Graph-theoretical Tools in Statistics

Mini-internship at PigeonLine

Report by Domenico Mergoni

Abstract

In this report, we study the possibility to apply tools and algorithms from Graph Theory and Optimisation Theory to the Statistical Analysis of data. In particular, we are interested in checking whether there might be a new graph-theoretical approach to dimensional analysis. In the main example of this report, the client is an airport but we hope our method can be easily generalised to other situations.

1 Introduction

We study the case of an airport; the company has access to data relative to approximately $N = 30.000$ clients, each of which replied to $k = 300$ circa questions. These questions include some categorical questions (the client has to decide between a small amount of options, which are not ordered in any way; an example could be the kind of work the person does), some open questions and some numerical questions (this can be similar to a categorical question, but the difference is that we have an order; we can take for example a rating between 1 and 5).
We want to explore the idea of examining this data using graph-theoretical results. In particular we are going to describe a graph of correlations and we are then going to suggest how to use this graph for dimensional reduction and analysis of customers preferences.
In the next Section, we are going to describe how to generate the graph of correlations. The difficulties of this step are generated from the fact that each question is different from another, and yet we need in some way to compare them; we are going to quickly examine how to go from the set of questions and answers to the graph of correlations, in which each question has the same role.
In the third Section, we are going to examine what can of information can be deduced from the graph of correlations.
In the last section, we try to generalise this method and apply it to a more wide set of situations. For example we are going to apply the aforementioned analysis to subcategories of the set of clients (depending on their level of satisfaction) and using this data we profile the differences between satisfied clients and clients with complaints.

2 From the data to the Correlation Graph

As mentioned in the introduction, the first step we need to make in our analysis is to create a Correlation Graph. In this section we clearly define the meaning of this graph and we give algorithms to determine it.

2.1 Graphs and weighted graphs, adjacency matrices

A *graph* is a pair $G = (V, E)$ that consists of a set V of vertices and a set E of edges; each edge is itself a set containing two vertices in V. Given two vertices x and y we often write that an edge xy is in G instead of writing $\{x, y\} \in E$. Given a set V of vertices, the *Complete Graph* over V is the graph that contains all the possible edges; i.e. is the only graph such that for any x and y in V we have $xy \in E$.
Any graph can be represented by its adjacency matrix as follows. Let $G = (V, E)$ be a graph; the adjacency matrix A representing G is the $\{0, 1\}$ symmetric matrix with columns and rows indexed by V such that for $x, y \in V$ we have $A_{xy} = 1$ if $xy \in E$, otherwise we have $xy = 1$.
A weighted graph G is a graph G with a weight function $w : E \to (0, 1]$ defined on its edges. It is easy to generalise the above definition of adjacency matrix also to weighted graphs.

2.2 The Correlation Graph

Given two real random variables X and Y, we can define the correlation coefficient (often indicated just by correlation) between them as $\rho_{XY} = \frac{|\mathrm{cov}(X,Y)|}{\sigma_X \sigma_Y}$ which is a number in $[0, 1]$ by Cauchy–Schwarz inequality. By its definition we can see that the correlation is a measure of the statistical relationship that there is between the two variables X and Y, normalised to be in the interval $[0, 1]$.
Let us now assume that we have a larger set $X_1, \dots, X_{300}$ of real random variables; it is of course possible to calculate the correlation between any two of them, it is therefore possible to define the symmetric matrix A such that for $i, j \in \{1, \dots, 300\}$ we have $A_{i,j} = \rho_{X_i, X_j}$.
By what we showed in the previous section, this matrix A describes uniquely a weighted (complete) graph; which we refer to as to the Correlation Graph of the random variables $X_1, \dots, X_{300}$. This graph has 300 vertices and as many edges as the pair of random variables with non-zero correlation.
It is important to keep in mind that the concept of weighted graph is more generic than the concept of correlation between real random variables; therefore it is possible to use this concept in a more general setting.

2.3 Correlation between categorical random variables

It is now time to give additional information regarding the specific project that is the main example of this report. In this case, the client is an airport that wants to analyse some data relative to its passengers; the data is as follows: About $N = 30.000$ passengers answered $k = 300$ questions; the questions include closed questions (categorical questions), open questions and numerical values. We assume there is a known way to translate the open questions in either numerical valuer or categorical answers (if the question for example is about what the passenger liked the most, we assume we already have an algorithm that understands the answer and is able to determine which part of the airport was the most liked; so we assume in this case that our answer is categorical). Therefore we assume that all questions are either numerical or categorical.

