Intelligent Price Comparator

Intelligent Price Comparator Software for E-Commerce

Jhonny Alexander Aldeia de Jesus
jhonny.aldeia@tecnico.ulisboa.pt
Instituto Superior Técnico

May 2015

***Abstract*—Price Comparators are tools that help internet users identifying product prices obtained by different e-commerce stores. These comparators help perform shopping, finding the best deals and offers.**

Thus, we intend to implement a Intelligent Price Comparator using a Data Mining (DM) technique, that searches for products in various e-commerce stores and finds results of similar products.

This project aims to evaluate existing technological capabilities, integrating internet information extraction and data analysis with DM techniques in price comparators, creating mechanisms that allow us to perform different approaches. The evaluation will be made based on a comparison where we highlight the differences, advantages and disadvantages of the intelligent price comparator and current market price comparators with existing methodologies that support the veracity of the results of the techniques applied.

***Keywords*— Web Crawler, Web Spider, Web Scraping, Data Mining, K-means, Internet, Price comparators, Service search.**

I. INTRODUCTION

A. Motivation

The high number of similar products on the market, competitiveness to be sold and the demand of these becomes an increasingly difficult task due to the quantity and diversity of choices to the customer. There is a great diversity of mechanisms that facilitate these tasks, for example, integrated search engine in online stores like: Fnac[1] and Worten[2], which help and provide information on desired products by the customer. On the internet we can find price comparison tools, which will depend further. The Price Comparators, are intended to look for products extracted from e-commerce stores, to later compare the price, and/or filter other features.

Price Comparators are tools to compare prices of products and exhibit different lists prices for products of the same genre. Price Comparators sites not selling products, as the online shops, but are product aggregators of these locals.

B. Objetives

The objective of this thesis consists to develop an Intelligent Price Comparator using a technique of Data Mining (DM), which search for products in various e-commerce stores of way to group products with similar characteristics. In turn, the Intelligent Price Comparator uses different approaches. Includes techniques of Web Scraping to extract information from the e-commerce shops, and Clustering as the Data Mining(DM) technique chosen for data analysis. Thus, the user selects directly the features of the required product, being presented as result, similar products or equal to the indicated features.

II. RELATED WORK

A. Price Comparators

Based on various searches on the internet, among them articles that evaluate the popularity of various price comparators [3][4][5][6][7], choose to study the Price Grabber[8], Buscapé [9] e KuantoKusta [10], for this international popularity and in Lusophone community (Buscapé e KuantoKusta).

The price comparators can provide *Application Programming Interface* (API)s that allow to query information for development of external applications and/or *Web Services* access points. In addition, they offer the means to load the information of the *e-commerce stores*. This process is called *Data Feed*.

B. Extraction of data from the internet pages

To extract data from internet, there are a few steps to take into account, starting with a set of Uniform Resource Locator (URL)s that we may exploring, hyperlink by hyperlink, until reach the desire web pages.

1) Web Crawlers: Web Crawlers [11] [12] is a Bot that systematically navigate the *World Wide Web* (WWW). Crawler starts with a list of URL's also called *Seeds*. As the Crawler visits these URLs, it identifies all the hyperlinks in the page and adds them to the list of URLs, and in turn this will be visited recursively. There are several programs for Crawlers which are already developed, such as*: Wget* [13], *Methabot* [14], *Googlebot* [15], *Msnbot* [16]. Each one fulfils its function like Crawler specialized and optimized in different tasks, for example: ***Google Boot*** is specialized in index pages of internet in his search engine***, Msnbot*** it's particularly used in the search engine from MSN, ***Wget*** and ***Methabot*** are Open Source tools that perform an optimized crawler to search for text faster and more streamlined, avoiding concerns over Hypertext Transfer Protocol protocols (HTTP), Hypertext Transfer Protocol Secure (HTTPS), Secure Socket Layer (SSL), and Internet Protocol version 6 (IPv6); The **Methabot** has the particularity of being

able to convert *HyperText Markup Language* (HTML) to *Extensible Markup Language* (XML).

2) Web Spider: Web Spider[17][18] is a crawler that visits URLs in *web* pages in order to create a list of URLs, process that is made simultaneously to visit all the URLs so as to create a web of all these.

3) Web Scraping: Web Scraping [19] is an information extract technique of information from the Web page (which is adopted by most search engines), which through the use of a web crawler transforms unstructured data on the web (typically pages in HTML format) in structured data that can be stored and analysed in a *DataBase* (DB).

4) Tools for *Web Scrapping:* Among the various existing tools for *Web Scrapping,* i opted for *Jsoup tool,* because it has similar semantics to the *jQuery* library and, from the selectors *Cascading Style Sheets (CSS)* with which find myself familiar, while the *Web-HarvestIt* has its own syntax. *Jsoup [20]* is a Java library for working with HTML, this provides an API for the extraction and manipulation of data, using the best of the *Document Object Model (DOM), CSS* and *jQuery-like* methods. *Jsoup* implements the *WHATWG* specification [21] *HyperText Markup* Language v5 (HTML5), and analyses HTML for the same DOM as modern browsers and is designed to handle all varieties of found HTML and in turn creates a parse tree.

C. Data Analysis

The technique of Data Mining (DM) chosen for analysis and data grouping of products was the Clustering, and the algorithm for this was the Incremental k-means, a variant of k-means which has the task of presenting the best grouping data, so, the best cluster.

The use of this variant is due to the application of an evaluation function termed F(k). The function F(k) evaluates which is best number of clusters in a particular grouping of data, ie, the selection of a optimum value of K.

1) Clustering: Clustering [22] is a technique that involves finding a collection of data that some way are similar, a cluster being a collection of objects that are similar to each other but distinct from other objects from another cluster [23].

The goal of creating a cluster is to determine a set of objects (data), figure 1, that does not have a feature set that resembles them. There is no criteria to define what constitutes a good group, which is defined by the user meet their needs [23].

