Lab 3: Classification Challenge

This assignment is due to Gradescope by the beginning of lab next week (3:00p on 2/26). You may work with others in your breakout room, but you must submit your own solution.

Introduction

The purpose of this lab is for you to practice implementing shallow ML pipelines in a common format: the classification challenge.

Organizations and researchers have long posted datasets as public "competitions" to see how well machine learning models can perform on associated classification and regression tasks. People participate in these challenges for a variety of reasons. In some sub-fields of machine learning, beating the state-of-the-art performance on a widely-used dataset can be worth an academic publication alone. Others use these challenges ways to practice ML programming, contribute to open-science initiatives, or win prizes (e.g., this ongoing competition that is offering $500,000 in prizes for accurate 10-day forecast predictions of stream flow volumes in the western U.S.).

This lab will let you experience an ML competition in a low-stakes environment. You will be provided with a dataset with the test set already removed. Your task will be to train a machine learning model using any of the shallow learning techniques we have covered in class. At any point before the lab is due, you may upload your current model to Gradescope, where it will be automatically applied to the test set. Its performance will then be posted on an *anonymized* leaderboard, so you can see how your models stack up against the rest of the class.

You are encouraged to upload your model to the leaderboard early and often. While you won't have access to the test set, submitting allows you to check whether changes to your model have improved or reduced test set performance. In real ML competitions, it is common for teams to submit a "naive" model at the beginning of the competition to set themselves a baseline for improvement.

At the end of the day, you will only be graded on 1) whether your model beats a simple depth 3 decision tree classifier programmed by Prof. Apthorpe, 2) whether your code demonstrates meaningful effort to improve model performance through data preprocessing, model selection, and hyperparameter optimization, and 3) your answers to open-ended questions.

However, there will be **extra credit** awarded based on the leaderboard! If your model is in place p for F_1 score, you will receive $\frac{10-p}{2}$ extra credit points. These points will be awarded once at the end of lab today and again when the lab is due next Friday. For example, if your model is in second place at the end of lab and fourth place next Friday, you will receive $\frac{10-2}{2} + \frac{10-4}{2} = 7$ points of extra credit.

Provided Files

- `Lab3.pdf`: This file
- `Lab3_pipeline.py`: Code scaffold
- `Lab3_train.csv`: Labeled training data
- `Lab3_questions.txt`: Open-ended questions

ML Task Instructions

The `Lab3_train.csv` file contains 10 years worth of daily weather observations from locations across Australia with 20% of the observations already removed for the leaderboard test set. Your goal will be to implement a ML pipeline that, when given a new weather observation, can predict whether it will rain on the day after the observation. The correct labels are encoded in the `RainTomorrow` column as 1 (will rain) or 0 (will not rain). The remaining columns encode the features, which should be self-explanatory from the column headers.

Your task is to modify the provided `Lab3_pipeline.py` file to implement an ML pipeline to perform the `RainTomorrow` classification task. Specifically, your `Lab3_pipeline.py` file must do the following:

- Import only built-in Python libraries or those from `sklearn`, `numpy`, `scipy`, `pandas`, `matplotlib`, or `seaborn`. If you need additional libraries, ask Prof. Apthorpe so they can be added to the automated leaderboard environment as well.
- Implement a class `WeatherPipeline` with the following methods:
  - A `fit()` method that loads the training data from `Lab3_train.csv`, preprocess the data, and trains a model to perform the `RainTomorrow` binary classification task. The model, as well as any necessary preprocessing information (e.g. OneHotEncoder or StandardScalar objects), should be saved as class fields for later prediction tasks.
  - A `predict()` method that loads the test data from `Lab3_test.csv`, preprocesses it, and predicts a `RainTomorrow` value for each row. You can assume that `Lab3_test.csv` has exactly the same format as `Lab3_train.csv` but without the `RainTomorrow` column. The `predict()` method should return the predictions as a NumPy array of 1s and 0s with exactly as many elements as there are rows in `Lab3_test.csv`.
  - Any other helper methods you choose.

Any of the above methods may also take optional keyword arguments if helpful for your development and testing. Your `Lab3_pipeline.py` file may also include a `main()` function or any other functions not in the WeatherPipeline class, but these functions will not be called by the automated leaderboard code.

Getting Started Hints

1. Test your model via cross-validation on the training set in between uploads to the leaderboard to save time.

2. This dataset has some missing data points that need to be handled. You should NOT drop any examples just because they include missing data. Instead you should replace the missing values with something more reasonable *that is the same type as the existing data in the column* (use `.dtype` to check the type of a Pandas column).

3. This dataset has a mix of nominal and numeric features, some of which may need to be re-encoded. Although decision trees are theoretically able to handle nominal and numeric features, the Scikit-Learn DecisionTreeClassifier implementation requires numeric features.

4. **Read `Lab3_questions.txt` before you start.** You will need to perform certain analyses to answer these questions, which you should plan for at the outset.

Leaderboard Submission

You may submit your model to the leaderboard as many times as you like before the due date next Friday. You will only be eligible for the first round of extra credit if you have submitted a model before the end of lab today. You may then continue to iterate until next Friday when the second round of extra credit will be awarded. You are encouraged to submit your model often to see how small changes affect test set performance.

When you upload your `Lab3_pipeline.py` file to Gradescope, it will ask you to supply a "leaderboard name." You may enter your real name if you don't mind not being anonymous. Otherwise enter something anonymous that you will remember.

Final Submission

Once you are satisfied with your model's performance on the test set, also submit your completed `Lab3_questions.txt` to Gradescope.

Extra Credit Opportunity

If you find a bug anywhere in this lab, please inform Prof. Apthorpe. The first student (or breakout room) to find any particular bug will be given a small amount of extra credit. This will help make the course better for students in future years.