STA6703 SML Take-Home Final Exam (100 pts + 10 pts bonus)

Directions

Please submit **one PDF** file including all your reports (answer + code + figures + comments; must be easily readable and have file size under a few megabytes) and **one R code script**. The R script is supplementary material to ensure that your code runs correctly. If you are using RMarkdown (not required), please also include your `.Rmd` file.

Place these two (or three) files in a folder, make a zip or rar archive, and submit the archive electronically via Dropbox file request at `tinyurl.com/nbliznyuk-submit-files` (on the landing page, enter your name so that we know it is you and email so that you get a confirmation).

This exam is "open books/notes/class videos/Dropbox course folder", "open R", but "closed internet". Absolutely no collaboration or discussion with anyone (except the course staff). Using or trying to obtain solutions of others is prohibited.

Deadline: 13-Dec-2022 (Tue), 10:00 AM EST. Submissions past this deadline will accrue a penalty of 0.5 pts per hour (12 pts per 24 hours), up to 72 hours delay; no solutions will be accepted after 10:00 AM EST on 16-Dec-2022 (Fri).

Please inform the instructor of suspected typos asap at nbliznyuk@ufl.edu. A publicly available FAQ will be kept in the same folder as the final exam.

Data for the individual problems is provided as dataframes in `SML.2022.final.RData` file.

For problems involving CV and/or parameter calibration, do everything "by hand" (i.e., without the use of `caret` or similar libraries) to earn full credit.

To ensure reproducibility of results across students, use the following code (with `seed=0`) to create your validation and training folds:

```
myCVids <- function(n, K, seed=0) {
  # balanced subsets generation (subset sizes differ by at most 1)
  # n is the number of observations/rows in the training set
  # K is the desired number of folds (e.g., 5 or 10)
  set.seed(seed);
  t = floor(n/K); r = n-t*K;
  id0 = rep((1:K),times=t)
  ids = sample(id0,t*K)
  if (r > 0) {ids = c(ids, sample(K,r))}
  ids
}
# Example
inds.part = myCVids(101, 5); # partition of the indices into K=5 sets
k = 1; isk = (inds.part == k); # k varies from 1 to K
valid.k = which(isk); train.k = which(!isk); # indices of kth valid and train folds
```

Problem 1: **[12 pts]** Conceptual questions (similar to quizzes).

1.1. For a general classification problem, briefly discuss merits and limitations of ROC curve vs confusion matrix.

1.2. Interpret the 45-degree line on the ROC plot. Characterize the classifier for which TPR=FPR=x for $x \in [0, 1]$.

1.3. Suppose one obtained a classifier with $TPR(x) = x^2$, where x=FPR for every $x \in [0, 1]$. Can this classifier be improved upon without acquiring more data? If so, explain how; if not, explain why.

1.4. Suppose a given very large population has 80% controls (0) and 20% cases (1). Consider the following classifier: flip a coin with probability of heads equal to p; if "heads", predict 1, else predict 0. Determine the TPR and FPR for this procedure; you can assume arbitrarily large sample size and sampling with replacement.

1.5. In the previous problem, how is the performance influenced by proportion of cases in the population (call this q)? E.g., q approaches from 0.5 to 0.

1.6 Consider a multiple linear regression model with $p = 40$ covariates. Suppose a dataset was simulated with $n = 50$ (sample size) and it is known that all true coefficients were chosen to be nonzero. If a ridge regression model is fitted to these data, what can be said about the optimal shrinkage parameter λ? (Specifically, is it going to be equal to 0?) What do you expect about the optimal value λ if the sample size n increases (using the same data-generating mechanism)? Briefly explain.

Problem 2: **[16 pts]** Upgrading K-means clustering to handle correlated features.

Revisit the procedure for K-means clustering as presented during lectures (e.g., slide 31/52 from `ch10.unsupervised.from.ISLR.1ed.pdf`) and consider the following setup: suppose that K is known, and the data come from the "normal theory" LDA/QDA model (that we discussed in ch.04 in supervised learning), but an "evil adversary" deleted the true group labels (i.e., the Y_i's). Let's try to recover the labels as follows: assume that the data (i.e., x_i's) from the kth cluster follow multivariate normal distribution $MVN(\mu_k, \Sigma_k)$. The means μ_k will correspond to the cluster centroids, while Σ_k will capture the covariance within cluster k.

