Abstract  The availability of large and diverse datasets has lead to many exciting opportunities for the development of tools for knowledge discovery. Artificial Intelligence algorithms have a long history of being used to analyze large and complex datasets. Recent studies have suggested that the commonly used diagnostic measure, the Body Mass Index, may not be the most appropriate indicator for obesity. In this study we use data from a comprehensive study conducted by the Center for Disease Control and a machine learning approach in order to develop a domain independent tool to produce a binary decision tree. We use this tool in order to assess the most important risk factors for obesity as well as to classify an individual’s risk for obesity.

1 Introduction

With the proliferation of large and diverse datasets, the need to develop and expand knowledge-discovery technology is constantly growing. There is a great need to use a large range of tools to extract information from these emerging databases. This paper presents a data mining approach that uses a machine learning algorithm in order to evaluate the most important factors in obesity while also providing a tool for the classification of new data.

The software tool that we have developed analyzes a large dataset provided by the Center for Disease Control (CDC) [1] in order to assess what the most important factors are that influence obesity in the population. Recent studies [2] have advanced the idea that the Body Mass Index (BMI) measure [3], despite its simplicity and wide use for diagnosis, may not be the best indicator for obesity. The goal of this study is to look beyond the BMI measure in order to find a better understanding of what may be the most influential factors in obesity. This research takes into consideration possible factors of obesity by considering individual’s health issues and habits along with regional factors.

The software tool was implemented using a data independent and objected oriented approach and can be used effectively with any other type of data. It uses a machine learning algorithm, Iterative Dichotomiser 3 (ID3) [4], which builds a decision tree based on the data provided. The resulting decision tree can then be used to evaluate an individual’s risk for obesity.

2 Background and Related Work

The main algorithms that have been developed for decision tree learning take a top-down, greedy approach through the space of decision trees. They are the Iterative Dichotomiser 3(ID3) [4] and its extension C4.5 [5] and were developed by J. Ross Quinlan. ID3 is popular amongst researchers and academics performing research in health and medical fields.

With the prevalence of Electronic Medical Records (EMR), it is much easier for researchers to gain accesses to large and robust medical databases. This has generated a large amount of research focused primarily on uncovering the causes of various diseases. Researchers in Taiwan used an ID3-based algorithm to produce a tree that was confirmed to be useful for clinical diagnosis of cerebrovascular disease [6]. Another study at Shiyou University in China [7] used the ID3 algorithm for finding hidden classification rules among data relating to
students’ physical strengths and sports aptitude levels. The tool has been successfully used to map students’ various fitness levels, enabling physical education instructors to improve their course plans and activities. In another study, researchers at the University of Michigan and the Huazhong University developed a workbench [8] for testing the integrity of machine parts. Their tool employs the ID3 algorithm as a basis for testing the tolerance of machine parts commonly used in the automotive industry. A related collection of machine learning algorithms for data mining tasks has been developed in Weka [9].

3 Data

In our research we used a dataset made available by the Center for Disease Control (CDC) [1]. The dataset is the result of a 2009 study into community health indicators. While the dataset contains a large pool of measurements, we have chosen to focus solely on data pertaining to obesity and obesity risk factors. From the CDC dataset the following ten attributes were selected: exercise frequency, fruit and vegetable intake, obesity, blood pressure, smoking habits, diabetes, being insured, access to primary care physicians, access to dental care, and access to community health centers. The data was collected during the 2008 census for each of the 3,141 current counties in the United States.

4 Methods

The Iterative Dichotomiser 3 (ID3) [4] algorithm was implemented and adapted to handle continuous valued attributes and training data with missing attribute values. ID3 was chosen over other comparable algorithms, such as C4.5, as it produces a relativity small decision tree with a boolean outcome. ID3 uses the statistical property called Information Gain. The Information Gain measures how well a given attribute separates the training data into the targeted classification, by computing the expected reduction in entropy caused by the partitioning of the data according to this attribute.

