How to spot a lie

For the polling industry, detecting lies has become a major issue. As massive data collection done by humans, human errors (accidentally or in purpose) are pretty common. Developing a system for detecting this errors and act accordingly is the subject of this article.

As students in the Universidad de Sonora Data Science graduate program and also working in the real world with Macrodata Analítica (MA) a research company based in Sonora, México we worked on analyzing actual polling data from 2018 state senate election.

Experience in the field shows us that a very important feature in detecting false polls is the time that takes a interviewer to do the interview. If a interviewer is doing the interview in a very short period of time, we can assume that the poll was done by the interviewer himself without interviewing the proper subject.

4 different data sets were provided by MA in order to detect outliers that will give us clues regarding which interviewers (identified by the Spanish word Encuestador) are probably not following the rules and making the most mistakes. The purpose of this work is to establish a future system that will help MA spot interviewers that are not doing their work properly.

The first step for spotting outliers is to have a tidy data set. We start by calling the proper Python packages.

[código con los paquetes]

We then merge the 4 datasets by calling the Pandas concat function.

[Código con concat]

Once we have the 4 datasets merged we start with a Exploratory Data Analysis in order to know the basics about our new merged dataset. We start by knowing the size of our data set using shape and explore the different columns our dataset has.

[Shape, index]

We now know that we have 5,823 data entries with 164 columns or variables. We also now decide which columns to use for the purpose of this project. We are going to use the first 15 columns in order to have a more compact data set and to use the necessary columns:

1. User
2. Poll duration
3. Interview acceptance
4. Stove in house.

In order to have only the variables of interest we drop the rest of the columns and start working with the mentioned above.

[Columnas de interés]

Moving on with our first task which is to have a tidy dataset, we now look for null values in our variables of interest. We do this by calling the isna function and by counting such data entries.

[busqueda de valores nulos]

We now have a workable dataset and can start looking for the unique interviewers. Interviewers have unique usernames that they use throughout the polling campaign. This will allow us to establish a pattern and the primary statistics for each of them. We can view the unique users by using the unique function.

[unique]

Part of working with data is always cleaning and much more when working with humans on the field. As part of this we detected that in the question regarding the use of a stove in the house (part of a 7 part questionnaire regarding the socio-economic household level) some of the answers were coded with an accent and some were coded without the accent, in order to solve this we used the replace function so we can have the same coded answer.

[Homologando respuesta]

One of the main reasons a poll could be done in a very short time is when the interviewer doesn’t accept such interview. That’s why as part of our analysis we are keeping this question for further analysis.

[Unique de entrevista]

As part of cleaning our data we now use label enconding in order to have a more simple way of identifying the users in a consecutive order. We use the preprocessing package from the sklearn library and use the LabelEncoder function.

[Label Encoding]

We use the same function for the interview acceptance question and stove using questions. We use 0 for the null values, 1 for doesn’t accept or doesn`t have stove and 2 for accepts interview and does have a stove.

[Label encoding para entrevista y estufa]

Now that we have our label encoding we can continue with our EDA. We use the describe function in order to have a first approach to the statistics of each column in our dataset.

[Describe]

Our first approach to the outlier detection and anomaly spotting will be with a scatterplot. Plotting our variable of interest, in this case the poll duration for each of the users.

[Primer scatterplot]

We can clearly see there’s a completely abnormal value for interviewer number “25”. We confirm this by using a boxplot graphic.

[primer boxplot]

As our first important decision in the outlier detection, a 5 order of magnitude value was detected and decided to eliminate it. Do to the nature of the data collection system, once you start a survey if you don’t complete it the timer keeps running, we can then assume that this outlier is a glitch in the data collection system.

We eliminate this data point by first detecting it and then getting rid of this specific row.

[Eliminación de entrada]

Once we got rid of this unusual outlier, we run a EDA again in order to get our new values and graph the dataset once again.

[Describe, Box plot y scatter plot sin outlier]

We can see 25 different data collectors with 5,022 different entries with the same 4 columns as before. A 600 seconds mean with a standard deviation of 436 seconds now gives us a more concise parametric statistics about how much time it takes for a user to get a full interview. The 1st quartile is 420 seconds and the 3rd quartile at around 673 seconds. This gives us our first look at real outliers both in the minimum and maximum amount of seconds for us to consider it a valid interview.

For us to have a better understanding of the amount of seconds for a minimum valid interview we are going to get rid of the interviews that did not accept to do the complete interview.

[eliminiación de no acepta]

The first technique used to analyze outliers is the K Nearest Neighbors. This technique uses a a machine learning algorithm that learns from certain distance neighbors and determines if a data entry is an outlier. Remember we are using the interview time in order to spot outliers.

The function used is NearestNeighbors from the sklean.neighbors package. We use 25 nearest points in order for the algorithm to learn.

[K N N]

We obtain the distance between 25 points and we graph this.

[Gráfica de KNN]

We now construct a new dataframe with the outliers with a distance between means of more than 15 seconds.

[matriz y data frame con datos de outliers]

Once we have identified the outliers we determine the number of them by calling the shape function on the new outlier dataframe, we now now we have 269 outliers from 5,022 data entries. We now graph the outliers with the whole dataset.

[Gráfica con outliers.]

We can now know the number of outliers per user. The top users with the most oultiers are 24, 20, 18 and 26 with more than 20.

[Gráfica con outliers por usuario]

But using just one method to spot outliers might not be the best idea. So in order to confirm the data we used isolation forrest method to have a “second opinion” on this regard. We used IsolationForrest from the sklearn.ensamble package.

[Llamado de paquete]

We first train our method by using 100 estimators with a 5% contamination. Based on this we should get a “score” for our seconds column. In order to identify the anomalies we use a -1 value for the ones we consider an anomaly.

[Isolation Forrest]

We store the anomaly values in a new data frame called anomalies. Lets look at this dataframe shape in order to get the total number of outliers.

[Nuevo dataframe de outliers]

We now use the same graph we used with the k neighbors in order to visualize the outliers and the rest of the original dataframe.

[Grafica con outliers y datos originales]

As part of the final analysis we now count the number of outliers per user. The top 4 users with the most number of outliers are 18, 24, 16 and 20. We can now conclude that interviewers 18, 24 and 20 are by both methods the top three with the most outliers and the proper adjusments must be made. Re training and close audit of this users is recommended and running both models is also recommended for a continuous improvement goal.

[Grafica de outliers por usuario con isolation Forrest]

Outliers in the polling industry are common. Only one user had no outliers. We must be on the lookout for the ones having the most of them so we can have the best results on the field.