Modify the procedure presented in class as follows:

1. In step 2.1, estimate the parameters (μ_k, Σ_k) as in LDA or QDA.
2. In step 2.2, assign x_i's to the most similar cluster in the sense of cluster pdf, i.e., to the cluster $k^* = \arg\max_k \phi(x_i|\mu_k, \Sigma_k)$, where $\phi(x|\mu_k, \Sigma_k)$ is the multivariate normal pdf with parameters (μ_k, Σ_k) evaluated at point x. (It would be easier to use log-pdf here.)

In what follows, for simplicity assume that $K = 2$ (known) and that $\dim(x) = 2$.

2.1. Implement *by hand* (to receive full credit) two versions of the procedure, V1 corresponding to the LDA (with $\Sigma_k = \Sigma$ for all $k = 1, \ldots, K$) and V2 corresponding to the QDA (Σ_k's different for different k). For partial credit, you can rely on the output of the LDA/QDA functions from ch.04.

2.2. **True/False**: if $\Sigma_k = \sigma^2 I$ for every k (i.e., an unknown multiple of the identity matrix), then the procedure V1 will produce exactly the same clustering as the original K-means clustering, assuming the same initialization in step 1. Justify your answer. (You are welcome to implement this special case and compare with K-means clustering if this helps.)

2.3. Multiple (7) simulated datasets are contained in the list `prob2.list`; i.e., the jth dataset can be extracted as `X = prob2.list[[j]]`. Apply your implementations of V1 and V2, present and discuss your

findings; in particular (among other things), discuss the impacts of Σ_k depending on k (i.e., group/cluster-specific covariance matrices vs a single covariance matrix shared across all groups/clusters). You might find it useful to visualize the scatterplots first. Datasets for $j = 1, 2$ were generated according to the assumptions of the subproblem 2.2 ($\Sigma_k = \sigma^2 I$ for $k = 1, 2$); here, you could (should?) compare your results to those produced by the original K-means clustering algorithm as a "sanity check".

Problem 3: [20 pts] Nested K-fold cross-validation.

In class, we used K-fold cross-validation complete the following two goals:

G1. For algorithms without tuning parameters (e.g., discriminant analysis or logistic regression or GAMs without variable selection), estimate the left-out test set predictive performance (RMSE or misclassification rate).

G2. For algorithms with tuning parameters, determine the optimal values of tuning parameters (e.g., calibration).

In real life, one is often interested in understanding the test set performance of an algorithm that requires calibration of tuning parameters. One way in which the two goals may be achieved is by nested CV: an inner loop perform CV to calibrate tuning parameters to achieve (G2), while the outer loop will do CV to estimate the test set performance. Specifically, if the ML method with tuning parameters tau is used (call it "methodX(tau)"), then the goal of the inner loop is to make calibration of these parameters automatic, thereby replacing "methodX(tau)" by "auto_methodX".

The goal of this problem is to carry out this nested 5-fold CV on the example from prelim (Problem 4). Unlike that problem, here we shall jointly achieve goals G1 and G2 based on the training dataset. This is particularly important in data-sparse situations, where one is unable to allocate a sufficiently large test dataset.

For this problem, use the data in `prob3.df`. Here, $n = 200$ and both rounds of the CV will use 5 folds. Let D be the set of row indices for the whole dataset.

1. Use function `myCVids` (with seed 0) to determine the 5 outer CV folds, call these $V_1, ..., V_5$. The ith outer training fold will be $T_i = D \backslash V_i$, where $\backslash$ is the set difference.
2. Apply function `myCVids` (with seed 1000) to T_i to determine 5 inner CV folds, $V_{i1}, ...V_{i5}$. Use CV with these folds (as usual) to calibrate the tuning parameter d of the algorithm.

Report: across $i = 1, ..., 5$, present 5-fold CV estimates of the test error (computed on the data in V_i) as a function of d (the degree of the polynomial); this will only use the outer loop above, and is essentially what you did on prelim 1. Repeat this for each $T_i = \cup_j V_{ij}$ across $j = 1, ..., 5$, i.e., produce 5-fold CV estimates of the test error (on the data in V_{ij}, $j = 1, ..., 5$) as a function of d and overlay these 6 plots on the same figure (i.e., one curve for each i from the inner loop, and one curve across all i from the outer loop).}

Additionally, report the combined CV estimate (over i) based on the best model for each i; you can think of this as the performance of an ensemble where each submodel is "autocalibrated".

Discuss: potential drawbacks of this approach extended to other ML techniques.

