Structural Equation Decision Model Based on Depth Self-learning

Jing Zhang
Chongqing Technology and Business institute, 401520, Chongqing, China

Abstract
Data stream has the characteristics of continuous data arrival, fast arrival and huge data size, which brings new challenges to the study of data learning field. Among them, the decision-making algorithm is the current research hotspot. In this paper, Structural Equation Decision Model Based on Depth Self-learning, abbreviated as SEMDc, is designed and implemented. Its main contributions are as follows: (1) this paper designs and implements a continuous attribute processing method based on depth self-learning (DS) on data stream for the first time. Compared to SEMD, the sample insertion time complexity of SEMDc is reduced from $O(n^2)$ to $O(n \log n)$. When the new sample arrives, SEMDc needs to update $O(\log n)$ attribute nodes, and SEMDc only needs to update the corresponding one node. (2) The method of selecting the best dividing node of continuous attributes of SEMDc is improved, and its time complexity degree decreases from $O(n \log n)$ to $O(n)$ . (3) Compared to SEMDc, SEMDc only needs to select the optimal node from fewer candidate-dividing nodes, candidate-dividing nodes decreases from $O(n)$ to $O(\log n)$. (4) improved the decision-making method of structural equation based on depth self-learning under the traditional data learning environment, and effectively processed the noise data, and improved the decision accuracy.

Key words: Data Flow, Depth Self-Learning, Continuous Attribute, Structural Equation Decision-Making, Increment; SEMD.

1. INTRODUCTION
As the interdisciplinary subject of statistics and artificial intelligence, data learning subject is becoming a research hotspot in recent years. Various data learning techniques have been proposed and widely used in order to obtain useful information from a large number of complex data. With the information generated in large amount, the data, which need to deal with increases by millions per day or even without limit at the speed of growth, how to extract useful information from these continuous data flow (data streams), has now become some major important challenges we, face (Henseler, Ringle and Sarstedt, 2015; Sarstedt and Hopkins, 2014; Götz, Liehr-Gobbers and Krafft, 2010; Sarstedt, Ringle and Smith, 2014).

The data stream expresses as $\{...,a_{i-1},a_i,a_{i+1},...\}$ represents any time stamp, which $a_i$ represents the data vector arriving at that timestamp. As they distinguish from the traditional data model, the data flow model has the following three characteristics: (1) high-speed data arrival, high real-time requirements. (2) the scale of the data is large, it is impossible to put all the data into the memory or even the hard disk; (3) the data is processed, unless intentionally saved, otherwise can not be re-taken out of processing, or data extraction costly again. The learning method must store all the data in the media and then access the storage medium, but because of the rapid data arrival and the large data size, the traditional data learning technology cannot meet the requirements of data learning.

Decision-making is a very important data-learning technology, which aims to construct a decision-making function or decision model based on the existing data set learning. It can map the new sample to a specific category. Traditional decision models include Decision model Bayesian decision making, back propagation method, association decision method, K-nearest neighbor decision maker, SVM, paradigmatic reasoning, evolutionary algorithm, rough set method and fuzzy set method etc. Among them, the decision model model is the most popular Decision-making method of data learning faces more challenges than the traditional decision-making in terms of real-time and storage limitations, and at the same time, such as the distinction between e-mail, personalized website, Computer intrusion detection, etc. also has a better application (Preacher, Zhang and Zyphur, 2016).

The Structural Equation Decision Model (SEMD) of Domingos and Hulten (Marsh, Morin and Parker, 2014) studies how to construct decision models on data streams. Their algorithms are able to guarantee the accuracy of the constructed decision model with a certain probability. Gama et al. SEMDc design and implement, and SEMD extend from two directions: the ability to deal with continuous attributes and Bayesian decision-making technology with stronger decision-making ability on leaf nodes.
Peng et al (Wolf E, Harrington and Clark, 2013) studied the decision-making method of structural equation based on depth self-learning in traditional data learning environment, which improves the anti-noise ability of the decision model and improves decision accuracy.

