Predicting Preferences of Voters from Opinion Polls by Machine Learning and Game Theory

Guillermo De Ita Luna[1], Aurelio López López[2], Josué Pérez Lucero[1]

[1]Universidad Autónoma de Puebla, [2]INAOE - Puebla
deita@cs.buap.mx, *allopez@inaoep.mx*, *d.josue.pl@gmail.com*

Abstract. We present a method to determine political profiles of a voters population through the application of learning algorithms and game theory. Our process began with a collection of surveys gathered in two ways, from a website and directly from voters. Having a linear hierarchy on the attributes expressing the political preferences of the voters allow to apply a zero-sum game approach to distribute the potential number of votes among the involved candidates. We have applied our model to do electoral prospection in a case study of the Major's election of our city, that took place in July 2013. The results were quite encouraging. Our approach has also the potential to be applied to new product advertisement campaigns.

Keywords: Data Mining, Learning Algorithms, Recognizing Political Profiles, Linear Hierarchy

1 Introduction

Currently, computer systems that support decision making are present in many applications, as in the case of Intelligent Web Decision Support Systems, providing users with technologies that understand their preferences and make recommendations based on them. Examples of such systems are Your *Amazon.com* Store, *Auto Choice Advisor* of General Motors, *IBM Solution Profiler*, and so on [4].

One of the common processes of decision making, which has been subject of many studies, is 'the vote' in democratic system. Voting is a process that describes the way in which the preferences of a group of people determine a collective decision. Choosing a representative is a process with a high degree of uncertainty.

Nowadays, demoscopic studies (opinion polls) are performed intended to determine some electoral preferences. Those surveys, as snapshots of a moment, allow us to make predictions for a very short term. Because of the lack of analytic tools for studying electoral decision making, we propose the use of learning algorithms to determine the political profiles of voters.

Different methods have been proposed to model and simulate electoral trends, some of them based on game theory. For example, there are models simulating a political contest via a competition among intelligent agents, where agents are not cooperative and compete for obtaining the highest number of votes. There are different ways to simulate the process of obtaining votes by intelligent agents. One of them is by applying strategies in a non-cooperative game [2], [7].

Also, the uniform random sampling process has been widely used through opinion surveys as a means of electoral prospection. The size of the sample depends primarily on the population variance: as population is more diverse regarding the event to estimate, the size of the sample increases, regardless of the overall size of the population [5].

Random sampling also requires increasing the size of the sample at an exponential rate with respect to the probability of success of the event that is sampled. That is, the recognition of events with high uncertainty requires a much larger sample than those events with less uncertainty. For example, if one needs to recognize electoral trends, while higher uncertainty is in the voters to decide their preference by only one of the contending candidates, besides of increasing the sample size, one has to consider respondent's uncertainty about their preference, which can lead then to answer contradictory or imprecisely the survey questions.

The use of statistical Methods have been the main tool for doing electoral surveys, but it has not been clear its effectiveness for this social task because the involved actors had abused of that tool for spreading results to convenience. Furthermore, sampling can be improved by considering additional knowledge about the population to be sampled.

In order to obtain knowledge about the voters, we propose to build political profiles of voters from initial surveys. And for this purpose, we apply techniques from data mining on the answers of respondents, to characterize constituencies in favor and against each candidate. This implies in turn, the design and implementation of more targeted surveys to relate needs of voters with their political preference.

A learning algorithm becomes a prediction model which has been commonly used in the field of artificial intelligence. We present in this article the application of learning algorithms as a technique for extracting information from opinion polls.

So, our main contributions are twofold: an innovative algorithm extending the processing of information produced by the learning algorithms, to improve mining; and the application of a game theory approach to reach an expected vote distribution among candidates.

This paper is organized as follows. Section 2 discusses opinion polls and its presence in the Web. Section 3 describes how data mining is achieved in our context. The way that we extract a hierarchy of central attributes for revealing voters preferences is detailed in section 4. In section 5, we present how to apply game theory for computing the distribution of votes among the candidates involved in a contest.

2 Political Polls

An opinion poll, in its traditional elaboration form, usually reflects outlying questions about candidates, and about the political competition, such as: popularity indexes, perceptions on the nature of the candidates or their images, the impact of their campaigns, etc. Often the factors that are measured through those surveys point more to the interest of the candidates or their parties, than to the interest or perception of voters.

