

Machine Learning

Prepared by Dr. Yasser Abdelhamid

DS 620
Machine Learning

Lecture 10



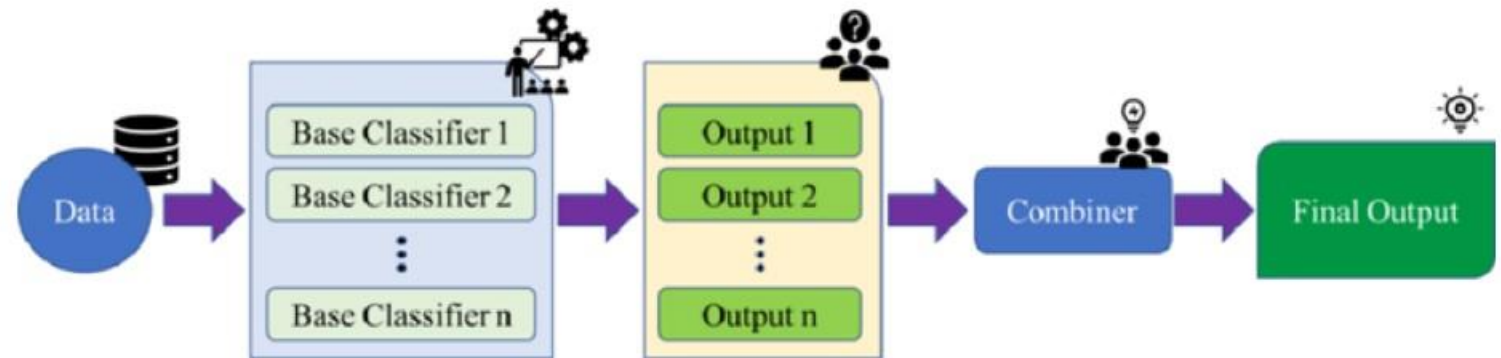


OUTLINE

- Ensemble learning
- Bagging
- Boosting
- Stacking
- Cascading
- Random forests

ENSEMBLE LEARNING

❖ Ensemble Learning is a technique that combines multiple individual models (learners) to achieve better predictive accuracy and generalization than any single model alone.



BENEFITS OF ENSEMBLE LEARNING

- ❖ Improved robustness,
- ❖ Reduction of bias and variance, and
- ❖ Better handling of complex datasets.

BASE CLASSIFIERS

- ❖ The **individual classifiers** in an ensemble are known as base classifiers.
- ❖ **Homogeneous base classifiers**
 - Base classifiers are all of the same kind (e.g. decision trees).
- ❖ **Heterogeneous base classifiers**
 - Base classifiers are not all of the same kind.

ENSEMBLE CLASSIFICATION ALGORITHM

- 1. Generate N classifiers for a given dataset**
- 2. For an unseen instance X**
 1. Compute the predicted classification of X for each of the N classifiers
 2. Select the classification that is most frequently predicted.

With N classifiers in the ensemble there will be a total of N votes and the classification with most votes wins.

THERE ARE SEVERAL WAYS IN WHICH AN ENSEMBLE CAN BE FORMED

- ❖ N trees generated using the **same tree generation algorithm**, with different parameter settings, all using the **same training data**.
- ❖ N trees generated using the **same tree generation algorithm**, all with **different training data** and **either with the same or with different parameter settings**.
- ❖ N trees generated using a **different subset of the attributes** for each one.

CLASSIFIER SELECTION STRATEGY

- ❖ Generate trees (or other classifiers) that are **diverse**, in the hope that some will give much better than 'standard' performance.
- ❖ Choose those classifiers with best performance.
- ❖ This leads naturally to the idea of generating a large number of classifiers in some random way and then **retaining only the best**.

ESTIMATING THE PERFORMANCE OF A CLASSIFIER



1. Divide the available data into a test set and the remainder.
2. For each candidate classifier
 - a) Divide the remaining data from step (1) into training data and validation data in some suitable fashion.
 - b) Generate a classifier using the training data.
 - c) Run the classifier against the validation data to give an estimate of its performance.
3. Use the performance estimates to find the best classifiers, e.g. all those with predictive accuracy greater than a specified percentage or perhaps the best X . If the number of classifiers remaining from this step is M , together they comprise an ensemble of size M .
4. Use the ensemble to classify each of the instances in the test set selected at step (1) and use the result as an estimate of the performance of the ensemble on (genuinely) unseen data.

BAGGING

- ❖ Bagging “bootstrap aggregating”
- ❖ Randomly select N instances, one-by-one, at each stage selecting from the full set of instances (we call this sampling with replacement).
- ❖ This will lead to a training set of N instances in which inevitably some of the instances will appear more than once, perhaps several times, and others will not appear at all.
- ❖ There are likely to be many instances left unselected by this process. Collect them together to form a validation set.