2. RELATED RESEARCH WORK

2.1. SEMD

SEMD (Very Fast Decision Tree) (Marsh, Morin and Parker, 2014) is a method based on Hoeffding inequality to establish a decision-making model for data learning environment, which is generated by replacing leaf nodes with branch nodes. The most important innovation is the use of Hoeffding inequality (Montazemi and Qahri-Saremi, 2015) Determine the number of samples required for the leaf node to become a branch node.

SEMD does not describe the processing of continuous attributes at first and only introduce in subsequent studies. For continuous attributes, when SEMD creates a new leaf node, for each successive attribute, select and store M from the first arriving sample, which sort by the sorted array when the sample arrives, and the intermediate nodes of the neighboring values of each of the two different decisions maintain as alternative partitioning nodes. Once a continuous attribute has been With M different values, no alternative partition nodes add, only the new samples use to evaluate the existing candidate partition nodes. According to the difference of the leaf nodes in the tree and the available memory size, and each leaf node uses a different M value.

2.2. SEMDc

SEMDc (Asparouhov and Muthén, 2016) extends SEMD from two aspects: processing continuous attributes and applying Bayesian decision makers at leaf nodes.

In reality, most of the data contains continuous attributes, the traditional decision-making model learning method needs to sort the continuous attribute values, and the sorting operation is very time-consuming, obviously not suitable for real-time data flow requirements. SEMDc for the issue of an effective Decision Tree-based Solution.

SEMDc in each continuous attribute division is divided into two branches, the division of the test form such as attri \( \leq \) cut point, left and right branches corresponding to the division test true and false, where cut_point for the property of all possible values. In order to compare the advantages and disadvantages of each partition, we need to calculate the number of sample values less than or equal to and greater than cut_point the number of distribution. SEMDc in each decision node of the leaf nodes save samples of the class distribution vector. For each continuous attribute \( j \) (Decision tree), each node of the binary decision tree corresponds to a value \( i \) of the attribute, while the two vectors VE and VH (dimension \( k \) ) are maintained on each tree node, and each node of the binary decision tree maintains a binary tree structure And the number of samples is stored separately. When the sample arrives at the node, all-consecutive attributes of the node need to update. The time consumption of the new sample insertion algorithm is \( O(n \log n) \) the number of different values \( n \) observed in the attribute \( j \).

SEMDc has proposed a method to deal with continuous attributes, but it has the potential to divide all the possible values of the continuous attributes as alternatives, which leads to a great deal of expense. Fayad et al (Merkle, You and Preacher, 2016) have proved that for the continuous attribute, it is possible to reduce the number of alternative partition nodes, which has not been applied in SEMDc, and therefore needs to be improved.

2.3. Decision - Making Method of Structural Equation

Peng et al (Wolf E, Harrington and Clark, 2013) proposed a decision-making method of structural equation under the traditional data-learning environment based on depth self-learning.

For continuous attributes, decision-making model decision-making methods are mostly used to discretize the continuous attributes into two or more discrete intervals by dividing them into discrete intervals. These methods, called steep discretization, are obtained with clear decision boundaries. However, there is a large amount of non-deterministic information and noise data in real data. It leads to the fact that decision boundaries are not clear in many cases.

Peng et al (Wolf E, Harrington and Clark, 2013) proposed a fuzzy discretized fuzzy decision-making model based on depth self-learning, which blurs the partition nodes of continuous attributes, which solves the problem of noise data by non-determinism of fuzzy decision model, Precision.

3. BASIC DEFINITION

Definition 1. Data flow. Assume \( t \) to represent any time stamp, \( a_i \) represents the data vector arriving at this timestamp, the data stream can be represented as \( \{\ldots, a_{i-1}, a_i, a_{i+1}, \ldots\} \).
**Definition 2.** Decision model. A decision model is a tree structure similar to a flowchart in which each internal node represents a partition on an attribute, with each tree branch representing a partitioned output, with each leaf node representing a category or category distribution.