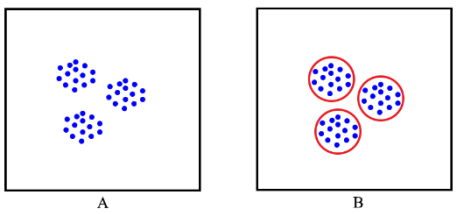

Fig. 1: Clusters

Incremental K-means

The randomness of K-means can be a problem when the groups should contain similar data between them. This is due the fact that initialization of k-means is made randomly choosing centroids to create clusters with similar data, long apart to each other. The start up of the centroids and a choice of a optimum k (number of *clusters*) can help solve this problem. There are possible approaches to solve this problem, such as *Incremental K-means.*

The dependence of randomness in *K-means* causes the existence of the possibility of creating groupings of data less desirable. The *Incremental K-means* is a variant of *K-means* that solves this problem by creating more consistent results, aiming to standardize the distribution of data by clusters, reducing the 2a distortion error.

$$I = S - N[d(w, x_0)]^2$$

(a) Distortion error

$$w_k = \frac{1}{N_k}\sum_{i=1}^{N_k} x_i^k = \frac{1}{N_k}(N_i w_i + N_j w_j)$$

(b) New centroid wk calculator

$$N_k = N_i + N_j$$

(c) Number of objects from k

Fig. 2: Incremental K-means

The difference between the *Incremental k-means* and k-means, lies in the boot and creation of clusters incrementally. The *Incremental k-means* is started by assigning a value to K, starting with *k = 1* and increasing 1 value in each iteration until it reaches the value of **K** assigned initially. As *K* is less than 2, the first centroids are chosen randomly, when k is greater than 1, one centroid is created in each iteration but randomly, this centroid is calculated based on the distribution of *clusters* and existing data, in order to maintain a uniform distribution of data. *Incremental k-means* includes a skip operation during the learning process, to correct the position of the worst placed cluster. The operation deals with this problem by removing the centroid of a cluster of an improper position, and creates a new 2b centroid that is inserted in the worst cluster (which is worse distortion of error) in each iteration by inserting it into a more promising position. The distortion of 2a error is presenting irregularities in the data distribution, evaluating if the cluster is in a bad position, that is, how the distortion of the error does not converge to a value, this correction will be made to stabilize the algorithm.

In the figure 2 above, the **N** represents the number of a cluster objects, **S** is the sum of distortions of clusters, **Wk** corresponds to the cluster centroid **k** and **x0** is the centre of Euclidean space, which represents the average of the distances of all objects. The

indices **i** and **j** represent the index of the worst cluster and closest to the worst cluster, respectively.

The algorithm consists of the following steps [24]:

Assign K = 1.

Phase 1 (Normal training)

1) If K = 1, choose an arbitrary point in the centroid of the cluster. If k>1, insert the new cluster centroid in the cluster with the highest distortion;
2) Assign each dataset object to the nearest cluster and updated their centroid (similar to k-means);
3) If the cluster centroid doesn't move, goes to step 2. Otherwise, goes to phase one, step 2;

Phase 2 (Increased clusters)

1) If K is smaller than a specific value, increase K by 1 and goes to Phase 1, Step 1. Otherwise, stops.

Choice of ideal K

The value K serves to indicate the number of *clusters* in a given data set, and the choice of this is very important for the quality of results. If, for a given population K is too small, the distribution of data based on its characteristics can not be correct because form will be few *clusters* containing data that may not be similar. In the case of K be very large, we would be creating too many *clusters* with similar data spread across multiple clusters, which does not make much sense, since the objective is to separate the similar data as close as possible for *clusters*. Considering this problem, the approach used to find the ideal, was the implementation of the F(K) function [25], testing various values of K, to determine the best number of *clusters.* F(K) function, may reveal more than one valid value of K. The algorithm of this function should start at K=2 and increment by one, until the result of F(K) converge to a value closer than 1.0, so, the algorithm stops. If K was 1, the value of F(K) was 1.0. If the initial values of F(K) for each K are to much different, the algorithm has to increase k until it begin to converge. When the initial results of F(K) starts to stabilize (to converge to 1.0) and appear different values to the converging value, this is a possible ideal K.

$$f(K) = \begin{cases} 1 & \text{if } K = 1 \\ \dfrac{S_K}{\alpha_K S_{K-1}} & \text{if } S_{K-1} \neq 0, \forall K > 1 \\ 1 & \text{if } S_{K-1} = 0, \forall K > 1 \end{cases}$$

Fig. 3: Function F(k)

In use of F(k) function it is important that the output ranges as least as possible when K remains unchanged. Therefore, the *k-means* algorithm must obtained consistent results to produce the best results of F(k) function performance evaluation . Thus it was previously introduced variant *Incremental K-means* that solves this problem, since the main k-means deficit is the dependence of randomness.

$$I_j = \sum_{i=1}^{N_j} [d(\mathrm{x}_{ji}, \mathrm{w}_j)]^2$$

(a) Distortion of cluster

$$S_K = \sum_{j=1}^{K} I_j$$

(b) Sum of Clusters of clusters

$$\alpha_K = \begin{cases} 1 - \dfrac{3}{4N_\mathrm{d}} & \text{if } K = 2 \text{ and } N_\mathrm{d} > 1 \\ \alpha_{K-1} + \dfrac{1 - \alpha_{K-1}}{6} & \text{if } K > 2 \text{ and } N_\mathrm{d} > 1 \end{cases}$$

(c) Weight factor of the data

Fig. 4: Additional functions for calculating F(k)

Initialization of F(k) algorithm may vary depending on how it intends to end. We can choose to finish at a fixed value, or stop the algorithm when stabilize to converge to 1.0. Briefly, the F (k) algorithm follows the next steps [25]:

- **Step 1**: Initialize k = 2, and increase value by 1 until the algorithm ends.
- **Step 2**: At the start of the algorithm, if the values of F (k) do not stabilize as to converge to 1.0, ignore the results of F (k).
- **Step 3**: When the values of F (k) stabilized as to converge to 1.0, the algorithm ends.
- **Step 4**: For each k, the values of F (k) that are different and after stabilization at the completion of the algorithm, will be considering k optimal. It is understood by values of F (k), considerable variation as the converging value.

