Intelligent tool for visual data analysis

Bachelor's Thesis

Jorge Oses Grijalba

Double Major in Mathematics and Computer Science
Computer Science Faculty
Complutense University of Madrid

March 2019

Documento maquetado con TEXIS v.1.0.

Este documento está preparado para ser imprimido a doble cara.

Intelligent tool for visual data analysis

Memoria que presenta para optar al título de Doctor en Informática

Jorge Oses Grijalba

Dirigida por el Doctor

Tutor no definido. Usa \`tutorPortada`

Double Major in Mathematics and Computer Science
Computer Science Faculty
Complutense University of Madrid

March 2019

Al duque de Béjar
y
a tí, lector carísimo

I can't go to a restaurant and
order food because I keep looking
at the fonts on the menu.
Donald Knuth

Agradecimientos

A todos los que la presente vieren y entendieren.

Inicio de las Leyes Orgánicas. Juan Carlos I

Groucho Marx decía que encontraba a la televisión muy educativa porque cada vez que alguien la encendía, él se iba a otra habitación a leer un libro. Utilizando un esquema similar, nosotros queremos agradecer al Word de Microsoft el habernos forzado a utilizar LaTeX. Cualquiera que haya intentado escribir un documento de más de 150 páginas con esta aplicación entenderá a qué nos referimos. Y lo decimos porque nuestra andadura con LaTeX comenzó, precisamente, después de escribir un documento de algo más de 200 páginas. Una vez terminado decidimos que nunca más pasaríamos por ahí. Y entonces caímos en LaTeX.

Es muy posible que hubíeramos llegado al mismo sitio de todas formas, ya que en el mundo académico a la hora de escribir artículos y contribuciones a congresos lo más extendido es LaTeX. Sin embargo, también es cierto que cuando intentas escribir un documento grande en LaTeX por tu cuenta y riesgo sin un enlace del tipo "*Author instructions*", se hace cuesta arriba, pues uno no sabe por donde empezar.

Y ahí es donde debemos agradecer tanto a Pablo Gervás como a Miguel Palomino su ayuda. El primero nos ofreció el código fuente de una programación docente que había hecho unos años atrás y que nos sirvió de inspiración (por ejemplo, el fichero `guionado.tex` de TeXIS tiene una estructura casi exacta a la suya e incluso puede que el nombre sea el mismo). El segundo nos dejó husmear en el código fuente de su propia tesis donde, además de otras cosas más interesantes pero menos curiosas, descubrimos que aún hay gente que escribe los acentos españoles con el `\'{\i}`.

No podemos tampoco olvidar a los numerosos autores de los libros y tutoriales de LaTeX que no sólo permiten descargar esos manuales sin coste adicional, sino que también dejan disponible el código fuente. Estamos pensando en Tobias Oetiker, Hubert Partl, Irene Hyna y Elisabeth Schlegl, autores del famoso "The Not So Short Introduction to LaTeX 2$_\varepsilon$" y en Tomás

Bautista, autor de la traducción al español. De ellos es, entre otras muchas cosas, el entorno `example` utilizado en algunos momentos en este manual.

También estamos en deuda con Joaquín Ataz López, autor del libro "Creación de ficheros LaTeX con GNU Emacs". Gracias a él dejamos de lado a WinEdt y a Kile, los editores que por entonces utilizábamos en entornos Windows y Linux respectivamente, y nos pasamos a emacs. El tiempo de escritura que nos ahorramos por no mover las manos del teclado para desplazar el cursor o por no tener que escribir `\emph` una y otra vez se lo debemos a él; nuestro ocio y vida social se lo agradecen.

Por último, gracias a toda esa gente creadora de manuales, tutoriales, documentación de paquetes o respuestas en foros que hemos utilizado y seguiremos utilizando en nuestro quehacer como usuarios de LaTeX. Sabéis un montón.

Y para terminar, a Donal Knuth, Leslie Lamport y todos los que hacen y han hecho posible que hoy puedas estar leyendo estas líneas.

