Assignment 3 writeup

Daniel Roberts

Please note that this is a follow up on the previous assignments write-up. As such, there are no new histograms or statistics table outputs are included. However, the code included with this write-up has been edited to fix issues with histogram binning (both width and only one bin issues).

In addition, as a full disclosure, note that significant collaboration Michael Fosco. Since we are working on the final project together, we wanted the structure and output of our pipelines to coincide so that we can use them in tandem for the final project. In addition, his comments and structure were helpful to me in creating my own code. Thus, structurally, my code is similar to his.

In order to complete this project, I created a pipeline that would loop across possible parameter permutations for a number of Machine Learning models. In addition, to test the models, I implemented a k-fold cross validation (5 in this case in particular). This was in order to make the data robust to particular test-train splits when calculating best fit.

In solving this problem I looked at a number of useful metrics. I looked at precision across a set of possible thresholds, recall across a set of possible thersholds, Area under the Curve (AUC), Accuracy, f1, Training Time, and Testing Time. I compared these across a set of possible models. The models I looked at were Logistic Regression, K Nearest Neighborhd, Random Forrest, Extra Trees, Ada Boost Classifier, Support Vector Machines, Gaussian NB, Decision Trees, and SGD. When choosing to include only one “best” models from each type, I saved the results for the highest AUC and highest Accuracy across all possible parameter choices. Then, I produced the table in final\_output.csv that compares the performance of the different models.

First, let us consider some characteristics of the optimal parameter choices within models for this particular problem. The Logistic Regression Estimator worked best with standard regularization (C=1), and no class weighting rather than balanced weighting. The K Nearest Neighbors algorithm performed essentially identically for number of neighbors, and weighting metric, Since both leaf size and number reached their upper bound, one would like to explore how far improved performance goes with expanding these figures holding lll, The simple Decision Tree classifier worked best with an entropy (information gain) criterion. As before, trees (including random forrests and extra trees) perform well with high “max depth”, and with larger minimum splits to prevent overfitting.

Across all models, Random Forest Classifiers performed the best in terms of AUC and Accuracy. The differences in accuracy though, were unfortunately not substantively better than the other models: they remained near the 93% rate that reflected the overall negative rate in the population. AUC however, was at an impressive level in the higher “.8” region. In addition, the Extra trees classifier (another form of random tree) and the Ada Boost classifier (boosting) performed nearly as well in AUC terms. However, the Random Forrest Classifier took by far the longest time to train (20-40 seconds for the AUC and accuracy maximizing). Thus, extra trees and boosting performed arguably the best within reasonable time to trai constraints. Simple Decision trees too, only had a slight AUC drop off compared to Random Forrests while having a very fast testing and training time comparatively, faster than Extra Trees or boosting. The reason all of these classifier's performed admirably is evident in the reasonable (and similar) precision-recall tradeoffs at lower threshold levels (of around .1-.25).

One interesting thing to note is that the SVC model did well in getting relatively high recall at thresholds like .2-.25. However, this can be attributed to the very low to zero precision rate at these same values (hence why it did not perform well on AUC metrics). Another interesting note is that KNN is the only model where testing time took longer than training time (and also shorter training time). Other notes time wise is that logistic regression models were very fast to test, as were decision trees. Decision trees had low relative training and testing time per model. Overall, I would recommend Decision Trees to someone trying to solve this problem. They are very fast and get good results, which are only marginally improved upon by the much more computationally intensive ensemble methods that are derivative from decision trees like extra trees or random forrests. A results table for this assignment can be found in the github repository, under the heading final\_output.csv, again with best AUC and best Accuracy results in the parameter space for each model. There are many entries (rather than the 2x Models figure one would expect for best accurac and best AUC) because a number of models had ties for best AUC or best Accuracy across their parameter space). The first columns (headed with .05-.85 alone in the header) are precision, and the rest are labeled well.