Remark. It is easy to assume that all variables are categorical variables. Indeed, if the question is: "How much did you spend in this airport?" the answers can be divided into categories (e.g. "0-20", "20-50", "50-100", "100+"); in this way, we do lose some information regarding the answers, but we do gain homogeneity among the random variables.

It would be interesting to make two algorithms, one that uses the actual values and the other one that uses this discretisation process; however, if one of the two options has to be chosen in advance, I believe that it would be better to consider all variables as categorical variables (indeed, we probably do not need all the information that is contained in the exact numerical value of a real random variables).

This last remark underlines the importance of having an algorithm to determine the correlation between two categorical random variables (and, if we proceed with the original data, between categorical and continuous random variables).

Algorithm 1. *Let X and Y be categorical random variables; let us assume X assumes values in $S_X = \{x_1, \dots, x_{\ell_X}\}$ and Y assumes values in $S_Y = \{y_1, \dots, y_{\ell_Y}\}$. Let us assume we have N random samples $q_1 = (a_1, b_1), \dots, q_N = (a_N, b_N) \in [\ell_X] \times [\ell_Y]$ of these random variables. This algorithm approximates the correlation between X and Y.*

Algorithm 1 Correlation of Categorical Variables

```
Require: $q_1, \dots, q_N; \ell_X; \ell_Y$
  $A \leftarrow zeros(\ell_X; \ell_Y)$
  $A_{+,\cdot} \leftarrow zeros(\ell_Y)$
  $A_{\cdot,+} \leftarrow zeros(\ell_X)$
  for $i = 1 : N$ do
      $A_{a_i,b_i} \leftarrow A_{a_i,b_i} + 1$                                   ▷ $O(N)$
      $A_{+,b_i} \leftarrow A_{+,b_i} + 1$
      $A_{a_i,+} \leftarrow A_{a_i,+} + 1$
  end for
  $F \leftarrow 0$
  for $i = 1 : \ell_X,\ j = 1 : \ell_Y$ do
      $F \leftarrow F + \frac{(A_{i,j} - A_{i,+}A_{+,j})^2}{A_{i,+}A_{+,j}}$                 ▷ $O(\ell_X + \ell_Y)$
  end for
  $T \leftarrow \sqrt{\frac{F}{\sqrt{(\ell_X - 1)(\ell_Y - 1)}}}$
```

We have that T is the wanted correlation between X and Y.

Remark. A nice aspect of this algorithm is that it can work in parallel; so we can divide the set of questions in 10 smaller sets and work independently on each and then work independently on each pair (if we have the possibility to use multiple calculators). Moreover, for any given pair (X, Y) the computational cost of this algorithm is minimal since it requires $O(N)$ time that is used uniquely to go through all the answers. Note that since we have $\frac{k^2-k}{2}$ pairs of answers this means that the total computational cost of creating the correlation graph is $O(k^2 N)$ assuming that all variables are categorical. Finally, it is quite important to take care of the memory in this algorithm, indeed, if we use new memory for each new pairs of variables (assuming that the average categorical variable has h options) we end up using $O(h^2 k^2)$ integers in the memory, which can be quite costly.

Finally, we have all the information we need to proceed if we decide to translate the real random variables in categorical variables (by dividing the range of possible values in a discrete number of subsets); if we want to use all the information contained into the real random variables, we can study the correlation between a real and a categorical random variable using the interclass correlation coefficient, but it seems that the versions already provided by the software R all assume that the real random variable is distributed following one of the most common random distributions, moreover, the cost of the calculations seem to increase quite rapidly with the number of options in the categorical random variable.

2.4 How to compute the Correlation Graph

Let us finally wrap-up all the hypothesis and write down the Algorithm we need to compute.

Algorithm 2. *Let $R \in \mathbb{N}^{k \times N}$ be the matrix of the answers of our passengers, such that for each question q and each passenger p we have $R_{q,p} \in [\ell_q]$. We assume that all the variables are categorical variables. We want to produce the matrix $G(R)$ that represents the correlation graph for these answers.*

Algorithm 2 Matrix associated to the correlation graph

```
Require: $R,\ \ell.$
  $G \leftarrow zeros(k;k)$
  for $i = 1:k,\ j = i:k$ do                                ▷ $O(k^2)$
      $G_{i,j} \leftarrow Algorithm\ 1((R_i, R_j), q_i, q_j)$
      $G_{j,i} \leftarrow G_{i,j}$                                    ▷ $O(N)$
  end for
```

Which is, we repeat for each pair of questions what we did for a pair of categorical variables.

