Machine Learning (CS60050) – Weekly Report

Kaushal Banthia (19CS10039)

Week 6: 15th – 17th September, 2021

Topics Covered:

- Minimum description length principle in Bayesian learning & Optimal Classifier
- Gibbs Algorithm
- Discriminant Functions
- Naïve Bayes Classifier
- Bayesian Network
- Conditional Independence
- Computation on Bayesian Network
- Inference through Bayesian Networks
- Bayesian Decision making (Losses and risks)
- Mining Association Rules
- Apriori algorithm
- Association and causality

Summary (Topic Wise):

- <u>Minimum description length principle in Bayesian learning & Optimal Classifier</u>

  - $h_{MAP} = argmax_{h \epsilon H} P(D|h) * P(h)$
  - $h_{MAP} = argmax_{h \epsilon H} (log_2 P(D|h) + log_2 P(h))$
  - $h_{MAP} = argmax_{h \epsilon H} (-log_2 P(D|h) - log_2 P(h))$
  - Information Theory states that optimal encoding length of a message of probability p is $-log_2 p$
  - We need to learn a target function of a classifier. $f: X \rightarrow V, where\ V = \{v_1, v_2, \dots, v_k\}$ is a set of possible class labels.
  - For a hypothesis space H with $h: X \rightarrow V$, $h_{MAP} = argmax_h \{P(h|D)\}$ may not be optimal. Thus, we consider all h in H for decision making, weighted by their posterior. $c(x) = argmax_{v \epsilon V} \{(\sum_h (P(v|h) * P(h|D))\}$. Here, c is learnt as an optimal classifier.

- <u>Gibbs Algorithm</u>

  - Instead of enumerating exhaustively, choose a hypothesis h randomly for an instance x with posterior distribution $P(h|D)$ and apply h on the instance x.
  - Performs sub-optimally and the expected error is at most twice of the optimal error when the prior has uniform distribution.

- <u>Discriminant Functions</u>

  - Bayesian classifiers can be expressed in the framework of classification based on a set of discriminant functions $g_i(x)$, with the rule that assign c_i if $g_i(x) > g_k(x)\ \forall k, except\ i$
  - This has the following computations involved: Assign c_i to X iff the probability $P(c_i|X)$ is the highest among all the $P(c_k|X)$ for all the k classes.

- $i = argmax_k\{P(c_k|X\} = argmax_k\{P(X|c_k) * P(c_k)\}$
- We face the following challenges in this computation.
    - Prior knowledge of probabilities of classes
    - Probability distributions in multidimensional feature spaces

- <u>Naïve Bayes Classifier</u>

    - Works on a simplified assumption that the attributes are conditionally independent (i.e., no dependence relation between the attributes).
    - Likelihood= $P(X|c_i) = \prod_{k=1}^{n} P(x_k|c_i)$
    - This causes a significant reduction in the cost of computation as it requires only the class distributions.
    - It is convenient to estimate $P(x_i|c_k)$
        - For a categorical or discrete variable: It equals the fraction of times the value occurred in a class.
        - For a continuous variable: It may use parametric modeling of Gaussian distribution. $g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{(x-\mu)^2}{2\sigma^2}}$ and $P(X|c_i) = g(x_k, \mu_{c_i}, \sigma_{c_i})$
    - To avoid zero probability, all the conditional probabilities should be non-zero, else likelihood becomes zero. Thus, we use the Laplacian Estimator / correction.
    - Pros:
        - Easy to implement
        - Good results obtained in most of the cases.
    - Cons:
        - There is an assumption of class conditional independence. Leads to a loss of accuracy.
        - In real life, dependencies exist among variables and cannot be modelled by the Naïve Bayes Classifier.

- <u>Bayesian Network</u>

    - It is a more general framework, for modelling conditional dependencies and represents the interaction between variables in a graph.
    - The graph is composed of nodes and the arcs between the nodes.
        - Node: Random variable X with probability of the random variable P(X).
        - A directed arc from X to Y: X influences Y with probability P(Y|X).
        - A directed acyclic graph (DAG) with no cycle.
        - Topology called structure which has P(X) and P(Y|X) as parameters.
    - We can form the graph by adding the nodes and arcs between two nodes, if they are not independent. (X and Y are independent, if they are not conditionally dependent), i.e., $P(Y|X) = P(Y),\ P(X|Y) = P(X)$ and $P(X,Y) = P(X)P(Y)$

- <u>Conditional Independence</u>

    - Conditional independence between X and Y, given the occurrence of a third event (Z) happens when $P(X,Y|Z) = P(X|Z)P(Y|Z)$
    - It can also be written as $P(X|Z) = P(X|Y,Z)$
    - When its value is known, Z blocks the path from Y to X. Thus, if Z is removed, there is no path between Y to X.

