Homework 2: Regularization, Trees and Boosting **CSCE 633-600**
Due: 11:59pm on October 26, 2023

Instructions for homework submission
a) Please write your code in the provided starter Jupyter Notebook.
b) Your submission should include a Jupyter Notebook that can be run seamlessly and performs all the required steps one after another. Any submission with a runtime error would result in lost points.
c) Make sure to comment your code and complete the required cells in the notebook.
d) Please start early :)
e) Total: 100 points

Part A - Regularization (30 points)

Use *Hitters Dataset* provided in *Homework 1*

1. Read Hitters Dataset into a pandas dataframe.
2. Preprocess the data. (Hint: remember all the preprocessing steps we used in Problem B(i) of Homework 1)
3. Split the data into 3 sets - train, validation, and test sets. You should have the following six splits: X_train, X_val, X_test and y_train, y_val, y_test.
4. Fill in the function *train_ridge* to implement a *Ridge Model* for Hitters dataset that returns a dictionary with *alpha values* as keys and corresponding *mean aucs* as value pair. Repeat the training process for n times and train the model each time for *max_iter* iterations with all *alpha_vals*. Compute the auc values with the validation set. Please describe your hyperparameter tuning procedures and optimal *alpha* in *alpha_val* that gives the best model.
5. Fill in the function *train_lasso* to use a *Lasso Model* for Hitters dataset that returns a dictionary with *alpha values* as keys and corresponding *mean aucs* as value pair. Repeat the training process for n times and train each time for *max_iter* iterations with all *alpha_vals*. Please describe your hyperparameter tuning procedures and optimal *alpha* in *alpha_val* that gives the best model.
6. Fill in the function *ridge_coefficients* that returns a tuple of trained ridge model with *max_iter* iterations and optimal *alpha* as well as the model's coefficients.
7. Fill in the function *lasso_coefficients* that returns a tuple of trained lasso model with *max_iter* iterations and optimal *alpha* as well as the model's coefficients.
8. Compare the coefficients from Lasso and Ridge models.
9. Fill in the function *ridge_area_under_curve* that returns area under curve measurements. Plot the ROC curve of the Ridge Model. Include axes labels, legend, and title in the Plot.
10. Fill in the function *lasso_area_under_curve* that returns area under curve measurements. Plot the ROC curve of the Lasso Model. Include axes labels, legend and title in the Plot.

Part B - Regression and Classification Trees (50 points)

In this problem, you will be coding up regression and classification trees from scratch. Trees are a special class of graphs with only directed edges sans any cycles. They fall under the category of directed acyclic graphs or DAGs. So, trees are DAGs where each child node has only one parent node. Since trees are easy to design recursively, it is super important that you are familiar with recursion. So, it is highly recommended that you brush up on recursion and tree-based search algorithms such as depth-first search (BFS) and breadth-first search (BFS). Your submission should include a script that can be run seamlessly and performs all the following steps one after another. Any submission with a runtime error would result in lost points.

1. Growing a maximum-depth regression tree.
2. Growing a two-class classification tree.

Part C - Boosting (20 points)

Now that we implemented regression and classification trees in part B, we would like to use a decision-tree-based ensemble Machine Learning algorithm for the dataset we used in part A. You can use the preprocessed *Hitters Dataset* you created in part A.

1. Fill in the function *train_XGB* to use a *XGBoost Model* with L2 regularization for Hitters dataset that returns a dictionary with *alpha_vals* as keys and corresponding *mean auc* as value pair. Repeat the training process for n times and train the model each time for *max_iter* iterations with all *alpha_vals*. Compute the auc values with the validation set. Please describe your hyperparameter tuning procedures and optimal *alpha* in *alpha_val* that gives the best model.
2. Train and tesst the model with the best parameters you found.
3. Plot the ROC curve for the XGBoost model and also print the area under curve measurements. Include axes labels, legend, and title in the Plot.
4. Compare the results of the XGBoost model with the Ridge and the Lasso models and report your findings.