15.076 Homework 5

Due: April 20, 11:59 pm

Problem 1: Graduate School Admission

Understanding graduate school admission plays a vital role both for the students applying, but more importantly, to the admission administrations to pinpoint factors that could potentially bias their decisions. However, it is important to keep in mind of the real-world applicability of your model: who will be using it? how easy it is to convince them to trust your model? As you will see, tree models are perfect match for such a scenario where interpretability is key.

In this problem, you will explore how to use Optimal Regression Trees to predict a student's chance of graduate school admission. Please download the admission.csv from Canvas, which is a dataset from `https://www.kaggle.com/mohansacharya/graduate-admissions`. Please make sure you have IAI installed correctly before starting. When you are unsure about any syntax, please go to: `https://docs.interpretable.ai/stable/OptimalTrees/quickstart/regression/#qs_ot_regression` for any questions.

a) Load the data, select *Chance of Admit* as your outcome, and the rest as your feature matrix X. Split the data into 80% train and 20% test using random seed 1, and make sure your specify the splitting is for a regression problem.

b) Construct a gridsearch with OptimalTreeRegressor and random seed 1. Try out some different values of max depth and min bucket, what values do you think are reasonable for them for this problem? Report the best parameter combination, as well as the MSE error for both train and test after training.

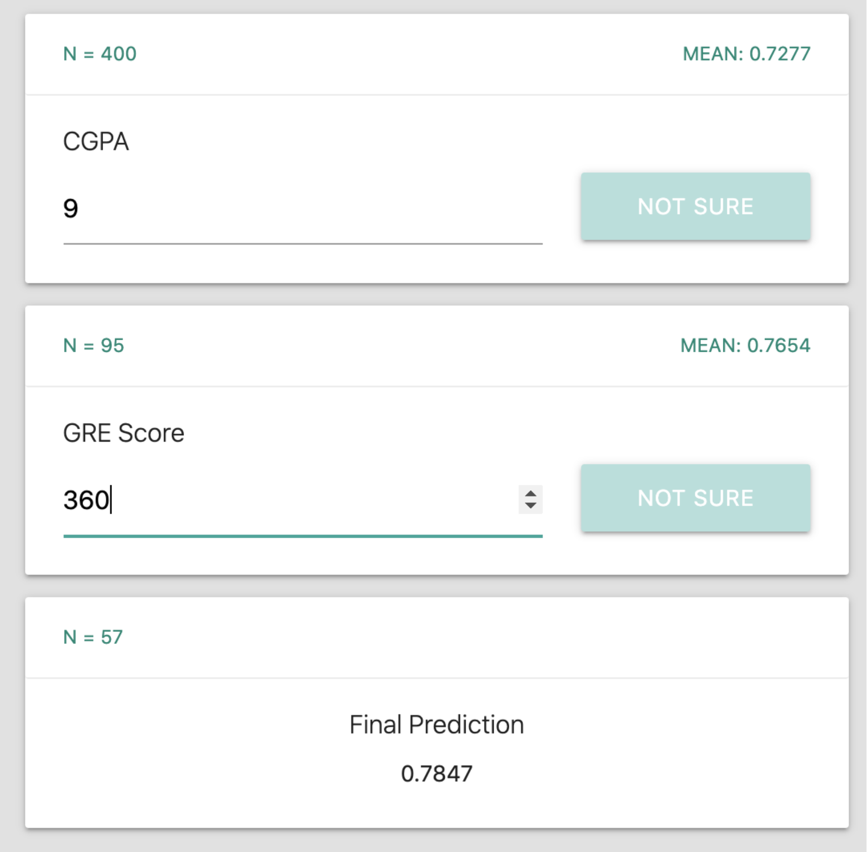
c) Interpret the trees by detailing the variables in the splits, the position of each splits (i.e. root/leaf), as well as the values of each feature split (i.e. Split on GRE Score 300). Use $IAI.variable_importance(lnr)$ to report the most important features. What have you observed that are the three most important factor of graduate admission? Do you think the admission metric has anything that can be improved upon?

d) Sometimes using only a small subset of the features can improve our performance drastically by reducing noisy signals. Use only GRE score and CPGA and the same gridsearch parameter, Report the MSE error for both train and test after training.

e) As discussed in lecture, besides vertical splits of the feature space, we could also construct hyperplane splits (such as a diagonal line). Construct an OptimalTreeRegressor with hyperplane using the same set of gridsearch parameters as above, does your performance improve? Report the MSE error for both train and test after training.

e) Save your best performing tree as an html file, also save it as a questionnaire file. Try open both and play with it. Attach a screenshot of you inputting an imaginary student's statistics in the questionnaire, what is their predicted chance of graduate admission?