2.5 Weighted vertices

This extra and final subsection gives us an additional tool to analyse our data. Some might have noticed that the adjacency matrix A has always weight 0 on the diagonal; we can use this extra space to store an additional information (entropy of each random variable). The entropy of a random variable tells us how much information the random variable gives us. The calculation is quite straightforward.

Algorithm 3. *Let R be vector with N entries that represents the answers to a question with ℓ options (so we have $R_i \in [\ell]$). Then we can calculate the entropy of R as follows.*

Algorithm 3 Entropy of a random variable

```
Require: $R,\ \ell$
  $V \leftarrow zeros(k)$
  for $i = 1:N$ do                                          ▷ $O(N)$
      $V_{R_i} \leftarrow V_{R_i} + 1$                                ▷ $O(N)$
  end for
  $V \leftarrow \frac{V}{N}$
  $F \leftarrow 0$
  for $i = 1:k$ do
      $F = F - V_i \log_2(V_i)$                                 ▷ $O(k)$
  end for
```

The entropy F is a measure of how much information is contained in the random variable.

3 Analysis of the Correlation Graph

We now operate in a somewhat simplified setting as follows. We assume that we have a matrix R of answers to a set of categorical questions, and we assume that we have an algorithm (Algorithm 2) that goes from R to the matrix A_R of the correlations between variables. Also, we are going to assume we know how to interpret the matrix A_R as the adjacency matrix of the weighted graph G_R (in which both the vertices and the edges are weighted).

3.1 Our method (the big picture)

We now have all the ingredients to start analysing the big picture. In the previous subsections we built the weighted graph G, now we explain how to interpret it and what are the next necessary step to extract information from it.

Remark. There are just a couple of notions that we have to always keep in mind.

- Each vertex of the graph represents a question and the set of all its answers. To each question is associated exactly one vertex (initially).
- The weight of the edge xy represents the correlation between the two vertices x and y. If the weight of xy is close to 1, it means that the questions x and y are strongly correlated; this means that knowing how a passenger answered x gives us a lot of (probabilistic) information regarding the answer that the same passenger gave to y. On the contrary, if the weight between x and y is close to 0, this means that knowing how a passenger answered x does not give us any information regarding the answer that the same passenger gave to y.
- Each vertex x has also a weight when considered by itself. The weight of a vertex represents its entropy. The entropy of x is close to 0 when most people answered the question x in the same way. The entropy of x is close to 1 when most of the possible values of x received approximately the same number of responses (remember that each question has a finite set of possible answers).

The above remark should be enough to clearly and correctly understand the situation; when we go from the matrix of answers to the correlation graph we lose a lot of information (for every question q, instead of the set of answers, we have the entropy of q and the correlation between q and all other questions) but on the other hand we have the possibility to deal with a simpler mathematical object.

A final preliminary consideration is as follows; in order to get G_R we just need the matrix R of the answers; but we can imagine R as to have two "parameter sets": the set of questions Q and the set of passengers P than answered. By this I mean that we have the possibility to consider only the answers of a particular group of passengers; for example, if we wanted to retrieve information on the experience of female passengers, we could select $P' \subseteq P$ the set of female passengers and consider the sub-matrix $R' \subseteq R$ which only has the columns corresponding to the passengers in P'. We can similarly consider a subset $Q' \subseteq Q$ of questions and only analyse those. Therefore, we can actually think G as G_R and R as $R_{Q,P}$ and hence the notation $G_{Q,P}$ makes sense. We are going to use this notation for example when we want to compare the graph of satisfied and not-satisfied clients.

Now we really do have all the elements to understand the analysis that we want to do.

3.1.1 Analysis of dimension

Assume for the moment that we have fixed a set of passengers P and a set of questions Q and that therefore we have a graph $G = G_{Q,P}$; the first analysis that we want to do is to understand which questions are relevant and which questions are not; in particular, we want an algorithm that selects a subset of "most relevant" questions. This is relevant for more than one reason; firstly, this allows for more specific questions: if someone wants to update the questions, this would be a good guide towards that; and this reduction also helps in future analysis, since we have a smaller set of questions to consider and therefore the analysis is going to be quicker. Finally, this analysis is useful when we try to partition the set of passengers: given a set of relevant questions (which is, a set of questions not correlated among themselves and with high variance), an interesting problem is to partition the set of passengers in such a way that when considering each set of passengers the questions have small variance and strong correlation (we are looking to partition the set of passengers into "types".

