NUTAN MAHARASHTRA VIDYA PRASARAK MANDAL'S

NUTAN COLLEGE OF ENGINEERING & RESEARCH (NCER)

Department of Computer Science & Engineering

BTCOC503

Machine Learning

Teaching Notes

Lecture Number	Topic to be covered
1	**Unit 5: Computational learning theory (06Hrs)** ➢ Computational learning theory
2	➢ PAC learning model
3	➢ Sample complexity
4	➢ VC Dimension
5	➢ Ensemble learning
6	➢ Type of Ensemble Learning Technique

: Submitted by:
Prof. S. B. Mehta

Unit 5: Computational learning theory

Computational learning theory:

- Computational Learning Theory (CoLT) is a field of AI research studying the design of machine learning algorithms to determine what sorts of problems are "learnable." The ultimate goals are to understand the theoretical underpinnings of deep learning programs, what makes them work or not, while improving accuracy and efficiency.
- Computational Learning Theory (CoLT) is mathematical and theoretical field related to analysis of Machine Learning algorithms.
- Computational learning theory is an investigation of theoretical aspects of machine learning, of what can and cannot be learned from data. In particular we are interested in the computational efficiency and limitations of learning from large (and small) amounts of data as well as in understanding the theoretical underpinnings of using unlabeled data. Computational learning theory is a multidisciplinary area bringing together techniques and approaches of computer science, statistics and applied mathematics.
- This research field merges many disciplines, such as probability theory, statistics, programming optimization, information theory, calculus and geometry.

PAC Model:

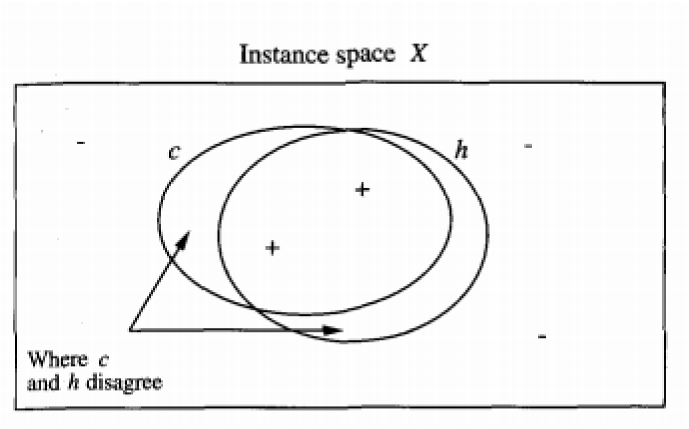
- PAC (Probably Approximately Correct) learning is a theoretical framework for analyzing machine learning algorithms as well as different problem types. It was developed by Leslie Valiant and is part of a field called computational learning theory.
- In computational learning theory, probably approximately correct (PAC) learning is a framework for mathematical analysis of machine learning. It was proposed in 1984 by Leslie Valiant.
- Probably approximately correct (PAC) learning is a theoretical framework for analyzing the generalization error of a learning algorithm in terms of its error on a training set and some measure of complexity. The goal is typically to show that an algorithm achieves low generalization error with high probability.
- PAC Model: is framework for mathematical analysis of ML algorithms.
- Goals of PAC :with high probability("probably") the selected hypothesis will have low error ("Approximately correct").

Assume that there is no error (noise) on data

P-Probably

A: Approximately

C:-Correct



- Definition: The true error (denoted errorD(h)) of hypothesis h with respect to target concept c and distribution D is the probability that h will misclassify an instance drawn at random according to D.
- $$error_{\mathcal{D}}(h) \equiv \Pr_{x \in \mathcal{D}} [c(x) \neq h(x)]$$
- Here the notation Pr indicates that the probability is taken over the instance x€D distribution V

- We begin by specifying the problem setting that defines the PAC learning model, then consider the questions of how many training examples and how much computation are required in order to learn various classes of target functions within this PAC model. For i the sake of simplicity, we restrict the discussion to the case of learning Boolean valued concepts from noise-free training data. However, many of the results can be extended to the more general scenario of learning real-valued target functions
- We cannot require the model to make perfect prediction every time. Instead, we hope the event that deviation is less than ε happens with high probability. In other words, machine learning can produce a good model with high probability.
- We can do two things by using PAC learning model. On the one hand, given the training data size, how accurate the model can reach. On the other hand, given the model’s accuracy requirement, what sample complexity do we need. If the data size cannot be required, then the problem is not learnable.