MAX VOTING / VOTING CLASSIFIER

- ❖ The max voting method is generally used for classification problems.
- ❖ In this technique, multiple models are used to make predictions for each data point. The predictions by each model are considered as a 'vote'.
- ❖ The predictions which we get from the majority of the models are used as the final prediction.

HARD VOTING

- ❖ Each model in the ensemble casts a single vote for its predicted class label, and the class label that receives the majority of votes is selected as the final prediction.

EXAMPLE

- ❖ Suppose three classifiers predicted the output class(A, A, B), so here the majority predicted A as output.
- ❖ Hence A will be the final prediction.

SOFT VOTING

- ❖ Each model provides a probability or a score for each class label.
- ❖ The final prediction is based on the average of these probabilities or scores.

EXAMPLE

❖ Consider an ensemble with three models, and their predicted **probabilities** for a binary classification task are as follows:

- Model 1: Class A - 0.8, Class B - 0.2
- Model 2: Class A - 0.6, Class B - 0.4
- Model 3: Class A - 0.7, Class B - 0.3

EXAMPLE (CONT.)

- ❖ In soft voting, the final prediction is obtained by averaging the probabilities for each class label:
 - Class A: $(0.8 + 0.6 + 0.7) / 3 = 0.7$
 - Class B: $(0.2 + 0.4 + 0.3) / 3 = 0.3$
- ❖ Based on the higher average probability for Class A, the final prediction would be Class A.

SOFT VOTING VS HARD VOTING

- ❖ Soft voting takes into account the **confidence** or **certainty** of the individual models by considering the probabilities or scores they assign to each class label.
- ❖ It can be advantageous when the models provide well-calibrated probabilities or scores.

BOOSTING

- ❖ Boosting is an ensemble learning technique that combines multiple weak learners to create a strong learner.
- ❖ Boosting focuses on building a sequence of models where each subsequent model aims to correct the mistakes of its predecessors.
- ❖ This iterative process allows boosting to create a powerful ensemble that can handle complex tasks and improve predictive performance.

START WITH WEAK LEARNERS

- ❖ Boosting starts with a weak learner, which is a simple model that performs slightly better than random guessing.
- ❖ Weak learners are typically models with low complexity, such as shallow decision trees.

SEQUENTIAL TRAINING

- ❖ The first weak learner is trained on the original training dataset.
- ❖ During training, the weak learner focuses on the instances that were misclassified or had high errors, attempting to learn from those mistakes.