Because of the lack of analytical tools to simulate voting trends, we have developed a novel method for recognizing the political profiles of a population of voters. A key element in our method is to recognize the set of 'main attributes', characterizing the political behavior of a sector of voters.

We propose the use of surveys as an initial way to recognize the relation between concerns and the political preferences of the voters. We have employed the Web as a medium to promote and collect those surveys. We call *profile* to a set of principal attributes of the voters which infer a value of class (a political decision). Our proposal consists of applying learning algorithms for recognizing automatically the profiles characterizing the political preferences of the voters.

A candidate (and sometimes, his campaign's team) tries to recognize and modify the profiles of the voters in different ways, creating so different political programs to be applied in accordance with the answer of the voters to those programs.

2.1 The Web as a Medium to Collect Political Polls

At the time when the news came over the ticker on 2011, the so-called *Facebook revolution* had already reached a new high. According to Hosni Mubarak [1], who was still president of Egypt at that time, and according to his regime, it was no longer sufficient to sporadically shut off cellular networks and social media services, such as *Facebook*, or *Twitter*. Instead, the entire Internet seemed to become an uncontrollable threat to the governmental machinery of power which made it necessary to take this drastic step for the first time ever.

The phenomenon of disseminating information via social media services against the will of regimes of totalitarian states was not new. Subsequent to the presidential elections in Iran in 2009, for instance, news concerning the opponents protests almost exclusively reached the foreign public via the micro-blogging platform *Twitter* and the video website *Youtube* [1].

Just as the web has been used for connecting people, political purpose has occupied information and preferences of the electorate. For example, in the election of President Obama, his campaign team addressed microsites to meet the expectations of the voters about the candidates, as expressed in social networks.

Social network analysis based on algorithms for mining the Web, blogs, and online forums has been used to identify trends and find the people launching these new trends. Those algorithms include the temporal computation of network centrality measures, the visualization of social networks as Cybermaps, a semantic process of mining and analyzing large amounts of text based on social network analysis, and sentiment analysis and information filtering methods. The temporal calculation of betweeness of concepts allow to extract and predict long-term trends on the popularity of relevant concepts such as brands, movies, and politicians [6].

3 Data Mining

A key element for building significant political programs for a population, is to recognize profiles characterizing the main concerns of voters. But that is the central problem of the political actors: How to identify the necessities, the actual interest of the voters, and how to characterize their political preferences?

We propose the use of learning algorithms such as ID3 and C4.5 to identify the main attributes characterizing political preferences of a population of voters.

For example, the aim of the ID3 algorithm is to build a decision tree that explains each of the instances, in a very compact form [8]. A limitation of this algorithm is that instances should have only discrete numerical values associated with each attribute, and if the response matrix contains significant noise, the rules of classification could include a high rate of error. The C4.5 algorithm (also called J48) improves ID3 algorithm by handling continuous and discrete attributes. Both algorithms (ID3 and C4.5) have been implemented in the Weka system (Waikato Environment for Knowledge Analysis), which is a popular suite of machine learning software written in Java and developed at the University of Waikato, New Zealand [3].

A decision tree is built by the application of concepts from Information Theory to the set of responses of the survey (expressed in the learning matrix). In each step, the learning matrix is examined to determine the attribute expressing the most important information on which base a decision. This attribute is then used as a decision node to split the data set into as many groups as the values of such attribute. At each subsequent decision node, the data set is split repeatedly. The result is a decision tree, i.e. a collection of nodes, where the leaf nodes represent a final classification of the examples (see figure 1).

