Machine Learning (CS60050) – Weekly Report

Kaushal Banthia (19CS10039)

Week 11: 27th – 29th October, 2021

Topics Covered:

- No learner perfect and Issues on combining learners
- Diversification and different techniques for it
- Model combination schemes
- Decision fusion
- Classifier combination rules
- Expectation, Bias and Variance
- Error correcting output codes (ECOC)
- Bagging and Boosting
- AdaBoost (Adaptive Boosting)
- Mixture of experts
- Stacked generalization and Cascading

Summary (Topic Wise):

- <u>No learner perfect and Issues on combining learners</u>

    - ➢ No Free Lunch Theorem: No single learning algorithm in any domain always induces the most accurate learner. Each learning algorithm dictates a certain model that comes with a set of assumptions. This inductive bias leads to error if the assumptions do not hold for the data.
    - ➢ With finite data, each algorithm converges to a different solution and fails under different circumstances.
    - ➢ A suitable combination of multiple base – learners should improve the accuracy.
    - ➢ But there are some issues with combining the learners:
        - Overheads of combining multiple learners - Increased space and time complexities. Also, the model combination may not increase the accuracy.
        - Thus, two key issues arise, namely, how do we generate base-learners that
        - complement each other and how do we combine the outputs of base-learners for maximum accuracy?

- <u>Diversification and different techniques for it</u>

    - ➢ We can use different techniques to diversify our training:
        - We can use different learning algorithms to train different base-learners using parametric and non-parametric methods.
        - We can use different Hyperparameters, with the same learning algorithm but use it with different hyperparameters.
        - Different Input Representations, like integrating different types of sensors/ measurements/ modalities.
            - ➢ Sensor fusion: Audio and Video of lip movement to recognize speech.
            - ➢ Random sub-space: Use different feature subsets in learning, as different learners will look from different points. Also, it reduces the curse of dimensionality.

- Different Training Sets can be used.
    - Different subsets of training sample (Bagging). They are trained serially so that instances on which the preceding base-learners are not accurate are given more emphasis in training later base-learners (boosting and cascading). This actively tries to generate complementary learners, instead of leaving this to chance.
    - Partitioning on locality of training space: Each base-learner trained on instances in a certain local part of the input space.

- Once we have the power to diversify, we face the question of diversity vs accuracy.
    - The base-learner are to be simple if it is not chosen for its accuracy. It is enough if it performs with an error rate less than 50% for binary classification. (Operates marginally better than random guesses). Final accuracy of combination should be high.
    - The base-learners are to be diverse if we require accuracy on different instances, specializing in subdomains of the problem.

- <u>Model combination schemes</u>

    - Multi-expert combination:
        - All the base-learners work in parallel.
        - A global approach would be, where all learners produce output given an input and a fusion of decisions (Voting, Stacking).
        - The local approach would be where the selected learners (mixture of experts) produce an output. It is a gating model for selecting experts by looking at the input.
    - Multi-stage combination:
        - It is a serial approach, where the next base-learner is trained with or tested on only those instances where the previous base-learners are not accurate enough.
        - Base-learners are sorted in complexity. Complex learners used if the preceding simpler learners are not confident enough (Cascading)

- <u>Decision fusion</u>

    - If there are L learners and $d_j(x)$ is the decision for the jth learner M_j, then $y = f(d_1(x), d_2(x), \dots , d_L(x)|\phi)$, where ϕ is the set of parameters and y is the final prediction of the combined learners.
    - For k outputs from each jth learner, $d_{ij},\ i = 1, 2, \dots k \ and \ j = 1, 2, .. L$
    - $y_i \ = f(d_{i1}\ (x), d_{i2}(x), \dots, d_{iL}(x)|\phi), i = 1, 2, .., k$
    - Predict on y_i, by assigning the ith class if y_i is maximum.

- <u>Classifier combination rules</u>

    - There are different types of fusion functions.
    - The sum rule is most widely used in practice, but the median rule is more robust to outliers. The minimum and the maximum rules are pessimistic and optimistic respectively. The product rule empowers each learner with a veto power in the 0/1 decision cases.
    - Note that after the combination rules, y_i need not necessarily sum up to 1.

- Bayesian combination rule: Weights approximating prior probabilities of models.
- Let $w_j = P(M_j),\ d_{ij} = P(C_i|x, M_j)$. Then $P(C_i|x) = \sum_{M_j} P(M_j)P(C_i|x, M_j)$
- Instead of all the models in the space, choose only those that have a high $P(M_j)$
- For each classifier if $P(error) < \frac{1}{2}$, an increase in the number of classifiers, lead to an increase in the accuracy by majority voting.