3.1.2 Comparison of graphs

Let P' and P'' be disjoint sets of passengers, then for some fixed set of questions Q we can compare the graphs $G_{Q,P'}$ and $G_{Q,P''}$; this allows us to analyse the difference between the two sets of people. We can get both a numerical value for how much these sets of people differ and an indication of what the similarities are and what the differences.

3.1.3 What do we need

In synthesis, we need an algorithm to reduce the dimension of a graph and an algorithm to compare two graphs.

3.2 Some important clarifications

Let us immediately start with some remarks about this setting.

Remark. Some of these remarks are about notation, in the others we point out the mathematical content that we are going to need in the following.

- The number of vertices of G_R (which is, the size of A_R) only depends on the number of rows of R, while the number of columns of R gives us an idea about how good our approximations are. When we do not care about the specific R, or whenever it is clear which R are we starting from, we just omit the R as index in A_R and G_R.
- The matrix A_R has entries in $[0, 1]$ and is symmetric.
- By a recent article by Langford, Schwertman, Owens (2001) we have that give X, Y and Z random variables, we have:

  $$\rho_{X,Y}\rho_{Y,Z} - \sqrt{(1-\rho^2_{X,Y})(1-\rho^2_{Y,Z})} \leq \rho_{X,Z} \leq \rho_{X,Y}\rho_{Y,Z} + \sqrt{(1-\rho^2_{X,Y})(1-\rho^2_{Y,Z})}.$$

  Which in our case means that for indices i, j, k we have:

  $$A_{i,j}A_{j,k} - \sqrt{(1-A^2_{i,j})(1-A^2_{j,k})} \leq A_{i,k} \leq A_{i,j}A_{j,k} + \sqrt{(1-A^2_{i,j})(1-A^2_{j,k})}.$$

An important consequence of the above remark is the following.

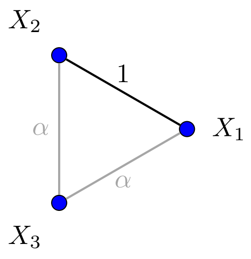

Figure 1: Partial transitivity of correlation.

Corollary 1. *Let X, Y and Z be random variables such that $\rho_{X,Y} = 1$; then we have that $\rho_{X,Z} = \rho_{Y,Z}$.*

In the above figure, we have that the two α values are actually the same; this means that we can reduce the graph G and still maintain the same information. We want to reduce the size of G since some useful algorithms that have graphs as inputs require exponential time in the number of vertices.
Even if it is not always possible to get two vertices with perfect correlation, there exists an approximated version of the above corollary, which still allows for good approximations.

Corollary 2. *Let X, Y and Z be random variables, let $\alpha = \rho_{X,Y} \geq 0.995$; then we have that $\rho_{X,Z} \in [\rho_{Y,Z} - 0.1, \rho_{Y,Z} + 0.1]$.*

We can obtain better approximations but not without imposing further conditions on $\rho_{Y,Z}$ as well.

3.3 Main algorithms

We now have all the instruments that we need to start working with our graph. In this subsection, we assume that we have already decided the graph G that we want to analyse. We refer to its adjacency matrix as to A.

3.3.1 Analysis of dimension

The idea behind the algorithm is as follows: we first reduce the edges of weight at least 0.995, we then partition the edges in 4 groups based on their weight, and we finally try to find a large set of vertices such that the edges amongst them have small weight.

Algorithm 4. *Let $A \in [0,1]^{k \times k}$ be the adjacency matrix of a graph with weighted edges and vertices. We find a large set of vertices spanning edges with small weight.*

At the end of this algorithm, we have three unweighted graphs B, C, D; they represent the random variables with low, average and high correlation respectively. Since we have the correlation data for every pair of random variables we have $B + C + D = ones(k, k)$.
In order to conclude the algorithm that provides us with a set of independent questions, we need to find a subset Q' of vertices such that many of the edges in Q' are in B. Ideally, we would be able to find a large clique in B, but there are more general algorithms that we can consider.
A very good reference for algorithms that allow us to find dense subgraphs of a graphs is the following: `https://people.cs.umass.edu/ barna/paper/dense-subgraph-full.pdf`.
In particular, we can use `Algorithm 2.1` and `Algorithm 3.1` of the text; the first