Abstract

aaaaaaaaaaaa

aaaaaaa

This document reflects my Bachelor's Thesis corresponding to the Double Degree in Mathematics and Computer Science, developed within the area of intelligent data analytics and 'Case Based Reasoning'. During the progress of the project, the principles applicable in any environment of data processing and the science behind it are explained generally and aimed to be usable in any kind of context by any user provided the right format of data. Nowadays, highly heterogeneous data collection and processing methods are employed in all industries, however the techniques employed to get useful information out of the data usually have a generalistic aim, and the work relevant to the field itself is often done manually. In this work we aim to provide an automated way to analyze information while taking into account information and techniques relevant to the field of the analysis. The objective of this Degree's Final Project is the development of a prototype capable of carrying this analysis while being able to learn based on user input. As a Proof of Concept, we have included several medical domains with each one having developed specific methods and techniques for them. To serve as a base for this analysis, we have also developed a system for storing, loading and analyzing the information of the domain and the information provided by the user. This system will be the backbone of our architecture and enable the Case Based Reasoning analysis to function correctly in very different situations, providing the metrics and functions needed for every case.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Abstract: In this chapter we outline the thesis, state what our objective is and what we hope to achieve and list the programming libraries, techniques and methods used to achieve our goal.

1.1 The Initial Problem

The need for performing analysis on large amounts of data to get very precise and specific information is becoming more and more present everyday in the jobs of data analysts and scientists in every field of work. Large amounts of time are wasted on repetitive tasks such as data wrangling, data transforming and the generation of tailored reports or collections of information with different objectives for diverse profiles with varying degrees of expertise. These reports are usually formed by a piece of text acompanied by some graphs. It is very common that from these huge amounts of data we want to extract some precise and relevant information to be presented to someone. These reports have a process behind them that entails the filtering, transformation and selection of the relevant information that will finally be part of the report. Here we have two questions to answer. First, we must choose what information to present, which is equivalent to choosing what information from the almost unlimited attributes that our data has is relevant to the user. It is clear that this has an objective part, in the sense that it is first and foremost a matter of which data is relevant within a certain domain of knowledge and a certain set of metrics, but it also has a subjective side, in the sense that it's not the same to present medical information to a patient or to present it to a doctor. The second answer, and perhaps the most subjective is how to present it. This second decision is related to things like choosing a type of graph, its colors, the font of the text, the words used... and almost an infinite list of subjective choices that build upon the previous more objective selection of information to form the final concept of a report of a piece of information.

1.2 Solution Proposed

We believe that there is no unique formula to generating each report, because it would require the abstraction of very different problems in very different situations and for an almost unlimited variety of users. Furthermore, what if we have to generate a report for a new user? Could this be similar to other reports presented for other users? We believe that most of these questions can't be answered by a rigid mathematically formulated system, and are best tackled by a mixed system that combines an objective analysis of the information through a variety of metrics, analysis of correlations, distributions and other objective metrics with a subjective approach that takes the final user into account. Instead, what we propose is a mixed system in which an expert provides an initial input that signals some of the important aspects of the information, and then a pool of experts validates the subjective way in which an information is presented to end up choosing a default report form to present to a new user of their class. To start, we categorize the users or people that will be presented with the report into groups. Then, these groups will provide knowledge of the relevant objective information that they're looking for in the data, like what analysis to perform or what values of certain metrics they'd consider to be relevant. Once this information is fed to the system, it's able to generate reports completing the subjective decisions from semi-random choices from a pool of computer generated graphs, color choices and text based reports. Then our pool of experts proceeds to validate the best report by a voting system based on an ELO tournament. The best selected report is then presented as default to users of this class. Each user will also be able to change the result presented to them in terms of both content (objective) and presentation (subjective), and because the system recognizes individual users it will remember their choices. Users will also have a profile attached to them containing relevant data to the presentation of the information that will allow us to define a metric of sorts between users to further use this single user customization to influence how information is presented to users of the same class once enough individual inputs have been recorded, possibly substituting the initial ELO based report which will always act as default.