Sample complexity:-

In machine learning, model complexity often refers to the number of features or terms included in a given predictive model, as well as whether the chosen model is linear, nonlinear, and so on. It can also refer to the algorithmic learning complexity or computational complexity.

In machine learning, model complexity often refers to the number of features or terms included in a given predictive model, as well as whether the chosen model is linear, nonlinear, and so on. It can also refer to the algorithmic learning complexity or Computational complexity

Sample Complexity: How many training examples are needed for a learner to converge (with high probability) to a successful hypothesis?

Computational Complexity: How much computational effort is needed for a learner to converge (with high probability) to a successful hypothesis?

Mistake Bound: How many training examples will the learner misclassify before converging to a successful hypothesis.

VC Dimension:-

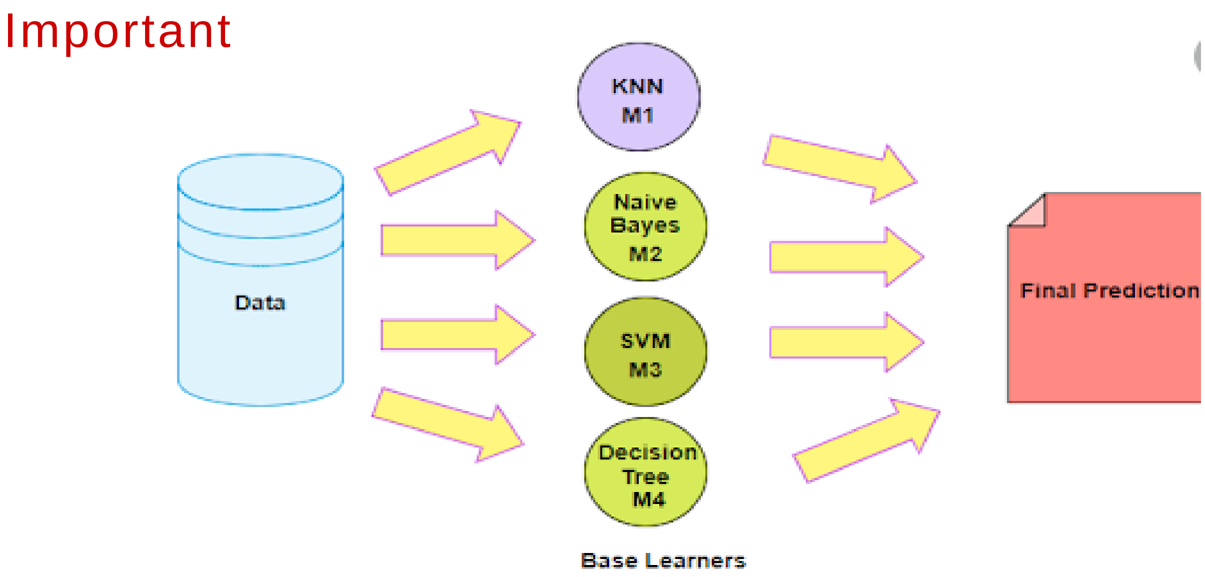
- In Vapnik–Chervonenkis theory, the Vapnik–Chervonenkis (VC) dimension is a measure of the capacity (complexity, expressive power, richness, or flexibility) of a set of functions that can be learned by a statistical binary classification algorithm. ... A much simpler alternative is to threshold a linear function.
- To make this notion more precise, we first define the notion of shattering a set of instances
- Shattering a set of points In order to shatter a configuration of points, the classifier must be able to, for every possible assignment of positive and negative for the points, perfectly partition the plane such that the positive points are separated from the negative points.