In short, a trend or irregular behaviour means that the function value F (k) is too high or too low value for which is converging, and this is how the value of K is chosen.

III. ARQUITETURA

A. Architecture

Intelligent Prices Comparator is composed by two major modules: *PC Core* (JAVA developed) and *PC Interface* (*Hypertext Pre Processor* (PHP) and HTML developed), as shown in figure 5.

PC Interface (*Price Comparator Interface*) is the module that deals with the interaction with the user, the main features communication with *PC CORE* (*Price Comparator Core*) and the Internet. *PC Interface* processes the user-entered data, communicates them to *PC Core*, and receives in response a list of products. Communication with the internet takes place when the user selects one of the listed products, being redirected to the specific product store, as can be seen in Figure 5, that represent the communication between *PC Core* and the *Internet.*

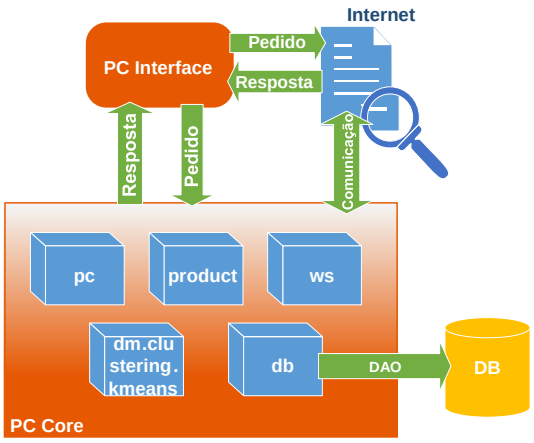

Fig. 5: Intelligent Prices Comparator

PC Core, is compound by the following *packages*:

- *Package* ***pc*** contains the Main function of the *PC Core process*, auxiliary functions for pre-processing of data extracted from the internet and provides communication for the data transaction between *PC Interface* and *PC Core*, via *exec* of PHP process.
- *Package* ***product*** is in charge for the organization of data products and communication with DB by the *Package db*.
- *Package* ***ws*** behaves the responsible features for the Web Scraping of e-commerce stores.
- *Package* ***dm.clustering.kmeans*** combines all for calculator functionalities of clustering by the k-means algorithm.
- *Package* ***db*** *presents a data abstraction layer Direct Access Object* (DAO) to handle with DB, running the applications in *Structured Query Language* (SQL) and representing the results in objects.

PC Core has two types of behaviours: The Mode 1 (Web Scraping), and Mode 2 (Setup and Execution). Mode 1 is for search and extract data from e-commerce stores on the Internet, and Mode 2, for setting up and running the program data. Mode 1 needs to run at least once, in order to collect data and create a population of products from online stores. If we want to update the DB, we can run Mode 1 again.

B. PC Core

PC Core contains the main features of Intelligent Price Comparator, the process to extract data from *online* stores, pre-processing of the obtained data, database queries, *clustering* implementation, and communication with the interface. As stated above, this consists of two main runs, one for the extraction of data from stores (Mode 1), and another for the configuration and normal program execution (Mode 2).

The implementation was developed in JAVA, because the tools used, provide APIs for this language, just as the acquired *k-means* [26], Articles based [22] [27], and adapted for the variant *Incremental K-means* [24].

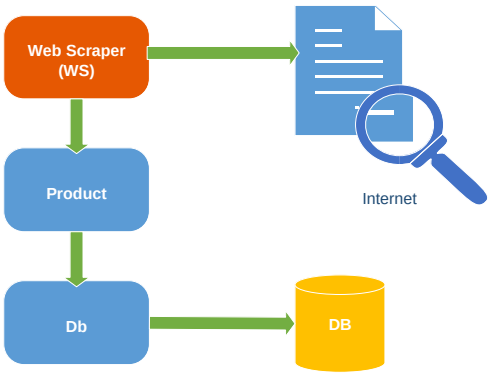

Fig. 6: *PC Core* - Mode 1

1) Mode 1 - WebScrapping: Mode 1 presented in figure 6 was designed to run *Web Scraping* and consists of *packages* ***ws, product***, e ***db.***

The product ***package***, aims to map the stores, products, and characteristics of the products, using the data abstraction layer from the *package* db. In this configuration, the WS uses the features of the product, to save the extracted data in a database drawn with a Relational Model (RM) simple and well defined.

O *Web Scraper (WS)*, by *Jsoup* tool, runs a *Web Spider* that searching in stores, by products *hyperlinks,* creating a summary of URLs. At the end of this process, the WS uses hyperlinks raised to analyse the structures of the pages, through pre-programmed functions for each type of store, and proceeds to extract the necessary information where this data is mapped by the features of the *product package* and stored in database. This process is launched by *PC Core application*, and the way in which it operates, is configured in Class *Main* found in *package pc*.

However, the process that i will describe, will run for a set of methods: ***execute, spideringURL, scrapingURL, getProductName, getProductPrice, getProductImgURL,*** and ***getProductFeature***, belonging to package Class *WebScraper* provided by *ws package*.

When *PC Core starts* in Mode 1, the execute mode is invoked, which makes two operations. First, invokes the *Web Spider* **(spideringURL** method) building a summary of hyperlinks stored in *ArrayList<String>*. The second, invokes the process of *Web Scraping through scrapingURL method*, for each URL collected by *Web Spider*.