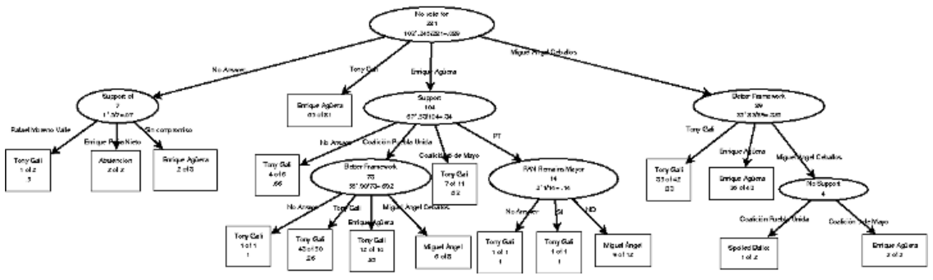

Fig. 1. Tree with relevant attributes for candidate: Antonio Gali

In order to determine the attribute which gives a maximum information gain, the entropy in the training matrix can be computed as:

$$Ent = -\Sigma_c(p(c) \cdot log_2 p(c)) \tag{1}$$

The entropy is an average of the amount of information necessary to classify an object, where $p(c)$ is the probability that such object corresponds with a specific value of class c. Also, the entropy of each attribute is computed based on the number of instances having the different values of the attribute. The information that each attribute A conveys is computed as:

$$Inf(A) = -\Sigma_v(p(v) \cdot (\Sigma_c p(\frac{c}{v}) \cdot log_2 p(\frac{c}{v}))) \tag{2}$$

where $p(\frac{c}{v})$ is the conditional probability to obtain the value of class c, given that the attribute A has the value v. And the average of the gain of an attribute A (the entropy of the attribute A), is computed as:

$$Ent(A) = -\Sigma_c(p(v) \cdot log_2 p(v)) \tag{3}$$

The information gain is computed for each attribute as:

$$Gain(A) = (Ent - Inf(A))/(Ent(A)) \tag{4}$$

and the attribute A with maximum information gain is chosen to be the splitting node on the current tree, in such a way that a branch from A is formed for each value in the domain of A, and then A is discarded of the set of attributes. The learning algorithm is then applied repeatedly to continue forming a complete decision tree.

We apply over 400 surveys and its intial pre-processing allow us to eliminate the positive fakes. We have codified the remaining surveys as source data for the Weka system, and we have built different decision trees formed by different options of class values, and also with a fix value of class, analyzing the correlation among different attributes.

We have used Weka since it supports several standard data mining tasks, such as: data preprocessing, visualization of the correlation among attributes, clustering, classification, regression, and feature selection [3].

After obtaining the decision trees, logical rules are formulated from those trees. A rule can be interpreted as the path from the root of the tree to a leaf node, indicating in each node along the path, the value of the attribute that the corresponding node has to take to be part of the rule.

The following set of decision trees show the relation between concerns of the voters with their political preference for each one of the two main candidates in the major election of Puebla, México, election realized in July 2013 (see figure 1 and 2).

We show only the trees characterizing the voters who are in favor of the two main candidates in the election under consideration. However, different trees were built considering for instance, the voters against each candidate.

4 Building a Hierarchy of the Main Attributes

There is an order among the attributes of a classification tree, given by the position of the attributes on the tree. This order corresponds to the minimum size of the path from the root to a class value (leaf node) of the tree. However, in certain applications is more useful to produce a linear order on the attributes according to the number of examples classified for each class value and the precision of such classification.

For example, in political campaigns, and with the purpose of characterizing the political preferences of the voters, is quite important to determine which attributes are more relevant than other. But this hierarchy must weight the attributes according to who attracts more voters and how that attribute appears in the inductive rules for each class value.

Now, we present an algorithm for sorting, in a linear and hierarchical form, the set of attributes appearing in a classification tree. In this case, the ordering is based on the number of instances classified for a class value, that we refer as 'positive class'.