Problem 4a: [14 pts] Best subsets selection (exhaustive enumeration) in logistic regression when p is low.

Let $p = 4$ so that we have $2^p = 16$ models. Let γ be the vector of inclusion indicators that determines the model, and α be the decimal representation (an integer) of the model γ (the same integer but in the binary representation). Reuse the code from "2020.09.03.mySubsets.R" under the "code" folder to generate all

subsets in this representation. Specifically, obtain the matrix of all models as `binM = myf(p)` and model ids as `ids = bin2dec(binM)`.

If Y is the response vector and X is the full design matrix with p columns (without the column of ones for the intercept), then model `gamma` may be fitted as `glm(Y ~ X[,gamma==1], family=binomial)`.

The goal here will be to use 5-fold CV to determine the optimal model.

Use the data frame `prob4.df`. Generate the CV folds using the function `myCVids` (with seed=0).

Report: plot of the CV estimates of the test misclassification error rate versus model id, as well as the selected covariates in the best model.

Problem 4b: [6 pts] ROC plot based on 5-fold CV.

For your best model, use 5-fold CV (same folds as in problem 4a) to construct the ROC curve plot. Your vector of predicted values for the ROC curve plot should be produced by training the best model on subset T_i and predicting on V_i. To get the entire vector of predicted values, put the 5 vectors of predicted values into a single vector; then compare vs truth (the whole vector Y) proceeding as with the usual ROC curve plot construction (reuse the code examples from the ISLR book).

Problem 5: [20 pts + 4 pts bonus] K-fold CV for calibration of tuning parameters of ML models for regression.

Use the data from `prob5.df`. Here, $p = 50$ and n will denote the size of the training dataset. For your test data, use rows 401:800 of the data frame.

For your training data, consider 3 different sets/sample sizes: set 1 - rows 1:100; set 2 - rows 1:200; set 3 - rows 1:400 of the data frame.

Use 5-fold CV (using function `myCVids` with seed=0) to calibrate tuning parameters for lasso regression, Random Forest (RF) and GBM. This should be done for each choice of $n_{train} = 100, 200, 400$ above.

For RF, use 100 as the number of trees and use `mtry=c(1:7,p)`.

For GBM, use 1000 trees, $\lambda = 0.01$ and interaction depths 1:7. (Report results for 1000 trees but feel free to tune this.)

For each method, report 3 CV RMSE curves (one curve for each n=100,200 and 400), overlaid on the same plot. Mark the optimal value of the CV RMSE and the corresponding value of the tuning parameter.

Additionally, report the 3-by-6 matrix with rows corresponding to the three ML methods and columns corresponding to the CV RMSE and test RMSE for each value of n for model with parameters chosen by CV. (I.e., two RMSE values for $n = 100$, then the two RMSE values for $n = 200$, etc.) The CV RMSE will be computed from the training data only. The test RMSE should be computed by fitting the model on the entire training dataset with parameters determined by the CV; then this model is used to predict on the test set.

Briefly discuss your findings; particularly, as n increases.

For RF and GBM, report and discuss variable importance (for the best models).

Discuss what you expect: (i) as training sample size increases but the test sample is held fixed; (ii) training set is held fixed but the test sample size increases.

Bonus: Suppose another 50 predictors are added (uncorrelated with the response) so that we now have 100 predictors. What do you expect to happen to the predictive performance of the methods? Explore by simulating extra columns, generated as iid Uniform(-1,1).

Problem 6: [**18 pts**] K-fold CV for calibration of tuning parameters of ML models for classification.

Use the data from `prob6.df`. Here, $p = 20$ and $n = 200$ for both training and test datasets. For your training data, use rows 1:200 of the data frame. For your test data, use rows 201:400 of the data frame.

Use 5-fold CV (using function `myCVids` with seed=0) to calibrate tuning parameters for the RF, GBM and the SVM (with radial kernel) classifiers.

The tuning parameters for RF and GBM are the same as in problem 5.

For each method, report CV misclassification error rate (MER) curves; this will be computed without using test data and used to select optimal tuning parameter values.

Additionally, report the test MER for each classifier with the parameters chosen by CV (the entire training dataset to train the model with this set of tuning parameters, then predictions are made for the test data to determine MER).

Lastly, overlay on the same plot ROC curves for the three classifiers (with optimal tuning parameters). Here, the ROC curves will be constructed using test data.

Briefly discuss.