If the target classification is boolean and the attribute S can have c different values, then the definition of the entropy [10] of a collection of examples S is:

$$ Entropy(S) = \sum_{i=1}^{c} - p_i \log_2 p_i, $$  \hspace{1cm} (1)

where $p_i$ is the proportion of S belonging to class i. The definition of the information gain, Gain(S, A) of an attribute A, relative to the collection of examples S, is:

$$ Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} * Entropy(S_v) $$  \hspace{1cm} (2)

where Values(A) is the set of all possible values for attribute A and $S_v$ is the subset of S for which attribute A has value v.

To make the ID3 algorithm work with the CDC data, we had to address two issues. Firstly, the attributes had values in a continuous rather than discrete range. Secondly, some of the attributes from the CDC data had missing values. To work with boolean values for the outcome, we had to find the appropriate threshold against which to compare the values of the continuous attributes. Even though it is common practice to choose the midpoint of the range as the representative threshold, we found that a more meaningful value was the median value of the dataset. To deal with the missing attribute values, we chose to exclude from consideration the data points with missing data.

For example, consider the data in Table 1, which has ten data points with attributes named No_Exercise, High_Blood_Pressure, Smoker, Diabetes, Uninsured, Few_Fruit_Veg, and the target classification is is_Obese. Note that representative thresholds have been used to transform the values for all attributes to boolean values.
An example for the computation of the information gain for the attributes High_Blood_Pressure and Smoker is shown in Figure 1.

Table 1: Training examples for the target outcome is_Obese. The table only shows a small sampling of the large training dataset used to build the decision tree.

<table>
<thead>
<tr>
<th>Subject</th>
<th>No_exercise</th>
<th>High_Blood_Pressure</th>
<th>Smoker</th>
<th>Diabetes</th>
<th>Uninsured</th>
<th>Few_Fruit_Veg</th>
<th>is_Obese</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sub1</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Sub2</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Sub3</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Sub4</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Sub5</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Sub6</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Sub7</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Sub8</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Sub9</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Sub10</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Figure 1: Entropy and Information Gain, as calculated for the High_Blood_Pressure and Smoker attributes at the root level. For High_Blood_Pressure, the data set has 115 positive examples and 1264 negative examples. The dataset results in collections of [98+, 599-] (High_Blood_Pressure = true) and [17+, 665-] (High_Blood_Pressure = false). The resulting information gain is .697 for High_Blood_Pressure compared to an information gain of only .102 for the attribute Smoker.
The ID3 algorithm uses a top down, greedy approach to building the decision tree. Once a node has been placed in the tree, the algorithm does not revisit it or reconsider its placement. The dataset is divided into two subsets: a training set that is used to build the tree, and a test set that is used to verify the validity of the decision tree. We have chosen a random sampling comprising 90% of the original dataset to serve as the training data set. The remaining 10% is used for the test set. The algorithm has been implemented such that the size of the two sets is variable and therefore can be changed during the course of the experiment.

ID3 uses a recursive algorithm to build the decision tree. The attribute with the highest overall Information Gain is selected to be the root of the tree. The chosen attribute is then removed from the set and the data set is split on the values of this attribute. The same method is then used recursively for each of the subtrees to find the highest information gain attribute from the remaining attribute set and using the remaining training set data for that specific branch. A base case is reached when either all attributes have been placed on the tree or the classification is the same for all the data points in the remaining training set.

5 Experimental Work

The application was built using Java [11] for the framework, MySQL [12] was used as the Database Management System, and the Java Universal Network and Graph Framework (JUNG) [13] was used for visualization of the decision tree.

The system was designed to be data independent, so that the application can handle a diverse range of other datasets. The application connects to a MySQL database and retrieves the data through the Java Database Connectivity (JDBC). Dependency injection, and more specifically constructor injection, was used to implement the database connection. This was done such that the specific database connection can be dynamically changed at runtime and is therefore database independent. By adopting a uniform approach to data handling, the application is not domain bound, and can easily be adapted to be used with a wide range of domain types. The application was used with aggregate health data, but can easily be modified to accommodate data types from a variety of other domains such as finance or material engineering.