**Definition 3.** Information gain. Assuming \( S \) is a set of data samples. Assume that the class label attribute has different values. \( s_i \) is the sample number of the category \( C_i \), the entropy of the sample set or the expected information

\[
I(s_1, s_2, \ldots, s_m) = -\sum_{i=1}^{m} p_i \log_2(p_i)
\]

Assume that attribute \( A \) has \( v \) different values \( \{a_1, a_2, \ldots, a_v\} \); You can use attribute \( A \) to divide \( S \) into \( v \) subsets \( \{S_{1}, S_{2}, \ldots, S_{v}\} \), Where \( S_j \) is the subset of \( a_j \) samples with the attribute \( A \) in \( S \), \( s_{ij} \) is the number of samples in the subset \( S_j \) whose category is \( C_j \), then the attribute \( A \) is divided into the entropy of the interval or the expected information is

\[
E(A) = \sum_{j=1}^{v} \frac{s_{ij} + \cdots + s_{mj}}{s} I(s_1, \ldots, s_m)
\]

In which \( \frac{s_{ij} + \cdots + s_{mj}}{s} \) is the weight of the subset \( S_j \), the number of samples equal to the number of subsets \( S_j \) divided by the total number of samples \( S \) in a given subset \( S_j \),

\[
I(s_{ij}, \ldots, s_{mj}) = -\sum_{i=1}^{m} p_j \log_2(p_j),
\]

Among them \( p_{ij} = \frac{|S_{ij}|}{|S_j|} \) is the probability that the sample belongs to \( S_j \), class \( C_j \). The information gain divided by attribute \( A \) is \( \text{Gain}(A) = I(s_1, s_2, \ldots, s_m) - E(A) \)

**Definition 4.** Dividing the nodes. In the process of constructing the decision model, the threshold \( T \) is used to divide the continuous attributes into two or more discrete intervals, and the threshold \( T \) is called the dividing node of continuous attributes.

**Definition 5.** Hoeffding inequality. Hoeffding inequality is a strict theoretical limitation of the error probability. Assuming \( \{X_i\}_{i=1}^{m} \) is random variates, \( 0 \leq X_i \leq r \) make \( X = \frac{1}{m} \sum_{i=1}^{m} X_i \), \( X \) the mathematical expectation is \( \mu \), for the given \( \varepsilon > 0 \), Hoeffding Inequalities like: \( \Pr(|X - \mu| \geq \varepsilon) \leq 2^{-2\mu^2/\varepsilon^2} \).

**Definition 6.** Hoeffding bound. Assuming that the range of the variable \( r \) is, after observing \( n \) sample, the sample observation average value is \( \bar{r} \), then Hoeffding bound To ensure the true value of samples with confidence \( 1 - \delta \) is in between the range of \( \bar{r} \pm \varepsilon \), where \( \varepsilon = \sqrt{\frac{\mu^2 \ln(1/\delta)}{2n}} \).

**Definition 7.** Smooth discretization. A steep discretization divides a continuous attribute into several discrete intervals that are not intersecting each other by a dividing node (threshold). Smooth discretization is the use of satisfaction \( \Omega \sum_{i=1}^{k} A_i(a) = 1, \forall a \in \Omega \) of the fuzzy partition \( Q = \{A_1, A_2, \ldots, A_k\} \) The continuous attributes are divided into several overlapping intervals. The smooth discretization is determined by three parameters, the first one is the intersection point \( T \) and the other two are the membership functions of \( A_i(a) + A_j(a) = 1 \) the fuzzy set sums \( A_i \) and \( A_j \).