The data extraction was not designed to be automatic, and the system design done to adapt easily to each store, requiring rewriting methods: *getProductName*, *getProductPrice*, *getProductImgURL*, and *getProductFeature*, using *Jsoup*, to the structure of each store. The methods described above are used by *scrapingURL method*.

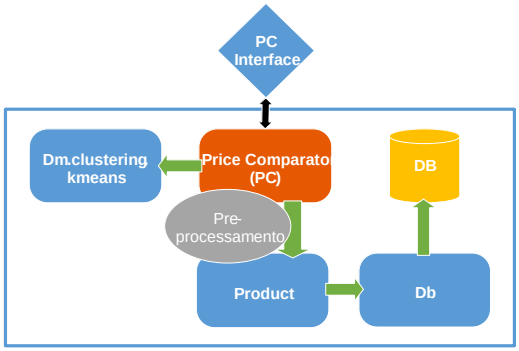

Fig. 7: *PC Core* - Mode 2

2) Mode 2 - Setup and Execution: Mode 2 is in charge of the main program execution, and for that need to communicate with the user interface by using the arguments received by the *main* function of JAVA (*package pc),* via the execution of exec PHP function, which calls the Main.class process with the parameters sent by the user interface.

Configuration

Price Comparator (PC) must be configured before being executed normally. This configuration in done by executing the fk.php file *script, consisting in analyse the data collected in Mode 1 and create clusters* for this data. This information is saved automatically in "KClusters.pc" file. This file, in figure 8, is a Java serialized data file, which keeps the data structure of the created clusters, found in the "bin" folder of the PC Core module, Figure 8. If we want new calculations, we can delete the file to able to make a new configuration, rerunning the *script* fk.php file.

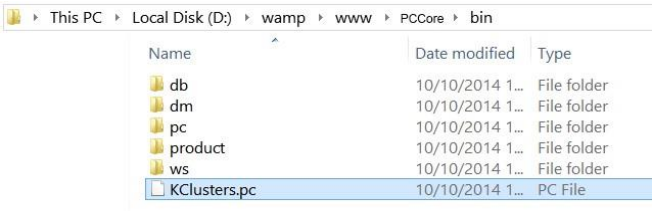

Fig. 8: Ficheiro KClusters.pc

The "fk.php" file it´s located in *PC Interface,* figure 9, and can be executed performed for example as follows: "http://localhost/PCInterface/fk.php".

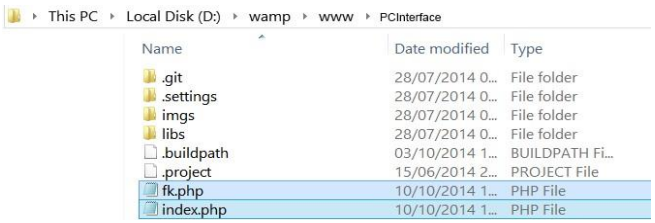

Fig. 9: fk.php file

Made this setting, the Price Comparator (PC) is prepared for normal program execution, so when a user does a search, the Price Comparator (PC) interprets the received data by the interface, create a new point with normalized values and seeks the cluster centroid closest to that point, which results that the cluster identifies similar associated products.

Execution

After the configuration the main execution of the program though the pre-processing and data analysis. The pre-processor is to clean the extracted data for them can use, and data analysis covers the creation of clusters.
Data analysis is performed after the pre-processing in the same *detailedClusteringKMeans* method of *Class PriceComparator*, **pc** package. The package *dm.clustering.kmeans*, contains all the implemented algorithm, including the evaluation function F (k). When the PC Core is in Mode 2, and the program has not started, must been created clusters for the data of population.

Price Comparator process sends the data prepared for the process "Clustering Kmeans". This process proceeds to read "KClusters.pc" data file, because contains information of clusters created for each K, and if it doesn't exist, proceed to implementation of Incremental K-means and F(k) function, then featuring on the screen, a list review of the F(k), where each row shows a K and F(k) result assigned to this K.

Based on the stabilized clusters and assessments presented on interface, we proceed to choose the k value that suits. Then, configure the variables in ClasseMain code, defining K in which operates the PC Core.

Incremental k-means algorithm model, consists in five classes: *Point*, *Cluster*, *Clusters*, *IncrementalKmeans* e *Fk*.

- Class *Point*, corresponds to the point that represents a cluster object, this point is of N dimensions, where N is the number of product features, in addition to this information also identifies the products for their respective Identifier (ID).
- Class *Cluster*, keeps a list of points *List<Point>*, its centroid (*Point*), methods to update the centroid and get the list of centroid points.
- Class *Clusters* corresponds to an *ArrayList of Class Cluster: Clusters extends ArrayList<Cluster>*. It contains methods to update the clusters, assigns points to clusters, and know the closest cluster to a point.
- Class *IncrementalKmeans runs the algorithm using all the features of classes the above headings.*
- Class *Fk performs the evaluation function, and takes* as argument in the constructor a list of clusters: *List<Cluster> clusters*, and k value used for the cluster list created.

C. PC Interface

The program has to types of interfaces, one will be merely exploratory, referred to as "Absolute Search", used to simulate the same behaviour of the existing prices comparators. The other interface, "Search for Parameters" is that which corresponds to the Intelligent Prices Comparator in this, the search of product is made through the choice of product characteristics of a unique genre for the study that will address

below. A1 Interface (PC Interface module) is implemented in PHP and HTML with jQuery Mobile library support.

D. Database (DB)

Database (DB) is constructed on MySQL. This is designed in a Relational Model (MR) and consists of three tables which to store the data drawn of online stores, which correspond to the study population.

The tables were organized into three types: *shop*, *product* and *features*.