Solution: See code and screenshot below


N = 400

MEAN: 0.7277

CGPA

9

N = 95

MEAN: 0.7654

GRE Score

360

N = 57

Final Prediction

0.7847

Problem 2: Bird Species Identification

Wildlife monitoring plays a vital rule in helping scientists improve their knowledge about wildlife presence and distribution, sometimes even animal behavior. This information helps policy makers to make and evaluate the effects of protection as well as recovery efforts. Many of these monitoring projects are community-based, volunteer-lead, where participants set up and maintain cameras stations to track different animal species. As you can imagine, these camera stations gather a vast amounts of high-quality image data. However, it remains a key obstacle to harness the potential of these data because of the huge cost of manual labor to label and analyze them.

In recent years, many efforts have been dedicated to help automate this process, and the nature of image-based database has made deep learning a particularly relevant methodology to be investigated. In this problem, you will explore how to use a simple convolutional neural network to classify sparrows, a type of bird that can be found frequently in Boston.

A skeleton jupyter notebook has been provided to you, please upload it to your google drive and launch it on Google Collab. If you prefer to use your own local machine, that is also completely acceptable for us. Now follow these steps:

- Download the Caltech-UCSB Birds-200-2011 dataset either by running the first line of the skeleton code (!wget), or manually from `https://drive.google.com/file/d/1hbzc_P1FuxMkcabkgn9ZKinBwW683j45/view?usp=sharing`
- If you downloaded it manually, upload it to your google drive.
- Extract the tar.gz file using the code provided in the skeleton.

a) Split the data into 70% train and 30% test using random seed 42, write a function to calculate the percentage of sparrows in train and in test. Report the percentages.

b) You should see that the two classes available (Sparrow vs. Not Sparrow) are not balanced from your calculation above. One way to improve this is data augmentation: generate synthetic, slightly modified minor class samples to amplify its presence in the original dataset. Use the ImageDataGenerator function to define 4 different types of augmentation techniques. Explain what augmentations you have applied here. You can find more information here: `https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/ImageDataGenerator`.

Note: In practice, more augmentations than 4 sometimes is recommended. If you have enough RAM, try more augmentations and see if the performance improve even further (this part is optional).

c) Prior to any model training, ensuring your data has been standardized is always a good practice. For the images provided:

- Resize each image to 224 x 224 (answer has been provided)
- Convert image to RGB type with 3 channels, use $load_img$ from `https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/load_img`
- Normalize each image's pixel values from 0 to 1 (i.e. divide by 255) using img_to_array function from `https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/img_to_array`

d) Now, let's build the neural network together! We have given you a starter of the first layer, as well as the last layer, follow structure detailed below and complete the intermediate layers.

- 1st: Convolutional 2D layer with 64 output filters, with a (5, 5) convolution window (kernel), Relu activation. (provided)
- 2nd: Maxpooling 2D layer with a pool size of (3, 3)

- 3rd: Dropout layer that random drops 25% of input units
- 4th: Convolutional 2D layer with 64 output filers, (3, 3) convolution window, relu activation.
- 5th: Maxpooling 2D layer with a pool size of (3, 3)
- 6th: Dropout layer that random drops 25% of input units
- 7th: Flatten layer
- 8th: Dense layer, output space dimension of 64, relu activation
- 9th: Dropout layer that random drops 25% of input units
- 10th: Dense layer, output space dimension of 1, sigmoid activation (provided)

e) What loss do you think is suitable for this binary classification problem? Use your favorite optimizer and "accuracy" as our evaluation metrics, Let the model run for 5 epochs. Detail the loss behavior you have observed.

f) Report the AUC, accuracy for both train and test. Also report the accuracy of our prediction for each individual sparrow class, which one do we perform best, which one do we perform worst? Is there any intuition behind why this is the case?

Solution: See code.