Machine Learning Assignment 2

1st Ming-Xuan Wu
Institute of Data Science
National Cheng Kung University
Tainan, Taiwan
Email: RE6124019@gs.ncku.edu.tw

***Abstract*—This paper embarks on a comprehensive exploration of supervised learning within the realm of machine learning, delving into in-depth research and practical implementations. The primary focus is on the analysis of a dataset sourced from Kaggle, specifically the Automobile Insurance Claims Prediction dataset. The overarching goal is to scrutinize the data and forecast whether policyholders are inclined to submit claims in the ensuing 6 months. The initial phase involves meticulous data preprocessing to streamline subsequent tasks.**

As the course progresses, the spotlight shifts towards Ensemble Learning, with detailed instruction covering techniques such as bagging, bootstrap, and others. Notably, the instruction on Support Vector Machines (SVM) places emphasis on elucidating the significance of the Radial Basis Function (RBF) kernel. In addressing the first question of the current assignment, I augment the linear classifier perceptron with an RBF kernel. This augmentation involves projecting low-dimensional data into a higher-dimensional space and implementing a voting mechanism to establish an Ensemble Learning framework.

Building on the previous assignment's implementation of the Multilayer Perceptron (MLP) method, the second question involves designing a two-layer MLP. Subsequently, this weak learner is incorporated into an ensemble learning approach reminiscent of random forest. A comparative analysis with traditional random forest methodology is then conducted, highlighting the integration of advanced techniques into the learning framework. This integration showcases the amalgamation of linear classifiers with non-linear transformations and the blending of MLPs with random forest principles, all with the overarching aim of achieving enhanced predictive capabilities.

***Index Terms*—Machine Learning, supervised learning, Classification, Feature engineering, Cross-Validation.**

I. Introduction

In this study, we undertake the task of implementing and evaluating several fundamental machine learning classifiers. The primary objective is to compare the performance of these classifiers and gain insights into their strengths and weaknesses. The classifiers under consideration include a basic linear classifier, a k-nearest neighbors (K-NN) classifier employing three distinct distance metrics, a naive decision tree classifier, and a decision tree classifier with a pruning algorithm.

Additionally, we explore feature engineering by developing an algorithm to assess "feature importance" for both linear classifiers and decision trees.To assess the classifiers' performance, we employ a carefully chosen evaluation criterion and validate the models using k-fold cross-validation. The evaluation criterion, yet to be specified, plays a crucial role in determining the effectiveness of each algorithm. Furthermore, we propose an algorithm to derive new features, acknowledging the potential limitations of the original feature set in improving model accuracy.To gain deeper insights into feature importance, we integrate the SHAP (SHapley Additive exPlanations) library with our implemented algorithms. This enables a comparative analysis between our findings and those obtained through SHAP, offering a comprehensive understanding of feature relevance.

Finally, we address the challenge of aggregating predictions from k classifiers in k-fold cross-validation. We design an algorithm for merging and aggregating these predictions, comparing the performance and complexity with the results obtained in the initial problem. To establish the true superiority of one model over another, we compare the results of 5-fold cross-validation with those obtained in a single train-test split scenario (Problem 1). This comparative analysis allows us to justify the robustness and reliability of the chosen classifiers, shedding light on their generalization capabilities and overall performance.

II. Method

A. *Ensemble Learning*

[1] [2]

1) Voting
Voting ensemble construction comprises three primary steps. Initially, we introduce the suite of sub-models to be employed. Subsequently, these models are compiled into a list structure. Finally, the modeling process follows established methodologies. This systematic procedure facilitates the amalgamation of various models into the ensemble framework.
2) Bagging
Bagging, short for bootstrap aggregating, involves three main steps. Initially, the bootstrap method, a form of sampling with replacement, is employed to obtain multiple subsets from the original dataset. These subsets are then utilized for model training. The process is repeated N times, typically exceeding 1000 iterations, to derive N statistical measures. Finally, these N measures are employed to compute the confidence interval for the desired statistical quantity. In the context of Bagging, the bootstrap method is used to draw N subsets from the entire dataset. Each subset is employed to train an individual model, and the final prediction is a combination

of the outputs from these N models. For classification problems, a voting mechanism is employed, while for regression problems, the outputs of the N models are averaged. Random Forest is an example of Bagging, employing a random method to construct a forest of decision trees. Each tree is independently trained with a subset of the data.

3) Boosting
Boosting is a machine learning algorithm designed to reduce bias in supervised learning. It involves training a series of weak classifiers and combining them to form a strong classifier. A representative algorithm in Boosting is AdaBoost (Adaptive Boosting). Initially, all training examples are assigned equal weights. The algorithm then iteratively trains on the dataset, assigning higher weights to incorrectly classified examples to emphasize learning from those errors. This process results in multiple predictive functions. Gradient Boost Decision Tree (GBDT) is another Boosting method that aims to minimize the residuals from the previous iteration. Unlike AdaBoost, GBDT builds a new model in the direction of reducing the residual (negative gradient). Stacking methods involve training a model to combine the outputs of various other models. Multiple diverse models are trained, and their outputs serve as inputs for training a final model to obtain the ultimate output. Logistic regression is commonly used as the combining strategy for Stacking.