- ***shop*** *this table correspond to ecommerce stores lists to be studied in this project. Is consists of three fields: an identifier that relates to the products the store, the name and URL of respective store.*
- ***product*** the table consists of a handle to associate the characteristics of the extracted products in the table *features*, a name the is part of the title presented by the merchand web page where the product, the price, and URLs, where one is for the image, and forwards the next product to be displayed on the Intelligent Comparator interface to the respective store.
- ***features*** *is the table that represents the type of character and the associated value, either numeric or text, although for this study will only consider numeric values*. It's worth noting that stored data at time of extraction, have not gone through a cleaning step (pre-processing), except price (inserted in the product table). This happens because at the design of this model, prices were assumed to be numerical. This happens because at time of this model design, prices were assumed to be numerical, but the stores websites, these comes with the euro currency symbol, comma and / or points of decimal places. For all other cases, as mentioned earlier, the features as they contain wide variety of textual and numeric information, the cleaning is done later in the clustering configuration (pre-processing stage.

E. Communication between PC Core *and* PC Interface is User Interaction

Communication between *PC Core* and *PC Interface* should only be performed when the PC Core is in Mode 2. If is tryed any interaction with the interface, and the PC Core module is not compiled or is compiled in Mode 1, will not be got any kind of response.

When is desire to perform a request of search from *PC Interface*, it is executed a PHP sequence, and forwards the data to *PC Core*.

When a request is received by *PC Core*, this evaluates the data and generates a response in a parameter string as follows:

```
System.out.println(obj.get("shop_name")+"#
    $#"+obj.get("price")+"#$#"+obj.get("
    product_name")+"#$#"+obj.get("url")+"#
    $#"+obj.get("url_img")) ;
```

Fig. 10: Data structure sent from PC Core

Where "#$#" corresponds data tabs. This string is interpreted by *PC Interface*, getting it in the $output variable, for display data in the interface.

However, if we execute the script of "fk.php" file, of the PC Interface module to configure the system, displays by the $output variable printing, a list of F(k) rating for different K clusters analysed.

IV. CASE STUDY

The case study will focus on discussion among the Prices Comparators existing in the market and the Intelligent Price Comparator. On that basis, the starting point was the analysis of the functioning of the existing Price Comparators, ie, as they present the products to the customer, as get information from e-commerce stores, the selection and presentation of research of a product, in has the desired characteristics.

Due to the large variety of products with different characteristics in the online market, it was decided that the most appropriate population for the study would be televisions (TV's) as they are a product of easy access, and that is in any *e-commerce* store. These have characteristics with similar designations in different selected stores, which facilitates the processing of information. E-commerce shops study are Worten [2] and Fnac [1], since they have a structure of the HTML document and organized, which facilitates the extraction of data.

A. Study variables

In an analysis of the project, taking into account the **televisions** as chosen product, were chosen as variables study, the following: **price, size diagonal screen**, **power consumption in standby mode** (***standby***), and number of ***High-Definition Multimedia Interface*** **(HDMI)** ports, as are common in the population study. These features were chosen for being easier to apply clustering in numeric variables. Clustering is a mathematical and statistical methods to group data in a Euclidean space of points of N dimensions, of which have to calculate distances between them for the first grouping data widths (clusters) allowed, and these two dimensions are chosen respectively the characteristics of the product, ie, in this case N has the value of 4, this means that to Data configuration, will be represented in a multidimensional space 4 of 4 dimensions.

For this case study, we generated a population of approximately 500 individuals, extracted from 6 online stores mentioned above, which subsequently needed pass by a cleaning process (pre-processing).

B. Data Extraction

The data extraction process consist of two parts: Web Spidering, and Web Scraping. Before starting the process, it was necessary to analyze the internal structure of HTML document of each store, for know how is formulated, this is made through the Google Chrome browser debugger. Note that this process is not fully automatic, because would approach Natural Language (NL) paradigms that would take us to another job. Following this analysis, it was necessary to understand how the data were

structured and their respective hyperlinks. For this task I used Jsoup[20] who had done the work *Web Spidering* and *Web Scraping*, taking as its starting point the *URLs* of each online store, and so, explore the HTML document these, creating hyperlinks summary of products, which then will travel one by one by applying the Web Scrapping technique. This procedure is slow, which prevents updating the data in real time on the DB, for each user iteration, new data would involve making the adjustment operations grouping (clustering) including waiting for the own data extraction.

The following code portion, 11 and 12 is a sample implementation of the method ***getProductPrice***.

```
@Override protected String getProductPrice(Document doc) {
    Elements price1 = doc.select("div.product
            -essential") ; price1 = price1.select("div.price-box") ;
    Elements price2 = price1.select("p[class= special-price]") ;
    if(!price2.hasText()) price2 = price1.
            select("p[class=regular-price]") ;
    price2 = price2.select("span[class=price] ") ;
    System.out.println("PRICE: "+ price2.text
            ()) ; return price2.text().replace("\euro", "")
            .replace(",", ".") ;
}
```

Fig. 11: The method Product Price - Worten[2]

```
@Override protected String getProductPrice(Document doc) {
    Elements price = doc.select("div. blk_inside.filled.pdg_v.txt_c") ;
    price = price.select("span.price") ; String priceText =
    price.text().replace("
            \euro", "").replace(".", "") .replace (
            ",", ".") ; priceText = Jsoup.clean(priceText,
            Whitelist.basic()); priceText = priceText.replace(" ", " ")
    ;
    System.out.println("PRICE: "+ priceText); return priceText;
}
```

Fig. 12: The method 'Product Price' - Fnac[1]

The ***getProductPrice*** method is in charge to extract the price of product*, and make a small* pre-processing, wiping characters of the currency and replacing the commas by points, being commas the decimal systems used in our language, in this case Portuguese, and the point corresponds to the decimal system programming language (floating point). This may serve as an example of a detail NL. In short, each segment of data extracted, are mapped in each iteration of ***scrapingURL*** method as a "product". The functionality for mapping are provided by the *product package*, which in turn store data using the DAO abstraction layer package *db*.