Algorithm 4 Analysis of dimension

```
Require: $A$
  %% Elimination of heavy edges
  for i=1:k, j=i+1:k do                                    ▷ $O(k^2)$
      if $A_{i,j} \geq 0.995$ then
          if $A_{i,i} \geq A_{j,j}$ then
              Remove row and column $j$
          else if $A_{i,i} < A_{j,j}$ then
              Remove row and column $i$
          end if
          Label the remaining index as $\{i, j\}$
      end if
  end for
  %% Partition of the edges
  $k \leftarrow$ size(A)
  $B, C, D \leftarrow zeros(k, k)$
  for i=1:k, j=i+1:k do                                    ▷ $O(k^2)$
      if $A_{i,j} \leq 0.3$ then
          $B_{i,j}, B_{j,i} \leftarrow 1$
      else if $0.3 < A_{i,j} \leq 0.6$ then
          $C_{i,j}, C_{j,i} \leftarrow 1$
      else if $0.6 < A_{i,j}$ then
          $D_{i,j}, D_{j,i} \leftarrow 1$
      end if
  end for
```

one needs polynomial time but only guarantees an approximation, while the second allows us to find a densest subgraph with at least a fixed number of vertices, but requires exponential time.
A suggestion would be to try the quickest algorithm first, consider whether the sub-graph obtained has an acceptable number of vertices and density; if this is not the case we can then try the second algorithm.
Last but not least, it could be interesting also to consider what happens if before applying the algorithm we removed all vertices with, say, weight lesser or equal than a certain constant. In this way, we would only be considering edges that give us more information. This notion can be implemented in the algorithms themselves when we need to decide which vertices to delete first (we would need to start from the vertices with smallest weight).

3.3.2 Comparison of graphs

We mentioned earlier that it can be useful to estimate how similar two graphs are among themselves, if for example we want to compare the correlation graph of two populations. We propose the following algorithm.

Algorithm 5. *Let $A, A' \in [0,1]^{k \times k}$ be the adjacency matrices of graphs with weighted edges and vertices. We find estimate the differences between A and A'.*

The algorithm outputs three values a, b, c that are between 0 and 1; a represents how different the graphs of light vertices are, while b and c do the same for the graphs of average or high weight.
If for example we have that c is close to 0, this would mean that there are a lot of questions that are highly correlated in both sets of answers.

Algorithm 5 Analysis of dimension

```
Require: $A, A', k$
  %% Partition of the edges
  $B, B', C, C', D, D' \leftarrow zeros(k, k)$
  for i=1:k, j=i+1:k do                                  ▷ $O(k^2)$
      if $A_{i,j} \leq 0.3$ then
          $B_{i,j}, B_{j,i} \leftarrow 1$
      else if $0.3 < A_{i,j} \leq 0.6$ then
          $C_{i,j}, C_{j,i} \leftarrow 1$
      else if $0.6 < A_{i,j}$ then
          $D_{i,j}, D_{j,i} \leftarrow 1$
      end if
  end for
  for i=1:k, j=i+1:k do                                  ▷ $O(k^2)$
      if $A'_{i,j} \leq 0.3$ then
          $B'_{i,j}, B'_{j,i} \leftarrow 1$
      else if $0.3 < A'_{i,j} \leq 0.6$ then
          $C'_{i,j}, C'_{j,i} \leftarrow 1$
      else if $0.6 < A'_{i,j}$ then
          $D'_{i,j}, D'_{j,i} \leftarrow 1$
      end if
  end for
  %% Comparison of each subgraph
  $b, c, d \leftarrow 0$
  for i=1:k, j=i+1:k do                                  ▷ $O(k^2)$
      $b = b + |B_{i,j} - B'_{i,j}|$
      $c = c + |C_{i,j} - C'_{i,j}|$
      $d = d + |D_{i,j} - D'_{i,j}|$
  end for
  $a = \frac{2 \cdot a}{k(k-1)}$
  $b = \frac{2 \cdot b}{k(k-1)}$
  $c = \frac{2 \cdot c}{k(k-1)}$
```

4 Further applications

We presented some of the possible applications of graph theory to the statistical analysis of some categorical random variables. This presentation is not meant to be exhaustive.

The concept of weighted graph with weighted vertices (being equivalent to a symmetric matrix with values between 0 and 1) seems to be a useful tool in answering statistical questions. The equivalence between the two concepts should not fool us: we are compelled to think about graphs in a different way than to matrices. Therefore we should expect to be able to utilise algorithm devised for graphs in a more interesting and direct way. We gave an example of this when we considered subgraphs with high density in the graph of edges with small weight: we utilised a purely graph-theoretical algorithm and we applied it to a problem of statistical nature.

Further improvements in this approach might bring other interesting results. In particular I believe it might be valuable to consider a more continuous approach to the weights of the edges; meaning that we could find a more sophisticated way to consider each single edge-weight independently instead of partitioning the set of edges in categories than we then consider uniformly.