4) Stacking
Stacking is a method where a model is trained to combine the outputs of other models. Initially, multiple distinct models are trained, and their outputs are utilized as inputs to train another model, yielding a final output. Theoretically, Stacking can represent the two Ensemble methods mentioned earlier, provided an appropriate model combination strategy is adopted. In practice, logistic regression is frequently employed as the combining strategy. The process involves training Tier 1 classifiers using bootstrap-sampled subsets from the entire training dataset. The outputs from these classifiers are then used to train a Tier 2 classifier. Stacking serves as a versatile approach for combining the strengths of different models, enhancing overall predictive performance.

B. Implementation

1) Question 1
In this study, I implemented the integration of kernel methods into ensemble learning across various machine learning models. Diverging from their conventional application in Support Vector Machines (SVMs), kernel methods are employed across diverse models to leverage their ability to map data into higher-dimensional spaces, facilitating the capture of intricate patterns. The primary emphasis lies in enhancing a Perceptron-based classifier using a Radial Basis Function (RBF) kernel. The Perceptron undergoes training on a dataset, emphasizing its capability to discern non-linear patterns. Concurrently, a kernelized decision tree is introduced into the ensemble, with each tree utilizing a distinct subset of features. The ensemble is consolidated through a weighted voting mechanism, where weights are assigned based on individual model performance during validation.
The implemented 2-layer Perceptron(MLP) [3], enhanced with the RBF kernel, undergoes training and validation phases, with a focus on monitoring accuracy and loss metrics. The ensemble's performance is comprehensively evaluated by assessing both the Perceptron and the kernelized decision tree. This approach aims to demonstrate the effectiveness of combining kernel-enhanced models in an ensemble learning framework, showcasing improvements in predictive performance across diverse learners.
This methodology provides a thorough exploration of integrating kernel methods into various learners, offering insights into their combined effects and their potential to enhance overall predictive performance in ensemble learning scenarios.

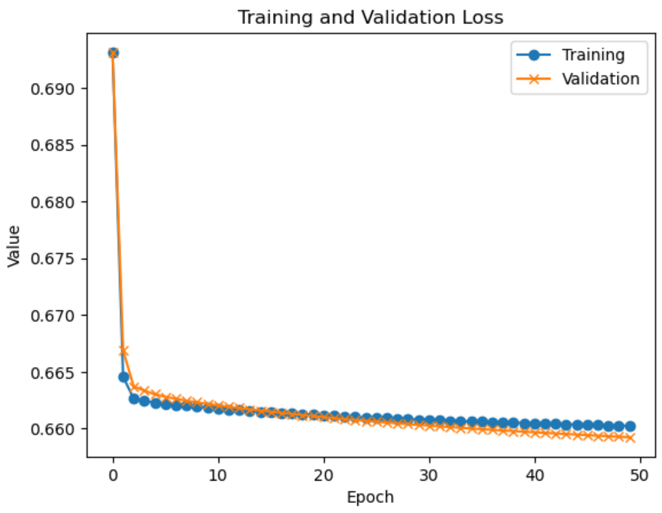

Fig. 1: Perceptron Train Val Loss

2) Question 2
The implementation involves defining an 2-layer MLP class capable of forward propagation, backpropagation, and training. Subsequently, a novel ensemble, termed 'DeepRandomForest,' is introduced, which encapsulates multiple MLPs trained on random subsets of data and features. The ensemble leverages a weighted averaging mechanism for combining individual MLP outputs. Additionally, for comparison, a traditional ensemble of Decision Trees (Random Forest) is implemented.
The accuracy of both the 'DeepRandomForest' and the traditional 'RandomForest' [4] is assessed using standard evaluation metrics. The study aims to provide insights into the efficacy of employing deep learning-

based non-tree weak learners in ensemble settings and their performance relative to traditional tree-based models.

This methodology offers a comprehensive investigation into the integration of deep learning models as weak learners in ensemble learning, shedding light on their potential and limitations compared to conventional tree-based approaches.

III. Conclusion

In summary, this machine learning task provided a foundational exploration of essential techniques within ensemble learning, encompassing key principles such as Voting, Bagging, Boosting, and Stacking.

The practical implementation involved employing Voting in the first scenario, where multiple models, each incorporating the RBF kernel with different gamma parameters, participated in a voting process. Notably, models with gamma values of 0.01, 0.0095, and 0.009025 exhibited comparable accuracies at 0.6264. However, the ensemble's performance post-Voting demonstrated a modest enhancement, yielding an accuracy of 0.61125. Two primary factors contributed to this outcome: firstly, challenges arose as the Perceptron struggled to precisely classify data after integrating the RBF kernel, leading to suboptimal predictive outcomes. Secondly, the efficacy of the Voting ensemble method in significantly improving model accuracy in this particular context was called into question.

Moving to the second task, the implementation involved a two-layer MLP weak learner, and a "DeepRandomForest" ensemble was constructed using Bagging Ensemble Learning. A comparative analysis with the traditional RandomForest revealed closely aligned accuracies, with "DeepRandomForest" achieving 0.61125 and RandomForest achieving 0.61125.

References

[1] T. G. Dietterich *et al.*, "Ensemble learning," *The handbook of brain theory and neural networks*, vol. 2, no. 1, pp. 110–125, 2002.

[2] R. Polikar, "Ensemble learning," *Ensemble machine learning: Methods and applications*, pp. 1–34, 2012.

[3] A. Pinkus, "Approximation theory of the mlp model in neural networks," *Acta numerica*, vol. 8, pp. 143–195, 1999.

[4] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.