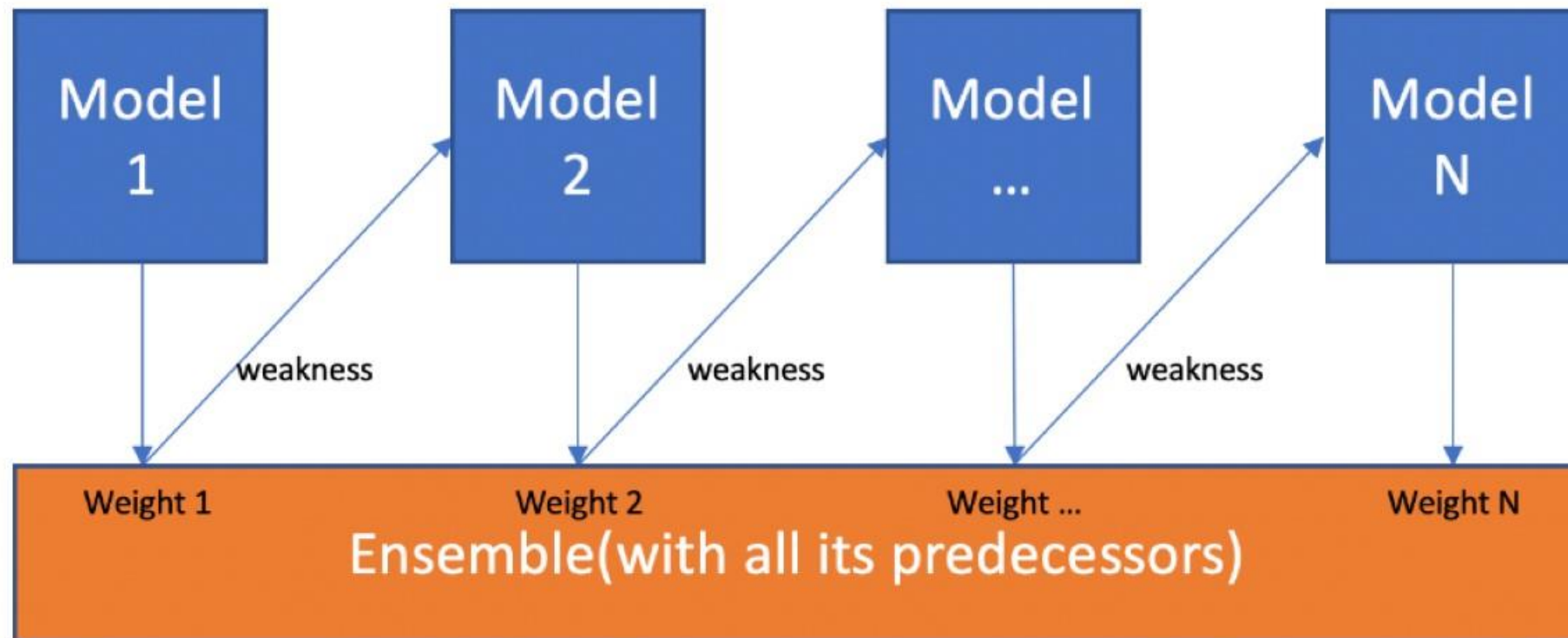
WEIGHTED TRAINING

- ❖ Each instance in the training dataset is assigned a **weight** based on its **difficulty in being correctly predicted by the previous weak learners**.
- ❖ Initially, all weights are equal, but as the boosting process continues, the weights of the misclassified instances increase, making them more influential in subsequent training iterations.

ITERATIVE PROCESS

- ❖ Boosting proceeds through a series of iterations, with each iteration producing a **new weak learner**.
- ❖ In each iteration, the training dataset is reweighted based on the misclassification errors made by the previous weak learners.

BOOSTING ITERATIVE PROCESS



BOOSTING ENSEMBLE PREDICTION

- ❖ To make predictions, boosting combines the predictions of all the weak learners.
- ❖ Each weak learner's prediction is weighted by its performance during training, giving more weight to those that performed better.
- ❖ The final prediction is typically obtained by majority voting for classification problems or weighted averaging for regression problems.

ADABOOST

- ❖ Adaboost (Adaptive Boosting) is one of the most well-known boosting algorithms.
- ❖ It adjusts the weights of misclassified instances to focus on the difficult examples, allowing subsequent weak learners to improve on those instances.

GRADIENT BOOSTING

- ❖ Gradient Boosting is another popular boosting algorithm that builds the ensemble by iteratively fitting new models to the residuals (errors) of the previous models.
- ❖ It uses gradient descent to find the optimal direction in the loss function space.

LIMITATIONS OF BOOSTING

- ❖ Boosting can be **sensitive to noisy data or outliers**, as it assigns higher weights to misclassified instances.
- ❖ The training process of boosting can be **computationally expensive** and time-consuming, especially with a large number of weak learners or iterations.

CASCADING

- ❖ Cascading ensemble learning is an ensemble learning technique that combines multiple models **sequentially**, with each model focusing on solving a specific subset of the problem.
- ❖ Unlike other ensemble methods where all models contribute to the final prediction simultaneously, cascading ensemble learning uses a **hierarchical** approach where the output of one model becomes the input for the next model in the cascade.

STACKING

- ❖ Stacking combines multiple classifications or regression models via a meta-classifier or a meta-regressor.
- ❖ The base-level models are trained on a complete training set, then the **meta-model is trained on the features that are outputs of the base-level model.**
- ❖ The base-level often consists of different learning algorithms and therefore stacking ensembles are often heterogeneous.

STEPS OF STACKING ENSEMBLE LEARNING

- ❖ Split the original training data into two or more subsets: the training set and the validation set.
- ❖ The training set is used to train the base models, while the validation set is used to generate predictions for the meta-model.

BASE MODEL TRAINING

- ❖ Select a set of diverse base models (e.g., decision trees, support vector machines, neural networks) that are capable of capturing different aspects of the data.
- ❖ Train each base model independently on the training set.
- ❖ The base models can use different algorithms, hyperparameters, or feature representations.

GENERATING BASE MODEL PREDICTIONS

- ❖ Use the trained base models to make predictions on the validation set.
- ❖ Each base model generates predictions for the validation set, resulting in a set of prediction values or probabilities.

CREATING THE META-MODEL TRAINING SET

- ❖ Combine the predictions from the base models with the original features of the validation set.
- ❖ The base model predictions serve as additional features in the meta-model training set.
- ❖ The meta-model training set now contains the original features and the predictions from the base models.

META-MODEL TRAINING

- ❖ Train the meta-model using the augmented training set (including the original features and base model predictions) and the corresponding true labels from the validation set.

