Assignment 04 - HPC and ML 2

Due Date

March 26, 2025 by 11:59 pm.

Parallel Computing (Part 1)

1. (4 points) Rewrite the following functions to make them faster. Add microbenchmarking to compare the run times.

```
# Total row sums
fun1 <- function(mat) {
  n <- nrow(mat)
  ans <- double(n)
  for (i in 1:n) {
    ans[i] <- sum(mat[i, ])
  }
  ans
}

fun1alt <- function(mat) {
  # YOUR CODE HERE
}

# Cumulative sum by row
fun2 <- function(mat) {
  n <- nrow(mat)
  k <- ncol(mat)
  ans <- mat
  for (i in 1:n) {
    for (j in 2:k) {
      ans[i,j] <- mat[i, j] + ans[i, j - 1]
    }
  }
  ans
}

fun2alt <- function(mat) {
  # YOUR CODE HERE
}


# Use the data with this code
set.seed(2315)
dat <- matrix(rnorm(200 * 100), nrow = 200)
```

2.a) (4 points) Write an R function that does Monte Carlo simulation to estimate π and use the function to estimate π with N=1,000, 100,000 and 1,000,000. - Generate N random points inside a unit square. - Count how many points fall within the unit circle. - Approximate π using $\pi \approx 4 \times$ (points inside circle) /

(total points) Explain what you see in the different number of simulations. How long does it take to run each simulation with the different N?

2.b) (4 points) Now write a function to run the Monte Carlo simulation of π for N=100,000 but do it 5,000 times. First use `lapply()` and then parallelize with `mclapply()` or `parLapply()` using 4 cores on your computer. Make sure you set the seed using `clusterSetRNGStream()`. Compare the run times with microbenchmark and make a boxplot comparing the serial and parallel run times. How long do you think it would take to run the simulation 5,000 times when N=1,000,000 in serial and parallel?

Machine Learning (Part 2)

For this part we will use the `hitters` dataset, which consists of data for 332 major league baseball players. The data are here. The main goal is to predict players' salaries (variable `Salary`) based on the features in the data. To do so you will practice many of the concepts in lab 10 (trees, bagging, random forest, boosting and xgboost). Please split the data into training and testing sets and use the same sets for all questions.

3. (3 points) Fit a regression tree to predict `Salary`, and appropriately prune it based on the optimal complexity parameter. Summarize.
4. (3 points) Predict `Salary` using bagging, construct a variable importance plot. Interpret your results.
5. (3 points) Repeat 4. using random forest.
6. (4 points) Perform boosting with 1,000 trees for a range of values of the shrinkage parameter λ. Produce a plot with different shrinkage values on the x-axis and corresponding training set MSE on the y-axis. Construct a variable importance plot.
7. (7 points) Repeat 6. using XGBoost (set up as a grid search on max_depth and eta and do a range of nrounds). Do the training in parallel using `doParallel()`, setting up to `allowParallel=TRUE` in `trainControl()`. Extract the results from the model and plot the RMSE (y-axis) vs the nrounds (x-axis) for each value of eta and max_depth. What are the best hyperparameters?
8. (3 points) Calculate the test MSE for each method and create a table of all results. Which approach has the best performance?
9. (2 points) Write a brief comparison of the variable importance differences for bagging, random forest, boosting, and XGBoost.