MIC theory proof with its application

Dongsheng Wang

Occidental college, Los Angeles, California, USA
dwang3@oxy.edu

Abstract. Measuring dependencies between two variables in an extremely large data set is an increasingly important problem, naturally then the methods to solve such problems warrants equal if not greater attention. This paper aims to overview an effective measure of dependence, the MIC. This statistical measure is equitable giving no preference to certain function types. It is also general, being able to analyze both linear and nonlinear function types as well as combinations and superpositions of both. The key methodology such as the definitions and steps of MIC are explained as well as a proof of the central recursive algorithm which allows realistic runtimes for MIC. Other heuristic and approximations that make it both an accurate and efficient algorithm are also covered, namely the purpose and effect of equipartition and the clumping of the master partition. MICe, an approximation of MIC is also explained. This approximation fully utilizes the two heuristics of equipartition and clumping. This paper also briefly explains why these simplifications can still provide accurate results with a significantly faster runtime.

Keywords: Mutual information, Entropy, Grid-partition, Correlation coefficient, Maximal Information Coefficient

1 Introduction

Within this report I highlight the intertwined relationship between data science and artificial intelligence, where artificial intelligence uses the methodology of data science for machine learning and analysis and data science in turn uses AI as an efficient tool for data analysis. The significance of detecting of associations need also be established, where their importance primarily lies in detecting interesting and undiscovered relations which can be further researched into potential new discoveries or fields of studies. But realistically when given extremely large datasets with hundreds if not thousands of variables each with a potential relationship to another, the task of finding correlations and associations seams unfeasible. Yet problems like these are emerging even more in both number and importance as academic subjects become more interdisciplinary and intertwined. A potential solution may simply to go through each pair of combination and rank the level of association between pairs through some measure. This measure needs to be general, meaning it can capture a wide variety of relationships, both linear, nonlinear as well as combinations and superpositions of both. It also needs to be equitable in scoring these relationships so one functional relationship

Y. Wang (ed.), *Proceedings of the 2024 2nd International Conference on Image, Algorithms and Artificial Intelligence (ICIAAI 2024)*, Advances in Computer Science Research 115,
https://doi.org/10.2991/978-94-6463-540-9_38

isn't scored higher than another when under the same noise conditions. The importance of this lies in the fact that "entire classes of relationships could be missed, as scores for those relationships might be dominated by those of other classes of relationships" [7].

I aim to report an established and highly effective method of discovering such relationships, the maximal information coefficient or MIC. The MIC is an effective measure of relationships, it can accurately identify both functional and nonfunctional relationships and provides equitable scores for functional relationships [4].

I cover the basic methodology of the process which can be broken down into three simple steps. First applying various grids of x by y to the graphed pair points and finding the partition that provides the maximal mutual information between the x columns partition and y rows partition. Second, normalizing the values by dividing by the upper bound of the mutual information which is log_2(min(|X|,|Y|)) and storing them in the characteristic matrix. Lastly, the largest value of the matrix is found which is the MIC [4]. A more detailed description for the various definitions needed to understand the process is also given. The definition of mutual information in this context, the idea of normalizing and a proof to provide context as well as defining the characteristic matrix is also given.

I also review the proof of the recursive formula the algorithm uses to compute the maximal mutual information. This is one of the ways the algorithm reduces computational time as by using the recursive formula the problem can be broken down into subproblem. Another method to reduce computational time is also described where partitioning one axis equally will produce larger entropies and by extension larger mutual information reducing the number of partition configurations to check. Finally I go over MICe , a more effective approximation that uses two methods, equipartition of the larger axis and clumping of the master partition to provide an accurate approximation that can be run in reasonable time.

2 Literature review

The primary focus in data science is to collect, clean and analyze data for useful insights that could help make predictions on other data. Ai is similar in that it requires data to be processed but this data is used to train machine learning algorithms so that the AI can perform complex tasks, decisions and predictions.

AI can be seen as an extension of data science. Data science aims to find insights, relations and useful information from data from which a human decision is made [3]. AI further learns from the insights obtained through methods of data science to enhance decision making processes and optimize operations.