**Definition 8.** Fuzzy information gain. The fuzzy entropy of the sample set \( S \) is

\[
E_f(S) = -\sum_{i=1}^{n} p(C_i, S) \log(C_i, S),
\]

Among them \( p(C_i, S) = \sum_{a_j \in C_i} \left(A_i(a_j) + A_j(a_j) \right) \). The attribute \( A \) is smoothed and discretized into fuzzy entropy or expectation information of multi-interval

\[
E_f(A) = \frac{N_1}{N_f} E_f(S_1) + \frac{N_2}{N_f} E_f(S_2).
\]

Among them,
\[ E_F(S_i) = -\sum_{i=1}^{m} p(C_i,S_i)\log(C_i,S_i), \]
\[ E_F(S_2) = -\sum_{i=1}^{m} p(C_i,S_2)\log(C_i,S_2), \]
\[ p(C_i,S) = \frac{N_{F_i}^{C_i}}{N_{F_i}}, k = 1,2, \]
\[ N_{F_i} = \sum_{j=1}^{n} \left(A_1(a_j) + A_2(a_j)\right), \]
\[ N_{F_i}^{S_i} = \sum_{j=1}^{n} A_1(a_j), N_{F_i}^{S_i} = \sum_{j=1}^{n} A_2(a_j), \]
\[ N_{F_i} = \sum_{a_i \in S_i} A_k(a_j), k = 1,2, \]

Fuzzy Information Gain \( Ga_{inf}(A) = E_F(S) - E_F(A). \)

4. SEDMDS Design and Technical Details

Based on SEMD and SEMDc, we improved a smooth discretization technique, and we designed and implemented a system called SEDMDS. It utilizes depth self-learning to maintain continuous attributes, and uses a more efficient optimal partitioning node selection method, which greatly reduces the algorithm execution time, the improved smooth discretization method and the combination of the depth self-learning structure to construct the structural equation decision model, which effectively solves the problem of noise data and improves the decision precision.

4.1. Depth Self-Learning Structure

For each successive attribute \( i \), SEDMDS maintains a depth of self-learning. Each node in the tree consists of keyValue, classTotals \([k]\), left and right pointers, prev and next pointers, etc. The keyValue is used to record the attributes of the samples arriving at the classTotals \([k]\). The left and right pointers are used to record the left and right children of the node (based on the value of \( \leq \text{keyValue} \)); the prev and next pointers are used to record the predecessors and successors of the nodes, respectively.

In addition, SEDMDS also maintains a head pointer for each successive attribute, head, for traversing the entire decision tree.

4.2. Decision Tree Update Process when New Samples Arrive

Data Flow Decision Algorithm has a very important problem in constructing a decision model is that the cost of storing the information to get the best partition is very large. The attribute value of the discrete attribute is usually not too big, so its class information saving cost will not be too high, the same choice of nodes will not be too much.

Domingos and Hulten (Marsh, Morin and Parker, 2014) proposed a method to solve the discrete attribute on the data stream, and used the sort array to solve the continuous attribute problem in its follow-up work. However, the sorted array Whether the insertion of new samples, or the use of sorting array to calculate the optimal allocation of nodes, the cost is very large.

In SEDMDS, each Hoeffding tree node maintains a depth self-learning for each successive attribute on the node before it becomes a leaf node.

When a new sample \((x,k)\) arrives, the decision tree corresponding to the continuous attribute \( i \) is updated as shown in Figure 1. The time complexity of the new sample insertion for SEMD is \( O(n^2) \), and the time complexity of SEDMDS is \( O(n\log n) \) (where, for the current node, \( n \) is the number of different values of successive attributes \( i \)).
Procedure $\text{InsertValue}(TBS\ Tree(x, k, fTBSTree))$

Begin
  While($fTBSTree->right!=\text{NULL} \| fTBSTree->left!=\text{NULL}$)
    If ($fTBSTree->keyValue == x$) then break;
    elseIf ($fTBSTree->keyValue > x$) then
      $fTBS\ Tree = fTBSTree->left$;
    else $fTBS\ Tree = fTBSTree->right$;
    Creates a new node $curr$ based on $x$ and $k$;
    If ($fTBS\ Tree.keyValue == x$) then
      $fTBSTree.classTotals[k]++;$
    elseIf ($fTBSTree.keyValue > x$) then $fTBSTree.left = curr$;
    else $fTBSTree.right = curr$;
    Threads the tree;
End

Figure 1. The process of inserting a decision tree of samples of class $k$ with value $x$

When a new sample arrives, SEMDc needs to update $O(\log n)$ decision tree node, and SEDMDS only needs to update one node.

