DASC 5420 - Exercise 4: CV and Bootstrap Problems

Deadline: 2024-03-25 by 11.59 p.m.

Cross-validation (CV) form scratch!

Algorithm for CV

```
Define sets of model parameter values to evaluate
for each parameter set do
    for each resampling iteration do
        Hold–out specific samples
        [Optional] Pre–process the data
        Fit the model on the remainder
        Predict the hold–out samples
    end
    Calculate the average performance across hold–out predictions
end
Determine the optimal parameter set
Fit the final model to all the training data using the optimal parameter set
```

source: http://topepo.github.io/caret/model-training-and-tuning.html

Question 1:

- Write your own R-code for cross-validation technique (10-fold, LOOCV) in linear regression problem from scratch and use it to a *Auto* data from *ISLR* package in R.
    - Load the *Auto* data and remove the last variable *(name)* to create a new data and use it to do the above task. Use the code to load data:

```
data(Auto, package="ISLR")
```

- In *Auto* data *mpg* is your outcome variable
- Select the significant variables first by fitting the linear regression model and use these predictors in cross-validation
- Evaluate the model performance (compute the CV error)

Sample code: The following code is one way to write a general K-fold cross-validation in R. You can modify the code to fo 10-fold and LOOCV for *Auto* data.

```
#-----------------------------------------------------------------------------
# K-fold CV (customized general function)
#-----------------------------------------------------------------------------
```

```r
KfoldCV <- function(data = Auto, K){
  # data: original data
  # K: number of fold
  # Y: outcome variable
#Randomly shuffle the data
set.seed(5420)
mydata = data[sample(nrow(data)),]

# Create K equally size folds
folds <- cut(seq(1, nrow(mydata)), breaks=K, labels=FALSE)
table(folds)

pred.error <- NULL # to store the prediction error results

# Perform K-fold cross validation
for(i in 1:K){
  #Segement your data by fold using the which() function to hold-out (test sample)
  testIndexe <- which(folds==i, arr.ind=TRUE)
  # Split train-test set
  test.data <- mydata[testIndexe, ]
  train.data <- mydata[-testIndexe, ]

  # Fitting
  #Use the train data to model fitting task: regression, classification and so on..
  # model.fit <- lm(.......)

  # Predict results
  #Use the test data for model prediction
  Y.hat <- predict(model.fit, newdata = test.data)

  # Model performance metrics
  #Prediction error calculation (say, MSE, Misclassification error etc.)

  # MSE <- mean((Y -  Y.hat)^2) # calculate MSE

  # Collecting results
  #Store the prediction error results
  pred.error[i] <- sqrt(MSE) # RMSE
}
return(pred.error)
}
# Implement the code for 10-fold and LOOCV data (note: you need to do some modifications in the code)
# Calculate the prediction error
```

Bootstrap!

Bootstrap procedure:

- Choose a number of bootstrap samples to perform
- For each bootstrap sample
  - Draw a sample with replacement with the chosen size
  - Calculate the statistic on the sample

- Calculate the mean of the calculated sample statistics.

Question 2:

Suppose we have a population which is generated from a Poisson($\lambda = 2.3$) distribution with pdf

$$f(x) = \frac{e^{-\lambda}\lambda^x}{x!}, \quad \lambda > 0.$$

We know that the *MLE* of λ in Poisson distribution is the sample mean $\left(\frac{1}{n}\sum_{i=1}^{n} x_i\right)$ of n observations in the sample.

Do bootstraping (B = 1000) to see the variability of the *MLE* as an estimator of the Poisson *parameter* $\lambda = 2.3$ for $n = 100$ by writing you own function. Calculate the bootstrap bias and bootstrap standard error of the bootstrap estimates of *MLE*. Also calculate the 95% bootstrap percentile confidence interval (you can use *quantile* function).

- Hints: You can follow the Bootstrap example we did in last class!