Both AI and Data science rely on a large amount of data [3]. In more general terms AI is better suited for mimicking simple human behavior that rely on learning from experience [3]. For deep insights that will affect strategic decision making, data science is better suited.

Detecting associations is significant in that discovering novel and unknown correlations could potentially lead to new discoveries or further investigation for causal relationships. Understanding relationships between variables is important when the goal is to optimize some outcome for example analyzing what variables will increase sales or what changes to the company will increase employee satisfaction, retention, and efficiency.

To further illustrate the importance of association detection and MIC, some concrete examples of applications are described. Analysis of factors connected to railway accidents is analyzed utilizing MIC [10]. Analysis of a cities flood susceptibility with respect to many different variables is analyzed with the case study of Zhengzhou city revealing that permeability, proportion of buildings and grasslands were the top 3 factors influencing flood susceptibility [11]. The Mic is also used as a method to identify differentially expressed genes and is shown to be as good as if not more effective than established methods [12].

MIC is better suited for detecting non linear relationships, whereas the Correlation Coefficient is better at detecting linear relationships than MIC. Both measures however suffer from increasing noise. When looking for unknown relationships, it is better to use MIC as it can detect a larger variety of more complex relationships than Correlation Coefficient.

3 Background

Before introducing the method of solving such problems, some background knowledge is needed. Entropy which is a core component of the concept of mutual information, is the measure of randomness of a variable, defined in equation 1 [5].

$$\mathrm{H(X)} = -\sum \mathrm{p(x)log\big(p(x)\big)} \tag{1}$$

Mutual information as the name implies measures how much knowing one variable will tell you about the other. When defined in terms of entropy, mutual information can be represented in 3 ways as shown in equation 2, 3 and 4 [5].

$$\mathrm{I(X;Y) = H(X) + H(Y) - H(X,Y)} \tag{2}$$

$$\mathrm{I(X;Y) = H(Y) - H(Y|X)} \tag{3}$$

$$\mathrm{I(X;Y) = H(X) - H(X|Y)} \tag{4}$$

Intuitively it can be understood that mutual information measures dependencies between two variables but looking at the randomness of both. If X and Y are related, by knowing exactly what X is one can reduce the entropy or randomness of Y as there is a functional relationship, whereas if X and Y are completely unrelated, by knowing X, the uncertainty or entropy in Y is not reduced at all. Essentially mutual information measures how much entropy or uncertainty is reduced in one variable by knowing the other.

4 MIC technical background

4.1 Definitions

A set of ordered pairs D can partition the graph via a grid separating the x values into X bins and y values into Y bins, this is known as a x by y grid [4].

With this grid D where all x and y are positive integers we can create two distributions, one from the bins of x and one from the bins of y where each bins probability is points in bin divided by total points [4].

Define $I^*(D, x, y)$ to be the mutual information of the x by y grid whose partition produces the largest possible mutual information between the x bin distribution and y bin distribution [4].

Now define the term $I^*(D, x, y)/ \log \min\{x, y\}$ in order to normalize the mutual information value. This division normalizes the values as different grid resolutions produce different maximum mutual information so in order to normalize them we divide each mutual information by the upper bound which is $\log \min\{x, y\}$ [4].

The Matrix containing all the normalized maximum mutual information values $I^*(D, x, y)/ \log \min\{x, y\}$ that have a grid size less than a predefined B(n) of sample size n is known as the characteristic matrix M(D). Where M xy (D) is the $I^*(D, x, y)/ \log \min\{x, y\}$ value of a x by y grid in the (x,y) cell of the matrix. The MIC score is the largest M xy (D) score in the entire matrix [4].

B(n) in its code implementation is controlled by the float variables alpha and n. if alpha is in (0,1] then B will be the maximum between n to the power of alpha and 4 where n is the number of samples. If alpha is greater than or equal to 4 then alpha defines directly the B parameter [2].

