Machine Learning Engineer Technical

Overview

One of the core tenets for OneThree Biotech is to be constantly innovating in both the fields of biology and machine learning. As we deal with these real-life problems we must find creative solutions that balance improving model performance, conserving model interpretability and accurately modeling biological mechanisms. Class imbalance is an ever-present issue in biological data, from predicting drug toxicity to novel drug targets. We are constantly benchmarking different techniques to account for class imbalance, many of which are methods published in scientific papers. Unfortunately, many of these methods do not include a coded implementation, leaving this job to our scientists and engineers.

In this task, you will implement the Biased Random Forest (BRAF) as outlined in the paper, "Biased Random Forest for Dealing with the Class Imbalance Problem". This paper describes a technique for combating class imbalance at the algorithm-level rather than the data-level. You will train this model with the publicly available Pima Diabetes dataset.

Coding Assignment

1. **Preprocess**
   You will use the diabetes.csv from the Pima Dataset to train/test your model. Please be sure to split this dataset into a test and training set. Certain values within this dataset are missing, please identify these values. You are responsible to imputing the unavailable values so that all data is complete when training the model. Feel free to use any technique you wish. Consider any other "preprocessing" steps you can think of that might improve model performance/interpretability.

2. **Training your BRAF**
   Once you have finished all data preprocessing, you will train the model with your training set. Use the following default parameters.

   Set K= 10, p = 0.5, s=100.

3. **Functional Requirements**
   Your program should do the following:

- Train your model on the Pima Dataset with user-specified hyperparameter flags
- Print out the following metrics: precision, recall, AUPRC and AUROC for 10-fold cross-validation on the training set and the test set.
- Create ROC and PRC curves that save as separate files for both cross-validation and test set results.

4. **Evaluation**
   You will be evaluated on the correctness of your implementation and the quality of your code. The code should be clean, easy to read, and well commented. Ideally, someone who read the linked paper could easily understand what your code is doing. Your code should be modular, with proper separation of concerns.

5. **Ground Rules**
   All code should be written in python. You should not use any machine learning APIs such as Scikit-Learn, TensorFlow, etc. However, feel free to look at any resource (blog posts, stack overflow, etc.). Make sure to cite any resource you use.

Contact Information

Please do not hesitate reaching out to Cory (coryandar@onethree.bio) if you have any questions or concerns.