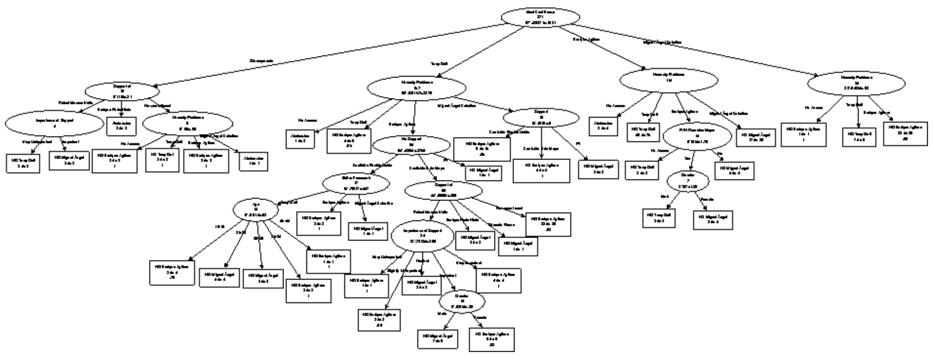

Fig. 2. Tree with relevant attributes for candidate: Enrique Agüera

Assuming that each leaf node of the tree has associated a pair of values: (p^+, $precs^+$), where p^+ is the ratio of the 'positive class', i.e. p^+ = $(num^+)/numTotal$, being num^+ the number of cases having the positive value of class and $numTotal$ the total number of cases in the node, that is: $numTotal = (num^+) + (num^-)$, where num^- is the number of cases classified with a class value different to 'positive class'. And the precision of that classification node is denoted by $precs^+ = \frac{(num^+)}{numTot^+}$, being $numTot^+$ the total number of rows on the learning matrix reaching the leaf node that satisfy the values of the attributes on the path, but they coincide or not with the 'positive class'.

Let us refer as $Order$ to the following procedure that builds a linear hierarchy on the attributes of a classification tree:

1. To each leaf node: Nh, its weight is calculated on the value of positive class, as: $PN^+ = p^+ \cdot precs^+$.
2. Calculate the weight of an internal node: NH_i where all of its children nodes are leaf nodes, as: $PN^+ = p^+ \cdot precs\_prom^+$, where $precs\_prom^+$ is the average of the precision over all positive leaf nodes (leaf nodes having 'positive class' value).
3. If the internal node N_i has as children nodes both leaf nodes and other internal nodes, its weight is computed as: $PN^+ = (p^+) \cdot (precs\_prom^+)$ where $precs\_prom^+$ = average$\{precs^+$: both positive leaf nodes and internal nodes with descendants having positive leaf nodes$\}$.

For each attribute At that appears as label of a node of the classification tree, its weight, denoted as PAt, is calculated as: $PAt = PN^+_{At}$ where PN^+_{At} is the weight calculated previously for the node labeled for the attribute At.

When a same attribute appears in multiple nodes of the classification tree, let say that the attribute At appears in the nodes $N_1, \ldots, N_r$ of the same tree, then $PAt = \Sigma_{i=1}^{r} PN^+_i$. And finally, the attributes are ordered decreasingly, according to the value PAt.

The previous algorithm computes the weights associated to the nodes of the tree in post-order, advancing from children to parent. Notice that the algorithm $Order$ builds a linear hierarchy among the attributes for any decision tree that has associated the values p^+ and $precs^+$ in each leaf node of the tree.

In our application, $Order$ builds a linear hierarchy among the main attributes in favor of each candidate. However, the recognition of profiles of voters who do not want to support a particular candidate is also important, because this will allow to determine improved campaign strategies. Therefore, we also build a linear hierarchy on the attributes expressing non support (antagonism) to each candidate. Both hierarchies allow us to measure the preferences and the oppositions for the candidates.

We show in Table 1 and Table 2, the results of applying $Order$ on the decision trees classifying as in favor and against each candidate of the considered contest. The list of attributes appearing in both tables resulting to be the most relevant attributes for determining favorable and unfavorable preferences of the voters.

Table 1 allows identifying that the main asset of the first candidate (Tony Gali) is that voters perceive that he counts with a very good campaign infrastructure. The second advantage is that voters are willing to support him, that is still above of the first asset of the other candidate, i.e. that people do not want to vote for the candidate of the party in office.

On the other hand, the main issue affecting the second candidate (Enrique Agüera) is the doubts of voters about his probity, as shown in Table 2. This disadvantage of Enrique Agüera is even quite upper than the obtained for other candidates. Also, the total of the unfavorable issues is overwhelming against the candidate.

Table 1. Attributes with Weights Favourable for Candidates

Preferred Candidate			
Tony Gali		Enrique Agüera	
Attributes	Weight	Attributes	Weight
Better Campaign Infrastructure	56.71	No Vote	39.23
Support	43.16	Better Campaign Infrastructure	16.22
No Vote	37.75	No Support	1
Party (PAN) Remains in Office	0.28	Support of	0.57
Support of	0.14		
Total	138.04	Total	57.02

5 Applying Game Theory

In economic as in political or social relations, it is common to find situations in which, as in games, the result depends on the conjunction of decisions made by different agents. In the theory of non-cooperative games, a zero sum game describes a situation in which the gain or loss of a player is balanced exactly with losses or gains of the other players [9]. This is called zero sum because if one adds the total earnings and subtracts the total losses of the participants, the final result is zero.

