Aprendizagem e Mineração de Dados

Mestrado em Engenharia Informática e Multimédia

Relatório

Final Project A

AMD_D_20

Pedro Gonçalves – 45890

Rúben Santos – 49063

Rodrigo Dias – 45881

1. Introduction

This project aims to explore the relation among the notions of data, information and knowledge. We will apply those concepts in specific scenarios and experiment (and develop) the techniques that allow us to “move” from data to knowledge.

We will also explore the classification and clustering that are nuclear to data-mining, knowledge discovery, machine learning and information retrieval. The techniques resort to statistics (e.g., 1R and Bayes rule), induction of decision trees (e.g., J4.8/C4.5, ID3) instance based (e.g., KNN with KDTree support).

To achieve so, we will use certain tools to manage data (e.g., PostgreSQL), to discover knowledge (e.g., Orange data mining) and to implement specific algorithms (e.g., via Python).

2. Scenario 1

In this first scenario, we will experiment with a very simple and small dataset, composed of not many attributes nor many records.

2.1. Scenario Description

The medical center “**MedKnow**” uses a database management system (**DBMS**) that contains all the data gathered, throughout time, about each patient’s visit to a doctor (that works at “**MedKnow**”). The current goal of the ophthalmology team is to analyze all the information accumulated (throughout time) in order to extract the patterns that provide useful indicators to support the prescription (and diagnosis) activity.

To achieve that goal they decided to contact the “**SoftKnow**” company and to send them the a file containing a data snippet (related with the lenses prescription activity) and proposed the following challenge: “send us a prototype of a system that would provide “**MedKnow**” not only the operational (daily-work) support but also the strategic perspective (useful patterns) that can be extracted from that daily-work data”. In this first scenario we will experiment with a very simple and small dataset.

2.2. Dataset Analysis

The data snippet sent by "MedKnow" contains all the data gathered, throughout time, about each patient's visit to a doctor in their ophtalmology team. It contains four specific patient conditions (variable attributes) and the type of lenses that were prescripted to that patient (the class label attribute).

The data snippet contains 5 attributes and 16 different records.

age: The age of the patient (discrete). Can be assigned 3 values: "young" - when the patient is still at a young age (normally 0 - 27); "pre-presbyopic" - when the patient isn't young anymore, but still hasn't developed any presbyopic condition (normally 28 - 38); "prebyopic" - when the patient has developed a presbyopic condition. Note that presbyopia is an eye condition in which the patient's eye slowly loses the ability to focus quickly on objects that are close. It's a disorder that affects everyone during the natural aging process.

prescription: The type of eye defect of the patient (discrete). Can be assigned 2 values: "hypermetrope" - when the patient can see distant objects but is unable to see nearby objects clearly; "myope" - when the patient is unable to see things clearly unless they're relatively close to the eyes.

astigmatic: Whether the patient suffers from astigmatism or not (discrete). Can be assigned 2 values: "yes" or "no". Note that astigmatism is a common and generally treatable imperfection in the curvature of the eye that causes blurred distance and near vision.

tear_rate: The tear rate of the patient (discrete). Can be assigned 2 values: "normal" or "reduced". Note that tear rate is defined as the percent decrease per minute of fluorescein concentration in the patient's tears after the instillation of fluorescein.

lenses: The type of lenses prescripted to the patient, based on the other attributes (class label, discrete). Can be assigned 3 values: "hard" if the patient is more advised to use hard contact lenses; "soft" if the patient is mroe advised to use soft contact lenses; "none" if the patient doesn't need lenses at all.

2.3. Database Implementation

Firstly, a relational database needs to be designed in order to support the daily work at the hospital. As it's illustrated by the **Entity-Relationship** diagram in **figure 1**, the involved entities will consist of **Doctor**, **Disease**, **Patient** and **Diagnostic**.

Patient: The individual being examined by the **Doctor**. Has a **name**, a **CC** (Cartão de Cidadão) **number** and a **birth date**;

Doctor: The individual that will be treating the **Patient**. Also has a **name**, a **CC** (Cartão de Cidadão) **number** and a **birth date**;

Diagnostic: After the examination, the **Doctor** reaches some conclusions about the Patient's visual health condition. This entity will consist of a table with all possible combinations of the values each attribute (**age** and **tear rate**) can assume, each combination identified by and **id**.

Disease: With the knowledge about the **Patient** condition (**Diagnostic**), the **Doctor** will identify which of the disease situations he fits in. Like the **Diagnostic** entity, this one also will consist of a table containing all possible combinations of each disease (**isAstigmatic**, **isMyope** and **isHypermetrope**), each combination identified by their **id**.