C. Pre-processing of data

The pre-processing consists in a treatment of the data collected, and prepares them to be able to be manipulated in this case, unwanted text cleaning and normalization of values. The type of election with numeric source values aims to expedite the study process and are not yet enough to think that the extracted data are in good condition, for use in mathematical and statistical calculation assessments.

Also, were found some obstacles in the population data, which can be solved, but on the other hand can influence the results. Further, explain how it was addressed and solved this problem.

Pre-processing is always started when runs the ***detailedClusteringKMeans*** *method* of Classe *PriceComparator*, *package* ***pc,*** and the absence of "KClusters.pc" file. This happens when *PC Core* operates in Mode 2 on configuration phase. Recall that this phase takes place in the first run of the program or by removal "KClusters.pc" file, and serves to create a data grouping, creating clusters with a certain K evaluated, storing this information in the "KClusters.pc" file that will always be used for carrying out a survey in Intelligent Prices Comparator.

The first debugging effected in data, was to associate the same names to similar features, which contained different words or letters. This solution is used to identify a feature under different designations and associate them as a unique name, for example: "Diagonal of Screen", "Size (visible screen)" and others, that means the same, in which case, was classified as "Diagonal".

In ***detailedClusteringKMeans*** *method* of Classe *PriceComparator*, contains the code of this approach that is made at initial query (SQL), which will serve to facilitate the data standardization and evaluation, which will then be mapped into points in a multidimensional space, to be analysed with the *Incremental K-means algorithm*.

After the first debugging, a second aproach found (also performed in ***detailedClusteringKMeans*** *method* of Class *PriceComparator*), which performs cleaning of data corresponding to the characteristic values in order to obtain only the numeric part, and removing unnecessary text, which can't has been added by the retailer as references. For example, the screen diagonal is accompanied by a number of inches symbol ", the standby power is accompanied by a W (Watts).

There are certain cases where the characteristics of products are not visible, causing values to null, this may mean that they do not really exist, their value is zero, or were not added by the merchant. Although few cases, they happen. The solution can interfere with the results, and so some care were considered.

Omissions applied to solve cases with null values:

- At the **HDMI**, in addition to the merchant forget to add, it may means that there are mainly HDMI ports. So for this value to be 0
- ***Standby consumption*** *there is always some kind consumption*, then we cannot say it's zero. In this situation it was considered the average of all the existing population: 0, 3.
- **The screen diagonal** always there! Just as in the previous paragraph we have the same problem. It can be considered zero. In use, the average value of the entire population of data: 40.

To end the pre-processing, we have to normalize the data. Standardisation is important and necessary for clustering

operations because the values have to be consistent. Again, this is performed in method detailedClusteringKMeans in *PriceComparator* Class, which is subsequently performed prior to pre-processing.

D. Data Analysis

This is the most important approach for the Intelligent Prices Comparator, it depends on this to choose the products to be displayed. We need to get consistent results, but above all, results that are close or equal to search.

It was decided to apply *Clustering in this case study, choosing as the clustering* technique to analyse and group the products, being the k-means algorithm chosen for its simplicity, computationally slight, and meets the requirements for the preparation of this study. The *k-means* is dependent on the choice of a K value which determines how many data packets will build. The problem with this algorithm is the dependence of randomness, and due to this reason, we use the variant incremental k-means [24]. The use of this variant is due to the method that we use to evaluate our choice of K, which is done by F(k) function [25]. F(k) function evaluates possible groupings better suited to population data, and indicates possible K's. When using F(k) function, it is important that the result varies as little as possible. Therefore, the algorithm must obtain consistent results, and thus, we use the *Incremental K-means*. O *Incremental K-means*, try to maximize uniformity in the distribution of clusters, and k-means no worries about that. For example, if we had a population of data, which have well separated data clusters, the *k-means*, do not guarantee the choice of a centroid for each of these clusters, but *Incremental k-means*, in each iteration can improve the choice of centroids insofar as K increases, so as to bring the centroid to achieve these groups.

E. Experimental Results

1) Results: The following graphs 13, 14, 15, represente the results of three runs on the same population of the case study data, using the variant *Incremental k-means and evaluation F(k) function*. The population comprises an approximate size of 500 televisons and 4 are the characteristics of the data in 4 dimensional clustering.

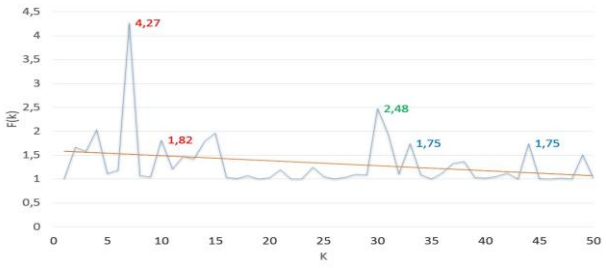

Fig. 13: Result 1

In this first result in figure 13, we can see the convergence of F(k) to 1.0 (orange line). Following the trend chart and taking into account that the algorithm started to stabilize at k=15, we can verify in the chart some fluctuations for the next K's. These fluctuations indicates where the best values for K: K=30 (green), it also being possible, as a second option, K=33 and K=44 (blue). The green is the best, because after the stabilization of algorithm, were the values of F(k) that showed greater fluctuations.

Fig. 14: Result 2

In this second result, figure 14, we can see many fluctuations in the chart beginning with F(k) values until 4.2. This fluctuations are not taken into account because the algorithm is not yet stable, starting to stabilize and converge in K=17. In this chart, the best K-values are K=31 (green), it also being possible, as a second option, the values K=22 and K=33 (blue).

Fig. 15: Result 3

In this third result, figure 15, so does the previous chart, there are very strong oscillations at first, but the graph begins to converge and stabilize sooner, which creates the possibility of smaller values of k. The best values of K are: K=23 and K=32 (green), it also being possible, as second option, the value K=26 (blue).