Further considering B(n), for any n points it is possible to make a n by n grid so that every point is in its own cell assuming no x values or y values are repeated. This implies maximal mutual information of 1 [1]. Although this may seem optimal it may result in many irrelevant or unnecessary relations being picked up or even noise being mistaken for relation. On the other hand, having minimal numbers of cells may result in complex and minute relationships to be overlooked [1]. The benefit however is faster runtime. To optimize between runtime and effective analysis, B(n) is shown to have optimal value when alpha is set to 0.6 [4].

4.2 Proof of log min{x, y} as upper bound

To prove that $\log \min\{x, y\}$ is the upper bound we start with:

$I(X;Y) = H(X) – H(X,Y) = H(Y)-H(X,Y)$, Therefore $I(X;Y) <= \min(H(X), H(Y))$

And $H(X) < \log(|X|)$ where $|X|$ is the cardinality or size of the alphabet. This is because entropy is maximal on a uniform distribution so $H(X) = - \Sigma p(x) * \log(p(x))$ Because its uniform all p(x) is the same and p(x) = 1/size of alphabet so $H(X) = - \log(p(x)) = -\log(1/\text{alphabet size}) = \log(\text{alphabet size})$ So entropy is upper bounded by the log of its alphabet size.

4.3 Basic methodology

The basic process of calculating the MIC between two variables X and Y can be split into 3 steps.

Step1:

The x,y points can be plot on a graph and for different grids of x by y find the partition such that the mutual information or I*(D, x, y) as defined above is maximized. The distribution of x is each the number of points in each x column partition divided by the total points, the same applies for the y distribution except it is rows.

To find the mutual information use equation 5 [5]:

$$I(X;Y) = \sum_{x\in\mathcal{X}} \sum_{y\epsilon\mathcal{Y}} p(x,y)\log\left(\frac{p(x,y)}{p(x)p(y)}\right. \quad (5)$$

Where p(x,y) is just the number of values in cell (x,y) divided by total number of points.

Step2:

The maximum values can be normalized by dividing each mutual information by its upper bound which is the log2(min(|X|,|Y|)) where |X| is the number of x columns for the partition and |Y| is the number of y rows of the partition, a minimum is used between the two as mutual information can be written in two ways, I(X;Y) = H(X) - H(X|Y) = H(Y) – H(Y|X) [5].

Step3:

The normalized values are added to a characteristic matrix to be stored where the largest value is the MIC. The storing method of the matrix is that the x,y cell of the matrix stores the maximum normalized mutual information of an x by y grid.

By intuition one might question why there is a need to store all values when all we need is the largest, naturally the grids of lower resolution will have lower values than grids of higher resolution so the largest MIC value will definitely come from grids of higher resolution. The reason for this is because the recursive method (which is what I review next) used in the algorithm to calculate the maximal mutual information requires the maximal mutual information of smaller grid resolutions, thus we are incentivized to keep track of all values to make calculating higher values easier.

5 Proof overview of recursive equation

5.1 Symbol definition

The Goal is to prove the recursive formula equation 6 shown below which is used to calculate the maximum mutual information of a grid [4].

$$F(m,l) = \max_{1\le i<m}\left\{\frac{i}{m}F(i,l-1) - \frac{m-i}{m}H(\langle i,m\rangle,Q)\right\} \quad (6)$$

In this equation F(m,l) is defined as max{H(P)-H(P,Q)} for all size ≤ l and for the first m points of D where $l > 1$ and $l < m \le n$. n represents the size of D , m is the first m points of D where D is a set of ordered pairs and l is the X axis partition size. P is the partition distribution of x and Q is the fixed partition distribution of y. Let P = ⟨0 = c0,...,cl = m⟩ be the x-axis partition maximizing H(P) − H(P,Q). Let #*,j denote the number of points in the j-th column of P and meaning that #*,j = cj − cj−1. An easy way to visualize this is to disregard the partitioning of the y axis and assume only the x axis has been partitioned, the number of points contained in the nth column of the x partition is denoted as #*,n. The same applies to the y partition except its #n,* . Define #i,j to be the number of points in the (i,j) cell of the partition grid [4].