- <u>Expectation, Bias and Variance</u>

  - Assume the d_j's are iids with the expected value $E(d_j)$ and variance $var(d_j)$
  - For simple average $\left(w_j = \frac{l}{L}\right)$, $E(y) = E\left(\frac{1}{L}\sum_j d_j\right) = \frac{1}{L}LE(d_j) = E(d_j)$
  - $var(y) = var\left(\frac{1}{L}\sum_j d_j\right) = \frac{1}{L^2} * L * var(d_j) = \frac{1}{L} * var(d_j)$
  - If they are not independent, then,
    $var(y) = var\left(\frac{1}{L}\sum_j d_j\right) = \frac{1}{L^2}(var(\sum_j d_j) = \frac{1}{L^2}(\sum_j var(d_j) + 2\sum_j \sum_{i<j} cov(d_i, d_j))$
  - Negative correlation may improve variance, but it is difficult to satisfy both accuracy of more than 50% which is negatively correlated.

- <u>Error correcting output codes (ECOC)</u>

  - For each class, a set of binary classification tasks is predefined, which is coded in a K x L matrix for K classes and L classifiers.
  - Each row represents the signature of a class and each column defines partitioning of classes into two sets labeled by either $+1$ or -1
  - The codes corresponding to class should follow error correcting codes principle by keeping sufficient distance (Hamming distance) between any pair of them.

- <u>Bagging and Boosting</u>

  - Bagging (Bootstrap Aggregating) is a voting method, where each base-learner is trained over slightly different training sets of similar structure and mathematical form, but with different sets of parameters. The sampling is done with replacement. Thus, it is possible to have repeated samples in the training set.
  - Used for both classifications and regression.
  - For regression, median is used to make the estimation more robust to outliers.
  - If a small change in the data causes a large variation in the model, then the learning is unstable (This is the case for decision trees and ANNs).
  - In boosting, we generate complementary base learners. We train the next base learner from the mistakes of the previous learners. In bagging, this training is left to the factor of chance and the instability of the training algorithm.
  - Boosting by three weak learners in tandem (Original boosting algorithm)
    1. Randomly divide training samples in 3 sets, X_1, X_2 & X_3
    2. Train d_1 with X_1 and test d_1 with X_2
    3. Form a training set X_2' for training d_2 with misclassified samples of X_2 and as many correctly classified samples by d1
    4. Train d_2 with X_2', and test X_3 with d_1 and d_2
    5. Train d_3 with instances disagreed by d_1 and d_2.
    6. Testing: If a sample X has same labels by d_1 and d_2, accept it, else accept the result from d_3

- AdaBoost (Adaptive Boosting)

  - Uses the same training set over and over. The training set need not be large.
  - The classifiers should be simple so that they do not overfit and each should perform with an error rate < $\frac{1}{2}$
  - Combines an arbitrary number of base learners, not just 3
  - Many variants exist as samples are drawn randomly to form a training set, each with varying probability. Easier to classify, smaller the probability.
  - Algorithm (Original):
    - Training:
      1. At each iteration i, train with the sample set and compute the training error e of classification.
      2. If $e > \frac{1}{2}$, stop, as no more classifiers required in the set.
      3. Else include the model d_i in the list and update the sampling prob. of each t^th training sample, by decreasing which are classified, correctly with the weight $w^{(i)}$: $p^{(t)} = w^{(i)} * p^{(t)}$, where $w^{(i)} = \frac{e}{1-e}$
      4. Normalize probabilities of samples at each iteration.
      5. $log\left(\frac{l}{w^{(i)}}\right)$ is taken as the weight of the decision from that model during voting.
    - Testing
      1. Given x calculate $d_j(x),\ j = 1,2,..,L$
      2. Calculate class outputs $y_i, i = 1,2,..,K$
         $$y_i = \sum_j \log\left(\frac{1}{w^{(j)}}\right) d_j(x)$$
      3. Assign the class with maximum y
    - Use simple classifiers so that error is not low.
    - If decision trees are used, then they are grown up to one or two levels (Decision stump).
    - Linear discriminant classifiers not useful due to low variance.

- Mixture of experts

  - In voting, weights are fixed for each classifier (expert).
  - In mixture of experts, depending upon inputs these weights would vary. Ideally local experts (on the locality of input) would have a weight close to 1 and the rest will have a weight close to 0
  - Voting is done by gating system.
  - Final classification score for each class is the weighted mean of votes.

- Stacked generalization and Cascading

  - Instead of linear combination, it could be any general functional forms with parameters ϕ, which are also learned → $f(d_1, d_2, \ldots, d_L|\phi)$
  - It could be a multilayer perceptron with input d_j's and output y.
  - Cascading involves ordering the Base learners (d_i's) in terms of complexity.
  - Each learner produces an output (y) with a confidence (w). The next base learner is used if the previous learners' decisions lack confidence.

Concepts Challenging to Comprehend: None yet.

Interesting and Exciting Concepts: Bagging and Boosting

Concepts not understood: None yet.

A novel idea: In Bagging (Bootstrap Aggregating), instead of sampling being done with replacement, we could do the sampling without replacement. This would mean that the datasets that are now created are ones that do not have any repeated data points in them. This would help the model, as there would be no redundancy in the model. Thus, we could reduce the size of the dataset, and help train the model faster and better, if we sample without replacement!