As it's possible that the same **Patient** could be **astigmatic**, **myope** and/or **hypermetrope**, the three conditions were separated and are now independent. Each of them can assume "**true**" or "**false**" as possible values.

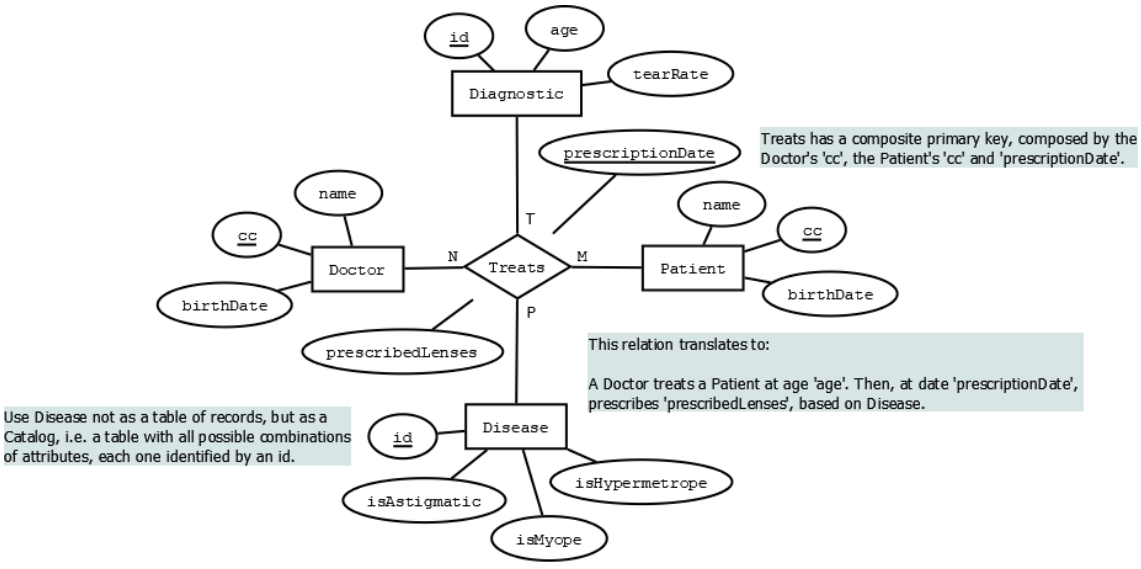

Figure 1 - Entity-Relationship Diagram

Note that there is a quaternary relation called Treats that will relate all four entities previously described. This relation will be our fifth table.

Now, the database schema needs to be implemented using **SQL**. The '**scripts'** folder contains the scripts used to create and populate de database.

2.4. Exporting Data

To export a selection of data to be analyzed, we will use the **SQL** command **CREATE VIEW** to group together the data we need, and the **SQL** command **COPY** to export it to an external file.

Something we need to consider is the fact that the data needs to be exported in a proper format for the **Orange** data-mining **framework**.

Every table in a relational database contains at least one header, with the name of the attributes in the table. The **Orange framework** needs two additional headers: A second header indicating if the attribute assumes either discrete or continuous values, and a third header indicating which attribute is the class label.

The data selection we will be exporting contains only useful information for the lenses prescription. For example, the **CC number** and the **name** of the **Patient** are not important in this context.

Here are the headers of the view created:

age	tearRate	isAstigmatic	isMyope	isHypermetrope	prescribedLenses
discrete	discrete	discrete	discrete	discrete	discrete
					class

Now, with the **COPY** command, this view will be exported to an external **txt** file, containing the previously described headers and all the records.

2.5. 1R Implementation

1R, short for **One Rule**, is a very simple, yet accurate, data-mining classification algorithm that generates one rule for each predictor in the data, and then selects the rule with the smallest total error.

To create the rule for each predictor, **1R** constructs a frequency table for each predictor against the target. In other words, **1R** will opt for the predictor that most influences the class label, meaning this algorithm only takes in consideration one attribute at a time, ignoring all others.

After implementing this algorithm in **Python**, we discovered that, as of now, with the dataset we currently have, the attribute with the smallest total error is '**isAstigmatic'**. This attribute will be our **1R** predictor. When '**isAstigmatic**' is true, the '**prescribedLenses**' should be '**hard**', and when '**isAstigmatic**' is false, '**prescribedLenses**' should be '**none**'.

