Machine Learning – Assignment II

All about supervised learning, too

- **Requirements**: For each assignment in this course, ensure you submit the following:
- **Document**: A comprehensive document in either PDF or Word format detailing your implementation process and any challenges you encountered.
- **Source Code & Compiled File**: Include both your source code and the compiled file (in .exe, .dmg, or .sh format). Accompany these with a README file that provides instructions on how to launch the compiled program.
- **Code Comments:** Ensure that your source code contains key comments explaining crucial parts of your implementation. If any portion of your code is derived from existing sources on the Internet, provide appropriate citations within your comments.
- **Note**: For this assignment, you are **prohibited** from using external packages, with the exception of those used for visualization purposes.

- **Problem set (110pt):**

Dataset: We are going to explore real-world applications and scenarios for this assignment. The dataset and its metadata can be found on Kaggle below

https://www.kaggle.com/datasets/ifteshanajnin/carinsuranceclaimprediction-classification?select=train.csv

Note that all the training/testing processes should perform on train.csv only, and you need to manually & randomly partition the train.csv into train/validation/test datasets.

- **(60pt) Application of Kernel Methods in Ensemble Learning Across Various Learners:**

This assignment focuses on the innovative application of kernel methods beyond their conventional use in Support Vector Machines (SVMs). Kernel methods are powerful in mapping data into higher-dimensional spaces, enabling complex decision boundaries in various types of models. Students are tasked with exploring and implementing kernel methods in different types of machine learning models (not limited to SVMs) and then combining these kernel-enhanced models in an ensemble learning framework. The key challenge here is to integrate kernel methods in various learners and to understand how their combined effect improves the overall predictive performance.

Example Hint 1:
Consider applying a Radial Basis Function (RBF) kernel to a neural network model. Train this model on a dataset, focusing on capturing non-linear patterns. Then, combine this neural network with a kernelized decision tree, where each tree in the ensemble uses a different subset of features. The ensemble can be integrated using a weighted voting mechanism, where weights are based on the validation performance of individual models.

Example Hint 2:
Experiment with integrating a polynomial kernel in a logistic regression model. Train multiple such models on random subsets of the data, each with a different degree of the polynomial kernel. Simultaneously, develop a set of simple kernelized k-nearest neighbors (KNN) models. Each KNN model uses a distinct value of k and a distinct kernel function. The final ensemble model combines the outputs of these logistic regression and KNN models, possibly using a technique like stacking or majority

voting..

- **(60pt) Ensemble of Deep Learning-Based Non-Tree Weak Learners**:

  Traditionally, ensemble methods like Random Forests utilize tree-based models as weak learners. This assignment encourages exploring the use of deep learning models, specifically simple 2-layer Multi-Layer Perceptrons (MLPs), as weak learners in an ensemble. Students need to design an ensemble learning framework where multiple such MLPs are trained on different subsets of the data or features and then aggregated to form a more robust model. The focus should be on analyzing the strengths and challenges of using these non-traditional weak learners compared to standard tree-based models and how to effectively aggregate their predictions.

  Example Hint 1:

  Develop a 'Deep Random Forest' where each 'tree' is replaced by a 2-layer MLP. Each MLP is trained on a random subset of features and instances. After training, the outputs of these MLPs can be combined using simple averaging or a more complex method like weighted averaging based on each MLP's performance on a validation set.

  Example Hint 2:

  Create an ensemble of small 2-layer MLPs, where each MLP is designed to capture different aspects of the data, such as focusing on different clusters or segments within the data. After training these MLPs individually, use a meta-learner, like a logistic regression model, to learn how to best combine their outputs. This meta-learner takes the outputs of all the MLPs as inputs and predicts the final output..