5.2 Proof

The following heavily cites source [4].

$$\mathrm{F(m,l)} = \max\{\mathrm{H(P)} - \mathrm{H(P,Q)}\} \tag{7}$$

$$\mathrm{F(m,l)} = \sum_{j=1}^{l} \frac{\#_{*,j}}{m} \log\left(\frac{m}{\#_{*,j}}\right) - \sum_{j=1}^{l} \sum_{i=1}^{|Q|} \frac{\#_{i,j}}{m} \log\left(\frac{m}{\#_{i,j}}\right) \tag{8}$$

Equation 8 is obtained by using the formula for entropy given in equation 1 and joint entropy given in equation 9 on the definition of F(m,l) which is given in equation 7 [4]:

$$\mathrm{H(X,Y)} = -\sum_{y \epsilon Y} \sum_{x \epsilon X} \mathrm{p(x,y)logp(x,y)} \tag{9}$$

Where p(x) is $\frac{\#_{*,j}}{m}$ and p(x,y) is $\frac{\#_{i,j}}{m}$, the probability inside the log is flipped because the negative sign of the equation is moved inside [4].

$$\mathrm{F(m,l)} = \sum_{j=1}^{l} \sum_{i=1}^{|Q|} \frac{\#_{i,j}}{m} \log\left(\frac{\#_{i,j}}{\#_{*,j}}\right) \tag{10}$$

$$\mathrm{F(m,l)} = \sum_{j=1}^{l-1} \sum_{i=1}^{|Q|} \frac{\#_{i,j}}{m} \log\left(\frac{\#_{i,j}}{\#_{*,j}}\right) + \sum_{i=1}^{|Q|} \frac{\#_{i,l}}{m} \log\left(\frac{\#_{i,l}}{\#_{*,l}}\right) \tag{11}$$

Equation 10 can be obtained by from equation 8 by replacing $\frac{\#_{*,j}}{m}$ with $\frac{\#_{i,j}}{m}$ and then doing a summation over I, artificially adding the summation over I so it can merge the with other part of equation 8. When merging the fractions inside the log cancels the m out leaving equation 10 [4].

Equation 11 is obtained from equation 10 by separating out the final term of the summation from j=1 to l [4].

$$\mathrm{F(m,l)} = \frac{c_{l-1}}{m} \sum_{j=1}^{l-1} \sum_{i=1}^{|Q|} \frac{\#_{i,j}}{c_{l-1}} \log\left(\frac{\#_{i,j}}{\#_{*,j}}\right) + \frac{\#_{*,l}}{m} \sum_{i=1}^{|Q|} \frac{\#_{i,l}}{\#_{*,l}} \log\left(\frac{\#_{i,l}}{\#_{*,l}}\right) \tag{12}$$

$$\mathrm{F(m,l)} = \frac{c_{l-1}}{m}\left(\mathrm{H(P')} - \mathrm{H(P',Q)}\right) - \frac{\#_{*,l}}{m} \mathrm{H}(\langle c_{l-1}, m\rangle, \mathrm{Q}) \tag{13}$$

$$\mathrm{F(m,l)} = \frac{c_{l-1}}{m} (\mathrm{F}(c_{l-1}, l-1) - \frac{m - c_{l-1}}{m} \mathrm{H}(\langle c_{l-1}, m\rangle, \mathrm{Q}) \tag{14}$$

Equation 12 is simple algebra, adding in an extra term which cancels out.

For equation 13 we simply use the entropy formula but instead of a partition where P = ⟨0 = c0,...,cl = m⟩, we use P' = ⟨0 = c0,...,cl-1⟩ so just the previous partition excluding the last column. We apply the same logic to the end part but P is now only a single column or partition, the partition from cl-1 to m, ie the last column [4].

In Equation 14 we use the given formulas to tidy the proof up, the right hand side is from the above definition where the nth partition is Cn – Cn-1 but we swap out Cn for m as they are the same.

This provides us with a method of recursion to compute the maximal entropy faster, reducing the amount of calculations needed and speeding up runtime.

