MACHINE LEARNING FOR DATA STREAMS

DECISION TREES

Note: A number between parenthesis corresponds to a certain survey:

1. **Mining High-Speed Data Streams (2000)_VFDT:** **No aparece en el survey (6)**
    - (1) Pages 3,4
        - This method essentially subsamples the data in order to achieve scalability in the construction of the decision tree. The idea is to show that the **entire decision tree constructed would be the same as the one built on sub-sampled data with high probability**.
        - The idea is to determine a random sample of sufficient size so that the tree constructed on the sample is the same as that constructed on the entire data set. The **Hoeffding bound** is used to show that the decision tree on the sub-sampled tree would make the same split as on the full stream with high probability. This approach can be used with a variety of criteria such as the gini-index, or information gain.
        - The number of examples required to produce the same split as the original data (with high probability) is determined. The Hoeffding bound is used to determine the number of relevant examples, so that this probabilistic guarantee may be achieved. If all splits in the decision tree are the same, then the same decisión tree will be created.
        - The Hoeffding tree can also be **applied to data streams**, by building the tree incrementally, as more examples stream in, from the higher levels to the lower levels. At any given node, one needs to wait until **enough tuples are available in order to make decisions about lower levels**. The memory requirements are modest, because only the counts of the different discrete values of the attributes (over different classes) need to be maintained in order to make Split decisions.
        - The **VFDT algorithm** is also based on the Hoeffding tree algorithm, though it makes a number of modifications. Specifically, it is **more aggressive about making choices in the tie breaking** of different attributes for splits. It also allows the deactivation of less promising leaf nodes. It is generally more memory efficient, because of these optimizations.
        - The original VFDT method is not designed for cases where the stream is evolving over time.
    - http://www.otnira.com/2013/03/28/hoeffding-tree-for-streaming-classification/

        - Hoeffding bound gives certain level of confidence on the best attribute to split the tree, hence we can build the model based on certain number of instances that we have seen.
        - Hoeffding-tree, which is a new decision-tree learning method for streaming that solves these following challenges:
            - Uncertainty in learning time. Learning in Hoeffding tree is **constant time per example** (instance) and this means Hoeffding tree is suitable for mining data streaming.
            - The resulting trees are nearly identical with trees built by conventional batch learner, given enough example to train the and build the Hoeffding tres
        - To achieve the streaming classification characteristics, the authors introduce Hoeffding bound to decide **how many examples of instances needed to achieve certain level of confidence** (i.e. the chosen instance attribute using the bound is the close to the attribute chosen when infinite examples are presented into the classifier).
        - What makes Hoeffding bound attractive is its ability to give the same results **regardless the probability distribution generating the observations**. However, the number of observations needed to reach certain values of \delta and \epsilon are different across probability distributions.
        - With probability 1-\delta , one attribute is superior compared to others when observed difference of information gain is greater than \epsilon.
        - The authors implemented the Hoeffding tree algorithm into Very Fast Decision Tree learner (VFDT) which includes some enhancements for practical use, such as node-limiting strategy, introduction of as tie breaking parameter, grace period of bound calculation, poor attributes removal, fast initialization by using conventional RAM-based learner and ability to rescan previously-seen examples when data rate is slow.

2. **Mining Time-Changing Data Streams (2001)_CVFDT: No aparece en los surveys (5), (6) y (7)**
    - (1) Pages 4,5
        - The original VFDT method is not designed for cases where the stream is evolving over time. The work in [47] extends this method to the case of concept-drifting data streams. This method is referred to as **CVFDT**. CVFDT incorporates two main ideas in order to address the additional challenges of drift:
            - A sliding window of training items is used to limit the impact of historical behavior.
            - Alternate subtrees at each internal node i are constructed.
        - Because of the sliding window approach, the main issue here is the update of the attribute frequency statistics at the nodes, as the

sliding window moves forward. For the incoming items, their statistics are added to the attribute frequencies in the current window, and the statistics of the expiring items at the other end of the window are decreased. Therefore, **when these statistics are updated, some nodes may no longer meet the Hoeffding bound, and somehow need to be replaced**.

3. **Efficient Decision Tree Construction on Streaming Data (2003): No aparece en los surveys (2), (3), (4), (5), (6) y (7)**
   - (1) Page 4
     - One of the major challenges of the VFDT family of methods is that it is naturally suited to categorical data. This is a natural derivative of the fact that decision tree splits implicitly asume categorical data. Furthermore, discretization of numerical to categorical data is often done offline in order to ensure good distribution of records into the discretized intervals. Of course, it is always possible to test all possible split points, while dealing with numerical data, but the number of posible split points may be very large, when the data is numerical. The bounds used for ensuring that the split is the same for the sampled and the original data may also not apply in this case. The technique in [48] uses **numerical interval pruning in order to reduce the number of possible split points**, and thereby make the approach more effective.
     - Furthermore, the work uses the properties of the gain function on the entropy in order to achieve the **same bound as the VFDT method** with the use of a smaller number of samples.