Due: Apr. 1, 2025 @ 11:59 p.m.

General notes to keep in mind:

- *All deliverables for the assignment must be submitted as a **single ZIP file per group** via the Brightspace D2L course shell. Submissions containing multiple ZIP files per group or those with a file that is not in the ZIP format will NOT be graded.*
- *The code submitted must be written purely using the Python programming language and it should execute within the Python 3.11.1 interpreter running on the Windows operating system (version 10 or above). The submitted code should NOT require external python modules other than scikit-learn 1.6.1 , matlplotlib 3.10.1, pandas 2.2.3 and their dependencies.*
- *Read the "Assignment code submission requirements" carefully and prepare the code accordingly. It is your responsibility to ensure that the submitted code executes. If the grader is unable to execute your code and/or your code does NOT adhere to the submission requirements, your code may not be graded.*
- *The written responses required to the questions in the assignments must be compiled into **single PDF** file named as `report.pdf`. You are encouraged to use LaTeX for typesetting your written responses, but however, the use of Microsoft Word™ or any other such programs is also acceptable.*

Prediction of conversion to Alzheimer's Disease (AD) using glucose metabolism measures

We once again examine the problem of diagnosing Alzheimer's disease (AD) or the dementia of Alzheimer's type (DAT) based on changes in brain glucose metabolism. However, instead of addressing the task of distinguishing between the stable normal control (sNC) and stable DAT (sDAT) groups that lie at extreme ends of the spectrum of AD disease (see Figure 1), we will tackle a more clinically relevant and challenging task of identifying individuals that will convert to AD among individuals suffering from mild cognitive impairment (MCI).

The specific goal of this assignment is to develop a **tree-based classifier that can predict if an individual belongs to the stable mild cognitive impairment (sMCI) group or the progressive mild cognitive impairment (pMCI) group** based on a high-dimensional glucose metabolism signature taken from several regions in the individual's brain.

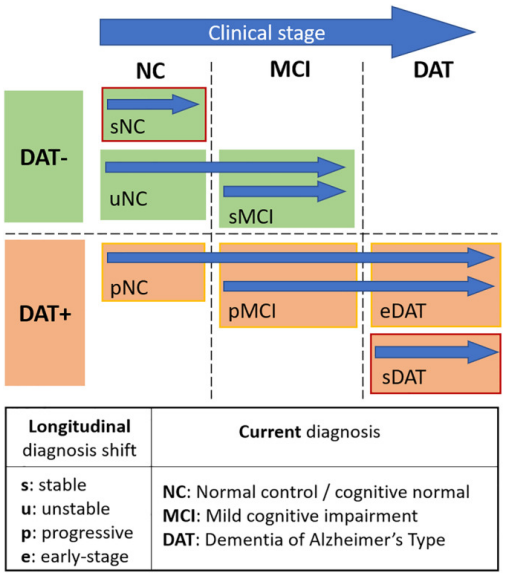

Figure 1: Different stages of disease progression in AD (see Popuri et. al. 2020).

Data

Data for this assignment can be downloaded from here. The "training" dataset consists of glucose metabolism features taken from 14 cortical brain regions (see `fdg_pet.feature.info.txt` for a list of these regions) across 202 sMCI and 202 pMCI individuals given in `train.fdg_pet.sMCI.csv` and `train.fdg_pet.pMCI.csv` respectively. The `test.fdg_pet.sMCI.csv` and `test.fdg_pet.pMCI.csv` files correspond to a "test" dataset with the same features of brain glucose metabolism taken from another 361 sMCI and 194 pMCI individuals respectively. Furthermore, brain glucose metabolism characteristics of yet another 100 sMCI and 100 pMCI individuals have been gathered and will be used as the "independent test" dataset to perform a "blinded" validation of your submitted tree-based classification model.

Performance reporting convention

Always report *accuracy*, *sensitivity*, *specificity*, *precision*, *recall* and *balanced accuracy* performance metrics when summarizing the performance ("Err") of a classification model.

Question 1 [25 marks]

Train a "single" *decision tree* classification model to discriminate between the sMCI and pMCI individuals based on the brain glucose metabolism features. Use the "training" dataset and a cross-validation (CV) based grid-search approach to tune the "feature testing criterion". In our class, we discussed the "entropy" measure (see *Lecture 10*, *Slide 6*), but the scikit-learn implementation of the decision tree classifier provides the option of using two other measures namely "gini" and "log loss". Using the "best" "feature testing criterion" setting, re-train on the entire "training" dataset to obtain the final decision tree classification model. Estimate the "Err" of this final model on the "test" dataset. Discuss the performance of the models explored during the "feature testing criterion" hyperparameter tuning phase.

Question 2 [25 marks]

Plot the final decision tree classification model obtained above using the "plot tree" method in scikit-learn. Briefly comment on the structure of the learned decision tree model. Which features have been chosen as the most important by the model?

Question 3 [25 marks]

Now train a *random forest* classifier for the sMCI vs pMCI classification task. Use the "training" dataset and a CV based grid-search approach to tune the "feature testing criterion" among the "gini", "entropy" and "log loss" measures available in scikit-learn random forest classifier implementation. Using the "best" "feature testing criterion" setting, re-train on the entire "training" dataset to obtain the final random forest model. Estimate the "Err" of this final model on the "test" dataset. Compare the performance of the "single" decision tree model from above with that of the obtained final random forest model.

Question 4 [25 marks]

Leveraging the experience you gained from the experiments thus far, design the "best" tree-based classification model for discriminating between the sMCI and pMCI groups using the glucose metabolism features derived from the 14 cortical brain regions. You may also employ any other strategies that we have not discussed in the course to train this "best" tree-based classifier. The only restriction is that, you may not use datasets other than the ones provided as part of this assignment. Submit this "best" tree-based classifier as the following method:

```
def predictMCIconverters(Xtest, data_dir):
"""Returns a vector of predictions with elements "0" for sMCI and "1" for pMCI,
corresponding to each of the N_test features vectors in Xtest

Xtest     N_test x 14 matrix of test feature vectors

data_dir  full path to the folder containing the following files:
          train.fdg_pet.sMCI.csv, train.fdg_pet.pMCI.csv,
          test.fdg_pet.sMCI.csv, test.fdg_pet.pMCI.csv
```

```
"""
```

The above method will be evaluated on the "independent test" dataset by the grader to determine the classification performance. See note below regarding the grading rubric for this question.

Note on grading

The grading for Question 1, Question 2 and Question 3 will be based on the appropriateness of the submitted code and the written responses. The grading for Question 4 will be based on the relative performance of your trained model. The submission(s) with the best performing model (referred below as 1^{st} ranked model) in terms of *balanced accuracy* (rounded to 4 decimal places) will receive full marks on Question 4 (i.e., 25 marks). All other submissions will receive marks that are proportional to the decrease in performance of their model with respect to the 1^{st} ranked model. For example, if the balanced accuracy of the model of a given submission is 10% lower than the 1^{st} ranked model, then that submission will receive 22.5 marks for Question 4.