1.3 Overview

The logical structure chosen for the program reflects the need for our tool to be a fully functional agent in and out of itself. We have designed it with a clear divider between a backend capable of storing the information and handling at the lowest possible level, which provides the frontend side with easy methods to get the information it needs, which is then processed taking who is going to look at it into account and then adequately presented to the user. A cornerstone of the program's functionality is to be able to remember decisions taken by a certain user and to be able to compare new data to old data of its kind. From these two necessities it is natural to consider some kind of identification system for our datasets, as automated as possible so it needs minimum user input and remains independent of the use case. For our program, if two datasets contain the exact same set of column names then they are considered to be comparable to each other, and every information stored about this kind of datasets will carry an identifier with the column names. From here onwards, the term 'domain' shall refer to information coming from the same kind of dataset.

1.4 Workflow

When a new dataset doesn't match any previous knowledge, our program automatically creates a new representation for these datasets which is stored along the others. If it detects a matching JSON with knowledge o its domain it loads that instead. Each representation of a domain stores data such as how many datasets have been loaded and a number of stats for each dataset and its columns depending on its types which will be specified later. Also, each domain has a number of 'profiles' associated which correspond to *who* this data is associated with. These profiles contain both historical data of the specific owner of the data (in our practical example, the patient data), and who will watch the report generated by this program, that is, the user of the program. The information that we're using will be stored in a specific JSON format for each kind that will be specified in chapter 2. When a dataset is introduced, the program loads the previous information, analyzes it, compares it and generates relevant information to the user. Then it updates the information with both the results of the analysis and user provided information. This workflow will be the basic use case of the program for every kind of data.

1.5 Structure

A clear module structure is provided so each module does a task in the workflow. The main modules on the backend side of our application are the Storage module, the CBRStorage module and the Analysis module. For the frontend, the logic structure will be split into the Reporter module and the Presentation module.

1.6 Tools

Our programming language of choice has been Python, particulary making use of its class to dictionary representation methods which make the work of manipulating the JSON structures much easier than using more rigid languages. A public repository has been created at (TODO:link here), and we have used Jupyter Notebooks for the testing and formation o a prototype which has been then moved into standard Python packages.

Notas bibliográficas

These are the bibliographical notes (**?**)

En el próximo capítulo...

This is the next chapter section

Chapter 2

The program structure

Abstract: In this chapter we provide an analysis of the program structure from an abstract point of view.

2.1 The Structure

As we have stated before, our program consists of a series of modules designed to interact with each other and provide the necessary tools to work smoothly regardless of the data being used. Here we take a look at each functional module, providing a clear abstract picture of what our tool really is.

2.2 Case presentation

The input for our program is really just an user who desires to analyze a certain dataset. From now on on this chapter, we assume we have previous information for both the domain the dataset belongs to and about the type of user that is trying to analyze it. We will expand on how the base cases are formed and how we adapt to new profiles and new domains on further chapters, but for now let's say that a process is in place to ensure that the default report is correct enough, the analysis performs its due processes and the result is at least acceptable and relevant to the final user.

2.3 Handling the Information

First and foremost we need a way to store information and to retrieve it effectively. We will store two different kinds of information : objective information about the results of analysis performed on a dataset and information on what information to present and how to present it to the user. Since the first is related to the domain and applies to all users, and the second concerns both the domain and the user, it makes sense to store them and handle them separately. Regarding the first kind, we now provide both a way to identify which domain it belongs to and a general description of what it contains, which will be specified later during the technical implementation chapter.

It contains statistics and properties from both columns and datasets, different for different types, as well as how they were measured. Saving how they're measured is important to measure new datasets and to be able to compare metrics effectively. Both the column specific stats and the dataset stats are subjective to change with each dataset added to the domain, so it is important that we're able to update this information readily and effectively. It's also important to be able to distinguish between domains. The solution proposed is to use the columns of each dataset as a unique key that identifies a domain, and their objective information will be stored as part of the same object. In the case of subjective kind, we combine this with a unique user identificator to make a unique key to identify the information.

Each of these structures has parts of the program dedicated to loading this information into the program and updating it when presented with new datasets and storing it overwriting the previous information, creating a dynamic system capable of learning from users and gathering new insights.

2.4 Analyzing the Information

This analysis will concern the objective side only, so we don't need to take the user into account for now. First a dataset is provided as input for our tool. From this dataset, we retrieve the information of the associated domain so we know how to perform the analysis on this new dataset, as it has to be comparable to the domain information. So, using the metrics and analysis detailed in the domain information, we perform them on the dataset and compare the results with those of the domain through a series of comparison metrics which will be detailed later. These results are then condensed into an object containing all the objective data from the comparisons. This output will be the input for the subjective side of our program, as this information will then be filtered and transformed in a way tailored to the user.