- There are a couple of theorems by Vapnik that establish that if some hypothesis has a finite VC dimension then the amount of data needed for the algorithm to learn is linear to the number of parameters it uses.
- This theorem is, in my opinion, the most important piece of theory in Machine Learning because it shows that an algorithm can learn from a training set and generalize to unseen data with an error that is bounded as long as the amount of data in the training set is linear to the VC dimension of the algorithm.

So using VC dimension we can:

- Prove that a Machine Learning algorithm can learn from the training set and generalize to unseen data.
- Prove that as more data is added the error of the algorithm decreases.
- Prove that the more complex the algorithm is the more data we need for it to generalize correctly.

Ensemble learning:-



- Ensemble models in machine learning combine the decisions from multiple models to improve the overall performance .Ensemble learning is a machine learning technique in which several models are combined to build a more powerful model. The ensemble is primarily used to improve the performance of the model. That is why ensemble methods are used in various machine learning competitions such as KDD, kaggle, etc. and also used in real-world problems.
- The main causes of error in learning models are due to noise, bias and variance. Ensemble methods help to minimize these factors. These methods are designed to improve the stability and the accuracy of Machine Learning algorithms.\

Example:

- Let's suppose that you have developed a health and fitness app. Before making it public, you wish to receive critical feedback to close down the potential loopholes, if any. You can resort to one of the following methods, read and decide which method is the best:
  1. You can take the opinion of your spouse or your closest friends.
  2. You can ask a bunch of your friends and office colleagues.
  3. You can launch a beta version of the app and receive feedback from the web development community and non-biased users.
- No brownie points for guessing the answer :D Yes, of course we will roll with the third option.
- Now, pause and think what you just did. You took multiple opinions from a large enough bunch of people and then made an informed decision based on them. This is what Ensemble methods also do.

Ensemble Learning Technique:-

1. Simple Technique
2. Bagging (Bootstrap AGGregatING)
3. Boosting

1.Simple Technique:-

- Taking the mode of the results

In this technique, multiple models are used to make predictions for each data point. The predictions by each model are considered as a separate vote. The prediction which we get from the majority of the models is used as the final prediction.

- Taking the average of the results

In this technique, we take an average of predictions from all the models and use it to make the final prediction.

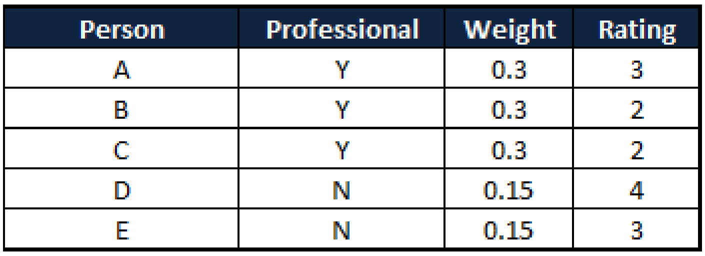
AVERAGE= sum(Rating*Number of people)/Total number of people= (1*5)+(2*13)+(3*45)+(4*7)+(5*2)/72 = 2.833 =Rounded to nearest integer would be 3

- Taking weighted average of the results

This is an extension of the averaging method. All models are assigned different weights defining the importance of each model for prediction. For instance, if about 25 of your responders are professional app developers, while others have no prior experience in this field, then the answers by these 25 people are given more importance as compared to the other people.