Another simple way to reduce the calculations to find the maximal entropy is to always partition the X and Y in a equal way, so that the number of points in each partition should be relatively the same. This is because from the formula of mutual information and the fact we fix one axis, by maximizing entropy we maximize the mutual information. Entropy is maximized when it is applied to a uniform distribution.

6 MICe

Although the MIC exhibits many extremely useful properties for a relation analysis tool such as generality and equitability its practicability is still limited by its computation speed. According to Shao and Liu "if the original approximate algorithm of MIC is directly applied into detecting bivariate correlations in high dimensional big data, the computation time is very long" [8].

The MICe , an approximate measure is designed to be faster than the standard method without losing much accuracy. Its effectiveness and importance is demonstrated by Albanese and Davide when used in conjunction with other MIC measures, "In particular, TICe is characterized by high power, which has been obtained at the cost of equitability, while MICe performs better on this side, showing reduced performances in terms of power. These two MIC-based measures compensate each other, and their combination is extremely promising as a data exploration tool" [9].

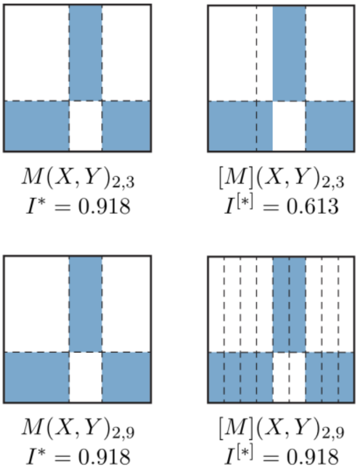
MICe is an alternative estimate to MIC_approx. MICe runs faster than MIC due to two primary changes. The first change is the values used in the matrix of MICe, called the equicharacteristic matrix is different. Previously only the grid size configuration that gave the maximum mutual information with a fixed grid dimension of l,k was used (divided by a normalizing value). For MICe we still use the mutual information however for the bigger dimension between l and k, that dimensions partition is a fixed equipartition. Essentially, we only need to iterate over the possible partition configurations of one axis instead of two as the larger one is a fixed equipartition. Although this approach may seem like the previously mentioned method of maximizing entropy through equipartition, the previous method required one axis to be a fixed equipartition while the other is iterated over, then the previously fixed axis is iterated over instead, and the other is equipartitioned. Essentially one axis is fixed the other isn't and once those calculations have been completed the same is done again but the axes are switched, with MICe only one iteration is needed rather than two. This new mutual information is again normalized and entered into the matrix [6].

The second way MICe improves runtime is by clumping. Although the original MIC_approx also utilizes clumping it is only for points within partitions whereas in MICe the clumping is for the master partition. The idea of a master partition can be thought of as the maximum possible resolution. For example, if I have n data points, for each axis the largest master partition is a partition of n as I can at most split them into n partitions each with one point. For the algorithm OPTIMIZEXAXIS which takes in a fixed partition of axis A, a master partition for axis B and a number K, the algorithm outputs the optimal partition for axis B given the fixed partition of A such that the partition of B is at most of k and k is a subset of the master partition. An easy way to understand the idea of a subset of a partition is to think of the partition as the maximum resolution. Every partition in your subset partition is made up of parts of the master partition. When the master partition is n there are no restrictions, however if my master partition was instead n/2 the smallest partition in my subset can contain at most n divided by n/2 points, (assuming the master partition is an equipartition since clumping is ultimately and emulation of equipartition). How this relates to clumping is that for MICe , in order to find the maximal mutual information we first equipartition and fix the larger axis, let's say its Y, next we need to find the optimal partition for the smaller axis X. If we set the set the fixed axis in OPTIMIZEXAXIS to be the larger equipartitioned axis, the algorithm outputs exactly what we want, an optimal partition of the smaller axis. However, realistically setting the master partition to n is unfeasible, this is where clumping comes in. Rather than having n as the master partition/maximum resolution, we "clump" the master partition so that it is c*k, where c is a constant and k is what we fixed the maximum partition size of the axis to be. Intuitively we are reducing the maximum resolution reducing lots of computations [6].