ENSEMBLE PREDICTION

- ❖ Once the meta-model is trained, it can be used to make predictions on new, unseen data.
- ❖ To make predictions, the base models are used to generate predictions on the test set.
- ❖ These predictions, along with the original features, are then fed into the trained meta-model to produce the final prediction.

RANDOM FORESTS

- ❖ Random Forest is an ensemble learning method that combines the predictions of **multiple decision trees** to make more **accurate and robust predictions**.
- ❖ It harnesses the power of decision trees by introducing **randomness** and **diversity** in the learning process.

BUILDING DECISION TREES

- ❖ Decision trees are powerful machine learning models that make predictions by dividing the input space into regions based on a set of decision rules.
- ❖ Each decision tree is built by recursively partitioning the data based on features and their values, aiming to create homogeneous subsets with respect to the target variable.

RANDOMIZATION

❖ Random sampling:

- Random Forest begins by randomly sampling the training data with replacement (bootstrapping) to create multiple subsets of the data.

❖ Random feature selection:

- At each node of the decision tree, only a subset of features is considered for splitting, introducing randomness and reducing correlation among trees.

❖ These two sources of randomness promote diversity among the decision trees, enhancing the collective predictive power of the ensemble.

AGGREGATING PREDICTIONS

- ❖ After constructing all the decision trees, Random Forest combines their predictions to make the final prediction.
- ❖ For **classification** tasks, the most common approach is to use **majority voting**. Each tree "votes" for a class label, and the class with the majority of votes becomes the final prediction.
- ❖ For **regression** tasks, the predictions of all the trees are **averaged** to obtain the final prediction.

ADVANTAGES OF RANDOM FOREST

❖ Reduced overfitting:

- By aggregating predictions from multiple decision trees, Random Forest mitigates the tendency of decision trees to overfit the training data.

❖ Improved generalization:

- Random Forest reduces variance and provides robust predictions by leveraging the combined knowledge of multiple trees.

❖ Robustness to noise and outliers:

- The ensemble nature of Random Forest makes it more resilient to noisy data and outliers compared to individual decision trees.

ADVANTAGES OF RANDOM FOREST(CONT.)

❖ Feature importance estimation:

- Random Forest can estimate the importance of features, providing insights into which features contribute most to the predictive performance.

❖ Scalability:

- Random Forest can handle large datasets efficiently, as the training process can be parallelized due to the independence among decision trees.

HYPERPARAMETERS AND TRAINING

- ❖ Random Forest has hyperparameters that need to be set before training, such as the number of trees, maximum depth of trees, minimum samples for splitting, and feature subset size.
- ❖ Random Forest can be trained using techniques like bootstrapping to create subsets of the training data and evaluate the model's performance using out-of-bag (OOB) error estimation.

OUT-OF-BAG (OOB) ERROR ESTIMATION

- ❖ Out-of-bag (OOB) error estimation is a technique used in Random Forest to estimate the performance of the model without the need for a separate validation set.
- ❖ It takes advantage of the randomness introduced during the training process.

OOB ERROR ESTIMATION IN RANDOM FOREST

- ❖ During the construction of each decision tree in the Random Forest, a random subset of the training data is selected through **bootstrapping**.
- ❖ Bootstrapping involves randomly sampling the training data with replacement, resulting in each subset having approximately two-thirds of the original data while **leaving out about one-third of the data**.
- ❖ The left out samples are referred to as **out-of-bag (OOB) samples**.

OOB ERROR ESTIMATION IN RANDOM FOREST (CONT.)

- ❖ Since each decision tree is built using a different subset of the data, the OOB samples for each tree are different.
- ❖ For each sample in the training dataset, the OOB samples that were not used to build the decision tree are identified.
- ❖ The OOB samples are then passed down the tree, and their predictions are obtained by following the decision rules of the tree.
- ❖ The OOB prediction for a sample is determined by aggregating the predictions from all the decision trees in the Random Forest that did not use that sample during their construction.

OOB ERROR ESTIMATION IN RANDOM FOREST (CONT.)

- ❖ The OOB predictions are compared to the true labels of the corresponding OOB samples.
- ❖ The OOB error is calculated by measuring the accuracy or loss between the predictions and the true labels.
- ❖ This error estimation provides an unbiased estimate of how well the Random Forest will perform on unseen data, as the OOB samples were not used in training.

EVALUATING MODEL PERFORMANCE

- ❖ The OOB error can serve as an **approximation of the model's performance on unseen data**, acting as an internal validation metric.
- ❖ It can be used to **compare different Random Forest models** with varying hyperparameters or feature configurations and aid in model selection.



SUMMARY

- Ensemble learning
- Bagging
- Boosting
- Stacking
- Cascading
- Random forests

