhu.Rmd`.
  - The result of knitting your RMarkdown file as `HW3_yourNetID.pdf`. For example, `HW3_rqzhu.pdf`. Please note that this must be a `.pdf` file. `.html` format cannot be accepted.
- Your resulting `.pdf` file will be considered as a report, which is the material that will determine the majority of your grade. Be sure to visibly include all `R` code and output that is relevant to answering the exercises.
- If you use the example homework `.Rmd` file (provided here) as a template, be sure to remove the directions section.
- Your `.Rmd` file should be written such that, if it is placed in a folder with any data you are asked to import, it will knit properly without modification.
- Include your Name and NetID in your report.
- **Late policy**: Please check our late policy posted on the course website. However, **you will lose 10% of your score**.

Question 1 [50 Points] A Simulation Study

We will perform a simulation study to compare the performance of several different spline methods. Consider the following settings:

- Training data $n = 30$: Generate x from $[-1, 1]$ uniformly, and then generate $y = \sin(\pi x) + \epsilon$, where ϵ's are iid standard normal
- Consider several different spline methods:
- Write your own code (you cannot use `bs()` or similar functions) to implement a continuous piecewise linear spline fitting. Choose knots at $(-0.5, 0, 0.5)$
- Use existing functions to implement a quadratic spline 2 knots. Choose your own knots.
- Use existing functions to implement a natural cubic spline with 3 knots. Choose your own knots.
- Use existing functions to implement a smoothing spline. Use the built-in ordinary leave-one-out cross-validation to select the best tuning parameter.
- After fitting these models, evaluate their performances by comparing the fitted functions with the true function value on an equispaced grid of 1000 points on $[-1, 1]$. Use the squared distance as the metric.
- Repeat the entire process 200 times. Record and report the mean, median, and standard deviation of the errors for each method. Also, provide an informative boxplot that displays the error distribution

for all models side-by-side.
- Comment on your findings. Which method would you prefer?

Question 2 [50 Points] Multi-dimensional Kernel and Bandwidth Selection

Let's consider a regression problem with multiple dimensions. For this problem, we will use the Combined Cycle Power Plant (CCPP) Data Set available at the UCI machine learning repository. The goal is to predict the net hourly electrical energy output (EP) of the power plant. Four variables are available: Ambient Temperature (AT), Ambient Pressure (AP), Relative Humidity (RH), and Exhaust Vacuum (EV). For more details, please go to the dataset webpage. We will use a kernel method to model the outcome. A multivariate Gaussian kernel function defines the distance between two points:

$$K_{\boldsymbol{\lambda}}(x_i, x_j) = e^{-\frac{1}{2}\sum_{k=1}^{p}((x_{ik}-x_{jk})/\lambda_k)^2}$$

The most crucial element in kernel regression is the bandwidth λ_k. A popular choice is the Silverman formula. The bandwidth for the kth variable is given by

$$\lambda_k = \left(\frac{4}{p+2}\right)^{\frac{1}{p+4}} n^{-\frac{1}{p+4}}\ \widehat{\sigma}_k,$$

where $\widehat{\sigma}_k$ is the estimated standard deviation for variable k, p is the number of variables, and n is the sample size. Based on this kernel function, use the Nadaraya-Watson kernel estimator to fit and predict the data. You should consider the following:

- Randomly select 2/3 of the data as training data, and rest as testing. Make sure you set a random seed. You do not need to repeat this process — just fix it and complete the rest of the questions
- Fit the model on the training samples using the kernel estimator and predict on the testing sample. Calculate the prediction error and compare this to a linear model
- The bandwidth selection may not be optimal in practice. Experiment a few choices and see if you can achieve a better result.
- During all calculations, make sure that you write your code efficiently to improve computational performance