Final Project – Machine Learning (CS4342, Whitehill, Spring 2021)

1 Teams

For the course final project, you should form teams of at least 3 people and at most 4 people. (If you need help finding teammates, fret not – we will facilitate.) Each team will spend the last few weeks of the course tackling a particular Kaggle competition. You are free to choose any competition that is **not** a "Getting Started" competition or a "Limited-Participation" competition. It is ok, however, if the competition has already closed.

2 Learning Goals

At a high level, your learning goals for this project, and your chosen competition, are to: (1) Apply the theory you have been learning in class to tackle a real-world problem. (2) Practice using some off-the-shelf machine learning software. In terms of machine learning practice, your first step will be to understand what you are trying to predict/perceive (the y values), and what kinds of features you have to predict/perceive them from (the x values). To get started, you should think of a **very simple** (i.e., almost braindead) baseline, just to make sure you understand the basic problem setup, how to submit predictions to Kaggle, etc. Over the course of the project, you will iteratively (a) brainstorm ways of improving accuracy of your predictions; (b) implement these strategies in code; and (c) assess how well they worked. In terms of software, you will likely start with sklearn because it offers a uniform interface to experiment with different models and has a relatively shallow learning curve. Later on, you should try more specialized packages, e.g. Keras for neural network training. Here is a link to help you get started: `https://keras.io/getting-started/sequential-model-guide`

3 Team Meetings

To help you get started, we (the TAs and I) will be meeting with you starting next week in weekly team meetings (20 min). Because there will be about 20 teams and there are only three of us, we will divide up the teams among the 3 course staff, and rotate when who meets with whom over the remaining 3 weeks of the course. Before your first team meeting with your assigned course staff member, try to meet as a group to discuss what your competition is about, what data are available, and some initial thoughts about how to get started (i.e., what might be predictive of what). In subsequent meetings, we will discuss your progress – e.g., what approaches you've tried, what accuracy/loss values they attained – and also plan next steps to try to improve the performance of your approach.

4 Requirements

All teams are required to explore both **shallow** (e.g., linear or softmax regression, SVMs, random forests, boosting models, etc.) as well as **deep** (i.e., multi-layer neural networks) architectures to tackle its selected Kaggle competition. Concretely, you should use sklearn for several shallow models, and a state-of-the-art neural networks package (Keras, TensorFlow, PyTorch, etc.; Keras is probably easiest) for deep models. To give each model the best chance of succeeding, you should perform a hyperparameter search. To optimize hyperparameters fairly without estimating their associated loss/accuracy value unfairly, you should use a held-out validation set (disjoint from the testing set).

Depending on your particular project, different machine learning strategies and techniques will be appropriate. For some, it may be useful to synthesize new training examples by performing label-preserving augmentation methods. For others, it might be more useful to manually design, based on intuition and domain knowledge, a new set of features based on the raw inputs provided by the competition (feature

engineering). It could be useful and interesting to estimate how the accuracy of a trained machine would grow as more and more training data are added (you can assess this by training on subsets of increasing size and measuring accuracy on a controlled test set).

This project will constitute 25% of your final course grade, and there are several people on each team. That means that, regardless of the particular set of models and techniques your team chooses to explore, the sum of work on your project should be significant. To claim credit for your work, you should also make sure that your final report describes it all (see below). Here are some suggestions for things you might explore:

- Feature engineering
- Data augmentation
- Literature search through published papers (e.g., on Google Scholar) for ideas on how to tackle the problem.
- Error analysis: what kinds of mistakes did your models make, and what clues about how to improve the models does this analysis suggest?
- Experiments on accuracy versus training set size.

5 Deliverables

There are two deliverables:

- **Models: Implementation and Training Code**: A sequence of machine learning models, and associated training procedures, to tackle your chosen prediction problem. The code (typically in Python, though it can be in any programming language of your choice) necessary to train your models will be submitted in a Zip file. Most teams will use off-the-shelf software (e.g., Keras, sklearn), in which case the implementation is given to you. However, some teams might possibly implement their own models (e.g., a variant of the stepwise classification from homework), in which case their implementations should be submitted as well.
- **Report**: A simple report that describes precisely (even a child should be able to understand it as long as they have a machine learning background!), comprehensively (it describes all the work your team expended on this project), and coherently (it should tell a story; it should not just be a set of loosely connected tasks). **Important**: Make sure that if you borrow an idea from another researcher that you cite them (names, title of their work, year, and where it was published). The format of your report should be as follows:

  1. Title of the competition, list of students in your group and your email addresses.
  2. Introduction: succinctly but precisely describe what the goal of the competition is, including (a) what you are trying to predict/perceive; (b) what you have to predict/perceive with; (c) why the problem is important or interesting.
  3. Methods: this is the bulk of your report. In chronological order, describe the sequence of models and techniques that you tried, starting from a **very** simple baseline model. For each model, report its performance, what you learned from the experiment, and which model/technique you therefore decided to try next. The shallow models you try will likely come first, and the deep ones will come later.
  4. Table of Results: Include a table that lists succinctly all the models/techniques you tried and their associated accuracy/loss values.
  5. Conclusions: For specific techniques that you tried (e.g., feature engineering according to a specific strategy), did it help? Which worked better – the shallow or the deep models?
  6. References: Did you borrow ideas from anyone else's work? If so, you must cite them!

6 Grading

Since each team may have selected a different Kaggle challenge, there will inherently be some variability of the particular criteria used to grade each project. The primary metric is **the amount of *demonstrated* work and ingenuity** towards tackling your problem. The work you invested should be made clearly in the report. Just implementing some nested for-loops of hyperparameter optimization and trying a few plug-and-play models in sklearn is very easy and requires neither much work nor ingenuity; correspondingly, it would receive a bad grade. Your report (not the code) is the primary means of demonstrating the effort you invested; hence, make sure that all your teams important accomplishments are described clearly. Obviously, the code you implemented must correspond to what you say you did in your report. Note that strong performance in terms of accuracy/loss values in the competition is **not** required to get an A on the project; however, competently and creatively tackling the problem is.

7 Submission

Submit your report and code in a file called `final_project_WPIUSERNAME1_WPIUSERNAME2....zip`.