2.6. ID3 and Naive Bayes Implementation

ID3 is one of the many algorithms used to build decision trees. A decision tree is a structure containing nodes and edges, and is built from a dataset. Each node is either used to make a decision (known as decision node) or represent an outcome (known as leaf node).

Naïve Bayes is a classification technique based on **Bayes' Theorem** with an assumption of independence among predictors. In simple terms, a **Naive Bayes** classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. **Naive Bayes** model is easy to build and particularly useful for very large data sets. Along with simplicity, **Naive Bayes** is known to outperform even highly sophisticated classification methods.

2.7. Evaluation and Conclusions

Using the **Repeated Stratified Holdout** method for the **ID3** and **Naïve Bayes** classifiers, we will now compare each of the three approaches to this problem (**1R**, **ID3** and **Naïve Bayes**, resorting to the following five evaluation metrics: **accuracy** (or success rate), **precision** (or positive predictive value), **recall** (or sensitivity), **F1 score** (or harmonic average of **precision** and **recall**) and **kappa score**.

For the **1R** algorithm, the following values were obtained:

- Accuracy: 71.43%.
- Precision: 51.19%.
- Recall: 71.43%.
- F1 score: 59.59%.
- Kappa score: 54.84%.

For the **ID3** algorithm, the following values were obtained:

- Accuracy: 71.43% (+/- 11.07%).
- Precision: 71.93% (+/- 15.36%).
- Recall: 71.43% (+/- 11.07%).
- F1 score: 69.01% (+/- 12.57%).
- Kappa score: 56.37% (+/- 16.15%).

For the **Naïve Bayes** algorithm, the following values were obtained:

- Accuracy: 77.14% (+/- 13.09%).
- Precision: 82.62% (+/- 11.11%).
- Recall: 77.14% (+/- 13.09%).
- F1 score: 78.44% (+/- 14.25%).
- Kappa score: 67.37% (+/- 22.41%).

Looking at the results, Naïve Bayes arguably seems like the best approach.

3. Scenario 2

For the second scenario, we will experiment with a much, much larger dataset. We will be able to see how the different algorithms, previously analyzed, perform with much larger datasets.

3.1. Introduction

The "**FungiData**" institute asked us to analyze a dataset with information regarding de edibility of mushrooms (i.e., if they are good to eat or poisonous).

3.2. Data Analysis

The dataset sent by "**FungiData**" contains **23** attributes, one of them being the class label (whether it's edible or not). All attributes are discrete and the dataset contains exactly **8416** instances, i.e. rows (records). There are **2480** missing values, denoted by "?".

To better analyze the data, we will the dataset sent by "**FungiData**" into the proper format required by the **Orange framework**. We used the same method as in **2.4** (Exporting Data). With the help of Orange, we can now observe the dataset in a more comfortable and compact way.

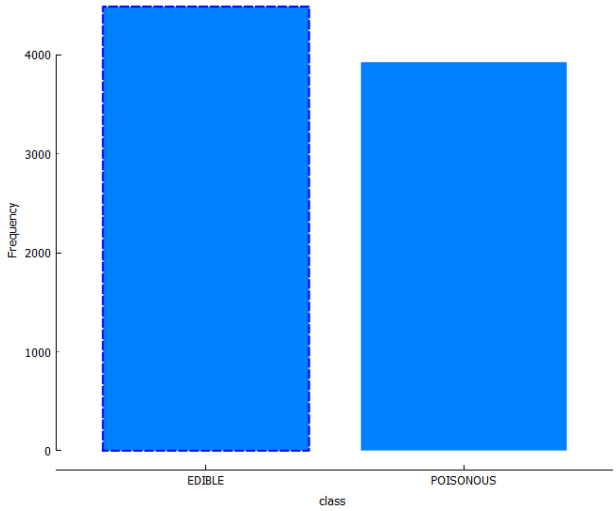

Figure 2 - Class values frequency

For instance, in **figure 2**, Orange generated a bar chart illustrating the frequency of each class value (edible and poisonous).

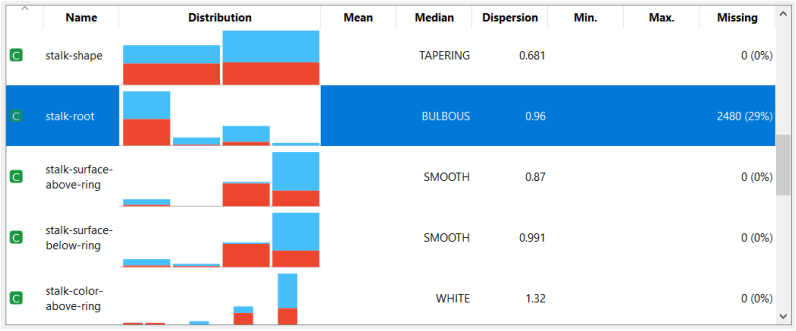

Figure 3 - Attribute Statistics

Orange also shows us some interesting statistics about every single attribute in the dataset, like dispersion and the percentage and exact number of missing values (**figure 3**).