- For specifying joint probabilities, there is no need to specify at all the possible data points. This causes significant savings for a large network.

- Computation on Bayesian Network

  - Given the value of any set of variables as evidence, we need to infer the probabilities of any other set of variables.
  - Thus, we make a probabilistic database, that is a machine that can answer queries regarding the values of random variables.
  - This makes the difference between unsupervised and supervised learning blurry.

- Inference through Bayesian Networks

  - We have $P(X_1, X_2, \dots, X_d) = \prod_{i=1}^{d} P(X_i | parent\ of\ X_i)$
  - Given any subset of X_i, calculate the probability distribution of some other subset of X_i by marginalizing over the joint. There is an exponential number of joint probability combinations. This does not exploit implied independencies. Also, there is redundancy of computing joint probability of the same subsets. This is because of efficient computation through belief propagation. It can also accommodate hidden variables (the values that are not known, but estimated from dependency of observed variables).
  - Special case for Naïve Bayes Classifier:

$$P(x_1, x_2, \dots, x_d, C) = P(C) * P(x_1|C) * P(x_2|C) \dots P(x_d|C)$$

$$P(C|x) = \frac{P(C) * P(x|C)}{P(x)} \ (Apply\ Bayesian\ Classification\ Rule)$$

$$P(x|C) = P(x_1|C) * P(x_2|C) \dots P(x_d|C)$$

- Bayesian Decision making (Losses and risks)

  - If a_i is the i^{th} action (assigns x to class C_i) and l_{ik} is the loss due to a_i, if $x \in C_k$, then the expected risk of taking the action a_i is $R(a_i|x) = \sum_k l_{ik} * P(C_k|x)$. Thus, we choose that a_i, which minimizes $R(a_i|x)$
  - This can be generalized according to utility theory, as follows:
    - Instead of loss consider gain U_{ik} for taking action a_i at state k (here given by class C_k).
    - Expected utility: $EU(a_i|x) = \sum_k U_{ik} P(C_k|x)$
    - Choose a_i if $EU(a_i|x)$ is maximum out of all actions a_i's.

- Mining Association Rules

  - An association rule is an implication $X \rightarrow Y$, where X is the antecedent and Y is the consequent.
  - We have three useful measures:
    - $Support\ (X, Y)$: $P(X, Y)$ (Indicates statistical significance and should be of considerable numbers as insignificant support with a high confidence is meaningless)
    - $Confidence(X \rightarrow Y)$: $P(Y|X) = \frac{P(X,Y)}{P(X)}$ (Indicates the strength of the rule and should be high, ideally close to 1. Also, it is significantly higher than P(Y))

- $Lift(X,Y) = \frac{P(X,Y)}{P(X).P(Y)} = \frac{P(Y|X)}{P(Y)}$ (For independent X and Y, Lift should be close to 1, otherwise it shows dependency. If Lift > 1, then most likely X makes Y, else (for Lift<1), Y makes X.)

- Apriori algorithm

  - To get association rules with high support and confidence from a database. It is possible to generalize association among more than 2 variables. It involves 2 steps.
    - Step 1: Finding frequent item sets (those which have enough support).
    - Step 2: Converting them to rules with enough confidence, by splitting the items into two, as items in the antecedent and items in the consequent.
  - Step 1 is done as follows:
    - Start searching for combination with lower cardinality.
    - Remove supersets in the combinations, which are not in the list of lower cardinality sets. If X is not frequent, do not search for any combination with X.
    - Requires (n+1) passes for searching largest n-itemset together.
  - Step 2 is done as follows:
    - For every itemset, split keeping all but 1 item in antecedent and 1 item in consequent. Remove those rules, which fail the test of confidence.
    - In every pass, reduce antecedent part and increase consequent part, as rules with larger consequent part are more useful.

- Association and causality

  - $X \rightarrow Y$ indicates association and not causality.
  - There may be hidden variables acting in the process which are not identified.

Concepts Challenging to Comprehend: None yet.

Interesting and Exciting Concepts: Naïve Bayes Classifier & Bayesian Network

Concepts not understood: None yet.

A novel idea: For calculating $Lift(X,Y) = \frac{P(X,Y)}{P(X).P(Y)} = \frac{P(Y|X)}{P(Y)}$, we need to calculate $Confidence(X \rightarrow Y)$: $P(Y|X) = \frac{P(X,Y)}{P(X)}$, for which, in turn, we have to calculate $Support\ (X,Y)$: $P(X,Y)$. In this case, the metrics denote different things, but still depend on each other. It is also said that an insignificant support with a high confidence is meaningless. But here, support and confidence are related to each other. Although the factor of P(Y) may offer some independence, but it would be better if we could use some other metrics that could be calculated without depending on each other.