Weekly report of lessons

Name: Mayank Kumar
Roll No: 19CS30029
The week: 25-10-2021 to 31-10-2021

The topics covered:

Ensemble Learning: No learner perfect, Issue on combining learners, Diversification Techniques, Diversity vs Accuracy, Model combination schemes, Decision fusion, Voting, Classifier combination rules, Bayesian combination rule, Expectation, Bias, Variance, Error correcting output codes, Bagging (Bootstrap Aggregating), Boosting, Boosting by three weak learners in tandem, Adaboost, Adaboost.M1: the original algorithm, Mixture of experts, Stacked generalization, Cascading

Summary topic wise:

No learner perfect: No Free Lunch Theorem states that no single learning algorithm induces the most accurate learner and dictates a model with assumptions which might fail under certain circumstances. So, accuracy may be improved by a combination of multiple base-learners.

Issue on combining learners: It increases space and time complexity with no guarantee of increased accuracy. The generation of complementary base-learners and methods to combine their outputs for max accuracy are two key issues.

Diversification Techniques:
- Different Algorithms train different base-learners using parametric and non-parametric methods.
- Different Hyperparameters can be used for the same learning algorithm, like number of hidden units in a multilayer perceptron, k in k-nearest neighbour, initial weights of ANN, etc.
- Different Input Representatives can be used by integrating different types of sensors, modalities or measurements. Sensor fusion recognizes speech while Random subspace uses different feature subsets in learning.
- Different Training Sets are trained serially to give more emphasis on instances on which previous learners did not produce accurate results by boosting, casting or trying to generate complementary learners. It also partitions on the locality of training space known as the mixture of experts.

Diversity vs Accuracy: The base-learner to be simple is chosen if it operates marginally better than random guesses and if the final accuracy of combination is high. The base-learners to be diverse must be accurate on different instances, specializing in subdomains of the problem.

Model combination schemes:
- Multi-expert combination: Base-learners work in parallel. Global approach produces outputs based on the inputs and fusion of decisions, and Local approach selects a mixture of experts by looking at input (gating model) to produce outputs.
- Multi-stage combination: Base-learners sorted in complexity. Serial approach in which next base -learner is trained or tested on instances where previous base-learners were not accurate enough.

Decision fusion: For L learners, decision $d_j(x)$ for jth learner M_j and set of parameters Φ, final prediction of combined learners, $y = f(d_1(x), d_2(x), ..., d_L(x) \mid \Phi)$
For k outputs from each jth learner, ith class is assigned if y_i is maximum. $y_i = f(d_{i1}(x), d_{i2}(x), ..., d_{iL}(x) \mid \Phi)$
Voting: Fusion function $y_i = \Sigma_j w_j d_{jl}$ and $w_j \geq 0, \Sigma_j w_j = 1$

Classifier combination rules: Different types of fusion functions
- Sum or average, weighted average, max, min, median, product, etc
- Sum rule most widely used in practice
- Median rule more robust to outliers
- Min/max rules respectively pessimistic and optimistic
- Product rule empowers each learner veto power
- After the combination rules, y_i not necessarily sums up to 1

Bayesian combination rule: Weights approximate prior probabilities. For $w_j = P(M_j)$, $d_{ij} = P(C_i \mid x, M_j)$, choose models with high $P(M_j)$. If P(error) < 0.5 for a classifier, accuracy increases with the increase of number of classifiers by majority voting.

Expectation, bias and variance: Assuming d_j's are iids, for simple average ($w_j = 1/L$):
$E(y) = E(\frac{1}{L}\Sigma_j d_j) = E(d_j)$ and $var(y) = var(\frac{1}{L}\Sigma_j d_j) = \frac{1}{L} var(d_j)$

If they are not independent, $var(y) = var(\frac{1}{L}\Sigma_j d_j) = \frac{1}{L^2}(\Sigma_j var(d_j) + 2\Sigma_j \Sigma_{i<j} cov(d_i, d_j))$

Error correcting output codes: For each class, a set of binary classification tasks is predefined and coded in K✕L matrix for K classes and L classifiers, where each row represents the signature of a class and each column defines partitioning of classes into 2 sets labeled by either 1 or -1. The codes corresponding to class should follow error correcting codes principle by using hamming distance between any pair of them.

Bagging (Bootstrap Aggregating): Voting method with base-learners trained on slightly different training sets like similar structure and mathematical form but with different set of parameters and sampling with replacement. It is used for both classification and regression and the learning is unstable.
Boosting: Complementary base-learners are generated by training the next base learner from the mistakes of the previous learners. The original boosting algorithm is a combination of 3 weak learners.
Boosting by three weak learners in tandem: Training samples are randomly divided into sets X_1, X_2, X_3. d_1 is trained with X_1 and tested with X_2. d_2 is trained with X_2' which was formed with misclassified samples of X_2 and as many as correctly classified samples by d_1. X_3 is tested with d_1 and d_2, and d_3 is trained with instances disagreed by d_1 and d_2. For testing, accept the sample X having the same label from d_1 and d_2, else accept the result from d_3.
AdaBoost (Adaptive Boosting): Same training set is used with simple classifiers to avoid overfitting. Each classifier should perform with error rate<½. It combines arbitrary number of base-learners.
Adaboost.M1: the original algorithm: For training, at each iteration i, train with sample set and compute training error e of classification. Stop if e>0.5, else, include model d_i in the list and update the sampling prob. of each t^th training sample.
For testing, given x, calculate $d_j(x)$ for j = 1 to L, and calculate outputs $y_i = \Sigma_j log(\frac{1}{w^{(j)}} d_j(x))$for i = 1 to K. Assign the class with maximum y.

Mixture of experts: Weights vary depending upon inputs. Ideally local experts would have weight close to 1 and rest close to 0.
Stacked generalization: It could be any general functional form with parameters Φ, which are also learners. It could be a multilayer perceptron.
Cascading: Base learners are ordered in terms of complexity and each learner produces output (y) with a confidence w. The next base-learner is used if the previous learners' decisions lack confidence.

Concepts challenging to comprehend:
None

Interesting and exciting concepts:
None

Concepts not understood:
None

Any novel idea of yours out of the lessons:
One of the key issues on combining learners was combining their outputs for maximum accuracy. In such cases, directly concatenating them is definitely not a good idea, and might give poor accuracy. Instead, what we can do is perform some combined dimensionality reduction for these outputs. To do the same, let us consider that the model assumes a set of latent factors that parallelly generate multiple outputs, and from these observed outputs, we move to the latent space for categorization of samples. This approach will be considering all the outputs properly while making a prediction for a sample.