Table 2. Attributes with Weights Unfavorable for Candidates

Non-Preferred Candidate			
Tony Gali		Enrique Agüera	
Attributes	Weight	Attributes	Weight
Honesty Problems	28.01	Honesty Problems	62.81
Most Confidence	16.07	No Support	34.71
Party (PAN) Remains in Office	1.78	Most Confidence	34.71
Support of	1.56	Support of	30.89
Gender	1.28	Support	6.25
Importance of Support	1	Importance of Support	6
		Better Campaign Infrastructure	5.88
		Age	4.57
		Gender	1.78
Total	49.7	Total	187.6

When modeling democratic elections, it is known that a voter may choose only one candidate, which implies that, of the total population, a vote cast for a candidate must be subtracted from the number of votes possible to the other candidates. This is exactly how zero-sum games work. Thus, the distribution of votes among candidates is modeled as a zero sum game, where votes of a population is distributed among the candidates participating in the election, and in the case of our model, another participant is added to accumulate the null votes.

The distribution of votes among candidates is simulated (including the 'canceled votes'), based on the linear hierarchy of attributes obtained in the previous section, that characterize the profiles in favour and against each of the candidates. The vote distribution procedure is as follows:

1. Algorithm C4.5 is applied to build a new decision tree taking as class attribute: $Gender$, which is one of the most distinguishable features among voters. The attributes 'High confidence' and 'Would not vote for' are removed from the learning matrix, since they were attributes previously used as the class to build profiles of voters.
2. Algorithm $Order$ is applied on this latter decision tree to build a new linear hierarchy on the tree attributes. The idea is that this latter hierarchy characterizes in a general way voters, differentiating them only by gender: Male or Female, so we call it General Hierarchy.
3. The list of attributes of the General Hierarchy is intersected with the list of attributes of the hierarchy 'in favour' of candidates and also intersected with the list of attributes of the hierarchy 'against' candidates, to prune the attributes that are not held as relevant for a general characterization of the voters.
4. The relative proportions of the attributes that characterize 'in favour' are added, as well as the proportions of the attributes that characterize 'against':
   $Cand_1\_At^+ = \sum_{i=1}^{m} Attribute_i^+\_Cand_1$,
   $Cand_1\_At^- = \sum_{i=1}^{m} Attribute_i^-\_Cand_1$
   And this is done for each of the candidates.
5. To calculate the expected number of votes per candidate, the following formula is applied:
   $Votes\_Cand_1 = (Cand_1\_At^+) \cdot (\#Voters\_inFavour)$

$+ (Cand_1\_At^-) \cdot (\#VotersAgainst)$
doing the same for each candidate, and the resulting value represents the expected number of votes that the candidate will get in the election.

6. The expected relative percentages for each of the n candidates can be calculated as:

$$\%Cand_i = \frac{Votes\_Cand_i}{Total\_Votes},\ i = 1, \ldots, n$$

where $Total\_Votes = \sum_{i=1}^{n} Votes\_Cand_i$.
The resulting percentages are the proportions of the expected vote for each candidate.

5.1 Electoral Surveying

Based on the weights in Tables 1 and 2, and applying the previously presented procedure for vote distribution, the expected percentages of votes for the candidates are obtained and presented in Figure 3. This figure includes a third candidate in the contest (M.A. Cevallos) and the expected number of null votes.

Although in the basic statistics, both candidates have obtained a same number of favourable votes, the application of our model provided additional information that statistical methods can not assess. For example, the profile of the followers for one of the candidates is more homogeneous because he obtains a higher positive weight (preferred candidate) and a less negative weight (non-preferred candidate) than the values obtained for the other candidates. This last result was confirmed with the actual values obtained in the election, where the candidate with the more homogeneous followers won the contest.

Figure 4 includes the results taken from the official web page (in Spanish). This chart details the candidate coalitions or parties in x axis, and the number of votes in the y axis. The first lefmost bar is labelled for the party coalition that supported the candidate Tony Gali and the second bar for Enrique Agüera, also competing in coalition. Note also that the expected number of null votes predicted by the model coincided with the actual proportion (fourth bar), i.e. was higher than the candidate in third place.