2.5 The Subjective Analysis

This analysis will concern the subjective side, and is to be performed after the previous step has been completed. From the representation of the objective analysis, we need to perform two tasks. One is the filtering of such information, selecting the information relevant to the user, and the other is choosing how to present this information to the user. To perform these two tasks we first load the information we know about the user, which has been stored separately from the objective information as we have previously stated. We then filter the information through the user provided profile, selecting which comparison is most relevant. Finally we form a report using the graphical information on how to present each bit of data to this user.

2.6 Presentation and Feedback

Once the user has been presented with the report, he's able to make changes to it using a graphical interface. When the report is saved and the user has exited the program, it initializes a shutting down routine on which the two information databases are updated with the changes made by the user (if they existed) and the new information provided by the dataset. This is the main way our program has of expanding its knowledge, which will then be used to present better reports to the same user and to adapt the information it presents to new users which will be similar to this one. So we get a double benefit, getting both better results for this user and for all users of the same domain and class each time someone uses our program.

Notas bibliográficas

These are the bibliographical notes (**?**)

En el próximo capítulo. . .

This is the next chapter section

Chapter 3

The Program Implementation : Architecture

Abstract: In this chapter we provide a technical analysis of our tool, examining how it works and what was designed for at a programming level, aswell as its module structure.

3.1 The Program Module Structure

3.2 The JSON Profile Storage Structure

Another kind of information is stored about the domains. This is the information concerning the **human** side of things, that is, **how** to interpret these stats and turn them into something that humans with different levels of familiarity can understand.

To do this, we provide use another storage class that will contain human-relevant data that will modify the objective comparison delivered by the analysis module.

A system of profiles is added to the object itself, inside a "profiles" key. The domain associated is clear as they share the same "attributelist" identification system.

The information contained in each one of these "profiles" serves two different purposes. It provides customizable elements of *how* the data will be presented to the user, and it keeps an historical record of this user's dataset results (similar to the one in the domain storage) making a historical following of a profile possible.

For each domain, there's a *default* profile. This provides a way to present the data when no previous knowledge of the profile is available. The automated processes of obtaining this profile and tuning the existing profiles from user feedback will be explained in further chapters.

3.3 The Profile Storage Module

This module makes use of the tools provided by our Storage Module, the class itself extends the Storage class providing the extra methods needed to use the "profile" system. It has the same methods as that class but also provides a way to change or load a single profile, its functioning mirrors the Storage

3.4 The Comparison Metrics

The purpose of these functions is to provide a way to measure the properties of a given dataset or knowledge domain. We can categorize them as follows: First the "measurement" metrics, used to get the information of a single dataset or domain.

- Dataset Metrics : they concern the dataset as a whole, like number of rows with missing values. dm :: (ds) –> num

- Single Column Metrics : they concern a certain column, and are based on the type of the column. For numerical columns we will have things like median, averages, deviations, distributions... For categorical columns we'll work with frequencies and things of the sort. scm :: (col) –> num

- Multiple Column Metrics : we will be looking for correlations and things of that sort. scm :: (col,col) –> num Time based metrics will be defined from this construct.

We will also have "comparison" metrics, used to compare datasets against their domains. These metrics will compare the output of two measurement metrics, both will have to spawn from the same function. compm :: (metric) –> num

Note that these metrics are not to provide "meaning" or any human-readable input, nor to be inherently comparable between each other outside of a framework of understanding of the domain (metric importance).

A mean to convert these machine cold metrics into human understanding will be provided in further modules. For now, we're not taking humans into account.

3.5 The Analysis Module

The analysis module will receive a dataset, use the Storage module to load its information, then analyze the dataset, which generates a similar object to the domain json, then producing a comparison of both.

The main methods for this module are *getstats*, *getcolumnstats* and *getdatasetstats*.

The *getstats* method is just a wrapper for the other two, calls them both and stores its results inside the Analyzer class.