Once the application has successfully queried the database and populated the appropriate data structures, the dataset is divided into two subsets, the training data and the testing data. The application uses a randomly chosen 90% of the original data for the training data set to build the decision tree. The application uses the Java shuffle algorithm [14] to randomize the data. During the construction and the testing of the tree, any data point that is found to have a missing attribute value is eliminated from the current process in order to maintain data integrity. The ID3 algorithm then uses a pre-order recursive traversal to build the tree. At each step, the entropy and information gain are computed for the remaining attributes, and the attribute with the highest information gain is placed in the tree. In the computation of entropy and information gain, if a data point has missing values for a given attribute, that entry is removed from consideration for the construction of the subtree.

The Java Universal Network and Graph Framework (JUNG) [13] was used to create a graphical representation of the decision tree. JUNG is a software library that provides a common and extendible language for the modeling, analysis, and visualization of data that can be represented as a graph or network. JUNG is written in Java, which allowed us to extend its libraries and modify them to work with our software. The application uses a depth first
traversal to navigate the tree for visualization. Figure 2 shows the root and the left subtree of the decision tree built using the CDC data. Each node is represented by a circle, with the name of the attribute appearing below it. The left and right edges connecting a node to its children are labeled with Yes and No respectively to allow for easy visual navigation of the tree. Similar to the edges, each outcome of a path is labeled with either a Yes (Y) or a No (N).

Figure 2: Left half of the resulting decision tree for the target outcome is Obese. The classification of a new data point follows the path from the root node to a leaf node using the point’s attribute values to take either left of right edges.

In addition to providing the user with the visualization of the decision tree, the application uses the remaining 10% of the original dataset to test the validity and accuracy of the tree. The classification attribute in the data that we used was obesity. In order to reduce the continuous valued attribute to a binary one, we used the national average as a threshold. Each data point in the testing set is classified using the decision tree and the final outcome is compared for validation with its own classification. The application keeps track of the total number of correct and incorrect classifications and presents the information to the user.

6 Results and Conclusions

We have used the developed tool on a dataset provided by the CDC [1] that had 3,141 data points each with 10 attributes. We tested the consistency of our application by recording the resulting trees over several runs. During each run a training dataset was randomly selected as a subset of the original dataset. We found that the attribute No_Exercise was placed as the root node 100% of the time. The attribute High_Blood_Pressure was placed on the second level of the tree 95% of the time, while the Smoker attribute was placed on the second level 5% of the time. Similarly, the attribute Few_Fruits_Veg was consistently placed on the last level of the tree. The only slight variation noted was in the middle of the tree where the attributes access to primary care physicians (Prim_Care-Phys_Rate) and access to dental care (Dentist_Rate) were placed interchangeably on levels 6 and 7 in the tree. This, however, does not affect the major findings for this study, as the top level attributes remain consistent. The
results showed that, using the CDC data, out of all the considered factors for obesity, the most important one was not getting enough exercise. The second most important factor was having high blood pressure, closely followed by being a smoker. The least important factor of the ones we have considered was not eating enough fruits and vegetables.

The machine learning application that we have developed has been specifically designed to be data independent and therefore can be used with any type of data sets. Potential areas such as material science, meteorology and banking, where the attributes have boolean values and outcomes, are good candidates for data mining using this tool. Any other data with continued valued attributes and for which threshold values can be chosen, that do not affect the classification of the target outcome, can be used with this tool. The dataset that has been used in this study had missing values for some attributes, a common problem in many large datasets. A further development of this application may be designed to handle the data points that have missing attribute values differently. Instead of excluding these data points from further consideration, the application can replace the missing values with values computed using statistical inference. We expect his approach to increase the robustness of the application.

References