For example: For posterity, I am trimming down the scale of the example to 5 people

WEIGHTED AVERAGE= (0.3*3)+(0.3*2)+(0.3*2)+(0.15*4)+(0.15*3) =3.15 = rounded to nearest integer would give us 3

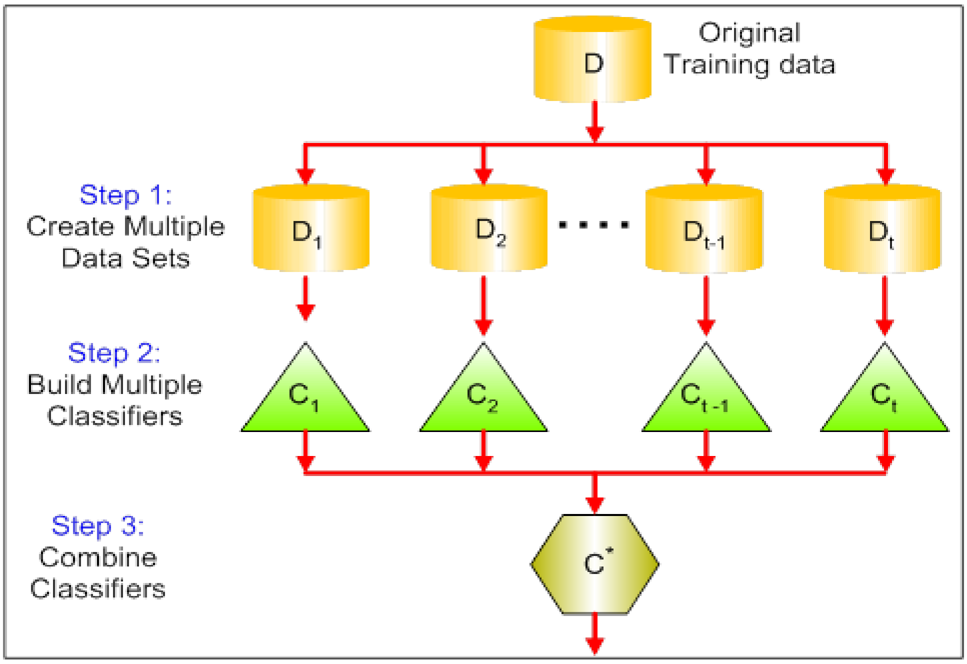


Person	Professional	Weight	Rating
A	Y	0.3	3
B	Y	0.3	2
C	Y	0.3	2
D	N	0.15	4
E	N	0.15	3

2. Bagging (Bootstrap AGGregatING) :

- Bagging tries to implement similar learners on small sample populations and then takes a mean of all the predictions. In generalized bagging, you can use different learners on different population. As you can expect this helps us to reduce the variance error.
- Bagging is a technique of combining the results from the various models to get the final predictions (maximum vote in classification and the average value in regression).
- Bagging is a way to decrease the variance in the prediction by generating additional data for

training from dataset using combinations with repetitions to produce multi-sets of the original data.

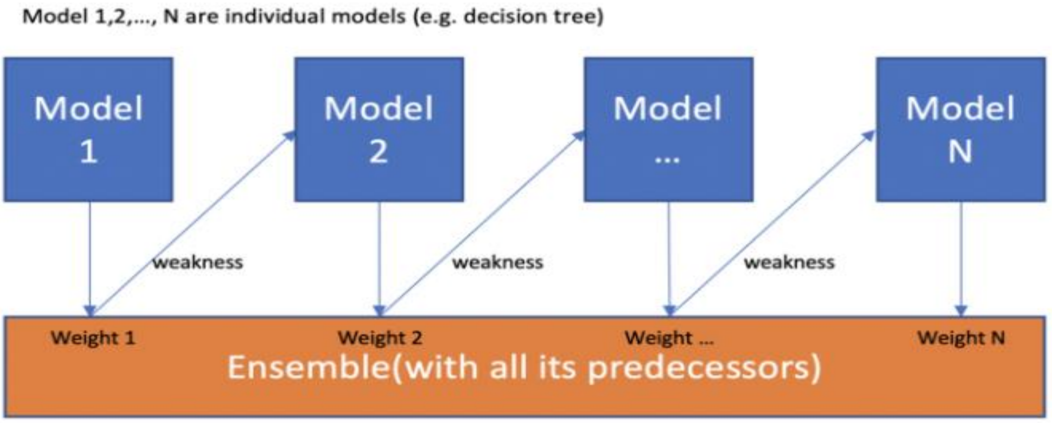

D
Original Training data
Step 1: Create Multiple Data Sets
D_1
D_2
D_{t-1}
D_t
Step 2: Build Multiple Classifiers
C_1
C_2
C_{t-1}
C_t
Step 3: Combine Classifiers
C^*