To summaries, the improvements of MICe is that it fully takes advantage of the benefits of equipartitioning to reduce the number of subproblems by half. It utilizes OPTIMIZEXAXIS to accurately calculate optimal partitions even with a less accurate master partition of c*k instead of n.

The reason why cutting corners in these areas doesn't affect accuracy is that in characteristic matrices, the boundary which is just the outer ring of values of the matrix is the only values that need to be accurate as the maximum MIC value will definitely be on grids of higher resolution which correspond to the boundary of the matrix.

The reason why cutting corners in these areas does not affect accuracy is due to the nature of characteristic matrices. In such matrices, the boundary, which is essentially the outer ring of values, is the only part that needs to be precise. This is because the maximum MIC value will be located on grids of highest resolution, which align with the boundary of the matrix. The method of equipartitioning complements this as the finer the equipartition the more accurate it is, as seen in the diagram below.



$M(X,Y)_{2,3}$
$I^* = 0.918$

$[M](X,Y)_{2,3}$
$I^{[*]} = 0.613$

$M(X,Y)_{2,9}$
$I^* = 0.918$

$[M](X,Y)_{2,9}$
$I^{[*]} = 0.918$

Fig 1. MIC and MICE comparison graphs

The right hand side of figure 1 [6], denotes equipartition and the left the normal partition, it is shown finer equipartitions can approximate and approach the actual value. The limit of how fine an equipartition is lies on the boundary of the matrix so we can still afford to be less accurate with non-boundary values.

7 Conclusion

Having established the importance of identifying correlations and relations, the MIC is a powerful tool that is able to overcome many barriers of previous methods while still maintaining effective runtime. Its equitability and ability to analyze most functions makes it an ideal candidate for solving the increasing number of big data problems as well as finding meaningful insights. Through an overview of the MIC it is also shown that it effectively utilizes heuristics such as equipartition and clumping to make approximations that are both accurate and feasible to run in the field. MICe is also explained as a faster and effective approximation making it an even more desirable candidate as a data analysis tool for the future.

Reference:

1. Lazarsfeld, John, Aaron Johnson, and Emmanuel Adeniran. "Differentially Private Maximal Information Coefficients." International Conference on Machine Learning. PMLR, 2022.
2. Albanese, Davide. “Python API¶.” Python API - Minepy 1.2.6 Documentation, minepy.readthedocs.io/en/latest/python.html. Accessed 13 July 2024.
3. “Difference between Data Science and Artificial Intelligence.” GeeksforGeeks, GeeksforGeeks, 24 June 2024, www.geeksforgeeks.org/difference-between-data-science-and-artificial-intelligence/.

4. Reshef, David N., et al. "Detecting novel associations in large data sets." science 334.6062 (2011): 1518-1524.
5. Yeung, Raymond W. Information theory and network coding. Springer Science & Business Media, 2008.
6. Reshef, Yakir A., et al. "Measuring dependence powerfully and equitably." Journal of Machine Learning Research 17.211 (2016): 1-63.
7. Reshef, David, et al. "Equitability analysis of the maximal information coefficient, with comparisons." arXiv preprint arXiv:1301.6314 (2013).
8. Shao Fubo, and Hui Liu. "The theoretical and experimental analysis of the maximal information coefficient approximate algorithm." Journal of Systems Science and Information 9.1 (2021): 95-104.
9. Albanese, Davide, et al. "A practical tool for maximal information coefficient analysis." GigaScience 7.4 (2018): giy032.
10. Shao, Fubo, Keping Li, and Xiaoming Xu. "Railway accidents analysis based on the improved algorithm of the maximal information coefficient." Intelligent Data Analysis 20.3 (2016): 597-613.
11. Wang, Hongfa, et al. "The application of integrating comprehensive evaluation and clustering algorithms weighted by maximal information coefficient for urban flood susceptibility." Journal of Environmental Management 344 (2023): 118846.
12. Liu, Han-Ming, et al. "Density distribution of gene expression profiles and evaluation of using maximal information coefficient to identify differentially expressed genes." PLoS one 14.7 (2019): e0219551.