4.3. Decision tree clue process when new sample arrives

When the new sample arrives, SEDMDS needs to trail the existing binary decision tree. If the new sample has the same value as the existing node in the decision tree, only need to modify the corresponding class statistic information, do not need to re-clue, Otherwise the decision tree needs to be re-clued.

When the new sample arrives and the decision tree needs to be updated, only the pointer information of the three nodes needs to be updated at most. And the time complexity of the update process is $O(1)$ that the update process is embedded in the decision tree when the new sample is updated $\text{InsertValue}(TBS\ Tree(x, k, TBS\ Tree))$. The threading mechanism introduced in SEDMDS does not increase the time complexity of the insertion of new samples, but is still $O(n\log n)$.

Procedure $fTBST\ TbestSplit(fTBSTreePtr\ ptr,\ int\ *belowPrev[\ ])$

Begin
  If ($ptr->next == \text{NULL}$) then break;
  For ($k = 0; < count; \ k++$)
    $*belowPrev[k] = ptr->classTotals[k]$;
  Calculates the fuzzy information gain using $*belowPrev[\ ];$
  $fTBST\ TbestSplit(ptr->next,\ belowPrev[\ ]);$
End

Figure 2. The best node selection algorithm for continuous attributes

4.4. Smooth Discretization Division process of Continuous Attributes

Using the feature of depth self-learning, we use a more efficient method to select the best node.

Suppose a decision model node contains a sample, continuous attributes $i$ of the sample values for different $a_1, a_2, \ldots, a_n$, the system will maintain a structural equation decision tree for the attribute, in all adjacent
values \( T = \left( a_i + a_{i+1} \right) / 2 \) as an alternative partitioning node for the attribute. To calculate the fuzzy information gain of the nodes, we need to know the sample value \( a_{tri} \leq T \) and \( a_{tri} > T \) class distribution. SEDMDS the attribute value of the decision node in SEDMDS TBS Tree. ClassTotal s \([k]\) is used to calculate the fuzzy information gain.

As shown in Fig. 2, according to the formula of fuzzy information gain, the fuzzy information gain of all candidate nodes can be calculated by traversing the whole decision tree from the head node according to the cueing order, so the best partition node of the continuous attribute is selected.

According to the membership function of the triangular function introduced in (Hair, Sarstedt and Ringle, 2012), SEDMDS achieves the smooth discretization of continuous attributes according to the maximum, minimum and sample number of attributes recorded by each successive attribute.

The time complexity of SEMDc to choose the best partition node is \( O(n \log n) \), and the time complexity of our SEDMDS is only \( O(n) \) (here \( n \) represent the number of different values observed for continuous attributes).

4.5. New Sample Decision Process

For the new sample, SEMD starts from the root node, tests on each branch node, completes the top-down traversal process, and the final leaf node is the sample decision.

SEDMDS uses TS model decision-making method of fuzzy decision-making model. To decide a new sample, first use \( \mathcal{T} \) operator (fuzzy multiplication \( \otimes \)) to calculate all the non-zero leaf nodes with certain membership degree of decision-making; then use \( \mathcal{S} \) operator (fuzzy plus \( \oplus \)) Finally, the final decision of the sample is determined by the de-fuzzification method (Xiong, Skitmore and Xia, 2014).

5. EXPERIMENTAL RESULTS

In order to validate our proposed fuzzy incremental decision model based on the decision tree of structural equation, we will greatly reduce the decision time complexity of data learning and improve the anti-noise data ability. We designed three groups of experiments to verify the algorithm execution time, Decision error rate and decision model size.

Experimental machine configuration: Pentium IV / 2G Hz CPU, memory size is 512MB, the operating system for Linux RedHat9.0. Experiment and the literature (Marsh, Morin and Parker, 2014) in the same data environment, are using tools which are called treeData generated data flow.