The graphics are different in that the first stabilizing begins immediately with a gradual convergence. The second and the third graphs are similar between them, but differ from the first, due the start, where we note the data had big fluctuations. The second chart had more later stabilization, as the third quickly stabilized after a very irregular start.

These fluctuations show the randomness in the boot of k-means, and in turn, shows the recovery that the incremental K-means is, to distribute evenly clusters, which can be seen in time the graph begins to stabilize.

In this case study, the values of K, can vary from K = 15 and K = 35, where K may be used any of those shown in the above graphs 13, 14, 15, variation of these K values for the same population to If the clusters created in different cases, as an update of the data, or even the implementation of the *Incremental K-means for the same* population on the same population of data several times (the value K always tend vary because the start of the algorithm is random)

2) Advantages and Disadvantages:

Taking into account all the case study, and the results below explain the differences between Intelligent Prices Comparator and the existing comparators.

Advantages of Intelligent Price Comparator face to existing comparators:

- It is fast for the user search for similar products because research becomes straightforward;
- The research is done in pursuit of the user want?, And not limited to the options presented;
- No loading data via files is required, as Data Feed processes made by other comparators, avoiding ask the shopkeeper, the data of your store;
- Products can easily keep up to date, it is not necessary to ask the shopkeepers any type of file or database to update, and these updates made directly from online stores.

Disadvantages of Smart Price Comparator face to existing comparators:

- It is necessary to mathematical and statistical calculations, creating mathematical functions, which can make the tedious process, turning something simple into something complex;
- The interface does not cease to be dependent on filtering;
- Computationally can become heavy, with populations of very large data;
- Very dependent on pre-processing of data. Example: an extracted number can be accompanied by a symbol of coin, or comma as the separator of decimal places (computationally point is the separator of decimal places);
- Data extraction is not easy to implement, especially if we are to fully automated and is very dependent on the HTML structure, which may be subject to constant changes, having to address NL paradigms. Example: we cannot know if a number is a price or other value, unless we have a grammar that identifies what can be (in the context of HTML).

V. CONCLUSIONS AND FUTURE WORK

A. CONCLUSIONS

In short, it implemented a Price Comparator that uses a data mining technique. This throughout the study underwent different approaches, these being the Internet Data Extraction, pre-processing and data analysis. The population chosen for study were flat, with about 500 products, with the following variables were chosen: **the price**, **HDMI port** number**, screen diagonal**, and **standby power**.

Among the different approaches, the pre-processing was more difficult due to irregularities in the extracted data. The most common cases of cleaning, added as text in a numerical value, have been solved without any inconvenience, however, there is another situation where products with similar characteristics, come in different denominations, which requires extra care in the treatment of information. Another annoyance was confronted me with null values, which are difficult to solve, because the product may not display a particular feature, or the shopkeeper forgot to put it in the presentation of the product. The solution was to assign them intermediate values or zero, depending on the situation.

For data extraction, we used Web Spidering visiting URLs in web pages in order to create a summary of the same URLs, and so reach the pages of the desired products. Later this process uses web scraping to extract the data of the page. The Web Spidering and Web Scraping, were executed by Jsoup tool that has a very familiar syntax to JavaScript, CSS and jQuery, and expedited the implementation process.

The data analysis was the most important, since the Intelligent Price Comparator entirely dependent on that for the presentation of data (products). This was done with a data mining technique to group data (Clustering), and used the Incremental k-means algorithm. The choice of the number of clusters, i.e., the number of possible for the data groups was made by a F(k) function that evaluates a set of *clusters* already created a distribution data.

In conclusion, the Intelligent Price Comparator, despite being responsive in searching for a particular product can sin in their performance in that the product data to increase, this is very dependent upon pre-processing the data and mathematical implementations. However, it has the advantage of not having to loading data, and updates of these are made directly from *online stores*.

B. FUTURE WORK

Based on this study and due to difficulties over implementations, it was possible to obtain knowledge of multiple situations, which have been omitted for simplicity but this study however, may give rise to other works. Throughout, the study mentioned problems where it is often necessary to Natural Language approaches, especially when information directly extracted HTML documents. Another problem was a limitation in automating the collection of data from *online* stores.

That said above, we consider the following cases for studies or future implementations.

Develop an application that automates the extraction of data, pattern recognition, in a certain language. For example, recognize products, identify the name, price, and features. This implies Natural Language approaches, and makes scalable data extraction for a comparison of prices.

Develop a program to adapt the text by using Data Mining. In the case of Smart comparison study, we chose only numeric values. It could make another approach including text, of course we would have to address Natural Language.