Both *getcolumnstats* and *getdatasetstats* compute the statistics for the given dataset. If there is previous knowledge of the domain, the stats that appear there are computed for the domain. If not, a standard set of frequencies for categorical values and medians and distributions for numerical values are calculated and used to populate the stats object.

The most important method is *getanalysis*. Once the stats for the datasets have been generated, if there's previous knowledge available the class runs an analysis comparing the metrics of the two, and generating an object with the result. For this to happen, each metric defined for the dataset must have an associated *comparison metric*.

If there is no previous knowledge then the dataset stats are passed along to the reporter with a field indicating that there was no previous knowledge.

In any case, at this level we've already filtered what is relevant and what is not from the comparison.

3.6 The Report Generation Module

This module stands at the edge between the backend and the frontend. It receives the information from the comparison between the dataset and the domain knowledge and extracts the relevant profile information.

Once this is done, it uses both to generate a report with all the information from both sides. The user-relevant info will modify what is shown and *how* that is shown, changing the graphical elements according to the user so the frontend modules are able to be logic-free.

It is able to directly modify the profile information by the proxy methods *modify* and *savehumaninfo*. Its main method, *generate*, will create and populate an attribute within itself called report.

This method is called when the class itself is generated but the report can be modified as any Python attribute if needed.

At this point, we have an object that represents the dataset compared to the historical data and data about the profile associated with the user. This information, however, is in the form of a JSON object and is not really human-readable. The job of the frontend modules is to take this information and turn it into something easy to understand.

Notas bibliográficas

These are the bibliographical notes (?)

En el próximo capítulo. . .

This is the next chapter section

Chapter 4

Seed Cases and ELO Tournament

Abstract: In this chapter we outline and explain the our techniques used to ensure new users and domains have a valid starting point from which users can productively use the tool.

4.1 Motivation

Our objective for this chapter will be to provide details on the process chosen to develop a solid knowledge basis on which we can build a use case for our program. What we mean by this basis is the knowledge of which metrics to use to provide a clear picture of the datasets belonging to the domain, as well as which metrics are relevant to a profile and how should they be shown, as with a graph, through a text report, which colors, etc. We will be using the input of an expert to determine the metrics to use, and will then generate a seed profile based on an ELO tournament with a pool of experts

4.2 Metric Seeds

Our objective for this chapter will be to provide details on the process chosen to develop a solid knowledge basis on which we can build a use case for our program. What we mean by this basis is the knowledge of which metrics to use to provide a clear picture of the datasets belonging to the domain, as well as which metrics are relevant to a profile and how should they be shown, as with a graph, through a text report, which colors, etc. We will be using the input of an expert to determine the metrics to use, and will then generate a seed profile based on an ELO tournament with a pool of experts

Notas bibliográficas

These are the bibliographical notes (**?**)

En el próximo capítulo. . .

This is the next chapter section

Chapter 5

CBR Application : User Input

Abstract: In this chapter we outline and explain the user side of the Case Based Reasoning methodology used to test our tool with humans and to better define its ideal workflow. We provide both an abstract approach and the decisions taken to make the implementation.

5.1 CBR Applications

This is the introductory texttt

Notas bibliográficas

These are the bibliographical notes (**?**)

En el próximo capítulo...

This is the next chapter section

Chapter 6

Proof Of Concept Medical Data

Abstract: In this chapter we see all of the pieces of our program come together and work in the analysis of a set of medical data. We will follow a step by step execution from the different user viewpoints and the program's viewpoint.

6.1 introduction

This is the introductory texttt

Notas bibliográficas

These are the bibliographical notes (**?**)

En el próximo capítulo...

This is the next chapter section

Appendix A

Así se hizo...

...

...

Abstract: ...

A.1 Introducción

...

–¿Qué te parece desto, Sancho? – Dijo Don Quijote –
Bien podrán los encantadores quitarme la ventura,
pero el esfuerzo y el ánimo, será imposible.

Segunda parte del Ingenioso Caballero
Don Quijote de la Mancha
Miguel de Cervantes

–Buena está – dijo Sancho –; fírmela vuestra merced.
–No es menester firmarla – dijo Don Quijote–,
sino solamente poner mi rúbrica.

Primera parte del Ingenioso Caballero
Don Quijote de la Mancha
Miguel de Cervantes