We can also generate a **Decision Tree** model and a **Random Forest** model as it's shown in **figure 4** and **figure 5**, respectively.

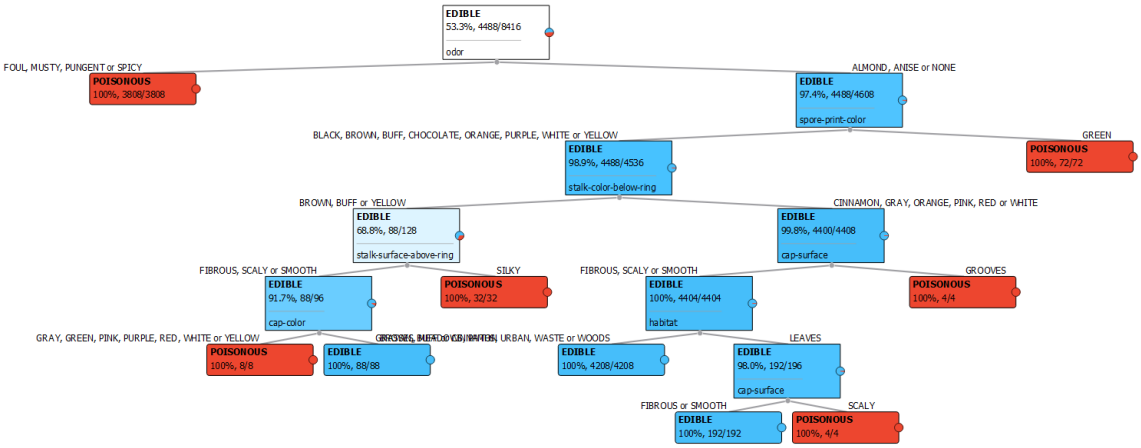

Figure 4 - Decision tree model

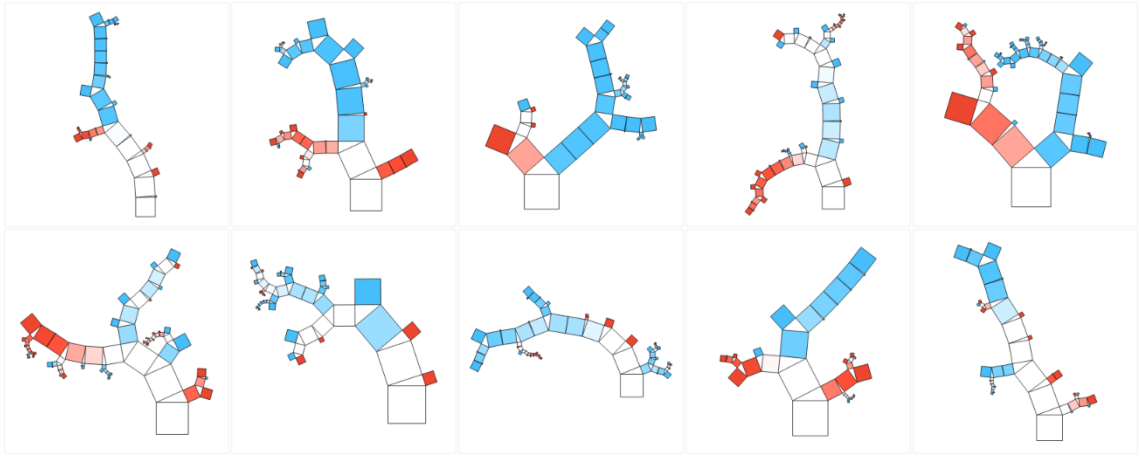

Figure 5 - Random forest model

4. Conclusion

We experimented with two types of datasets. A very simple and minimalist one and a larger one, with thousands of records. This latter case, we saw that the **Orange** software helps a lot with viewing and having a graphical reference for our data, as well as interpreting it. We used essentially three data-mining algorithms, each of one having their level of complexity and precision. This process of moving from "**data**" into "**knowledge**" is very sensible and should be approached in a very methodical way, in order to achieve precise results with the least amount of work and complexity possible.