- Bagging is one of the Ensemble construction techniques which is also known as Bootstrap Aggregation. Bootstrap establishes the foundation of Bagging technique. Bootstrap is a sampling technique in which we select “n” observations out of a population of “n” observations. But the selection is entirely random, i.e., each observation can be chosen from the original population so that each observation is equally likely to be selected in each iteration of the bootstrapping process. After the bootstrapped samples are formed, separate models are trained with the bootstrapped samples. In real experiments, the bootstrapped samples are drawn from the training set, and the sub-models are tested using the testing set. The final output prediction is combined across the projections of all the sub-models.
- Bootstrap Aggregating is an ensemble method. First, we create random samples of the training data set with replacment (sub sets of training data set). Then, we build a model (classifier or Decision tree) for each sample. Finally, results of these multiple models are combined using average or majority voting.
- As each model is exposed to a different subset of data and we use their collective output at the end, so we are making sure that problem of overfitting is taken care of by not clinging too closely to our training data set. Thus, Bagging helps us to reduce the variance error.
- Combinations of multiple models decreases variance, especially in the case of unstable models, and may produce a more reliable prediction than a single model.
- Random forest technique actually uses this concept but it goes a step ahead to further reduce the variance by randomly choosing a subset of features as well for each bootstrapped sample to make the splits while training (My next post will detail all about Random forest technique)

3. Boosting:

- Boosting in general decreases the bias error and builds strong predictive models. Boosting has shown better predictive accuracy than bagging, but it also tends to over-fit the training data as well).
- Boosting is a general ensemble method that creates a strong classifier from a number of weak classifiers. This is done by building a model from the training data, then creating a second model that attempts to correct the errors from the first model.
- Boosting is an iterative technique which adjust the weight of an observation based on the last

classification. If an observation was classified incorrectly, it tries to increase the weight of this observation and vice versa. Boosting in general decreases the bias error and builds strong predictive models. However, they may sometimes over fit on the training data.


Model 1,2,..., N are individual models (e.g. decision tree)
Model 1
Model 2
Model ...
Model N
weakness
weakness
weakness
Weight 1
Weight 2
Weight ...
Weight N
Ensemble(with all its predecessors)

One is weak, together is strong, learning from past is the best

- Boosting is an ensemble modeling technique which attempts to build a strong classifier from the number of weak classifiers. It is done building a model by using weak models in series. Firstly, a model is built from the training data. Then the second model is built which tries to correct the errors present in the first model. This procedure is continued and models are added until either the complete training data set is predicted correctly or the maximum number of models are added.
- AdaBoost was the first really successful boosting algorithm developed for the purpose of binary classification. AdaBoost is short for Adaptive Boosting and is a very popular boosting technique which combines multiple “weak classifiers” into a single “strong classifier”. It was formulated by Yoav Freund and Robert Schapire. They also won the 2003 Gödel Prize for their work.

How is a Boosting Model Trained?

The training process depends on the Boosting algorithm that we are using (Adaboost vs LigthGBM vs XGBoost…), but generally it follows this pattern:

1. **All the data samples start with the same weights.** These samples are used to train an individual model (a Decision Tree lets say).
2. The prediction error for each sample is calculated, increasing the **weights of those samples** which have had a greater error, to make them more important for the training of following individual model.
3. Depending on how well this **individual model** did on its predictions, it **gets assigned an importance/weight or amount of say**. A model that outputs very good predictions will have a high amount of say in the final decision.
4. **The weighted data is passed on to the posterior model,** and 2) and 3) are repeated.
5. Number 4) is repeated until we have reached an certain number of models or until the error is bellow a certain threshold.

(Subject In-charge)

(Prof.S.B.Mehta)