5.1. Execution Time Comparison

The time complexity analysis and comparison of the algorithm is shown in Table 1.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Sample Insertion Time complexity</th>
<th>Information gain calculation time complexity</th>
<th>Number of alternative partition nodes complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEMD</td>
<td>( O(n^2) )</td>
<td>( O(n) )</td>
<td>( O(\log n) )</td>
</tr>
<tr>
<td>SEMDc</td>
<td>( O(n \log n) )</td>
<td>( O(n \log n) )</td>
<td>( O(n) )</td>
</tr>
<tr>
<td>SEDMDS</td>
<td>( O(n \log n) )</td>
<td>( O(n) )</td>
<td>( O(\log n) )</td>
</tr>
</tbody>
</table>

The main purpose of the SEMDc is to show that the decision-making technique will improve the performance of decision-making on decision nodes. In addition, the introduction of Bayesian decision-making will greatly increase the processing time of the system, so the execution time of SEMDc will be In order to compare with the processing time of the continuous attribute processing method mentioned in SEMDc, we implement the continuous attribute processing part according to the algorithm provided in (Asparouhov and Muthén, 2016).

Table 2 shows the comparison results of the three algorithms' execution time experiments. In this experiment, the experimental data are generated by treeData. In order to better compare the processing ability of successive attributes, the data used in this experiment are 20 continuous attributes. There is no discrete attribute; the number of samples is 107. 10 times the average of the experiment, the results show: SEDMDS than SEMD average execution time decreased by 16.66%, SEDMDS than SEMDc average execution time by 6.25%.

<table>
<thead>
<tr>
<th>NO of samples</th>
<th>Algorithm execution time / s</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SEMD</td>
</tr>
<tr>
<td>10000</td>
<td>4.66</td>
</tr>
<tr>
<td>20736</td>
<td>9.96</td>
</tr>
</tbody>
</table>
5.2. Decision Error Rate Comparison

Experiments in (Wolf E, Harrington and Clark, 2013) show that the smooth discretization method can effectively solve the noise data problem and improve the decision precision. With the effective combination of the decision tree of structure equation, the smooth discretization method is applied to the data stream environment. The accuracy of decision-making is shown in Figure 3. As shown in Figure 3, the decision error rate of SEMD is approximately 12.5% and that of SEDMDS approximates to 8% under 10% noise data. Experiments show that the smooth discretization can improve the accuracy of decision-making role.

![Figure 3. SEMD and SEDMDS accuracy comparison under 10% noise data](image)

5.3. Decision Model Size Comparison

The smooth discretization method based on depth self-learning for continuous attributes in SEDMDS does not change the generating framework of decision model. It only uses new data structure to improve decision speed and decision precision, so it does not change decision model size (decision model Number of nodes).

6. CONCLUSION

Based on SEMD and SEMDc, this paper improves the smooth discretization method and designs and implements the algorithm of SEDMDS, which is based on depth self-learning in dataflow environment. For the processing of continuous attributes, we design and implement the new method of the structural equation decision tree. So, the time complexity of new sample insertion and the selection of the best partitioning nodes are greatly reduced. For the most time-consuming part for sample processing and comparing with SEMD, the time complexity of new sample insertion is reduced by the time complexity of the sample processing. The time complexity is reduced from $O(n^2)$ to $O(n \log n)$, for the calculation of the node information gain, as compared with SEMDc, the time complexity is reduced from $O(n \log n)$ to $O(n)$. Making use of the conclusion proposed by Fayyad (Merkle, You and Preacher, 2016), as compared with SEMDc, and the number of candidate dividing
nodes is reduced from $O(n)$ to $O(\log n)$. The smooth discretization method is applied to the data stream environment and improves the precision of the decision-making in regard to the problem of noise data and the effective combination of decision tree of structure equation.

SEDMDS does not consider the concept drift problem (Preacher, Zhang and Zyphur, 2016; Sideridis, Simos and Papanicolaou, 2014; Fino, Melogno and Iliceto, 2014; Chiorri, Marsh and Ubbiali, 2016), CSEMD (Chiorri, Marsh and Ubbiali, 2016) has provided a solution to solve the concept drift, whether we can extend the current method to the existence of the concept drift of the situation is our next research focus.

REFERENCES