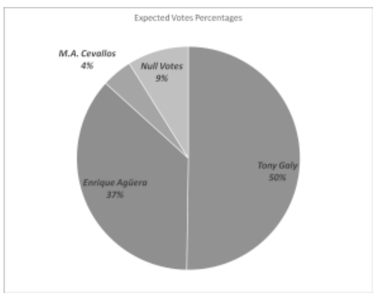

Fig. 3. Vote Distribution according to our Electoral Prospection Model

In elections held on July 4, 2013, Tony Gali won the election for 9.05 points ahead of his closest contender Enrique Agüera. While the results of our model: Decision Trees + Heuristics to build a hierarchy of attributes + Distribution of Votes, obtained a difference of 13.71 points in favour of Tony Gali, the difference between actual and our result is 4.66 percentage points, being acceptable for a computer model that predicted the expected results of the election, four days ahead of the election date.

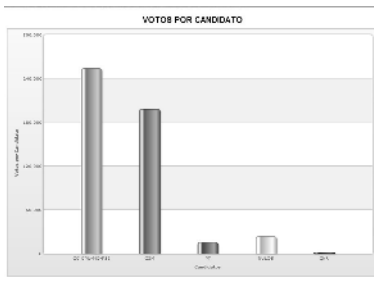

Fig. 4. Final Percentages of Votes, as Declared by the Electoral Institute of the State of Puebla

6 Conclusions

We have shown here that learning algorithms can be applied to determine the most relevant attributes characterizing the political preference of a population (called the profiles of the voters), in favour and against each of the contending candidates in a democratic election. The set of profiles is automatically generated after processing the opinion polls via learning algorithms.

We proposed a new method to obtain a linear hierarchy on the attributes that appear in a decision tree. The attributes hierarchy weights the number of instances which have chosen a specific class value, and whose logical rules (paths) passes through the attributes in question.

So, if one wishes to characterize political preferences of a population, it is important to determine which attributes are more relevant than other. But this hierarchy has to balance the attributes according to whether there are more (or less) voters who use that attribute to make its political decision.

In the election of our case study, the contest was won by the candidate with 9.05 percentage points above its nearest competitor. The difference between reality and our model was 4.66 percentage points, whereas traditional statistical methods had obtained a tie between the two leading candidates. The results of our model represent an acceptable outcome for a computer model that predicted the expected results of the election, four days prior to the election date.

Our approach has implication for political science, where it can support analysis of the main issues concerning the electorate, from the public opinion expressed in social media. We foresee that our methodology can also be applied to model product advertising campaigns, where from customer surveys, it can characterize the product of interest, in addition to contrast such product against other competing advertised goods.

References

1. Hans Ulrich Buhl, From Revolution to Participation: SocialMedia and the Democratic Decision-Making Process, BISE Editorial, 2011.
2. D. Borrajo, Inteligencia artificial: métodos y técnicas, Centro de Estudios Ramón Areces, 1997.
3. Bouckaert, R. and Frank, E. and Hall, M. and Kirkby, R. and Reutemann, P. and Seewald, A. and Scuse, D., Weka Manual for Version 3-6-6, University of Waikato, 2008.
4. Burstein, Frada and Holsapple, CW., Handbook on Decision Support Systems 1: Basic Themes, International Handbooks on Information Systems, Springer, 2008.
5. Camarero, L. and Almazán, A. and Arribas, J.M. and Mañas, B. and Vallejos, A.F., Estadística para la investigación social, Alfaomega Edit, 2013.
6. Gloor, P. A. and Krauss, J. and Nann, S. and Fischbach, K. and Schoder, D., Web Science 2.0: Identifying Trends through Semantic Social Network Analysis, ICCSE, 2009.
7. Quant, Marieke and Borm, Peter and Reijnierse, Hans, Congestion network problems and related games, European journal of operational research Elsevier, 2006.
8. Quinlan John Ross, C4. 5: programs for machine learning, Morgan Kaufmann, 1993.
9. Fernández Ruiz Jorge, Teoría de juegos. su aplicación en economía, El Colegio de México, Centro de Estudios Económicos, 2010.