REFERENCES

[1] L. FNAC PORTUGAL ACDLDMPT, "Fnac," 2014, http://www.fnac.pt.
[2] S. WORTEN EQUIPAMENTOS PARA O LAR, "Worten," 2014, http://www.worten.pt/store.
[3] S. Inc, "Ecommerce marketing blog," 2014, http://www.shopify.com/blog/7068398-10-best-comparison-shoppingengines-to-increase-ecommerce-sales.
[4] I. I. M. LLC, "Search engine watch," 2014, http://searchenginewatch.com/article/2097413/The-10-Best-Shopping-Engines.
[5] digitalks, "Digitalks - comparadores de preços," 2014, http://digitalks.com.br/guia-de-empresas-categoria/comparadoresde-precos/.
[6] e commercebrasil, "E-commercebrasil - comparadores de preços" 2014, http://www.ecommercebrasil.com.br/fornecedorescategoria/comparadores-de-precos/.
[7] eBizMBA Inc, "ebizmba," 2014, http://www.ebizmba.com/articles/shoppingwebsites.
[8] I. PriceGrabber.com, "Pricegrabber," 2014, http://www.pricegrabber.com.
[9] B. Company, "Buscapé," 2014, http://www.buscape.pt.
[10] K. Kusta, "Kuanto kusta," 2014, http://www.kuantokusta.pt.
[11] P. Gupta and K. Johari, "Implementation of web crawler," in Emerging Trends in Engineering and Technology (ICETET), 2009 2nd International Conference on, Dec 2009, pp. 838–843.
[12] G. Jawaheer and P. Kostkova, "Web crawlers on a health related portal:Detection, characterisation and implications," in Developments in Esystems Engineering (DeSE), 2011, Dec 2011, pp. 24–29.
[13] G. O. System, "Wget," 2010, http://www.gnu.org/software/wget.
[14] E. Romanus, "Methabot," http://sourceforge.net/projects/methabot/.
[15] G. Inc., "Googlebot," http://support.google.com/webmasters/bin/answer.py?hl=pt-BR&answer=182072.
[16] M. Corporation, "Msnbot," http://en.wikipedia.org/wiki/Msnbot.
[17] L. Jiang and H. Zhang, "Multi-agent based individual web spider system," in World Automation Congress (WAC), 2010, Sept 2010, pp. 177–181.
[18] X. Han, X. Li, and Q. Zheng, "Research on web information extraction based on spider algorithm and dom thinking," in Information Networking and Automation (ICINA), 2010 International Conference on, vol. 2, Oct 2010, pp. V2–182–V2–185.
[19] S. Malik and S. A. M. Rizvi, "Information extraction using web usage mining, web scrapping and semantic annotation," in Computational Intelligence and Communication Networks (CICN), 2011 International Conference on, Oct 2011, pp. 465–469.
[20] J. Hedley, "Jsoup," http://jsoup.org/.
[21] Whatwg, "Whatwg," http://www.whatwg.org/specs/web-apps/currentwork multipage/.
[22] C. Antunes, Data Mining e Data Warehousing, 2nd ed., Departamento de Engenharia Informática, Instituto Superior Técnico, Março, 2008, pp. 87–97.
[23] http://home.deib.polimi.it, "Clustering: An introduction," http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/index.html.
[24] D. Pham, S. Dimov, and C. Nguyen, "An incremental k-means algorithm," Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, vol. 218, no. 7, pp. 783–795, 2004.
[25] D. T. Pham, S. S. Dimov, and C. Nguyen, "Selection of k in k-means clustering," Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, vol. 219, no. 1, pp. 103–119, 2005.
[26] https://www.blogger.com/profile/17797460307331045438, "A java implementation of k-means," 2014, http://moderntone.blogspot.pt/2013/04/a-java-implementation-ofk-means.html.
[27] K.-M.Clustering, http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/kmeans.html.
[28] I. PriceGrabber.com, "Pricegrabber documentation," 2014, https://partner.pricegrabber.com/mss_main.php?sec=2.
[29] B. Company, "Buscapé documentação," 2014, http://developer.buscape.com/pt/product/buscape/.
[30] ——, "Buscapé api," 2014, http://developer.buscape.com/pt/product/buscape/manualapi.html.
[31] ——, "Lomadee," 2014, http://br.lomadee.com/.
[32] K. Kusta, "Kuanto kusta api," 2014, http://kuantokustaapi.appspot.com/.
[33] G. Inc, "Google," 2014, https://www.google.com.
[34] Y. Inc, "Yahoo," 2014, http://www.yahoo.com/.
[35] ——, "Yahoo! store," 2014, https://smallbusiness.yahoo.com/.
[36] W3C, "World wide web consortium," 2014, http://www.w3.org/.
[37] T. W. Blog, "Google crawler figure," 2014, http://www.wpromote.com/blog/seo/helping-web-crawlers-help-youhow-to-create-an-xml-sitemap/attachment/google-crawling-sitemaps-2/.
[38] M. M and E. Jacob, "Article: Design and development of a programmable meta search engine," International Journal of Computer Applications, vol. 74, no. 5, pp. 6–11, July 2013, published by Foundation of Computer Science, New York, USA.
[39] T. Bruggemann and M. Breitner, "Mobile price comparison services," in Mobile Commerce and Services, 2005. WMCS '05. The Second IEEE International Workshop on, July 2005, pp. 193–201.
[40] W. Harvest, "Web harvest," http://web-harvest.sourceforge.net/.
[41] L. W. Extractor, "Link web extractor," http://www.linkws.com/br/marketing/extractor_apres.htm.
[42] LinkWS, "Linkws," http://www.linkws.com/.
[43] P. K. Agarwal and D. Hou, "Lecture 18: Clustering and classification," October 30, 2003, https://www.cs.duke.edu/courses/fall03/cps260/notes/lecture18.pdf.
[44] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, University of California Press, 1967, pp. 1:281–297.
[45] J. C. Dunn, "A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters," Journal of Cybernetics, pp. 3: 32–57, 1973.
[46] J. B. MacQueen, "Pattern recognition with fuzzy objective function algoritms," Plenum Press, New York, 1981.
[47] S. C. Johnson, "Hierarchical clustering schemes," Psychometrika, pp. 2:241–254, 1967.
[48] C. Abar, "O conceito 'fuzzy'," 2004, http://www.pucsp.br/ logica/-Fuzzy.htm.
[49] O. do Nascimento Souza, "Introdução à teoria dos conjuntos fuzzy," Março 8, 2010, http://www.ime.unicamp.br/valle/PDFfiles/osmar10.pdf.
[50] F. C.-M. Clustering, http://home.deib.polimi.it/matteucc/Clustering/tutorial_html/cmeans.[51] G. H. B. P. P. R. I. H. W. . T. W. D. M. S. A. U. S. E. V. . I. . Mark Hall, Eibe Frank, http://www.cs.waikato.ac.nz/ml/weka/index.html.
[52] G. G. P. License, http://www.gnu.org/licenses/gpl.html.
[53] R. Alves, "Declarative approach to data extraction of web pages," pp. 90–91, universidade Nova de Lisboa, Faculdade de Ciências e Tecnologia, Department of Computer Science. November 2009. http://run.unl.pt/bitstream/10362/5822/1/Alves_2009.pdf.