STA 4364 Final Exam

due Monday December 14 by 11:59PM on Webcourses

Submission Format: Please submit your midterm in either Jupyter Notebook or R Markdown format. You can submit at most one file for each problem. If you want, you can include multiple problems in the same file.

Dataset: All problems in this exam will use the Bank Marketing dataset in the file `bank_data.csv`. Descriptions of the features are available here. The variable of interest is the `y` feature, a binary outcome that indicates if a customer has made a deposit. The original dataset has been edited so that there are an equal number of positive and negative outcomes. All string-valued features are categorical and all number-valued features are numerical. Although the dataset description says that the `duration` feature might not be appropriate for true predictions, you will use it as a feature for this problem. Split your dataset into a training set with 80% of observations and a validation set with 20% of observations.

Problem 1: (30 points) This problem will examine logistic regression.

(a) Learn a logistic regression model using the full set of variables. From your full set of variables, select the most relevant features and create a logistic regression model using the reduced set of features. Plot an ROC curve and report the AUC for the full and reduced model on both the training and validation sets (4 curves in all). Comment on the degree of overfitting that you observe. Compare the performance of the full and reduced model on the validation set.

(b) Learn a LASSO logistic regression model (the R command `model.matrix()` might be useful for formatting your dataframe to use with `glmnet`). Tune the value of λ using 10-fold cross-validation. Visualize the cross-validation error across different values of lambda, and report the value of λ that minimizes cross-validation error. Report the features that your LASSO model selects at the optimal value of λ, and compare these features to the features you selected in part a). Make an ROC curve and calculate the AUC for the training and validation data for your LASSO model.

Problem 2: (40 points) This problem will examined bootstrap intervals using gradient-boosted decision trees.

(a) Learn a gradient-boosted decision tree to predict `y`. Tune the number of trees, the tree depth, and the learning rate either using cross-validation or using performance on the validation set. Report the accuracy of your model on the training and validation data, and plot an ROC curve and report the AUC for predictions on the training and validation data.

(b) The bootstrap can be used to create confidence intervals for many quantities. In this problem, you will create bootstrap intervals for your predicted probabilities (the probabilities used to make your ROC curve) from part a). Recall the steps of bootstrapping:

- Create a bootstrap training set by sampling the rows/observations of the original training set with replacement. The `sklearn` function `sklearn.utils.resample` might be useful for this.
- Train a model using the bootstrap training set, and calculate your quantity of interest.
- Repeat the first two steps for many bootstrap samples (use a `for` loop). The bootstrap standard deviation estimate is then the sample standard deviation of the quantities of interest across the bootstrap samples.

Calculate 100 bootstrap samples for each predicted probability in the validation set (this can be stored in a matrix, where each row is a validation observation and each column is a single bootstrap outcome). Calculate the standard deviations across your bootstrap samples (one standard deviation for each validation observation). Make a 95% confidence interval for the predicted probability of each validation observation, centered at the predicted probabilities from part a) and using the bootstrap standard deviation. Report the accuracy of predictions from part a) over only observations whose intervals contain 0.5 (uncertain predictions). Report the accuracy of predictions from part a) over only observations whose intervals do not contain 0.5 (more certain predictions).

Problem 3: (30 points) In this problem, you will use a fully-connected neural network to model the data. Remember to 1-hot encode all categorical variables to make your feature matrix fully numerical.

(a) Set up a neural network with three layers:

- A fully connected layer with one input node for every column of your data matrix, and 200 output nodes.
- A fully connected layer with 200 input nodes and 75 output nodes.
- A fully connected layer with 75 input nodes and 2 output nodes, one for each class.

Use batch norm followed by ReLU activation after each layer except for the last.

(b) Train your neural network for 1000 epochs (sweeps through the dataset). Use a batch size of 32 and the Adam optimizer with learning rate 0.0001. Plot the training and validation loss and the training and validation accuracy for each epoch (you might want to use the `plt.show()` command rather than `plt.savefig()` command to put graphs directly in your document) Identify the approximate epochs where overfitting begins to occur. Re-train your model by early stopping at the threshold where overfitting begins (use a reduced number of epochs). Create an ROC curve and report the AUC on the training and testing data for your early-stopping neural network. Comment on the degree of overfitting.